OMAP™

Public Version

OMAP34xx Multimedia Device Silicon Revision 3.1.x

Texas Instruments OMAP™ Family of Products

Technical Reference Manual



Literature Number: SWPU223
July 2007–Revised February 2010

WARNING: EXPORT NOTICE

Recipient agrees to not knowingly export or re-export, directly or indirectly, any product or technical data (as defined by the U.S., EU, and other Export Administration Regulations) including software, or any controlled product restricted by other applicable national regulations, received from Disclosing party under this Agreement, or any direct product of such technology, to any destination to which such export or re-export is restricted or prohibited by U.S. or other applicable laws, without obtaining prior authorisation from U.S. Department of Commerce and other competent Government authorities to the extent required by those laws. This provision shall survive termination or expiration of this Agreement.

According to our best knowledge of the state and end-use of this product or technology, and in compliance with the export control regulations of dual-use goods in force in the origin and exporting countries, this technology is classified as follows:

US ECCN: 3E991

EU ECCN: EAR99

And may require export or re-export license for shipping it in compliance with the applicable regulations of certain countries.

Contents

Preface **163**

1 Introduction **175**
1.1 Overview 176
1.2 Environment 177
1.3 Description 179
1.3.1 MPU Subsystem 179
1.3.2 IVA2.2 Subsystem 180
1.3.3 On-Chip Memory 181
1.3.4 External Memory Interfaces 181
1.3.5 DMA Controllers 181
1.3.6 Multimedia 182
1.3.7 Comprehensive Power Management 183
1.3.8 Peripherals 183
1.4 Package-On-Package Concept 184
1.5 OMAP34xx Family 186
1.5.1 Device Features 186
1.5.2 Device Identification 188
1.5.3 General Recommendations Relative to Unavailable Features/Modules 189

2 Memory Mapping **191**
2.1 Introduction 192
2.2 Global Memory Space Mapping 194
2.3 L3 and L4 Memory Space Mapping 197
2.3.1 L3 Memory Space Mapping 197
2.3.2 L4 Memory Space Mapping 198
2.3.2.1 L4-Core Memory Space Mapping 198
2.3.2.2 L4-Wakeup Memory Space Mapping 201
2.3.2.3 L4-Peripheral Memory Space Mapping 202
2.3.2.4 L4-Emulation Memory Space Mapping 203
2.3.3 Register Access Restrictions 204
2.4 IVA2.2 Subsystem Memory Space Mapping 206
2.4.1 IVA2.2 Subsystem Internal Memory and Cache Allocation 206
2.4.1.1 IVA2.2 Subsystem Memory Hierarchy 206
2.4.1.2 IVA2.2 Cache Allocation 207
2.4.2 DSP Access to L2 Memories 208
2.4.2.1 DSP Access to L2 ROM 208
2.4.2.2 DSP Access to L2 RAM 208
2.4.3 DSP and EDMA Access to Memories and Peripherals 208
2.4.4 L3 Interconnect View of the IVA2.2 Subsystem Memory Space 209
2.4.5 DSP View of the IVA2.2 Subsystem Memory Space 210
2.4.6 EDMA View of the IVA2.2 Subsystem Memory Space 211

3 MPU Subsystem 213
3.1 MPU Subsystem Overview 214
3.1.1 Introduction 214
3.1.2 Features 215
3.2 MPU Subsystem Integration 216
3.2.1 MPU Subsystem Clock and Reset Distribution 218
3.2.1.1 Clock Distribution 218
3.2.1.2 Reset Distribution 219
3.2.2 ARM Subchip 220
3.2.2.1 ARM Overview 220
3.2.2.2 ARM Description 220
3.2.2.2.1 Public ARM Cortex-A8 Instruction, Data, and Private Peripheral Port 220
3.2.2.2.2 MPU Subsystem Features 220
3.2.2.3 Clock, Reset, and Power Management 222
3.2.2.3.1 Clocks 222
3.2.2.3.2 Reset 222
3.2.2.3.3 Power Management 222
3.2.3 AXI2OCP and I2Async Bridges 222
3.2.3.1 Bridges Overview 222
3.2.3.2 AXI2OCP Description 223
3.2.3.3 Clocks, Reset, and Power Management 224
3.2.3.3.1 Clocks 224
3.2.3.3.2 Reset 224
3.2.3.3.3 Power Management 224
3.2.4 Interrupt Controller 224
3.2.4.1 Clocks 224
3.2.4.2 Reset 224
3.2.4.3 Power Management 224
3.3 MPU Subsystem Functional Description 225
3.3.1 Interrupts 225
3.3.2 Power Management 225
3.3.2.1 Power Domains 225
3.3.2.2 Power States 226
3.3.2.3 Power Modes 226
3.3.2.4 Transitions 230
3.4 MPU Subsystem Basic Programming Model 231
3.4.1 Clock Control 231
3.4.2 MPU Power Mode Transitions 231
3.4.2.1 Basic Power-On Reset 231
3.4.2.2 MPU to Standby Mode 231
3.4.2.3 MPU Out of Standby Mode 231
3.4.2.4 MPU Power-On from a Powered-Off State 232
3.4.3 Neon Power Mode Transition 232
3.4.4 ARM Programming Model 232

4 Power Reset and Clock Management 233
4.1 Introduction to Power Management 234
4.1.1 Goal of Power Management 234
4.1.2 Power-Management Techniques 234

4.1.2.1 Dynamic Voltage and Frequency Scaling 234
4.1.2.2 Dynamic Power Switching 235
4.1.2.3 Standby Leakage Management 236
4.1.2.4 DPS Versus SLM 236
4.1.2.5 Combining Power-Management Techniques 236
4.1.3 Architectural Blocks for Power Management 237
4.1.3.1 Clock Domain 237
4.1.3.2 Power Domain 238
4.1.3.3 Voltage Domain 239
4.1.4 Device Power-Management Architecture 240
4.1.4.1 Module Interface and Functional Clocks 242
4.1.4.2 Autoidle Clock Control 242
4.2 PRCM Overview 244
4.2.1 Introduction 244
4.2.2 PRCM Features 246
4.3 PRCM Environment 247
4.3.1 External Clock Signals 248
4.3.2 External Reset Signals 249
4.3.3 External Power Signals 250
4.4 PRCM Integration 251
4.4.1 Power-Management Scheme, Reset, and Interrupt Requests 254
4.4.1.1 Power Domain 254
4.4.1.2 Resets 254
4.4.1.3 Interrupt Requests 255
4.5 PRCM Reset Manager Functional Description 256
4.5.1 Overview 256
4.5.2 General Characteristics of Reset Signals 256
4.5.2.1 Scope 257
4.5.2.2 Occurrence 257
4.5.2.3 Source Type 257
4.5.3 Reset Sources 258
4.5.3.1 Global Reset Sources 258
4.5.3.2 Local Reset Sources 259
4.5.4 Reset Distribution 260
4.5.5 Power Domain Reset Descriptions 261
4.5.5.1 MPU Power Domain 261
4.5.5.2 NEON Power Domain 262
4.5.5.3 IVA2 Power Domain 262
4.5.5.4 CORE Power Domain 262
4.5.5.5 DSS Power Domain 263
4.5.5.6 CAM Power Domain 263
4.5.5.7 USBHOST Power Domain 263
4.5.5.8 SGX Power Domain 263
4.5.5.9 WKUP Power Domain 263
4.5.5.10 PER Power Domain 264
4.5.5.11 DPLL Power Domains 264
4.5.5.12 EFUSE Power Domain 264
4.5.5.13 BANDGAP Logic 264
4.5.5.14 External Warm Reset Assertion 265

4.5.6 Reset Logging 265
4.5.6.1 PRCM Reset Logging Mechanism 265
4.5.6.2 SCM Reset Logging 265
4.5.7 Reset Management Overview 266
4.5.8 Reset Summary 271
4.5.9 Reset Sequences 274
4.5.9.1 Power-Up Sequence 274
4.5.9.2 Global Warm Reset Sequence 276
4.5.9.3 IVA2.2 Subsystem Power-Up Sequence 279
4.5.9.4 IVA2 Software Reset Sequence 282
4.5.9.5 IVA2 Global Warm Reset Sequence 284
4.5.9.6 IVA2 Power Domain Wake-Up Cold Reset Sequence 286
4.6 PRCM Power Manager Functional Description 289
4.6.1 Overview 289
4.6.1.1 Introduction 289
4.6.1.2 Device Partitioning 290
4.6.1.3 Memory and Logic Power Management 292
4.6.1.4 Power Domain States 292
4.6.1.5 Power State Transitions 293
4.6.1.6 Device Power Modes 293
4.6.1.7 Isolation Between Power Domains 294
4.6.2 Power Domain Implementation 294
4.6.2.1 Device Power Domains 294
4.6.2.2 Power Domain Memory Status 296
4.6.2.3 Power Domain State Transition Rules 296
4.6.2.4 Power Domain Dependencies 296
4.6.2.5 Power Domain Controls 296
4.6.2.5.1 Power Domain Hardware Control 297
4.6.2.5.2 Power Domain Software Controls 297
4.7 PRCM Clock Manager Functional Description 298
4.7.1 Overview 298
4.7.1.1 Interface and Functional Clocks 299
4.7.2 External Clock I/Os 299
4.7.2.1 External Clock Inputs 299
4.7.2.1.1 32-kHz Always-On Clock 299
4.7.2.1.2 High-Frequency System Clock 299
4.7.2.1.3 Alternate Clock 300
4.7.2.2 External Clock Outputs 300
4.7.2.3 Summary 300
4.7.3 Internal Clock Generation 301
4.7.3.1 PRM 303
4.7.3.2 CM 305
4.7.3.3 DPLLs 307
4.7.3.3.1 DPLL1 (MPU) and DPLL2 (IVA2) 308
4.7.3.3.2 DPLL3 (CORE) 309
4.7.3.3.3 DPLL4 (Peripherals) 310
4.7.3.3.4 DPLL5 (Peripherals) 311
4.7.3.3.5 DPLL Clock Summary 312

4.7.3.4 Summary 312
4.7.4 Clock Distribution 313
4.7.4.1 Power Domain Clock Distribution 313
4.7.4.1.1 MPU Power Domain 313
4.7.4.1.2 IVA2 Power Domain 313
4.7.4.1.3 SGX Power Domain 314
4.7.4.1.4 CORE Power Domain 315
4.7.4.1.5 EFUSE Power Domain 319
4.7.4.1.6 DSS Power Domain 319
4.7.4.1.7 CAM Power Domain 320
4.7.4.1.8 USBHOST Power Domain 322
4.7.4.1.9 WKUP Power Domain 323
4.7.4.1.10 PER Power Domain 324
4.7.4.1.11 DPLL Domains 325
4.7.4.2 Clock Distribution Summary 326
4.7.4.2.1 Power Domain Source Clocks 326
4.7.4.2.2 Peripheral Module Clocks 327
4.7.5 External Clock Controls 328
4.7.5.1 Clock Request (sys_clkreq) Control 328
4.7.5.2 System Clock Oscillator Control 329
4.7.5.3 External Output Clock1 (sys_clkout1) Control 332
4.7.5.4 External Output Clock2 (sys_clkout2) Control 332
4.7.6 DPLL Control 332
4.7.6.1 DPLL Multiplier and Divider Factors 332
4.7.6.2 DPLL Modes 333
4.7.6.3 DPLL Low-Power Mode 334
4.7.6.4 DPLL Clock Path Power Down 335
4.7.6.5 Recalibration 335
4.7.6.6 DPLL Programming Sequence 336
4.7.7 Internal Clock Controls 337
4.7.7.1 PRM Source-Clock Controls 338
4.7.7.2 CM Source-Clock Controls 340
4.7.7.3 Common Interface Clock Controls 341
4.7.7.4 DPLL Source-Clock Controls 342
4.7.7.5 SGX Power Domain Clock Controls 343
4.7.7.6 CORE Power Domain Clock Controls 344
4.7.7.7 EFUSE Power Domain Clock Controls 346
4.7.7.8 DSS Power Domain Clock Controls 346
4.7.7.9 CAM Power Domain Clock Controls 347
4.7.7.10 USBHOST Power Domain Clock Controls 348
4.7.7.11 WKUP Power Domain Clock Controls 349
4.7.7.12 PER Power Domain Clock Controls 350
4.7.8 Clock Configurations 352
4.7.8.1 Processor Clock Configurations 353
4.7.8.2 Interface and Peripheral Functional Clock Configurations 354
4.8 PRCM Idle and Wake-Up Management 356
4.8.1 Overview 356
4.8.2 Sleep Transition 358
4.8.3 Wakeup 358

4.8.4 Device Wake-Up Events 359
4.8.5 Sleep and Wake-Up Dependencies 364
4.8.5.1 Sleep Dependencies 364
4.8.5.2 Wake-Up Dependencies 367
4.8.6 USBHOST/USBTLL Save-and-Restore Management 369
4.8.6.1 USBHOST SAR Sequences 371
4.8.6.1.1 Save Sequence on Sleep Transition 371
4.8.6.1.2 Restore Sequence on Wake-Up Transition 371
4.8.6.2 USB TLL SAR Sequences 371
4.8.6.2.1 Save Sequence on Sleep Transition 371
4.8.6.2.2 Restore Sequence on Wake-Up Transition 371
4.9 PRCM Interrupts 372
4.10 PRCM Voltage Management Functional Description 373
4.10.1 Overview 374
4.10.2 Voltage Domains 376
4.10.3 Voltage Domain Dependencies 377
4.10.4 Voltage-Control Architecture 379
4.10.5 VDD1 and VDD2 Control 380
4.10.5.1 Direct Control With VMODE Signals 381
4.10.5.2 Direct Voltage Control With I^2C Interface 381
4.10.5.3 Voltage Controller and Dedicated I^2C Interface 382
4.10.6 Analog Cells, LDOs, and Level Shifter Controls 382
4.10.6.1 SRAM Voltage Control 382
4.10.6.2 Wake-Up and Emulation Voltage Control 382
4.11 PRCM Off-Mode Management 383
4.11.1 Overview 383
4.11.2 Device Off-Mode Configuration 383
4.11.2.1 Overview 383
4.11.2.2 I/O Wake-Up Mechanism 384
4.11.3 CORE Power Domain Off-Mode Sequences 385
4.11.3.1 Sleep Sequences (Transition From On to Retention/Off) 385
4.11.3.2 Wake-Up Sequences (Transition From Retention/Off to On) 386
4.11.4 Device Off-Mode Sequences 386
4.11.4.1 Sleep Sequences 386
4.11.4.1.1 Device Off-Mode Transition Without Using the SYS_OFF_MODE Signal 387
4.11.4.1.2 Device Off-Mode Transition Using Only the SYS_OFF_MODE Signal 387
4.11.4.2 Wake-Up Sequences 388
4.11.4.2.1 Device Wakeup from Off Mode Without Using the SYS_OFF_MODE Signal 388
4.11.4.2.2 Device Wakeup From Off Mode Using Only the SYS_OFF_MODE Signal 389
4.12 PRCM Basic Programming Model 389
4.12.1 Global Registers 389
4.12.1.1 Revision Information Registers 389
4.12.1.2 PRCM Configuration Registers 389
4.12.1.3 Interrupt Configuration Registers 389
4.12.1.3.1 MPU Interrupt Event Sources 390
4.12.1.3.2 MPU Interrupt Registers 390
4.12.1.3.3 IVA2.2 Interrupt Event Sources 391
4.12.1.3.4 IVA2 Interrupt Registers 391

4.12.1.4 Event Generator Control Registers 391
4.12.1.5 Output Signal Polarity Control Registers 391
4.12.1.5.1 CM_POLCTRL (CM Polarity Control Register) 392
4.12.1.5.2 PRM_POLCTRL (PRM Polarity Control Register) 394
4.12.1.6 SRAM Precharge Time Control Register 394
4.12.1.6.1 PRM_SRAM_PCHARGE (Voltage SRAM Precharge Counter Register) 394
4.12.2 Clock Management Registers 394
4.12.2.1 System Clock Control Registers 394
4.12.2.1.1 PRM_CLKSRC_CTRL (Clock Source Control Register) 394
4.12.2.1.2 PRM_CLKSETUP (Source-Clock Setup Register) 394
4.12.2.1.3 PRM_CLKSEL (Source-Clock Selection Register) 395
4.12.2.2 External Clock Output Control Registers 395
4.12.2.2.1 PRM_CLKOUT_CTRL (Clock Out Control Register) 395
4.12.2.2.2 CM_CLKOUT_CTRL (Clock Out Control Register) 395
4.12.2.3 DPLL Clock Control Registers 395
4.12.2.3.1 CM_CLKSELn_PLL_ <processor_name> (Processor DPLL Clock Selection Register) 395
4.12.2.3.2 CM_CLKSELn_PLL (DPLL Clock Selection Register) 396
4.12.2.3.3 CM_CLKEN_PLL_ <processor_name> (Processor DPLL Clock Enable Register) 396
4.12.2.3.4 CM_CLKEN_PLL (DPLL Enable Register) 396
4.12.2.3.5 CM_AUTOIDLE_PLL_ <processor_name> (Processor DPLL Autoidle Register) 397
4.12.2.3.6 CM_AUTOIDLE_PLL (DPLL Autoidle Register) 397
4.12.2.3.7 CM_AUTOIDLE1_PLL (DPLL5 Autoidle Register) 397
4.12.2.3.8 CM_IDLEST_CKGEN (Source-Clock Idle-Status Register) 397
4.12.2.3.9 CM_IDLEST2_CKGEN (DPLL5 Source-Clock Idle-Status Register) 398
4.12.2.3.10 CM_IDLEST_PLL_ <processor_name> (Processor DPLL Idle-Status Register) 398
4.12.2.4 Power-Domain Clock Control Registers 398
4.12.2.4.1 CM_CLKSEL_ <domain_name> (Clock Select Register) 398
4.12.2.4.2 CM_FCLKEN_ <domain_name> (Functional Clock Enable Register) 399
4.12.2.4.3 CM_ICLKEN_ <domain_name> (Interface Clock Enable Register) 399
4.12.2.4.4 CM_AUTOIDLE_ <domain_name> (Autoidle Register) 400
4.12.2.4.5 CM_IDLEST_ <domain_name> (Idle-Status Register) 400
4.12.2.4.6 CM_CLKSTCTRL_ <domain_name> (Clock State Control Register) 401
4.12.2.4.7 CM_CLKSTST_ <domain_name> (Clock State Status Register) 402
4.12.2.4.8 CM_SLEEPDEP_ <domain_name> (Sleep Dependency Control Register) 402
4.12.2.5 Domain Wake-Up Control Registers 403
4.12.2.5.1 PM_WKEN_ <domain_name> (Wake-Up Enable Register) 403
4.12.2.5.2 PM_WKST_ <domain_name> (Wake-Up Status Register) 404
4.12.2.5.3 PM_WKDEP_ <domain_name> (Wake–Up Dependency Register) 404
4.12.2.5.4 PM_ <processor_name> GRPSEL_ <domain_name> (Processor Group Selection Register) 405
4.12.3 Reset Management Registers 406
4.12.3.1 Reset Control 406
4.12.3.1.1 PRM_RSTTIME (Reset Time Register) 406
4.12.3.1.2 RM_RSTCTRL_ <domain_name> (Reset Control Register) 406
4.12.3.1.3 RM_RSTST_ <domain_name> (Reset Status Register) 407
4.12.4 Power Management Registers 408
4.12.4.1 PM_PWSTCTRL_ <domain_name> (Power State Control Register) 408
4.12.4.2 PM_PWSTST_ <domain_name> (Power State Status Register) 410
4.12.4.3 PM_PREPWSTST_ <domain_name> (Previous Power State Status Register) 411

4.12.5 Voltage Management Registers 411
4.12.5.1 External Voltage Control Register Descriptions 411
4.12.5.1.1 PRM_VOLTSETUP (Voltage Setup Time Register) 412
4.12.5.1.2 PRM_VOLTOFFSET (Voltage Offset Register) 413
4.12.5.1.3 PRM_VOLTCTRL (Voltage Source Control Register) 413
4.12.5.2 Voltage Controller Registers 414
4.12.5.2.1 PRM_VC_SMPS_SA (Voltage Controller SMPS Slave Address Register) 414
4.12.5.2.2 PRM_VC_SMPS_VOL_RA (Voltage Controller SMPS Voltage Register Address Register) 414
4.12.5.2.3 PRM_VC_SMPS_CMD_RA (Voltage Controller SMPS Command Register Address Register) 414
4.12.5.2.4 PRM_VC_CMD_VAL_0 and PRM_VC_CMD_VAL_1 (Voltage Controller Command and Voltage Value Register 0 and 1) 414
4.12.5.2.5 PRM_VC_CH_CONF (Voltage Controller Channel Configuration Register) 414
4.12.5.2.6 PRM_VC_I2C_CFG (Voltage Controller I^2C Interface Configuration Register) 415
4.12.5.2.7 PRM_VC_BYPASS_VAL (Voltage Controller Bypass Command Register) 415
4.12.6 Generic Programming Examples 415
4.12.6.1 Clock Control 415
4.12.6.1.1 Enabling and Disabling the Functional Clocks 415
4.12.6.1.2 Enabling and Disabling the Interface Clocks 417
4.12.6.1.3 Enabling and Disabling the Inactive State 418
4.12.6.1.4 Processor Clock Control 419
4.12.6.2 Reset Management 422
4.12.6.3 Wake-Up Control 422
4.12.6.4 Voltage Controller Initialization Basic Programming Model 424
4.12.6.5 Event Generator Programming Examples 426
4.13 PRCM Use Cases and Tips 427
4.13.1 Voltage Control Using VMODE 427
4.13.1.1 Introduction 427
4.13.1.2 Programming Sequence 427
4.13.1.2.1 Initialization Procedure 427
4.13.1.2.2 VMODE Signals Toggling 428
4.13.1.2.3 Summary Flow Chart 429
4.14 PRCM Register Manual 430
4.14.1 CM Module Registers 430
4.14.1.1 CM Module Summary 430
4.14.1.2 IVA2_CM Registers 430
4.14.1.3 OCP_System_Reg_CM Registers 437
4.14.1.4 MPU_CM Registers 438
4.14.1.5 CORE_CM Registers 444
4.14.1.6 SGX_CM Registers 458
4.14.1.7 WKUP_CM Registers 462
4.14.1.8 Clock_Control_Reg_CM Registers 467
4.14.1.9 DSS_CM Registers 480
4.14.1.10 CAM_CM Registers 486
4.14.1.11 PER_CM Registers 491
4.14.1.12 EMU_CM Registers 502
4.14.1.13 Global_Reg_CM Registers 507
4.14.1.14 NEON_CM Registers 508
4.14.1.15 USBHOST_CM Registers 509

4.14.2 PRM Module Registers ... 514
4.14.2.1 PRM Module Summary ... 514
4.14.2.2 IVA2_PRM Registers ... 515
4.14.2.3 OCP_System_Reg_PRM Registers ... 525
4.14.2.4 MPU_PRM Registers ... 530
4.14.2.5 CORE_PRM Registers ... 537
4.14.2.6 SGX_PRM Registers ... 553
4.14.2.7 WKUP_PRM Registers ... 557
4.14.2.8 Clock_Control_Reg_PRM Registers ... 562
4.14.2.9 DSS_PRM Registers ... 563
4.14.2.10 CAM_PRM Registers ... 568
4.14.2.11 PER_PRM Registers ... 572
4.14.2.12 EMU_PRM Registers ... 585
4.14.2.13 Global_Reg_PRM Registers ... 587
4.14.2.14 NEON_PRM Registers ... 601
4.14.2.15 USBHOST_PRM Registers ... 605

5 Interconnect ... 613

5.1 Interconnect Overview ... 614
5.1.1 Terminology ... 614
5.1.2 Architecture Overview ... 616
5.1.3 Module Distribution ... 618
5.1.3.1 L3 Interconnect Agents ... 618
5.1.3.2 L4-Core Agents ... 620
5.1.3.3 L4-Per Agents ... 620
5.1.3.4 L4-Emu Agents ... 621
5.1.3.5 L4-Wakeup Agents ... 622
5.1.4 Connectivity Matrix ... 622
5.2 L3 Interconnect ... 623
5.2.1 Overview ... 623
5.2.2 L3 Interconnect Integration ... 624
5.2.2.1 Clocking, Reset, and Power-Management Scheme ... 624
5.2.2.1.1 Clocks ... 624
5.2.2.1.2 Resets ... 624
5.2.2.1.3 Power Domain ... 624
5.2.2.1.4 Power Management ... 624
5.2.2.2 Hardware Requests ... 625
5.2.2.2.1 Interrupt Requests ... 625
5.2.3 L3 Interconnect Functional Description ... 625
5.2.3.1 Initiator Identification ... 625
5.2.3.2 Register Target ... 625
5.2.3.3 L3 Protection and Firewalls ... 626
5.2.3.3.1 Protection Region ... 627
5.2.3.3.2 Priority Level Overview ... 629
5.2.3.3.3 Read and Write Permission ... 631
5.2.3.3.4 REQ_INFO_PERMISSION Configuration ... 631
5.2.3.3.5 L3 Firewall Registers Overview ... 632
5.2.3.3.6 L3 Firewall Error-Logging Registers ... 633
5.2.3.3.7 L3 Firewall and System Control Module ... 633

5.2.3.4 Error Handling ... 635
5.2.3.4.1 Error Detection and Logging ... 635
5.2.3.4.2 Time-Out ... 637
5.2.3.4.3 Error Steering ... 638
5.2.3.4.4 Global Error Reporting ... 639
5.2.4 L3 Interconnect Basic Programming Model ... 643
5.2.4.1 General Recommendation ... 643
5.2.4.2 Initialization ... 643
5.2.4.3 Error Analysis ... 643
5.2.4.3.1 Time-Out Handling ... 645
5.2.4.3.2 Acknowledging Errors ... 646
5.2.4.4 Typical Example of Firewall Programming Example ... 646
5.2.5 L3 Interconnect Register Manual ... 650
5.2.5.1 L3 Initiator Agent (L3 IA) ... 650
5.2.5.1.1 L3 Initiator Agent (L3 IA) Registers Description ... 651
5.2.5.2 L3 Target Agent (L3 TA) ... 656
5.2.5.2.1 L3 Target Agent (L3 TA) Registers Description ... 658
5.2.5.3 Register Target (RT) ... 662
5.2.5.3.1 Register Target (RT) Registers Description ... 662
5.2.5.4 Protection Mechanism (PM) ... 664
5.2.5.4.1 Protection Mechanism (PM) Registers Description ... 666
5.2.5.5 Sideband Interconnect (SI) ... 674
5.2.5.5.1 Sideband Interconnect (SI) Registers Description ... 674
5.3 L4 Interconnects ... 677
5.3.1 Overview ... 677
5.3.1.1 L4-Core Interconnect ... 679
5.3.1.2 L4-Per Interconnect ... 679
5.3.1.3 L4-Emu Interconnect ... 680
5.3.1.4 L4-Wakeup Interconnect ... 681
5.3.2 L4 Interconnects Integration ... 682
5.3.2.1 Clocking, Reset, and Power-Management Scheme ... 682
5.3.2.1.1 Clocks ... 682
5.3.2.1.2 Resets ... 682
5.3.2.1.3 Power Domain ... 682
5.3.2.1.4 Power Management ... 683
5.3.3 L4 Interconnects Functional Description ... 683
5.3.3.1 L4-Interconnects Initiator Identification ... 683
5.3.3.2 Endianness Management ... 683
5.3.3.3 L4 Protection and Firewalls ... 683
5.3.3.3.1 Protection Mechanism ... 683
5.3.3.3.2 Protection Group ... 684
5.3.3.3.3 Segments and Regions ... 685
5.3.3.3.4 L4 Firewall Address and Protection Registers Setting ... 691
5.3.3.4 Error Handling ... 692
5.3.3.4.1 Overview ... 692
5.3.3.4.2 Error Logging ... 692
5.3.3.4.3 TA Software Reset ... 694
5.3.3.4.4 Error Reporting ... 694
5.3.4 L4 Interconnects Register Manual ... 695

5.3.4.1 L4 Iniator Agent (L4 IA) 697
5.3.4.1.1 L4 Iniator Agent (L4 IA) Registers Description 697
5.3.4.2 L4 Target Agent (L4 TA) 702
5.3.4.2.1 L4 Target Agent (L4 TA) Registers Description 709
5.3.4.3 L4 Link Register Agent (LA) 712
5.3.4.3.1 L4 Link Register Agent (LA) Registers Description 713
5.3.4.4 L4 Address Protection (AP) 717
5.3.4.4.1 L4 Address Protection (AP) Registers Description 718

6 Interprocessor Communication 729

6.1 IPC Overview 730
6.2 IPC Integration 731
6.2.1 Clocking, Reset, and Power-Management Scheme 731
6.2.1.1 Clocks 731
6.2.1.1.1 Module Clocks 731
6.2.1.2 Resets 731
6.2.1.2.1 Hardware Reset 731
6.2.1.2.2 Software Reset 731
6.2.1.3 Power Domains 732
6.2.1.4 Power Management 732
6.2.1.4.1 System Power Management 732
6.2.1.4.2 Module Power Management 732
6.2.2 Hardware Requests 733
6.2.2.1 Interrupt Requests 733
6.2.2.2 Idle Handshake Protocol 733
6.3 IPC Mailbox Functional Description 734
6.3.1 Block Diagram 734
6.3.2 Mailbox Assignment 735
6.3.2.1 Description 735
6.3.3 Sending and Receiving Messages 735
6.3.3.1 Description 735
6.3.4 16-Bit Register Access 736
6.3.4.1 Description 736
6.4 IPC Mailbox Basic Programming Model 737
6.4.1 Initialization Flow for the Mailbox Module 737
6.4.1.1 Software Reset 737
6.4.1.2 Idle Mode and Clock Configuration 737
6.4.2 Mailbox Assignment 737
6.4.3 Mailbox Communication Preparation 737
6.4.4 Mailbox Communication Sequence 738
6.4.5 Example of Communication 738
6.4.5.1 Sending a Message (Polling Method) 739
6.4.5.2 Sending a Message (Interrupt Method) 739
6.4.5.3 Receiving Messages (Interrupt Method) 740
6.5 IPC Mailbox Use Cases and Tips 742
6.5.1 Camcorder Use Case: How to Configure the Mailbox Module for Communication Between the MPU and the IVA2.2 Subsystems 742
6.5.1.1 Overview 742
6.5.1.2 Programming Flow 742

6.5.1.2.1 Initial Configuration 742
6.5.1.2.2 Operational Mode 744
6.6 IPC Mailbox Register Manual 748
6.6.1 Mailbox Register Mapping Summary 748
6.6.2 Register Description 748
7 System Control Module 755
7.1 SCM Overview 756
7.2 SCM Environment 756
7.2.1 Functional Interfaces 758
7.2.1.1 Basic SCM Pins 758
7.2.1.2 SCM Interface Description 758
7.3 SCM Integration 758
7.3.1 Clocking, Reset, and Power-Management Scheme 760
7.3.1.1 Clock 760
7.3.1.2 Resets 760
7.3.1.3 Power Domain 760
7.3.1.4 Power Management 761
7.3.1.4.1 System Power Management 761
7.3.1.4.2 Module Power Saving 761
7.3.2 Hardware Requests 762
7.4 SCM Functional Description 762
7.4.1 Block Diagram 762
7.4.2 SCM Initialization 764
7.4.3 Wake-up Control Module 764
7.4.4 Pad Functional Multiplexing and Configuration 764
7.4.4.1 Mode Selection 766
7.4.4.2 Pull Selection 767
7.4.4.3 Pad Multiplexing Register Fields 768
7.4.4.4 System Off Mode 779
7.4.4.4.1 Save-and-Restore Mechanism 779
7.4.4.4.2 Wake-Up Event Detection 780
7.4.5 Extended-Drain I/O Pin and PBIAS Cell 781
7.4.5.1 PBIAS Cells 783
7.4.5.2 Extended-Drain I/Os 784
7.4.6 Band Gap Voltage and Temperature Sensor 785
7.4.6.1 Band Gap Voltage Reference 787
7.4.6.2 Temperature Sensor 787
7.4.6.2.1 Single Conversion Mode (CONTCONV = 0) 787
7.4.6.2.2 Continuous Conversion Mode (CONTCONV = 1) 787
7.4.6.2.3 ADC Codes Versus Temperature 788
7.4.7 Functional Register Description 789
7.4.7.1 Static Device Configuration Registers 789
7.4.7.2 Control CSIRXFE Register 789
7.4.7.3 MPU and/or DSP (IVA2.2) MSuspend Configuration Registers 789
7.4.7.4 IVA2.2 Boot Registers 790
7.4.7.5 PBIAS LITE Control Register 790
7.4.7.6 Temperature Sensor Control Register 791
7.4.7.7 CSI Reciever Control Register 791

7.4.8 Protection Status Registers 791
7.4.9 SDRC Registers 792
7.4.10 Debug and Observability 792
7.4.10.1 Description 792
7.4.10.2 Observability Tables 794
7.4.11 Electomagnetic Interference Reduction for Clocking Generation (Spreading) 829
7.4.11.1 Overview 829
7.4.11.2 Integration 830
7.4.11.2.1 Clocking, Reset, and Power-Management Scheme 830
7.4.11.3 Functional Description 831
7.4.11.3.1 Spreading Generation Block 831
7.4.11.3.2 Spread Spectrum Clocking 832
7.4.11.3.3 Frequency Limitations 836
7.4.11.4 Basic Programming Model 837
7.4.11.4.1 SSC Configuration 837
7.5 SCM Programming Model 838
7.5.1 Feature Settings 838
7.5.1.1 Force Pad Configuration MuxMode by High-Speed USB 838
7.5.1.2 Video Driver 838
7.5.1.3 McBSP1 Internal Clock 838
7.5.1.4 McBSP2 Internal Clock 839
7.5.1.5 McBSP3 Internal Clock 839
7.5.1.6 McBSP4 Internal Clock 839
7.5.1.7 McBSP5 Internal Clock 839
7.5.1.8 MMC/SD/SDIO1 Module Input Clock Selection 839
7.5.1.9 MMC/SD/SDIO2 Module Input Clock Selection 839
7.5.1.10 Setting Sensitivity on sys_ndmareq[6:0] Input Pins 840
7.5.1.11 I^2C I/O Internal Pull-up Enable 840
7.5.1.12 SDRC I/O Drive Strength Selection 840
7.5.1.13 GPMC I/O Drive Strength Selection 841
7.5.1.14 MCBSP2 I/O Drive Strength Selection 843
7.5.1.15 MCSPI1 I/O Drive Strength Selection 843
7.5.1.16 Force MPU Writes to Be Nonposted 843
7.5.2 Extended-Drain I/Os and PBIAS Cells Programming Guide 843
7.5.2.1 PBIAS Error Generation 846
7.5.2.2 Critical Timing Requirements 846
7.5.2.3 Speed Control and Voltage Supply State 847
7.5.3 Off Mode Preliminary Settings 847
7.5.4 Pad Configuration Programming Points 847
7.5.5 I/O Power Optimization Guidelines 849
7.6 SCM Register Manual 851
7.6.1 SCM Instance Summary 851
7.6.2 SCM Register Summary 851
7.6.3 SCM Register Description 857
7.6.3.1 INTERFACE Register Description 857
7.6.3.2 PADCONFS Register Description 859
7.6.3.3 GENERAL Register Description 872
7.6.3.4 MEM_WKUP Register Description 924
7.6.3.5 PADCONFS_WKUP Register Description 924

7.6.3.6 GENERAL_WKUP Register Description 926

8 Memory Management Units 931

8.1 MMU Overview 932
8.2 MMU Integration 933
8.2.1 Clock Domains 933
8.2.2 Power Management 934
8.2.2.1 System Power Management 934
8.2.2.2 Module Power Saving 934
8.2.3 Reset 935
8.2.4 Interrupts 935
8.3 MMU Functional Description 937
8.3.1 MMU Benefits 937
8.3.2 MMU Architecture 938
8.3.2.1 MMU Address Translation Process 939
8.3.3 Translation Tables 940
8.3.3.1 Translation Table Hierarchy 940
8.3.3.2 First-Level Translation Table 941
8.3.3.2.1 First-Level Descriptor Format 942
8.3.3.2.2 First-Level Page Descriptor Format 942
8.3.3.2.3 First-Level Section Descriptor Format 943
8.3.3.2.4 Section Translation Summary 943
8.3.3.2.5 Supersection Translation Summary 943
8.3.3.3 Two-Level Translation 944
8.3.3.3.1 Second-Level Descriptor Format 945
8.3.3.3.2 Small Page Translation Summary 945
8.3.3.3.3 Large Page Translation Summary 946
8.3.4 Translation Lookaside Buffer 946
8.3.4.1 TLB Entry Format 947
8.3.5 MMU Error Handling 948
8.3.6 MMU Instance Design Parameters 948
8.4 MMU Basic Programming Model 950
8.4.1 Writing TLB Entries Statically 951
8.4.1.1 Protecting TLB Entries 951
8.4.1.2 Deleting TLB Entries 952
8.4.1.3 Reading TLB Entries 952
8.4.2 Programming the MMU Dynamically 952
8.4.2.1 Programming the MMU Using First- and Second-Level Translation Tables 953
8.5 MMU Register Manual 957
8.5.1 Register Mapping Summary 957
8.5.2 MMU Register Description 958

9 DMA 971

9.1 SDMA Module Overview 972
9.2 SDMA Controller Environment 974
9.2.1 Environment Overview 974
9.2.2 SDMA Request Scheme 974
9.3 SDMA Module Integration 975
9.3.1 External SDMA Request Interface Description 975
9.3.2 Clocking, Reset, and Power-Management Scheme 976

9.3.2.1 Clocking ... 976
9.3.2.2 Resets ... 977
9.3.2.2.1 Asynchronous Hardware Reset ... 977
9.3.2.2.2 Software Reset Through the Configuration Port ... 977
9.3.2.3 Power Domain ... 977
9.3.3 Hardware Requests ... 977
9.3.3.1 Interrupts to the MPU Subsystem ... 977
9.3.3.2 DMA Requests to the SDMA Controller ... 978
9.4 SDMA Functional Description ... 981
9.4.1 Logical Channel Transfer Overview ... 981
9.4.2 FIFO Queue Memory Pool ... 983
9.4.3 Addressing Modes ... 983
9.4.4 Packed Accesses ... 987
9.4.5 Burst Transactions ... 988
9.4.6 Endianism Conversion ... 988
9.4.7 Transfer Synchronization ... 988
9.4.7.1 Software Synchronization ... 988
9.4.7.2 Hardware Synchronization ... 989
9.4.8 Thread Budget Allocation ... 991
9.4.9 FIFO Budget Allocation ... 992
9.4.10 Chained Logical Channel Transfers ... 992
9.4.11 Reprogramming an Active Channel ... 993
9.4.12 Interrupt Generation ... 993
9.4.13 Packet Synchronization ... 993
9.4.14 Graphics Acceleration Support ... 994
9.4.15 Supervisor Modes ... 995
9.4.16 Posted and Nonposted Writes ... 995
9.4.17 Disabling a Channel During Transfer ... 995
9.4.18 FIFO Draining Mechanism ... 996
9.4.19 Reset ... 996
9.4.20 Power Management ... 996
9.4.20.1 Interconnect Clock Auto-Idle ... 996
9.4.20.2 Automatic Standby Mode ... 996
9.5 SDMA Basic Programming Model ... 997
9.5.1 Setup Configuration ... 997
9.5.2 Software-Triggered (Nonsynchronized) Transfer ... 997
9.5.3 Hardware-Synchronized Transfer ... 999
9.5.4 Synchronized Transfer Monitoring Using CDAC ... 1001
9.5.5 Concurrent Software and Hardware Synchronization ... 1001
9.5.6 Chained Transfer ... 1001
9.5.7 90° Clockwise Image Rotation ... 1002
9.5.8 Graphic Operations ... 1003
9.6 SDMA Use Cases and Tips ... 1003
9.6.1 Camcorder Use Case: How to Configure SDMA to Handle Transfers With McBSP2 and MMC to External DRAM ... 1003
9.6.1.1 Introduction ... 1003
9.6.1.2 SDMA Configuration to Transfer Data Between the McBSP and External DRAM ... 1003
9.6.1.2.1 Overview ... 1003
9.6.1.2.2 Environment ... 1004

9.6.1.2.3 Data Path ... 1004
9.6.1.2.4 Programming Flow ... 1005
9.6.1.3 SDMA Configuration to Transfer Data Between MMC and External DRAM ... 1007
9.6.1.3.1 Overview ... 1007
9.6.1.3.2 Programming Flow ... 1008
9.7 SDMA Registers Manual ... 1011
9.7.1 SDMA Instance Summary ... 1011
9.7.2 SDMA Register Summary ... 1011
9.7.3 SDMA Register Description ... 1012
10 Interrupt Controller ... 1037
10.1 Interrupt Controller Overview ... 1038
10.2 Interrupt Controller Environment ... 1040
10.3 MPU Subsystem INTCPS Integration ... 1041
10.3.1 Clocking, Reset, and Power Management Scheme ... 1041
10.3.1.1 MPU Subsystem INTC Clocks ... 1041
10.3.1.2 Hardware and Software Reset ... 1041
10.3.1.3 Power Management ... 1042
10.3.2 Interrupt Request Lines ... 1042
10.4 Interrupt Controller Functional Description ... 1045
10.4.1 Interrupt Processing ... 1046
10.4.1.1 Input Selection ... 1047
10.4.1.2 Masking ... 1047
10.4.1.2.1 Individual Masking ... 1047
10.4.1.2.2 Priority Masking ... 1047
10.4.1.3 Priority Sorting ... 1047
10.4.2 Register Protection ... 1048
10.4.3 Module Power Saving ... 1048
10.4.4 Interrupt Latency ... 1048
10.5 Interrupt Basic Programming Model ... 1049
10.5.1 Initialization Sequence ... 1049
10.5.2 MPU INTC Processing Sequence ... 1049
10.5.3 MPU INTC Preemptive Processing Sequence ... 1053
10.5.4 MPU INTC Spurious Interrupt Handling ... 1056
10.6 Interrupt Controller Register Manual ... 1057
10.6.1 Instance Summary ... 1057
10.6.2 Register Summary ... 1057
10.6.3 MPU INTC Register Descriptions ... 1058
10.6.4 Modem INTC Register Descriptions ... 1068
11 Memory Subsystem ... 1071
11.1 General-Purpose Memory Controller ... 1072
11.1.1 General-Purpose Memory Controller Overview ... 1072
11.1.1.1 GPMC Features ... 1073
11.1.2 GPMC Environment ... 1073
11.1.3 GPMC Integration ... 1077
11.1.3.1 Description ... 1077
11.1.3.2 Clocking, Reset, and Power-Management Scheme ... 1078
11.1.3.2.1 Clocking ... 1078
11.1.3.2.2 Hardware Reset ... 1078

11.1.3.2.3 Software Reset 1078
11.1.3.2.4 Power Domain, Power Saving, and Reset Management 1078
11.1.3.2.5 Hardware Requests 1079
11.1.3.3 GPMC Address and Data Bus 1079
11.1.3.3.1 GPMC I/O Configuration Setting (in Default Pinout Mode 0) 1079
11.1.3.3.2 GPMC CS0 Default Configuration at IC Reset 1080
11.1.4 GPMC Functional Description 1080
11.1.4.1 Description 1080
11.1.4.2 L3 Interconnect Interface 1081
11.1.4.3 Address Decoder, GPMC Configuration, and Chip-Select Configuration Register File 1082
11.1.4.4 Error Correction Code Engine 1082
11.1.4.5 Prefetch and Write-Posting Engine 1083
11.1.4.6 External Device/Memory Port Interface 1083
11.1.5 GPMC Basic Programming Model 1083
11.1.5.1 Chip-Select Base Address and Region Size Configuration 1083
11.1.5.2 Access Protocol Configuration 1085
11.1.5.2.1 Supported Devices 1085
11.1.5.2.2 Access Size Adaptation and Device Width 1085
11.1.5.2.3 Address/Data-Multiplexing Interface 1085
11.1.5.2.4 Address and Data Bus 1085
11.1.5.2.5 Asynchronous and Synchronous Access 1086
11.1.5.2.6 Page and Burst Support 1086
11.1.5.2.7 System Burst Versus External Device Burst Support 1086
11.1.5.3 Timing Setting 1087
11.1.5.3.1 Read Cycle Time and Write Cycle Time (RDCYCLETIME/WRCYCLETIME) 1088
11.1.5.3.2 nCS: Chip-Select Signal Control Assertion/Deassertion Time (CSONTIME/CSRDOFFTIME/CSWROFFTIME/CSEXTRADELAY) 1089
11.1.5.3.3 nADV/ALE: Address Valid/Address Latch Enable Signal Control Assertion/Deassertion Time (ADVONTIME/ADVRDOFFTIME/ADVWROFFTIME/ADVEXTRADELAY) 1089
11.1.5.3.4 nOE/nRE: Output Enable/Read Enable Signal Control Assertion/Deassertion Time (OEONTIME/OEOFFTIME/OEEXTRADELAY) 1090
11.1.5.3.5 nWE: Write Enable Signal Control Assertion/Deassertion Time (WEONTIME/WEOFFTIME/WEEXTRADELAY) 1090
11.1.5.3.6 GPMC_CLK 1091
11.1.5.3.7 GPMC_CLK and Control Signals Setup and Hold 1091
11.1.5.3.8 Access Time (RDACCESSTIME / WRACCESSTIME) 1091
11.1.5.3.9 Page Burst Access Time (PAGEBURSTACCESSTIME) 1092
11.1.5.3.10 Bus Keeping Support 1093
11.1.5.4 WAIT Pin Monitoring Control 1093
11.1.5.4.1 Wait Monitoring During an Asynchronous Read Access 1093
11.1.5.4.2 Wait Monitoring During an Asynchronous Write Access 1095
11.1.5.4.3 Wait Monitoring During a Synchronous Read Access 1095
11.1.5.4.4 Wait Monitoring During a Synchronous Write Access 1096
11.1.5.4.5 WAIT With NAND Device 1097
11.1.5.4.6 Idle Cycle Control Between Successive Accesses 1097
11.1.5.4.7 Slow Device Support (TIMEPARAGRANULARITY Parameter) 1099
11.1.5.5 gpmc_io_dir Pin 1099
11.1.5.6 Reset 1099
11.1.5.7 WRITE PROTECT (nWP) 1100
11.1.5.8 BYTE ENABLE (nBE1/nBE0) 1100

11.1.5.9 Asynchronous Access Description 1100
11.1.5.9.1 Asynchronous Single Read 1100
11.1.5.9.2 Asynchronous Single Write 1103
11.1.5.9.3 Asynchronous Multiple (Page Mode) Read 1105
11.1.5.10 Synchronous Access 1106
11.1.5.10.1 Synchronous Single Read 1107
11.1.5.10.2 Synchronous Single Write 1109
11.1.5.10.3 Synchronous Multiple (Burst) Read (4-, 8-, 16-Word16 Burst With Wraparound Capability) 1110
11.1.5.10.4 Synchronous Multiple (Burst) Write 1112
11.1.5.11 pSRAM Basic Programming Model 1114
11.1.5.12 Error Handling 1115
11.1.5.13 Boot Configuration 1115
11.1.5.14 NAND Device Basic Programming Model 1115
11.1.5.14.1 NAND Memory Device in Byte or Word16 Stream Mode 1115
11.1.5.14.2 NAND Device-Ready Pin 1121
11.1.5.14.3 ECC Calculator 1122
11.1.5.14.4 Prefetch and Write-Posting Engine 1138
11.1.6 GPMC Use Cases and Tips 1146
11.1.6.1 How to Set GPMC Timing Parameters for Typical Accesses 1146
11.1.6.1.1 External Memory Attached to the GPMC Module 1146
11.1.6.1.2 Typical GPMC Setup 1146
11.1.6.2 How to Choose a Suitable Memory to Use With the GPMC 1152
11.1.6.2.1 Supported Memories or Devices 1152
11.1.6.2.2 GPMC Features and Settings 1154
11.1.7 GPMC Register Manual 1155
11.1.7.1 GPMC Instance Summary 1155
11.1.7.2 GPMC Register Summary 1155
11.1.7.3 GPMC Register Description 1156
11.2 SDRAM Controller (SDRC) Subsystem 1183
11.2.1 SDRC Subsystem Overview 1183
11.2.1.1 Features 1184
11.2.2 SDRC Subsystem Environment 1185
11.2.2.1 SDRC Subsystem Description 1186
11.2.2.2 External Interface Configuration 1188
11.2.2.2.1 CS0, CS1 Memory Spaces 1188
11.2.2.2.2 AC Timing Control 1188
11.2.2.2.3 Address Multiplexing 1188
11.2.3 SDRC Subsystem Integration 1194
11.2.3.1 Clocking, Reset, and Power Management Scheme 1194
11.2.3.1.1 Clocking 1194
11.2.3.1.2 Hardware Reset 1195
11.2.3.1.3 Software Reset 1195
11.2.3.1.4 Power Management 1195
11.2.4 SDRC Subsystem Functional Description 1196
11.2.4.1 SDRAM Memory Scheduler 1196
11.2.4.1.1 Memory Access Scheduling 1198
11.2.4.1.2 Arbitration Policy 1198
11.2.4.1.3 Internal Class Arbitration 1200

11.2.4.1.4 Firewalls 1201
11.2.4.1.5 Rotation Engine 1204
11.2.4.1.6 Violation Reporting 1206
11.2.4.2 Module Power Saving 1206
11.2.4.3 System Power Management 1206
11.2.4.4 SDRC 1206
11.2.4.4.1 CS0-CS1 Memory Spaces 1207
11.2.4.4.2 Bank Tracking 1208
11.2.4.4.3 Address Multiplexing 1209
11.2.4.4.4 Bank Allocation Setting 1209
11.2.4.4.5 Data Multiplexing During Write Operations 1213
11.2.4.4.6 Data Demultiplexing During Read Operations 1214
11.2.4.4.7 Refresh Management 1216
11.2.4.4.8 System Power Management 1216
11.2.4.4.9 Power-Saving Features 1217
11.2.4.4.10 SDRC Power-Down Mode 1219
11.2.4.4.11 Controlled Delay Line 1220
11.2.4.5 Mode Registers 1223
11.2.4.5.1 Mode Register (MR) 1223
11.2.4.5.2 Extended Mode Register 2 (EMR2) 1223
11.2.5 SDRC Subsystem Basic Programming Model 1224
11.2.5.1 SMS Basic Programming Model 1224
11.2.5.1.1 SMS Firewall Usage 1224
11.2.5.1.2 VRFB Context Configuration 1224
11.2.5.1.3 Memory-Access Scheduler Configuration 1226
11.2.5.1.4 Error Logging 1226
11.2.5.2 SDRC Configuration 1227
11.2.5.2.1 IP Revision 1227
11.2.5.2.2 Reset Behavior 1227
11.2.5.3 SDRC Setup 1227
11.2.5.3.1 Chip-Select Configuration 1228
11.2.5.3.2 Memory Configuration 1228
11.2.5.3.3 SDRAM AC Timing Parameters 1228
11.2.5.3.4 DLL/CDL Configuration 1229
11.2.5.3.5 Mode Register Programming and Modes of Operation 1230
11.2.5.3.6 Autorefresh Management 1231
11.2.5.3.7 Page Closure Strategy 1232
11.2.5.4 Manual Software Commands 1232
11.2.5.4.1 Low-Power SDR/Mobile DDR Initialization Sequence 1234
11.2.5.4.2 Read/Write Access 1234
11.2.5.4.3 Memory Power Management 1235
11.2.5.5 Error Management 1237
11.2.6 SDRC Use Cases and Tips 1237
11.2.6.1 How to Program the VRFB 1237
11.2.6.1.1 VRFB Rotation Mechanism 1237
11.2.6.1.2 Setting a VRFB Context 1239
11.2.6.1.3 Applicative Use Case and Tips 1242
11.2.6.2 SMS Mode of Operation 1245
11.2.6.2.1 SDRAM Memory Scheduler and Arbitration Policy 1245

11.2.6.2.2 Arbitration Decision 1246
11.2.6.2.3 Arbitration Granularity 1248
11.2.6.2.4 How these Mechanisms Interact 1250
11.2.6.3 Typical SDRC connection to an External SDRAM Device 1254
11.2.6.3.1 External Memory Attached to the SDRC Module 1255
11.2.6.3.2 DDR-SDRAM Memory General Facts 1255
11.2.6.3.3 SDRC Typical Setup 1257
11.2.6.4 Camcorder Use Case: How to Configure the VRFB 1260
11.2.6.4.1 Overview 1261
11.2.6.4.2 Environment 1261
11.2.6.4.3 Data Path 1262
11.2.6.4.4 Programming Flow 1262
11.2.6.5 Understanding SDRAM Subsystem Address Spaces 1265
11.2.6.5.1 Physical vs Virtual Address Spaces 1265
11.2.6.5.2 CS Memory Spaces 1267
11.2.6.6 How to Choose a Suitable SDRAM 1269
11.2.6.6.1 SDRAM Device Parameters 1269
11.2.6.6.2 SDRC Controller Characteristics 1270
11.2.6.6.3 SDRAM Device Compatibility Verification 1270
11.2.7 SMS Register Manual 1272
11.2.7.1 SMS Instance Summary 1272
11.2.7.2 SMS Register Summary 1272
11.2.7.3 SMS Register Description 1272
11.2.8 SDRC Register Manual 1285
11.2.8.1 SDRC Instance Summary 1285
11.2.8.2 SDRC Register Summary 1285
11.2.8.3 SDRC Register Description 1285
11.3 On-Chip Memory Subsystem 1303
11.3.1 OCM Subsystem Overview 1303
11.3.2 OCM Subsystem Integration 1304
11.3.2.1 Description 1304
11.3.2.2 Clocking, Reset, and Power-Management Scheme 1304
11.3.2.2.1 Clocking 1304
11.3.2.2.2 Hardware Reset 1305
11.3.2.2.3 Power Domain 1305
11.3.3 OCM Subsystem Functional Description 1305
11.3.3.1 OCM_ROM 1305
11.3.3.2 OCM_RAM 1305

12 Camera Subsystem 1307

12.1 Camera ISP Overview 1308
12.1.1 Camera ISP Features 1310
12.2 Camera ISP Environment 1313
12.2.1 Camera ISP Functions 1313
12.2.2 Camera ISP Signal Descriptions 1313
12.2.3 Camera ISP Modes 1315
12.2.4 Camera ISP Protocols and Data Formats 1318
12.2.4.1 Parallel Generic Configuration Protocol and Data Format (8, 10, 11, 12 Bits) 1318
12.2.4.2 Parallel Generic Configuration: JPEG Sensor Connection on the Parallel Interface 1319

12.2.4.3 ITU-R BT.656 Protocol and Data Format (8, 10 Bits) 1319
12.2.4.4 Camera Serial Interface (CSI1) Protocol and Data Formats 1321
12.2.4.4.1 Pixel Data Format 1325
12.2.4.5 Camera Serial Interface (CSI2) Protocol and Data Format 1333
12.2.4.5.1 Physical Layer 1333
12.2.4.5.2 Lane Merger 1334
12.2.4.5.3 Protocol Layer 1335
12.2.4.5.4 CSI2 Operating Modes 1341
12.3 Camera ISP Integration 1359
12.3.1 Clocking, Reset, and Power-Management Scheme 1359
12.3.1.1 Clocks 1359
12.3.1.1.1 Clock Tree 1360
12.3.1.1.2 Clock Descriptions 1360
12.3.1.1.3 Clock Configuration 1360
12.3.1.2 Power Management 1362
12.3.1.2.1 Local Power Management 1362
12.3.1.2.2 System Power Management 1362
12.3.1.3 Power Domain 1363
12.3.1.4 Resets 1363
12.3.1.4.1 Hardware Reset 1363
12.3.1.4.2 Software Reset 1363
12.3.2 Hardware Requests 1364
12.3.2.1 Interrupt Requests 1364
12.4 Camera ISP Functional Description 1370
12.4.1 Block Diagram 1370
12.4.1.1 Possible Data Paths Inside the Camera ISP 1372
12.4.1.1.1 RGB RAW Data 1373
12.4.1.1.2 YUV4:2:2 Data 1374
12.4.1.1.3 JPEG Data 1374
12.4.2 CSI1 Receiver 1374
12.4.2.1 CSI1 Receiver Features 1375
12.4.2.2 CSI1 Receiver Functional Description 1375
12.4.2.2.1 Overview 1375
12.4.2.2.2 Complex I/O 1375
12.4.2.2.3 Physical Layer 1376
12.4.2.2.4 Protocol Layer 1376
12.4.2.2.5 Memory Read Channel 1379
12.4.3 CSI2 Receiver 1381
12.4.3.1 CSI2 Receiver Features 1381
12.4.3.2 CSI2 Receiver Block Diagram 1381
12.4.3.3 Physical Layer Lane Configuration 1381
12.4.3.4 ECC and Checksum Generation 1382
12.4.3.4.1 ECC 1382
12.4.3.4.2 Checksum 1382
12.4.3.5 Short Packet 1383
12.4.3.6 Virtual Channel and Context 1383
12.4.3.7 DMA Engine 1384
12.4.3.7.1 Progressive Frame to Progressive Storage 1385
12.4.3.7.2 Interlaced Frame to Progressive Storage 1385

12.4.3.8 Complex I/O 1386
12.4.4 Timing Control 1388
12.4.4.1 Timing-Control Features 1388
12.4.4.2 Timing Control 1388
12.4.4.2.1 Timing Generator 1388
12.4.4.2.2 Control-Signal Generator 1389
12.4.5 Bridge-Lane Shifter 1392
12.4.6 Video-Processing Front End 1392
12.4.6.1 CCDC 1392
12.4.6.1.1 CCDC Features 1392
12.4.6.1.2 CCDC Block Diagram 1394
12.4.6.1.3 CCDC Functional Operations 1395
12.4.6.1.4 DMA 1406
12.4.6.1.5 Memories 1406
12.4.7 Video-Processing Back End 1406
12.4.7.1 Preview Engine Features 1406
12.4.7.1.1 Preview Block Diagram 1407
12.4.7.1.2 Input Interface 1407
12.4.7.1.3 Input Formatter/Averager 1408
12.4.7.1.4 Dark-Frame Write 1408
12.4.7.1.5 Inverse A-Law 1409
12.4.7.1.6 Dark-Frame Subtract or Shading Compensation 1409
12.4.7.1.7 Horizontal Median Filter 1409
12.4.7.1.8 Noise Filter and Faulty Pixel Correction 1409
12.4.7.1.9 White Balance 1410
12.4.7.1.10 CFA Interpolation 1410
12.4.7.1.11 Black Adjustment 1410
12.4.7.1.12 RGB Blending 1411
12.4.7.1.13 Gamma Correction 1411
12.4.7.1.14 RGB to YCbCr Conversion, Luminance Enhancement, Chrominance Suppression, Contrast and Brightness, and 4:2:2 Downsampling and Output Clipping 1411
12.4.7.1.15 Write-Buffer Interface 1412
12.4.7.2 Resizer 1413
12.4.7.2.1 Features 1413
12.4.7.2.2 Block Diagram 1413
12.4.7.2.3 Input and Output Interfaces 1414
12.4.7.2.4 Horizontal and Vertical Resizing 1415
12.4.7.2.5 Resampling Algorithm 1419
12.4.7.2.6 Luma Edge Enhancement 1424
12.4.8 Statistics Collection Modules 1424
12.4.8.1 Statistics Collection: H3A 1424
12.4.8.1.1 Features 1424
12.4.8.1.2 Autofocus Engine 1425
12.4.8.1.3 AE/AWB Engine 1425
12.4.8.2 Statistics Collection: Histogram 1425
12.4.8.2.1 Features 1425
12.4.8.2.2 Block Diagram 1426
12.4.8.2.3 Input Interface 1426
12.4.8.2.4 White Balance 1426

12.4.8.2.5 Histogram Binning 1427
12.4.9 Central-Resource Shared Buffer Logic 1428
12.4.9.1 Block Diagram 1429
12.4.9.2 Functional Operations 1430
12.4.9.2.1 Parameters 1430
12.4.9.2.2 Write-Buffer Logic (WBL) and Write Buffer 1430
12.4.9.2.3 Read Buffer Logic (RBL) and Read Buffer 1430
12.4.9.2.4 Arbitration 1431
12.4.9.3 Memories 1431
12.4.9.4 Debug Registers 1431
12.4.10 Circular Buffer 1432
12.4.10.1 Feature List 1432
12.4.10.2 Interrupts 1433
12.4.10.3 Functional Description 1433
12.4.10.3.1 Bandwidth Control Feedback Loop 1433
12.4.10.3.2 Window Management 1437
12.4.10.3.3 CPU Interaction 1441
12.4.11 MMU Logic 1441
12.4.11.1 MMU Features 1441
12.4.11.2 MMU Functional Description 1441
12.5 Camera ISP Basic Programming Model 1441
12.5.1 Programming the CSI1 Receiver 1441
12.5.1.1 Hardware Setup/Initialization 1441
12.5.1.1.1 Reset Behavior 1441
12.5.1.2 Event and Status Checking 1442
12.5.1.3 Register Accessibility During Frame Processing 1442
12.5.1.4 Enable/Disable the Hardware 1442
12.5.1.5 Complex I/O Control 1443
12.5.1.6 Debug Mode 1443
12.5.1.7 Video Port 1443
12.5.1.8 Region of Interest 1444
12.5.1.9 Destination Format 1444
12.5.1.10 Frame Acquisition 1445
12.5.1.11 Synchronization Codes 1445
12.5.1.12 Status Data 1445
12.5.1.13 Pixel Data Region 1446
12.5.1.14 Memory Read Channel 1448
12.5.1.14.1 Write Data From Sensor to Memory 1448
12.5.2 Programming the CSI2 Receiver 1448
12.5.2.1 Enable CSI2 Interface 1449
12.5.2.2 Reset Management 1449
12.5.2.3 Enable Video/Picture Acquisition 1449
12.5.2.4 Disable Video/Picture Acquisition 1450
12.5.2.5 Capture a Finite Number of Frames 1451
12.5.2.6 Configure a Periodic Event During Frame Acquisition 1451
12.5.2.7 Linking a Context to a Virtual Channel and a Data Type 1451
12.5.2.8 Progressive and Interleaved Frame Configuration 1453
12.5.3 Programming the Timing CTRL Module 1453

12.5.3.1 Timing Generator 1453
12.5.3.2 Camera-Control Signal Generator 1454
12.5.3.2.1 Vertical Synchro-Based Control-Signal Generation or Externally-Generated cam_global_reset 1454
12.5.3.2.2 Internally-Generated cam_global_reset-Based Control-Signal Generation 1455
12.5.3.2.3 STROBE and PRESTROBE Signal Generation for Red-Eye Removal 1456
12.5.4 Programming the CCDC 1456
12.5.4.1 CCDC Hardware Setup/Initialization 1456
12.5.4.1.1 Reset Behavior 1456
12.5.4.1.2 Register Setup 1456
12.5.4.1.3 Pixel Selection (Framing) Register Dependencies 1458
12.5.4.2 Enable/Disable Hardware 1461
12.5.4.3 Events and Status Checking 1461
12.5.4.3.1 Interrupts 1461
12.5.4.3.2 CCDC_VD0_IRQ and CCDC_VD1_IRQ Interrupts 1461
12.5.4.3.3 CCDC_VD2_IRQ Interrupt 1462
12.5.4.3.4 Status Checking 1462
12.5.4.4 Register Accessibility During Frame Processing 1462
12.5.4.5 Interframe Operations 1463
12.5.4.6 CCDC Operations 1463
12.5.4.6.1 Image-Sensor Configuration 1463
12.5.4.6.2 Image-Signal Processing 1466
12.5.4.7 Summary of Constraints 1473
12.5.5 Programming the Preview Engine 1473
12.5.5.1 Preview Hardware Setup/Initialization 1473
12.5.5.1.1 Reset Behavior 1473
12.5.5.1.2 Register Setup 1473
12.5.5.1.3 Table Setup 1475
12.5.5.2 Enable/Disable Hardware 1476
12.5.5.3 Events and Status Checking 1476
12.5.5.4 Register Accessibility During Frame Processing 1476
12.5.5.5 Interframe Operations 1477
12.5.5.6 Summary of Constraints 1477
12.5.6 Programming the Resizer 1478
12.5.6.1 Resizer Hardware Setup/Initialization 1478
12.5.6.1.1 Reset Behavior 1478
12.5.6.1.2 Register Setup 1478
12.5.6.2 Enable/Disable Hardware 1479
12.5.6.3 Events and Status Checking 1479
12.5.6.4 Register Accessibility During Frame Processing 1480
12.5.6.5 Inter-Frame Operations 1480
12.5.6.5.1 Multiple Passes for Large Resizing Operations 1481
12.5.6.5.2 Processing Time Calculation 1481
12.5.6.6 Summary of Constraints 1482
12.5.7 Programming the H3A 1483
12.5.7.1 Hardware Setup/Initialization 1483
12.5.7.1.1 Reset Behavior 1483
12.5.7.1.2 Register Setup 1483
12.5.7.2 Enable/Disable Hardware 1484

12.5.7.3 Event and Status Checking 1485
12.5.7.4 Register Accessibility During Frame Processing 1485
12.5.7.5 Interframe Operations 1485
12.5.7.6 Summary of Constraints 1485
12.5.8 Programming the Histogram 1486
12.5.8.1 Hardware Setup/Initialization 1486
12.5.8.1.1 Reset Behavior 1486
12.5.8.1.2 Reset of Histogram Output Memory 1486
12.5.8.1.3 Register Setup 1486
12.5.8.2 Enable/Disable Hardware 1487
12.5.8.3 Event and Status Checking 1487
12.5.8.4 Register Accessibility During Frame Processing 1488
12.5.8.5 Interframe Operations 1488
12.5.8.6 Summary of Constraints 1488
12.5.9 Programming the Central-Resource SBL 1489
12.5.9.1 Hardware Setup/Initialization 1489
12.5.9.1.1 Reset Behavior 1489
12.5.9.1.2 Register Setup 1489
12.5.9.2 Enable/Disable Hardware 1489
12.5.9.3 Event and Status Checking 1489
12.5.9.4 Register Accessibility During Frame Processing 1490
12.5.9.5 Camera ISP Bandwidth Adjustments 1491
12.5.9.5.1 Input From CCDC Video-Port Interface 1491
12.5.9.5.2 Input From Memory 1491
12.5.10 Programming the Circular Buffer (CBUFF) 1492
12.5.10.1 Hardware Setup/Initialization 1492
12.5.10.2 Reset Behavior 1492
12.5.10.3 Register Setup 1493
12.5.10.4 Event and status Checking 1493
12.5.10.4.1 Interrupts 1493
12.5.10.4.2 Status Checking 1493
12.5.10.5 Register Accessibility During Frame Processing 1493
12.5.10.6 Operations 1494
12.6 Camera ISP Register Manual 1495
12.6.1 Camera ISP Instance Summary 1495
12.6.2 Camera ISP Registers 1495
12.6.2.1 Camera ISP Register Summary 1495
12.6.2.2 Camera ISP Register Description 1495
12.6.3 Camera ISP_CBUFF Registers 1521
12.6.3.1 Camera ISP_CBUFF Register Summary 1521
12.6.3.2 Camera ISP_CBUFF Register Description 1522
12.6.4 Camera ISP_CSI1B Registers 1529
12.6.4.1 Camera ISP_CSI1B Register Summary 1529
12.6.4.2 Camera ISP_CSI1B Register Description 1530
12.6.5 Camera ISP_CCDC Registers 1549
12.6.5.1 Camera ISP_CCDC Register Summary 1549
12.6.5.2 Camera ISP_CCDC Register Description 1550
12.6.6 Camera ISP_HIST Registers 1582
12.6.6.1 Camera ISP_HIST Register Summary 1582

12.6.6.2 Camera ISP_HIST Register Description 1582
12.6.7 Camera ISP_H3A Registers 1589
12.6.7.1 Camera ISP_H3A Register Summary 1589
12.6.7.2 Camera ISP_H3A Register Description 1589
12.6.8 Camera ISP_PREVIEW Registers 1603
12.6.8.1 Camera ISP_PREVIEW Register Summary 1603
12.6.8.2 Camera ISP_PREVIEW Register Description 1604
12.6.9 Camera ISP_RESIZER Registers 1629
12.6.9.1 Camera ISP_RESIZER Register Summary 1629
12.6.9.2 Camera ISP_RESIZER Register Description 1630
12.6.10 Camera ISP_SBL Registers 1654
12.6.10.1 Camera ISP_SBL Register Summary 1654
12.6.10.2 Camera ISP_SBL Register Description 1655
12.6.11 Camera ISP_CSI2A Registers 1699
12.6.11.1 Camera ISP_CSI2A Register Summary 1700
12.6.11.2 Camera ISP_CSI2A Register Description 1700
12.6.12 Camera CSI2PHY_SCP Registers 1726
12.6.12.1 Camera CSI2PHY_SCP Register Summary 1726
12.6.12.2 Camera CSI2PHY_SCP Register Description 1726

13 2D/3D Graphics Accelerator 1731

13.1 SGX Overview 1732
13.1.1 POWERVR SGX Main Features 1733
13.1.2 SGX 3D Features 1733
13.1.3 Universal Scalable Shader Engine (USSE) – Key Features 1734
13.2 SGX Integration 1735
13.2.1 Clocking, Reset, and Power-Management Scheme 1735
13.2.1.1 Clocks 1735
13.2.1.2 Resets 1736
13.2.1.3 Power Management 1736
13.2.2 Hardware Requests 1736
13.2.2.1 Interrupt Request 1736
13.3 SGX Functional Description 1737
13.3.1 SGX Block Diagram 1737
13.3.2 SGX Elements Description 1737
13.4 SGX Register Mapping 1738

14 IVA2.2 Subsystem 1739

14.1 IVA2.2 Subsystem Overview 1740
14.1.1 IVA2.2 Subsystem Key Features 1740
14.2 IVA2.2 Subsystem Integration 1742
14.2.1 Clocking, Reset, and Power-Management Scheme 1743
14.2.1.1 Clocks 1743
14.2.1.1.1 IVA2.2 Clocks 1743
14.2.1.2 Resets 1743
14.2.1.2.1 Hardware Resets 1743
14.2.1.2.2 Software Resets 1745
14.2.1.3 Power Domain 1745
14.2.2 Hardware Requests 1747
14.2.2.1 DMA Requests 1747

14.2.2.2 Interrupt Requests 1748
14.3 IVA2.2 Subsystem Functional Description 1751
14.3.1 DSP Megamodule 1752
14.3.1.1 DSP Overview 1753
14.3.1.2 Program Memory Controller Overview 1754
14.3.1.3 DMC Overview 1754
14.3.1.4 UMC Overview 1755
14.3.1.5 EMC Overview 1756
14.3.1.6 Memory Protection Overview 1756
14.3.1.7 INTC 1756
14.3.1.7.1 Event Type 1757
14.3.1.7.2 Event Behavior 1758
14.3.1.7.3 Event Detection 1758
14.3.1.7.4 Event Selection 1759
14.3.1.7.5 Event Combination 1759
14.3.1.7.6 Interrupt Event Error 1760
14.3.1.7.7 PDC Overview 1760
14.3.1.8 Other DSP Reference Documents 1760
14.3.2 DMA Engines 1761
14.3.2.1 EDMA 1761
14.3.2.1.1 Third-Party Channel Controller 1762
14.3.2.1.2 Third-Party Transfer Controller 1767
14.3.2.1.3 EDMA Hardware Parameters 1770
14.3.2.2 EDMA Access to Video Accelerator/Sequencer 1771
14.3.2.3 IDMA 1772
14.3.3 MMU 1772
14.3.3.1 MMU VA-to-PA Translation 1773
14.3.3.2 MMU Configuration 1774
14.3.4 Video Sequencer 1774
14.3.4.1 DMA Access 1775
14.3.4.2 Sequencer Interrupts 1775
14.3.4.3 Sequencer to DSP Megamodule Interrupts 1776
14.3.4.4 I/O Interrupts 1776
14.3.4.5 IRQ Generic 1777
14.3.4.6 Sequencer Core 1777
14.3.4.7 Exception Handling 1777
14.3.4.8 Sequencer Memory Mapping 1778
14.3.5 iLF and iME Common Part 1779
14.3.5.1 iLF/iME Control 1779
14.3.5.1.1 Commands 1779
14.3.5.1.2 Execution States 1779
14.3.5.2 Instruction Set 1780
14.3.5.2.1 LoadPStack(), iLF Only 1782
14.3.5.2.2 LoadPStack(), iME Only 1782
14.3.5.2.3 LoadInstBuf() 1783
14.3.5.2.4 GenerateIT() 1784
14.3.5.2.5 EndPgm() 1784
14.3.5.3 Parameters in L2 1784
14.3.5.3.1 Instructions 1785

14.3.6 iLF Module 1785
14.3.6.1 iLF Control 1786
14.3.6.1.1 Instruction Set 1786
14.3.6.1.2 Commands 1787
14.3.6.1.3 Execution States 1787
14.3.6.2 Instructions 1787
14.3.6.2.1 ParseEdge() 1788
14.3.6.2.2 FilterEdge() 1789
14.3.6.2.3 LoadEFPT() 1790
14.3.6.3 Parameters 1790
14.3.6.3.1 Instructions 1790
14.3.6.3.2 Parameter Stack 1790
14.3.6.3.3 Loop-Filter Parameter Stack and EFPT for H264 1793
14.3.6.3.4 Loop-Filter Parameter Stack for H263 1794
14.3.6.3.5 Loop-Filter Parameter Stack for WMV9 1795
14.3.6.3.6 Loop-Filter Parameter Stack and EFPT for REAL9 1795
14.3.6.3.7 Input/Write Buffer 1796
14.3.6.3.8 Parse-Edge Parameters 1798
14.3.7 iME Module 1798
14.3.7.1 iME Control 1799
14.3.7.1.1 Instruction Set 1799
14.3.7.1.2 Commands 1800
14.3.7.1.3 Execution States 1801
14.3.7.2 Instructions 1801
14.3.7.2.1 ErrorCalc() 1801
14.3.7.2.2 SaveErrs() 1802
14.3.7.2.3 SaveStatus() 1803
14.3.7.2.4 RestoreErrs() 1803
14.3.7.2.5 Mcompare () 1804
14.3.7.2.6 Mcompare2 () 1804
14.3.7.2.7 Mcompare4 () 1805
14.3.7.2.8 ClearStatus() 1806
14.3.7.2.9 LoadRefBlk() 1806
14.3.7.2.10 LoadRefBlk_Ind() 1807
14.3.7.2.11 SaveBestMatch() 1807
14.3.7.2.12 RestoreBestMatch() 1808
14.3.7.2.13 ExitOnMinReached() 1808
14.3.7.2.14 WaitOnSignal() 1809
14.3.7.2.15 Memcopy() 1809
14.3.7.2.16 Interpolate() 1810
14.3.7.3 Parameters 1811
14.3.7.3.1 Instruction Buffer in L2 1811
14.3.7.3.2 Generic Parameter Stack in L2 1811
14.3.7.3.3 Interpolate Parameters 1812
14.3.7.3.4 Pixel Format in L2 1814
14.3.7.3.5 Motion Vector Cost Table Format in L2 1814
14.3.7.3.6 Error Table 1815
14.3.8 Video Accelerator/Sequencer Local Interconnect 1816

14.3.9 SL2 Interface 1816
14.3.9.1 BWO 1817
14.3.9.2 Arbiter 1818
14.3.9.3 Restrictions on SL2 Memory Usage 1818
14.3.9.4 Error Management 1818
14.3.10 Wake-Up Generator 1818
14.3.10.1 Interrupts, DMA Requests, and Event Management 1819
14.3.10.1.1 Event Generation 1820
14.3.10.1.2 Individual Event Masking 1820
14.3.10.1.3 Individual Event Mask Clear 1821
14.3.10.2 Idle Handshake 1822
14.3.11 SYSC Module 1822
14.3.11.1 Divided Clock Generation 1823
14.3.11.2 Clock Management, Power-Down, and Wake-Up 1823
14.3.11.3 Boot Configuration 1824
14.3.11.4 Interconnect Optimization 1824
14.3.11.5 Video Accelerator/Sequencer SYSC 1824
14.3.11.5.1 Reset 1824
14.3.11.5.2 Power Management 1825
14.3.11.5.3 Interrupt Handler 1825
14.3.12 Local Memories 1825
14.3.12.1 ROM Overview 1827
14.3.12.2 RAM Overview 1827
14.3.13 Local Interconnect Network 1827
14.3.13.1 Endianness 1828
14.3.14 Error Reporting 1828
14.4 IVA2.2 Subsystem Basic Programming Model 1829
14.4.1 IVA2.2 Boot 1829
14.4.1.1 IVA2.2 Boot Configuration 1829
14.4.1.1.1 IDLE Boot Mode 1830
14.4.1.1.2 Wait in Self Loop Mode 1830
14.4.1.1.3 Default Config Cache Mode 1830
14.4.1.1.4 User Defined Bootstrap Mode 1831
14.4.1.2 Example of IVA2.2 Boot 1831
14.4.1.2.1 Boot Under MPU Control 1831
14.4.1.2.2 Autonomous Boot 1833
14.4.2 Sequencer Boot/Reset 1834
14.4.3 Cache Management 1834
14.4.3.1 Cache-Size Configuration 1834
14.4.3.2 Cache Mode Configuration 1836
14.4.3.3 Cacheability Settings 1837
14.4.3.4 Coherence Maintenance 1837
14.4.3.4.1 Memory-Mapped L1P and L1D Coherence 1837
14.4.3.4.2 Memory-Mapped L2 Coherence 1837
14.4.3.4.3 Device Memory Coherence 1837
14.4.3.4.4 Global Cache Management 1838
14.4.3.4.5 Block Cache Management 1839
14.4.3.4.6 Write-Back Completion 1841
14.4.3.4.7 Performance Consideration Timing 1842

14.4.4 DMA Management 1843
14.4.4.1 Transfers From/to Device Memories/Peripherals (EDMA) 1843
14.4.4.2 Internal Memory-to-Memory Transfer (IDMA) 1843
14.4.4.3 Programming an EDMA Transfer 1844
14.4.4.4 Defining a Logical Channel 1844
14.4.4.4.1 Single Logical Channel Definition 1844
14.4.4.4.2 Controlling Submission Granularity 1845
14.4.4.4.3 Linking to Another Logical Channel 1845
14.4.4.4.4 Chaining Logical Channel 1845
14.4.4.5 Prioritizing Defined Transfers 1846
14.4.4.5.1 Mapping Between DMA/QDMA Events and Event Queues 1846
14.4.4.5.2 Mapping a Queue to a Transfer Controller 1846
14.4.4.5.3 Handling Priority 1847
14.4.4.5.4 Aged Priority 1847
14.4.4.5.5 Optimizing 2D Transfers 1847
14.4.4.6 Starting the Transfer 1847
14.4.4.6.1 Assigning a Logical Channel to a Trigger Event 1848
14.4.4.6.2 Manual Trigger (Software-Synchronized Transfers) 1848
14.4.4.6.3 Hardware Trigger (Hardware-Synchronized Transfers) 1848
14.4.4.6.4 Automatic Trigger (QDMA) 1848
14.4.4.6.5 Offloaded Configuration (Using IDMA) 1849
14.4.4.6.6 Direct Configuration to Transfer Channel (Not Recommended) 1850
14.4.4.6.7 DMA Completion Mode 1850
14.4.4.6.8 Partial Versus Total Completion 1850
14.4.4.6.9 Tracking DMA Completion 1851
14.4.4.6.10 DMA Interrupt Service Routine 1852
14.4.4.6.11 Benchmarking 1852
14.4.5 IVA2.2 Extended Function Interface 1852
14.4.5.1 Overview 1852
14.4.5.2 C64x+ EFI Instructions 1853
14.4.5.3 C64x+ EFI Use in IVA2.2 1857
14.4.5.3.1 Read Registers Using the EFI Programming Model 1857
14.4.5.3.2 Write Registers Using the EFI Programming Model 1857
14.4.6 iME and iLF Basic Programming Model 1858
14.4.6.1 Typical Use 1858
14.4.7 Interrupt Management 1859
14.4.7.1 Interrupt Flow in IVA2.2 Subsystem 1860
14.4.7.2 Event Combined Programming Sequence 1862
14.4.7.3 Event <-> Interrupt Mapping Programming Sequence 1862
14.4.7.4 Interrupt Exception Programming Sequence 1862
14.4.7.5 Interrupt Controller Basic Programming Model for Power Down of IVA2.2 Subsystem 1863
14.4.7.6 Interrupt Controller Basic Programming Model for Power On of IVA2.2 Subsystem 1864
14.4.7.7 Video and Sequencer Module interrupt Handling 1866
14.4.7.7.1 Sequencer Interrupt 1866
14.4.7.7.2 DSP Megamodule Interrupt 1867
14.4.8 Memory Management 1868
14.4.8.1 External Memory 1868
14.4.8.1.1 Cacheability 1868

14.4.8.1.2 Virtual Addressing 1868
14.4.8.2 Internal Memory 1868
14.4.8.2.1 Memory Protection 1868
14.4.8.2.2 Bandwidth Management 1873
14.4.8.3 SL2 Memory Management 1875
14.4.8.3.1 SL2 Performance Optimizations 1875
14.4.8.3.2 SL2 Performance Limitations 1875
14.4.8.3.3 SL2 Illegal Accesses 1875
14.4.9 IVA2.2 Power Management 1876
14.4.9.1 Clock Management 1876
14.4.9.1.1 Clock Configuration 1876
14.4.9.1.2 Clock Gating 1876
14.4.9.2 Reset Management 1877
14.4.9.3 Power-Down and Wake-Up Management 1877
14.4.9.4 Powering Down L2$ Memory While IVA2 is Active 1880
14.4.9.5 Video and Sequencer Module Management 1881
14.4.9.5.1 Module Dynamic Power Savings 1881
14.4.9.5.2 System Dynamic Power Savings 1881
14.4.10 Error Identification Process 1882
14.4.10.1 Error Reporting for IDMA Module 1882
14.4.10.2 Error Reporting for EDMA Module 1882
14.4.10.3 Error Reporting for the L3 Interconnect 1883
14.4.11 Recommendations for Static Settings 1883
14.5 IVA2.2 Subsystem Register Manual 1884
14.5.1 IC Registers 1885
14.5.1.1 IC Register Mapping Summary 1885
14.5.1.2 IC Register Descriptions 1885
14.5.2 SYS Registers 1893
14.5.2.1 SYS Register Mapping Summary 1893
14.5.2.2 SYS Register Descriptions 1893
14.5.3 IDMA Registers 1896
14.5.3.1 IDMA Register Mapping Summary 1896
14.5.3.2 IDMA Register Descriptions 1896
14.5.4 XMC Registers 1907
14.5.4.1 XMC Register Mapping Summary 1908
14.5.4.2 XMC Register Descriptions 1909
14.5.5 TPCC Registers 1934
14.5.5.1 TPCC Register Mapping Summary 1934
14.5.5.2 TPCC Register Descriptions 1937
14.5.6 TPTC0 and TPTC1 Registers 2033
14.5.6.1 TPTC0 and TPTC1 Register Mapping Summary 2033
14.5.6.2 TPTC0 and TPTC1 Register Descriptions 2035
14.5.7 SYSC Registers 2058
14.5.7.1 SYSC Register Mapping Summary 2058
14.5.7.2 SYSC Register Descriptions 2059
14.5.8 WUGEN Registers 2063
14.5.8.1 WUGEN Register Mapping Summary 2063
14.5.8.2 WUGEN Register Descriptions 2064
14.5.9 SEQ Registers 2081

14.5.9.1 SEQ Register Mapping Summary 2081
14.5.9.2 SEQ Register Descriptions 2082
14.5.10 Video System Controller Registers 2089
14.5.10.1 Video System Controller Register Mapping Summary 2089
14.5.10.2 Video System Controller Register Descriptions 2089
14.5.11 iME Registers 2096
14.5.11.1 iME Register Mapping Summary 2096
14.5.11.2 iME Register Descriptions 2097
14.5.12 iLF Registers 2107
14.5.12.1 iLF Register Mapping Summary 2107
14.5.12.2 iLF Register Descriptions 2108
14.5.13 IA_GEM Registers 2118
14.5.13.1 IA_GEM Register Mapping Summary 2118
14.5.13.2 IA_GEM Register Descriptions 2118
14.5.14 IA_EDMA Registers 2119
14.5.14.1 IA_EDMA Register Mapping Summary 2119
14.5.14.2 IA_EDMA Register Descriptions 2119
14.5.15 IA_SEQ Registers 2120
14.5.15.1 IA_SEQ Register Mapping Summary 2121
14.5.15.2 IA_SEQ Register Descriptions 2121

15 Display Subsystem 2123

15.1 Display Subsystem Overview 2124
15.2 Display Subsystem Environment 2129
15.2.1 LCD Support 2129
15.2.1.1 Parallel Interface 2129
15.2.1.1.1 Parallel Interface in RFBI Mode (MIPI DBI Protocol) 2130
15.2.1.1.2 Parallel Interface in Bypass Mode (MIPI DPI Protocol) 2132
15.2.1.1.3 LCD Output and Data Format for the Parallel Interface 2133
15.2.1.1.4 Transaction Timing Diagrams 2137
15.2.1.2 SDI Serial Interface 2144
15.2.1.3 DSI Serial Interface 2147
15.2.2 LCD Support With MIPI DSI 1.0 Protocol and Data Format 2147
15.2.2.1 Physical Layer 2147
15.2.2.1.1 Data/Clock Configuration 2148
15.2.2.1.2 ULPS 2149
15.2.2.2 Video Port (VP) Interface 2149
15.2.2.2.1 Video Port Used for Video Mode 2150
15.2.2.2.2 Video Port Used on Command Mode 2155
15.2.2.2.3 Burst Mode 2157
15.2.2.3 Multilane Layer 2158
15.2.2.3.1 SoT and EoT in Multilane Configurations 2158
15.2.2.3.2 Lane Splitter 2158
15.2.2.4 Protocol Layer 2159
15.2.2.4.1 Short Packet 2159
15.2.2.4.2 Long Packet 2160
15.2.2.4.3 Data Identifier 2161
15.2.2.4.4 Virtual Channel ID - VC Field, DI[7:6] 2161
15.2.2.4.5 Data Type Field DT[5:0] 2161

15.2.2.4.6 Pixel Data Formats in Video Mode 2162
15.2.2.4.7 Synchronization Codes 2162
15.2.2.4.8 Blanking 2162
15.2.2.4.9 Frame Structures 2167
15.2.2.4.10 Virtual Channels 2170
15.2.2.5 Pixel Data Formats 2170
15.2.2.5.1 24 Bits per Pixel - RGB Color Format, Long Packet 2171
15.2.2.5.2 18 Bits per Pixel (Loosely Packed) - RGB Color Format, Long Packet 2172
15.2.2.5.3 18 Bits per Pixel (Packed) - RGB Color Format, Long Packet 2173
15.2.2.5.4 16 Bits per Pixel - RGB Color Format, Long Packet 2174
15.2.3 LCD Output With TI FlatLink3G Data Format for the SDI Module 2174
15.2.4 TV Display Support 2175
15.2.4.1 TV Output and Data Format 2178
15.2.4.2 Digital-to-Analog Converter 2178
15.3 Display Subsystem Integration 2179
15.3.1 Clocking, Reset, and Power-Management Scheme 2181
15.3.1.1 Clocks 2181
15.3.1.2 Resets 2184
15.3.1.2.1 Hardware Reset 2184
15.3.1.2.2 Software Reset 2184
15.3.1.3 Power Domain 2185
15.3.1.4 Power Management 2185
15.3.1.4.1 Clock Activity Mode 2185
15.3.1.4.2 Autoidle Mode 2186
15.3.1.4.3 Idle Mode 2186
15.3.1.4.4 SDI Idle Mode 2186
15.3.1.4.5 Wake-Up Mode 2187
15.3.1.4.6 Standby Mode 2187
15.3.2 Hardware Requests 2190
15.3.2.1 DMA Requests 2190
15.3.2.1.1 Display Controller DMA Request (Line Trigger) 2190
15.3.2.1.2 DSI Protocol Engine DMA Request 2191
15.3.2.1.3 RFBI DMA Request 2191
15.3.2.2 Interrupt Requests 2191
15.3.2.2.1 DISPC Interrupt Request 2192
15.3.2.2.2 DSI Interrupt Request 2193
15.4 Display Subsystem Functional Description 2196
15.4.1 Block Diagram 2196
15.4.2 Display Controller Functionalities 2197
15.4.2.1 Display Modes 2198
15.4.2.1.1 LCD Output 2198
15.4.2.1.2 Digital Output 2198
15.4.2.2 Graphics Pipeline 2198
15.4.2.2.1 Graphics Memory Format 2198
15.4.2.2.2 Color Look-Up Table/Gamma Table 2200
15.4.2.3 Video Pipeline 2202
15.4.2.3.1 Video Memory Formats 2202
15.4.2.3.2 Color Space Conversion 2203
15.4.2.3.3 Hardware Cursor 2206

15.4.2.3.4 Up-/Down-Sampling 2206
15.4.2.4 Overlay Support 2210
15.4.2.4.1 Priority Rule 2210
15.4.2.4.2 Transparency Color Keys 2215
15.4.2.4.3 Overlay Optimization (Only Available in Normal Mode) 2217
15.4.2.5 Active/Passive Matrix Display Data Path 2217
15.4.2.5.1 Color Phase Rotation 2217
15.4.2.5.2 Passive Matrix Display Dithering Logic 2218
15.4.2.5.3 Passive Matrix Display Output FIFO 2219
15.4.2.5.4 Multiple Cycle Output Format 2219
15.4.2.6 Video Line Buffer 2220
15.4.2.7 Synchronized Buffer Update 2220
15.4.2.8 Rotation 2220
15.4.2.9 Multiple Buffer Support 2220
15.4.3 DSI Protocol Engine Functionalities 2221
15.4.3.1 DSI Protocol Architecture 2221
15.4.3.2 Clock Requirements 2222
15.4.3.2.1 Timing Parameters for an LP to HS Transaction 2223
15.4.3.2.2 Timing Parameters for an HS to LP Transaction 2224
15.4.3.2.3 Extra LP Transitions 2225
15.4.3.3 DSI Transfer Modes 2226
15.4.3.3.1 Video Mode 2226
15.4.3.3.2 Command Mode 2226
15.4.3.3.3 Video + Command Modes 2227
15.4.3.3.4 Burst Modes 2227
15.4.3.3.5 Interleaving Mode 2228
15.4.3.4 Power Management 2232
15.4.3.5 Serial Configuration Port (SCP) Interface 2232
15.4.3.5.1 Shadowing Register 2232
15.4.3.5.2 Busy Signal 2233
15.4.3.6 Power Control 2233
15.4.3.6.1 Complex I/O Power Control Commands 2234
15.4.3.6.2 DSI PLL Power Control Commands 2235
15.4.3.7 Timers 2237
15.4.3.7.1 Twakeup Timer 2237
15.4.3.7.2 ForceTxStopMode FSM 2238
15.4.3.7.3 TurnRequest FSM 2238
15.4.3.7.4 Peripheral Reset Timer 2239
15.4.3.7.5 HS TX Timer 2239
15.4.3.7.6 LP RX Timer 2240
15.4.3.8 Bus Turnaround 2241
15.4.3.9 PHY Triggers 2242
15.4.3.9.1 Reset 2243
15.4.3.9.2 Tearing Effect 2243
15.4.3.9.3 Acknowledge 2244
15.4.3.10 ECC Generation 2244
15.4.3.11 Checksum Generation for Long Packet Payloads 2245
15.4.3.12 End of Transfer Packet 2246

15.4.4 DSI PLL Controller Functionalities 2246
15.4.4.1 DSI PLL Controller Overview 2246
15.4.4.2 DSI PLL Controller Architecture 2247
15.4.4.3 DSI PLL Operations 2248
15.4.4.4 DSI PLL Controller Shadowing Mechanism 2249
15.4.4.5 Error Handling 2249
15.4.5 DSI Complex I/O Functionalities 2249
15.4.5.1 DSI Complex I/O Overview 2249
15.4.5.2 DSI Complex I/O Architecture 2250
15.4.6 RFBI Functionalities 2250
15.4.6.1 RFBI FIFO 2251
15.4.6.2 RFBI Interconnect FIFO 2251
15.4.6.3 Input Pixel Formats 2251
15.4.6.4 Output Parallel Modes 2251
15.4.6.5 Unmodified Bits 2252
15.4.6.6 Bypass Mode 2252
15.4.6.7 Send Commands 2252
15.4.6.8 Read/Write 2252
15.4.7 Video Encoder Functionalities 2253
15.4.7.1 Test Pattern Generation 2254
15.4.7.2 Luma Stage 2255
15.4.7.3 Chroma Stage 2255
15.4.7.4 Subcarrier and Burst Generation 2255
15.4.7.5 Closed Caption Encoding 2256
15.4.7.6 Wide-Screen Signaling (WSS) Encoding 2258
15.4.7.7 Video DAC Architecture 2260
15.4.7.8 Video DC/AC Coupled TV Load 2260
15.4.7.9 TV Detection/Disconnection Pulse Generation and Usage 2260
15.4.7.9.1 TV Detection/Disconnection Pulse Generation 2261
15.4.7.9.2 TV Detection Procedure 2261
15.4.7.9.3 TV Disconnection Procedure 2262
15.4.7.9.4 Recommended TV Detection/Disconnection Pulse Waveform 2262
15.4.7.9.5 TV Detection/Disconnection Usage 2263
15.4.7.10 Video DAC Bypass Mode 2264
15.4.7.11 Video Dual-DAC Test Mode 2265
15.4.8 SDI Functionalities 2266
15.5 Display Subsystem Basic Programming Model 2268
15.5.1 Display Subsystem Reset 2268
15.5.2 Display Subsystem Configuration Phase 2268
15.5.3 Display Controller Basic Programming Model 2269
15.5.3.1 Display Controller Configuration 2270
15.5.3.2 Graphics Layer Configuration 2270
15.5.3.2.1 Graphics DMA Registers 2270
15.5.3.2.2 Graphics Layer Configuration Registers 2272
15.5.3.2.3 Graphics Window Attributes 2272
15.5.3.3 Video Layer Configuration 2275
15.5.3.3.1 Video DMA Registers 2275
15.5.3.3.2 Video Configuration Register 2276
15.5.3.3.3 Video Window Attributes 2277

15.5.3.3.4 Video Up-/Down-Sampling Configuration 2278
15.5.3.3.5 Video Color Space Conversion Configuration 2280
15.5.3.4 Rotation/Mirroring Display Subsystem Settings 2280
15.5.3.4.1 Image Data Formats 2281
15.5.3.4.2 Image Data from On-Chip SRAM 2281
15.5.3.4.3 Image Data from External SRAM 2286
15.5.3.4.4 Additional configuration when using YUV format 2289
15.5.3.5 LCD-Specific Control Registers 2290
15.5.3.5.1 LCD Attributes 2290
15.5.3.5.2 LCD Timings 2291
15.5.3.5.3 LCD Overlay 2294
15.5.3.5.4 LCD TDM 2294
15.5.3.5.5 LCD Spatial/Temporal Dithering 2294
15.5.3.5.6 LCD Color Phase Rotation 2295
15.5.3.6 TV Set-Specific Control Registers 2298
15.5.3.6.1 Digital Timings 2299
15.5.3.6.2 Digital Frame/Field Size 2299
15.5.3.6.3 Digital Overlay 2299
15.5.4 DSI Protocol Engine Basic Programming Model 2299
15.5.4.1 Software Reset 2300
15.5.4.2 Power Management 2300
15.5.4.3 Interrupts 2300
15.5.4.4 Global Register Controls 2300
15.5.4.5 Virtual Channels 2301
15.5.4.6 Packets 2301
15.5.4.7 DSI Complex I/O 2302
15.5.4.8 Video Mode 2302
15.5.4.9 Video Port Data Bus 2303
15.5.4.10 Command Mode 2303
15.5.4.10.1 Command Mode TX FIFO 2303
15.5.4.10.2 Command Mode RX FIFO 2306
15.5.4.10.3 Command Mode DMA Requests 2307
15.5.4.11 Ultra-Low Power State 2308
15.5.4.11.1 Entering ULPS 2308
15.5.4.11.2 Exiting ULPS 2308
15.5.4.12 DSI Programming Sequence Example 2310
15.5.4.12.1 Video Mode Transfer 2310
15.5.4.12.2 Command Mode Transfer Example 1 2310
15.5.4.12.3 Command Mode Transfer Example 2 2311
15.5.5 DSI PLL Controller Basic Programming Model 2312
15.5.5.1 Software Reset 2312
15.5.5.2 DSI PLL Programming Blocks 2313
15.5.5.3 DSI PLL Go Sequence 2313
15.5.5.4 DSI PLL Clock Gating Sequence 2315
15.5.5.5 DSI PLL Lock Sequence 2316
15.5.5.6 DSI PLL Error Handling 2320
15.5.5.7 DSI PLL Recommended Values 2320
15.5.6 DSI Complex I/O Basic Programming Model 2321

15.5.6.1 Software Reset 2321
15.5.6.2 Reset-Done Bits 2321
15.5.6.3 Pad Configuration 2322
15.5.6.4 Display Timing Configuration 2322
15.5.6.4.1 High-Speed Clock Transmission 2322
15.5.6.4.2 High-Speed Data Transmission 2324
15.5.6.4.3 Other DSI_PHY Transmission and Reception 2325
15.5.6.5 Error Handling 2325
15.5.7 RFBI Basic Programming Model 2326
15.5.7.1 DISPC Control Registers 2326
15.5.7.2 RFBI Control Registers 2326
15.5.7.2.1 High Threshold 2326
15.5.7.2.2 Bypass Mode 2327
15.5.7.2.3 Enable 2327
15.5.7.2.4 Configuration Selection 2327
15.5.7.2.5 ITE Bit 2327
15.5.7.2.6 Number of Pixels to Transfer 2328
15.5.7.2.7 Programmable Line Number 2328
15.5.7.3 RFBI Configuration 2328
15.5.7.3.1 Parallel Mode 2329
15.5.7.3.2 Trigger Mode 2329
15.5.7.3.3 VSYNC Pulse Width (Minimum Value) 2329
15.5.7.3.4 HSYNC Pulse Width (Minimum Value) 2329
15.5.7.3.5 Cycle Format 2329
15.5.7.3.6 Unused Bits 2329
15.5.7.3.7 RFBI Timings 2329
15.5.7.3.8 RFBI State-Machine 2331
15.5.7.3.9 RFBI Configuration Flow Charts 2332
15.5.8 Video Encoder Basic Programming Model 2335
15.5.8.1 Video Encoder Software Reset 2335
15.5.8.2 Video DAC Settings 2336
15.5.8.3 Video Encoder Programming Sequence 2336
15.5.8.4 Video Encoder Register Settings 2336
15.5.9 SDI Basic Programming Model 2337
15.5.9.1 SDI Configuration 2338
15.5.9.1.1 SDI PLL Configuration 2338
15.5.9.1.2 Signal Features Configuration 2338
15.5.9.1.3 Number of Data Pairs 2338
15.5.9.2 SDI Power-Management Programming Sequence 2339
15.5.9.2.1 SDI Reset State 2339
15.5.9.2.2 SDI Power_On Sequence 2339
15.5.9.2.3 SDI Power-Down Sequence 2340
15.5.9.3 SDI Start Sequence 2341
15.5.9.4 SDI Stop Sequence 2342
15.5.9.5 Clock Source/Frequency Change Sequence 2342
15.5.9.5.1 Complete Sequence 2343
15.5.9.5.2 Simplified Sequence When LCD and PCD Are Swapped 2345
15.5.9.6 SDI Error Management 2345
15.6 Display Subsystem Use Cases and Tips 2346

15.6.1 How to Configure the Scaling Unit in the DISPC Module 2346
15.6.1.1 Filtering 2346
15.6.1.1.1 Vertical Filtering 2346
15.6.1.1.2 Horizontal Filtering 2348
15.6.1.2 Scaling Algorithms 2349
15.6.1.3 Scaling Settings 2350
15.6.1.3.1 Register List 2350
15.6.1.3.2 Enabling 2351
15.6.1.3.3 Factor 2352
15.6.1.3.4 Initial Phase 2352
15.6.1.3.5 Coefficients 2353
15.6.2 Display Low-Power Refresh Settings 2357
15.6.2.1 Display Low-Power Refresh Overview 2357
15.6.2.2 Display Subsystem Clock 2358
15.6.2.2.1 Display Subsystem Clock Configuration 2358
15.6.2.2.2 Display Subsystem Clock Enable 2359
15.6.2.3 DPLL4 in Low-Power Mode 2359
15.6.2.4 Autoidle and Smart Idle 2360
15.6.2.4.1 Autoidle 2360
15.6.2.4.2 Smart-Idle 2360
15.6.2.5 FIFO Thresholds 2360
15.6.2.5.1 FIFO Threshold Settings to Reduce Power Consumption 2360
15.6.2.6 Vertical and Horizontal Timings 2361
15.6.2.6.1 Horizontal and Vertical Timing Settings to Reduce Power Consumption 2362
15.6.3 How to Configure the Serial Display Interface Module With FlatLink3G Protocol 2362
15.6.3.1 SDI PLL Architecture 2362
15.6.3.2 SDI PLL Configuration 2362
15.6.3.3 Application Example: HVGA Display 2367
15.6.3.3.1 HVGA Display 2367
15.6.3.3.2 SDI PLL Settings for 1-Channel Mode: 2368
15.6.3.3.3 SDI PLL Settings for 2-Channel Mode: 2368
15.6.4 How to Interface OMAP Device With SN65LVDS302 Receiver for an XGA Display Application 2368
15.6.4.1 Hardware Connections 2369
15.6.4.2 SN65LVDS302 Receiver Description 2370
15.6.4.3 SDI Software Settings 2370
15.6.4.3.1 SDI Configuration 2370
15.6.4.3.2 Signal Features Configuration 2370
15.6.4.3.3 SDI PLL Configuration 2371
15.6.4.4 SN65LVDS302 Receiver Settings 2371
15.6.4.4.1 Receiver Power-Up 2371
15.6.4.4.2 Receiver Modes and Transitions 2372
15.6.4.4.3 Parity Error Detection and Handling 2373
15.6.5 Camcorder Use Case: How to Configure the Display Subsystem When Connected With a QVGA LCD Panel 2373
15.6.5.1 Overview 2374
15.6.5.2 Environment 2374
15.6.5.2.1 LCD panel Features 2375
15.6.5.3 Data Path 2375
15.6.5.4 Programming Flow 2377

15.6.5.4.1 Pads Multiplexing Configuration 2378
15.6.5.4.2 Display Subsystem Initialization 2379
15.6.5.4.3 Video1 Channel Configuration 2381
15.6.5.4.4 Interrupts Enable 2384
15.6.5.4.5 Display Panel Configuration 2385
15.6.5.4.6 LCD Enable 2387
15.6.6 How to Configure the DSI PLL in Video Mode 2388
15.6.7 DSI Video Mode Using the DISPC Video Port 2391
15.6.7.1 Display Subsystem Clock Configuration 2393
15.6.7.2 Configure DSI, DSI PLL and Complex I/O 2393
15.6.7.2.1 Reset DSI Modules 2393
15.6.7.2.2 Set Up DSI DPLL 2393
15.6.7.2.3 Switch to DSI PLL Clock Source 2394
15.6.7.2.4 Set Up DSI Protocol Engine 2394
15.6.7.2.5 Configure DSI_PHY 2396
15.6.7.2.6 Drive Stop State 2397
15.6.7.3 Initialization of the External MIPI Display Controller 2397
15.6.7.4 Configure the DISPC 2397
15.6.7.4.1 Reset DISPC 2397
15.6.7.4.2 Configure DISPC Timing, Window, and Color 2397
15.6.7.5 Enable Video Mode Using the DISPC Video Port 2398
15.6.8 DSI Command Mode Using the DISPC Video Port 2399
15.6.8.1 Display Subsystem Use Cases and Tips 2399
15.6.8.1.1 Configure DSS Clocks at the PRCM Module 2401
15.6.8.1.2 Configure DSI Protocol Engine, DSI PLL, and Complex I/O 2401
15.6.8.1.3 Initialization of the External MIPI LCD Controller 2406
15.6.8.1.4 Configure the DISPC 2406
15.6.8.1.5 Enable Command Mode Using DISPC Video Port 2407
15.6.8.1.6 Send Frame Data to LCD Panel Using Automatic TE 2408
15.7 Display Subsystem Register Manual 2409
15.7.1 Display Subsystem Register Mapping Summary 2409
15.7.2 Register Descriptions 2415
15.7.2.1 Display Subsystem and SDI Registers 2415
15.7.2.2 Display Controller Registers 2422
15.7.2.3 RFBI Registers 2465
15.7.2.4 Video Encoder Registers 2479
15.7.2.5 DSI Protocol Engine Registers 2505
15.7.2.6 DSI complex I/O Registers 2547
15.7.2.7 DSI PLL Control Module Registers 2550
16 Timers 2557
16.1 Timers Overview 2558
16.2 General-Purpose Timers 2559
16.2.1 GP Timers Overview 2559
16.2.1.1 GP Timers Features 2559
16.2.2 GP Timers Environment 2560
16.2.2.1 GP Timers External System Interface 2560
16.2.3 GP Timers Integration 2562
16.2.3.1 Clocking, Reset, and Power-Management Scheme 2562

16.2.3.1.1 Clock Management 2562
16.2.3.1.2 Wake-Up Capability 2565
16.2.3.1.3 Reset and Power Management 2566
16.2.3.2 Software Reset 2566
16.2.3.3 GP Timer Interrupts 2567
16.2.4 GP Timers Functional Description 2568
16.2.4.1 GP Timers Block Diagram 2569
16.2.4.2 Timer Mode Functionality 2570
16.2.4.2.1 1-ms Tick Generation (Only GPTIMER1, GPTIMER2, and GPTIMER10) 2571
16.2.4.3 Capture Mode Functionality 2573
16.2.4.4 Compare Mode Functionality 2574
16.2.4.5 Prescaler Functionality 2575
16.2.4.6 Pulse-Width Modulation 2575
16.2.4.7 Timer Counting Rate 2576
16.2.5 Timer Under Emulation 2577
16.2.6 Accessing GP Timer Registers 2578
16.2.6.1 Writing to Timer Registers 2578
16.2.6.1.1 Write Posting Synchronization Mode 2578
16.2.6.1.2 Write Nonposting Synchronization Mode 2579
16.2.6.2 Reading From Timer Counter Registers 2579
16.3 General-Purpose Timers Register Manual 2580
16.3.1 GP Timer Register Map 2580
16.3.1.1 Instance Summary 2580
16.3.2 GP Timer Register Mapping Summary 2580
16.3.3 GP Timer Register Descriptions 2583
16.4 Watchdog Timers 2602
16.4.1 WDTs Overview 2602
16.4.1.1 WDT Features 2602
16.4.2 WDT Integration 2603
16.4.2.1 Clocking, Reset, and Power-Management Scheme 2603
16.4.2.1.1 Clock Management 2603
16.4.2.1.2 Reset and Power Management 2605
16.4.2.2 Interrupts 2606
16.4.3 WDTs Functional Description 2606
16.4.3.1 General WDT Operation 2606
16.4.3.2 Reset Context 2606
16.4.3.3 Overflow/Reset Generation 2607
16.4.3.4 Prescaler Value/Timer Reset Frequency 2607
16.4.3.5 Triggering a Timer Reload 2608
16.4.3.6 Start/Stop Sequence for WDTs (Using WDTi.WSPR Register) 2609
16.4.3.7 Modifying Timer Count/Load Values and Prescaler Setting 2609
16.4.3.8 Watchdog Counter Register Access Restriction (WDTi.WCRR Register) 2609
16.4.3.9 WDT Interrupt Generation 2609
16.4.3.10 WDT Under Emulation 2610
16.5 Watchdog Timer Register Manual 2611
16.5.1 Instance Summary 2611
16.5.2 WDT Register Mapping Summary 2611
16.5.3 WDT Register Descriptions 2612
16.6 32-kHz Synchronized Timer 2619

16.6.1 32-kHz Sync Timer Functional Description ... 2619
16.6.1.1 Reading the 32-kHz Sync Timer ... 2619
16.6.1.2 32-kHz Sync Timer Features ... 2619
16.6.2 32-kHz Sync Timer Environment ... 2619
16.6.3 32-kHz Sync Timer Integration ... 2620
16.6.3.1 Clocking, Reset, and Power-Management Scheme ... 2620
16.6.3.2 Interrupts ... 2620
16.6.3.3 Sync Timer 32k and MSuspend Signal ... 2620
16.7 32-kHz Sync Timer Register Manual ... 2621
16.7.1 32-kHz Sync Timer Instance Summary ... 2621
16.7.2 32-kHz Sync Timer Register Mapping Summary ... 2621
16.7.3 32-kHz Sync Timer Register Descriptions ... 2621

17 UART/IrDA/CIR ... 2623

17.1 UART/IrDA/CIR Overview ... 2624
17.1.1 UART Features ... 2624
17.1.2 IrDA Features ... 2625
17.1.3 CIR Features ... 2626
17.2 UART/IrDA/CIR Environment ... 2627
17.2.1 System Using UART Communication with Hardware Handshake ... 2627
17.2.2 System Using IrDA Communication Protocol ... 2627
17.2.3 System Using CIR Communication Protocol with Remote Control ... 2628
17.2.4 UART Interface Description ... 2628
17.2.4.1 UART Interface Description ... 2628
17.2.4.2 UART Protocol and Data Format ... 2628
17.2.5 IrDA Functional Interfaces ... 2629
17.2.5.1 UART3 Interface Description ... 2629
17.2.5.2 IrDA Protocol and Data Format ... 2629
17.2.5.2.1 SIR Mode ... 2629
17.2.5.2.2 SIR Free Format Mode ... 2632
17.2.5.2.3 MIR Mode ... 2633
17.2.5.2.4 FIR Mode ... 2634
17.2.6 CIR Functional Interfaces ... 2636
17.2.6.1 CIR Interface Description ... 2636
17.2.6.2 CIR Protocol and Data Format ... 2636
17.2.6.2.1 Carrier Modulation ... 2636
17.2.6.2.2 Pulse Duty Cycle ... 2637
17.2.6.2.3 Consumer IR Encoding/Decoding ... 2637
17.3 UART/IrDA/CIR Integration ... 2640
17.3.1 Clocking, Reset, and Power-Management Scheme ... 2640
17.3.1.1 Clocking ... 2640
17.3.1.2 Hardware Reset ... 2641
17.3.1.3 Software Reset ... 2641
17.3.1.4 Power Domain ... 2641
17.3.2 Hardware Requests ... 2642
17.3.2.1 Interrupts ... 2642
17.3.2.2 DMA Requests ... 2642
17.3.2.3 Wake-up Request ... 2642
17.4 UART/IrDA/CIR Functional Description ... 2644

17.4.1 UART/IrDA/CIR Block Description 2644
17.4.2 FIFO Management 2645
17.4.2.1 FIFO Trigger 2646
17.4.2.1.1 Transmit FIFO Trigger 2646
17.4.2.1.2 Receive FIFO Trigger 2646
17.4.2.2 FIFO Interrupt Mode 2647
17.4.2.3 FIFO Polled Mode Operation 2648
17.4.2.4 FIFO DMA Mode Operation 2648
17.4.2.4.1 DMA Transfers (DMA Mode 1, 2, or 3) 2649
17.4.2.4.2 DMA Transmission 2652
17.4.2.4.3 DMA Reception 2652
17.4.3 Mode Selection 2653
17.4.3.1 Register Access Modes 2653
17.4.3.1.1 Operational Mode and Configuration Modes 2653
17.4.3.1.2 Register Access Submode 2653
17.4.3.1.3 Registers Available for the Register Access Modes 2654
17.4.3.2 UART/IrDA (SIR, MIR, FIR)/CIR Mode Selection 2655
17.4.3.2.1 Registers Available for the UART Function 2655
17.4.3.2.2 Registers Available for the IrDA Function (UART3 Only) 2656
17.4.3.2.3 Registers Available for the CIR Function (UART3 Only) 2657
17.4.4 Protocol Formatting 2658
17.4.4.1 UART Mode 2658
17.4.4.1.1 UART Clock Generation: Baud Rate Generation 2658
17.4.4.1.2 Choosing the Appropriate Divisor Value 2658
17.4.4.1.3 UART Data Formatting 2659
17.4.4.1.4 UART Mode Interrupt Management 2663
17.4.4.2 IrDA Mode (UART3 Only) 2664
17.4.4.2.1 IrDA Clock Generation: Baud Generator 2664
17.4.4.2.2 Choosing the Appropriate Divisor Value 2665
17.4.4.2.3 IrDA Data Formatting 2666
17.4.4.2.4 SIR Mode DATA Formatting 2667
17.4.4.2.5 MIR and FIR Mode Data Formatting 2668
17.4.4.2.6 IrDA Mode Interrupt Management 2668
17.4.4.3 CIR Mode (UART3 Only) 2669
17.4.4.3.1 CIR Mode Clock Generation 2669
17.4.4.3.2 CIR Data Formatting 2671
17.4.4.3.3 CIR Mode Interrupt Management 2671
17.4.5 Power Management 2672
17.4.5.1 UART Mode Power Management 2672
17.4.5.1.1 Module Power Saving 2672
17.4.5.1.2 System Power Saving 2673
17.4.5.2 IrDA Mode Power Management (UART3 Only) 2673
17.4.5.2.1 Module Power Saving 2673
17.4.5.2.2 System Power Saving 2673
17.4.5.3 CIR Mode Power Management (UART3 Only) 2673
17.4.5.3.1 Module Power Saving 2673
17.4.5.3.2 System Power Saving 2673
17.5 UART/IrDA/CIR Basic Programming Model 2674
17.5.1 UART Programming Model 2674

17.5.1.1 Quick start ... 2674
17.5.1.1.1 Software Reset ... 2674
17.5.1.1.2 FIFOs and DMA Settings ... 2674
17.5.1.1.3 Protocol, Baud Rate, and Interrupt Settings ... 2675
17.5.1.2 Hardware and Software Flow Control Configuration ... 2676
17.5.1.2.1 Hardware Flow Control Configuration ... 2677
17.5.1.2.2 Software Flow Control Configuration ... 2677
17.5.2 IrDA Programming Model (UART3 Only) ... 2678
17.5.2.1 SIR Mode ... 2678
17.5.2.1.1 Receive ... 2679
17.5.2.1.2 Transmit ... 2679
17.5.2.2 MIR Mode ... 2680
17.5.2.2.1 Receive ... 2680
17.5.2.2.2 Transmit ... 2680
17.5.2.2.3 FIR Mode ... 2681
17.6 UART/IrDA/CIR Register Manual ... 2682
17.6.1 UART/IrDA/CIR Instance Summary ... 2682
17.6.2 UART/IrDA/CIR Register Summary ... 2682
17.6.3 UART/IrDA/CIR Register Description ... 2684
18 I^2C ... 2723
18.1 High-Speed I^2C Controller Overview ... 2724
18.2 High-Speed I^2C Controller Environment ... 2726
18.2.1 Multimaster HS I^2C Controllers in I^2C Mode ... 2726
18.2.1.1 Multimaster HS I^2C Controller Pins for Typical Connections in I^2C Mode ... 2726
18.2.1.2 I^2C Interface Typical Connections ... 2726
18.2.1.3 I^2C Typical Connection Protocol and Data Format ... 2726
18.2.1.3.1 Serial Data Format ... 2727
18.2.1.3.2 Data Validity ... 2727
18.2.1.3.3 Start and Stop Conditions ... 2727
18.2.1.3.4 Addressing ... 2728
18.2.1.3.5 Master Transmitter ... 2729
18.2.1.3.6 Master Receiver ... 2729
18.2.1.3.7 Slave Transmitter ... 2729
18.2.1.3.8 Slave Receiver ... 2729
18.2.1.3.9 Bus Arbitration ... 2729
18.2.1.3.10 I^2C Clock Generation and Synchronization ... 2730
18.2.2 Multimaster High-Speed I^2C Controllers in SCCB Mode ... 2730
18.2.2.1 Multimaster HS I^2C Controller Pins for Typical Connections in SCCB Mode ... 2732
18.2.2.2 SCCB Interface Typical Connections ... 2732
18.2.2.3 SCCB Typical Connection Protocol and Data Format ... 2733
18.2.2.3.1 Serial Transmission Timing Diagram ... 2733
18.2.2.3.2 SCCB Transmission Data Formats ... 2733
18.2.3 High-Speed I^2C Controller for Communication With Power Chip(s) ... 2734
18.2.3.1 HS I^2C Controller I2C4 Pins for Typical Connections ... 2735
18.2.3.2 HS I^2C Controller I2C4 Interface Typical Connections ... 2735
18.2.3.3 I^2C Typical Connections Protocol and Data Format for I2C4 ... 2736
18.2.3.3.1 Serial Data Format ... 2736
18.2.3.3.2 Data Validity ... 2736

18.2.3.3.3 S and P Conditions 2736
18.2.3.3.4 Addressing 2736
18.3 High-Speed I2C Controller Integration 2738
18.3.1 Clocking, Reset, and Power-Management Scheme 2738
18.3.1.1 Clocks 2738
18.3.1.1.1 Module Clocks 2738
18.3.1.2 Power Management 2739
18.3.1.2.1 Module Power Saving 2739
18.3.1.2.2 System Power Management 2739
18.3.1.2.3 Wake-Up Capability 2741
18.3.1.3 Resets 2742
18.3.1.3.1 Hardware Reset 2742
18.3.1.3.2 Software Reset 2742
18.3.1.4 Power Domain 2743
18.3.2 Hardware Requests 2743
18.3.2.1 DMA Requests 2743
18.3.2.2 Interrupt Requests 2743
18.4 High-Speed I2C Controller Functional Description 2746
18.4.1 Block Diagram 2746
18.4.2 Transmit Mode in I2C Mode 2746
18.4.3 Receive Mode in I2C Mode 2747
18.4.4 FIFO Management 2747
18.4.4.1 FIFO Interrupt Mode Operation 2747
18.4.4.2 FIFO Polling Mode Operation 2749
18.4.4.3 FIFO DMA Mode Operation (I2C Mode Only) 2749
18.4.4.4 Draining Feature (I2C Mode Only) 2750
18.4.5 Programmable Multislave Channel Feature (I2C Mode Only) 2751
18.4.6 Automatic Blocking of the I2C Clock Feature (I2C Mode Only) 2751
18.4.7 Clocking 2752
18.4.8 Noise Filter 2753
18.4.9 System Test Mode 2753
18.4.10 Write and Read Operations in SCCB Mode 2754
18.4.11 Power Chip Communication Operations 2754
18.5 High-Speed I2C Controller Basic Programming Model 2755
18.5.1 Multimaster HS I2C Controller Basic Programming Model in I2C Mode 2755
18.5.1.1 Main Program 2755
18.5.1.1.1 Configure the Module Before Enabling the I2C Controller 2755
18.5.1.1.2 Initialize the I2C Controller 2755
18.5.1.1.3 Configure Slave Address and the Data Control Register 2756
18.5.1.1.4 Initiate a Transfer 2756
18.5.1.1.5 Receive Data 2756
18.5.1.1.6 Transmit Data 2756
18.5.1.2 Interrupt Subroutine Sequence 2757
18.5.1.3 Programming Flow Diagrams 2758
18.5.2 High-Speed I2C Controller Basic Programming Model in SCCB Mode 2766
18.5.2.1 Main Program 2766
18.5.2.1.1 Configure the Module Before Enabling the I2C Controller 2766
18.5.2.1.2 Initialize the I2C Controller 2767
18.5.2.1.3 Initiate a Transfer 2767

18.5.2.1.4 Receive Data 2767
18.5.2.1.5 Transmit Data 2767
18.5.2.2 Interrupt Subroutine Sequence 2767
18.5.2.3 Programming Flow Diagrams 2768
18.5.3 Master Transmitter HS I2C Controller I2C4 Basic Programming Model for Communication With Power Chips 2773
18.5.3.1 Configure the Voltage Controller Registers 2773
18.5.3.2 Configure the Master Transmitter HS I2C Controller I2C4 2773
18.5.3.3 Configure the External Power Chip(s) 2773
18.6 High-Speed Multimaster I2C Use Cases and Tips 2774
18.6.1 Camcorder Use Case: How to Configure the I2C Modules When Connecting TWL4030 Companion chip and VS6650 Digital Camera Device 2774
18.6.1.1 Overview 2774
18.6.1.2 Environment 2774
18.6.1.2.1 Data Transfer Formats 2775
18.6.1.3 Programming Flow 2776
18.6.1.3.1 Initial Configuration 2776
18.6.1.3.2 Operational Mode 2778
18.6.1.3.3 Flowcharts 2780
18.7 High-Speed I2C Controllers Register Manual 2785
18.7.1 Multimaster HS I2C Controller Register Mapping Summary 2785
18.7.2 Register Description 2786

19 Multichannel SPI 2807

19.1 McSPI Overview 2808
19.2 McSPI Environment 2811
19.2.1 SPI Interface in Master Mode 2812
19.2.2 SPI Interface in Slave Mode 2812
19.3 McSPI Functional Interface 2814
19.3.1 Basic McSPI Pins for Master Mode 2814
19.3.2 Basic McSPI Pins for Slave Mode 2814
19.3.3 Multichannel SPI Protocol and Data Format 2815
19.3.3.1 Transfer Format 2816
19.4 McSPI Integration 2819
19.4.1 McSPI Description 2819
19.4.2 Clocking, Reset, and Power-Management Scheme 2819
19.4.2.1 Clocking 2819
19.4.2.2 Power Domain 2820
19.4.2.3 Hardware Reset 2820
19.4.2.4 Software Reset 2820
19.4.3 Hardware Requests 2821
19.4.3.1 DMA Requests 2821
19.4.3.2 Interrupt Requests 2822
19.4.3.3 Wake-Up Requests 2822
19.5 McSPI Functional Description 2823
19.5.1 McSPI Block Diagram 2823
19.5.2 Master Mode 2823
19.5.2.1 Master Mode Features 2823
19.5.2.2 Master Transmit-and-Receive Mode (Full Duplex) 2824
19.5.2.3 Master Transmit-Only Mode (Half Duplex) 2825

19.5.2.4 Master Receive-Only Mode (Half Duplex) 2825
19.5.2.5 Single-Channel Master Mode 2826
19.5.2.5.1 Programming Tips When Switching to Another Channel 2826
19.5.2.5.2 Force spim_csx Mode 2826
19.5.2.5.3 Turbo Mode 2827
19.5.2.6 Start Bit Mode 2828
19.5.2.7 Chip-Select Timing Control 2828
19.5.2.8 Programmable SPI Clock (spim_clk) 2829
19.5.2.8.1 Clock Ratio Granularity 2829
19.5.3 Slave Mode 2830
19.5.3.1 Dedicated Resources 2830
19.5.3.2 Slave Transmit-and-Receive Mode 2832
19.5.3.3 Slave Transmit-Only Mode 2832
19.5.3.4 Slave Receive-Only Mode 2833
19.5.4 FIFO Buffer Management 2834
19.5.4.1 Buffer Almost Full 2835
19.5.4.2 Buffer Almost Empty 2836
19.5.4.3 End of Transfer Management 2837
19.5.5 Interrupts 2837
19.5.5.1 Interrupt Events in Master Mode 2837
19.5.5.1.1 TXx_EMPTY 2837
19.5.5.1.2 TXx_UNDERFLOW 2838
19.5.5.1.3 RXx_ FULL 2838
19.5.5.1.4 End Of Word Count 2838
19.5.5.2 Interrupt Events in Slave Mode 2838
19.5.5.2.1 TXx_EMPTY 2838
19.5.5.2.2 TXx_UNDERFLOW 2839
19.5.5.2.3 RXx_FULL 2839
19.5.5.2.4 RX0_OVERFLOW 2839
19.5.5.2.5 End Of Word Count 2839
19.5.5.3 Interrupt-Driven Operation 2839
19.5.5.4 Polling 2840
19.5.6 DMA Requests 2840
19.5.7 Power Saving Management 2840
19.5.7.1 Normal Mode 2840
19.5.7.2 Idle Mode 2841
19.5.7.2.1 Wake-Up Event in Smart-Idle Mode 2842
19.5.7.2.2 Transitions From Smart-Idle Mode to Normal Mode 2842
19.5.7.2.3 Force-Idle Mode 2843
19.6 McSPI Basic Programming Model 2844
19.6.1 Initialization of Modules 2844
19.6.2 Transfer Procedures without FIFO 2844
19.6.2.1 Common Transfer Procedure 2845
19.6.2.2 End-of-Transfer Procedure 2845
19.6.2.3 Transmit and Receive Procedure 2847
19.6.2.4 Transmit-Only Procedure 2848
19.6.2.4.1 Based on Interrupt Requests 2848
19.6.2.4.2 Transmit-Only Based on DMA Write Requests 2848

19.6.2.5 Receive-Only Procedure 2849
19.6.2.5.1 Master Normal Receive-Only Procedure 2849
19.6.2.5.2 Master Turbo Receive-Only Procedure 2851
19.6.2.5.3 Slave Receive-Only Procedure 2853
19.6.2.6 McSPI Configuration and Operations Example 2855
19.6.2.6.1 McSPI Initialization Sequence 2855
19.6.2.6.2 Operations for the First Slave (On Channel 0) 2855
19.6.2.6.3 Programming in Interrupt Mode 2856
19.6.2.6.4 Operations for the Second Slave (on Channel 1) in Polling Mode 2856
19.6.3 Transfer Procedures with FIFO 2857
19.6.3.1 Common Transfer Procedure 2857
19.6.3.2 Transmit-Receive Procedure With Word Count (WCNT≠0) 2860
19.6.3.3 Transmit-Receive Procedure Without Word Count (WCNT=0) 2861
19.6.3.4 Transmit-Only Procedure 2862
19.6.3.5 Receive-Only Procedure With Word Count (WCNT≠0) 2863
19.6.3.6 Receive-Only Procedure Without Word Count (WCNT=0) 2864
19.7 McSPI Use Cases and Tips 2865
19.7.1 How to Configure the McSPI Interface When Connected with an EPSON VGA FlatLink™ 3G Device 2865
19.7.1.1 Overview 2865
19.7.1.2 Environment 2865
19.7.1.3 Data Path 2866
19.7.1.4 Programming Flow 2867
19.7.1.4.1 McSPI Module Configuration 2868
19.7.1.4.2 'SOFT RESET', 'SLEEP OUT' and 'DISPLAY ON' Commands 2869
19.7.1.4.3 'READ DISPLAY STATUS' Command 2869
19.8 McSPI Register Manual 2871
19.8.1 McSPI Instance Summary 2871
19.8.2 McSPI Register Summary 2871
19.8.3 McSPI Register Description 2872

20 HDQ/1-Wire 2893

20.1 HDQ/1-Wire Overview 2894
20.2 HDQ/1-Wire Environment 2895
20.2.1 HDQ/1-Wire Functional Interface 2895
20.2.2 HDQ and 1-Wire (SDQ) Protocols 2895
20.2.2.1 HDQ Protocol Initialization (Default) 2895
20.2.2.2 1-Wire (SDQ) Protocol Initialization 2896
20.2.2.3 Communication Sequence (HDQ and 1-Wire Protocols) 2896
20.3 HDQ/1-Wire Integration 2898
20.3.1 Clocking, Reset, and Power Management Scheme 2898
20.3.1.1 HDQ/1-Wire Clocks 2898
20.3.1.2 HDQ/1-Wire Reset Scheme 2898
20.3.1.3 HDQ/1-Wire Power Domain 2899
20.3.2 Hardware Requests 2899
20.4 HDQ/1-Wire Functional Description 2900
20.4.1 HDQ/1-Wire Block Diagram 2900
20.4.2 HDQ Mode (Default) 2901
20.4.2.1 HDQ Mode Features 2901
20.4.2.2 Description 2901

20.4.2.3 Single-Bit Mode 2902
20.4.2.4 Interrupt Conditions 2902
20.4.3 1-Wire Mode 2903
20.4.3.1 1-Wire Mode Features 2903
20.4.3.2 Description 2903
20.4.3.3 1-Wire Single-Bit Mode Operation 2903
20.4.3.4 Interrupt Conditions 2904
20.4.3.5 Status Flags 2904
20.4.4 Module Power Saving 2904
20.4.4.1 Autoidle Mode 2904
20.4.4.2 Power-Down Mode 2904
20.4.5 System Power Management and Wakeup 2904
20.5 HDQ/1-Wire Basic Programming Model 2906
20.5.1 Module Initialization Sequence 2906
20.5.1.1 Mode Selection 2906
20.5.1.2 Reset/Initialization 2906
20.5.2 HDQ Protocol Basic Programming Model 2906
20.5.2.1 Write Operation 2906
20.5.2.2 Read Operation 2907
20.5.3 1-Wire Mode (SDQ) Basic Programming Model 2907
20.5.3.1 Write Operation 2907
20.5.3.2 Read Operation 2908
20.5.3.3 1-Wire Bit Mode Operation 2908
20.5.4 Power Management 2908
20.5.4.1 Module Power-Down Mode 2909
20.5.4.2 System Idle Mode 2909
20.6 HDQ/1-Wire Use Cases and Tips 2909
20.6.1 How to Configure the HDQ/1-Wire when Connected with a BQ27000 Gauge 2910
20.6.1.1 Environment 2910
20.6.1.2 Programming Flow 2910
20.6.1.3 Pad Configuration and HDQ/1-Wire Clock and Power Management 2910
20.6.1.4 HDQ/1-Wire Software Reset 2911
20.6.1.5 Interrupts Enable 2911
20.6.1.6 Read and Write Operations 2912
20.7 HDQ/1-Wire Register Manual 2913
20.7.1 HDQ/1-Wire Instance Summary 2913
20.7.2 HDQ/1-Wire Register Mapping Summary 2913
20.7.3 HDQ/1-Wire Register Description 2914

21 Multi-Channel Buffered Serial Port 2919

21.1 McBSP Overview 2920
21.1.1 McBSP Features 2920
21.1.2 SIDETONE Core 2921
21.2 McBSP Environment 2923
21.2.1 McBSP Functions 2923
21.2.2 McBSP Signals Descriptions 2923
21.2.3 McBSP Functions Description 2924
21.2.3.1 McBSP Modes 2924
21.2.3.2 McBSP Functions 2926

21.2.3.2.1 McBSP Function 1: Control and Data 2926
21.2.3.2.2 McBSP Function 2: Audio Data 2926
21.2.3.2.3 McBSP Function 3: Voice Data 2926
21.2.4 McBSP Protocols and Data Formats 2927
21.2.4.1 Words, Frames, and Phases Definitions 2927
21.2.4.1.1 Words or Channels 2927
21.2.4.1.2 Frames 2927
21.2.4.1.3 Phases 2928
21.2.4.2 Serial Protocol and Data Formats 2928
21.2.4.2.1 Protocol 2928
21.2.4.2.2 Data Format 2928
21.2.4.3 Audio Protocol and Data Formats 2929
21.2.4.3.1 Protocol 2929
21.2.4.3.2 Data Formats 2929
21.2.4.4 Voice Protocol and Data Formats 2931
21.2.4.4.1 Protocol 2931
21.2.4.4.2 Data Formats 2931
21.3 McBSP Integration 2932
21.3.1 Signal Source Control 2936
21.3.1.1 McBSP1 Module (6 Pins Configuration) 2936
21.3.1.2 McBSP2, 3, 4, and 5 modules (4 pins configuration) 2937
21.3.2 Clocking, Reset, and Power Management Scheme 2937
21.3.2.1 Power Domain 2937
21.3.2.2 Clocks 2937
21.3.2.2.1 McBSP1 Clocks 2937
21.3.2.2.2 McBSP2 Clocks 2938
21.3.2.2.3 McBSP3 Clocks 2939
21.3.2.2.4 McBSP4 Clocks 2940
21.3.2.2.5 McBSP5 Clocks 2941
21.3.2.2.6 SIDETONE Clock 2942
21.3.2.3 Hardware and Software Reset 2943
21.3.2.4 Power Management 2943
21.3.2.4.1 McBSP Operating States 2943
21.3.2.4.2 McBSP Acknowledgment Modes 2943
21.3.2.4.3 Wake-Up Capability 2945
21.3.2.4.4 Analysis of the Receiver Smart Idle Behavior 2946
21.3.3 Hardware Requests 2949
21.3.3.1 DMA Requests 2949
21.3.3.2 Interrupt Requests 2949
21.3.3.2.1 McBSP Interrupt Requests 2949
21.3.3.2.2 SIDETONE_McBSP Interrupt Requests 2951
21.4 McBSP Functional Description 2953
21.4.1 Block Diagram 2953
21.4.2 McBSP Data Transfer Process 2955
21.4.2.1 Data Transfer Process for 8- / 12- / 16- / 20- / 24- / 32-bits Long Words 2956
21.4.2.2 Bit Reordering (Option to Transfer LSB First) 2957
21.4.2.3 Clocking and Framing Data 2957
21.4.2.3.1 Clocking 2957
21.4.2.3.2 Serial Words 2958

21.4.2.3.3 Frames and Frame Synchronization 2959
21.4.2.3.4 Detecting Frame-Synchronization Pulses, Even in Reset State 2959
21.4.2.3.5 Ignoring Frame-Synchronization Pulses 2960
21.4.2.3.6 Frame Frequency 2960
21.4.2.3.7 Maximum Frame Frequency 2960
21.4.2.4 Frame Phases (Dual-Phase Frame I2S Support) 2961
21.4.2.4.1 Number of Phases, Words, and Bits per Frame 2961
21.4.2.4.2 Single-Phase Frame Example 2961
21.4.2.4.3 Dual-Phase Frame Example 2962
21.4.2.5 McBSP Reception 2963
21.4.2.6 McBSP Transmission 2964
21.4.2.7 Enable/Disable the Transmit and Receive Processes 2964
21.4.2.8 MCBSP Data Transfer Mode 2965
21.4.2.8.1 Transmit Full Cycle Mode 2965
21.4.2.8.2 Transmit Half Cycle Mode 2966
21.4.2.8.3 Receive Full Cycle Mode 2966
21.4.2.8.4 Receive Half Cycle Mode 2966
21.4.3 McBSP SRG 2967
21.4.3.1 Clock Generation in the SRG 2968
21.4.3.2 Frame Sync Generation in the SRG 2969
21.4.3.2.1 Choosing the Width of the Frame-sync Pulse 2969
21.4.3.2.2 Controlling the Period Between the Starting Edges of Frame Sync Pulses 2969
21.4.3.2.3 Keeping FSG Synchronized to an External Clock 2970
21.4.3.3 Synchronizing SRG Outputs to an External Clock 2970
21.4.3.3.1 Operating the Transmitter Synchronously with the Receiver 2970
21.4.3.3.2 Synchronization Examples 2970
21.4.4 McBSP Exception/Error Conditions 2971
21.4.4.1 Introduction 2971
21.4.4.2 Overrun in the Receiver 2972
21.4.4.3 Unexpected Receive Frame-sync Pulse 2973
21.4.4.3.1 Possible Responses to Receive Frame-sync Pulses 2973
21.4.4.3.2 Example of an Unexpected Receive Frame-sync Pulse 2973
21.4.4.3.3 Preventing Unexpected Receive Frame-sync Pulses 2974
21.4.4.4 Underflow in the Receiver 2974
21.4.4.5 Underflow in the Transmitter 2974
21.4.4.6 Unexpected Transmit Frame-sync Pulse 2975
21.4.4.6.1 Possible Responses to Transmit Frame-sync Pulses 2975
21.4.4.6.2 Example of Unexpected Transmit Frame-Synchronization Pulse 2975
21.4.4.6.3 Preventing Unexpected Transmit Frame-sync Pulses 2975
21.4.4.7 Overflow in the Transmitter 2976
21.4.5 McBSP DMA Configuration 2976
21.4.6 Multichannel Selection Modes 2977
21.4.6.1 Channels, Blocks, and Partitions 2977
21.4.6.2 Multichannel Selection 2977
21.4.6.3 Configuring a Frame for Multichannel Selection 2977
21.4.6.4 Using Eight Partitions 2978
21.4.6.5 Receive Multichannel Selection Mode 2979
21.4.6.6 Using Two Partitions (Legacy Only) 2979

21.4.6.7 Transmit Multichannel Selection Modes 2980
21.4.6.7.1 Disabling/Enabling Versus Masking/Unmasking 2981
21.4.6.7.2 Activity on McBSP Pins for Different Values of XMCM 2981
21.4.7 SIDETONE Mode (ALP) 2983
21.4.7.1 Introduction 2983
21.4.7.2 SIDETONE Interface 2983
21.4.7.3 Data Processing Path 2985
21.4.7.4 Data Processing 2986
21.4.7.4.1 Filtering 2986
21.4.7.4.2 Applying Gain 2987
21.4.7.4.3 Enabling SIDETONE 2987
21.4.7.4.4 FIR Accuracy 2987
21.4.7.5 Interrupt Operation 2987
21.5 McBSP Basic Programming Model 2988
21.5.1 McBSP Core 2988
21.5.1.1 McBSP Initialization Procedure 2988
21.5.1.2 Reset and Initialization Procedure for the Sample Rate Generator 2991
21.5.1.3 Data Transfer DMA Request Configuration 2994
21.5.1.4 Interrupt Configuration 2994
21.5.1.4.1 L4-Compliant Interrupt Line 2994
21.5.1.4.2 Legacy Interrupt Line 2995
21.5.1.5 Receiver Configuration 2996
21.5.1.5.1 Resetting (Step 1) and Enabling (Step 3) the Receiver 2996
21.5.1.5.2 Programming the McBSP Registers for the Desired Receiver Configuration (Step 2) 2997
21.5.1.6 Transmitter Configuration 3005
21.5.1.6.1 Resetting (Step 1) and Enabling (Step 3) the Transmitter 3005
21.5.1.6.2 Programming the McBSP Registers for the Desired Transmitter Operation (Step 2) 3006
21.5.1.7 General-Purpose I/O on the McBSP Pins (Legacy Only) 3012
21.5.1.8 Data Packing Examples 3013
21.5.1.8.1 Data Packing Using Frame Length and Word Length 3013
21.5.1.8.2 Data Packing Using Word Length and the Frame-Sync Ignore Function 3014
21.5.2 SIDETONE Feature 3015
21.5.2.1 SIDETONE Activation Procedure 3015
21.5.2.2 SIDETONE Initialization Procedure 3016
21.5.2.3 SIDETONE FIR Coefficients Writing 3016
21.5.2.4 SIDETONE FIR Coefficients Reading 3016
21.6 McBSP Use Case and Tips 3017
21.6.1 Camcorder Use Case: How to Configure the McBSP2 to Receive Voice Stream From the TWL4030 Device 3017
21.6.1.1 Overview 3017
21.6.1.2 Programming Flow 3018
21.7 McBSP Register Manual 3020
21.7.1 McBSP Register Mapping Summary 3020
21.7.2 SIDETONE Register Mapping Summary 3025
21.7.3 McBSP Register Description 3026
21.7.4 SIDETONE Register Description 3068

22 MMC/SD/SDIO Card Interface 3075

22.1 MMC/SD/SDIO Overview 3076
22.1.1 MMC/SD/SDIO Features 3078

22.2 MMC/SD/SDIO Environment ... 3080
22.2.1 MMC/SD/SDIO Connected to an MMC, an SD, or an SDIO Card ... 3080
22.2.2 MMC/SD/SDIO Connected to an MMC, an SD, or an SDIO Card Through an External Transceiver Device ... 3080
22.2.3 MMC/SD/SDIO Functional Interfaces ... 3081
22.2.3.1 Basic MMC/SD/SDIOi Pins Without External Transceiver ... 3081
22.2.3.2 Basic MMC/SD/SDIO2 Pins with External Transceiver ... 3082
22.2.3.3 MMC/SD/SDIO Protocol and Data Format ... 3082
22.2.3.3.1 Protocol ... 3083
22.2.3.3.2 Data Format ... 3085
22.3 MMC/SD/SDIO Integration ... 3088
22.3.1 Clocking, Reset, and Power-Management Scheme ... 3088
22.3.1.1 Clocks ... 3088
22.3.1.1.1 Module Clocks ... 3088
22.3.1.1.2 Power Management ... 3089
22.3.1.2 Resets ... 3091
22.3.1.2.1 Hardware Reset ... 3091
22.3.1.2.2 Software Reset ... 3091
22.3.1.3 Power Domain ... 3092
22.3.2 Hardware Requests ... 3092
22.3.2.1 DMA Requests ... 3092
22.3.2.1.1 DMA Receive Mode ... 3092
22.3.2.1.2 DMA Transmit Mode ... 3093
22.3.2.2 Interrupt Requests ... 3094
22.3.2.2.1 Interrupt-Driven Operation ... 3095
22.3.2.2.2 Polling ... 3095
22.4 MMC/SD/SDIO Functional Description ... 3096
22.4.1 Description ... 3096
22.4.2 Mode Selection ... 3099
22.4.3 Buffer Management ... 3099
22.4.3.1 Data Buffer ... 3099
22.4.3.1.1 Data Buffer Status ... 3101
22.4.4 Transfer Process ... 3102
22.4.4.1 Different Types of Commands ... 3102
22.4.4.2 Different Types of Responses ... 3102
22.4.5 Transfer or Command Status and Errors Reporting ... 3102
22.4.6 Transfer Stop ... 3103
22.4.7 MMC CE-ATA Command Completion Disable Management ... 3104
22.5 MMC/SD/SDIO Basic Programming Model ... 3105
22.5.1 MMC/SD/SDIO Host Controller Initialization Flow ... 3105
22.5.1.1 Enable Interface and Functional clock for MMC Controller ... 3105
22.5.1.2 MMCHS Soft Reset Flow ... 3106
22.5.1.3 Set MMCHS Default Capabilities ... 3106
22.5.1.4 Wake-Up Configuration ... 3107
22.5.1.5 MMC Host and Bus Configuration ... 3108
22.5.2 Basic Operations for MMC/SD/SDIO Host Controller ... 3109
22.5.2.1 Card Detection, Identification, and Selection ... 3109
22.5.2.2 Read/Write Transfer Flow in DMA Mode with Interrupt ... 3111
22.5.2.3 Read/Write Transfer Flow in DMA Mode with Polling ... 3112

22.5.2.4 Read/Write Transfer Flow without DMA with Polling ... 3113
22.5.2.5 Read/Write Transfer Flow in CE-ATA Mode ... 3114
22.5.2.6 Suspend-Resume Flow ... 3114
22.5.2.6.1 Suspend Flow ... 3115
22.5.2.6.2 Resume Flow ... 3116
22.5.2.7 Basic Operations - Steps Detailed ... 3116
22.5.2.7.1 Command Transfer Flow ... 3117
22.5.2.7.2 MMCHS Clock Frequency Change ... 3119
22.5.3 MMC/SD/SDIO1 Bus Voltage Selection ... 3119
22.6 MMC/SD/SDIO Use Cases and Tips ... 3120
22.6.1 MMCHS Controller Usage ... 3120
22.6.1.1 Overview ... 3120
22.6.1.2 Environment ... 3121
22.6.1.2.1 Command and Data Transfer Formats ... 3121
22.6.1.3 Programming Flow ... 3122
22.6.1.3.1 Initial Configuration ... 3123
22.6.1.3.2 MMC Card Identification ... 3125
22.6.1.3.3 MMC Bus Setting Change After Card Identification ... 3127
22.6.1.3.4 Reading the CSD Register of a MMC Card ... 3128
22.6.1.3.5 MMC Write Transfer ... 3130
22.6.1.3.6 MMC Read Transfer ... 3132
22.6.1.3.7 Dealing with High Capacity Cards ... 3133
22.7 MMC/SD/SDIO Register Manual ... 3134
22.7.1 MMC/SD/SDIO Instance Summary ... 3134
22.7.2 MMC/SD/SDIO Registers Mapping Summary ... 3134

23 Memory Stick PRO Host Controller ... 3173
23.1 Memory Stick PRO Host Controller Overview ... 3174
23.1.1 Main Features ... 3174

24 High-Speed USB Host Subsystem and High-Speed USB OTG Controller ... 3175
24.1 High-Speed USB OTG Controller ... 3177
24.1.1 High-Speed USB OTG Controller Overview ... 3177
24.1.1.1 Main Features ... 3178
24.1.2 High-Speed USB OTG Controller Environment ... 3179
24.1.2.1 High-Speed USB Controller Functional Interfaces ... 3180
24.1.2.1.1 Basic High-Speed USB Controller Pins ... 3180
24.1.2.1.2 High-Speed USB Controller Interface Description ... 3180
24.1.3 High-Speed USB OTG Controller Integration ... 3181
24.1.3.1 Clocking, Reset, and Power-Management Scheme ... 3181
24.1.3.1.1 Clocks ... 3181
24.1.3.1.2 Resets ... 3182
24.1.3.1.3 Power-Management Scheme ... 3182
24.1.3.1.4 Power Domain ... 3185
24.1.3.2 Hardware Requests ... 3185
24.1.3.2.1 Interrupt Requests ... 3185
24.1.3.2.2 IDLE Handshake Protocol ... 3186
24.1.3.2.3 MSTANDBY Handshake Protocol ... 3186
24.1.3.2.4 Wake-Up Request ... 3186
24.1.4 High-Speed USB OTG Controller Functional Description ... 3187

24.1.4.1 Inventra™ MUSBMHDRC 3187
24.1.4.2 Configuration 3188
24.1.4.3 Basic Operation 3188
24.1.4.3.1 Module Initialization 3188
24.1.4.3.2 Transaction Handling 3189
24.1.4.4 Optional Features 3189
24.1.4.4.1 Double Packet Buffering 3189
24.1.4.4.2 DMA 3189
24.1.4.5 Automatic Packet Splitting/Combining for Bulk Transfers 3190
24.1.4.6 High-Bandwidth Isochronous Endpoints 3191
24.1.5 High-Speed USB OTG Controller Basic Programming Model 3191
24.1.5.1 High-Speed USB Controller Interface Selection 3191
24.1.5.2 Enable Simulation Acceleration Features 3191
24.1.5.3 Enabling MSTANDBY in Force-Standby Mode 3191
24.1.5.4 Power Management Basic Programming Model 3191
24.1.5.4.1 High-Speed USB Controller Not Used for Application 3192
24.1.5.4.2 High-Speed USB Controller in Host Mode 3192
24.1.5.4.3 High-Speed USB Controller in Peripheral Mode 3192
24.1.5.4.4 High-Speed USB Controller in Host/Peripheral Mode 3193
24.1.6 High-Speed USB OTG Controller Register Manual 3194
24.1.6.1 High-Speed USB Register Mapping Summary 3194
24.1.6.2 Register Description 3194
24.2 High-Speed USB Host Subsystem 3198
24.2.1 High-Speed USB Host Subsystem Overview 3198
24.2.1.1 Main Features 3199
24.2.2 High-Speed USB Host Subsystem Environment 3201
24.2.2.1 Standard USB Implementation: Transceiver Connection 3201
24.2.2.2 TLL Connection 3201
24.2.2.3 ULPI Interfaces 3202
24.2.2.3.1 Transceiver Interface Configurations 3205
24.2.2.3.2 TLL Configurations 3205
24.2.2.3.3 High-Speed USB Host Subsystem Functional Interfaces 3207
24.2.2.4 Serial Interfaces 3209
24.2.2.4.1 Encoding in Serial Mode 3211
24.2.2.4.2 Sideband Signals for Serial Modes 3214
24.2.2.4.3 Transceiver Interface Configurations 3215
24.2.2.4.4 TLL Configurations 3218
24.2.2.4.5 High-Speed USB Host Subsystem Interface Description 3221
24.2.3 High-Speed USB Host Subsystem Integration 3224
24.2.3.1 Reset, Clocking, and Power-Management Scheme 3225
24.2.3.1.1 High-Speed USB Host Subsystem Resets 3225
24.2.3.1.2 High-Speed USB Host Subsystem Clocks 3226
24.2.3.1.3 Power-Management Scheme 3229
24.2.3.2 Hardware Requests 3232
24.2.3.2.1 Interrupt Requests 3232
24.2.3.2.2 IDLE Handshake Protocol 3233
24.2.3.2.3 MSTANDBY Handshake Protocol 3233
24.2.3.2.4 Wake-Up Request 3233
24.2.4 High-Speed USB Host Subsystem Functional Description 3234

24.2.4.1 High-Speed USB Host Controller Functionality 3234
24.2.4.1.1 High-Speed USB Host Controller Architecture 3234
24.2.4.1.2 OHCI Implementation Specifications 3235
24.2.4.1.3 UTMI Ports 3236
24.2.4.1.4 ULPI Ports 3236
24.2.4.1.5 Port Status 3236
24.2.4.1.6 Save and Restore 3237
24.2.4.1.7 Burst Control 3237
24.2.4.2 USBTLL Module Functionality 3237
24.2.4.2.1 Channels and Ports 3237
24.2.4.2.2 Channel Architecture 3238
24.2.4.2.3 Channel Configuration 3239
24.2.4.2.4 VBUS Management and Emulations 3241
24.2.4.2.5 Multimode Serial Port 3242
24.2.4.2.6 Attach/Connect Emulation for Serial TLL Modes 3243
24.2.4.2.7 Save and Restore 3244
24.2.5 High-Speed USB Host Subsystem Basic Programming Model 3246
24.2.5.1 Selecting and Configuring USB Connectivity 3246
24.2.5.1.1 ULPI Interface Selection 3247
24.2.5.1.2 Serial Interface Selection 3248
24.2.5.2 USBTLL Registers 3248
24.2.5.2.1 TLL Control and Status Registers 3249
24.2.5.2.2 ULPI PHY-Side Registers 3249
24.2.6 High-Speed USB Host Subsystem Register Manual 3250
24.2.6.1 USBTLL ULPI PHY-Side Register Space 3250
24.2.6.2 L4-Core Interconnect Register Space 3251
24.2.6.3 High-Speed USB Host Subsystem Register Mapping Summary 3251
24.2.6.4 Register Description 3255
24.2.6.4.1 USBTLL Registers 3255
24.2.6.4.2 UHH_config Registers 3287
24.2.6.4.3 OHCI Registers 3292
24.2.6.4.4 EHCI Registers 3308

25 General-Purpose Interface 3325

25.1 General-Purpose Interface Overview 3326
25.1.1 Global Features 3326
25.2 General-Purpose Interface Environment 3328
25.2.1 GPIO as a Keyboard Interface 3328
25.2.2 General-Purpose Interface Functional Interfaces 3330
25.2.2.1 Basic General-Purpose Interface Pins 3330
25.3 General-Purpose Interface Integration 3331
25.3.1 Description 3331
25.3.1.1 Clocking, Reset, and Power-Management Scheme 3331
25.3.1.1.1 Clocking 3331
25.3.1.1.2 Reset 3332
25.3.1.1.3 Power Domain 3332
25.3.1.1.4 Power Management 3332
25.3.1.2 Hardware Requests 3335
25.3.1.2.1 Interrupt Requests 3335

25.4 General-Purpose Interface Functional Description 3338
25.4.1 Interrupt and Wake-Up Features 3339
25.4.1.1 Synchronous Path: Interrupt Request Generation 3339
25.4.1.2 Asynchronous Path: Wake-Up Request Generation 3340
25.4.1.3 Interrupt (or Wake-Up) Line Release 3341
25.5 General-Purpose Interface Basic Programming Model 3342
25.5.1 Power Saving by Grouping the Edge/Level Detection 3342
25.5.2 Set-and-Clear Instructions 3342
25.5.2.1 Description 3342
25.5.2.2 Clear Instruction 3342
25.5.2.2.1 Clear Registers Addresses 3342
25.5.2.2.2 Clear Instruction Example 3343
25.5.2.3 Set Instruction 3343
25.5.2.3.1 Set Registers Addresses 3343
25.5.2.3.2 Set Instruction Example 3344
25.5.3 Interrupt and Wakeup 3344
25.5.3.1 Involved Configuration Registers 3344
25.5.3.2 Description 3345
25.5.4 Data Input (Capture)/Output (Drive) 3346
25.5.5 Debouncing Time 3347
25.6 General-Purpose Interface Register Manual 3348
25.6.1 General-Purpose Interface Register Mapping Summary 3348
25.6.2 Register Descriptions 3350

26 Initialization 3367

26.1 Initialization Overview 3368
26.1.1 Terminology 3368
26.1.2 Initialization Process 3368
26.2 Preinitialization 3369
26.2.1 Power Connections 3369
26.2.2 Clock and Reset 3371
26.2.2.1 Clock and Reset Overview 3371
26.2.2.2 Clock Configuration 3372
26.2.2.2.1 Required System Input Clocks 3372
26.2.2.2.2 Optional System Input Clock: sys_altclk 3373
26.2.2.2.3 Optional System Output Clock: sys_clkout1 and sys_clkout2 3373
26.2.2.3 Reset Configuration 3373
26.2.3 Boot Configuration 3374
26.3 Power, Clocks, and Reset Power-Up Sequence 3378
26.4 Device Initialization by ROM Code 3378
26.4.1 Booting Overview 3378
26.4.1.1 Booting Types 3378
26.4.1.2 Main Features 3379
26.4.2 Memory Map 3381
26.4.2.1 ROM Memory Map 3381
26.4.2.2 RAM Memory Map 3383
26.4.3 Overall Booting Sequence 3385
26.4.4 Startup and Configuration 3386
26.4.4.1 Startup 3386

26.4.4.2 Clocking Configuration 3386
26.4.4.3 Booting Device List Setup 3387
26.4.4.4 Software Booting Configuration 3388
26.4.5 Peripheral Booting 3390
26.4.5.1 Overview 3390
26.4.5.2 UART 3393
26.4.5.3 USB 3393
26.4.5.3.1 USB Driver Descriptors 3393
26.4.5.3.2 USB Customized Descriptors 3397
26.4.5.3.3 USB Driver Functionality 3397
26.4.6 Fast External Booting 3398
26.4.6.1 Overview 3398
26.4.6.2 External Booting 3399
26.4.7 Memory Booting 3399
26.4.7.1 Overview 3399
26.4.7.2 Non-XIP Memory 3400
26.4.7.3 XIP Memory 3402
26.4.7.3.1 GPMC Initialization 3402
26.4.7.4 NAND 3403
26.4.7.4.1 Initialization and NAND Detection 3403
26.4.7.4.2 Read Sector Procedure 3410
26.4.7.5 OneNAND/Flex-OneNAND 3411
26.4.7.5.1 Initialization and OneNAND/Flex-OneNAND Detection 3411
26.4.7.5.2 OneNAND/Flex-OneNAND Read Sector Procedure 3411
26.4.7.5.3 Flex-OneNAND Support Limitations 3412
26.4.7.6 MMC/SD Cards 3413
26.4.7.6.1 Connectivity Constraints 3414
26.4.7.6.2 Initialization and MMC/SD Card Detection 3416
26.4.7.6.3 Read Sector Procedure 3417
26.4.7.6.4 File System Handling 3418
26.4.7.7 DiskOnChip 3424
26.4.8 Image Format 3424
26.4.8.1 Overview 3424
26.4.8.2 CH 3425
26.4.8.2.1 CHSETTINGS 3426
26.4.8.2.2 CHRAM 3427
26.4.8.2.3 CHFLASH 3428
26.4.8.2.4 CHMMCSD 3429
26.4.8.3 Image Format for GP Devices 3430
26.4.8.4 Image Execution 3430
26.4.9 Tracing 3431
26.5 Wake-Up Booting by ROM Code 3432
26.6 Debug Configuration 3436
26.6.1 Overview 3436
26.6.2 JTAG Port Signal Description 3436
26.6.3 Initial Scan Chain Configuration 3437
26.6.4 Adding TAPs to the Scan Chain 3437
26.6.5 Debugger Address Space 3438

A Glossary 3439

List of Figures

1-1 OMAP34xx High-Tier Environment 178
1-2 OMAP34xx Block Diagram 179
1-3 POP Concept 184
1-4 Stacked Memory Package on the POP Device 184
1-5 Stacked Memory Package on the POP Device 185
2-1 Interconnect Overview 193
2-2 IVA2.2 Subsystem Memory Hierarchy 207
2-3 L1D RAM Cache Allocation Example (L3 Interconnect View) 208
2-4 IVA2.2 iMMU Address Translation 209
3-1 MPU Subsystem Overview 214
3-2 MPU Subsystem Integration Overview 217
3-3 MPU Subsystem Clocking Scheme 218
3-4 MPU Subsystem Reset Scheme 219
3-5 Bridges Overview 223
3-6 MPU Subsystem Power Domain Overview 225
4-1 Comparison of Energy Consumption With/Without DVFS 234
4-2 Comparison of Energy Consumed With/Without DPS 235
4-3 Performance Level and Applied Power-Management Techniques 237
4-4 Generic Clock Domain 238
4-5 Generic Power Domain 238
4-6 Generic Voltage Domain 240
4-7 Voltage, Power, and Clock Domain Hierarchical Architecture 241
4-8 Functional and Interface Clocks 242
4-9 PRCM Overview 245
4-10 PRCM Functional External Interface (Detailed View) 247
4-11 External Clock Interface 248
4-12 PRCM External Clock Sources 249
4-13 External Reset Signals 250
4-14 Power Control Interface for External Power IC 250
4-15 PRCM Integration 252
4-16 Device Power Domains 253
4-17 PRCM Reset Signals 255
4-18 Reset Manager Interface With Generic Power Domain 256
4-19 Reset Sources Overview 258
4-20 Reset Destination Overview 261
4-21 External Warm Reset Interface 265
4-22 Device Reset Manager Overview 267
4-23 Power Domain Reset Management: Part 1 268
4-24 Power Domain Reset Management: Part 2 269
4-25 Power Domain Reset Management: Part 3 270
4-26 Power-Up Sequence 274
4-27 Warm Reset Sequence 277
4-28 IVA2.2 Subsystem Power-Up Reset Sequence 280
4-29 IVA2 Software Reset Sequence 283
4-30 IVA2 Global Warm Reset Sequence 285
4-31 IVA2 Power Domain Power Transition Reset Sequence 287
4-32 Power Control Overview 290
4-33 PRCM Clock Manager Overview 298
4-34 External Clock I/O 299
4-35 Internal Clock Sources 302
4-36 PRM Clock Generator 304
4-37 CM Clock Generator Functional Overview 306

4-38 Generic DPLL Functional Diagram 307
4-39 DPLL3 Clocks 309
4-40 DPLL4 Clocks 310
4-41 DPLL5 Clocks 311
4-42 MPU Power Domain Clocking Scheme 313
4-43 IVA2 Power Domain Clocking Scheme 314
4-44 SGX Power Domain Clocking Scheme 315
4-45 CORE Clock Signals: Part 1 316
4-46 CORE Clock Signals: Part 2 317
4-47 CORE Clock Signals: Part 3 318
4-48 EFUSE Clock Signals 319
4-49 DSS Clock Signals 320
4-50 CAM Clock Signals 321
4-51 USBHOST Clock Signals 322
4-52 WKUP Clock Signals 323
4-53 PER Clock Signals 324
4-54 DPLL Clock Signals 325
4-55 System Clock Oscillator Controls 331
4-56 Common PRM Source-Clock Controls 338
4-57 Common CM Source-Clock Controls 340
4-58 Common Interface Clock Controls 341
4-59 DPLL Power Domain Clock Controls 342
4-60 SGX Power Domain Clock Controls 343
4-61 CORE Power Domain Clock Controls: Part 1 344
4-62 CORE Power Domain Clock Controls: Part 2 345
4-63 EFUSE Power Domain Clock Controls 346
4-64 DSS Power Domain Clock Controls 347
4-65 CAM Power Domain Clock Controls 348
4-66 USBHOST Power Domain Clock Controls 348
4-67 WKUP Power Domain Clock Controls 349
4-68 PER Power Domain Clock Controls: Part 1 350
4-69 PER Power Domain Clock Controls: Part 2 351
4-70 Power Domain Sleep/Wake-Up Transition 356
4-71 Device Power Reset and Clock Controllers 357
4-72 Save-and-Restore Sequence 370
4-73 Overview of Device Voltage Domains 374
4-74 Overview of Device Voltage Distribution 375
4-75 PRM Voltage Control Architecture 380
4-76 Voltage Transition Controlled by sys_nvmode2 381
4-77 Device Off-Mode Control Overview 384
4-78 sys_clkout2 Gating Polarity Control 393
4-79 Off Mode Wakeup Using I^2C 412
4-80 OFF Mode Wakeup Using SYS_OFF_MODE 413
4-81 Functional Clock Basic Programming Model 416
4-82 Functional Clock Switching 417
4-83 Interface Clock Basic Programming Model 418
4-84 Domain Inactive STATE Basic Programming Model 419
4-85 Processor Clock Basic Programming Model 421
4-86 Wake-up Basic Programming Model 423
4-87 Voltage Controller Initialization Flow Chart 424
4-88 Voltage Control Through VMODE Flow Chart 429
5-1 Interconnect Architecture Overview 617
5-2 L3 Interconnect Overview 623

5-3 Flow Chart of the Protection Mechanism 627
5-4 L3 Firewall Implementation 628
5-5 L3 Region Overlay and Priority Level Overview 630
5-6 Example of REQ_INFO_PERMISSION Register 632
5-7 L3 Error Reporting Structure 636
5-8 Global Error Routing 640
5-9 L3 Error Routing 641
5-10 Typical Error Analysis Sequence 644
5-11 Firewall Configuration Solution 1 648
5-12 Firewall Configuration Solution 2 649
5-13 L4 Interconnect Overview 678
5-14 L4 Initiator-Target Connectivity for L4-Core and L4-Per 678
5-15 Example of CONNID_BIT_VECTOR 684
5-16 L4 Firewall Overview 686
5-17 L4 Error Reporting 694
6-1 Simplified Block Diagram of the IPC 730
6-2 IPC Integration 731
6-3 Mailbox Block Diagram 734
6-4 Example of Communication 741
6-5 Overview 742
6-6 Initial Configuration Flow Chart 744
6-7 MPU Subsystem Message Sending Flow Chart 745
6-8 IVA2.2 Subsystem Message Receiving Flow Chart 745
6-9 IVA2.2 Subsystem Message Sending Flow Chart 746
6-10 MPU Subsystem Message Receiving Flow Chart 747
7-1 SCM Overview 756
7-2 SCM Environment Overview 757
7-3 SCM Interface Signals 758
7-4 SCM Integration 759
7-5 Internal Clock Implementation 762
7-6 SCM Block Diagram 763
7-7 Pad Configuration Register Functionality 764
7-8 Pad Configuration Diagram 766
7-9 System Off Mode Pad Control Overview 779
7-10 Save-and-Restore Mechanism Overview 780
7-11 Wake-Up Event Detection Overview 781
7-12 Functional Block Diagram 782
7-13 Extended-Drain I/O 784
7-14 Functional Block Diagram 786
7-15 Single Conversion Mode (CONTCONV = 0) 787
7-16 Continuous Conversion Mode (CONTCONV = 1) 788
7-17 Overview of the Debug and Observability Register Functionality 793
7-18 DPLL With EMI Reduction Feature 829
7-19 DPLL-D Integration 830
7-20 Spreading Generation Block Diagram 831
7-21 Modulation Profiles 833
7-22 Effect of the SSC in Frequency 834
7-23 Effect of the SSC in the Time Domain 834
7-24 Peak Reduction Caused by Spreading 835
7-25 Supported Spreading Frequency and Deviation 836
7-26 Supported Safe Operating Regions and Jitter Impact 836
7-27 Flow Chart 845
7-28 VDDS Ramps Up Before VDD2 847

7-30 I/O Power Optimization Flow Chart 850
8-1 OMAP34xx MMU Instances 932
8-2 Camera MMU System Integration 933
8-3 IVA2.2 MMU System Integration 933
8-4 MMU Address Translation 937
8-5 MMU Usage Examples 938
8-6 MMU Architecture 939
8-7 Translation Process 940
8-8 Translation Hierarchy 941
8-9 First-Level Descriptor Address Calculation 941
8-10 Detailed First-Level Descriptor Address Calculation 942
8-11 Section Translation Summary 943
8-12 Supersection Translation Summary 944
8-13 Two-Level Translation 944
8-14 Small Page Translation Summary 945
8-15 Large Page Translation Summary 946
8-16 TLB Entry Lock Mechanism 947
8-17 TLB Entry Structure 948
8-18 MMU Configuration Strategies 950
8-19 MMUn Translation Table Hierarchy 953
8-20 Translation of a Supersection 954
8-21 Translation of a Section 954
8-22 Translation of a Large Page Included in a Page Table 955
8-23 Translation of an Extended Small Page Included in a Page Table 956
9-1 SDMA Overview 973
9-2 Edge-Sensitive DMA Request Scheme 974
9-3 Transition-Sensitive DMA Request Scheme 974
9-4 SDMA Controller Integration 975
9-5 Example of External DMA Requests Use to the SDMA Controller 976
9-6 SDMA Controller Top-Level Block Diagram 981
9-7 Example Showing Double-Index Addressing, Elements, Frames, and Strides 985
9-8 Addressing Mode Example (a) 985
9-9 Addressing Mode Example (b) 985
9-10 Addressing Mode Example (c) 986
9-11 Example of a 90° Clockwise Image Rotation 987
9-12 2-D Graphic Transparent Color Block Diagram 995
9-13 Overview 1004
9-14 Environment 1004
9-15 Data Flow 1005
9-16 Overview 1008
10-1 Interrupt Controllers Highlight 1039
10-2 Interrupts From External Devices 1040
10-3 MPU Subsystem INTCPS Integration 1041
10-4 Top-Level Block Diagram 1046
10-5 IRQ/**FIQ** Processing Sequence 1052
10-6 Nested IRQ/**FIQ** Sequence 1055
11-1 GPMC Environment 1072
11-2 GPMC to 16-Bit Address/Data-Multiplexed Memory 1074
11-3 GPMC to 16-Bit NAND Device 1075
11-4 GPMC Integration in the OMAP Device 1077
11-5 GPMC Functional Diagram 1081
11-6 Chip-Select Address Mapping and Decoding Mask 1084
11-7 Asynchronous Single Read on a Nonmultiplexed Address/Data Device 1088

11-8 Wait Behavior During an Asynchronous Single Read Access (GPMCFCLKDivider = 1) 1094
11-9 Wait Behavior During a Synchronous Read Burst Access 1096
11-10 Asynchronous Single Read on an Address/Data-Nonmultiplexed Device 1101
11-11 Asynchronous Single Read on an Address/Data-Multiplexed Device 1102
11-12 Asynchronous Single Write on an Address/Data-Nonmultiplexed Device 1103
11-13 Asynchronous Single Write on an Address/Data-Multiplexed Device 1104
11-14 Asynchronous Multiple (Page Mode) Read 1105
11-15 Synchronous Single Read (GPMCFCLKDIVIDER = 0) 1107
11-16 Synchronous Single Read (GPMCFCLKDIVIDER = 1) 1108
11-17 Synchronous Single Write on an Address/Data-Multiplexed Device 1109
11-18 Synchronous Multiple (Burst) Read (GPMCFCLKDIVIDER = 0) 1110
11-19 Synchronous Multiple (Burst) Read (GPMCFCLKDIVIDER = 1) 1111
11-20 Synchronous Multiple (Burst) Write 1112
11-21 Synchronous Multiple Write (Burst Write) in Address/Data-Multiplexed Mode 1114
11-22 NAND Command Latch Cycle 1118
11-23 NAND Address Latch Cycle 1118
11-24 NAND Data Read Cycle 1119
11-25 NAND Data Write Cycle 1120
11-26 Hamming Code Accumulation Algorithm (1/2) 1124
11-27 Hamming Code Accumulation Algorithm (2/2) 1125
11-28 ECC Computation for a 256-Byte Data Stream (Read or Write) 1125
11-29 ECC Computation for a 512-Byte Data Stream (Read or Write) 1126
11-30 128 Word16 ECC Computation 1127
11-31 256 Word16 ECC Computation 1127
11-32 Manual Mode Sequence and Mapping 1132
11-33 NAND Page Mapping and ECC: Per-Sector Schemes 1136
11-34 NAND Page Mapping and ECC: Pooled Spare Schemes 1137
11-35 NAND Page Mapping and ECC: Per-Sector Schemes, With Separate ECC 1138
11-36 NAND Read Cycle Optimization Timing Description 1144
11-37 GPMC Connection to an External NOR Flash Memory 1147
11-38 Synchronous Burst Read Access (Timing Parameters in Clock Cycles) 1149
11-39 Asynchronous Single Read Access (Timing Parameters in Clock Cycles) 1150
11-40 Asynchronous Single Write Access (Timing Parameters in Clock Cycles) 1151
11-41 SDRC Subsystem Environment 1184
11-42 SDRC Subsystem Connections to SDR SDRAM 1186
11-43 SDRC Subsystem Connections to DDR SDRAM 1187
11-44 SDRC SDR/DDR-SDRAM System Address Multiplexing Schemes (1 of 3) 1191
11-45 SDRC SDR/DDR-SDRAM System Address Multiplexing Schemes (2 of 3) 1192
11-46 SDRC SDR/DDR-SDRAM System Address Multiplexing Schemes (3 of 3) 1193
11-47 SDRC Integration to the OMAP Device 1194
11-48 SMS Top-Level Diagram 1197
11-49 Region Organization 1203
11-50 SDRC Architecture 1207
11-51 CS0/CS1 Chip-Select Start Address Slots 1208
11-52 SDRAM Controller Block Diagram 1210
11-53 Address Multiplexing Scheme According to BANKALLOCATION 1211
11-54 Simplified View of Bank-Row-Column vs Row-Bank-Column Bank Allocation 1212
11-55 Data Multiplexing Scheme 1214
11-56 Data Demultiplexing Scheme 1215
11-57 Generic DDR Data-Write and Data-Read Waveforms 1221
11-58 Required Synchronization DFF Input Signals 1221
11-59 DLL/CDL Architecture 1222
11-60 Simplified DLL/CDL Block Diagram 1223

11-61 Natural Scan Order 1226
11-62 SDRC Subsystem Overview 1238
11-63 YUV Format: Pixel Representation 1239
11-64 VRFB Context Configuration 1240
11-65 Example of VRFB Context 1 Configuration 1241
11-66 Display a Rotated QVGA Image 1243
11-67 Arbitration Granularity Versus Arbitration Decision 1246
11-68 BURST-COMPLETE On Class 2-Group 3 1247
11-69 Priority Between Classes 1248
11-70 Idle Cycle Mechanism Within A Burst 1249
11-71 Example of EXTENDEDGRANT Mechanism 1250
11-72 Arbitration Between Classes 1251
11-73 Arbitration Within a Class 1252
11-74 Generic Arbitration Decision 1253
11-75 Arbitration Granularity 1254
11-76 Mobile DDR SDRAM to SDRC Interface 1255
11-77 SDRAM Power-Up Sequence 1256
11-78 Normal Operating Sequence 1257
11-79 SDRAM Burst-Read Timing Diagram 1259
11-80 SDRC Camcorder Use Case Overview 1261
11-81 SDRC Camcorder Use Case Environment 1262
11-82 VRFB Actual Image Size vs Programmed Image Size 1264
11-83 SDRC Address Space in MPU Global Address Space 1266
11-84 CS Start and End Address Configuration Example 1269
11-85 OCM Subsystem Overview 1303
11-86 OCM Subsystem Integration to the Device 1304
12-1 Camera ISP Highlight 1309
12-2 Parallel Interface in Generic Configuration 1315
12-3 Parallel Interface in ITU-R BT.656 Configuration 1316
12-4 CSI2, CSI1 Serial Interface Configuration 1317
12-5 Synchronization Signals and Frame Timing in SYNC Mode 1318
12-6 Synchronization Signals and Data Timing in SYNC Mode 1318
12-7 SYNC Mode Clock Gating 1319
12-8 JPEG Stream Timing Diagrams 1319
12-9 Data Timing With Embedded Synchronization Signals (8-Bit Case) 1320
12-10 CSI1 Receiver Configuration Interface Signals 1322
12-11 Example of 0xFF00 0002 Transmission 1324
12-12 YUV422 Big Endian 1325
12-13 YUV422 Little Endian 1326
12-14 YUV420 1327
12-15 RGB888 1328
12-16 RGB565 1329
12-17 RGB444 1329
12-18 RAW8 1330
12-19 RAW10 1331
12-20 RAW12 1332
12-21 JPEG8 and JPEG8 FSP 1333
12-22 Two Data-Lane Merger Configuration 1334
12-23 One Data-Lane Configuration 1335
12-24 CSI2 Protocol Layer With Short and Long Packets 1335
12-25 Short Packet Structure 1335
12-26 Long Packet Structure 1336
12-27 Data Identifier Structure 1337

12-28 Virtual Channel 1337
12-29 General Frame Structure (Informative) 1340
12-30 Digital Interlaced Video Frame (Informative) 1341
12-31 YUV420 8-Bit 1342
12-32 YUV420 10-Bit 1343
12-33 YUV420 8-Bit Legacy 1344
12-34 YUV420 8-Bit + CSPS 1345
12-35 YUV420 10-Bit + CSPS 1346
12-36 YUV422 8-Bit 1347
12-37 YUV422 10-Bit 1347
12-38 RGB565 1348
12-39 RGB888 1349
12-40 RGB666 1350
12-41 RGB444 1351
12-42 RGB555 1351
12-43 RAW6 1352
12-44 RAW7 1353
12-45 RAW8 1354
12-46 RAW10 1355
12-47 RAW12 1356
12-48 RAW14 1357
12-49 JPEG8 1358
12-50 Camera ISP Integration 1359
12-51 Clock Tree 1360
12-52 Interrupt Generation Tree 1364
12-53 Camera ISP Block Diagram 1371
12-54 Camera ISP/Data Path/RAW RGB Images 1373
12-55 Camera ISP/Data Path/YUV4:2:2 Images 1374
12-56 Camera ISP/Data Path/JPEG Images 1374
12-57 CSI1 Receiver Block Diagram 1375
12-58 Synchronization State-Machine 1377
12-59 CSI1 Frame Structure: Non-JPEG Data Format 1378
12-60 Frame Structure: JPEG8 Data Format 1378
12-61 CSI1 Data Structure 1379
12-62 CSI2 Receiver Block Diagram 1381
12-63 SHORT_PACKET Field Format 1383
12-64 Virtual Channel to Context 1384
12-65 Pixel Data Destination Setting in Progressive and Interlaced Mode 1386
12-66 Complex I/O Overview 1386
12-67 Complex I/O Power FSM 1387
12-68 RxMode and StopState FSM 1387
12-69 Timing-Control Module 1388
12-70 Control-Signal Generation 1389
12-71 Use of cam_global_reset With Global Reset Release Camera Modules 1391
12-72 CCDC Block Diagram 1394
12-73 Optical Clamp Representation 1397
12-74 Data Formatter Conversion Area Selection 1399
12-75 CCDC/Culling: Example for Decimation Pattern 1402
12-76 A-Law Table 1403
12-77 CCDC/Line-Output Control: Sample Formats of Input and Output Images 1404
12-78 Preview Engine Block Diagram 1407
12-79 Horizontal Distances for Different Patterns 1408
12-80 Resizer Process 1414

12-81 Resizer in Memory-Input Mode 1415
12-82 Typical Sample-Rate Converter 1416
12-83 Resizer Functionality 1416
12-84 Resizer Approximation Scheme 1416
12-85 Cutoff Frequency for Low-Pass Filter 1416
12-86 Alignment of Input Pixels to Tap Coefficients 1418
12-87 Pseudo-Code Description of the Resizer Algorithm in the 4-Tap/8-Phase Mode 1419
12-88 Pseudo-Code Description of the Resizer Algorithm in the 7-Tap/4-Phase Mode 1420
12-89 Histogram Process 1426
12-90 Color Pattern Index 1427
12-91 Region Priority 1428
12-92 Central-Resource SBL Block Diagram 1429
12-93 Single Slice Buffer (Write Mode) 1434
12-94 Single Slice Buffer Example (Write Mode) 1434
12-95 Circular Buffer Control Feedback Loop Example 1436
12-96 Extended Slice Buffer Example 1437
12-97 CSI1_CTRL.VP_CLK_POL Settings 1444
12-98 SOF and EOF Region Settings 1446
12-99 Pixel Data Region Settings 1447
12-100 Pixel Data Destination Settings 1448
12-101 CSI2 Receiver Global Reset Flow Chart 1449
12-102 cam_strobe Signal-Generation for Red-Eye Removal 1456
12-103 Dependencies Among Framing Settings in Data Flow 1460
12-104 CCDC_VD0_IRQ/CCDC_VD1_IRQ Interrupt Behavior When VDPOL = 0 1462
12-105 CCDC_VD0_IRQ/CCDC_VD1_IRQ Interrupt Behavior When VDPOL = 1 1462
12-106 CCDC_VD2_IRQ Interrupt Behavior 1462
12-107 HS/VS Sync Pulse Output Timings 1465
12-108 Mosaic Filter - CCDC_COLPTN Bit Field Settings 1466
12-109 Data Packing - Pixel Ordering 1471
12-110 Clipping Window Before Output to Memory 1472
12-111 Firmware Interactions for Memory-Input Resizing 1480
12-112 Video-Port Interface Bandwidth Balancing 1491
12-113 Memory Read Bandwidth Balancing 1492
13-1 Graphics Accelerator Highlight 1732
13-2 SGX Subsystem Integration 1735
13-3 SGX Block Diagram 1737
14-1 IVA2.2 Subsystem Highlight 1740
14-2 IVA2.2 Subsystem Integration 1742
14-3 IVA2.2 Subsystem Resets 1744
14-4 IVA2.2 Power Domain 1746
14-5 IVA2.2 EDMA Requests 1747
14-6 IVA2.2 Interrupt Management 1749
14-7 IVA2.2 Subsystem Block Diagram 1752
14-8 DSP Megamodule Block Diagram 1753
14-9 DSP Megamodule INTC Block Diagram 1757
14-10 Interrupt Selector Block Diagram 1759
14-11 IVA2.2 EDMA Overview 1762
14-12 TPCC Block Diagram 1763
14-13 DMA/QDMA Channel Mapping and PaRAM Entry 1765
14-14 TPTC Block Diagram 1768
14-15 Transfer Geometry 1769
14-16 IVA2.2 MMU Block Diagram 1773
14-17 IVA2.2 MMU Translation Table Hierarchy 1774

14-18 Sequencer Block Diagram 1775
14-19 Sequencer Memory Mapping 1778
14-20 iLF/iME Execution State-Machine 1780
14-21 iLF/iME Instruction Sequence Format in L2 Memory 1785
14-22 iLF Overview 1786
14-23 iLF Stack Parameter Format 1791
14-24 iLF Pixel Format in L2 Memory 1797
14-25 iLF Internal Pixel Format 1797
14-26 iLF Edge Numbering 1798
14-27 iME Overview 1799
14-28 iME Pixel Format in L2 1814
14-29 SL2 Memory Interface Block Diagram 1817
14-30 IVA2.2 WUGEN Description 1819
14-31 WUGEN Event Generation 1820
14-32 WUGEN Event Masking 1821
14-33 WUGEN Event Mask Clear 1822
14-34 SYSC Block Diagram 1823
14-35 IVA2.2 Local Memories Hierarchy 1826
14-36 IVA2 Boot Mode Configuration 1829
14-37 IVA2 Boot Basic Programming Model 1833
14-38 iME/iLF Typical Use Flowchart 1859
14-39 IVA2.2 Interrupt Flow 1861
14-40 Process of Identifying Source Event of an Interrupt 1867
14-41 L1P Memory Protection Registers 1868
14-42 L1D Memory Protection Registers 1869
14-43 L2 Memory Protection Registers 1869
14-44 IVA2 Power Off 1878
14-45 IVA2 Power Down 1879
14-46 IVA2 Wake Up 1880
15-1 Display Subsystem Highlight 2125
15-2 LCD Support Parallel Interface (RFBI Mode) 2130
15-3 External Generation of TE Signal Based on Logical OR Operation Between HSYNC and VSYNC (Active-High) 2131
15-4 LCD Support Parallel Interface (Bypass Mode) 2132
15-5 LCD Pixel Data Monochrome4 Passive Matrix 2133
15-6 LCD Pixel Data Monochrome8 Passive Matrix 2134
15-7 LCD Pixel Data Color Passive Matrix 2134
15-8 LCD Pixel Data Color12 Active Matrix 2135
15-9 LCD Pixel Data Color16 Active Matrix 2136
15-10 LCD Pixel Data Color18 Active Matrix 2136
15-11 LCD Pixel Data Color24 Active Matrix 2137
15-12 RFBI Data Stall Signal Diagram 2137
15-13 RFBI Data Stall Signal Diagram With Handcheck 2138
15-14 Command Data Write 2138
15-15 Display Data Read 2139
15-16 Read to Write and Write to Read 2139
15-17 Active Matrix Timing Diagram of Configuration 1 (Start of Frame) 2140
15-18 Active Matrix Timing Diagram of Configuration 1 (Between Lines) 2140
15-19 Active Matrix Timing Diagram of Configuration 1 (Between Frames) 2141
15-20 Active Matrix Timing Diagram of Configuration 1 (End of Frame) 2141
15-21 Active Matrix Timing Diagram of Configuration 2 (Start of Frame) 2141
15-22 Active Matrix Timing Diagram of Configuration 2 (Between Lines) 2141
15-23 Active Matrix Timing Diagram of Configuration 2 (Between Frames) 2142
15-24 Active Matrix Timing Diagram of Configuration 2 (End of Frame) 2142

15-25 Active Matrix Timing Diagram of Configuration 3 (Start of Frame) 2142
15-26 Active Matrix Timing Diagram of Configuration 3 (Between Lines) 2143
15-27 Active Matrix Timing Diagram of Configuration 3 (Between Frames) 2143
15-28 Active Matrix Timing Diagram of Configuration 3 (End of Frame) 2143
15-29 Passive Matrix Timing Diagram (Start of Frame) 2144
15-30 Passive Matrix Timing Diagram (Between Lines) 2144
15-31 Passive Matrix Timing Diagram (Between Frames) 2144
15-32 Passive Matrix Timing Diagram (End of Frame) 2144
15-33 Typical SDI Connection 2145
15-34 Typical DSI Connection 2147
15-35 DSI Video Mode Without Burst (No-Line Buffer) 2152
15-36 DSI Video Mode Without Burst (One-Line Buffer) 2153
15-37 DSI Video Mode With Burst (Two-Line Buffers) 2154
15-38 Stall Timing With Pixel on Rising Edge 2156
15-39 Stall Timing With Pixel on Falling Edge 2156
15-40 Data Flow in Command Mode Using the Video Port 2157
15-41 Two Data Lane Configuration 2158
15-42 One Data Lane Configuration 2158
15-43 Two Packets Using Two-Data Lane Configuration (Example) 2159
15-44 Protocol Layer With Short and Long Packets 2159
15-45 Short Packet Structure 2160
15-46 Long Packet Structure 2160
15-47 Data Identifier Structure 2161
15-48 Virtual Channel Controller 2161
15-49 DSI Video Mode: Nonburst Transfer With VE and HE 2164
15-50 DSI Video Mode: Nonburst Transfer Without VE and HE 2165
15-51 DSI Video Mode: Burst Transfer Without VE and HE 2166
15-52 DSI General Frame Structure 2167
15-53 DSI General Frame Structure Using Burst Mode 2168
15-54 DSi General Frame Structure Using Burst Mode and Interleaving 2169
15-55 24 Bits per Pixel RGB Color Format, Long Packet 2171
15-56 18 Bits per Pixel (Loosely Packed) RGB Color Format, Long Packet 2172
15-57 18 Bits per Pixel (Packed) RGB Color Format, Long Packet 2173
15-58 16 Bits per Pixel RGB Color Format, Long Packet 2174
15-59 24 Bits Per Pixel With One Data Channel 2175
15-60 24 Bits Per Pixel With Two Data Channels 2175
15-61 24 Bits Per Pixel With Three Data Channels 2175
15-62 TV Display Interface (s-video mode) 2176
15-63 TV Display Interface (Composite Mode) 2177
15-64 Display Subsystem Integration 2180
15-65 Display Subsystem Clock Tree 2181
15-66 Display Subsystem DMA Tree 2190
15-67 DSI Interrupt Tree 2191
15-68 DISPC and DSS Interrupts Tree 2192
15-69 Display Subsystem Full Schematic 2196
15-70 Display Controller Architecture Overview 2197
15-71 Palette/Gamma Correction Architecture 2201
15-72 YCbCr 4:2:2 to YCbCr 4:4:4 (0- or 180-Degree Rotation) 2204
15-73 YCbCr 4:2:2 to YCbCr 4:4:4 (90- or 270-Degree Rotation) 2204
15-74 Interpolation of the Missing Chrominance Component 2204
15-75 YCbCr to RGB Registers (VIDFULLRANGE=0) 2205
15-76 YCbCr to RGB Registers (VIDFULLRANGE=1) 2205
15-77 Color Space Conversion Macro-Architecture 2206

15-78 Video Upsampling 2207
15-79 Resampling Macro-Architecture (3-Coefficient Processing) 2208
15-80 Overlay Manager in Normal Mode 2211
15-81 Display Attributes in Normal Mode 2211
15-82 Overlay Manager in Alpha Mode 2212
15-83 Display Attributes in Alpha Mode 2213
15-84 Alpha Blending Macro Architecture 2214
15-85 Video Source Transparency Example 2216
15-86 Graphics Destination Transparency Example 2216
15-87 Color Phase Rotation Matrix 2218
15-88 Color Phase Rotation Macro Architecture 2218
15-89 DSI Protocol Engine 2221
15-90 DSI Transmitter/Receiver Data Flow 2222
15-91 LP to HS Timing 2223
15-92 HS to LP Timing 2224
15-93 HS Command Mode Interleaving 2229
15-94 LP Command Mode Interleaving 2231
15-95 Complex I/O Power FSM 2234
15-96 DSI PLL Power FSM 2235
15-97 DSI PLL HS Clock FSM 2237
15-98 ForceTxStopMode FSM 2238
15-99 TurnRequest FSM 2239
15-100 High-Speed TX Timer FSM 2240
15-101 Low-Power RX Timer FSM 2241
15-102 64-Bit ECC Generation on TX Side 2245
15-103 Checksum Transmission 2245
15-104 16 Bit CRC Generation Using a Shift Register 2246
15-105 DSI PLL Controller Overview 2247
15-106 DSI PLL Reference Diagram 2248
15-107 DSI Complex I/O Architecture 2250
15-108 RFBI Architecture Overview 2251
15-109 Video Encoder Architecture Overview 2254
15-110 Closed Captioning Timing 2257
15-111 WSS Timing 2259
15-112 Dual 10-Bit Video DAC Architecture 2260
15-113 DC-Coupling TV Detect Waveforms for TV Connected and Disconnected 2263
15-114 AC-Coupling TV Detect Waveforms for TV Connected and Disconnected 2263
15-115 GPIO Signal Waveform Proposal for TV Detection/Disconnection in DC-Coupling Mode 2264
15-116 GPIO Signal Waveform Proposal for TV Detection/Disconnection in AC-Coupling Mode 2264
15-117 DAC Test Mode in Composite Video Mode 2265
15-118 DAC Test Mode in Separate video Mode 2266
15-119 SDI Architecture Overview 2267
15-120 Overlay Optimization: Case 1 2273
15-121 Overlay Optimization: Case 2 2274
15-122 Overlay Optimization: Case 3 2274
15-123 Overlay Optimization: Case 4 2275
15-124 90° DMA Rotation Example 2281
15-125 Rotation/Mirroring Settings 2284
15-126 90° Rotation With Mirroring 2285
15-127 Offset for VRFB Rotation 2287
15-128 Offset for VRFB Rotation With Mirroring 2289
15-129 Timing Values Description (Active Matrix Display) 2291
15-130 PCDmin Formulas (V Down-Sampling Only) 2293

15-131 Color Phase Rotation Matrix 2295
15-132 Color Phase Rotation Matrix (R Component Only) 2295
15-133 Color Phase Rotation Matrix (G Component Only) 2295
15-134 Color Phase Rotation Matrix (B Component Only) 2295
15-135 Diagonal Matrix Configuration 2296
15-136 Example - Diagonal Matrix Configuration 2296
15-137 Image With and Without CPR (Diagonal Matrix) 2297
15-138 Example - Image With and Without CPR (Standard Matrix) 2298
15-139 DSI PLL Programming Blocks 2313
15-140 DSI PLL Go Sequence (Manual Mode) 2314
15-141 DSI PLL Go Sequence (Automatic Mode) 2315
15-142 Gated Mode Sequence 2316
15-143 DSI PLL Programming Sequence 2318
15-144 High-Speed Clock Transmission 2322
15-145 High-Speed Data Transmission 2324
15-146 How to Use RFBI 2333
15-147 RFBI Initial Configuration 2334
15-148 RFBI Output Enable 2335
15-149 SDI Start Sequence 2341
15-150 SDI Stop Sequence 2342
15-151 SDI Clock Source/Frequency Change Sequence Part A 2343
15-152 SDI Clock Source/Frequency Change Sequence Part B 2344
15-153 Vertical Filtering Macro Architecture (Three Taps) 2346
15-154 Vertical Filtering Macro Architecture (Five Taps) 2347
15-155 Horizontal Filtering Macro Architecture (Five Taps) 2348
15-156 Vertical Up-/Down-Sampling Algorithm 2349
15-157 Horizontal Up-/Down-Sampling Algorithm 2350
15-158 QVGA LCD Timings 2361
15-159 SDI PLL Architecture 2362
15-160 Main Flowchart 2363
15-161 Flowchart SDI 1-Data Pair 2364
15-162 Flowchart SDI 2-Data Pairs 2365
15-163 Flowchart SDI 3-Data Pairs 2366
15-164 HVGA Display 2367
15-165 Hardware Connections for FlatLink3G Application 2369
15-166 SN65LVDS302 Receiver Modes and Transitions 2372
15-167 Overview 2374
15-168 Environment 2375
15-169 Display Subsystem Data Flow 2376
15-170 Display Controller Data Flow 2377
15-171 Display Panel Configuration for the Camcorder Use Case 2377
15-172 Display Subsystem Configuration for Camcorder Use Case 2378
15-173 Display Subsystem Initialization 2379
15-174 Software Reset Flowchart 2380
15-175 Display Panel Configuration Flowchart 2385
15-176 QVGA LCD Panel Timings 2386
15-177 DSI Clock Tree in Video Mode 2388
15-178 Overview 2392
15-179 Overview 2400
16-1 Timers 2558
16-2 GP Timers Overview 2559
16-3 GP Timers External System Interface 2560
16-4 GP Timer Integration 2562

16-5 Wake-Up Request Generation 2566
16-6 Block Diagram of GPTIMER3 through GPTIMER9 and GPTIMER11 2569
16-7 Block Diagram of GPTIMER1, GPTIMER2, and GPTIMER10 2570
16-8 GPTi.TCRR Timing Value 2571
16-9 Block Diagram of the 1-ms Tick Module 2572
16-10 Capture Wave Example for GPTi.TCLR[13] CAPT_MODE = 0 2574
16-11 Capture Wave Example for GPTi.TCLR[13] CAPT_MODE = 1 2574
16-12 Timing Diagram of PWM With GPTi.TCLR[7] SCPWM Bit = 0 2576
16-13 Timing Diagram of PWM With GPTi.TCLR[7] SCPWM Bit = 1 2576
16-14 WDTs Block Diagram 2602
16-15 WDT Integration 2603
16-16 32-Bit WDT Functional Block Diagram 2606
16-17 WDT General Functional View 2607
16-18 32-kHz Sync Timer Block Diagram 2619
17-1 UART Module 2624
17-2 UART Mode Bus System Overview 2627
17-3 IrDA System Overview 2627
17-4 CIR System Overview 2628
17-5 UART Frame Data Format 2629
17-6 IrDA SIR Frame Format 2630
17-7 IrDA SIR Encoding Mechanism 2631
17-8 IrDA SIR Decoding Mechanism 2632
17-9 SIR Free Format Mode 2633
17-10 MIR Transmit Frame Format 2633
17-11 MIR Baud Rate Adjustment Mechanism 2634
17-12 SIP 2634
17-13 CIR Pulse Modulation 2636
17-14 CIR Modulation Duty Cycle 2637
17-15 RC-5 Bit Encoding 2638
17-16 SIRC Bit Encoding 2638
17-17 RC-5 Standard Packet Format 2639
17-18 SIRC Packet Format 2639
17-19 SIRC Bit Transmission Example 2639
17-20 UART Functional Integration 2640
17-21 UART/IrDA/CIR Block Diagram 2644
17-22 FIFO Management Registers 2645
17-23 Receive FIFO Interrupt Request Generation 2647
17-24 Transmit FIFO Interrupt Request Generation 2647
17-25 Receive FIFO DMA Request Generation (32 Characters) 2649
17-26 Transmit FIFO DMA Request Generation (56 Spaces) 2650
17-27 Transmit FIFO DMA Request Generation (8 Spaces) 2651
17-28 Transmit FIFO DMA Request Generation (1 Space) 2651
17-29 Transmission Process 2652
17-30 Reception Process 2652
17-31 Baud Rate Generation 2658
17-32 Baud Rate Generator 2665
17-33 CIR Mode Block Components 2670
18-1 HS I^2C Controllers 2724
18-2 Multimaster HS I^2C Controllers and Typical Connections to I^2C Devices 2726
18-3 Multimaster HS I^2C Controller Interface Signals in I^2C Mode 2726
18-4 I^2C Data Transfer 2727
18-5 Bit Transfer on the I^2C Bus 2727
18-6 S and P Condition Events 2728

18-7 I²C Data Transfer Formats in F/S Mode 2728
18-8 I²C Data Transfers in HS Mode 2729
18-9 Arbitration Between Master Transmitters 2730
18-10 Synchronization of I²C Clock Generators 2730
18-11 Multimaster HS I²C Controllers and Typical Connections to SCCB Devices 2731
18-12 Multimaster HS I²C Controller Interface Signals in SCCB Mode 2732
18-13 3-wire SCCB Transmission Timing Diagram 2733
18-14 SCCB Transmission Data Formats 2733
18-15 Typical Connection Between the HS I²C Controller and Power Chip(s) 2735
18-16 HS I²C Controller I2C4 Interface Signals 2735
18-17 I²C Data Transfer Format in F/S Mode for the I2C4 Module 2736
18-18 I²C Data Transfer Format in HS Mode for the I2C4 Module 2737
18-19 HS I²C Controller Integration 2738
18-20 Wake-up Generation Flow 2741
18-21 Multimaster HS I²C Controller Block Diagram 2746
18-22 Receive FIFO Interrupt Request Generation 2748
18-23 Transmit FIFO Interrupt Request Generation 2748
18-24 Receive FIFO DMA Request Generation 2749
18-25 Transmit FIFO Request Generation (High Threshold) 2750
18-26 Transmit FIFO Request Generation (Low Threshold) 2750
18-27 I²C Clock Generation 2752
18-28 I²C Setup Procedure 2758
18-29 I²C Master Transmitter Mode, Polling Method, in F/S and HS Modes 2759
18-30 I²C Master Receiver Mode, Polling Method, in F/S and HS Modes 2760
18-31 I²C Master Transmitter Mode, Interrupt Method, in F/S and HS Modes 2761
18-32 I²C Master Receiver Mode, Interrupt Method, in F/S and HS Modes 2762
18-33 I²C Master Transmitter Mode, DMA Method in F/S and HS Modes 2763
18-34 I²C Master Receiver Mode, DMA Method in F/S and HS Modes 2764
18-35 I²C Slave Transmitter/Receiver Mode, Polling 2765
18-36 I²C Slave Transmitter/Receiver Mode, Interrupt 2766
18-37 SCCB Setup Procedure 2768
18-38 SCCB Master Transmitter Mode, Polling 2769
18-39 SCCB Master Receiver Mode, Polling 2770
18-40 SCCB Master Transmitter Mode, Interrupt 2771
18-41 SCCB Master Receiver Mode, Interrupt 2772
18-42 Overview 2774
18-43 Environment 2775
18-44 Data Transfer Format 2775
18-45 Initial Configuration Flowchart 2781
18-46 Data Writing into the TWL4030 Companion Chip Register by the I2C1 Module 2782
18-47 Data Writing into the VS6650 Digital Camera Device Register by the I2C3 Module 2783
19-1 Multichannel Modules SPI1, SPI2, SPI3, and SPI4 2809
19-2 Typical Application Using the McSPI 2811
19-3 McSPI Master Mode (Full-Duplex) 2812
19-4 McSPI Master Single Mode (Receive-Only) 2812
19-5 McSPI Slave Mode (Full Duplex) 2813
19-6 McSPI Slave Single Mode (Transmit Only) 2813
19-7 McSPI Interface Signals in Master Mode 2814
19-8 McSPI Interface Signals in Slave Mode 2814
19-9 Phase and Polarity Combinations 2816
19-10 Full-Duplex Transfer Format With PHA = 0 2817
19-11 Extended SPI Transfer With a Start-Bit (SBE = 1) 2818
19-12 McSPI Integration 2819

19-13 McSPI Block Diagram 2823
19-14 SPI Full-Duplex Transmission (Example) 2825
19-15 Continuous Transfers With spim_csx Maintained Active (Single-Data-Pin Interface Mode) 2827
19-16 Continuous Transfers With spim_csx Maintained Active (Dual-Data-Pin Interface Mode) 2827
19-17 Chip-Select SPIEN Timing Controls 2828
19-18 Example of McSPI Slave With One Master and Multiple Slave Devices on Channel 0 2831
19-19 SPI Half-Duplex Transmission (Transmit-Only Slave) 2833
19-20 SPI Half-Duplex Transmission (Receive-Only Slave) 2834
19-21 Buffer Use in Transmit Direction Only 2834
19-22 Buffer Use in Receive Direction Only 2835
19-23 Buffer Used For Both Transmit/Receive Directions 2835
19-24 Buffer Almost Full Level (AFL) 2836
19-25 Buffer Almost Empty Level (AEL) 2836
19-26 Module Initialization Flow 2844
19-27 Common Transfer Sequence: Main Process 2845
19-28 Transmit and Receive (Master and Slave) 2847
19-29 Transmit-Only With Interrupts (Master and Slave) 2848
19-30 Transmit-Only With DMA (Master and Slave) 2849
19-31 Receive Only With Interrupt (Master Normal) 2850
19-32 Receive-Only With DMA (Master Normal) 2851
19-33 Receive-Only With Interrupt (Master Turbo) 2852
19-34 Receive-Only With DMA (Master Turbo) 2853
19-35 Receive Only (Slave) 2854
19-36 Two SPI Transfers With PHA = 0 (Flexibility of McSPI) 2855
19-37 Common Transfer Sequence/Main Process 2858
19-38 Transmit-Receive With Word Count 2860
19-39 Transmit-Receive Without Word Count 2861
19-40 Transmit-Only 2862
19-41 Receive-Only With Word Count 2863
19-42 Receive-Only Without Word Count 2864
19-43 Overview 2865
19-44 Environment 2866
19-45 McSPI Data Flow 2867
20-1 HDQ/1-Wire Highlight 2894
20-2 HDQ/1-Wire Typical Application System Overview 2895
20-3 HDQ Break-Pulse Timing Diagram 2896
20-4 1-Wire (SDQ) Reset Timing Diagram 2896
20-5 HDQ/1-Wire Transmitted Bit Timing 2897
20-6 HDQ/1-Wire Communication Sequence 2897
20-7 HDQ/1-Wire Integration 2898
20-8 HDQ/1-Wire Block Diagram 2900
20-9 Protocol Registers Description 2901
20-10 Environment 2910
20-11 HDQ/1-Wire Configuration in HDQ Mode 2910
20-12 Software Reset Flowchart 2911
21-1 McBSP Highlight 2920
21-2 SIDETONE Core Architecture 2922
21-3 Mode Overview of McBSP1 Module 2925
21-4 Mode Overview of McBSPi Module 2925
21-5 DBB Data Application 2926
21-6 Audio Data Application 2926
21-7 Voice Data Application 2927
21-8 McBSP Reception/Transmission Signal Activity 2928

21-9 Serial Data Formats 2929
21-10 TDM Data Format; Word Width: 32 Bits; Data Length: 24 Bits 2929
21-11 I2S Data Format; Word Width: 32 Bits; Data Length: 24 Bits 2930
21-12 Left Justified Data Format; Word Width: 32 Bits; Data Length: 24 Bits 2930
21-13 Right Justified Data Format; Word Width: 32 Bits; Data Length: 24 Bits 2930
21-14 PCM Protocol - Mode 1 Data Format 2931
21-15 PCM Protocol - Mode 2 Data Format 2931
21-16 McBSP1 Integration 2932
21-17 McBSP2 Integration 2933
21-18 McBSP3 Integration 2934
21-19 McBSP4 Integration 2935
21-20 McBSP5 Integration 2936
21-21 McBSP1, McBSP4 and McBSP5 Block Diagrams 2953
21-22 McBSP2 Block Diagram 2954
21-23 McBSP3 Block Diagram 2955
21-24 McBSP Data Transfer Paths 2956
21-25 McBSP2 Data Transfer Paths 2956
21-26 Conceptual Block Diagram for Clock and Frame Generation 2957
21-27 Clock Signal Control of Bit Transfer Timing 2958
21-28 McBSP Operating at Maximum Packet Frequency 2960
21-29 Single-Phase Frame for a McBSP Data Transfer 2962
21-30 Dual-Phase Frame for a McBSP Data Transfer 2963
21-31 McBSP Reception Physical Data Path 2963
21-32 McBSP Reception Signal Activity 2963
21-33 McBSP Transmission Physical Data Path 2964
21-34 McBSP Transmission Signal Activity 2964
21-35 Transmit Full Cycle Timing Diagram 2965
21-36 Transmit Half Cycle Timing Diagram 2966
21-37 Receive Full Cycle Timing Diagram 2966
21-38 Receive Half Cycle Timing Diagram 2966
21-39 Conceptual Block Diagram of the Sample Rate Generator 2967
21-40 CLKG Synchronization and FSG Generation (GSYNC = 1 and CLKGDV = 0x1) 2971
21-41 CLKG Synchronization and FSG Generation (GSYNC = 1 and CLKGDV = 0x3) 2971
21-42 Overrun in the McBSP Receiver 2973
21-43 Unexpected Frame-sync Pulse During a McBSP Reception 2973
21-44 Proper Positioning of Receive Frame-sync Pulses 2974
21-45 Unexpected Frame-sync Pulse During a McBSP Transmission 2975
21-46 Proper Positioning of Transmit Frame-sync Pulses 2976
21-47 McBSP Data Transfer in 8-Partition Mode 2979
21-48 Alternating Between Partitions A and B Channels 2980
21-49 Activity on McBSP Pins When XMCM=0b00 2982
21-50 Activity on McBSP Pins When XMCM=0b01 2982
21-51 Activity on McBSP Pins When XMCM=0b10 2982
21-52 Activity on McBSP Pins When XMCM=0b11 2983
21-53 SIDETONE Data Path 2984
21-54 McBSP to SIDETONE Data Exchange 2985
21-55 SIDETONE to McBSP Data Exchange 2985
21-56 SIDETONE Processed Data Interfaces 2986
21-57 Flow Diagram of McBSP Initialization Procedure for Master Mode 2989
21-58 Flow Diagram of McBSP Initialization Procedure for Slave Mode 2990
21-59 Flow Diagram for the SRG Registers Programmation 2993
21-60 Important Tasks to Configure the McBSP Receiver (Part 1) 2997
21-61 Important Tasks to Configure the McBSP Receiver (Part 2) 2998

21-62 Range of Programmable Data Delay 3000
21-63 2-Bit Data Delay Used to Skip a Framing Bit 3001
21-64 Data Externally Clocked on a Rising Edge and Sampled on a Falling Edge 3003
21-65 Frame of Period 16 CLKG Periods and Active Width of 2 CLKG Periods 3003
21-66 Important Tasks to Configure the McBSP Transmitter (Part 1) 3006
21-67 Important Tasks to Configure the McBSP Transmitter (Part 2) 3007
21-68 Range of Programmable Data Delay 3009
21-69 2-Bit Data Delay Used to Skip a Framing Bit 3009
21-70 Four 8-bit Data Words Transferred To/From McBSP Module 3014
21-71 One 32-bit Data Word Transferred To/From McBSP Module 3014
21-72 8-bit Data Words Transferred at Maximum Packet Frequency 3015
21-73 Configuring the Data Stream as a Continuous 32-bit Word 3015
21-74 Overview 3017
21-75 I2S Format 3017
22-1 MMC/SD/SDIO1 and 3 Overview 3077
22-2 MMC/SD/SDIO2 Overview 3078
22-3 MMC/SD/SDIO Connected to an MMC, an SD, or an SDIO Card Without External Transceiver 3080
22-4 MMC/SD/SDIO2 Connected to an MMC, an SD, or an SDIO Card with External Transceiver 3081
22-5 MMC/SD/SDIOi Interface Signals 3081
22-6 MMC/SD/SDIO2 Interface Signals 3082
22-7 Sequential Read Operation (MMC Cards Only) 3083
22-8 Sequential Write Operation (MMC Cards Only) 3083
22-9 Multiple Block Read Operation 3084
22-10 Multiple Block Write Operation with Card Busy Signal 3084
22-11 Command Token Format 3085
22-12 Response Token Format (R1, R3, R4, R5, R6) 3085
22-13 Response Token Format (R2) 3085
22-14 Data Token Format for 1-Bit Transfers 3086
22-15 Data Token Format for 4-Bit Transfers 3086
22-16 Data Token Format for 8-Bit Transfers 3087
22-17 MMC/SD/SDIO1 Integration 3088
22-18 DMA Receive Mode 3093
22-19 DMA Transmit Mode 3094
22-20 MMC/SD/SDIO Diagram 3097
22-21 Buffer Management for a Write 3100
22-22 Buffer Management for a Read 3101
22-23 MMC/SD/SDIO Controller Meta Initialization Steps 3105
22-24 MMC/SD/SDIO Controller Software Reset Flow 3106
22-25 MMC/SD/SDIO Controller Wake-Up Configuration 3107
22-26 MMC/SD/SDIO Controller Bus Configuration 3108
22-27 MMC/SD/SDIO Controller Card Identification and Selection - Part 1 3109
22-28 MMC/SD/SDIO Controller Card Identification and Selection - Part 2 3110
22-29 MMC/SD/SDIO Controller Read/Write Transfer Flow in DMA Mode with Interrupt 3111
22-30 MMC/SD/SDIO Controller Read/Write Transfer Flow in DMA Mode with Polling 3112
22-31 MMC/SD/SDIO Controller Read/Write Transfer Flow without DMA with Polling 3113
22-32 MMC/SD/SDIO Controller Read/Write in CE-ATA Mode 3114
22-33 MMC/SD/SDIO Controller Suspend Flow 3115
22-34 MMC/SD/SDIO Controller Resume Flow 3116
22-35 MMC/SD/SDIO Controller Command Transfer Flow with Polling 3117
22-36 MMC/SD/SDIO Controller Command Transfer Flow with Interrupts 3118
22-37 MMC/SD/SDIO Controller Clock Frequency Change Flow 3119
22-38 Overview 3120
22-39 Environment 3121

22-40 Command Transfer 3122
22-41 Data Read Transfer 3122
22-42 Data Write Transfer 3122
23-1 Memory Stick PRO Host Controller Overview 3174
24-1 USB Modules Overview 3176
24-2 High-Speed USB Controller Highlight 3177
24-3 High-Speed USB Controller Typical Application System 3179
24-4 High-Speed USB Controller Functional Interface Signals 3180
24-5 High-Speed USB Controller Integration 3181
24-6 High-Speed USB Controller 3187
24-7 High-Speed USB Host Subsystem Highlight 3199
24-8 USB Connection 3201
24-9 High-Speed USB Host Controller Connection—With and Without TLL 3202
24-10 High-Speed USB Host Controller Typical Application System – ULPI Interfaces 3203
24-11 High-Speed USB Host Subsystem Typical Application System - ULPI TLL Interfaces 3204
24-12 ULPI Interfaces – 12-Pin/8-Bit Data SDR Version 3205
24-13 ULPI TLL Interfaces –12-Pin/8-Bit Data SDR Version 3206
24-14 ULPI TLL Interfaces – 8-Pin/4-Bit Data DDR Version 3206
24-15 High-Speed USB Host Subsystem Functional Interface Signals 3207
24-16 High-Speed USB Host Subsystem Typical Application System 3210
24-17 Serial Interface Sideband Integration - Transceiver Configuration 3214
24-18 Serial Interface Sideband Integration - TLL Configuration 3215
24-19 6-Pin Unidirectional Using DAT/SE0 Signaling 3216
24-20 6-Pin Unidirectional Using DP/DM Signaling 3216
24-21 3-Pin Bidirectional Using DAT/SE0 Signaling 3217
24-22 4-Pin Bidirectional Using DP/DM Signaling 3217
24-23 6-Pin Unidirectional TLL Using DAT/SE0 Signaling 3218
24-24 6-Pin Unidirectional TLL Using DP/DM Signaling 3219
24-25 3-Pin Bidirectional TLL Using DAT/SE0 Signaling 3219
24-26 4-Pin Bidirectional TLL Using DP/DM Signaling 3220
24-27 2-Pin Bidirectional TLL Using DP/DM Encoding, With 4-Pin Bidirectional USB Device 3220
24-28 2-Pin Bidirectional TLL Using DAT/SE0 Encoding, With 3-Pin Bidirectional USB Device 3221
24-29 High-Speed USB Subsystem Integration 3225
24-30 High-Speed USB Host Controller Architecture 3235
24-31 USBTLL Channel 3238
24-32 Per-Configuration Datapath Through USBTLL 3240
24-33 Selecting and Configuring High-Speed USB Host Subsystem Connectivity 3246
25-1 General-Purpose Interface Overview 3327
25-2 General-Purpose Interface Typical Application System Overview 3328
25-3 General-Purpose Interface Used as a Keyboard Interface 3329
25-4 General-Purpose Interface Integration Overview 3331
25-5 General-Purpose Interface Description 3338
25-6 Synchronous Path 3338
25-7 Asynchronous Path 3339
25-8 Interrupt Request Generation 3340
25-9 Wake-Up Request Generation 3341
25-10 Write @GPIO_CLEARDATAOUT Register Example 3343
25-11 Write @GPIO_SETIRQENABLEx Register Example 3344
26-1 Initialization Process 3368
26-2 OMAP34xx and TWL4030 Power Connections 3369
26-3 Clock and Reset Environment 3371
26-4 Clock Interface 3372
26-5 ROM Code Architecture 3380

26-6 32KB ROM Memory Map 3381
26-7 64KB RAM Memory Map of GP Devices 3383
26-8 Overall Booting Sequence 3385
26-9 Device List Setup 3388
26-10 Common Peripheral Booting Protocol 3390
26-11 Peripheral Booting Procedure 3392
26-12 Customer USB Descriptor Selection 3397
26-13 Fast External Boot Procedure 3399
26-14 Memory Booting Procedure 3400
26-15 Detailed Memory Booting for Non-XIP Devices 3401
26-16 NAND Device Detection 3407
26-17 NAND ID2 Detection 3408
26-18 Bad NAND Invalid Block Detection 3410
26-19 OneNAND/Flex-OneNAND Read Sector 3412
26-20 MMC/SD Booting 3414
26-21 TWL5030 Connectivity Constraints to Support MMC/SD/eMMC/eSD Booting on MMC Port 1 and MMC Port 2 3415
26-22 TWL5030 Connectivity Constraints to Support MMC/SD/eMMC/eSD Booting on MMC Port 1 3416
26-23 MMC/SD Detection Procedure 3417
26-24 SD/MMC Booting 3419
26-25 MBR Detection Procedure 3421
26-26 Get MBR Partition 3422
26-27 Image Format 3425
26-28 CH Format 3426
26-29 CONTROL_SAVE_RESTORE_MEM Format 3434

List of Tables

1-1 Summary of Memories Supported by the POP Interface ... 185
1-2 Table of Features ... 186
1-3 CONTROL_FEATURE_OMAP_STATUS ... 187
1-4 Register Call Summary for Register CONTROL_FEATURE_OMAP_STATUS ... 187
1-5 CONTROL_PRODUCTION_ID ... 187
1-6 Register Call Summary for Register CONTROL_PRODUCTION_ID ... 188
1-7 Device Identification Registers ... 188
1-8 CONTROL_IDCODE Register Definition ... 188
1-9 Hawkeye Number Value ... 189
1-10 Revision Number Value ... 189
1-11 CONTROL_PRODUCTION_ID Register Silicon Type Identification ... 189
1-12 CONTROL_DIE_ID ... 189
2-1 Global Memory Space Mapping ... 195
2-2 L3 Control Register Mapping ... 197
2-3 L4-Core Memory Space Mapping ... 199
2-4 L4-Wakeup Memory Space Mapping ... 201
2-5 L4-Peripheral Memory Space Mapping ... 202
2-6 L4-Emulation Memory Space Mapping ... 203
2-7 Register Access Restrictions ... 205
2-8 L3 Interconnect View of the IVA2.2 Subsystem Memory Space ... 209
2-9 DSP View of the IVA2.2 Subsystem Memory Space ... 210
2-10 EDMA View of the IVA2.2 Subsystem Memory Space ... 211
3-1 MPU Clock Generator Clock Signals ... 219
3-2 MPU Subsystem Reset Signals ... 219
3-3 ARM Core Key Features ... 220
3-4 MPU Subsystem Clock Signal ... 222
3-5 ARM Reset Signals ... 222
3-6 Bridge Clock Signals ... 224
3-7 MPU Subsystem Reset Signal ... 224
3-8 Bridge Clock Signals ... 224
3-9 MPU Subsystem Reset Signal ... 224
3-10 Overview of the MPU Subsystem Power Domain ... 226
3-11 MPU Power States ... 226
3-12 MPU DPLL Power Modes ... 226
3-13 MPU Retention Modes ... 227
3-14 MPU Subsystem Operation Power Modes ... 227
3-15 Power Mode Allowable Transitions ... 230
4-1 States of a Clock Domain ... 238
4-2 External Clock Signals ... 248
4-3 External Reset Signals ... 250
4-4 Power Control Interface ... 250
4-5 PRCM Power Domains ... 254
4-6 PRCM Reset Signals ... 254
4-7 Global Reset Sources ... 258
4-8 Local Reset Sources ... 259
4-9 MPU Power Domain Reset Signal ... 261
4-10 NEON Power Domain Reset Signal ... 262
4-11 IVA2 Power Domain Reset Signals ... 262
4-12 CORE Power Domain Reset Signals ... 262

4-13 DSS Power Domain Reset Signal 263
4-14 CAM Power Domain Reset Signal 263
4-15 USBHOST Power Domain Reset Signal 263
4-16 SGX Power Domain Reset Signal 263
4-17 WKUP Power Domain Reset Signals 263
4-18 PER Power Domain Reset Signal 264
4-19 DPLL Power Domain Reset Signals 264
4-20 EFUSE Power Domain Reset Signal 264
4-21 BANDGAP Logic Reset Signal 264
4-22 Global Reset Summary 271
4-23 Local Reset Summary 272
4-24 Power Domain Modules 291
4-25 Power Domain States 293
4-26 Domain Power Control Summary 294
4-27 System Clock Input Configurations 300
4-28 External Clock I/Os 300
4-29 DPLL Output Clocks 312
4-30 Source-Clock Summary 312
4-31 Clock Distribution 326
4-32 Peripheral Module Functional Clock Frequencies 328
4-33 sys_clkreq Pad Direction Control 329
4-34 System Clock Operation Modes 330
4-35 Oscillator Controls 331
4-36 DPLL Multiplier and Divider Factors 332
4-37 DPLL Power Modes 333
4-38 DPLL Power Mode Support 333
4-39 DPLL Power Mode Control 334
4-40 LP Mode Control 334
4-41 Clock Path Power-Down Control 335
4-42 DPLL Recalibration Controls 336
4-43 Common PRM Source-Clock Gating Controls 338
4-44 Common CM Source-Clock Gating Controls 341
4-45 Common Interface Clock-Gating Controls 341
4-46 DPLL Power Domain Clock-Gating Controls 342
4-47 SGX Power Domain Clock-Gating Controls 343
4-48 CORE Power Domain Clock-Gating Controls 345
4-49 EFUSE Power Domain Clock-Gating Control 346
4-50 DSS Power Domain Clock-Gating Controls 347
4-51 CAM Power Domain Clock-Gating Controls 348
4-52 USBHOST Power Domain Clock-Gating Controls 348
4-53 WKUP Power Domain Clock-Gating Controls 349
4-54 PER Power Domain Clock-Gating Controls 352
4-55 Processor Clock Configuration Controls 353
4-56 Processor Clock Configurations 354
4-57 Interface Clock Configuration Controls 354
4-58 Functional Clock Configuration Controls 355
4-59 Power-State-Related Sleep Transition Actions 358
4-60 MPU Power Domain Wake-Up Events 359
4-61 NEON Power Domain Wake-Up Events 360
4-62 IVA2 Power Domain Wake-Up Events 360
4-63 SGX Power Domain Wake-Up Events 361

4-64 CORE Power Domain Wake-Up Events 361
4-65 DSS Power Domain Wake-Up Events 362
4-66 CAM Power Domain Wake-Up Events 362
4-67 USBHOST Power Domain Wake-Up Events 362
4-68 PER Power Domain Wake-Up Events 362
4-69 EMU Power Domain Wake-Up Events 363
4-70 WKUP Power Domain Wake-Up Events 363
4-71 Clock Domain Mute Conditions 364
4-72 Sleep Dependencies 366
4-73 Wake-Up Dependencies 368
4-74 Interrupt Descriptions 372
4-75 MPU Interrupt Event Descriptions 372
4-76 IVA2 Interrupt Event Descriptions 373
4-77 Voltage Domain Controls Summary 376
4-78 VDD1 Voltage Domain Dependencies 377
4-79 VDD2 Voltage Domain Dependencies 378
4-80 Remaining Voltage Domain Dependencies 378
4-81 GFX Functional Clock Ratio Settings 399
4-82 Interface Clock Autoidle Settings 400
4-83 Clock State Transition Settings 401
4-84 Sleep Dependency Settings 403
4-85 VDD1 and VDD2 Voltage Control Through VMODE 427
4-86 CM Instance Summary 430
4-87 IVA2_CM Register Summary 430
4-88 CM_FCLKEN_IVA2 431
4-89 Register Call Summary for Register CM_FCLKEN_IVA2 431
4-90 CM_CLKEN_PLL_IVA2 431
4-91 Register Call Summary for Register CM_CLKEN_PLL_IVA2 432
4-92 CM_IDLEST_IVA2 432
4-93 Register Call Summary for Register CM_IDLEST_IVA2 433
4-94 CM_IDLEST_PLL_IVA2 433
4-95 Register Call Summary for Register CM_IDLEST_PLL_IVA2 433
4-96 CM_AUTOIDLE_PLL_IVA2 433
4-97 Register Call Summary for Register CM_AUTOIDLE_PLL_IVA2 434
4-98 CM_CLKSEL1_PLL_IVA2 434
4-99 Register Call Summary for Register CM_CLKSEL1_PLL_IVA2 434
4-100 CM_CLKSEL2_PLL_IVA2 435
4-101 Register Call Summary for Register CM_CLKSEL2_PLL_IVA2 435
4-102 CM_CLKSTCTRL_IVA2 436
4-103 Register Call Summary for Register CM_CLKSTCTRL_IVA2 436
4-104 CM_CLKSTST_IVA2 436
4-105 Register Call Summary for Register CM_CLKSTST_IVA2 437
4-106 OCP_System_Reg_CM Register Summary 437
4-107 CM_REVISION 437
4-108 Register Call Summary for Register CM_REVISION 437
4-109 CM_SYSCONFIG 438
4-110 Register Call Summary for Register CM_SYSCONFIG 438
4-111 MPU_CM Register Summary 438
4-112 CM_CLKEN_PLL_MPU 438
4-113 Register Call Summary for Register CM_CLKEN_PLL_MPU 439
4-114 CM_IDLEST_MPU 440

4-115 Register Call Summary for Register CM_IDLEST_MPU 440
4-116 CM_IDLEST_PLL_MPU 440
4-117 Register Call Summary for Register CM_IDLEST_PLL_MPU 441
4-118 CM_AUTOIDLE_PLL_MPU 441
4-119 Register Call Summary for Register CM_AUTOIDLE_PLL_MPU 441
4-120 CM_CLKSEL1_PLL_MPU 441
4-121 Register Call Summary for Register CM_CLKSEL1_PLL_MPU 442
4-122 CM_CLKSEL2_PLL_MPU 442
4-123 Register Call Summary for Register CM_CLKSEL2_PLL_MPU 443
4-124 CM_CLKSTCTRL_MPU 443
4-125 Register Call Summary for Register CM_CLKSTCTRL_MPU 444
4-126 CM_CLKSTST_MPU 444
4-127 Register Call Summary for Register CM_CLKSTST_MPU 444
4-128 CORE_CM Register Summary 444
4-129 CM_FCLKEN1_CORE 445
4-130 Register Call Summary for Register CM_FCLKEN1_CORE 446
4-131 CM_FCLKEN3_CORE 447
4-132 Register Call Summary for Register CM_FCLKEN3_CORE 447
4-133 CM_ICLKEN1_CORE 447
4-134 Register Call Summary for Register CM_ICLKEN1_CORE 449
4-135 CM_ICLKEN3_CORE 449
4-136 Register Call Summary for Register CM_ICLKEN3_CORE 450
4-137 CM_IDLEST1_CORE 450
4-138 Register Call Summary for Register CM_IDLEST1_CORE 452
4-139 CM_IDLEST3_CORE 452
4-140 Register Call Summary for Register CM_IDLEST3_CORE 453
4-141 CM_AUTOIDLE1_CORE 453
4-142 Register Call Summary for Register CM_AUTOIDLE1_CORE 455
4-143 CM_AUTOIDLE3_CORE 456
4-144 Register Call Summary for Register CM_AUTOIDLE3_CORE 456
4-145 CM_CLKSEL_CORE 456
4-146 Register Call Summary for Register CM_CLKSEL_CORE 457
4-147 CM_CLKSTCTRL_CORE 457
4-148 Register Call Summary for Register CM_CLKSTCTRL_CORE 458
4-149 CM_CLKSTST_CORE 458
4-150 Register Call Summary for Register CM_CLKSTST_CORE 458
4-151 SGX_CM Register Summary 458
4-152 CM_FCLKEN_SGX 459
4-153 Register Call Summary for Register CM_FCLKEN_SGX 459
4-154 CM_ICLKEN_SGX 459
4-155 Register Call Summary for Register CM_ICLKEN_SGX 460
4-156 CM_IDLEST_SGX 460
4-157 Register Call Summary for Register CM_IDLEST_SGX 460
4-158 CM_CLKSEL_SGX 460
4-159 Register Call Summary for Register CM_CLKSEL_SGX 461
4-160 CM_SLEEPDEP_SGX 461
4-161 Register Call Summary for Register CM_SLEEPDEP_SGX 461
4-162 CM_CLKSTCTRL_SGX 461
4-163 Register Call Summary for Register CM_CLKSTCTRL_SGX 462
4-164 CM_CLKSTST_SGX 462
4-165 Register Call Summary for Register CM_CLKSTST_SGX 462

4-166 WKUP_CM Register Summary 463
4-167 CM_FCLKEN_WKUP 463
4-168 Register Call Summary for Register CM_FCLKEN_WKUP 463
4-169 CM_ICLKEN_WKUP 464
4-170 Register Call Summary for Register CM_ICLKEN_WKUP 464
4-171 CM_IDLEST_WKUP 465
4-172 Register Call Summary for Register CM_IDLEST_WKUP 465
4-173 CM_AUTOIDLE_WKUP 466
4-174 Register Call Summary for Register CM_AUTOIDLE_WKUP 466
4-175 CM_CLKSEL_WKUP 467
4-176 Register Call Summary for Register CM_CLKSEL_WKUP 467
4-177 Clock_Control_Reg_CM Register Summary 467
4-178 CM_CLKEN_PLL 468
4-179 Register Call Summary for Register CM_CLKEN_PLL 470
4-180 CM_CLKEN2_PLL 470
4-181 Register Call Summary for Register CM_CLKEN2_PLL 471
4-182 CM_IDLEST_CKGEN 471
4-183 Register Call Summary for Register CM_IDLEST_CKGEN 472
4-184 CM_IDLEST2_CKGEN 473
4-185 Register Call Summary for Register CM_IDLEST2_CKGEN 473
4-186 CM_AUTOIDLE_PLL 473
4-187 Register Call Summary for Register CM_AUTOIDLE_PLL 474
4-188 CM_AUTOIDLE2_PLL 474
4-189 Register Call Summary for Register CM_AUTOIDLE2_PLL 474
4-190 CM_CLKSEL1_PLL 475
4-191 Register Call Summary for Register CM_CLKSEL1_PLL 477
4-192 CM_CLKSEL2_PLL 477
4-193 Register Call Summary for Register CM_CLKSEL2_PLL 477
4-194 CM_CLKSEL3_PLL 477
4-195 Register Call Summary for Register CM_CLKSEL3_PLL 478
4-196 CM_CLKSEL4_PLL 478
4-197 Register Call Summary for Register CM_CLKSEL4_PLL 478
4-198 CM_CLKSEL5_PLL 479
4-199 Register Call Summary for Register CM_CLKSEL5_PLL 479
4-200 CM_CLKOUT_CTRL 479
4-201 Register Call Summary for Register CM_CLKOUT_CTRL 480
4-202 DSS_CM Register Summary 480
4-203 CM_FCLKEN_DSS 481
4-204 Register Call Summary for Register CM_FCLKEN_DSS 481
4-205 CM_ICLKEN_DSS 481
4-206 Register Call Summary for Register CM_ICLKEN_DSS 482
4-207 CM_IDLEST_DSS 482
4-208 Register Call Summary for Register CM_IDLEST_DSS 482
4-209 CM_AUTOIDLE_DSS 482
4-210 Register Call Summary for Register CM_AUTOIDLE_DSS 483
4-211 CM_CLKSEL_DSS 483
4-212 Register Call Summary for Register CM_CLKSEL_DSS 484
4-213 CM_SLEEPDEP_DSS 484
4-214 Register Call Summary for Register CM_SLEEPDEP_DSS 485
4-215 CM_CLKSTCTRL_DSS 485
4-216 Register Call Summary for Register CM_CLKSTCTRL_DSS 485

4-217 CM_CLKSTST_DSS 485
4-218 Register Call Summary for Register CM_CLKSTST_DSS 486
4-219 CAM_CM Register Summary 486
4-220 CM_FCLKEN_CAM 486
4-221 Register Call Summary for Register CM_FCLKEN_CAM 487
4-222 CM_ICLKEN_CAM 487
4-223 Register Call Summary for Register CM_ICLKEN_CAM 487
4-224 CM_IDLEST_CAM 488
4-225 Register Call Summary for Register CM_IDLEST_CAM 488
4-226 CM_AUTOIDLE_CAM 488
4-227 Register Call Summary for Register CM_AUTOIDLE_CAM 488
4-228 CM_CLKSEL_CAM 489
4-229 Register Call Summary for Register CM_CLKSEL_CAM 489
4-230 CM_SLEEPDEP_CAM 489
4-231 Register Call Summary for Register CM_SLEEPDEP_CAM 490
4-232 CM_CLKSTCTRL_CAM 490
4-233 Register Call Summary for Register CM_CLKSTCTRL_CAM 490
4-234 CM_CLKSTST_CAM 491
4-235 Register Call Summary for Register CM_CLKSTST_CAM 491
4-236 PER_CM Register Summary 491
4-237 CM_FCLKEN_PER 492
4-238 Register Call Summary for Register CM_FCLKEN_PER 493
4-239 CM_ICLKEN_PER 493
4-240 Register Call Summary for Register CM_ICLKEN_PER 495
4-241 CM_IDLEST_PER 495
4-242 Register Call Summary for Register CM_IDLEST_PER 496
4-243 CM_AUTOIDLE_PER 497
4-244 Register Call Summary for Register CM_AUTOIDLE_PER 498
4-245 CM_CLKSEL_PER 499
4-246 Register Call Summary for Register CM_CLKSEL_PER 500
4-247 CM_SLEEPDEP_PER 500
4-248 Register Call Summary for Register CM_SLEEPDEP_PER 500
4-249 CM_CLKSTCTRL_PER 500
4-250 Register Call Summary for Register CM_CLKSTCTRL_PER 501
4-251 CM_CLKSTST_PER 501
4-252 Register Call Summary for Register CM_CLKSTST_PER 501
4-253 EMU_CM Register Summary 502
4-254 CM_CLKSEL1_EMU 502
4-255 Register Call Summary for Register CM_CLKSEL1_EMU 505
4-256 CM_CLKSTCTRL_EMU 505
4-257 Register Call Summary for Register CM_CLKSTCTRL_EMU 506
4-258 CM_CLKSTST_EMU 506
4-259 Register Call Summary for Register CM_CLKSTST_EMU 506
4-260 CM_CLKSEL2_EMU 506
4-261 Register Call Summary for Register CM_CLKSEL2_EMU 507
4-262 CM_CLKSEL3_EMU 507
4-263 Register Call Summary for Register CM_CLKSEL3_EMU 507
4-264 Global_Reg_CM Register Summary 507
4-265 CM_POLCTRL 508
4-266 Register Call Summary for Register CM_POLCTRL 508
4-267 NEON_CM Register Summary 508

4-268 CM_IDLEST_NEON 508
4-269 Register Call Summary for Register CM_IDLEST_NEON 509
4-270 CM_CLKSTCTRL_NEON 509
4-271 Register Call Summary for Register CM_CLKSTCTRL_NEON 509
4-272 USBHOST_CM Register Summary 510
4-273 CM_FCLKEN_USBHOST 510
4-274 Register Call Summary for Register CM_FCLKEN_USBHOST 510
4-275 CM_ICLKEN_USBHOST 511
4-276 Register Call Summary for Register CM_ICLKEN_USBHOST 511
4-277 CM_IDLEST_USBHOST 511
4-278 Register Call Summary for Register CM_IDLEST_USBHOST 512
4-279 CM_AUTOIDLE_USBHOST 512
4-280 Register Call Summary for Register CM_AUTOIDLE_USBHOST 512
4-281 CM_SLEEPDEP_USBHOST 512
4-282 Register Call Summary for Register CM_SLEEPDEP_USBHOST 513
4-283 CM_CLKSTCTRL_USBHOST 513
4-284 Register Call Summary for Register CM_CLKSTCTRL_USBHOST 513
4-285 CM_CLKSTST_USBHOST 514
4-286 Register Call Summary for Register CM_CLKSTST_USBHOST 514
4-287 PRM Instance Summary 514
4-288 IVA2_PRM Register Summary 515
4-289 RM_RSTCTRL_IVA2 515
4-290 Register Call Summary for Register RM_RSTCTRL_IVA2 516
4-291 RM_RSTST_IVA2 516
4-292 Register Call Summary for Register RM_RSTST_IVA2 518
4-293 PM_WKDEP_IVA2 518
4-294 Register Call Summary for Register PM_WKDEP_IVA2 519
4-295 PM_PWSTCTRL_IVA2 519
4-296 Register Call Summary for Register PM_PWSTCTRL_IVA2 520
4-297 PM_PWSTST_IVA2 521
4-298 Register Call Summary for Register PM_PWSTST_IVA2 522
4-299 PM_PREPWSTST_IVA2 522
4-300 Register Call Summary for Register PM_PREPWSTST_IVA2 523
4-301 PRM_IRQSTATUS_IVA2 523
4-302 Register Call Summary for Register PRM_IRQSTATUS_IVA2 524
4-303 PRM_IRQENABLE_IVA2 524
4-304 Register Call Summary for Register PRM_IRQENABLE_IVA2 525
4-305 OCP_System_Reg_PRM Register Summary 525
4-306 PRM_REVISION 525
4-307 Register Call Summary for Register PRM_REVISION 525
4-308 PRM_SYSCONFIG 526
4-309 Register Call Summary for Register PRM_SYSCONFIG 526
4-310 PRM_IRQSTATUS_MPU 526
4-311 Register Call Summary for Register PRM_IRQSTATUS_MPU 528
4-312 PRM_IRQENABLE_MPU 529
4-313 Register Call Summary for Register PRM_IRQENABLE_MPU 530
4-314 MPU_PRM Register Summary 530
4-315 RM_RSTST_MPU 531
4-316 Register Call Summary for Register RM_RSTST_MPU 532
4-317 PM_WKDEP_MPU 532
4-318 Register Call Summary for Register PM_WKDEP_MPU 533

4-319 PM_EVGENCTRL_MPU 533
4-320 Register Call Summary for Register PM_EVGENCTRL_MPU 533
4-321 PM_EVGENONTIM_MPU 534
4-322 Register Call Summary for Register PM_EVGENONTIM_MPU 534
4-323 PM_EVGENOFFTIM_MPU 534
4-324 Register Call Summary for Register PM_EVGENOFFTIM_MPU 534
4-325 PM_PWSTCTRL_MPU 535
4-326 Register Call Summary for Register PM_PWSTCTRL_MPU 535
4-327 PM_PWSTST_MPU 536
4-328 Register Call Summary for Register PM_PWSTST_MPU 536
4-329 PM_PREPWSTST_MPU 537
4-330 Register Call Summary for Register PM_PREPWSTST_MPU 537
4-331 CORE_PRM Register Summary 537
4-332 RM_RSTST_CORE 538
4-333 Register Call Summary for Register RM_RSTST_CORE 539
4-334 PM_WKEN1_CORE 539
4-335 Register Call Summary for Register PM_WKEN1_CORE 540
4-336 PM_MPUGRPSEL1_CORE 540
4-337 Register Call Summary for Register PM_MPUGRPSEL1_CORE 542
4-338 PM_IVA2GRPSEL1_CORE 543
4-339 Register Call Summary for Register PM_IVA2GRPSEL1_CORE 545
4-340 PM_WKST1_CORE 545
4-341 Register Call Summary for Register PM_WKST1_CORE 547
4-342 PM_WKST3_CORE 547
4-343 Register Call Summary for Register PM_WKST3_CORE 548
4-344 PM_PWSTCTRL_CORE 548
4-345 Register Call Summary for Register PM_PWSTCTRL_CORE 549
4-346 PM_PWSTST_CORE 550
4-347 Register Call Summary for Register PM_PWSTST_CORE 550
4-348 PM_PREPWSTST_CORE 551
4-349 Register Call Summary for Register PM_PREPWSTST_CORE 551
4-350 PM_WKEN3_CORE 552
4-351 Register Call Summary for Register PM_WKEN3_CORE 552
4-352 PM_IVA2GRPSEL3_CORE 552
4-353 Register Call Summary for Register PM_IVA2GRPSEL3_CORE 552
4-354 PM_MPUGRPSEL3_CORE 553
4-355 Register Call Summary for Register PM_MPUGRPSEL3_CORE 553
4-356 SGX_PRM Register Summary 553
4-357 RM_RSTST_SGX 554
4-358 Register Call Summary for Register RM_RSTST_SGX 554
4-359 PM_WKDEP_SGX 555
4-360 Register Call Summary for Register PM_WKDEP_SGX 555
4-361 PM_PWSTCTRL_SGX 555
4-362 Register Call Summary for Register PM_PWSTCTRL_SGX 556
4-363 PM_PWSTST_SGX 556
4-364 Register Call Summary for Register PM_PWSTST_SGX 557
4-365 PM_PREPWSTST_SGX 557
4-366 Register Call Summary for Register PM_PREPWSTST_SGX 557
4-367 WKUP_PRM Register Summary 557
4-368 PM_WKEN_WKUP 558
4-369 Register Call Summary for Register PM_WKEN_WKUP 558

4-370 PM_MPUGRPSEL_WKUP 559
4-371 Register Call Summary for Register PM_MPUGRPSEL_WKUP 559
4-372 PM_IVA2GRPSEL_WKUP 560
4-373 Register Call Summary for Register PM_IVA2GRPSEL_WKUP 560
4-374 PM_WKST_WKUP 561
4-375 Register Call Summary for Register PM_WKST_WKUP 561
4-376 Clock_Control_Reg_PRM Register Summary 562
4-377 PRM_CLKSEL 562
4-378 Register Call Summary for Register PRM_CLKSEL 562
4-379 PRM_CLKOUT_CTRL 563
4-380 Register Call Summary for Register PRM_CLKOUT_CTRL 563
4-381 DSS_PRM Register Summary 563
4-382 RM_RSTST_DSS 564
4-383 Register Call Summary for Register RM_RSTST_DSS 564
4-384 PM_WKEN_DSS 565
4-385 Register Call Summary for Register PM_WKEN_DSS 565
4-386 PM_WKDEP_DSS 565
4-387 Register Call Summary for Register PM_WKDEP_DSS 566
4-388 PM_PWSTCTRL_DSS 566
4-389 Register Call Summary for Register PM_PWSTCTRL_DSS 566
4-390 PM_PWSTST_DSS 567
4-391 Register Call Summary for Register PM_PWSTST_DSS 567
4-392 PM_PREPWSTST_DSS 567
4-393 Register Call Summary for Register PM_PREPWSTST_DSS 568
4-394 CAM_PRM Register Summary 568
4-395 RM_RSTST_CAM 568
4-396 Register Call Summary for Register RM_RSTST_CAM 569
4-397 PM_WKDEP_CAM 569
4-398 Register Call Summary for Register PM_WKDEP_CAM 570
4-399 PM_PWSTCTRL_CAM 570
4-400 Register Call Summary for Register PM_PWSTCTRL_CAM 570
4-401 PM_PWSTST_CAM 571
4-402 Register Call Summary for Register PM_PWSTST_CAM 571
4-403 PM_PREPWSTST_CAM 571
4-404 Register Call Summary for Register PM_PREPWSTST_CAM 572
4-405 PER_PRM Register Summary 572
4-406 RM_RSTST_PER 572
4-407 Register Call Summary for Register RM_RSTST_PER 573
4-408 PM_WKEN_PER 573
4-409 Register Call Summary for Register PM_WKEN_PER 575
4-410 PM_MPUGRPSEL_PER 575
4-411 Register Call Summary for Register PM_MPUGRPSEL_PER 577
4-412 PM_IVA2GRPSEL_PER 577
4-413 Register Call Summary for Register PM_IVA2GRPSEL_PER 579
4-414 PM_WKST_PER 579
4-415 Register Call Summary for Register PM_WKST_PER 581
4-416 PM_WKDEP_PER 582
4-417 Register Call Summary for Register PM_WKDEP_PER 582
4-418 PM_PWSTCTRL_PER 583
4-419 Register Call Summary for Register PM_PWSTCTRL_PER 583
4-420 PM_PWSTST_PER 583

4-421 Register Call Summary for Register PM_PWSTST_PER 584
4-422 PM_PREPWSTST_PER 584
4-423 Register Call Summary for Register PM_PREPWSTST_PER 585
4-424 EMU_PRM Register Summary 585
4-425 RM_RSTST_EMU 585
4-426 Register Call Summary for Register RM_RSTST_EMU 586
4-427 PM_PWSTST_EMU 586
4-428 Register Call Summary for Register PM_PWSTST_EMU 586
4-429 Global_Reg_PRM Register Summary 587
4-430 PRM_VC_SMPS_SA 587
4-431 Register Call Summary for Register PRM_VC_SMPS_SA 588
4-432 PRM_VC_SMPS_VOL_RA 588
4-433 Register Call Summary for Register PRM_VC_SMPS_VOL_RA 588
4-434 PRM_VC_SMPS_CMD_RA 588
4-435 Register Call Summary for Register PRM_VC_SMPS_CMD_RA 589
4-436 PRM_VC_CMD_VAL_0 589
4-437 Register Call Summary for Register PRM_VC_CMD_VAL_0 589
4-438 PRM_VC_CMD_VAL_1 589
4-439 Register Call Summary for Register PRM_VC_CMD_VAL_1 590
4-440 PRM_VC_CH_CONF 590
4-441 Register Call Summary for Register PRM_VC_CH_CONF 591
4-442 PRM_VC_I2C_CFG 591
4-443 Register Call Summary for Register PRM_VC_I2C_CFG 591
4-444 PRM_VC_BYPASS_VAL 592
4-445 Register Call Summary for Register PRM_VC_BYPASS_VAL 592
4-446 PRM_RSTCTRL 592
4-447 Register Call Summary for Register PRM_RSTCTRL 593
4-448 PRM_RSTTIME 593
4-449 Register Call Summary for Register PRM_RSTTIME 593
4-450 PRM_RSTST 594
4-451 Register Call Summary for Register PRM_RSTST 595
4-452 PRM_VOLTCTRL 595
4-453 Register Call Summary for Register PRM_VOLTCTRL 596
4-454 PRM_SRAM_PCHARGE 596
4-455 Register Call Summary for Register PRM_SRAM_PCHARGE 597
4-456 PRM_CLKSRC_CTRL 597
4-457 Register Call Summary for Register PRM_CLKSRC_CTRL 598
4-458 PRM_OBS 598
4-459 Register Call Summary for Register PRM_OBS 598
4-460 PRM_VOLTSETUP1 598
4-461 Register Call Summary for Register PRM_VOLTSETUP1 599
4-462 PRM_VOLTOFFSET 599
4-463 Register Call Summary for Register PRM_VOLTOFFSET 599
4-464 PRM_CLKSETUP 599
4-465 Register Call Summary for Register PRM_CLKSETUP 600
4-466 PRM_POLCTRL 600
4-467 Register Call Summary for Register PRM_POLCTRL 600
4-468 PRM_VOLTSETUP2 601
4-469 Register Call Summary for Register PRM_VOLTSETUP2 601
4-470 NEON_PRM Register Summary 601
4-471 RM_RSTST_NEON 601

4-472 Register Call Summary for Register RM_RSTST_NEON 602
4-473 PM_WKDEP_NEON 603
4-474 Register Call Summary for Register PM_WKDEP_NEON 603
4-475 PM_PWSTCTRL_NEON 603
4-476 Register Call Summary for Register PM_PWSTCTRL_NEON 604
4-477 PM_PWSTST_NEON 604
4-478 Register Call Summary for Register PM_PWSTST_NEON 604
4-479 PM_PREPWSTST_NEON 604
4-480 Register Call Summary for Register PM_PREPWSTST_NEON 605
4-481 USBHOST_PRM Register Summary 605
4-482 RM_RSTST_USBHOST 606
4-483 Register Call Summary for Register RM_RSTST_USBHOST 606
4-484 PM_WKEN_USBHOST 607
4-485 Register Call Summary for Register PM_WKEN_USBHOST 607
4-486 PM_MPUGRPSEL_USBHOST 607
4-487 Register Call Summary for Register PM_MPUGRPSEL_USBHOST 608
4-488 PM_IVA2GRPSEL_USBHOST 608
4-489 Register Call Summary for Register PM_IVA2GRPSEL_USBHOST 608
4-490 PM_WKST_USBHOST 608
4-491 Register Call Summary for Register PM_WKST_USBHOST 609
4-492 PM_WKDEP_USBHOST 609
4-493 Register Call Summary for Register PM_WKDEP_USBHOST 610
4-494 PM_PWSTCTRL_USBHOST 610
4-495 Register Call Summary for Register PM_PWSTCTRL_USBHOST 610
4-496 PM_PWSTST_USBHOST 611
4-497 Register Call Summary for Register PM_PWSTST_USBHOST 611
4-498 PM_PREPWSTST_USBHOST 612
4-499 Register Call Summary for Register PM_PREPWSTST_USBHOST 612
5-1 MCmd Qualifier Description 615
5-2 MReqInfo Qualifier Description 615
5-3 SResp Qualifier Description 615
5-4 L3 Initiator Agents 618
5-5 L3 Target Agents 618
5-6 L4-Core Initiator Agent 620
5-7 L4-Core Target Agents 620
5-8 L4-Per Initiator Agent 620
5-9 L4-Per Target Agents 621
5-10 L4-Emu Initiator Agents 621
5-11 L4-Emu Target Agents 621
5-12 L4-Wakeup Initiator Agent 622
5-13 L4-Wakeup Target Agents 622
5-14 Connectivity Matrix 622
5-15 L3 Interconnect Clocks 624
5-16 L3 Interconnect Reset 624
5-17 L3 Interconnect Power Domain 624
5-18 L3 Interconnect Hardware Requests 625
5-19 InitiatorID Definition 625
5-20 Target Firewall and Region Configuration 626
5-21 L3 Firewall Size Parameter Definition 629
5-22 MReqInfo Parameter Combinations 631
5-23 L3 Firewall Permission-Setting Registers 632

5-24 L3 Firewall Error Logging Registers 633
5-25 Error Types 637
5-26 CODE Field Definition 637
5-27 L3 Timeout Register Target and Agent Programming 638
5-28 L3 External Input Flags 640
5-29 L3_SI_FLAG_STATUS_0 for Application Error 641
5-30 L3_SI_FLAG_STATUS_1 for Debug Error 642
5-31 Error Clearing 646
5-32 MReqInfo Parameter Example 647
5-33 Instance Summary 650
5-34 Initiator Agent Common Register Summary 651
5-35 Initiator Agent Common Register Summary 651
5-36 Initiator Agent Common Register Summary 651
5-37 Initiator Agent Common Register Summary 651
5-38 L3_IA_COMPONENT 652
5-39 Register Call Summary for Register L3_IA_COMPONENT 652
5-40 L3_IA_CORE 652
5-41 Register Call Summary for Register L3_IA_CORE 652
5-42 L3_IA_AGENT_CONTROL 652
5-43 Register Call Summary for Register L3_IA_AGENT_CONTROL 654
5-44 L3_IA_AGENT_STATUS 654
5-45 Register Call Summary for Register L3_IA_AGENT_STATUS 655
5-46 L3_IA_ERROR_LOG 655
5-47 Register Call Summary for Register L3_IA_ERROR_LOG 656
5-48 L3_IA_ERROR_LOG_ADDR 656
5-49 Register Call Summary for Register L3_IA_ERROR_LOG_ADDR 656
5-50 Target Agent Common Register Summary 657
5-51 Target Agent Common Register Summary 657
5-52 Target Agent Common Register Summary 657
5-53 Target Agent Common Register Summary 657
5-54 L3_TA_COMPONENT 658
5-55 Register Call Summary for Register L3_TA_COMPONENT 658
5-56 L3_TA_CORE 658
5-57 Register Call Summary for Register L3_TA_CORE 658
5-58 L3_TA_AGENT_CONTROL 659
5-59 Register Call Summary for Register L3_TA_AGENT_CONTROL 659
5-60 L3_TA_AGENT_STATUS 660
5-61 Register Call Summary for Register L3_TA_AGENT_STATUS 661
5-62 L3_TA_ERROR_LOG 661
5-63 Register Call Summary for Register L3_TA_ERROR_LOG 661
5-64 L3_TA_ERROR_LOG_ADDR 661
5-65 Register Call Summary for Register L3_TA_ERROR_LOG_ADDR 662
5-66 RT Register Summary 662
5-67 L3_RT_COMPONENT 662
5-68 Register Call Summary for Register L3_RT_COMPONENT 663
5-69 L3_RT_NETWORK 663
5-70 Register Call Summary for Register L3_RT_NETWORK 663
5-71 L3_RT_INITID_READBACK 663
5-72 Register Call Summary for Register L3_RT_INITID_READBACK 663
5-73 L3_RT_NETWORK_CONTROL 664
5-74 Register Call Summary for Register L3_RT_NETWORK_CONTROL 664

5-75 Protection Mechanism Common Register Summary ... 665
5-76 Protection Mechanism Common Register Summary ... 665
5-77 Protection Mechanism Common Register Summary ... 665
5-78 L3_PM_ERROR_LOG ... 666
5-79 Register Call Summary for Register L3_PM_ERROR_LOG ... 666
5-80 L3_PM_CONTROL ... 667
5-81 Register Call Summary for Register L3_PM_CONTROL ... 667
5-82 L3_PM_ERROR_CLEAR_SINGLE ... 667
5-83 Register Call Summary for Register L3_PM_ERROR_CLEAR_SINGLE ... 668
5-84 L3_PM_ERROR_CLEAR_MULTI ... 668
5-85 Register Call Summary for Register L3_PM_ERROR_CLEAR_MULTI ... 668
5-86 L3_PM_REQ_INFO_PERMISSION_i ... 668
5-87 Register Call Summary for Register L3_PM_REQ_INFO_PERMISSION_i ... 669
5-88 Reset Value for REQ_INFO_PERMISSION ... 669
5-89 L3_PM_READ_PERMISSION_i ... 669
5-90 Register Call Summary for Register L3_PM_READ_PERMISSION_i ... 670
5-91 L3_PM_WRITE_PERMISSION_i ... 670
5-92 Register Call Summary for Register L3_PM_WRITE_PERMISSION_i ... 671
5-93 Bit Availability and Initialization Values for L3_PM_READ_PERMISSION_i and L3_PM_WRITE_PERMISSION_i ... 672
5-94 L3_PM_ADDR_MATCH_k ... 673
5-95 Register Call Summary for Register L3_PM_ADDR_MATCH_k ... 673
5-96 Reset Value for L3_PM_ADDR_MATCH_k ... 673
5-97 SI Register Summary ... 674
5-98 L3_SI_CONTROL ... 674
5-99 Register Call Summary for Register L3_SI_CONTROL ... 675
5-100 L3_SI_FLAG_STATUS_0 ... 675
5-101 Register Call Summary for Register L3_SI_FLAG_STATUS_0 ... 675
5-102 L3_SI_FLAG_STATUS_1 ... 675
5-103 Register Call Summary for Register L3_SI_FLAG_STATUS_1 ... 676
5-104 L4-Core Target Agents ... 679
5-105 L4-Per Target Agents ... 680
5-106 L4-Emu Target Agents ... 680
5-107 L4-Emu Initiator Agents ... 681
5-108 L4-Wakeup Target Agents ... 681
5-109 L4-Wakeup Initiator Agents ... 681
5-110 L4 Interconnect Clocks ... 682
5-111 L4 Interconnect Hardware Reset ... 682
5-112 L4 Interconnect Power Domains ... 682
5-113 Region Allocation for L4-Core Interconnect ... 686
5-114 Region Allocation for L4-Per Interconnect ... 689
5-115 Region Allocation for L4-Emu Interconnect ... 690
5-116 L4 Firewall Register Description Overview ... 691
5-117 L4 Time-Out Link and TA Programming ... 693
5-118 L4 Time-Out TA Programming ... 693
5-119 L4- Core Instance Summary ... 695
5-120 L4-Per Instance Summary ... 695
5-121 L4-Emu Instance Summary ... 696
5-122 L4-WKUP Instance Summary ... 696
5-123 L4 IA Register Summary (1) ... 697
5-124 L4 IA Register Summary (2) ... 697

5-125 L4_IA_COMPONENT_L 697
5-126 Register Call Summary for Register L4_IA_COMPONENT_L 698
5-127 L4_IA_COMPONENT_H 698
5-128 Register Call Summary for Register L4_IA_COMPONENT_H 698
5-129 L4_IA_CORE_L 698
5-130 Register Call Summary for Register L4_IA_CORE_L 698
5-131 L4_IA_CORE_H 699
5-132 Register Call Summary for Register L4_IA_CORE_H 699
5-133 L4_IA_AGENT_CONTROL_L 699
5-134 Register Call Summary for Register L4_IA_AGENT_CONTROL_L 699
5-135 L4_IA_AGENT_CONTROL_H 700
5-136 Register Call Summary for Register L4_IA_AGENT_CONTROL_H 700
5-137 L4_IA_AGENT_STATUS_L 700
5-138 Register Call Summary for Register L4_IA_AGENT_STATUS_L 700
5-139 L4_IA_AGENT_STATUS_H 701
5-140 Register Call Summary for Register L4_IA_AGENT_STATUS_H 701
5-141 L4_IA_ERROR_LOG_L 701
5-142 Register Call Summary for Register L4_IA_ERROR_LOG_L 701
5-143 L4_IA_ERROR_LOG_H 702
5-144 Register Call Summary for Register L4_IA_ERROR_LOG_H 702
5-145 CORE_TA Common Register Summary 702
5-146 CORE_TA Common Register Summary 702
5-147 CORE_TA Common Register Summary 703
5-148 CORE_TA Common Register Summary 703
5-149 CORE_TA Common Register Summary 703
5-150 CORE_TA Common Register Summary 703
5-151 CORE_TA Common Register Summary 704
5-152 CORE_TA Common Register Summary 704
5-153 CORE_TA Common Register Summary 704
5-154 CORE_TA Common Register Summary 704
5-155 CORE_TA Common Register Summary 705
5-156 CORE_TA Common Register Summary 705
5-157 CORE_TA Common Register Summary 705
5-158 CORE_TA Common Register Summary 705
5-159 PER_TA Common Register Summary 706
5-160 PER_TA Common Register Summary 706
5-161 PER_TA Common Register Summary 706
5-162 PER_TA Common Register Summary 707
5-163 PER_TA Common Register Summary 707
5-164 PER_TA Common Register Summary 707
5-165 PER_TA Common Register Summary 707
5-166 EMU_TA Common Register Summary 708
5-167 EMU_TA Common Register Summary 708
5-168 WKUP_TA Common Register Summary 708
5-169 WKUP_TA Common Register Summary 708
5-170 L4_TA_COMPONENT_L 709
5-171 Register Call Summary for Register L4_TA_COMPONENT_L 709
5-172 L4_TA_COMPONENT_H 709
5-173 Register Call Summary for Register L4_TA_COMPONENT_H 709
5-174 L4_TA_CORE_L 710
5-175 Register Call Summary for Register L4_TA_CORE_L 710

5-176 L4_TA_CORE_H 710
5-177 Register Call Summary for Register L4_TA_CORE_H 710
5-178 L4_TA_AGENT_CONTROL_L 710
5-179 Register Call Summary for Register L4_TA_AGENT_CONTROL_L 711
5-180 L4_TA_AGENT_CONTROL_H 711
5-181 Register Call Summary for Register L4_TA_AGENT_CONTROL_H 711
5-182 L4_TA_AGENT_STATUS_L 712
5-183 Register Call Summary for Register L4_TA_AGENT_STATUS_L 712
5-184 L4_TA_AGENT_STATUS_H 712
5-185 Register Call Summary for Register L4_TA_AGENT_STATUS_H 712
5-186 L4 LA Register Summary 713
5-187 L4_LA_COMPONENT_L 713
5-188 Register Call Summary for Register L4_LA_COMPONENT_L 713
5-189 L4_LA_COMPONENT_H 713
5-190 Register Call Summary for Register L4_LA_COMPONENT_H 713
5-191 L4_LA_NETWORK_L 714
5-192 Register Call Summary for Register L4_LA_NETWORK_L 714
5-193 L4_LA_NETWORK_H 714
5-194 Register Call Summary for Register L4_LA_NETWORK_H 714
5-195 L4_LA_INITIATOR_INFO_L 714
5-196 Register Call Summary for Register L4_LA_INITIATOR_INFO_L 715
5-197 Reset value for L4_LA_INITIATOR_INFO_L 715
5-198 L4_LA_INITIATOR_INFO_H 715
5-199 Register Call Summary for Register L4_LA_INITIATOR_INFO_H 716
5-200 Reset value for L4_LA_INITIATOR_INFO_H 716
5-201 L4_LA_NETWORK_CONTROL_L 716
5-202 Register Call Summary for Register L4_LA_NETWORK_CONTROL_L 716
5-203 L4_LA_NETWORK_CONTROL_H 717
5-204 Register Call Summary for Register L4_LA_NETWORK_CONTROL_H 717
5-205 L4 AP Register Summary 717
5-206 L4 AP Register Summary 718
5-207 L4_AP_COMPONENT_L 719
5-208 Register Call Summary for Register L4_AP_COMPONENT_L 719
5-209 L4_AP_COMPONENT_H 719
5-210 Register Call Summary for Register L4_AP_COMPONENT_H 719
5-211 L4_AP_SEGMENT_i_L 719
5-212 Register Call Summary for Register L4_AP_SEGMENT_i_L 720
5-213 L4_AP_SEGMENT_i_L Reset Values 720
5-214 L4_AP_SEGMENT_i_H 720
5-215 Register Call Summary for Register L4_AP_SEGMENT_i_H 720
5-216 L4_AP_SEGMENT_i_H reset values 720
5-217 L4_AP_PROT_GROUP_MEMBERS_k_L 720
5-218 Register Call Summary for Register L4_AP_PROT_GROUP_MEMBERS_k_L 721
5-219 L4_AP_PROT_GROUP_MEMBERS_k_H 721
5-220 Register Call Summary for Register L4_AP_PROT_GROUP_MEMBERS_k_H 721
5-221 L4_AP_PROT_GROUP_ROLES_k_L 721
5-222 Register Call Summary for Register L4_AP_PROT_GROUP_ROLES_k_L 721
5-223 L4_AP_PROT_GROUP_ROLES_k_H 722
5-224 Register Call Summary for Register L4_AP_PROT_GROUP_ROLES_k_H 722
5-225 L4_AP_REGION_l_L 722
5-226 Register Call Summary for Register L4_AP_REGION_l_L 722

5-227 L4_AP_REGION_I_H 722
5-228 Register Call Summary for Register L4_AP_REGION_I_H 723
5-229 Reset Values for CORE_AP L4_AP_REGION_I_L and L4_AP_REGION_I_H 723
5-230 Reset Values for PER_AP L4_AP_REGION_I_L and L4_AP_REGION_I_H 726
5-231 Reset Values for EMU_AP L4_AP_REGION_I_L and L4_AP_REGION_I_H 727
5-232 Reset Values for WKPUP_AP L4_AP_REGION_I_L and L4_AP_REGION_I_H 728
6-1 Mailbox Power Management Modes 732
6-2 Register Print After the Mailbox Module Initialization 743
6-3 Register Print After the Interrupt Enabling 743
6-4 Register Print After the MPU Subsystem Message Sending 744
6-5 Register Print After the IVA2.2 Subsystem Message Receiving 745
6-6 Register Print After the IVA2.2 Subsystem Message Sending 746
6-7 Register Print After the MPU Subsystetm Message Receiving 746
6-8 Mailbox Instance Summary 748
6-9 MLB Register Summary 748
6-10 MAILBOX_REVISION 748
6-11 Register Call Summary for Register MAILBOX_REVISION 749
6-12 MAILBOX_SYSCONFIG 749
6-13 Register Call Summary for Register MAILBOX_SYSCONFIG 750
6-14 MAILBOX_SYSSTATUS 750
6-15 Register Call Summary for Register MAILBOX_SYSSTATUS 750
6-16 MAILBOX_MESSAGE_m 751
6-17 Register Call Summary for Register MAILBOX_MESSAGE_m 751
6-18 MAILBOX_FIFOSTATUS_m 751
6-19 Register Call Summary for Register MAILBOX_FIFOSTATUS_m 751
6-20 MAILBOX_MSGSTATUS_m 752
6-21 Register Call Summary for Register MAILBOX_MSGSTATUS_m 752
6-22 MAILBOX_IRQSTATUS_u 752
6-23 Register Call Summary for Register MAILBOX_IRQSTATUS_u 753
6-24 MAILBOX_IRQENABLE_u 753
6-25 Register Call Summary for Register MAILBOX_IRQENABLE_u 753
7-1 SCM I/O Description 758
7-2 Mode Selection 766
7-3 Pull Selection 767
7-4 Core Control Module Pad Configuration Register Fields 768
7-5 Core Control Module D2D Pad Configuration Register Fields 775
7-6 Wake-Up Control Module Pad Configuration Register Fields 777
7-7 Bit Directions for CONTROL_PADCONF_x Registers 781
7-8 PBIAS Cell and Extended-Drain I/O Pin CONTROL_PBIAS_LITE Bit Controls 782
7-9 Power Supplies 783
7-10 Band Gap Voltage and Temperature Sensor Signals Description 786
7-11 ADC Codes Versus Temperature 788
7-12 Static Device Configuration Registers 789
7-13 Control CSIRXFE Register 789
7-14 MSuspendMux Control Registers 789
7-15 IVA2.2 Boot Registers 790
7-16 IVA2.2 Boot Modes 790
7-17 PBIAS Control Register 791
7-18 Temperature Sensor Register 791
7-19 CSI Receiver Control Register 791
7-20 Protection Status Registers 791

7-21 SDRC Registers 792
7-22 Observability Registers 793
7-23 Internal Signals Multiplexed on OBSMUX0 795
7-24 Internal Signals Multiplexed on OBSMUX1 795
7-25 Internal Signals Multiplexed on OBSMUX2 796
7-26 Internal Signals Multiplexed on OBSMUX3 797
7-27 Internal Signals Multiplexed on OBSMUX4 798
7-28 Internal Signals Multiplexed on OBSMUX5 798
7-29 Internal Signals Multiplexed on OBSMUX6 799
7-30 Internal Signals Multiplexed on OBSMUX7 801
7-31 Internal Signals Multiplexed on OBSMUX8 802
7-32 Internal Signals Multiplexed on OBSMUX9 803
7-33 Internal Signals Multiplexed on OBSMUX10 804
7-34 Internal Signals Multiplexed on OBSMUX11 805
7-35 Internal Signals Multiplexed on OBSMUX12 806
7-36 Internal Signals Multiplexed on OBSMUX13 806
7-37 Internal Signals Multiplexed on OBSMUX14 807
7-38 Internal Signals Multiplexed on OBSMUX15 808
7-39 Internal Signals Multiplexed on OBSMUX16 809
7-40 Internal Signals Multiplexed on OBSMUX17 810
7-41 Internal Signals Multiplexed on WKUPOBSMUX0 811
7-42 Internal Signals Multiplexed on WKUPOBSMUX1 812
7-43 Internal Signals Multiplexed on WKUPOBSMUX2 813
7-44 Internal Signals Multiplexed on WKUPOBSMUX3 814
7-45 Internal Signals Multiplexed on WKUPOBSMUX4 815
7-46 Internal Signals Multiplexed on WKUPOBSMUX5 816
7-47 Internal Signals Multiplexed on WKUPOBSMUX6 817
7-48 Internal Signals Multiplexed on WKUPOBSMUX7 818
7-49 Internal Signals Multiplexed on WKUPOBSMUX8 819
7-50 Internal Signals Multiplexed on WKUPOBSMUX9 820
7-51 Internal Signals Multiplexed on WKUPOBSMUX10 821
7-52 Internal Signals Multiplexed on WKUPOBSMUX11 822
7-53 Internal Signals Multiplexed on WKUPOBSMUX12 823
7-54 Internal Signals Multiplexed on WKUPOBSMUX13 824
7-55 Internal Signals Multiplexed on WKUPOBSMUX14 825
7-56 Internal Signals Multiplexed on WKUPOBSMUX15 826
7-57 Internal Signals Multiplexed on WKUPOBSMUX16 827
7-58 Internal Signals Multiplexed on WKUPOBSMUX17 828
7-59 Control Signals 844
7-60 Voltage Configuration 844
7-61 PBIAS Error Signal Truth Table 846
7-29 Pin Types 849
7-62 SCM Instance Summary 851
7-63 INTERFACE Register Summary 851
7-64 PADCONFS Register Summary 851
7-65 General Register Summary 855
7-66 MEM_WKUP Register Summary 856
7-67 PADCONFS_WKUP Register Summary 856
7-68 GENERAL_WKUP Register Summary 857
7-69 CONTROL_REVISION 857
7-70 Register Call Summary for Register CONTROL_REVISION 858

7-71 CONTROL_SYSCONFIG ... 858
7-72 Register Call Summary for Register CONTROL_SYSCONFIG ... 858
7-73 CONTROL_PADCONF_X ... 859
7-74 Register Call Summary for Register CONTROL_PADCONF_X ... 860
7-75 CONTROL_PADCONF_CAPABILITIES ... 862
7-76 CONTROL_PADCONF_OFF ... 872
7-77 Register Call Summary for Register CONTROL_PADCONF_OFF ... 872
7-78 CONTROL_DEVCONF0 ... 873
7-79 Register Call Summary for Register CONTROL_DEVCONF0 ... 873
7-80 CONTROL_MSUSPENDMUX_0 ... 874
7-81 Register Call Summary for Register CONTROL_MSUSPENDMUX_0 ... 876
7-82 CONTROL_MSUSPENDMUX_1 ... 876
7-83 Register Call Summary for Register CONTROL_MSUSPENDMUX_1 ... 879
7-84 CONTROL_MSUSPENDMUX_2 ... 879
7-85 Register Call Summary for Register CONTROL_MSUSPENDMUX_2 ... 881
7-86 CONTROL_MSUSPENDMUX_3 ... 881
7-87 Register Call Summary for Register CONTROL_MSUSPENDMUX_3 ... 881
7-88 CONTROL_MSUSPENDMUX_4 ... 881
7-89 Register Call Summary for Register CONTROL_MSUSPENDMUX_4 ... 882
7-90 CONTROL_MSUSPENDMUX_5 ... 882
7-91 Register Call Summary for Register CONTROL_MSUSPENDMUX_5 ... 883
7-92 CONTROL_DEVCONF1 ... 883
7-93 Register Call Summary for Register CONTROL_DEVCONF1 ... 885
7-94 CONTROL_CSIRXFE ... 885
7-95 Register Call Summary for Register CONTROL_CSIRXFE ... 886
7-96 CONTROL_PROT_ERR_STATUS ... 886
7-97 Register Call Summary for Register CONTROL_PROT_ERR_STATUS ... 888
7-98 CONTROL_PROT_ERR_STATUS_DEBUG ... 888
7-99 Register Call Summary for Register CONTROL_PROT_ERR_STATUS_DEBUG ... 889
7-100 CONTROL_STATUS ... 889
7-101 Register Call Summary for Register CONTROL_STATUS ... 889
7-102 CONTROL_GENERAL_PURPOSE_STATUS ... 890
7-103 Register Call Summary for Register CONTROL_GENERAL_PURPOSE_STATUS ... 890
7-104 CONTROL_RPUB_KEY_H_0 ... 890
7-105 Register Call Summary for Register CONTROL_RPUB_KEY_H_0 ... 890
7-106 CONTROL_RPUB_KEY_H_1 ... 891
7-107 Register Call Summary for Register CONTROL_RPUB_KEY_H_1 ... 891
7-108 CONTROL_RPUB_KEY_H_2 ... 891
7-109 Register Call Summary for Register CONTROL_RPUB_KEY_H_2 ... 891
7-110 CONTROL_RPUB_KEY_H_3 ... 891
7-111 Register Call Summary for Register CONTROL_RPUB_KEY_H_3 ... 891
7-112 CONTROL_RPUB_KEY_H_4 ... 892
7-113 Register Call Summary for Register CONTROL_RPUB_KEY_H_4 ... 892
7-114 CONTROL_USB_CONF_0 ... 892
7-115 Register Call Summary for Register CONTROL_USB_CONF_0 ... 892
7-116 CONTROL_USB_CONF_1 ... 892
7-117 Register Call Summary for Register CONTROL_USB_CONF_1 ... 893
7-118 CONTROL_FUSE_OPP1_VDD1 ... 893
7-119 Register Call Summary for Register CONTROL_FUSE_OPP1_VDD1 ... 893
7-120 CONTROL_FUSE_OPP2_VDD1 ... 893
7-121 Register Call Summary for Register CONTROL_FUSE_OPP2_VDD1 ... 893

7-122 CONTROL_FUSE_OPP3_VDD1 893
7-123 Register Call Summary for Register CONTROL_FUSE_OPP3_VDD1 894
7-124 CONTROL_FUSE_OPP4_VDD1 894
7-125 Register Call Summary for Register CONTROL_FUSE_OPP4_VDD1 894
7-126 CONTROL_FUSE_OPP5_VDD1 894
7-127 Register Call Summary for Register CONTROL_FUSE_OPP5_VDD1 894
7-128 CONTROL_FUSE_OPP1_VDD2 894
7-129 Register Call Summary for Register CONTROL_FUSE_OPP1_VDD2 895
7-130 CONTROL_FUSE_OPP2_VDD2 895
7-131 Register Call Summary for Register CONTROL_FUSE_OPP2_VDD2 895
7-132 CONTROL_FUSE_OPP3_VDD2 895
7-133 Register Call Summary for Register CONTROL_FUSE_OPP3_VDD2 895
7-134 CONTROL_FUSE_SR 895
7-135 Register Call Summary for Register CONTROL_FUSE_SR 896
7-136 CONTROL_IVA2_BOOTADDR 896
7-137 Register Call Summary for Register CONTROL_IVA2_BOOTADDR 896
7-138 CONTROL_IVA2_BOOTMOD 896
7-139 Register Call Summary for Register CONTROL_IVA2_BOOTMOD 896
7-140 CONTROL_DEBOBS_0 897
7-141 Register Call Summary for Register CONTROL_DEBOBS_0 897
7-142 CONTROL_DEBOBS_1 897
7-143 Register Call Summary for Register CONTROL_DEBOBS_1 897
7-144 CONTROL_DEBOBS_2 898
7-145 Register Call Summary for Register CONTROL_DEBOBS_2 898
7-146 CONTROL_DEBOBS_3 898
7-147 Register Call Summary for Register CONTROL_DEBOBS_3 898
7-148 CONTROL_DEBOBS_4 899
7-149 Register Call Summary for Register CONTROL_DEBOBS_4 899
7-150 CONTROL_DEBOBS_5 899
7-151 Register Call Summary for Register CONTROL_DEBOBS_5 899
7-152 CONTROL_DEBOBS_6 900
7-153 Register Call Summary for Register CONTROL_DEBOBS_6 900
7-154 CONTROL_DEBOBS_7 900
7-155 Register Call Summary for Register CONTROL_DEBOBS_7 900
7-156 CONTROL_DEBOBS_8 901
7-157 Register Call Summary for Register CONTROL_DEBOBS_8 901
7-158 CONTROL_PROG_IO0 901
7-159 Register Call Summary for Register CONTROL_PROG_IO0 903
7-160 CONTROL_PROG_IO1 903
7-161 Register Call Summary for Register CONTROL_PROG_IO1 904
7-162 CONTROL_DSS_DPLL_SPREADING 904
7-163 Register Call Summary for Register CONTROL_DSS_DPLL_SPREADING 905
7-164 CONTROL_CORE_DPLL_SPREADING 905
7-165 Register Call Summary for Register CONTROL_CORE_DPLL_SPREADING 906
7-166 CONTROL_PER_DPLL_SPREADING 906
7-167 Register Call Summary for Register CONTROL_PER_DPLL_SPREADING 907
7-168 CONTROL_USBHOST_DPLL_SPREADING 907
7-169 Register Call Summary for Register CONTROL_USBHOST_DPLL_SPREADING 908
7-170 CONTROL_SDRC_SHARING 908
7-171 Register Call Summary for Register CONTROL_SDRC_SHARING 908
7-172 CONTROL_SDRC_MCFG0 909

7-173 Register Call Summary for Register CONTROL_SDRC_MCFG0 909
7-174 CONTROL_SDRC_MCFG1 909
7-175 Register Call Summary for Register CONTROL_SDRC_MCFG1 909
7-176 CONTROL_MODEM_FW_CONFIGURATION_LOCK 910
7-177 Register Call Summary for Register CONTROL_MODEM_FW_CONFIGURATION_LOCK 910
7-178 Type Value For CONTROL_MODEM_FW_CONFIGURATION_LOCK 910
7-179 CONTROL_MODEM_MEMORY_RESOURCES_CONF 910
7-180 Register Call Summary for Register CONTROL_MODEM_MEMORY_RESOURCES_CONF 911
7-181 Type Value For CONTROL_MODEM_MEMORY_RESOURCES_CONF 911
7-182 CONTROL_MODEM_GPMC_DT_FW_REQ_INFO 912
7-183 Register Call Summary for Register CONTROL_MODEM_GPMC_DT_FW_REQ_INFO 912
7-184 Type Value For CONTROL_MODEM_GPMC_DT_FW_REQ_INFO 912
7-185 CONTROL_MODEM_GPMC_DT_FW_RD 912
7-186 Register Call Summary for Register CONTROL_MODEM_GPMC_DT_FW_RD 912
7-187 Type Value For CONTROL_MODEM_GPMC_DT_FW_RD 912
7-188 CONTROL_MODEM_GPMC_DT_FW_WR 913
7-189 Register Call Summary for Register CONTROL_MODEM_GPMC_DT_FW_WR 913
7-190 Type Value For CONTROL_MODEM_GPMC_DT_FW_WR 913
7-191 CONTROL_MODEM_GPMC_BOOT_CODE 913
7-192 Register Call Summary for Register CONTROL_MODEM_GPMC_BOOT_CODE 914
7-193 Type Value For CONTROL_MODEM_GPMC_BOOT_CODE 914
7-194 CONTROL_MODEM_SMS_RG_ATT1 914
7-195 Register Call Summary for Register CONTROL_MODEM_SMS_RG_ATT1 914
7-196 Type Value For CONTROL_MODEM_SMS_RG_ATT1 914
7-197 CONTROL_MODEM_SMS_RG_RDPERM1 914
7-198 Register Call Summary for Register CONTROL_MODEM_SMS_RG_RDPERM1 914
7-199 Type Value For CONTROL_MODEM_SMS_RG_RDPERM1 915
7-200 CONTROL_MODEM_SMS_RG_WRPERM1 915
7-201 Register Call Summary for Register CONTROL_MODEM_SMS_RG_WRPERM1 915
7-202 Type Value For CONTROL_MODEM_SMS_RG_WRPERM1 915
7-203 CONTROL_MODEM_D2D_FW_DEBUG_MODE 915
7-204 Register Call Summary for Register CONTROL_MODEM_D2D_FW_DEBUG_MODE 916
7-205 Type Value For CONTROL_MODEM_D2D_FW_DEBUG_MODE 916
7-206 CONTROL_DPF_OCM_RAM_FW_ADDR_MATCH 916
7-207 Register Call Summary for Register CONTROL_DPF_OCM_RAM_FW_ADDR_MATCH 916
7-208 CONTROL_DPF_OCM_RAM_FW_REQINFO 916
7-209 Register Call Summary for Register CONTROL_DPF_OCM_RAM_FW_REQINFO 916
7-210 CONTROL_DPF_OCM_RAM_FW_WR 917
7-211 Register Call Summary for Register CONTROL_DPF_OCM_RAM_FW_WR 917
7-212 CONTROL_DPF_REGION4_GPMC_FW_ADDR_MATCH 917
7-213 Register Call Summary for Register CONTROL_DPF_REGION4_GPMC_FW_ADDR_MATCH 917
7-214 CONTROL_DPF_REGION4_GPMC_FW_REQINFO 918
7-215 Register Call Summary for Register CONTROL_DPF_REGION4_GPMC_FW_REQINFO 918
7-216 CONTROL_DPF_REGION4_GPMC_FW_WR 918
7-217 Register Call Summary for Register CONTROL_DPF_REGION4_GPMC_FW_WR 918
7-218 CONTROL_DPF_REGION1_IVA2_FW_ADDR_MATCH 918
7-219 Register Call Summary for Register CONTROL_DPF_REGION1_IVA2_FW_ADDR_MATCH 919
7-220 CONTROL_DPF_REGION1_IVA2_FW_REQINFO 919
7-221 Register Call Summary for Register CONTROL_DPF_REGION1_IVA2_FW_REQINFO 919
7-222 CONTROL_DPF_REGION1_IVA2_FW_WR 919
7-223 Register Call Summary for Register CONTROL_DPF_REGION1_IVA2_FW_WR 919

7-224 CONTROL_PBIAS_LITE 920
7-225 Register Call Summary for Register CONTROL_PBIAS_LITE 921
7-226 CONTROL_TEMP_SENSOR 921
7-227 Register Call Summary for Register CONTROL_TEMP_SENSOR 921
7-228 CONTROL_CSI 922
7-229 Register Call Summary for Register CONTROL_CSI 922
7-230 CONTROL_DPF_MAD2D_FW_ADDR_MATCH 922
7-231 Register Call Summary for Register CONTROL_DPF_MAD2D_FW_ADDR_MATCH 922
7-232 CONTROL_DPF_MAD2D_FW_REQINFO 923
7-233 Register Call Summary for Register CONTROL_DPF_MAD2D_FW_REQINFO 923
7-234 CONTROL_DPF_MAD2D_FW_WR 923
7-235 Register Call Summary for Register CONTROL_DPF_MAD2D_FW_WR 923
7-236 CONTROL_IDCODE 923
7-237 Register Call Summary for Register CONTROL_IDCODE 924
7-238 CONTROL_PADCONF_WKUP_CAPABILITIES 925
7-239 CONTROL_WKUP_CTRL 926
7-240 Register Call Summary for Register CONTROL_WKUP_CTRL 926
7-241 CONTROL_WKUP_DEBOBS_0 926
7-242 Register Call Summary for Register CONTROL_WKUP_DEBOBS_0 927
7-243 Type Value For CONTROL_WKUP_DEBOBS_0 Register 927
7-244 CONTROL_WKUP_DEBOBS_1 927
7-245 Register Call Summary for Register CONTROL_WKUP_DEBOBS_1 928
7-246 Type Value For CONTROL_WKUP_DEBOBS_1 Register 928
7-247 CONTROL_WKUP_DEBOBS_2 928
7-248 Register Call Summary for Register CONTROL_WKUP_DEBOBS_2 929
7-249 Type Value For CONTROL_WKUP_DEBOBS_2 Register 929
7-250 CONTROL_WKUP_DEBOBS_3 929
7-251 Register Call Summary for Register CONTROL_WKUP_DEBOBS_3 929
7-252 Type Value For CONTROL_WKUP_DEBOBS_3 Register 930
7-253 CONTROL_WKUP_DEBOBS_4 930
7-254 Register Call Summary for Register CONTROL_WKUP_DEBOBS_4 930
7-255 Type Value For CONTROL_WKUP_DEBOBS_4 Register 930
8-1 Power Domains of the MMU Instances 934
8-2 Power Domains of the MMU Instances 934
8-3 Reset Domains of the MMU Instances 935
8-4 Interrupts of the MMU Instances 935
8-5 First-Level Descriptor Format 942
8-6 Second-Level Descriptor Format 945
8-7 Design Parameters of the MMU Instances 949
8-8 MMU Instance Summary 957
8-9 MMU Register Summary 957
8-10 MMU_REVISION 958
8-11 Register Call Summary for Register MMU_REVISION 958
8-12 MMU_SYSCONFIG 958
8-13 Register Call Summary for Register MMU_SYSCONFIG 959
8-14 MMU_SYSSTATUS 959
8-15 Register Call Summary for Register MMU_SYSSTATUS 960
8-16 MMU_IRQSTATUS 960
8-17 Register Call Summary for Register MMU_IRQSTATUS 961
8-18 MMU_IRQENABLE 961
8-19 Register Call Summary for Register MMU_IRQENABLE 962

8-20 MMU_WALKING_ST ... 962
8-21 Register Call Summary for Register MMU_WALKING_ST ... 962
8-22 MMU_CNTL ... 962
8-23 Register Call Summary for Register MMU_CNTL ... 963
8-24 MMU_FAULT_AD ... 963
8-25 Register Call Summary for Register MMU_FAULT_AD ... 963
8-26 MMU_TTB ... 964
8-27 Register Call Summary for Register MMU_TTB ... 964
8-28 MMU_LOCK ... 964
8-29 Register Call Summary for Register MMU_LOCK ... 964
8-30 MMU_LD_TLB ... 965
8-31 Register Call Summary for Register MMU_LD_TLB ... 965
8-32 MMU_CAM ... 965
8-33 Register Call Summary for Register MMU_CAM ... 966
8-34 MMU_RAM ... 966
8-35 Register Call Summary for Register MMU_RAM ... 967
8-36 MMU_GFLUSH ... 967
8-37 Register Call Summary for Register MMU_GFLUSH ... 967
8-38 MMU_FLUSH_ENTRY ... 967
8-39 Register Call Summary for Register MMU_FLUSH_ENTRY ... 968
8-40 MMU_READ_CAM ... 968
8-41 Register Call Summary for Register MMU_READ_CAM ... 968
8-42 MMU_READ_RAM ... 969
8-43 Register Call Summary for Register MMU_READ_RAM ... 969
8-44 MMU_EMU_FAULT_AD ... 969
8-45 Register Call Summary for Register MMU_EMU_FAULT_AD ... 970
9-1 Description External DMA Request Pin ... 975
9-2 SDMA Interrupt Mapping to the MPU Subsystem ... 978
9-3 SDMA Request Mapping ... 978
9-4 Parameter Values for Addressing Mode Examples (a), (b), and (c) ... 986
9-5 Equations for Rotation ... 986
9-6 Example Parameter Values for a 90° Clockwise Image Rotation ... 987
9-7 Buffering Disable ... 990
9-8 Logical DMA Channel Events ... 993
9-9 Registers Print ... 1007
9-10 Registers Print ... 1009
9-11 SDMA Instances Summary ... 1011
9-12 SDMA Register Summary ... 1011
9-13 DMA4_REVISION ... 1012
9-14 Register Call Summary for Register DMA4_REVISION ... 1012
9-15 DMA4_IRQSTATUS_Lj ... 1012
9-16 Register Call Summary for Register DMA4_IRQSTATUS_Lj ... 1013
9-17 DMA4_IRQENABLE_Lj ... 1013
9-18 Register Call Summary for Register DMA4_IRQENABLE_Lj ... 1013
9-19 DMA4_SYSSTATUS ... 1014
9-20 Register Call Summary for Register DMA4_SYSSTATUS ... 1014
9-21 DMA4_OCP_SYSCONFIG ... 1014
9-22 Register Call Summary for Register DMA4_OCP_SYSCONFIG ... 1015
9-23 DMA4_CAPS_0 ... 1015
9-24 Register Call Summary for Register DMA4_CAPS_0 ... 1016
9-25 DMA4_CAPS_2 ... 1016

9-26 Register Call Summary for Register DMA4_CAPS_2 1017
9-27 DMA4_CAPS_3 1018
9-28 Register Call Summary for Register DMA4_CAPS_3 1018
9-29 DMA4_CAPS_4 1019
9-30 Register Call Summary for Register DMA4_CAPS_4 1020
9-31 DMA4_GCR 1020
9-32 Register Call Summary for Register DMA4_GCR 1021
9-33 DMA4_CCRi 1022
9-34 Register Call Summary for Register DMA4_CCRi 1024
9-35 DMA4_CLNK_CTRLi 1024
9-36 Register Call Summary for Register DMA4_CLNK_CTRLi 1025
9-37 DMA4_CICRi 1025
9-38 Register Call Summary for Register DMA4_CICRi 1026
9-39 DMA4_CSRi 1026
9-40 Register Call Summary for Register DMA4_CSRi 1028
9-41 DMA4_CSDPi 1028
9-42 Register Call Summary for Register DMA4_CSDPi 1030
9-43 DMA4_CENi 1030
9-44 Register Call Summary for Register DMA4_CENi 1030
9-45 DMA4_CFNi 1031
9-46 Register Call Summary for Register DMA4_CFNi 1031
9-47 DMA4_CSSAi 1031
9-48 Register Call Summary for Register DMA4_CSSAi 1031
9-49 DMA4_CDSAi 1032
9-50 Register Call Summary for Register DMA4_CDSAi 1032
9-51 DMA4_CSEIi 1032
9-52 Register Call Summary for Register DMA4_CSEIi 1032
9-53 DMA4_CSFIi 1033
9-54 Register Call Summary for Register DMA4_CSFIi 1033
9-55 DMA4_CDEIi 1033
9-56 Register Call Summary for Register DMA4_CDEIi 1033
9-57 DMA4_CDFIi 1033
9-58 Register Call Summary for Register DMA4_CDFIi 1034
9-59 DMA4_CSACi 1034
9-60 Register Call Summary for Register DMA4_CSACi 1034
9-61 DMA4_CDACi 1034
9-62 Register Call Summary for Register DMA4_CDACi 1034
9-63 DMA4_CCENi 1035
9-64 Register Call Summary for Register DMA4_CCENi 1035
9-65 DMA4_CCFNi 1035
9-66 Register Call Summary for Register DMA4_CCFNi 1035
9-67 DMA4_COLORi 1036
9-68 Register Call Summary for Register DMA4_COLORi 1036
10-1 MPU Subsystem INTC Clock Rates 1041
10-2 Hardware and Software Reset 1042
10-3 Interrupt Lines Incoming and Outgoing 1042
10-4 Interrupt Mapping to the MPU Subsystem 1042
10-5 INTC Instance Summary 1057
10-6 MPU INTC Register Summary 1057
10-7 Modem INTC Register Summary 1058
10-8 INTCPS_REVISION 1058

10-9 Register Call Summary for Register INTCPS_REVISION 1058
10-10 INTCPS_SYSCONFIG 1058
10-11 Register Call Summary for Register INTCPS_SYSCONFIG 1059
10-12 INTCPS_SYSSTATUS 1059
10-13 Register Call Summary for Register INTCPS_SYSSTATUS 1059
10-14 INTCPS_SIR_IRQ 1059
10-15 Register Call Summary for Register INTCPS_SIR_IRQ 1060
10-16 INTCPS_SIR_FIQ 1060
10-17 Register Call Summary for Register INTCPS_SIR_FIQ 1060
10-18 INTCPS_CONTROL 1060
10-19 Register Call Summary for Register INTCPS_CONTROL 1061
10-20 INTCPS_PROTECTION 1061
10-21 Register Call Summary for Register INTCPS_PROTECTION 1061
10-22 INTCPS_IDLE 1062
10-23 Register Call Summary for Register INTCPS_IDLE 1062
10-24 INTCPS_IRQ_PRIORITY 1062
10-25 Register Call Summary for Register INTCPS_IRQ_PRIORITY 1062
10-26 INTCPS_FIQ_PRIORITY 1063
10-27 Register Call Summary for Register INTCPS_FIQ_PRIORITY 1063
10-28 INTCPS_THRESHOLD 1063
10-29 Register Call Summary for Register INTCPS_THRESHOLD 1063
10-30 INTCPS_ITRn 1064
10-31 Register Call Summary for Register INTCPS_ITRn 1064
10-32 INTCPS_MIRn 1064
10-33 Register Call Summary for Register INTCPS_MIRn 1064
10-34 INTCPS_MIR_CLEARn 1065
10-35 Register Call Summary for Register INTCPS_MIR_CLEARn 1065
10-36 INTCPS_MIR_SETn 1065
10-37 Register Call Summary for Register INTCPS_MIR_SETn 1065
10-38 INTCPS_ISR_SETn 1066
10-39 Register Call Summary for Register INTCPS_ISR_SETn 1066
10-40 INTCPS_ISR_CLEARn 1066
10-41 Register Call Summary for Register INTCPS_ISR_CLEARn 1066
10-42 INTCPS_PENDING_IRQn 1067
10-43 Register Call Summary for Register INTCPS_PENDING_IRQn 1067
10-44 INTCPS_PENDING_FIQn 1067
10-45 Register Call Summary for Register INTCPS_PENDING_FIQn 1067
10-46 INTCPS_ILRm 1067
10-47 Register Call Summary for Register INTCPS_ILRm 1068
10-48 INTC_SYSCONFIG 1068
10-49 Register Call Summary for Register INTC_SYSCONFIG 1068
10-50 INTC_IDLE 1069
10-51 Register Call Summary for Register INTC_IDLE 1069
11-1 GPMC I/O Description 1075
11-2 GPMC Pin Multiplexing Options 1076
11-3 Idle Cycle Insertion Configuration 1098
11-4 Chip-Select Configuration for NAND Interfacing 1116
11-5 ECC Enable Settings 1123
11-6 Flattened BCH Codeword Mapping (512 Bytes + 104 Bits) 1128
11-7 Aligned Message Byte Mapping in 8-bit NAND 1129
11-8 Aligned Message Byte Mapping in 16-bit NAND 1129

11-9 Aligned Nibble Mapping of Message in 8-bit NAND 1129
11-10 Misaligned Nibble Mapping of Message in 8-bit NAND 1130
11-11 Aligned Nibble Mapping of Message in 16-bit NAND 1130
11-12 Misaligned Nibble Mapping of Message in 16-bit NAND (1 Unused Nibble) 1130
11-13 Misaligned Nibble Mapping of Message in 16-bit NAND (2 Unused Nibbles) 1130
11-14 Misaligned Nibble Mapping of Message in 16-bit NAND (3 Unused Nibbles) 1130
11-15 Prefetch Mode Configuration 1140
11-16 Write-Posting Mode Configuration 1142
11-17 GPMC Signals 1146
11-18 Useful Timing Parameters on the Memory Side 1147
11-19 Calculating GPMC Timing Parameters 1148
11-20 AC Characteristics for Asynchronous Read Access 1149
11-21 GPMC Timing Parameters for Asynchronous Read Access 1150
11-22 AC Characteristics for Asynchronous Single Write (Memory Side) 1150
11-23 GPMC Timing parameters for Asynchronous Single Write 1151
11-24 Supported Memory Interfaces 1152
11-25 NAND Interface Bus Operations Summary 1153
11-26 NOR Interface Bus Operations Summary 1154
11-27 GPMC Instance Summary 1155
11-28 GPMC Registers Mapping Summary 1155
11-29 GPMC_REVISION 1156
11-30 Register Call Summary for Register GPMC_REVISION 1156
11-31 GPMC_SYSCONFIG 1156
11-32 Register Call Summary for Register GPMC_SYSCONFIG 1157
11-33 GPMC_SYSSTATUS 1157
11-34 Register Call Summary for Register GPMC_SYSSTATUS 1157
11-35 GPMC_IRQSTATUS 1158
11-36 Register Call Summary for Register GPMC_IRQSTATUS 1159
11-37 GPMC_IRQENABLE 1159
11-38 Register Call Summary for Register GPMC_IRQENABLE 1160
11-39 GPMC_TIMEOUT_CONTROL 1160
11-40 Register Call Summary for Register GPMC_TIMEOUT_CONTROL 1160
11-41 GPMC_ERR_ADDRESS 1161
11-42 Register Call Summary for Register GPMC_ERR_ADDRESS 1161
11-43 GPMC_ERR_TYPE 1161
11-44 Register Call Summary for Register GPMC_ERR_TYPE 1162
11-45 GPMC_CONFIG 1162
11-46 Register Call Summary for Register GPMC_CONFIG 1163
11-47 GPMC_STATUS 1163
11-48 Register Call Summary for Register GPMC_STATUS 1164
11-49 GPMC_CONFIG1_i 1164
11-50 Register Call Summary for Register GPMC_CONFIG1_i 1166
11-51 GPMC_CONFIG2_i 1166
11-52 Register Call Summary for Register GPMC_CONFIG2_i 1167
11-53 GPMC_CONFIG3_i 1167
11-54 Register Call Summary for Register GPMC_CONFIG3_i 1168
11-55 GPMC_CONFIG4_i 1168
11-56 Register Call Summary for Register GPMC_CONFIG4_i 1169
11-57 GPMC_CONFIG5_i 1169
11-58 Register Call Summary for Register GPMC_CONFIG5_i 1170
11-59 GPMC_CONFIG6_i 1170

11-60 Register Call Summary for Register GPMC_CONFIG6_i 1171
11-61 GPMC_CONFIG7_i 1171
11-62 Register Call Summary for Register GPMC_CONFIG7_i 1172
11-63 GPMC_NAND_COMMAND_i 1172
11-64 Register Call Summary for Register GPMC_NAND_COMMAND_i 1172
11-65 GPMC_NAND_ADDRESS_i 1172
11-66 Register Call Summary for Register GPMC_NAND_ADDRESS_i 1172
11-67 GPMC_NAND_DATA_i 1172
11-68 Register Call Summary for Register GPMC_NAND_DATA_i 1173
11-69 GPMC_PREFETCH_CONFIG1 1173
11-70 Register Call Summary for Register GPMC_PREFETCH_CONFIG1 1174
11-71 GPMC_PREFETCH_CONFIG2 1175
11-72 Register Call Summary for Register GPMC_PREFETCH_CONFIG2 1175
11-73 GPMC_PREFETCH_CONTROL 1175
11-74 Register Call Summary for Register GPMC_PREFETCH_CONTROL 1175
11-75 GPMC_PREFETCH_STATUS 1176
11-76 Register Call Summary for Register GPMC_PREFETCH_STATUS 1176
11-77 GPMC_ECC_CONFIG 1177
11-78 Register Call Summary for Register GPMC_ECC_CONFIG 1178
11-79 GPMC_ECC_CONTROL 1178
11-80 Register Call Summary for Register GPMC_ECC_CONTROL 1178
11-81 GPMC_ECC_SIZE_CONFIG 1179
11-82 Register Call Summary for Register GPMC_ECC_SIZE_CONFIG 1180
11-83 GPMC_ECCj_RESULT 1180
11-84 Register Call Summary for Register GPMC_ECCj_RESULT 1181
11-85 GPMC_BCH_RESULT0_i 1181
11-86 Register Call Summary for Register GPMC_BCH_RESULT0_i 1181
11-87 GPMC_BCH_RESULT1_i 1181
11-88 Register Call Summary for Register GPMC_BCH_RESULT1_i 1181
11-89 GPMC_BCH_RESULT2_i 1182
11-90 Register Call Summary for Register GPMC_BCH_RESULT2_i 1182
11-91 GPMC_BCH_RESULT3_i 1182
11-92 Register Call Summary for Register GPMC_BCH_RESULT3_i 1182
11-93 GPMC_BCH_SWDATA 1182
11-94 Register Call Summary for Register GPMC_BCH_SWDATA 1183
11-95 SDRC Subsystem I/O Description 1187
11-96 SDRC Address Multiplexing Scheme Selection vs SDRAM Configurations (x16 Memory Interface) 1189
11-97 SDRC Address Multiplexing Scheme Selection vs SDRAM Configurations (x32 Memory Interface) 1189
11-98 Arbitration Class Allocation 1198
11-99 ReqInfo Parameters Ordering 1201
11-100 VRFB Contexts Virtual Address Spaces vs Rotation Angle 1204
11-101 Mobile DDR SDRAM AC Timing Parameters 1212
11-102 SDRC Data Lane Configurations 1213
11-103 Dynamic Power Saving Configurations 1218
11-104 Memory Configuration 1228
11-105 Programmable AC Parameters 1229
11-106 Nonprogrammable AC Parameters 1229
11-107 Calculating Image Size 1242
11-108 SDRC Signals Description 1256
11-109 Mobile DDR SDRAM Address Configuration 1257
11-110 Mobile DDR SDRAM AC Timings Parameters 1258

11-111 Calculation of the SDRC.SDRC_ACTIM_CTRLA_p Timing Parameters 1260
11-112 Calculation of the SDRC.SDRC_ACTIM_CTRLB_p (p = 0) Timing Parameters 1260
11-113 Calculation of the SDRC.SDRC_RFR_CTRL_p (p = 0) Timing Parameter 1260
11-114 VRFB Use Case Summarizing Register Print 1265
11-115 SDRC and SMS Configuration Register Space 1266
11-116 VRFB Contexts vs Rotation Angle 1267
11-117 SDRAM vs SDRC Controller Characteristics 1271
11-118 SMS Instance Summary 1272
11-119 SMS Register Summary 1272
11-120 SMS_REVISION 1273
11-121 Register Call Summary for Register SMS_REVISION 1273
11-122 SMS_SYSCONFIG 1273
11-123 Register Call Summary for Register SMS_SYSCONFIG 1274
11-124 SMS_SYSSTATUS 1274
11-125 Register Call Summary for Register SMS_SYSSTATUS 1274
11-126 SMS_RG_ATTi 1274
11-127 Register Call Summary for Register SMS_RG_ATTi 1275
11-128 SMS_RG_RDPERMi 1275
11-129 Register Call Summary for Register SMS_RG_RDPERMi 1275
11-130 SMS_RG_WRPERMi 1275
11-131 Register Call Summary for Register SMS_RG_WRPERMi 1275
11-132 SMS_RG_STARTj 1276
11-133 Register Call Summary for Register SMS_RG_STARTj 1276
11-134 SMS_RG_ENDj 1276
11-135 Register Call Summary for Register SMS_RG_ENDj 1276
11-136 SMS_CLASS_ARBITER0 1277
11-137 Register Call Summary for Register SMS_CLASS_ARBITER0 1277
11-138 SMS_CLASS_ARBITER1 1278
11-139 Register Call Summary for Register SMS_CLASS_ARBITER1 1278
11-140 SMS_CLASS_ARBITER2 1279
11-141 Register Call Summary for Register SMS_CLASS_ARBITER2 1279
11-142 SMS_INTERCLASS_ARBITER 1280
11-143 Register Call Summary for Register SMS_INTERCLASS_ARBITER 1280
11-144 SMS_CLASS_ROTATIONm 1280
11-145 Register Call Summary for Register SMS_CLASS_ROTATIONm 1280
11-146 SMS_ERR_ADDR 1281
11-147 Register Call Summary for Register SMS_ERR_ADDR 1281
11-148 SMS_ERR_TYPE 1281
11-149 Register Call Summary for Register SMS_ERR_TYPE 1282
11-150 SMS_POW_CTRL 1283
11-151 Register Call Summary for Register SMS_POW_CTRL 1283
11-152 SMS_ROT_CONTROLn 1283
11-153 Register Call Summary for Register SMS_ROT_CONTROLn 1283
11-154 SMS_ROT_SIZEn 1284
11-155 Register Call Summary for Register SMS_ROT_SIZEn 1284
11-156 SMS_ROT_PHYSICAL_BAn 1284
11-157 Register Call Summary for Register SMS_ROT_PHYSICAL_BAn 1284
11-158 SDRC Instance Summary 1285
11-159 SDRC Register Summary 1285
11-160 SDRC_REVISION 1286
11-161 Register Call Summary for Register SDRC_REVISION 1286

11-162 SDRC_SYSCONFIG 1286
11-163 Register Call Summary for Register SDRC_SYSCONFIG 1287
11-164 SDRC_SYSSTATUS 1287
11-165 Register Call Summary for Register SDRC_SYSSTATUS 1287
11-166 SDRC_CS_CFG 1288
11-167 Register Call Summary for Register SDRC_CS_CFG 1288
11-168 SDRC_SHARING 1288
11-169 Register Call Summary for Register SDRC_SHARING 1289
11-170 SDRC_ERR_ADDR 1289
11-171 Register Call Summary for Register SDRC_ERR_ADDR 1289
11-172 SDRC_ERR_TYPE 1290
11-173 Register Call Summary for Register SDRC_ERR_TYPE 1290
11-174 SDRC_DLLA_CTRL 1291
11-175 Register Call Summary for Register SDRC_DLLA_CTRL 1292
11-176 SDRC_DLLA_STATUS 1292
11-177 Register Call Summary for Register SDRC_DLLA_STATUS 1293
11-178 SDRC_POWER_REG 1293
11-179 Register Call Summary for Register SDRC_POWER_REG 1294
11-180 SDRC_MCFG_p 1294
11-181 Register Call Summary for Register SDRC_MCFG_p 1295
11-182 SDRC_MR_p 1297
11-183 Register Call Summary for Register SDRC_MR_p 1298
11-184 SDRC_EMR2_p 1298
11-185 Register Call Summary for Register SDRC_EMR2_p 1299
11-186 SDRC_ACTIM_CTRLA_p 1299
11-187 Register Call Summary for Register SDRC_ACTIM_CTRLA_p 1300
11-188 SDRC_ACTIM_CTRLB_p 1300
11-189 Register Call Summary for Register SDRC_ACTIM_CTRLB_p 1300
11-190 SDRC_RFR_CTRL_p 1301
11-191 Register Call Summary for Register SDRC_RFR_CTRL_p 1301
11-192 SDRC_MANUAL_p 1301
11-193 Register Call Summary for Register SDRC_MANUAL_p 1302
12-1 Camera ISP Functions 1313
12-2 I/O Description 1313
12-3 Video Timing Reference Codes for SAV and EAV 1320
12-4 F, V, H Signal Descriptions 1320
12-5 F, V, H Protection (Error-Correction) Bits 1321
12-6 BT.656 Mode Data Format in SDRAM 1321
12-7 I/O Description for Serial Interface CSI1 1321
12-8 Synchronization Codes 1322
12-9 CSI1 Image Data Operating Modes and Alignment Constraints 1323
12-10 I/O Description for CSI2 Serial Interface 1333
12-11 Long Packet Structure Description 1337
12-12 CSI2 Pixel Format Modes 1338
12-13 Synchronization Codes 1339
12-14 Clock Descriptions 1360
12-15 cam_xclka Configuration 1361
12-16 cam_xclkb Configuration 1361
12-17 Camera ISP Interrupts 1365
12-18 CBUFF Interrupt Details 1366
12-19 CSI1 Receiver Interrupt Details 1367

12-20 CSI2 Receiver Event Generation (CSI2_IRQSTATUS and CSI2_IRQENABLE Registers) 1368
12-21 CSI2 Receiver Event Generation (CSI2_COMPLEXIO_IRQSTATUS and CSI2_COMPLEXIO_IRQENABLE Registers) 1368
12-22 CSI2 Receiver Event Generation (CSI2_CTx_IRQSTATUS and CSI2_CTx_IRQENABLE Registers) 1369
12-23 Allowed Data Flows for Hardware at the Input of the CCDC Module 1372
12-24 ECC Event Logging 1382
12-25 Possible Time-Out Value for RxMode Counter 1387
12-26 Control-Signal Generator: CNTCLK Frequencies 1390
12-27 Data-Lane Shifter 1392
12-28 Allowed Data Flows Through the CCDC 1394
12-29 Reformatter Output Limitations 1399
12-30 CCDC_SDOFST Description 1403
12-31 Memory Output Format for RAW Data 1405
12-32 Memory Output Format for YUV Data 1406
12-33 Image Cropping by Preview Functions 1412
12-34 Resizer Use Constraints 1413
12-35 Arrangement of the Filter Coefficients 1417
12-36 Input Size Calculations 1418
12-37 Processing Example for 1:2:56 Horizontal Resize 1423
12-38 White Balance Field-to-Pattern Assignments 1427
12-39 Regions and Bins for Histogram 1427
12-40 Central Resource SBL Fixed Parameters 1430
12-41 Central Resource SBL Number of Request Registers 1432
12-42 Internal Variables 1437
12-43 Internal State After Reset 1438
12-44 Address Identification 1438
12-45 Address Translation 1439
12-46 Window Level Increment 1440
12-47 Window Level Comparison 1440
12-48 CSI1_CTRL1[7:3] FORMAT and CSI1_CTRL[11] VP_ONLY_EN = 1 Settings 1444
12-49 CSI2 Receiver-Supported Data Types 1451
12-50 CCDC Required Configuration Parameters 1457
12-51 CCDC Conditional Configuration Parameters 1457
12-52 Conventional Readout Pattern 1 to 1 1467
12-53 Dual Readout Pattern 1 to 1 1468
12-54 Dual Readout Pattern 1 to 3 1469
12-55 CCDC_ALAW [2:0] GWDI 1470
12-56 Preview Engine Required Configuration Parameters 1474
12-57 Preview Engine Conditional Configuration Parameters 1474
12-58 Resizer Required Configuration Parameters 1478
12-59 Resizer Conditional Configuration Parameters 1479
12-60 How to Set Input Height and Width 1482
12-61 AF Engine Required Configuration Parameters 1483
12-62 AF Engine Conditional Configuration Parameters 1484
12-63 AEW Engine Required Configuration Parameters 1484
12-64 Histogram Required Configuration Parameters 1486
12-65 Histogram Conditional Configuration Parameters 1487
12-66 SBL Write-Buffer Overflow Events 1489
12-67 SBL Read-Buffer Underflow Events 1490
12-68 Camera ISP Instance Summary 1495
12-69 ISP Register Summary 1495

12-70 ISP_REVISION 1496
12-71 Register Call Summary for Register ISP_REVISION 1496
12-72 ISP_SYSCONFIG 1496
12-73 Register Call Summary for Register ISP_SYSCONFIG 1497
12-74 ISP_SYSSTATUS 1497
12-75 Register Call Summary for Register ISP_SYSSTATUS 1497
12-76 ISP_IRQ0ENABLE 1498
12-77 Register Call Summary for Register ISP_IRQ0ENABLE 1500
12-78 ISP_IRQ0STATUS 1500
12-79 Register Call Summary for Register ISP_IRQ0STATUS 1504
12-80 ISP_IRQ1ENABLE 1504
12-81 Register Call Summary for Register ISP_IRQ1ENABLE 1506
12-82 ISP_IRQ1STATUS 1507
12-83 Register Call Summary for Register ISP_IRQ1STATUS 1510
12-84 TCTRL_GRESET_LENGTH 1511
12-85 Register Call Summary for Register TCTRL_GRESET_LENGTH 1511
12-86 TCTRL_PSTRB_REPLAY 1511
12-87 Register Call Summary for Register TCTRL_PSTRB_REPLAY 1512
12-88 ISP_CTRL 1512
12-89 Register Call Summary for Register ISP_CTRL 1515
12-90 TCTRL_CTRL 1515
12-91 Register Call Summary for Register TCTRL_CTRL 1517
12-92 TCTRL_FRAME 1518
12-93 Register Call Summary for Register TCTRL_FRAME 1518
12-94 TCTRL_PSTRB_DELAY 1518
12-95 Register Call Summary for Register TCTRL_PSTRB_DELAY 1518
12-96 TCTRL_STRB_DELAY 1519
12-97 Register Call Summary for Register TCTRL_STRB_DELAY 1519
12-98 TCTRL_SHUT_DELAY 1519
12-99 Register Call Summary for Register TCTRL_SHUT_DELAY 1520
12-100 TCTRL_PSTRB_LENGTH 1520
12-101 Register Call Summary for Register TCTRL_PSTRB_LENGTH 1520
12-102 TCTRL_STRB_LENGTH 1520
12-103 Register Call Summary for Register TCTRL_STRB_LENGTH 1521
12-104 TCTRL_SHUT_LENGTH 1521
12-105 Register Call Summary for Register TCTRL_SHUT_LENGTH 1521
12-106 ISP_CBUFF Register Summary 1521
12-107 CBUFF_REVISION 1522
12-108 Register Call Summary for Register CBUFF_REVISION 1522
12-109 CBUFF_SYSCONFIG 1522
12-110 Register Call Summary for Register CBUFF_SYSCONFIG 1522
12-111 CBUFF_SYSSTATUS 1523
12-112 Register Call Summary for Register CBUFF_SYSSTATUS 1523
12-113 CBUFF_IRQSTATUS 1523
12-114 Register Call Summary for Register CBUFF_IRQSTATUS 1524
12-115 CBUFF_IRQENABLE 1524
12-116 Register Call Summary for Register CBUFF_IRQENABLE 1525
12-117 CBUFFx_CTRL 1525
12-118 Register Call Summary for Register CBUFFx_CTRL 1526
12-119 CBUFFx_STATUS 1527
12-120 Register Call Summary for Register CBUFFx_STATUS 1527

12-121 CBUFFx_START 1527
12-122 Register Call Summary for Register CBUFFx_START 1528
12-123 CBUFFx_END 1528
12-124 Register Call Summary for Register CBUFFx_END 1528
12-125 CBUFFx_WINDOWSIZE 1528
12-126 Register Call Summary for Register CBUFFx_WINDOWSIZE 1529
12-127 CBUFFx_THRESHOLD 1529
12-128 Register Call Summary for Register CBUFFx_THRESHOLD 1529
12-129 ISP_CSI1B Register Summary 1529
12-130 CSI1B_REVISION 1530
12-131 Register Call Summary for Register CSI1B_REVISION 1530
12-132 CSI1B_SYSCONFIG 1531
12-133 Register Call Summary for Register CSI1B_SYSCONFIG 1531
12-134 CSI1B_SYSSTATUS 1531
12-135 Register Call Summary for Register CSI1B_SYSSTATUS 1532
12-136 CSI1B_IRQENABLE 1532
12-137 Register Call Summary for Register CSI1B_IRQENABLE 1533
12-138 CSI1B_IRQSTATUS 1533
12-139 Register Call Summary for Register CSI1B_IRQSTATUS 1534
12-140 CSI1B_LCM_IRQENABLE 1535
12-141 Register Call Summary for Register CSI1B_LCM_IRQENABLE 1535
12-142 CSI1B_LCM_IRQSTATUS 1535
12-143 Register Call Summary for Register CSI1B_LCM_IRQSTATUS 1536
12-144 CSI1B_CTRL 1536
12-145 Register Call Summary for Register CSI1B_CTRL 1537
12-146 CSI1B_DBG 1537
12-147 Register Call Summary for Register CSI1B_DBG 1538
12-148 CSI1B_GNQ 1538
12-149 Register Call Summary for Register CSI1B_GNQ 1538
12-150 CSI1B_CTRL1 1538
12-151 Register Call Summary for Register CSI1B_CTRL1 1540
12-152 CSI1B_CODE 1540
12-153 Register Call Summary for Register CSI1B_CODE 1540
12-154 CSI1B_STAT_START 1540
12-155 Register Call Summary for Register CSI1B_STAT_START 1541
12-156 CSI1B_STAT_SIZE 1541
12-157 Register Call Summary for Register CSI1B_STAT_SIZE 1541
12-158 CSI1B_SOF_ADDR 1541
12-159 Register Call Summary for Register CSI1B_SOF_ADDR 1541
12-160 CSI1B_EOF_ADDR 1542
12-161 Register Call Summary for Register CSI1B_EOF_ADDR 1542
12-162 CSI1B_DAT_START 1542
12-163 Register Call Summary for Register CSI1B_DAT_START 1542
12-164 CSI1B_DAT_SIZE 1543
12-165 Register Call Summary for Register CSI1B_DAT_SIZE 1543
12-166 CSI1B_DAT_PING_ADDR 1543
12-167 Register Call Summary for Register CSI1B_DAT_PING_ADDR 1543
12-168 CSI1B_DAT_PONG_ADDR 1544
12-169 Register Call Summary for Register CSI1B_DAT_PONG_ADDR 1544
12-170 CSI1B_DAT_OFST 1544
12-171 Register Call Summary for Register CSI1B_DAT_OFST 1544

12-172 CSI1B_LCM_CTRL 1545
12-173 Register Call Summary for Register CSI1B_LCM_CTRL 1546
12-174 CSI1B_LCM_VSIZE 1546
12-175 Register Call Summary for Register CSI1B_LCM_VSIZE 1547
12-176 CSI1B_LCM_HSIZE 1547
12-177 Register Call Summary for Register CSI1B_LCM_HSIZE 1547
12-178 CSI1B_LCM_PREFETCH 1547
12-179 Register Call Summary for Register CSI1B_LCM_PREFETCH 1548
12-180 CSI1B_LCM_SRC_ADDR 1548
12-181 Register Call Summary for Register CSI1B_LCM_SRC_ADDR 1548
12-182 CSI1B_LCM_SRC_OFST 1548
12-183 Register Call Summary for Register CSI1B_LCM_SRC_OFST 1548
12-184 CSI1B_LCM_DST_ADDR 1549
12-185 Register Call Summary for Register CSI1B_LCM_DST_ADDR 1549
12-186 CSI1B_LCM_DST_OFST 1549
12-187 Register Call Summary for Register CSI1B_LCM_DST_OFST 1549
12-188 ISP_CCDC Register Summary 1550
12-189 CCDC_PID 1551
12-190 Register Call Summary for Register CCDC_PID 1551
12-191 CCDC_PCR 1551
12-192 Register Call Summary for Register CCDC_PCR 1552
12-193 CCDC_SYN_MODE 1552
12-194 Register Call Summary for Register CCDC_SYN_MODE 1554
12-195 CCDC_HD_VD_WID 1554
12-196 Register Call Summary for Register CCDC_HD_VD_WID 1555
12-197 CCDC_PIX_LINES 1555
12-198 Register Call Summary for Register CCDC_PIX_LINES 1556
12-199 CCDC_HORZ_INFO 1556
12-200 Register Call Summary for Register CCDC_HORZ_INFO 1556
12-201 CCDC_VERT_START 1557
12-202 Register Call Summary for Register CCDC_VERT_START 1557
12-203 CCDC_VERT_LINES 1557
12-204 Register Call Summary for Register CCDC_VERT_LINES 1558
12-205 CCDC_CULLING 1558
12-206 Register Call Summary for Register CCDC_CULLING 1558
12-207 CCDC_HSIZE_OFF 1559
12-208 Register Call Summary for Register CCDC_HSIZE_OFF 1559
12-209 CCDC_SDOFST 1559
12-210 Register Call Summary for Register CCDC_SDOFST 1561
12-211 CCDC_SDR_ADDR 1561
12-212 Register Call Summary for Register CCDC_SDR_ADDR 1561
12-213 CCDC_CLAMP 1561
12-214 Register Call Summary for Register CCDC_CLAMP 1562
12-215 CCDC_DCSUB 1562
12-216 Register Call Summary for Register CCDC_DCSUB 1563
12-217 CCDC_COLPTN 1563
12-218 Register Call Summary for Register CCDC_COLPTN 1565
12-219 CCDC_BLKCMP 1565
12-220 Register Call Summary for Register CCDC_BLKCMP 1566
12-221 CCDC_FPC 1566
12-222 Register Call Summary for Register CCDC_FPC 1567

12-223 CCDC_FPC_ADDR 1567
12-224 Register Call Summary for Register CCDC_FPC_ADDR 1567
12-225 CCDC_VDINT 1568
12-226 Register Call Summary for Register CCDC_VDINT 1568
12-227 CCDC_ALAW 1569
12-228 Register Call Summary for Register CCDC_ALAW 1569
12-229 CCDC_REC656IF 1569
12-230 Register Call Summary for Register CCDC_REC656IF 1570
12-231 CCDC_CFG 1570
12-232 Register Call Summary for Register CCDC_CFG 1571
12-233 CCDC_FMTCFG 1571
12-234 Register Call Summary for Register CCDC_FMTCFG 1572
12-235 CCDC_FMT_HORZ 1573
12-236 Register Call Summary for Register CCDC_FMT_HORZ 1573
12-237 CCDC_FMT_VERT 1574
12-238 Register Call Summary for Register CCDC_FMT_VERT 1574
12-239 CCDC_FMT_ADDR_i 1574
12-240 Register Call Summary for Register CCDC_FMT_ADDR_i 1575
12-241 CCDC_PRGEVEN0 1575
12-242 Register Call Summary for Register CCDC_PRGEVEN0 1575
12-243 CCDC_PRGEVEN1 1576
12-244 Register Call Summary for Register CCDC_PRGEVEN1 1576
12-245 CCDC_PRGODD0 1576
12-246 Register Call Summary for Register CCDC_PRGODD0 1577
12-247 CCDC_PRGODD1 1577
12-248 Register Call Summary for Register CCDC_PRGODD1 1577
12-249 CCDC_VP_OUT 1578
12-250 Register Call Summary for Register CCDC_VP_OUT 1578
12-251 CCDC_LSC_CONFIG 1579
12-252 Register Call Summary for Register CCDC_LSC_CONFIG 1580
12-253 CCDC_LSC_INITIAL 1580
12-254 Register Call Summary for Register CCDC_LSC_INITIAL 1581
12-255 CCDC_LSC_TABLE_BASE 1581
12-256 Register Call Summary for Register CCDC_LSC_TABLE_BASE 1581
12-257 CCDC_LSC_TABLE_OFFSET 1581
12-258 Register Call Summary for Register CCDC_LSC_TABLE_OFFSET 1582
12-259 ISP_HIST Register Summary 1582
12-260 HIST_PID 1582
12-261 Register Call Summary for Register HIST_PID 1583
12-262 HIST_PCR 1583
12-263 Register Call Summary for Register HIST_PCR 1583
12-264 HIST_CNT 1584
12-265 Register Call Summary for Register HIST_CNT 1584
12-266 HIST_WB_GAIN 1585
12-267 Register Call Summary for Register HIST_WB_GAIN 1585
12-268 HIST_Rn_HORZ 1585
12-269 Register Call Summary for Register HIST_Rn_HORZ 1586
12-270 HIST_Rn_VERT 1586
12-271 Register Call Summary for Register HIST_Rn_VERT 1586
12-272 HIST_ADDR 1586
12-273 Register Call Summary for Register HIST_ADDR 1587

12-274 HIST_DATA 1587
12-275 Register Call Summary for Register HIST_DATA 1587
12-276 HIST_RADD 1587
12-277 Register Call Summary for Register HIST_RADD 1587
12-278 HIST_RADD_OFF 1588
12-279 Register Call Summary for Register HIST_RADD_OFF 1588
12-280 HIST_H_V_INFO 1588
12-281 Register Call Summary for Register HIST_H_V_INFO 1588
12-282 ISP_H3A Register Summary 1589
12-283 H3A_PID 1590
12-284 Register Call Summary for Register H3A_PID 1590
12-285 H3A_PCR 1590
12-286 Register Call Summary for Register H3A_PCR 1591
12-287 H3A_AFPAX1 1591
12-288 Register Call Summary for Register H3A_AFPAX1 1592
12-289 H3A_AFPAX2 1592
12-290 Register Call Summary for Register H3A_AFPAX2 1592
12-291 H3A_AFPAXSTART 1593
12-292 Register Call Summary for Register H3A_AFPAXSTART 1593
12-293 H3A_AFIIRSH 1593
12-294 Register Call Summary for Register H3A_AFIIRSH 1593
12-295 H3A_AFBUFST 1594
12-296 Register Call Summary for Register H3A_AFBUFST 1594
12-297 H3A_AFCOEF010 1594
12-298 Register Call Summary for Register H3A_AFCOEF010 1595
12-299 H3A_AFCOEF032 1595
12-300 Register Call Summary for Register H3A_AFCOEF032 1595
12-301 H3A_AFCOEF054 1595
12-302 Register Call Summary for Register H3A_AFCOEF054 1596
12-303 H3A_AFCOEF076 1596
12-304 Register Call Summary for Register H3A_AFCOEF076 1596
12-305 H3A_AFCOEF098 1596
12-306 Register Call Summary for Register H3A_AFCOEF098 1596
12-307 H3A_AFCOEF0010 1597
12-308 Register Call Summary for Register H3A_AFCOEF0010 1597
12-309 H3A_AFCOEF110 1597
12-310 Register Call Summary for Register H3A_AFCOEF110 1597
12-311 H3A_AFCOEF132 1598
12-312 Register Call Summary for Register H3A_AFCOEF132 1598
12-313 H3A_AFCOEF154 1598
12-314 Register Call Summary for Register H3A_AFCOEF154 1598
12-315 H3A_AFCOEF176 1599
12-316 Register Call Summary for Register H3A_AFCOEF176 1599
12-317 H3A_AFCOEF198 1599
12-318 Register Call Summary for Register H3A_AFCOEF198 1599
12-319 H3A_AFCOEF1010 1600
12-320 Register Call Summary for Register H3A_AFCOEF1010 1600
12-321 H3A_AEWWIN1 1600
12-322 Register Call Summary for Register H3A_AEWWIN1 1601
12-323 H3A_AEWINSTART 1601
12-324 Register Call Summary for Register H3A_AEWINSTART 1601

12-325 H3A_AEWINBLK 1601
12-326 Register Call Summary for Register H3A_AEWINBLK 1602
12-327 H3A_AEWSUBWIN 1602
12-328 Register Call Summary for Register H3A_AEWSUBWIN 1602
12-329 H3A_AEWBUFST 1603
12-330 Register Call Summary for Register H3A_AEWBUFST 1603
12-331 ISP_PREVIEW Register Summary 1603
12-332 PRV_PID 1604
12-333 Register Call Summary for Register PRV_PID 1604
12-334 PRV_PCR 1605
12-335 Register Call Summary for Register PRV_PCR 1607
12-336 PRV_HORZ_INFO 1607
12-337 Register Call Summary for Register PRV_HORZ_INFO 1608
12-338 PRV_VERT_INFO 1608
12-339 Register Call Summary for Register PRV_VERT_INFO 1608
12-340 PRV_RSDR_ADDR 1609
12-341 Register Call Summary for Register PRV_RSDR_ADDR 1609
12-342 PRV_RADR_OFFSET 1609
12-343 Register Call Summary for Register PRV_RADR_OFFSET 1610
12-344 PRV_DSDR_ADDR 1610
12-345 Register Call Summary for Register PRV_DSDR_ADDR 1610
12-346 PRV_DRKF_OFFSET 1611
12-347 Register Call Summary for Register PRV_DRKF_OFFSET 1611
12-348 PRV_WSDR_ADDR 1611
12-349 Register Call Summary for Register PRV_WSDR_ADDR 1611
12-350 PRV_WADD_OFFSET 1612
12-351 Register Call Summary for Register PRV_WADD_OFFSET 1612
12-352 PRV_AVE 1612
12-353 Register Call Summary for Register PRV_AVE 1613
12-354 PRV_HMED 1613
12-355 Register Call Summary for Register PRV_HMED 1614
12-356 PRV_NF 1614
12-357 Register Call Summary for Register PRV_NF 1614
12-358 PRV_WB_DGAIN 1615
12-359 Register Call Summary for Register PRV_WB_DGAIN 1615
12-360 PRV_WBGAIN 1615
12-361 Register Call Summary for Register PRV_WBGAIN 1616
12-362 PRV_WBSEL 1616
12-363 Register Call Summary for Register PRV_WBSEL 1618
12-364 PRV_CFA 1618
12-365 Register Call Summary for Register PRV_CFA 1618
12-366 PRV_BLKADJOFF 1619
12-367 Register Call Summary for Register PRV_BLKADJOFF 1619
12-368 PRV_RGB_MAT1 1619
12-369 Register Call Summary for Register PRV_RGB_MAT1 1619
12-370 PRV_RGB_MAT2 1620
12-371 Register Call Summary for Register PRV_RGB_MAT2 1620
12-372 PRV_RGB_MAT3 1620
12-373 Register Call Summary for Register PRV_RGB_MAT3 1621
12-374 PRV_RGB_MAT4 1621
12-375 Register Call Summary for Register PRV_RGB_MAT4 1621

12-376 PRV_RGB_MAT5 ... 1621
12-377 Register Call Summary for Register PRV_RGB_MAT5 ... 1622
12-378 PRV_RGB_OFF1 ... 1622
12-379 Register Call Summary for Register PRV_RGB_OFF1 ... 1622
12-380 PRV_RGB_OFF2 ... 1622
12-381 Register Call Summary for Register PRV_RGB_OFF2 ... 1623
12-382 PRV_CSC0 ... 1623
12-383 Register Call Summary for Register PRV_CSC0 ... 1623
12-384 PRV_CSC1 ... 1624
12-385 Register Call Summary for Register PRV_CSC1 ... 1624
12-386 PRV_CSC2 ... 1624
12-387 Register Call Summary for Register PRV_CSC2 ... 1625
12-388 PRV_CSC_OFFSET ... 1625
12-389 Register Call Summary for Register PRV_CSC_OFFSET ... 1625
12-390 PRV_CNT_BRT ... 1626
12-391 Register Call Summary for Register PRV_CNT_BRT ... 1626
12-392 PRV_CSUP ... 1626
12-393 Register Call Summary for Register PRV_CSUP ... 1627
12-394 PRV_SETUP_YC ... 1627
12-395 Register Call Summary for Register PRV_SETUP_YC ... 1627
12-396 PRV_SET_TBL_ADDR ... 1627
12-397 Register Call Summary for Register PRV_SET_TBL_ADDR ... 1628
12-398 PRV_SET_TBL_DATA ... 1628
12-399 Register Call Summary for Register PRV_SET_TBL_DATA ... 1628
12-400 PRV_CDC_THRx ... 1628
12-401 Register Call Summary for Register PRV_CDC_THRx ... 1629
12-402 ISP_RESIZER Register Summary ... 1629
12-403 RSZ_PID ... 1630
12-404 Register Call Summary for Register RSZ_PID ... 1630
12-405 RSZ_PCR ... 1630
12-406 Register Call Summary for Register RSZ_PCR ... 1631
12-407 RSZ_CNT ... 1631
12-408 Register Call Summary for Register RSZ_CNT ... 1632
12-409 RSZ_OUT_SIZE ... 1633
12-410 Register Call Summary for Register RSZ_OUT_SIZE ... 1633
12-411 RSZ_IN_START ... 1633
12-412 Register Call Summary for Register RSZ_IN_START ... 1634
12-413 RSZ_IN_SIZE ... 1634
12-414 Register Call Summary for Register RSZ_IN_SIZE ... 1634
12-415 RSZ_SDR_INADD ... 1635
12-416 Register Call Summary for Register RSZ_SDR_INADD ... 1635
12-417 RSZ_SDR_INOFF ... 1635
12-418 Register Call Summary for Register RSZ_SDR_INOFF ... 1636
12-419 RSZ_SDR_OUTADD ... 1636
12-420 Register Call Summary for Register RSZ_SDR_OUTADD ... 1636
12-421 RSZ_SDR_OUTOFF ... 1636
12-422 Register Call Summary for Register RSZ_SDR_OUTOFF ... 1637
12-423 RSZ_HFILT10 ... 1637
12-424 Register Call Summary for Register RSZ_HFILT10 ... 1637
12-425 RSZ_HFILT32 ... 1638
12-426 Register Call Summary for Register RSZ_HFILT32 ... 1638

12-427 RSZ_HFILT54 1638
12-428 Register Call Summary for Register RSZ_HFILT54 1638
12-429 RSZ_HFILT76 1639
12-430 Register Call Summary for Register RSZ_HFILT76 1639
12-431 RSZ_HFILT98 1639
12-432 Register Call Summary for Register RSZ_HFILT98 1639
12-433 RSZ_HFILT1110 1640
12-434 Register Call Summary for Register RSZ_HFILT1110 1640
12-435 RSZ_HFILT1312 1640
12-436 Register Call Summary for Register RSZ_HFILT1312 1640
12-437 RSZ_HFILT1514 1641
12-438 Register Call Summary for Register RSZ_HFILT1514 1641
12-439 RSZ_HFILT1716 1641
12-440 Register Call Summary for Register RSZ_HFILT1716 1641
12-441 RSZ_HFILT1918 1642
12-442 Register Call Summary for Register RSZ_HFILT1918 1642
12-443 RSZ_HFILT2120 1642
12-444 Register Call Summary for Register RSZ_HFILT2120 1642
12-445 RSZ_HFILT2322 1643
12-446 Register Call Summary for Register RSZ_HFILT2322 1643
12-447 RSZ_HFILT2524 1643
12-448 Register Call Summary for Register RSZ_HFILT2524 1643
12-449 RSZ_HFILT2726 1644
12-450 Register Call Summary for Register RSZ_HFILT2726 1644
12-451 RSZ_HFILT2928 1644
12-452 Register Call Summary for Register RSZ_HFILT2928 1644
12-453 RSZ_HFILT3130 1645
12-454 Register Call Summary for Register RSZ_HFILT3130 1645
12-455 RSZ_VFILT10 1645
12-456 Register Call Summary for Register RSZ_VFILT10 1645
12-457 RSZ_VFILT32 1646
12-458 Register Call Summary for Register RSZ_VFILT32 1646
12-459 RSZ_VFILT54 1646
12-460 Register Call Summary for Register RSZ_VFILT54 1647
12-461 RSZ_VFILT76 1647
12-462 Register Call Summary for Register RSZ_VFILT76 1647
12-463 RSZ_VFILT98 1647
12-464 Register Call Summary for Register RSZ_VFILT98 1648
12-465 RSZ_VFILT1110 1648
12-466 Register Call Summary for Register RSZ_VFILT1110 1648
12-467 RSZ_VFILT1312 1648
12-468 Register Call Summary for Register RSZ_VFILT1312 1649
12-469 RSZ_VFILT1514 1649
12-470 Register Call Summary for Register RSZ_VFILT1514 1649
12-471 RSZ_VFILT1716 1649
12-472 Register Call Summary for Register RSZ_VFILT1716 1650
12-473 RSZ_VFILT1918 1650
12-474 Register Call Summary for Register RSZ_VFILT1918 1650
12-475 RSZ_VFILT2120 1650
12-476 Register Call Summary for Register RSZ_VFILT2120 1651
12-477 RSZ_VFILT2322 1651

12-478 Register Call Summary for Register RSZ_VFILT2322 1651
12-479 RSZ_VFILT2524 1651
12-480 Register Call Summary for Register RSZ_VFILT2524 1652
12-481 RSZ_VFILT2726 1652
12-482 Register Call Summary for Register RSZ_VFILT2726 1652
12-483 RSZ_VFILT2928 1652
12-484 Register Call Summary for Register RSZ_VFILT2928 1653
12-485 RSZ_VFILT3130 1653
12-486 Register Call Summary for Register RSZ_VFILT3130 1653
12-487 RSZ_YENH 1653
12-488 Register Call Summary for Register RSZ_YENH 1654
12-489 ISP_SBL Register Summary 1654
12-490 SBL_PID 1656
12-491 Register Call Summary for Register SBL_PID 1656
12-492 SBL_PCR 1656
12-493 Register Call Summary for Register SBL_PCR 1658
12-494 SBL_GLB_REG_0 1658
12-495 Register Call Summary for Register SBL_GLB_REG_0 1659
12-496 SBL_GLB_REG_1 1659
12-497 Register Call Summary for Register SBL_GLB_REG_1 1660
12-498 SBL_GLB_REG_2 1660
12-499 Register Call Summary for Register SBL_GLB_REG_2 1661
12-500 SBL_GLB_REG_3 1661
12-501 Register Call Summary for Register SBL_GLB_REG_3 1662
12-502 SBL_GLB_REG_4 1662
12-503 Register Call Summary for Register SBL_GLB_REG_4 1663
12-504 SBL_GLB_REG_5 1663
12-505 Register Call Summary for Register SBL_GLB_REG_5 1664
12-506 SBL_GLB_REG_6 1664
12-507 Register Call Summary for Register SBL_GLB_REG_6 1665
12-508 SBL_GLB_REG_7 1665
12-509 Register Call Summary for Register SBL_GLB_REG_7 1666
12-510 SBL_CCDC_WR_0 1666
12-511 Register Call Summary for Register SBL_CCDC_WR_0 1667
12-512 SBL_CCDC_WR_1 1667
12-513 Register Call Summary for Register SBL_CCDC_WR_1 1667
12-514 SBL_CCDC_WR_2 1668
12-515 Register Call Summary for Register SBL_CCDC_WR_2 1668
12-516 SBL_CCDC_WR_3 1668
12-517 Register Call Summary for Register SBL_CCDC_WR_3 1669
12-518 SBL_CCDC_FP_RD_0 1669
12-519 Register Call Summary for Register SBL_CCDC_FP_RD_0 1669
12-520 SBL_CCDC_FP_RD_1 1669
12-521 Register Call Summary for Register SBL_CCDC_FP_RD_1 1670
12-522 SBL_PRV_RD_0 1670
12-523 Register Call Summary for Register SBL_PRV_RD_0 1671
12-524 SBL_PRV_RD_1 1671
12-525 Register Call Summary for Register SBL_PRV_RD_1 1671
12-526 SBL_PRV_RD_2 1671
12-527 Register Call Summary for Register SBL_PRV_RD_2 1672
12-528 SBL_PRV_RD_3 1672

12-529 Register Call Summary for Register SBL_PRV_RD_3 1673
12-530 SBL_PRV_WR_0 1673
12-531 Register Call Summary for Register SBL_PRV_WR_0 1673
12-532 SBL_PRV_WR_1 1673
12-533 Register Call Summary for Register SBL_PRV_WR_1 1674
12-534 SBL_PRV_WR_2 1674
12-535 Register Call Summary for Register SBL_PRV_WR_2 1674
12-536 SBL_PRV_WR_3 1675
12-537 Register Call Summary for Register SBL_PRV_WR_3 1675
12-538 SBL_PRV_DK_RD_0 1675
12-539 Register Call Summary for Register SBL_PRV_DK_RD_0 1676
12-540 SBL_PRV_DK_RD_1 1676
12-541 Register Call Summary for Register SBL_PRV_DK_RD_1 1676
12-542 SBL_PRV_DK_RD_2 1677
12-543 Register Call Summary for Register SBL_PRV_DK_RD_2 1677
12-544 SBL_PRV_DK_RD_3 1677
12-545 Register Call Summary for Register SBL_PRV_DK_RD_3 1678
12-546 SBL_RSZ_RD_0 1678
12-547 Register Call Summary for Register SBL_RSZ_RD_0 1678
12-548 SBL_RSZ_RD_1 1679
12-549 Register Call Summary for Register SBL_RSZ_RD_1 1679
12-550 SBL_RSZ_RD_2 1679
12-551 Register Call Summary for Register SBL_RSZ_RD_2 1680
12-552 SBL_RSZ_RD_3 1680
12-553 Register Call Summary for Register SBL_RSZ_RD_3 1680
12-554 SBL_RSZ1_WR_0 1681
12-555 Register Call Summary for Register SBL_RSZ1_WR_0 1681
12-556 SBL_RSZ1_WR_1 1681
12-557 Register Call Summary for Register SBL_RSZ1_WR_1 1682
12-558 SBL_RSZ1_WR_2 1682
12-559 Register Call Summary for Register SBL_RSZ1_WR_2 1682
12-560 SBL_RSZ1_WR_3 1682
12-561 Register Call Summary for Register SBL_RSZ1_WR_3 1683
12-562 SBL_RSZ2_WR_0 1683
12-563 Register Call Summary for Register SBL_RSZ2_WR_0 1683
12-564 SBL_RSZ2_WR_1 1684
12-565 Register Call Summary for Register SBL_RSZ2_WR_1 1684
12-566 SBL_RSZ2_WR_2 1684
12-567 Register Call Summary for Register SBL_RSZ2_WR_2 1685
12-568 SBL_RSZ2_WR_3 1685
12-569 Register Call Summary for Register SBL_RSZ2_WR_3 1685
12-570 SBL_RSZ3_WR_0 1685
12-571 Register Call Summary for Register SBL_RSZ3_WR_0 1686
12-572 SBL_RSZ3_WR_1 1686
12-573 Register Call Summary for Register SBL_RSZ3_WR_1 1686
12-574 SBL_RSZ3_WR_2 1687
12-575 Register Call Summary for Register SBL_RSZ3_WR_2 1687
12-576 SBL_RSZ3_WR_3 1687
12-577 Register Call Summary for Register SBL_RSZ3_WR_3 1688
12-578 SBL_RSZ4_WR_0 1688
12-579 Register Call Summary for Register SBL_RSZ4_WR_0 1688

12-580 SBL_RSZ4_WR_1 ... 1688
12-581 Register Call Summary for Register SBL_RSZ4_WR_1 ... 1689
12-582 SBL_RSZ4_WR_2 ... 1689
12-583 Register Call Summary for Register SBL_RSZ4_WR_2 ... 1689
12-584 SBL_RSZ4_WR_3 ... 1690
12-585 Register Call Summary for Register SBL_RSZ4_WR_3 ... 1690
12-586 SBL_HIST_RD_0 ... 1690
12-587 Register Call Summary for Register SBL_HIST_RD_0 ... 1691
12-588 SBL_HIST_RD_1 ... 1691
12-589 Register Call Summary for Register SBL_HIST_RD_1 ... 1691
12-590 SBL_H3A_AF_WR_0 ... 1692
12-591 Register Call Summary for Register SBL_H3A_AF_WR_0 ... 1692
12-592 SBL_H3A_AF_WR_1 ... 1692
12-593 Register Call Summary for Register SBL_H3A_AF_WR_1 ... 1693
12-594 SBL_H3A_AEAWB_WR_0 ... 1693
12-595 Register Call Summary for Register SBL_H3A_AEAWB_WR_0 ... 1693
12-596 SBL_H3A_AEAWB_WR_1 ... 1693
12-597 Register Call Summary for Register SBL_H3A_AEAWB_WR_1 ... 1694
12-598 SBL_CSIA_WR_0 ... 1694
12-599 Register Call Summary for Register SBL_CSIA_WR_0 ... 1694
12-600 SBL_CSIA_WR_1 ... 1695
12-601 Register Call Summary for Register SBL_CSIA_WR_1 ... 1695
12-602 SBL_CSIA_WR_2 ... 1695
12-603 Register Call Summary for Register SBL_CSIA_WR_2 ... 1696
12-604 SBL_CSIA_WR_3 ... 1696
12-605 Register Call Summary for Register SBL_CSIA_WR_3 ... 1696
12-606 SBL_CSIB_WR_0 ... 1696
12-607 Register Call Summary for Register SBL_CSIB_WR_0 ... 1697
12-608 SBL_CSIB_WR_1 ... 1697
12-609 Register Call Summary for Register SBL_CSIB_WR_1 ... 1697
12-610 SBL_CSIB_WR_2 ... 1698
12-611 Register Call Summary for Register SBL_CSIB_WR_2 ... 1698
12-612 SBL_CSIB_WR_3 ... 1698
12-613 Register Call Summary for Register SBL_CSIB_WR_3 ... 1699
12-614 SBL_SDR_REQ_EXP ... 1699
12-615 Register Call Summary for Register SBL_SDR_REQ_EXP ... 1699
12-616 ISP_CSI2A Register Summary ... 1700
12-617 CSI2_REVISION ... 1700
12-618 Register Call Summary for Register CSI2_REVISION ... 1701
12-619 CSI2_SYSCONFIG ... 1701
12-620 Register Call Summary for Register CSI2_SYSCONFIG ... 1701
12-621 CSI2_SYSSTATUS ... 1702
12-622 Register Call Summary for Register CSI2_SYSSTATUS ... 1702
12-623 CSI2_IRQSTATUS ... 1702
12-624 Register Call Summary for Register CSI2_IRQSTATUS ... 1704
12-625 CSI2_IRQENABLE ... 1704
12-626 Register Call Summary for Register CSI2_IRQENABLE ... 1705
12-627 CSI2_CTRL ... 1706
12-628 Register Call Summary for Register CSI2_CTRL ... 1707
12-629 CSI2_DBG_H ... 1707
12-630 Register Call Summary for Register CSI2_DBG_H ... 1707

12-631 CSI2_GNQ 1708
12-632 Register Call Summary for Register CSI2_GNQ 1708
12-633 CSI2_COMPLEXIO_CFG1 1708
12-634 Register Call Summary for Register CSI2_COMPLEXIO_CFG1 1709
12-635 CSI2_COMPLEXIO1_IRQSTATUS 1710
12-636 Register Call Summary for Register CSI2_COMPLEXIO1_IRQSTATUS 1713
12-637 CSI2_SHORT_PACKET 1713
12-638 Register Call Summary for Register CSI2_SHORT_PACKET 1713
12-639 CSI2_COMPLEXIO1_IRQENABLE 1713
12-640 Register Call Summary for Register CSI2_COMPLEXIO1_IRQENABLE 1715
12-641 CSI2_DBG_P 1716
12-642 Register Call Summary for Register CSI2_DBG_P 1716
12-643 CSI2_TIMING 1716
12-644 Register Call Summary for Register CSI2_TIMING 1717
12-645 CSI2_CTx_CTRL1 1717
12-646 Register Call Summary for Register CSI2_CTx_CTRL1 1718
12-647 CSI2_CTx_CTRL2 1719
12-648 Register Call Summary for Register CSI2_CTx_CTRL2 1721
12-649 CSI2_CTx_DAT_OFST 1721
12-650 Register Call Summary for Register CSI2_CTx_DAT_OFST 1722
12-651 CSI2_CTx_DAT_PING_ADDR 1722
12-652 Register Call Summary for Register CSI2_CTx_DAT_PING_ADDR 1722
12-653 CSI2_CTx_DAT_PONG_ADDR 1723
12-654 Register Call Summary for Register CSI2_CTx_DAT_PONG_ADDR 1723
12-655 CSI2_CTx_IRQENABLE 1723
12-656 Register Call Summary for Register CSI2_CTx_IRQENABLE 1724
12-657 CSI2_CTx_IRQSTATUS 1724
12-658 Register Call Summary for Register CSI2_CTx_IRQSTATUS 1725
12-659 CSI2_CTx_CTRL3 1726
12-660 Register Call Summary for Register CSI2_CTx_CTRL3 1726
12-661 CSI2PHY_SCP Register Summary 1726
12-662 CSI2PHY_CFG0 1727
12-663 Register Call Summary for Register CSI2PHY_CFG0 1727
12-664 CSI2PHY_CFG1 1728
12-665 Register Call Summary for Register CSI2PHY_CFG1 1729
13-1 Clock Descriptions 1735
13-2 Instance Summary 1738
14-1 IVA2.2 Internal Clock 1743
14-2 IVA2.2 Subsystem EDMA Request Mappings 1747
14-3 IVA2.2 Interrupt Mappings 1749
14-4 IVA2.2 EDMA Hardware Parameters 1770
14-5 EDMA Memory Mapping for the Video Accelerator/Sequencer 1771
14-6 Sequencer External Interrupt 1776
14-7 Sequencer Out Interrupt Interface 1776
14-8 iLF/iME Common Commands 1779
14-9 iLF/iME Execution States 1779
14-10 iLF/iME Common Instruction Set List 1780
14-11 iLF/iME PROGRAMBUFFERLINENMSB Register Detail 1781
14-12 iLF/iME PROGRAMBUFFERLINENLSB Register Detail 1781
14-13 iLF LoadPStack() MSB Format 1782
14-14 iLF LoadPStack() LSB Format 1782

14-15 iME LoadPStack() MSB Format 1782
14-16 iME LoadPStack() LSB Format 1783
14-17 iLF/iME LoadInstBuff() MSB Format 1783
14-18 iLF/iME LoadInstBuff() LSB Format 1783
14-19 iLF/iME GenerateIT() MSB Format 1784
14-20 iLF/iME GenerateIT() LSB Format 1784
14-21 iLF/iME EndPgm() MSB Format 1784
14-22 iLF/iME EndPgm() LSB Format 1784
14-23 iLF Instruction Set List 1786
14-24 iLF ParseEdge() MSB Format 1788
14-25 iLF ParseEdge() LSB Format 1788
14-26 iLF FilterEdge() MSB Format 1789
14-27 iLF FilterEdge() LSB Format 1789
14-28 iLF LoadEFPT() MSB Format 1790
14-29 iLF LoadEFPT() LSB Format 1790
14-30 iLF Stack Configure Data Format 1792
14-31 iLF CMR Code 1792
14-32 iLF Parse-Edge Parameter Stack Data 1793
14-33 iLF Loop Filter Parameter Stack for H264 1793
14-34 iLF EFPT Table for H264 1794
14-35 iLF Loop Filter Parameter Stack for H263 1794
14-36 iLF Loop-Filter Parameter Stack for WMV9 1795
14-37 iLF Loop-Filter Parameter Stack for REAL9 1795
14-38 iLF EFPT Table for REAL9 1795
14-39 iLF FilterEdge() Load and Store Parameter 1796
14-40 iME Instruction Set 1799
14-41 iME Specific Command 1800
14-42 iME ErrorCalc() MSB Format 1802
14-43 iME ErrorCalc() LSB Format 1802
14-44 iME SaveErrs() MSB Format 1802
14-45 iME SaveErrs() LSB Format 1802
14-46 iME SaveStatus() MSB Format 1803
14-47 iME SaveStatus() LSB Format 1803
14-48 iME RestoreErrs() MSB Format 1803
14-49 iME RestoreErrs() LSB Format 1803
14-50 iME MCompare() MSB Format 1804
14-51 iME MCompare() LSB Format 1804
14-52 iME MCompare2() MSB Format 1805
14-53 iME MCompare2() LSB Format 1805
14-54 iME MCompare4() MSB Format 1805
14-55 iME MCompare4() LSB Format 1806
14-56 iME ClearStatus() MSB Format 1806
14-57 iME ClearStatus() LSB Format 1806
14-58 iME LoadRefBlk() MSB Format 1806
14-59 iME LoadRefBlk() LSB Format 1807
14-60 iME LoadRefBlk_Ind() MSB Format 1807
14-61 iME LoadRefBlk_Ind() LSB Format 1807
14-62 iME SaveBestMatch() MSB Format 1807
14-63 iME SaveBestMatch() LSB Format 1808
14-64 iME RestoreBestMatch() MSB Format 1808
14-65 iME RestoreBestMatch() LSB Format 1808

14-66 iME ExitOnMinReached() MSB Format 1808
14-67 iME ExitOnMinReached() LSB Format 1809
14-68 iME WaitOnSignal() MSB Format 1809
14-69 iME WaitOnSignal() LSB Format 1809
14-70 iME Memcopy() MSB Format 1809
14-71 iME Memcopy() LSB Format 1810
14-72 iME LoadMVCT() LSB Format 1810
14-73 iME Interpolate() MSB Format 1810
14-74 iME Interpolate() LSB Format 1810
14-75 iME Motion Vector Parameter Format 1811
14-76 iME Motion Vector Type 1811
14-77 iME Interpolate Coefficient Register Bank 1812
14-78 iME Interpolate Parameter Stack for MPEG4/H263 1812
14-79 iME Interpolate Parameter Stack for H264 1813
14-80 iME Interpolate Parameter Stack for WMV9 1813
14-81 iME Motion Vector Offset Index 1815
14-82 Error Table Format 1815
14-83 Video Accelerator/Sequencer Memory Mapping 1816
14-84 LSYS Input Interrupts 1825
14-85 IVA2.2 DSP Megamodule Cache Controller Features 1826
14-86 Boot Loader Configuration 1829
14-87 PDCCMD Programmed Value in IDLE Boot Mode 1830
14-88 Header Format Used in Defautl Config Cache Mode 1830
14-89 Header Format Used in User Defined Bootstrap Mode 1831
14-90 Cache Size Specified by L1PMODE 1834
14-91 Cache Size Specified by L1DMODE 1835
14-92 Cache Size Specified by L2MODE 1835
14-93 Switching Cache Modes 1835
14-94 Default Cache Configuration 1836
14-95 Cache Mode Configuration 1836
14-96 IVA2.2 Megamodule Memory Protection Page Registers 1870
14-97 Request-Type Access Controls 1871
14-98 Instance Summary 1884
14-99 IC Register Summary 1885
14-100 EVTFLAGi 1885
14-101 Register Call Summary for Register EVTFLAGi 1885
14-102 EVTSETi 1886
14-103 Register Call Summary for Register EVTSETi 1886
14-104 EVTCLRi 1886
14-105 Register Call Summary for Register EVTCLRi 1886
14-106 EVTMASKi 1887
14-107 Register Call Summary for Register EVTMASKi 1887
14-108 MEVTFLAGi 1887
14-109 Register Call Summary for Register MEVTFLAGi 1887
14-110 EXPMASKi 1888
14-111 Register Call Summary for Register EXPMASKi 1888
14-112 MEXPFLAGi 1888
14-113 Register Call Summary for Register MEXPFLAGi 1888
14-114 INTMUXj 1888
14-115 Register Call Summary for Register INTMUXj 1889
14-116 INTXSTAT 1889

14-117 Register Call Summary for Register INTXSTAT 1890
14-118 INTXCLR 1890
14-119 Register Call Summary for Register INTXCLR 1890
14-120 INTDMASK 1890
14-121 Register Call Summary for Register INTDMASK 1891
14-122 EVTASRT 1891
14-123 Register Call Summary for Register EVTASRT 1892
14-124 SYS Register Summary 1893
14-125 PDCCMD 1893
14-126 Register Call Summary for Register PDCCMD 1894
14-127 REVID 1895
14-128 Register Call Summary for Register REVID 1895
14-129 IDMA Register Summary 1896
14-130 IDMA0_STAT 1896
14-131 Register Call Summary for Register IDMA0_STAT 1897
14-132 IDMA0_MASK 1897
14-133 Register Call Summary for Register IDMA0_MASK 1898
14-134 IDMA0_SOURCE 1899
14-135 Register Call Summary for Register IDMA0_SOURCE 1899
14-136 IDMA0_DEST 1899
14-137 Register Call Summary for Register IDMA0_DEST 1899
14-138 IDMA0_COUNT 1900
14-139 Register Call Summary for Register IDMA0_COUNT 1900
14-140 IDMA1_STAT 1900
14-141 Register Call Summary for Register IDMA1_STAT 1900
14-142 IDMA1_SOURCE 1901
14-143 Register Call Summary for Register IDMA1_SOURCE 1901
14-144 IDMA1_DEST 1901
14-145 Register Call Summary for Register IDMA1_DEST 1901
14-146 IDMA1_COUNT 1902
14-147 Register Call Summary for Register IDMA1_COUNT 1902
14-148 CPUARBE 1902
14-149 Register Call Summary for Register CPUARBE 1903
14-150 IDMAARBE 1903
14-151 Register Call Summary for Register IDMAARBE 1903
14-152 SDMAARBE 1904
14-153 Register Call Summary for Register SDMAARBE 1904
14-154 MDMAARBE 1904
14-155 Register Call Summary for Register MDMAARBE 1905
14-156 ICFGMPFAR 1905
14-157 Register Call Summary for Register ICFGMPFAR 1905
14-158 ICFGMPFSR 1905
14-159 Register Call Summary for Register ICFGMPFSR 1906
14-160 ICFGMPFCR 1906
14-161 Register Call Summary for Register ICFGMPFCR 1906
14-162 IBUSERR 1906
14-163 Register Call Summary for Register IBUSERR 1907
14-164 IBUSERRCLR 1907
14-165 Register Call Summary for Register IBUSERRCLR 1907
14-166 XMC Register Summary 1908
14-167 L2CFG 1909

14-168 Register Call Summary for Register L2CFG 1909
14-169 L1PCFG 1910
14-170 Register Call Summary for Register L1PCFG 1910
14-171 L1PCC 1910
14-172 Register Call Summary for Register L1PCC 1911
14-173 L1DCFG 1911
14-174 Register Call Summary for Register L1DCFG 1911
14-175 L1DCC 1911
14-176 Register Call Summary for Register L1DCC 1912
14-177 CPUARBU 1912
14-178 Register Call Summary for Register CPUARBU 1912
14-179 IDMAARBU 1913
14-180 Register Call Summary for Register IDMAARBU 1913
14-181 SDMAARBU 1913
14-182 Register Call Summary for Register SDMAARBU 1913
14-183 UCARBU 1914
14-184 Register Call Summary for Register UCARBU 1914
14-185 CPUARBD 1914
14-186 Register Call Summary for Register CPUARBD 1915
14-187 IDMAARBD 1915
14-188 Register Call Summary for Register IDMAARBD 1915
14-189 SDMAARBD 1916
14-190 Register Call Summary for Register SDMAARBD 1916
14-191 UCARBD 1916
14-192 Register Call Summary for Register UCARBD 1916
14-193 L2WBAR 1917
14-194 Register Call Summary for Register L2WBAR 1917
14-195 L2WWC 1917
14-196 Register Call Summary for Register L2WWC 1917
14-197 L2WIBAR 1918
14-198 Register Call Summary for Register L2WIBAR 1918
14-199 L2WIWC 1918
14-200 Register Call Summary for Register L2WIWC 1918
14-201 L2IBAR 1918
14-202 Register Call Summary for Register L2IBAR 1919
14-203 L2IWC 1919
14-204 Register Call Summary for Register L2IWC 1919
14-205 L1PIBAR 1919
14-206 Register Call Summary for Register L1PIBAR 1919
14-207 L1PIWC 1920
14-208 Register Call Summary for Register L1PIWC 1920
14-209 L1DWIBAR 1920
14-210 Register Call Summary for Register L1DWIBAR 1920
14-211 L1DWIWC 1920
14-212 Register Call Summary for Register L1DWIWC 1921
14-213 L1DWBAR 1921
14-214 Register Call Summary for Register L1DWBAR 1921
14-215 L1DWWC 1921
14-216 Register Call Summary for Register L1DWWC 1921
14-217 L1DIBAR 1922
14-218 Register Call Summary for Register L1DIBAR 1922

14-219 L1DIWC 1922
14-220 Register Call Summary for Register L1DIWC 1922
14-221 L2WB 1922
14-222 Register Call Summary for Register L2WB 1923
14-223 L2WBINV 1923
14-224 Register Call Summary for Register L2WBINV 1923
14-225 L2INV 1924
14-226 Register Call Summary for Register L2INV 1924
14-227 L1PINV 1924
14-228 Register Call Summary for Register L1PINV 1924
14-229 L1DWB 1925
14-230 Register Call Summary for Register L1DWB 1925
14-231 L1DWBINV 1925
14-232 Register Call Summary for Register L1DWBINV 1925
14-233 L1DINV 1926
14-234 Register Call Summary for Register L1DINV 1926
14-235 MARi 1926
14-236 Register Call Summary for Register MARi 1926
14-237 L2MPFAR 1927
14-238 Register Call Summary for Register L2MPFAR 1927
14-239 L2MPFSR 1927
14-240 Register Call Summary for Register L2MPFSR 1927
14-241 L2MPFCR 1928
14-242 Register Call Summary for Register L2MPFCR 1928
14-243 L2MPPAj 1928
14-244 Register Call Summary for Register L2MPPAj 1929
14-245 L1PMPFAR 1929
14-246 Register Call Summary for Register L1PMPFAR 1929
14-247 L1PMPFSR 1929
14-248 Register Call Summary for Register L1PMPFSR 1930
14-249 L1PMPFCR 1930
14-250 Register Call Summary for Register L1PMPFCR 1930
14-251 L1PMPPAk 1931
14-252 Register Call Summary for Register L1PMPPAk 1931
14-253 L1DMPFAR 1932
14-254 Register Call Summary for Register L1DMPFAR 1932
14-255 L1DMPFSR 1932
14-256 Register Call Summary for Register L1DMPFSR 1932
14-257 L1DMPFCR 1933
14-258 Register Call Summary for Register L1DMPFCR 1933
14-259 L1DMPPAk 1933
14-260 Register Call Summary for Register L1DMPPAk 1934
14-261 TPCC Register Summary 1934
14-262 TPCC_PID 1937
14-263 Register Call Summary for Register TPCC_PID 1937
14-264 TPCC_CCCFG 1937
14-265 Register Call Summary for Register TPCC_CCCFG 1938
14-266 TPCC_DCHMAPi 1939
14-267 Register Call Summary for Register TPCC_DCHMAPi 1939
14-268 TPCC_QCHMAPj 1939
14-269 Register Call Summary for Register TPCC_QCHMAPj 1939

14-270 TPCC_DMAQNUM0 1940
14-271 Register Call Summary for Register TPCC_DMAQNUM0 1941
14-272 TPCC_DMAQNUM1 1941
14-273 Register Call Summary for Register TPCC_DMAQNUM1 1942
14-274 TPCC_DMAQNUM2 1942
14-275 Register Call Summary for Register TPCC_DMAQNUM2 1943
14-276 TPCC_DMAQNUM3 1944
14-277 Register Call Summary for Register TPCC_DMAQNUM3 1945
14-278 TPCC_DMAQNUM4 1945
14-279 Register Call Summary for Register TPCC_DMAQNUM4 1946
14-280 TPCC_DMAQNUM5 1946
14-281 Register Call Summary for Register TPCC_DMAQNUM5 1947
14-282 TPCC_DMAQNUM6 1948
14-283 Register Call Summary for Register TPCC_DMAQNUM6 1949
14-284 TPCC_DMAQNUM7 1949
14-285 Register Call Summary for Register TPCC_DMAQNUM7 1950
14-286 TPCC_QDMAQNUM 1950
14-287 Register Call Summary for Register TPCC_QDMAQNUM 1951
14-288 TPCC_QUETCMAP 1952
14-289 Register Call Summary for Register TPCC_QUETCMAP 1952
14-290 TPCC_QUEPRI 1952
14-291 Register Call Summary for Register TPCC_QUEPRI 1953
14-292 TPCC_EMR 1953
14-293 Register Call Summary for Register TPCC_EMR 1954
14-294 TPCC_EMRH 1954
14-295 Register Call Summary for Register TPCC_EMRH 1955
14-296 TPCC_EMCR 1955
14-297 Register Call Summary for Register TPCC_EMCR 1956
14-298 TPCC_EMCRH 1956
14-299 Register Call Summary for Register TPCC_EMCRH 1957
14-300 TPCC_QEMR 1957
14-301 Register Call Summary for Register TPCC_QEMR 1958
14-302 TPCC_QEMCR 1958
14-303 Register Call Summary for Register TPCC_QEMCR 1958
14-304 TPCC_CCERR 1958
14-305 Register Call Summary for Register TPCC_CCERR 1959
14-306 TPCC_CCERRCLR 1959
14-307 Register Call Summary for Register TPCC_CCERRCLR 1960
14-308 TPCC_EEVAL 1960
14-309 Register Call Summary for Register TPCC_EEVAL 1960
14-310 TPCC_DRAEj 1960
14-311 Register Call Summary for Register TPCC_DRAEj 1961
14-312 TPCC_DRAEHj 1961
14-313 Register Call Summary for Register TPCC_DRAEHj 1962
14-314 TPCC_QRAEj 1962
14-315 Register Call Summary for Register TPCC_QRAEj 1963
14-316 TPCC_Q0Ek 1963
14-317 Register Call Summary for Register TPCC_Q0Ek 1963
14-318 TPCC_Q1Ek 1964
14-319 Register Call Summary for Register TPCC_Q1Ek 1964
14-320 TPCC_QSTATI 1964

14-321 Register Call Summary for Register TPCC_QSTATl 1965
14-322 TPCC_QWMTHRA 1965
14-323 Register Call Summary for Register TPCC_QWMTHRA 1965
14-324 TPCC_QWMTHRB 1966
14-325 Register Call Summary for Register TPCC_QWMTHRB 1966
14-326 TPCC_CCSTAT 1966
14-327 Register Call Summary for Register TPCC_CCSTAT 1967
14-328 TPCC_MPFAR 1967
14-329 Register Call Summary for Register TPCC_MPFAR 1968
14-330 TPCC_MPFSR 1968
14-331 Register Call Summary for Register TPCC_MPFSR 1968
14-332 TPCC_MPFCR 1969
14-333 Register Call Summary for Register TPCC_MPFCR 1969
14-334 TPCC_MPPAG 1969
14-335 Register Call Summary for Register TPCC_MPPAG 1970
14-336 TPCC_MPPAj 1971
14-337 Register Call Summary for Register TPCC_MPPAj 1972
14-338 TPCC_ER 1972
14-339 Register Call Summary for Register TPCC_ER 1973
14-340 TPCC_ECR 1973
14-341 Register Call Summary for Register TPCC_ECR 1974
14-342 TPCC_ECRH 1974
14-343 Register Call Summary for Register TPCC_ECRH 1975
14-344 TPCC_ESR 1975
14-345 Register Call Summary for Register TPCC_ESR 1976
14-346 TPCC_ESRH 1976
14-347 Register Call Summary for Register TPCC_ESRH 1977
14-348 TPCC_CER 1977
14-349 Register Call Summary for Register TPCC_CER 1978
14-350 TPCC_CERH 1978
14-351 Register Call Summary for Register TPCC_CERH 1979
14-352 TPCC_EER 1979
14-353 Register Call Summary for Register TPCC_EER 1980
14-354 TPCC_EECR 1980
14-355 Register Call Summary for Register TPCC_EECR 1981
14-356 TPCC_EESR 1981
14-357 Register Call Summary for Register TPCC_EESR 1982
14-358 TPCC_SER 1982
14-359 Register Call Summary for Register TPCC_SER 1983
14-360 TPCC_SERH 1983
14-361 Register Call Summary for Register TPCC_SERH 1984
14-362 TPCC_SECR 1984
14-363 Register Call Summary for Register TPCC_SECR 1985
14-364 TPCC_SECRH 1985
14-365 Register Call Summary for Register TPCC_SECRH 1986
14-366 TPCC_IER 1986
14-367 Register Call Summary for Register TPCC_IER 1987
14-368 TPCC_IERH 1987
14-369 Register Call Summary for Register TPCC_IERH 1988
14-370 TPCC_IECR 1988
14-371 Register Call Summary for Register TPCC_IECR 1989

14-372 TPCC_IECRH 1989
14-373 Register Call Summary for Register TPCC_IECRH 1990
14-374 TPCC_IESR 1990
14-375 Register Call Summary for Register TPCC_IESR 1991
14-376 TPCC_IESRH 1991
14-377 Register Call Summary for Register TPCC_IESRH 1992
14-378 TPCC_IPR 1992
14-379 Register Call Summary for Register TPCC_IPR 1993
14-380 TPCC_IPRH 1993
14-381 Register Call Summary for Register TPCC_IPRH 1994
14-382 TPCC_ICR 1994
14-383 Register Call Summary for Register TPCC_ICR 1995
14-384 TPCC_ICRH 1995
14-385 Register Call Summary for Register TPCC_ICRH 1996
14-386 TPCC_IEVAL 1996
14-387 Register Call Summary for Register TPCC_IEVAL 1996
14-388 TPCC_QER 1996
14-389 Register Call Summary for Register TPCC_QER 1997
14-390 TPCC_QEER 1997
14-391 Register Call Summary for Register TPCC_QEER 1997
14-392 TPCC_QEECR 1998
14-393 Register Call Summary for Register TPCC_QEECR 1998
14-394 TPCC_QEESR 1998
14-395 Register Call Summary for Register TPCC_QEESR 1999
14-396 TPCC_QSER 1999
14-397 Register Call Summary for Register TPCC_QSER 1999
14-398 TPCC_QSECR 1999
14-399 Register Call Summary for Register TPCC_QSECR 2000
14-400 TPCC_ER_Rn 2000
14-401 Register Call Summary for Register TPCC_ER_Rn 2000
14-402 TPCC_ECR_Rn 2001
14-403 Register Call Summary for Register TPCC_ECR_Rn 2001
14-404 TPCC_ECRH_Rn 2002
14-405 Register Call Summary for Register TPCC_ECRH_Rn 2002
14-406 TPCC_ESR_Rn 2003
14-407 Register Call Summary for Register TPCC_ESR_Rn 2003
14-408 TPCC_ESRH_Rn 2004
14-409 Register Call Summary for Register TPCC_ESRH_Rn 2004
14-410 TPCC_CER_Rn 2005
14-411 Register Call Summary for Register TPCC_CER_Rn 2006
14-412 TPCC_CERH_Rn 2006
14-413 Register Call Summary for Register TPCC_CERH_Rn 2007
14-414 TPCC_EER_Rn 2007
14-415 Register Call Summary for Register TPCC_EER_Rn 2008
14-416 TPCC_EECR_Rn 2008
14-417 Register Call Summary for Register TPCC_EECR_Rn 2008
14-418 TPCC_EESR_Rn 2009
14-419 Register Call Summary for Register TPCC_EESR_Rn 2009
14-420 TPCC_SER_Rn 2010
14-421 Register Call Summary for Register TPCC_SER_Rn 2010
14-422 TPCC_SERH_Rn 2011

14-423 Register Call Summary for Register TPCC_SERH_Rn 2011
14-424 TPCC_SECR_Rn 2012
14-425 Register Call Summary for Register TPCC_SECR_Rn 2012
14-426 TPCC_SECRH_Rn 2013
14-427 Register Call Summary for Register TPCC_SECRH_Rn 2013
14-428 TPCC_IER_Rn 2014
14-429 Register Call Summary for Register TPCC_IER_Rn 2014
14-430 TPCC_IERH_Rn 2015
14-431 Register Call Summary for Register TPCC_IERH_Rn 2015
14-432 TPCC_IECR_Rn 2016
14-433 Register Call Summary for Register TPCC_IECR_Rn 2016
14-434 TPCC_IECRH_Rn 2017
14-435 Register Call Summary for Register TPCC_IECRH_Rn 2017
14-436 TPCC_IESR_Rn 2018
14-437 Register Call Summary for Register TPCC_IESR_Rn 2018
14-438 TPCC_IESRH_Rn 2019
14-439 Register Call Summary for Register TPCC_IESRH_Rn 2019
14-440 TPCC_IPR_Rn 2020
14-441 Register Call Summary for Register TPCC_IPR_Rn 2020
14-442 TPCC_IPRH_Rn 2021
14-443 Register Call Summary for Register TPCC_IPRH_Rn 2021
14-444 TPCC_ICR_Rn 2022
14-445 Register Call Summary for Register TPCC_ICR_Rn 2022
14-446 TPCC_ICRH_Rn 2023
14-447 Register Call Summary for Register TPCC_ICRH_Rn 2023
14-448 TPCC_IEVAL_Rn 2024
14-449 Register Call Summary for Register TPCC_IEVAL_Rn 2024
14-450 TPCC_QER_Rn 2024
14-451 Register Call Summary for Register TPCC_QER_Rn 2025
14-452 TPCC_QEER_Rn 2025
14-453 Register Call Summary for Register TPCC_QEER_Rn 2025
14-454 TPCC_QEECR_Rn 2025
14-455 Register Call Summary for Register TPCC_QEECR_Rn 2026
14-456 TPCC_QEESR_Rn 2026
14-457 Register Call Summary for Register TPCC_QEESR_Rn 2026
14-458 TPCC_QSER_Rn 2026
14-459 Register Call Summary for Register TPCC_QSER_Rn 2027
14-460 TPCC_QSECR_Rn 2027
14-461 Register Call Summary for Register TPCC_QSECR_Rn 2027
14-462 TPCC_OPTm 2028
14-463 Register Call Summary for Register TPCC_OPTm 2029
14-464 TPCC_SRCm 2029
14-465 Register Call Summary for Register TPCC_SRCm 2029
14-466 TPCC_ABCNTm 2030
14-467 Register Call Summary for Register TPCC_ABCNTm 2030
14-468 TPCC_DSTm 2030
14-469 Register Call Summary for Register TPCC_DSTm 2030
14-470 TPCC_BIDXm 2031
14-471 Register Call Summary for Register TPCC_BIDXm 2031
14-472 TPCC_LNKm 2031
14-473 Register Call Summary for Register TPCC_LNKm 2032

14-474 TPCC_CIDXm 2032
14-475 Register Call Summary for Register TPCC_CIDXm 2033
14-476 TPCC_CCNTm 2033
14-477 Register Call Summary for Register TPCC_CCNTm 2033
14-478 TPTC0 and TPTC1 Register Summary 2033
14-479 TPTCj_PID 2035
14-480 Register Call Summary for Register TPTCj_PID 2035
14-481 TPTCj_TCCFG 2035
14-482 Register Call Summary for Register TPTCj_TCCFG 2036
14-483 TPTCj_TCSTAT 2036
14-484 Register Call Summary for Register TPTCj_TCSTAT 2037
14-485 TPTCj_INTSTAT 2037
14-486 Register Call Summary for Register TPTCj_INTSTAT 2037
14-487 TPTCj_INTEN 2038
14-488 Register Call Summary for Register TPTCj_INTEN 2038
14-489 TPTCj_INTCLR 2038
14-490 Register Call Summary for Register TPTCj_INTCLR 2039
14-491 TPTCj_INTCMD 2039
14-492 Register Call Summary for Register TPTCj_INTCMD 2039
14-493 TPTCj_ERRSTAT 2039
14-494 Register Call Summary for Register TPTCj_ERRSTAT 2040
14-495 TPTCj_ERREN 2040
14-496 Register Call Summary for Register TPTCj_ERREN 2041
14-497 TPTCj_ERRCLR 2041
14-498 Register Call Summary for Register TPTCj_ERRCLR 2041
14-499 TPTCj_ERRDET 2042
14-500 Register Call Summary for Register TPTCj_ERRDET 2042
14-501 TPTCj_ERRCMD 2043
14-502 Register Call Summary for Register TPTCj_ERRCMD 2043
14-503 TPTCj_RDRATE 2043
14-504 Register Call Summary for Register TPTCj_RDRATE 2043
14-505 TPTCj_POPT 2044
14-506 Register Call Summary for Register TPTCj_POPT 2045
14-507 TPTCj_PSRC 2045
14-508 Register Call Summary for Register TPTCj_PSRC 2045
14-509 TPTCj_PCNT 2045
14-510 Register Call Summary for Register TPTCj_PCNT 2046
14-511 TPTCj_PDST 2046
14-512 Register Call Summary for Register TPTCj_PDST 2046
14-513 TPTCj_PBIDX 2046
14-514 Register Call Summary for Register TPTCj_PBIDX 2046
14-515 TPTCj_PMPPRXY 2047
14-516 Register Call Summary for Register TPTCj_PMPPRXY 2047
14-517 TPTCj_SAOPT 2047
14-518 Register Call Summary for Register TPTCj_SAOPT 2048
14-519 TPTCj_SASRC 2049
14-520 Register Call Summary for Register TPTCj_SASRC 2049
14-521 TPTCj_SACNT 2049
14-522 Register Call Summary for Register TPTCj_SACNT 2049
14-523 TPTCj_SADST 2050
14-524 Register Call Summary for Register TPTCj_SADST 2050

14-525 TPTCj_SABIDX 2050
14-526 Register Call Summary for Register TPTCj_SABIDX 2050
14-527 TPTCj_SAMPPRXY 2051
14-528 Register Call Summary for Register TPTCj_SAMPPRXY 2051
14-529 TPTCj_SACNTRLD 2051
14-530 Register Call Summary for Register TPTCj_SACNTRLD 2052
14-531 TPTCj_SASRCBREF 2052
14-532 Register Call Summary for Register TPTCj_SASRCBREF 2052
14-533 TPTCj_SADSTBREF 2052
14-534 Register Call Summary for Register TPTCj_SADSTBREF 2053
14-535 TPTCj_DFCNTRLD 2053
14-536 Register Call Summary for Register TPTCj_DFCNTRLD 2053
14-537 TPTCj_DFSRCBREF 2053
14-538 Register Call Summary for Register TPTCj_DFSRCBREF 2053
14-539 TPTCj_DFDSTBREF 2054
14-540 Register Call Summary for Register TPTCj_DFDSTBREF 2054
14-541 TPTCj_DFOPTi 2054
14-542 Register Call Summary for Register TPTCj_DFOPTi 2055
14-543 TPTCj_DFSRCi 2055
14-544 Register Call Summary for Register TPTCj_DFSRCi 2055
14-545 TPTCj_DFCNTi 2056
14-546 Register Call Summary for Register TPTCj_DFCNTi 2056
14-547 TPTCj_DFDSTi 2056
14-548 Register Call Summary for Register TPTCj_DFDSTi 2056
14-549 TPTCj_DFBIDXi 2057
14-550 Register Call Summary for Register TPTCj_DFBIDXi 2057
14-551 TPTCj_DFMPPRXYi 2057
14-552 Register Call Summary for Register TPTCj_DFMPPRXYi 2058
14-553 SYSC Register Summary 2058
14-554 SYSC_REVISION 2059
14-555 Register Call Summary for Register SYSC_REVISION 2059
14-556 SYSC_SYSCONFIG 2059
14-557 Register Call Summary for Register SYSC_SYSCONFIG 2059
14-558 SYSC_LICFG0 2060
14-559 Register Call Summary for Register SYSC_LICFG0 2060
14-560 SYSC_LICFG1 2061
14-561 Register Call Summary for Register SYSC_LICFG1 2061
14-562 SYSC_BOOTADDR 2061
14-563 Register Call Summary for Register SYSC_BOOTADDR 2061
14-564 SYSC_BOOTMOD 2062
14-565 Register Call Summary for Register SYSC_BOOTMOD 2062
14-566 WUGEN Register Summary 2063
14-567 WUGEN_REVISION 2064
14-568 Register Call Summary for Register WUGEN_REVISION 2064
14-569 WUGEN_SYSCONFIG 2064
14-570 Register Call Summary for Register WUGEN_SYSCONFIG 2064
14-571 WUGEN_MEVT0 2065
14-572 Register Call Summary for Register WUGEN_MEVT0 2065
14-573 WUGEN_MEVT1 2066
14-574 Register Call Summary for Register WUGEN_MEVT1 2066
14-575 WUGEN_MEVT2 2067

14-576 Register Call Summary for Register WUGEN_MEVT2 2067
14-577 WUGEN_MEVTCLR0 2068
14-578 Register Call Summary for Register WUGEN_MEVTCLR0 2069
14-579 WUGEN_MEVTCLR1 2069
14-580 Register Call Summary for Register WUGEN_MEVTCLR1 2070
14-581 WUGEN_MEVTCLR2 2070
14-582 Register Call Summary for Register WUGEN_MEVTCLR2 2071
14-583 WUGEN_MEVTSET0 2072
14-584 Register Call Summary for Register WUGEN_MEVTSET0 2073
14-585 WUGEN_MEVTSET1 2073
14-586 Register Call Summary for Register WUGEN_MEVTSET1 2074
14-587 WUGEN_MEVTSET2 2074
14-588 Register Call Summary for Register WUGEN_MEVTSET2 2075
14-589 WUGEN_PENDEVT0 2076
14-590 Register Call Summary for Register WUGEN_PENDEVT0 2076
14-591 WUGEN_PENDEVT1 2077
14-592 Register Call Summary for Register WUGEN_PENDEVT1 2077
14-593 WUGEN_PENDEVT2 2078
14-594 Register Call Summary for Register WUGEN_PENDEVT2 2078
14-595 WUGEN_PENDEVTCLR0 2079
14-596 Register Call Summary for Register WUGEN_PENDEVTCLR0 2080
14-597 WUGEN_PENDEVTCLR1 2080
14-598 Register Call Summary for Register WUGEN_PENDEVTCLR1 2080
14-599 WUGEN_PENDEVTCLR2 2081
14-600 Register Call Summary for Register WUGEN_PENDEVTCLR2 2081
14-601 SEQ Register Mapping Summary 2082
14-602 SEQ_REVISION 2082
14-603 Register Call Summary for Register SEQ_REVISION 2082
14-604 SEQ_SYSCONFIG 2082
14-605 Register Call Summary for Register SEQ_SYSCONFIG 2083
14-606 SEQ_IRQMASK 2083
14-607 Register Call Summary for Register SEQ_IRQMASK 2084
14-608 SEQ_IRQCLR 2084
14-609 Register Call Summary for Register SEQ_IRQCLR 2085
14-610 SEQ_IRQSET 2085
14-611 Register Call Summary for Register SEQ_IRQSET 2086
14-612 SEQ_IRQSTATE 2086
14-613 Register Call Summary for Register SEQ_IRQSTATE 2087
14-614 SEQ_SWICLR 2087
14-615 Register Call Summary for Register SEQ_SWICLR 2087
14-616 SEQ_SWISET 2088
14-617 Register Call Summary for Register SEQ_SWISET 2088
14-618 SEQ_SWISTATE 2088
14-619 Register Call Summary for Register SEQ_SWISTATE 2088
14-620 VIDEOSYSC Register Mapping Summary 2089
14-621 VIDEOSYSC_REVISION 2089
14-622 Register Call Summary for Register VIDEOSYSC_REVISION 2089
14-623 VIDEOSYSC_SYSCONFIG 2090
14-624 Register Call Summary for Register VIDEOSYSC_SYSCONFIG 2090
14-625 VIDEOSYSC_IRQMASK 2090
14-626 Register Call Summary for Register VIDEOSYSC_IRQMASK 2091

14-627 VIDEOSYSC_IRQCLR 2091
14-628 Register Call Summary for Register VIDEOSYSC_IRQCLR 2091
14-629 VIDEOSYSC_IRQSET 2092
14-630 Register Call Summary for Register VIDEOSYSC_IRQSET 2092
14-631 VIDEOSYSC_IRQSTATE 2092
14-632 Register Call Summary for Register VIDEOSYSC_IRQSTATE 2093
14-633 VIDEOSYSC_CLKCTL 2093
14-634 Register Call Summary for Register VIDEOSYSC_CLKCTL 2093
14-635 VIDEOSYSC_CLKDIV 2094
14-636 Register Call Summary for Register VIDEOSYSC_CLKDIV 2094
14-637 VIDEOSYSC_CLKST 2095
14-638 Register Call Summary for Register VIDEOSYSC_CLKST 2095
14-639 iME Register Mapping Summary 2096
14-640 iME_REVISION 2097
14-641 Register Call Summary for Register iME_REVISION 2097
14-642 iME_SYSCONFIG 2097
14-643 Register Call Summary for Register iME_SYSCONFIG 2098
14-644 iME_SYSSTATUS 2098
14-645 Register Call Summary for Register iME_SYSSTATUS 2098
14-646 iME_PROGRAMBUFFERLINENLSBi 2098
14-647 Register Call Summary for Register iME_PROGRAMBUFFERLINENLSBi 2098
14-648 iME_PROGRAMBUFFERLINENMSBi 2099
14-649 Register Call Summary for Register iME_PROGRAMBUFFERLINENMSBi 2099
14-650 iME_ERRORTABLEj 2099
14-651 Register Call Summary for Register iME_ERRORTABLEj 2099
14-652 iME_REFERENCEBLOCKk 2100
14-653 Register Call Summary for Register iME_REFERENCEBLOCKk 2100
14-654 iME_COEFFREGBANKl 2100
14-655 Register Call Summary for Register iME_COEFFREGBANKl 2100
14-656 iME_PARAMETERSTACKLj 2100
14-657 Register Call Summary for Register iME_PARAMETERSTACKLj 2101
14-658 iME_PARAMETERSTACKHj 2101
14-659 Register Call Summary for Register iME_PARAMETERSTACKHj 2101
14-660 iME_XMVCTm 2101
14-661 Register Call Summary for Register iME_XMVCTm 2101
14-662 iME_YMVCTm 2101
14-663 Register Call Summary for Register iME_YMVCTm 2102
14-664 iME_MINERRORTHRESHOLD 2102
14-665 Register Call Summary for Register iME_MINERRORTHRESHOLD 2102
14-666 iME_ABSMINREACHED 2102
14-667 Register Call Summary for Register iME_ABSMINREACHED 2102
14-668 iME_CPUSTATUSREG 2103
14-669 Register Call Summary for Register iME_CPUSTATUSREG 2103
14-670 iME_IRQLOG 2104
14-671 Register Call Summary for Register iME_IRQLOG 2104
14-672 iME_LATESTERRORS 2104
14-673 Register Call Summary for Register iME_LATESTERRORS 2104
14-674 iME_CONFIGREG 2105
14-675 Register Call Summary for Register iME_CONFIGREG 2105
14-676 iME_SL2INSTADDRESS 2105
14-677 Register Call Summary for Register iME_SL2INSTADDRESS 2106

14-678 iME_COMMANDREG 2106
14-679 Register Call Summary for Register iME_COMMANDREG 2106
14-680 iLF Register Mapping Summary 2107
14-681 iLF_REVISION 2108
14-682 Register Call Summary for Register iLF_REVISION 2108
14-683 iLF_SYSCONFIG 2108
14-684 Register Call Summary for Register iLF_SYSCONFIG 2109
14-685 iLF_SYSSTATUS 2109
14-686 Register Call Summary for Register iLF_SYSSTATUS 2109
14-687 iLF_PROGRAMBUFFERLINENLSBi 2109
14-688 Register Call Summary for Register iLF_PROGRAMBUFFERLINENLSBi 2109
14-689 iLF_PROGRAMBUFFERLINENMSBi 2110
14-690 Register Call Summary for Register iLF_PROGRAMBUFFERLINENMSBi 2110
14-691 iLF_PARAMETERSTACKUPj 2110
14-692 Register Call Summary for Register iLF_PARAMETERSTACKUPj 2110
14-693 iLF_PARAMETERSTACKLWk 2111
14-694 Register Call Summary for Register iLF_PARAMETERSTACKLWk 2111
14-695 iLF_EFPTABLEENTRYl 2111
14-696 Register Call Summary for Register iLF_EFPTABLEENTRYl 2111
14-697 iLF_INOUTBUFFERm 2111
14-698 Register Call Summary for Register iLF_INOUTBUFFERm 2112
14-699 iLF_CPUSTATUSREG 2112
14-700 Register Call Summary for Register iLF_CPUSTATUSREG 2113
14-701 iLF_IRQLOG 2113
14-702 Register Call Summary for Register iLF_IRQLOG 2113
14-703 iLF_EFPTD 2113
14-704 Register Call Summary for Register iLF_EFPTD 2114
14-705 iLF_CONFIGREG 2114
14-706 Register Call Summary for Register iLF_CONFIGREG 2114
14-707 iLF_PARSEDDATAREG0 2114
14-708 Register Call Summary for Register iLF_PARSEDDATAREG0 2115
14-709 iLF_PARSEDDATAREG1 2115
14-710 Register Call Summary for Register iLF_PARSEDDATAREG1 2115
14-711 iLF_PARSEDDATAREG2 2115
14-712 Register Call Summary for Register iLF_PARSEDDATAREG2 2116
14-713 iLF_INSTBUFFER_ADDRESS 2116
14-714 Register Call Summary for Register iLF_INSTBUFFER_ADDRESS 2116
14-715 iLF_LINESFILTERPROTOTYPES 2116
14-716 Register Call Summary for Register iLF_LINESFILTERPROTOTYPES 2117
14-717 iLF_CLIPLIMITSENTRYn 2117
14-718 Register Call Summary for Register iLF_CLIPLIMITSENTRYn 2117
14-719 iLF_COMMANDREG 2117
14-720 Register Call Summary for Register iLF_COMMANDREG 2118
14-721 IA_GEM Register Mapping Summary 2118
14-722 GEM_AGENT_STATUS 2118
14-723 Register Call Summary for Register GEM_AGENT_STATUS 2119
14-724 IA_EDMA Register Mapping Summary 2119
14-725 EDMA_AGENT_STATUS 2119
14-726 Register Call Summary for Register EDMA_AGENT_STATUS 2120
14-727 IA_SEQ Register Mapping Summary 2121
14-728 SEQ_AGENT_STATUS 2121

14-729 Register Call Summary for Register SEQ_AGENT_STATUS 2122
15-1 I/O Pad Mode Selection 2129
15-2 LCD Interface Signals (RFBI Mode) 2130
15-3 LCD Interface Signals (Bypass Mode) 2132
15-4 Number of Displayed Pixels per Pixel Clock Cycle Based on Display Type 2133
15-5 Programmable Timing Fields in RFBI Mode 2138
15-6 Programmable Fields in Bypass Mode 2139
15-7 Interface Signals Between the SDI Module and LCD Panel 2146
15-8 I/O Description for DSI Serial Interface 2147
15-9 DSI Lane Configuration 2148
15-10 Video Interface for DSI Protocol Engine 2149
15-11 Video Interface in the Context of Video Mode 2150
15-12 Video Interface in the Context of Command Mode 2155
15-13 Pixel Data Format in Video Mode 2162
15-14 Synchronization Codes 2162
15-15 Sync Short Packet Values 2163
15-16 Virtual Channel Values 2170
15-17 TV Display Interface Pins 2177
15-18 Display Subsystem Clocks 2182
15-19 Possible Digital Clock Division for the Video Encoder 2184
15-20 SDI PLL Operation Modes 2187
15-21 DSS DMA Requests Description 2190
15-22 Display Subsystem Interrupts 2192
15-23 DSI Global Interrupts 2193
15-24 DSI Complex I/O Interrupts 2194
15-25 DSI Virtual Channel Interrupts 2195
15-26 Functional Clock Frequency Requirement in RGB16 & YUV422—Active Matrix Display 2209
15-27 Functional Clock Frequency Requirement in RGB24—Active Matrix Display 2209
15-28 Alpha Blending 4-Bit Values 2215
15-29 8-Bit Interface Configuration/24-Bit Mode 2219
15-30 Maximum Width Allowed 2220
15-31 LP to HS Timing Parameters 2223
15-32 HS to LP Timing Parameters 2225
15-33 Extra NULL Packet Header 2225
15-34 Extra NULL Packet Payload 2226
15-35 DSI PLL Operation Modes When Not Locked 2248
15-36 16-Bit Interface Configuration/24-Bit Mode 2252
15-37 Read/Write Function Description 2253
15-38 Minimum Cycle Time for CSx/WE Always Asserted 2253
15-39 100/100 Color Bar Table 2254
15-40 VENC_S_CARR Register Recommended Values 2255
15-41 Closed-Caption Data Format 2256
15-42 Closed-Caption Run Clock Frequency Settings 2257
15-43 Closed-Caption Standard Timing Values 2257
15-44 Wide-Screen Signaling Run Clock Frequency Settings 2258
15-45 Shadow Registers 2269
15-46 Vertical/Horizontal Accumulator Phase 2279
15-47 Color Space Conversion Register Values 2280
15-48 90-degree DMA Rotation Example Description 2282
15-49 DMA Rotation Register Settings 2284
15-50 Video Rotation Register Settings (With RGB24 Packet Format) 2285

15-51 Register Settings for DMA Rotation With Mirroring 2286
15-52 VRFB Rotation - DMA Settings 2286
15-53 VRFB Rotation With Mirroring - DMA Settings 2288
15-54 Video Rotation Register Settings (YUV Only) 2289
15-55 Video Rotation With Mirroring Register Settings (YUV only) 2290
15-56 Programming Rules 2291
15-57 Pixel Clock Frequency Limitations - RGB16 and YUV422 Active Matrix Display 2292
15-58 Pixel Clock Frequency Limitations - RGB16 and YUV422 Passive Matrix Display - Mono4 2292
15-59 Pixel Clock Frequency Limitations - RGB16 and YUV422 Passive Matrix Display - Mono8 2292
15-60 Pixel Clock Frequency Limitations - RGB16 and YUV422 Passive Matrix Display - Color 2293
15-61 Register Access Width Limitations 2300
15-62 Virtual Channel TX FIFO Size Values 2305
15-63 Virtual Channel TX FIFO Start Address 2305
15-64 Virtual Channel RX FIFO Size Values 2306
15-65 Virtual Channel RX FIFO Start Address 2306
15-66 Recommended Programming Values 2320
15-67 RFBI Behavior 2327
15-68 RFBI Timings Configuration 2331
15-69 Video Encoder Register Programming Values 2336
15-70 PLL Divisor Example Values for TI FlatLink3G 2338
15-71 SDI Pixel Data Format 2339
15-72 Vertical FIR Coefficients Corresponding Table (3-Tap Configuration) 2350
15-73 Vertical FIR Coefficients Corresponding Table (5-Tap Configuration) 2351
15-74 Horizontal FIR Coefficients Corresponding Table (5-Tap Configuration) 2351
15-75 Vertical/Horizontal Accumulator Phase 2353
15-76 Up-Sampling Vertical Filter Coefficients (Three Taps) 2354
15-77 Up-Sampling Vertical Filter Coefficients (Five Taps) 2354
15-78 Up-Sampling Horizontal Filter Coefficients (Five Taps) 2354
15-79 Down-Sampling Vertical Filter Coefficients (Three Taps) 2355
15-80 Down-Sampling Vertical Filter Coefficients (Five Taps) 2356
15-81 Down-Sampling Horizontal Filter Coefficients (Five Taps) 2356
15-82 SN65LVDS302 Receiver Mode Transitions 2373
15-83 Registers Print for QVGA LCD panel Pads Multiplexing Configuration 2378
15-84 Registers Print for Display Subsytem Clock Management 2380
15-85 Registers Print for Display Subsystem Power Management 2380
15-86 Registers Print for Display Subsystem Software Reset 2381
15-87 VRFB Rotation Configuration 2382
15-88 VRFB Rotation Configuration for a VGA Display (UYVY format) 2382
15-89 Video Rotation Register Settings (UYVY Only) 2383
15-90 Registers Print for Video1 Channel Configuration 2383
15-91 Registers Print for Interrupts Enable 2385
15-92 Registers Print for Display Panel Configuration 2387
15-93 Registers Print for LCD Enable 2387
15-94 Ratio R 2388
15-95 Main Steps 2392
15-96 PRCM Registers 2393
15-97 Resets 2393
15-98 DSI PLL Configuration Registers 2393
15-99 DSI Control Registers 2394
15-100 DSI Complex I/O Registers 2395
15-101 DSI Timing Registers 2395

15-102 Calculate DSI_PHY Timing 2396
15-103 Drive Stop State 2397
15-104 Reset DISPC 2397
15-105 Configure DISPC Registers 2397
15-106 Configure Color Space Coefficient Registers 2398
15-107 Configure DISPC_CONTROL 2398
15-108 Configure DISPC_VID1_ATTRIBUTES 2398
15-109 Enable DISPC 2399
15-110 Main Sequence 2400
15-111 Configure DSS Clocks at the PRCM Module 2401
15-112 Configure DSI Protocol Engine, DSI PLL, and Complex I/O 2401
15-113 Reset DSI Modules 2401
15-114 Configure DSI PLL 2402
15-115 Switch to DSI PLL Clock Source 2403
15-116 DSI Control Registers 2403
15-117 DSI Complex I/O Registers 2403
15-118 DSI Timing Registers 2404
15-119 Configure DSI_PHY Timing 2405
15-120 Drive Stop State 2406
15-121 Initialization of the External MIPI LCD Controller 2406
15-122 Reset DISPC 2406
15-123 Configure DISPC Registers 2406
15-124 Configure DISPC_CONTROL 2407
15-125 Enable Command Mode and Automatic TE 2407
15-126 Send Frame Data to LCD Panel Using Automatic TE 2408
15-127 Instance Summary 2409
15-128 Display Subsystem Register Mapping Summary 2409
15-129 Display Controller Register Mapping Summary 2410
15-130 Display Controller VID1 Register Summary 2410
15-131 Display Controller VID2 Register Summary 2411
15-132 RFBI Register Mapping Summary 2412
15-133 Video Encoder Register Mapping Summary 2412
15-134 DSI Protocol Engine Register Mapping Summary 2413
15-135 DSI_PHY Register Mapping Summary 2414
15-136 DSI PLL Controller Register Mapping Summary 2415
15-137 DSS_REVISIONNUMBER 2415
15-138 Register Call Summary for Register DSS_REVISIONNUMBER 2415
15-139 DSS_SYSCONFIG 2415
15-140 Register Call Summary for Register DSS_SYSCONFIG 2416
15-141 DSS_SYSSTATUS 2416
15-142 Register Call Summary for Register DSS_SYSSTATUS 2416
15-143 DSS_IRQSTATUS 2417
15-144 Register Call Summary for Register DSS_IRQSTATUS 2417
15-145 DSS_CONTROL 2417
15-146 Register Call Summary for Register DSS_CONTROL 2418
15-147 DSS_SDI_CONTROL 2418
15-148 Register Call Summary for Register DSS_SDI_CONTROL 2419
15-149 DSS_PLL_CONTROL 2419
15-150 Register Call Summary for Register DSS_PLL_CONTROL 2420
15-151 DSS_SDI_STATUS 2421
15-152 Register Call Summary for Register DSS_SDI_STATUS 2422

15-153 DISPC_REVISION 2422
15-154 Register Call Summary for Register DISPC_REVISION 2422
15-155 DISPC_SYSCONFIG 2423
15-156 Register Call Summary for Register DISPC_SYSCONFIG 2424
15-157 DISPC_SYSSTATUS 2424
15-158 Register Call Summary for Register DISPC_SYSSTATUS 2424
15-159 DISPC_IRQSTATUS 2424
15-160 Register Call Summary for Register DISPC_IRQSTATUS 2427
15-161 DISPC_IRQENABLE 2427
15-162 Register Call Summary for Register DISPC_IRQENABLE 2428
15-163 DISPC_CONTROL 2429
15-164 Register Call Summary for Register DISPC_CONTROL 2432
15-165 DISPC_CONFIG 2432
15-166 Register Call Summary for Register DISPC_CONFIG 2434
15-167 DISPC_DEFAULT_COLOR_m 2435
15-168 Register Call Summary for Register DISPC_DEFAULT_COLOR_m 2435
15-169 DISPC_TRANS_COLOR_m 2436
15-170 Register Call Summary for Register DISPC_TRANS_COLOR_m 2436
15-171 DISPC_LINE_STATUS 2436
15-172 Register Call Summary for Register DISPC_LINE_STATUS 2436
15-173 DISPC_LINE_NUMBER 2437
15-174 Register Call Summary for Register DISPC_LINE_NUMBER 2437
15-175 DISPC_TIMING_H 2437
15-176 Register Call Summary for Register DISPC_TIMING_H 2437
15-177 DISPC_TIMING_V 2438
15-178 Register Call Summary for Register DISPC_TIMING_V 2438
15-179 DISPC_POL_FREQ 2439
15-180 Register Call Summary for Register DISPC_POL_FREQ 2439
15-181 DISPC_DIVISOR 2440
15-182 Register Call Summary for Register DISPC_DIVISOR 2440
15-183 DISPC_GLOBAL_ALPHA 2440
15-184 Register Call Summary for Register DISPC_GLOBAL_ALPHA 2441
15-185 DISPC_SIZE_DIG 2441
15-186 Register Call Summary for Register DISPC_SIZE_DIG 2441
15-187 DISPC_SIZE_LCD 2442
15-188 Register Call Summary for Register DISPC_SIZE_LCD 2442
15-189 DISPC_GFX_BAj 2442
15-190 Register Call Summary for Register DISPC_GFX_BAj 2443
15-191 DISPC_GFX_POSITION 2443
15-192 Register Call Summary for Register DISPC_GFX_POSITION 2443
15-193 DISPC_GFX_SIZE 2444
15-194 Register Call Summary for Register DISPC_GFX_SIZE 2444
15-195 DISPC_GFX_ATTRIBUTES 2444
15-196 Register Call Summary for Register DISPC_GFX_ATTRIBUTES 2446
15-197 DISPC_GFX_FIFO_THRESHOLD 2446
15-198 Register Call Summary for Register DISPC_GFX_FIFO_THRESHOLD 2446
15-199 DISPC_GFX_FIFO_SIZE_STATUS 2447
15-200 Register Call Summary for Register DISPC_GFX_FIFO_SIZE_STATUS 2447
15-201 DISPC_GFX_ROW_INC 2447
15-202 Register Call Summary for Register DISPC_GFX_ROW_INC 2447
15-203 DISPC_GFX_PIXEL_INC 2448

15-204 Register Call Summary for Register DISPC_GFX_PIXEL_INC 2448
15-205 DISPC_GFX_WINDOW_SKIP 2448
15-206 Register Call Summary for Register DISPC_GFX_WINDOW_SKIP 2448
15-207 DISPC_GFX_TABLE_BA 2449
15-208 Register Call Summary for Register DISPC_GFX_TABLE_BA 2449
15-209 DISPC_VIDn_BAj 2449
15-210 Register Call Summary for Register DISPC_VIDn_BAj 2449
15-211 DISPC_VIDn_POSITION 2450
15-212 Register Call Summary for Register DISPC_VIDn_POSITION 2450
15-213 DISPC_VIDn_SIZE 2450
15-214 Register Call Summary for Register DISPC_VIDn_SIZE 2451
15-215 DISPC_VIDn_ATTRIBUTES 2451
15-216 Register Call Summary for Register DISPC_VIDn_ATTRIBUTES 2453
15-217 DISPC_VIDn_FIFO_THRESHOLD 2454
15-218 Register Call Summary for Register DISPC_VIDn_FIFO_THRESHOLD 2454
15-219 DISPC_VIDn_FIFO_SIZE_STATUS 2454
15-220 Register Call Summary for Register DISPC_VIDn_FIFO_SIZE_STATUS 2454
15-221 DISPC_VIDn_ROW_INC 2455
15-222 Register Call Summary for Register DISPC_VIDn_ROW_INC 2455
15-223 DISPC_VIDn_PIXEL_INC 2455
15-224 Register Call Summary for Register DISPC_VIDn_PIXEL_INC 2455
15-225 DISPC_VIDn_FIR 2456
15-226 Register Call Summary for Register DISPC_VIDn_FIR 2456
15-227 DISPC_VIDn_PICTURE_SIZE 2456
15-228 Register Call Summary for Register DISPC_VIDn_PICTURE_SIZE 2457
15-229 DISPC_VIDn_ACCUI 2457
15-230 Register Call Summary for Register DISPC_VIDn_ACCUI 2457
15-231 DISPC_VIDn_FIR_COEF_Hi 2458
15-232 Register Call Summary for Register DISPC_VIDn_FIR_COEF_Hi 2458
15-233 DISPC_VIDn_FIR_COEF_HVi 2458
15-234 Register Call Summary for Register DISPC_VIDn_FIR_COEF_HVi 2458
15-235 DISPC_VIDn_CONV_COEF0 2459
15-236 Register Call Summary for Register DISPC_VIDn_CONV_COEF0 2459
15-237 DISPC_VIDn_CONV_COEF1 2459
15-238 Register Call Summary for Register DISPC_VIDn_CONV_COEF1 2460
15-239 DISPC_VIDn_CONV_COEF2 2460
15-240 Register Call Summary for Register DISPC_VIDn_CONV_COEF2 2460
15-241 DISPC_VIDn_CONV_COEF3 2460
15-242 Register Call Summary for Register DISPC_VIDn_CONV_COEF3 2461
15-243 DISPC_VIDn_CONV_COEF4 2461
15-244 Register Call Summary for Register DISPC_VIDn_CONV_COEF4 2461
15-245 DISPC_DATA_CYCLEk 2461
15-246 Register Call Summary for Register DISPC_DATA_CYCLEk 2462
15-247 DISPC_VIDn_FIR_COEF_Vi 2462
15-248 Register Call Summary for Register DISPC_VIDn_FIR_COEF_Vi 2462
15-249 DISPC_CPR_COEF_R 2463
15-250 Register Call Summary for Register DISPC_CPR_COEF_R 2463
15-251 DISPC_CPR_COEF_G 2463
15-252 Register Call Summary for Register DISPC_CPR_COEF_G 2464
15-253 DISPC_CPR_COEF_B 2464
15-254 Register Call Summary for Register DISPC_CPR_COEF_B 2464

15-255 DISPC_GFX_PRELOAD 2464
15-256 Register Call Summary for Register DISPC_GFX_PRELOAD 2465
15-257 DISPC_VIDn_PRELOAD 2465
15-258 Register Call Summary for Register DISPC_VIDn_PRELOAD 2465
15-259 RFBI_REVISION 2465
15-260 Register Call Summary for Register RFBI_REVISION 2465
15-261 RFBI_SYSCONFIG 2466
15-262 Register Call Summary for Register RFBI_SYSCONFIG 2466
15-263 RFBI_SYSSTATUS 2467
15-264 Register Call Summary for Register RFBI_SYSSTATUS 2467
15-265 RFBI_CONTROL 2467
15-266 Register Call Summary for Register RFBI_CONTROL 2468
15-267 RFBI_PIXEL_CNT 2469
15-268 Register Call Summary for Register RFBI_PIXEL_CNT 2469
15-269 RFBI_LINE_NUMBER 2469
15-270 Register Call Summary for Register RFBI_LINE_NUMBER 2469
15-271 RFBI_CMD 2470
15-272 Register Call Summary for Register RFBI_CMD 2470
15-273 RFBI_PARAM 2470
15-274 Register Call Summary for Register RFBI_PARAM 2470
15-275 RFBI_DATA 2471
15-276 Register Call Summary for Register RFBI_DATA 2471
15-277 RFBI_READ 2471
15-278 Register Call Summary for Register RFBI_READ 2472
15-279 RFBI_STATUS 2472
15-280 Register Call Summary for Register RFBI_STATUS 2472
15-281 RFBI_CONFIGi 2472
15-282 Register Call Summary for Register RFBI_CONFIGi 2473
15-283 RFBI_ONOFF_TIMEi 2474
15-284 Register Call Summary for Register RFBI_ONOFF_TIMEi 2474
15-285 RFBI_CYCLE_TIMEi 2475
15-286 Register Call Summary for Register RFBI_CYCLE_TIMEi 2475
15-287 RFBI_DATA_CYCLE1_i 2475
15-288 Register Call Summary for Register RFBI_DATA_CYCLE1_i 2476
15-289 RFBI_DATA_CYCLE2_i 2476
15-290 Register Call Summary for Register RFBI_DATA_CYCLE2_i 2477
15-291 RFBI_DATA_CYCLE3_i 2477
15-292 Register Call Summary for Register RFBI_DATA_CYCLE3_i 2478
15-293 RFBI_VSYNC_WIDTH 2478
15-294 Register Call Summary for Register RFBI_VSYNC_WIDTH 2478
15-295 RFBI_HSYNC_WIDTH 2478
15-296 Register Call Summary for Register RFBI_HSYNC_WIDTH 2479
15-297 VENC_REV_ID 2479
15-298 Register Call Summary for Register VENC_REV_ID 2479
15-299 VENC_STATUS 2479
15-300 Register Call Summary for Register VENC_STATUS 2480
15-301 VENC_F_CONTROL 2480
15-302 Register Call Summary for Register VENC_F_CONTROL 2481
15-303 VENC_VIDOUT_CTRL 2481
15-304 Register Call Summary for Register VENC_VIDOUT_CTRL 2481
15-305 VENC_SYNC_CTRL 2482

15-306 Register Call Summary for Register VENC_SYNC_CTRL 2483
15-307 VENC_LLEN 2483
15-308 Register Call Summary for Register VENC_LLEN 2483
15-309 VENC_FLENS 2483
15-310 Register Call Summary for Register VENC_FLENS 2484
15-311 VENC_HFLTR_CTRL 2484
15-312 Register Call Summary for Register VENC_HFLTR_CTRL 2484
15-313 VENC_CC_CARR_WSS_CARR 2484
15-314 Register Call Summary for Register VENC_CC_CARR_WSS_CARR 2485
15-315 VENC_C_PHASE 2485
15-316 Register Call Summary for Register VENC_C_PHASE 2485
15-317 VENC_GAIN_U 2485
15-318 Register Call Summary for Register VENC_GAIN_U 2486
15-319 VENC_GAIN_V 2486
15-320 Register Call Summary for Register VENC_GAIN_V 2486
15-321 VENC_GAIN_Y 2486
15-322 Register Call Summary for Register VENC_GAIN_Y 2487
15-323 VENC_BLACK_LEVEL 2487
15-324 Register Call Summary for Register VENC_BLACK_LEVEL 2487
15-325 VENC_BLANK_LEVEL 2487
15-326 Register Call Summary for Register VENC_BLANK_LEVEL 2488
15-327 VENC_X_COLOR 2488
15-328 Register Call Summary for Register VENC_X_COLOR 2488
15-329 VENC_M_CONTROL 2489
15-330 Register Call Summary for Register VENC_M_CONTROL 2489
15-331 VENC_BSTAMP_WSS_DATA 2490
15-332 Register Call Summary for Register VENC_BSTAMP_WSS_DATA 2490
15-333 VENC_S_CARR 2490
15-334 Register Call Summary for Register VENC_S_CARR 2491
15-335 VENC_LINE21 2491
15-336 Register Call Summary for Register VENC_LINE21 2491
15-337 VENC_LN_SEL 2491
15-338 Register Call Summary for Register VENC_LN_SEL 2492
15-339 VENC_L21_WC_CTL 2492
15-340 Register Call Summary for Register VENC_L21_WC_CTL 2493
15-341 VENC_HTRIGGER_VTRIGGER 2493
15-342 Register Call Summary for Register VENC_HTRIGGER_VTRIGGER 2493
15-343 VENC_SAVID_EAVID 2493
15-344 Register Call Summary for Register VENC_SAVID_EAVID 2494
15-345 VENC_FLEN_FAL 2494
15-346 Register Call Summary for Register VENC_FLEN_FAL 2494
15-347 VENC_LAL_PHASE_RESET 2494
15-348 Register Call Summary for Register VENC_LAL_PHASE_RESET 2495
15-349 VENC_HS_INT_START_STOP_X 2495
15-350 Register Call Summary for Register VENC_HS_INT_START_STOP_X 2495
15-351 VENC_HS_EXT_START_STOP_X 2495
15-352 Register Call Summary for Register VENC_HS_EXT_START_STOP_X 2496
15-353 VENC_VS_INT_START_X 2496
15-354 Register Call Summary for Register VENC_VS_INT_START_X 2496
15-355 VENC_VS_INT_STOP_X_VS_INT_START_Y 2496
15-356 Register Call Summary for Register VENC_VS_INT_STOP_X_VS_INT_START_Y 2497

15-357 VENC_VS_INT_STOP_Y_VS_EXT_START_X 2497
15-358 Register Call Summary for Register VENC_VS_INT_STOP_Y_VS_EXT_START_X 2497
15-359 VENC_VS_EXT_STOP_X_VS_EXT_START_Y 2497
15-360 Register Call Summary for Register VENC_VS_EXT_STOP_X_VS_EXT_START_Y 2498
15-361 VENC_VS_EXT_STOP_Y 2498
15-362 Register Call Summary for Register VENC_VS_EXT_STOP_Y 2498
15-363 VENC_AVID_START_STOP_X 2498
15-364 Register Call Summary for Register VENC_AVID_START_STOP_X 2498
15-365 VENC_AVID_START_STOP_Y 2499
15-366 Register Call Summary for Register VENC_AVID_START_STOP_Y 2499
15-367 VENC_FID_INT_START_X_FID_INT_START_Y 2499
15-368 Register Call Summary for Register VENC_FID_INT_START_X_FID_INT_START_Y 2499
15-369 VENC_FID_INT_OFFSET_Y_FID_EXT_START_X 2500
15-370 Register Call Summary for Register VENC_FID_INT_OFFSET_Y_FID_EXT_START_X 2500
15-371 VENC_FID_EXT_START_Y_FID_EXT_OFFSET_Y 2500
15-372 Register Call Summary for Register VENC_FID_EXT_START_Y_FID_EXT_OFFSET_Y 2500
15-373 VENC_TVDETGP_INT_START_STOP_X 2501
15-374 Register Call Summary for Register VENC_TVDETGP_INT_START_STOP_X 2501
15-375 VENC_TVDETGP_INT_START_STOP_Y 2501
15-376 Register Call Summary for Register VENC_TVDETGP_INT_START_STOP_Y 2501
15-377 VENC_GEN_CTRL 2502
15-378 Register Call Summary for Register VENC_GEN_CTRL 2503
15-379 VENC_OUTPUT_CONTROL 2503
15-380 Register Call Summary for Register VENC_OUTPUT_CONTROL 2504
15-381 VENC_OUTPUT_TEST 2504
15-382 Register Call Summary for Register VENC_OUTPUT_TEST 2504
15-383 DSI_REVISION 2505
15-384 Register Call Summary for Register DSI_REVISION 2505
15-385 DSI_SYSCONFIG 2505
15-386 Register Call Summary for Register DSI_SYSCONFIG 2506
15-387 DSI_SYSSTATUS 2506
15-388 Register Call Summary for Register DSI_SYSSTATUS 2507
15-389 DSI_IRQSTATUS 2507
15-390 Register Call Summary for Register DSI_IRQSTATUS 2509
15-391 DSI_IRQENABLE 2510
15-392 Register Call Summary for Register DSI_IRQENABLE 2511
15-393 DSI_CTRL 2511
15-394 Register Call Summary for Register DSI_CTRL 2514
15-395 DSI_COMPLEXIO_CFG1 2515
15-396 Register Call Summary for Register DSI_COMPLEXIO_CFG1 2517
15-397 DSI_COMPLEXIO_IRQSTATUS 2517
15-398 Register Call Summary for Register DSI_COMPLEXIO_IRQSTATUS 2520
15-399 DSI_COMPLEXIO_IRQENABLE 2520
15-400 Register Call Summary for Register DSI_COMPLEXIO_IRQENABLE 2522
15-401 DSI_CLK_CTRL 2522
15-402 Register Call Summary for Register DSI_CLK_CTRL 2524
15-403 DSI_TIMING1 2524
15-404 Register Call Summary for Register DSI_TIMING1 2525
15-405 DSI_TIMING2 2526
15-406 Register Call Summary for Register DSI_TIMING2 2527
15-407 DSI_VM_TIMING1 2527

15-408 Register Call Summary for Register DSI_VM_TIMING1 2527
15-409 DSI_VM_TIMING2 2527
15-410 Register Call Summary for Register DSI_VM_TIMING2 2528
15-411 DSI_VM_TIMING3 2528
15-412 Register Call Summary for Register DSI_VM_TIMING3 2528
15-413 DSI_CLK_TIMING 2529
15-414 Register Call Summary for Register DSI_CLK_TIMING 2529
15-415 DSI_TX_FIFO_VC_SIZE 2529
15-416 Register Call Summary for Register DSI_TX_FIFO_VC_SIZE 2530
15-417 DSI_RX_FIFO_VC_SIZE 2530
15-418 Register Call Summary for Register DSI_RX_FIFO_VC_SIZE 2531
15-419 DSI_COMPLEXIO_CFG2 2531
15-420 Register Call Summary for Register DSI_COMPLEXIO_CFG2 2533
15-421 DSI_RX_FIFO_VC_FULLNESS 2533
15-422 Register Call Summary for Register DSI_RX_FIFO_VC_FULLNESS 2534
15-423 DSI_VM_TIMING4 2534
15-424 Register Call Summary for Register DSI_VM_TIMING4 2534
15-425 DSI_TX_FIFO_VC_EMPTINESS 2534
15-426 Register Call Summary for Register DSI_TX_FIFO_VC_EMPTINESS 2535
15-427 DSI_VM_TIMING5 2535
15-428 Register Call Summary for Register DSI_VM_TIMING5 2535
15-429 DSI_VM_TIMING6 2536
15-430 Register Call Summary for Register DSI_VM_TIMING6 2536
15-431 DSI_VM_TIMING7 2536
15-432 Register Call Summary for Register DSI_VM_TIMING7 2537
15-433 DSI_STOPCLK_TIMING 2537
15-434 Register Call Summary for Register DSI_STOPCLK_TIMING 2537
15-435 DSI_VCn_CTRL 2537
15-436 Register Call Summary for Register DSI_VCn_CTRL 2540
15-437 DSI_VCn_TE 2541
15-438 Register Call Summary for Register DSI_VCn_TE 2541
15-439 DSI_VCn_LONG_PACKET_HEADER 2542
15-440 Register Call Summary for Register DSI_VCn_LONG_PACKET_HEADER 2542
15-441 DSI_VCn_LONG_PACKET_PAYLOAD 2542
15-442 Register Call Summary for Register DSI_VCn_LONG_PACKET_PAYLOAD 2543
15-443 DSI_VCn_SHORT_PACKET_HEADER 2543
15-444 Register Call Summary for Register DSI_VCn_SHORT_PACKET_HEADER 2543
15-445 DSI_VCn_IRQSTATUS 2544
15-446 Register Call Summary for Register DSI_VCn_IRQSTATUS 2545
15-447 DSI_VCn_IRQENABLE 2545
15-448 Register Call Summary for Register DSI_VCn_IRQENABLE 2547
15-449 DSI_PHY_CFG0 2547
15-450 Register Call Summary for Register DSI_PHY_CFG0 2547
15-451 DSI_PHY_CFG1 2548
15-452 Register Call Summary for Register DSI_PHY_CFG1 2548
15-453 DSI_PHY_CFG2 2548
15-454 Register Call Summary for Register DSI_PHY_CFG2 2549
15-455 DSI_PHY_CFG5 2549
15-456 Register Call Summary for Register DSI_PHY_CFG5 2550
15-457 DSI_PLL_CONTROL 2550
15-458 Register Call Summary for Register DSI_PLL_CONTROL 2550

15-459 DSI_PLL_STATUS 2551
15-460 Register Call Summary for Register DSI_PLL_STATUS 2552
15-461 DSI_PLL_GO 2552
15-462 Register Call Summary for Register DSI_PLL_GO 2553
15-463 DSI_PLL_CONFIGURATION1 2553
15-464 Register Call Summary for Register DSI_PLL_CONFIGURATION1 2553
15-465 DSI_PLL_CONFIGURATION2 2554
15-466 Register Call Summary for Register DSI_PLL_CONFIGURATION2 2555
16-1 Input/Output Description 2561
16-2 Clock, Power, and Reset Domains for GP Timers 2563
16-3 GP Timer PRCM Clock Selection Bits 2563
16-4 GP Timer PRCM Clock Control Bits 2563
16-5 IDLEMODE Settings 2564
16-6 CLOCKACTIVITY Settings 2565
16-7 Timer Interrupt Names and Processor IRQ Mapping 2567
16-8 Value Loaded in GPTi.TCRR to Generate 1-ms Tick 2572
16-9 Prescaler/Timer Reload Values Versus Contexts 2575
16-10 Prescaler Clock Ratio Values 2576
16-11 Value and Corresponding Interrupt Period 2577
16-12 GP Timer Instance Summary 2580
16-13 GPTIMER1 to GPTIMER4 Register Summary 2581
16-14 GPTIMER5 to GPTIMER8 Register Summary 2582
16-15 GPTIMER9 to GPTIMER11 Register Summary 2583
16-16 TIDR 2583
16-17 Register Call Summary for Register TIDR 2584
16-18 TIOCP_CFG 2584
16-19 Register Call Summary for Register TIOCP_CFG 2585
16-20 TISTAT 2586
16-21 Register Call Summary for Register TISTAT 2586
16-22 TISR 2587
16-23 Register Call Summary for Register TISR 2587
16-24 TIER 2588
16-25 Register Call Summary for Register TIER 2588
16-26 TWER 2589
16-27 Register Call Summary for Register TWER 2589
16-28 TCLR 2590
16-29 Register Call Summary for Register TCLR 2591
16-30 TCRR 2591
16-31 Register Call Summary for Register TCRR 2592
16-32 TLDR 2592
16-33 Register Call Summary for Register TLDR 2592
16-34 TTGR 2593
16-35 Register Call Summary for Register TTGR 2593
16-36 TWPS 2594
16-37 Register Call Summary for Register TWPS 2595
16-38 TMAR 2595
16-39 Register Call Summary for Register TMAR 2596
16-40 TCAR1 2596
16-41 Register Call Summary for Register TCAR1 2596
16-42 TSICR 2597
16-43 Register Call Summary for Register TSICR 2597

16-44 TCAR2 2598
16-45 Register Call Summary for Register TCAR2 2598
16-46 TPIR 2598
16-47 Register Call Summary for Register TPIR 2599
16-48 TNIR 2599
16-49 Register Call Summary for Register TNIR 2599
16-50 TCVR 2599
16-51 Register Call Summary for Register TCVR 2600
16-52 TOCR 2600
16-53 Register Call Summary for Register TOCR 2600
16-54 TOWR 2600
16-55 Register Call Summary for Register TOWR 2601
16-56 WD Timers Default State for GP and EMU devices 2602
16-57 Clock, Power, and Reset Domains for WDTs 2604
16-58 WDT PRCM Clock Control Bits 2604
16-59 IDLEMODE Settings 2604
16-60 CLOCKACTIVITY Settings 2605
16-61 WDT Interrupt Names and Processor IRQ Mapping 2606
16-62 Count and Prescaler Default Reset Values 2607
16-63 Prescaler Clock Ratios 2607
16-64 Reset Period Examples 2608
16-65 Default WDT Time Periods 2608
16-66 WDT Instance Summary 2611
16-67 WDTIMER2 Register Summary 2611
16-68 WDTIMER3 Register Summary 2611
16-69 WIDR 2612
16-70 Register Call Summary for Register WIDR 2612
16-71 WD_SYSCONFIG 2612
16-72 Register Call Summary for Register WD_SYSCONFIG 2613
16-73 WD_SYSSTATUS 2613
16-74 Register Call Summary for Register WD_SYSSTATUS 2614
16-75 WISR 2614
16-76 Register Call Summary for Register WISR 2614
16-77 WIER 2615
16-78 Register Call Summary for Register WIER 2615
16-79 WCLR 2615
16-80 Register Call Summary for Register WCLR 2616
16-81 WCRR 2616
16-82 Register Call Summary for Register WCRR 2616
16-83 WLDR 2616
16-84 Register Call Summary for Register WLDR 2617
16-85 WTGR 2617
16-86 Register Call Summary for Register WTGR 2617
16-87 WWPS 2617
16-88 Register Call Summary for Register WWPS 2618
16-89 WSPR 2618
16-90 Register Call Summary for Register WSPR 2618
16-91 Clock, Power, and Reset Domains for 32-kHz Sync Timer 2620
16-92 32-kHz Sync Timer Instance Summary 2621
16-93 32-kHz Sync Timer Register Summary 2621
16-94 REG_32KSYNCNT_REV 2621

16-95 Register Call Summary for Register REG_32KSYNCNT_REV 2622
16-96 REG_32KSYNCNT_SYSCONFIG 2622
16-97 Register Call Summary for Register REG_32KSYNCNT_SYSCONFIG 2622
16-98 REG_32KSYNCNT_CR 2622
16-99 Register Call Summary for Register REG_32KSYNCNT_CR 2622
17-1 UART Mode Baud Rates, Divisor Values, and Error Rates 2625
17-2 UART IrDA Mode Baud Rates, Divisor Values, and Error Rates 2626
17-3 UART I/O Pin Description 2628
17-4 UART3 I/O Description 2629
17-5 EFR_REG[0-1] IR Address Checking Options 2632
17-6 FIR Transmit Frame Format 2635
17-7 4-PPM format 2635
17-8 FIR Preamble, Start Flag, and Stop Flag 2635
17-9 FIR Data Byte Transmission Order Example 2635
17-10 CIR I/O Description 2636
17-11 UART Clocks 2640
17-12 Reset Domain 2641
17-13 Power Domain 2641
17-14 Interrupt Mapping to MPU Subsystem 2642
17-15 Interrupt Mapping to IVA2.2 Subsystem 2642
17-16 UART DMA Requests to System DMA 2642
17-17 UART DMA Requests to IVA2.2 Subsystem DMA 2642
17-18 Wake-Up Requests from PRCM 2642
17-19 TX FIFO Trigger Level Setting Summary 2646
17-20 RX FIFO Trigger Level Setting Summary 2646
17-21 UART/IrDA/CIR Register Access Mode Programming (Using LCR_REG) 2653
17-22 Sub-Configuration_Mode_A Mode Summary 2653
17-23 Sub-Configuration_Mode_B Mode Summary 2653
17-24 Sub-Operational_Mode Mode Summary 2653
17-25 UART/IrDA/CIR Register Access Mode Overview 2654
17-26 UART Mode Selection 2655
17-27 UART Mode Register Overview 2655
17-28 IrDA Mode Register Overview 2656
17-29 CIR Mode Register Overview 2657
17-30 UART Baud Rate Settings (48-MHz Clock) 2658
17-31 UART Parity Bit Encoding 2659
17-32 EFR_REG[0-3] Software Flow Control Options 2660
17-33 UART Mode Interrupts 2663
17-34 IrDA Baud Rates Settings 2665
17-35 IrDA Mode Interrupts 2669
17-36 Duty Cycle 2671
17-37 CIR Mode Interrupts 2672
17-38 UART/IrDA/CIR Instance Summary 2682
17-39 UART/IrDA/CIR Register Summary 2682
17-40 DLL_REG 2684
17-41 Register Call Summary for Register DLL_REG 2684
17-42 RHR_REG 2684
17-43 Register Call Summary for Register RHR_REG 2685
17-44 THR_REG 2685
17-45 Register Call Summary for Register THR_REG 2685
17-46 IER_REG 2686

17-47 Register Call Summary for Register IER_REG 2687
17-48 DLH_REG 2688
17-49 Register Call Summary for Register DLH_REG 2689
17-50 FCR_REG 2689
17-51 Register Call Summary for Register FCR_REG 2690
17-52 IIR_REG 2690
17-53 Register Call Summary for Register IIR_REG 2691
17-54 EFR_REG 2692
17-55 Register Call Summary for Register EFR_REG 2693
17-56 LCR_REG 2694
17-57 Register Call Summary for Register LCR_REG 2695
17-58 MCR_REG 2695
17-59 Register Call Summary for Register MCR_REG 2696
17-60 XON1_ADDR1_REG 2696
17-61 Register Call Summary for Register XON1_ADDR1_REG 2696
17-62 LSR_REG 2697
17-63 Register Call Summary for Register LSR_REG 2698
17-64 XON2_ADDR2_REG 2699
17-65 Register Call Summary for Register XON2_ADDR2_REG 2699
17-66 XOFF1_REG 2700
17-67 Register Call Summary for Register XOFF1_REG 2700
17-68 TCR_REG 2700
17-69 Register Call Summary for Register TCR_REG 2701
17-70 MSR_REG 2701
17-71 Register Call Summary for Register MSR_REG 2702
17-72 SPR_REG 2702
17-73 Register Call Summary for Register SPR_REG 2702
17-74 XOFF2_REG 2702
17-75 Register Call Summary for Register XOFF2_REG 2702
17-76 TLR_REG 2703
17-77 Register Call Summary for Register TLR_REG 2703
17-78 MDR1_REG 2703
17-79 Register Call Summary for Register MDR1_REG 2704
17-80 MDR2_REG 2705
17-81 Register Call Summary for Register MDR2_REG 2706
17-82 TXFLL_REG 2706
17-83 Register Call Summary for Register TXFLL_REG 2706
17-84 SFLSR_REG 2707
17-85 Register Call Summary for Register SFLSR_REG 2707
17-86 RESUME_REG 2708
17-87 Register Call Summary for Register RESUME_REG 2708
17-88 TXFLH_REG 2708
17-89 Register Call Summary for Register TXFLH_REG 2708
17-90 RXFLL_REG 2709
17-91 Register Call Summary for Register RXFLL_REG 2709
17-92 SFREGL_REG 2709
17-93 Register Call Summary for Register SFREGL_REG 2710
17-94 SFREGH_REG 2710
17-95 Register Call Summary for Register SFREGH_REG 2710
17-96 RXFLH_REG 2711
17-97 Register Call Summary for Register RXFLH_REG 2711

17-98 BLR_REG 2711
17-99 Register Call Summary for Register BLR_REG 2712
17-100 UASR_REG 2712
17-101 Register Call Summary for Register UASR_REG 2713
17-102 ACREG_REG 2713
17-103 Register Call Summary for Register ACREG_REG 2714
17-104 SCR_REG 2714
17-105 Register Call Summary for Register SCR_REG 2715
17-106 SSR_REG 2716
17-107 Register Call Summary for Register SSR_REG 2716
17-108 EBLR_REG 2716
17-109 Register Call Summary for Register EBLR_REG 2717
17-110 MVR_REG 2717
17-111 Register Call Summary for Register MVR_REG 2717
17-112 SYSC_REG 2718
17-113 Register Call Summary for Register SYSC_REG 2718
17-114 SYSS_REG 2719
17-115 Register Call Summary for Register SYSS_REG 2719
17-116 WER_REG 2719
17-117 Register Call Summary for Register WER_REG 2720
17-118 CFPS_REG 2720
17-119 Register Call Summary for Register CFPS_REG 2721
18-1 Input/Output 2726
18-2 Input/Output 2732
18-3 Input/Output Description 2735
18-4 Multimaster HS I^2C Controller Power Management Modes 2740
18-5 State of the Interface and Functional Clocks When the Module is in Idle Mode 2740
18-6 Wake-up Events 2741
18-7 Multimaster HS I^2C Controller DMA Requests 2743
18-8 Multimaster HS I^2C Controller Interrupt Requests 2743
18-9 Multimaster HS I^2C Controller Interrupt Events 2744
18-10 Operation Mode Selection 2746
18-11 RX and TX FIFO Depths 2747
18-12 t_{LOW} and t_{high}Values of the I^2C Clock 2752
18-13 List of tests for the Multimaster HS I^2C Controllers 2753
18-14 Registers Print for the I2Ci Module Initialization 2776
18-15 Registers Print for the I2Ci Module Clock Generation Configuration 2776
18-16 Registers Print for the I2Ci Slave Address Configuration 2777
18-17 Registers Print for the I2Ci Slave Address Configuration 2777
18-18 Registers Print for the I2Ci Module Enabling 2778
18-19 Registers Print for the Transfer Configuration 2778
18-20 Registers Print for the I2Ci Bus Status Checking 2778
18-21 Registers Print for the Transfer Initiation 2779
18-22 Registers Print for the Data Transfer 2779
18-23 Registers Print for the Transfer Completion 2780
18-24 Variables and Constants 2784
18-25 Instance Summary 2785
18-26 Register Summary 2785
18-27 I2C_REV 2786
18-28 Register Call Summary for Register I2C_REV 2786
18-29 I2C_IE 2786

18-30 Register Call Summary for Register I2C_IE 2787
18-31 I2C_STAT 2788
18-32 Register Call Summary for Register I2C_STAT 2790
18-33 I2C_WE 2790
18-34 Register Call Summary for Register I2C_WE 2791
18-35 I2C_SYSS 2791
18-36 Register Call Summary for Register I2C_SYSS 2792
18-37 I2C_BUF 2792
18-38 Register Call Summary for Register I2C_BUF 2792
18-39 I2C_CNT 2793
18-40 Register Call Summary for Register I2C_CNT 2793
18-41 I2C_DATA 2794
18-42 Register Call Summary for Register I2C_DATA 2794
18-43 I2C_SYSC 2794
18-44 Register Call Summary for Register I2C_SYSC 2795
18-45 I2C_CON 2795
18-46 Register Call Summary for Register I2C_CON 2796
18-47 I2C_OA0 2797
18-48 Register Call Summary for Register I2C_OA0 2797
18-49 I2C_SA 2798
18-50 Register Call Summary for Register I2C_SA 2798
18-51 I2C_PSC 2798
18-52 Register Call Summary for Register I2C_PSC 2798
18-53 I2C_SCLL 2799
18-54 Register Call Summary for Register I2C_SCLL 2799
18-55 I2C_SCLH 2800
18-56 Register Call Summary for Register I2C_SCLH 2800
18-57 I2C_SYSTEST 2800
18-58 Register Call Summary for Register I2C_SYSTEST 2801
18-59 I2C_BUFSTAT 2801
18-60 Register Call Summary for Register I2C_BUFSTAT 2802
18-61 I2C_OA1 2802
18-62 Register Call Summary for Register I2C_OA1 2802
18-63 I2C_OA2 2803
18-64 Register Call Summary for Register I2C_OA2 2803
18-65 I2C_OA3 2803
18-66 Register Call Summary for Register I2C_OA3 2803
18-67 I2C_ACTOA 2804
18-68 Register Call Summary for Register I2C_ACTOA 2804
18-69 I2C_SBLOCK 2804
18-70 Register Call Summary for Register I2C_SBLOCK 2805
19-1 McSPI I/O Description (Master Mode) 2814
19-2 McSPI I/O Description (Slave Mode) 2815
19-3 SPI Master Clock Rates 2815
19-4 Phase and Polarity Combinations 2816
19-5 McSPI Clocks 2819
19-6 Power Domain 2820
19-7 McSPI Hardware Reset 2820
19-8 DMA Requests 2821
19-9 Interrupt Requests 2822
19-10 Wake-Up Requests 2822

19-11 SPI Master Clock Rates 2829
19-12 CLKSPIO High/Low Time Computation 2829
19-13 Clock Granularity Examples 2830
19-14 FIFO Writes, Word Length Relationship 2835
19-15 Smart-Idle Mode and Wake-Up Capabilities 2842
19-16 End-of-Transfer Sequences 2846
19-17 End-of-Transfer Types 2859
19-18 EPSON VGA Configuration Commands 2868
19-19 McSPI Configuration Registers Print 2868
19-20 Display Configuration Registers Print 2869
19-21 Display Status Check Registers Print 2870
19-22 McSPI Instance Summary 2871
19-23 McSPI Register Summary 2871
19-24 MCSPI_REVISION 2872
19-25 Register Call Summary for Register MCSPI_REVISION 2872
19-26 MCSPI_SYSCONFIG 2872
19-27 Register Call Summary for Register MCSPI_SYSCONFIG 2873
19-28 MCSPI_SYSSTATUS 2874
19-29 Register Call Summary for Register MCSPI_SYSSTATUS 2874
19-30 MCSPI_IRQSTATUS 2874
19-31 Register Call Summary for Register MCSPI_IRQSTATUS 2877
19-32 MCSPI_IRQENABLE 2877
19-33 Register Call Summary for Register MCSPI_IRQENABLE 2879
19-34 MCSPI_WAKEUPENABLE 2879
19-35 Register Call Summary for Register MCSPI_WAKEUPENABLE 2879
19-36 MCSPI_SYST 2880
19-37 Register Call Summary for Register MCSPI_SYST 2881
19-38 MCSPI_MODULCTRL 2882
19-39 Register Call Summary for Register MCSPI_MODULCTRL 2882
19-40 MCSPI_CHxCONF 2883
19-41 Register Call Summary for Register MCSPI_CHxCONF 2886
19-42 MCSPI_CHxSTAT 2886
19-43 Register Call Summary for Register MCSPI_CHxSTAT 2887
19-44 MCSPI_CHxCTRL 2888
19-45 Register Call Summary for Register MCSPI_CHxCTRL 2888
19-46 MCSPI_TXx 2889
19-47 Register Call Summary for Register MCSPI_TXx 2889
19-48 MCSPI_RXx 2890
19-49 Register Call Summary for Register MCSPI_RXx 2890
19-50 MCSPI_XFERLEVEL 2891
19-51 Register Call Summary for Register MCSPI_XFERLEVEL 2892
20-1 I/O Description 2895
20-2 HDQ/1-Wire Command Byte 2897
20-3 Registers Print for HDQ/1-Wire Configuration 2910
20-4 Registers Print for HDQ/1-Wire Software Reset 2911
20-5 Registers Print for HDQ/1-Wire Interrupts Enable 2912
20-6 Instance Summary 2913
20-7 HDQ/1-Wire Register Summary 2913
20-8 HDQ_REVISION 2914
20-9 Register Call Summary for Register HDQ_REVISION 2914
20-10 HDQ_TX_DATA 2914

20-11 Register Call Summary for Register HDQ_TX_DATA 2914
20-12 HDQ_RX_DATA 2915
20-13 Register Call Summary for Register HDQ_RX_DATA 2915
20-14 HDQ_CTRL_STATUS 2915
20-15 Register Call Summary for Register HDQ_CTRL_STATUS 2916
20-16 HDQ_INT_STATUS 2916
20-17 Register Call Summary for Register HDQ_INT_STATUS 2917
20-18 HDQ_SYSCONFIG 2917
20-19 Register Call Summary for Register HDQ_SYSCONFIG 2918
20-20 HDQ_SYSSTATUS 2918
20-21 Register Call Summary for Register HDQ_SYSSTATUS 2918
21-1 Functions Description 2923
21-2 Input/Output Description 2924
21-3 Clocking Signals Input to McBSP Module 2937
21-4 Software Reset Signals to All McBSP Modules 2943
21-5 State of Clocks When the Module is in Idle State 2944
21-6 McBSP Smart Idle Mode Configuration Behavior 2947
21-7 McBSP DMA Requests 2949
21-8 McBSP Common Interrupt Requests 2949
21-9 McBSP Transmit Interrupt Requests 2950
21-10 McBSP Receive Interrupt Requests 2950
21-11 McBSP Transmit Interrupt Events 2950
21-12 McBSP Receive Interrupt Events 2951
21-13 SIDETONE_McBSP Interrupt Requests 2951
21-14 SIDETONE_McBSP Interrupt Events 2951
21-15 Receiver Clock Mode 2958
21-16 Phases, Words and Bits per Frame Control Bit 2961
21-17 Assumptions for the Single-Phase Frame Example 2962
21-18 Assumptions for the Dual-Phase Frame Example 2962
21-19 Effects of DLB and ALB Bits on Clock Modes 2968
21-20 Eight Partitions – Receive Channel Assignment and Control 2978
21-21 Eight Partitions – Transmit Channel Assignment and Control 2978
21-22 Selecting a Transmit Multichannel Selection Mode With the XMCM Bit Field 2980
21-23 McBSP Channel Control Options 2981
21-24 McBSP Configuration in Function of the SRG Clock Source Selected 2991
21-25 Input Clock Selection for Sample Rate Generator 2994
21-26 How to Calculate the Length of the Receive Frame 3000
21-27 Example: Use of RJUST Bit Field With 12-bit Data Value 0xABC 3001
21-28 Example: Use of RJUST Bit Field With 20-bit Data Value 0xABCDE 3001
21-29 FSRM and GSYNC Effects on Frame-Sync Signal and mcbsp_fsr Pin 3002
21-30 CLKRM Effect on Receive Clock Signal and mcbsp_clkr Pin 3004
21-31 How to Calculate the Length of the Transmit Frame 3008
21-32 How FSXM and FSGM Select the Source of Transmit Frame-Sync Pulses 3010
21-33 CLKXM Bit Effect on Transmit Clock and MCBSPLP.CLKX Pin 3011
21-34 Using McBSP Pins for General-Purpose I/O 3013
21-35 Selection of the SIDETONE Input and Output Channels 3016
21-36 McBSP2 Registers Print 3019
21-37 OMAP34xx Instance Summary 3020
21-38 McBSP1 Registers Mapping Summary 3020
21-39 McBSP5 Registers Mapping Summary 3021
21-40 McBSP2 Registers Mapping Summary 3022

21-41 McBSP3 Registers Mapping Summary ... 3023
21-42 McBSP4 Registers Mapping Summary ... 3024
21-43 SIDETONE_McBSP2 Registers Mapping Summary ... 3025
21-44 SIDETONE_McBSP3 Registers Mapping Summary ... 3025
21-45 MCBSPLP_DRR_REG ... 3026
21-46 Register Call Summary for Register MCBSPLP_DRR_REG ... 3026
21-47 MCBSPLP_DXR_REG ... 3027
21-48 Register Call Summary for Register MCBSPLP_DXR_REG ... 3027
21-49 MCBSPLP_SPCR2_REG ... 3027
21-50 Register Call Summary for Register MCBSPLP_SPCR2_REG ... 3028
21-51 MCBSPLP_SPCR1_REG ... 3029
21-52 Register Call Summary for Register MCBSPLP_SPCR1_REG ... 3030
21-53 MCBSPLP_RCR2_REG ... 3031
21-54 Register Call Summary for Register MCBSPLP_RCR2_REG ... 3031
21-55 MCBSPLP_RCR1_REG ... 3032
21-56 Register Call Summary for Register MCBSPLP_RCR1_REG ... 3033
21-57 MCBSPLP_XCR2_REG ... 3033
21-58 Register Call Summary for Register MCBSPLP_XCR2_REG ... 3034
21-59 MCBSPLP_XCR1_REG ... 3034
21-60 Register Call Summary for Register MCBSPLP_XCR1_REG ... 3035
21-61 MCBSPLP_SRGR2_REG ... 3035
21-62 Register Call Summary for Register MCBSPLP_SRGR2_REG ... 3036
21-63 MCBSPLP_SRGR1_REG ... 3037
21-64 Register Call Summary for Register MCBSPLP_SRGR1_REG ... 3037
21-65 MCBSPLP_MCR2_REG ... 3037
21-66 Register Call Summary for Register MCBSPLP_MCR2_REG ... 3039
21-67 MCBSPLP_MCR1_REG ... 3039
21-68 Register Call Summary for Register MCBSPLP_MCR1_REG ... 3040
21-69 MCBSPLP_RCERA_REG ... 3041
21-70 Register Call Summary for Register MCBSPLP_RCERA_REG ... 3041
21-71 MCBSPLP_RCERB_REG ... 3041
21-72 Register Call Summary for Register MCBSPLP_RCERB_REG ... 3042
21-73 MCBSPLP_XCERA_REG ... 3042
21-74 Register Call Summary for Register MCBSPLP_XCERA_REG ... 3042
21-75 MCBSPLP_XCERB_REG ... 3043
21-76 Register Call Summary for Register MCBSPLP_XCERB_REG ... 3043
21-77 MCBSPLP_PCR_REG ... 3043
21-78 Register Call Summary for Register MCBSPLP_PCR_REG ... 3045
21-79 MCBSPLP_RCERC_REG ... 3046
21-80 Register Call Summary for Register MCBSPLP_RCERC_REG ... 3046
21-81 MCBSPLP_RCERD_REG ... 3046
21-82 Register Call Summary for Register MCBSPLP_RCERD_REG ... 3047
21-83 MCBSPLP_XCERC_REG ... 3047
21-84 Register Call Summary for Register MCBSPLP_XCERC_REG ... 3047
21-85 MCBSPLP_XCERD_REG ... 3047
21-86 Register Call Summary for Register MCBSPLP_XCERD_REG ... 3048
21-87 MCBSPLP_RCERE_REG ... 3048
21-88 Register Call Summary for Register MCBSPLP_RCERE_REG ... 3048
21-89 MCBSPLP_RCERF_REG ... 3049
21-90 Register Call Summary for Register MCBSPLP_RCERF_REG ... 3049
21-91 MCBSPLP_XCERE_REG ... 3049

21-92 Register Call Summary for Register MCBSPLP_XCERE_REG 3049
21-93 MCBSPLP_XCERF_REG 3050
21-94 Register Call Summary for Register MCBSPLP_XCERF_REG 3050
21-95 MCBSPLP_RCERG_REG 3050
21-96 Register Call Summary for Register MCBSPLP_RCERG_REG 3051
21-97 MCBSPLP_RCERH_REG 3051
21-98 Register Call Summary for Register MCBSPLP_RCERH_REG 3051
21-99 MCBSPLP_XCERG_REG 3051
21-100 Register Call Summary for Register MCBSPLP_XCERG_REG 3052
21-101 MCBSPLP_XCERH_REG 3052
21-102 Register Call Summary for Register MCBSPLP_XCERH_REG 3052
21-103 MCBSPLP_REV_REG 3053
21-104 Register Call Summary for Register MCBSPLP_REV_REG 3053
21-105 MCBSPLP_RINTCLR_REG 3053
21-106 Register Call Summary for Register MCBSPLP_RINTCLR_REG 3053
21-107 MCBSPLP_XINTCLR_REG 3054
21-108 Register Call Summary for Register MCBSPLP_XINTCLR_REG 3054
21-109 MCBSPLP_ROVFLCLR_REG 3054
21-110 Register Call Summary for Register MCBSPLP_ROVFLCLR_REG 3054
21-111 MCBSPLP_SYSCONFIG_REG 3055
21-112 Register Call Summary for Register MCBSPLP_SYSCONFIG_REG 3055
21-113 MCBSPLP_THRSH2_REG 3056
21-114 Register Call Summary for Register MCBSPLP_THRSH2_REG 3056
21-115 MCBSPLP_THRSH1_REG 3057
21-116 Register Call Summary for Register MCBSPLP_THRSH1_REG 3057
21-117 MCBSPLP_IRQSTATUS_REG 3057
21-118 Register Call Summary for Register MCBSPLP_IRQSTATUS_REG 3060
21-119 MCBSPLP_IRQENABLE_REG 3060
21-120 Register Call Summary for Register MCBSPLP_IRQENABLE_REG 3061
21-121 MCBSPLP_WAKEUPEN_REG 3062
21-122 Register Call Summary for Register MCBSPLP_WAKEUPEN_REG 3063
21-123 MCBSPLP_XCCR_REG 3063
21-124 Register Call Summary for Register MCBSPLP_XCCR_REG 3064
21-125 MCBSPLP_RCCR_REG 3065
21-126 Register Call Summary for Register MCBSPLP_RCCR_REG 3066
21-127 MCBSPLP_XBUFFSTAT_REG 3066
21-128 Register Call Summary for Register MCBSPLP_XBUFFSTAT_REG 3066
21-129 MCBSPLP_RBUFFSTAT_REG 3067
21-130 Register Call Summary for Register MCBSPLP_RBUFFSTAT_REG 3067
21-131 MCBSPLP_SSELCR_REG 3067
21-132 Register Call Summary for Register MCBSPLP_SSELCR_REG 3068
21-133 MCBSPLP_STATUS_REG 3068
21-134 Register Call Summary for Register MCBSPLP_STATUS_REG 3068
21-135 ST_REV_REG 3069
21-136 Register Call Summary for Register ST_REV_REG 3069
21-137 ST_SYSCONFIG_REG 3069
21-138 Register Call Summary for Register ST_SYSCONFIG_REG 3069
21-139 ST_IRQSTATUS_REG 3070
21-140 Register Call Summary for Register ST_IRQSTATUS_REG 3070
21-141 ST_IRQENABLE_REG 3070
21-142 Register Call Summary for Register ST_IRQENABLE_REG 3070

21-143 ST_SGAINCR_REG ... 3071
21-144 Register Call Summary for Register ST_SGAINCR_REG ... 3071
21-145 ST_SFIRCR_REG ... 3071
21-146 Register Call Summary for Register ST_SFIRCR_REG ... 3072
21-147 ST_SSELCR_REG ... 3072
21-148 Register Call Summary for Register ST_SSELCR_REG ... 3073
22-1 MMC/SD/SDIOi I/O Description ... 3081
22-2 MMC/SD/SDIO2 I/O Description ... 3082
22-3 Relation Between Configuration and Name of Response Type ... 3086
22-4 Smart Idle Mode and Wake-Up Capabilities ... 3090
22-5 MMC, SD, SDIO responses in the MMCHS_RSPxx registers ... 3102
22-6 CC and TC Values Upon Error Detected ... 3102
22-7 MMC/SD/SDIOi Controller Transfer Stop Command Summary ... 3104
22-8 Register Print for the MMCHS1 controller's clocks Initialization ... 3123
22-9 MMCHS Controller Voltage Capabilities Initialization ... 3123
22-10 MMC Controller Default Initialization Values ... 3124
22-11 MMCHS Controller INIT Procedure Start ... 3124
22-12 MMCHS Controller Pre-Card Identification Configuration ... 3124
22-13 Sending CMD0 ... 3125
22-14 Sending CMD5 ... 3125
22-15 Sending CMD8 ... 3125
22-16 Sending CMD55 ... 3126
22-17 Sending CMD1 ... 3126
22-18 Sending CMD2 ... 3127
22-19 Sending CMD3 ... 3127
22-20 MMC Bus Setting Change Table ... 3127
22-21 Sending CMD9 ... 3128
22-22 MMCHS_SYSCTL Value ... 3128
22-23 Sending CMD7 ... 3129
22-24 Setting Data Bus Width with CMD6 ... 3129
22-25 MMCHS_CON Value ... 3130
22-26 Enabling High Speed with CMD6 ... 3130
22-27 MMCHS_SYSCTL Value ... 3130
22-28 Setting Block Length ... 3131
22-29 Setting Number of Blocks ... 3131
22-30 CMD25 Issuing ... 3131
22-31 CMD18 Issuing ... 3132
22-32 Instance Summary ... 3134
22-33 MMC/SD/SDIO1 Register Summary ... 3134
22-34 MMCHS_SYSCONFIG ... 3135
22-35 Register Call Summary for Register MMCHS_SYSCONFIG ... 3136
22-36 MMCHS_SYSSTATUS ... 3136
22-37 Register Call Summary for Register MMCHS_SYSSTATUS ... 3136
22-38 MMCHS_CSRE ... 3137
22-39 Register Call Summary for Register MMCHS_CSRE ... 3137
22-40 MMCHS_SYSTEST ... 3137
22-41 Register Call Summary for Register MMCHS_SYSTEST ... 3141
22-42 MMCHS_CON ... 3141
22-43 Register Call Summary for Register MMCHS_CON ... 3144
22-44 MMCHS_PWCNT ... 3145
22-45 Register Call Summary for Register MMCHS_PWCNT ... 3145

22-46 MMCHS_BLK 3145
22-47 Register Call Summary for Register MMCHS_BLK 3146
22-48 MMCHS_ARG 3146
22-49 Register Call Summary for Register MMCHS_ARG 3147
22-50 MMCHS_CMD 3147
22-51 Register Call Summary for Register MMCHS_CMD 3149
22-52 MMCHS_RSP10 3149
22-53 Register Call Summary for Register MMCHS_RSP10 3149
22-54 MMCHS_RSP32 3150
22-55 Register Call Summary for Register MMCHS_RSP32 3150
22-56 MMCHS_RSP54 3150
22-57 Register Call Summary for Register MMCHS_RSP54 3150
22-58 MMCHS_RSP76 3151
22-59 Register Call Summary for Register MMCHS_RSP76 3151
22-60 MMCHS_DATA 3151
22-61 Register Call Summary for Register MMCHS_DATA 3152
22-62 MMCHS_PSTATE 3152
22-63 Register Call Summary for Register MMCHS_PSTATE 3154
22-64 MMCHS_HCTL 3154
22-65 Register Call Summary for Register MMCHS_HCTL 3156
22-66 MMCHS_SYSCTL 3156
22-67 Register Call Summary for Register MMCHS_SYSCTL 3158
22-68 MMCHS_STAT 3158
22-69 Register Call Summary for Register MMCHS_STAT 3162
22-70 MMCHS_IE 3163
22-71 Register Call Summary for Register MMCHS_IE 3164
22-72 MMCHS_ISE 3165
22-73 Register Call Summary for Register MMCHS_ISE 3166
22-74 MMCHS_AC12 3167
22-75 Register Call Summary for Register MMCHS_AC12 3167
22-76 MMCHS_CAPA 3168
22-77 Register Call Summary for Register MMCHS_CAPA 3169
22-78 MMCHS_CUR_CAPA 3170
22-79 Register Call Summary for Register MMCHS_CUR_CAPA 3170
22-80 MMCHS_REV 3171
22-81 Register Call Summary for Register MMCHS_REV 3171
24-1 Input/Output Description 3180
24-2 USB Clocks 3181
24-3 High-Speed USB Interface Clock 3182
24-4 High-Speed USB Master Interface Power Management Modes 3184
24-5 High-Speed USB Slave Interface Power Management Modes 3185
24-6 High-Speed USB Interrupts 3186
24-7 High-Speed USB Controller Instance Summary 3194
24-8 TI-Specific High-Speed USB Controller Register Summary 3194
24-9 OTG_REVISION 3194
24-10 Register Call Summary for Register OTG_REVISION 3195
24-11 OTG_SYSCONFIG 3195
24-12 Register Call Summary for Register OTG_SYSCONFIG 3195
24-13 OTG_SYSSTATUS 3196
24-14 Register Call Summary for Register OTG_SYSSTATUS 3196
24-15 OTG_INTERFSEL 3196

24-16 Register Call Summary for Register OTG_INTERFSEL 3196
24-17 OTG_SIMENABLE 3197
24-18 Register Call Summary for Register OTG_SIMENABLE 3197
24-19 OTG_FORCESTDBY 3197
24-20 Register Call Summary for Register OTG_FORCESTDBY 3197
24-21 USB Connectivity Modes 3200
24-22 I/O Description 3208
24-23 Signaling Between High-Speed USB Host Subsystem and 6-Pin Unidirectional USB Transceiver (DAT/SE0 Signaling) 3211
24-24 Signaling Between High-Speed USB Host Subsystem and 6-Pin Unidirectional USB Transceiver (DP/DM Signaling) 3212
24-25 Signaling Between High-Speed USB Host Subsystem and 3-Pin Bidirectional USB Transceiver Using DAT/SE0 Signaling 3212
24-26 Signaling Between High-Speed USB Host Subsystem and 4-Pin Bidirectional USB Transceiver Using DP/DM Signaling 3213
24-27 Pullup/Pulldown Configuration for DP/DM Encoding 3221
24-28 Pullup/Pulldown Configuration for DAT/SE0 Encoding 3221
24-29 I/O Description 3221
24-30 High-Speed USB Host Subsystem Reset Description 3226
24-31 High-Speed USB Host Subsystem Clocks 3227
24-32 High-Speed USB Controller L3 Master Interface Clock 3227
24-33 USBTLL Module Interface Clock 3228
24-34 High-Speed USB Host Controller PRCM Clock Control Bits 3229
24-35 High-Speed USB Host Controller MIDLEMODE Settings 3230
24-36 High-Speed USB Host Controller SIDLEMODE Settings 3230
24-37 High-Speed USB Host Controller CLOCKACTIVITY Settings 3231
24-38 USBTLL Module PRCM Clock Control Bits 3231
24-39 USBTLL Module SIDLEMODE Settings 3232
24-40 USBTLL Module CLOCKACTIVITY Settings 3232
24-41 High-Speed USB Host Subsystem Interrupts 3232
24-42 USBTLL Channel USB Ports 3238
24-43 USBTLL Channel Configuration 3239
24-44 VBUS Level Software Reporting for Serial Transceiver Configuration 3241
24-45 Emulation of VBUS Levels for UTMI-to-ULPI TLL Mode 3242
24-46 Serial Mode Description, Signal Functionality 3243
24-47 Pullup Enable Emulation in Serial TLL Modes 3244
24-48 USBTLL Registers Impacted by the SAR Context 3244
24-49 USB Connectivity Mode Description 3248
24-50 ULPI Registers Mapping Summary (For a Single ULPI Port) 3250
24-51 High-Speed USB Host Subsystem Instance Summary 3251
24-52 USBTLL Registers Mapping Summary (L4-Core Interconnect Register Space) 3252
24-53 UHH_config Registers Mapping Summary 3253
24-54 OHCI Registers Mapping Summary 3253
24-55 EHCI Registers Mapping Summary 3254
24-56 USBTLL_REVISION 3255
24-57 Register Call Summary for Register USBTLL_REVISION 3255
24-58 USBTLL_SYSCONFIG 3255
24-59 Register Call Summary for Register USBTLL_SYSCONFIG 3256
24-60 USBTLL_SYSSTATUS 3256
24-61 Register Call Summary for Register USBTLL_SYSSTATUS 3256
24-62 USBTLL_IRQSTATUS 3257
24-63 Register Call Summary for Register USBTLL_IRQSTATUS 3257

24-64 USBTLL_IRQENABLE 3257
24-65 Register Call Summary for Register USBTLL_IRQENABLE 3258
24-66 TLL_SHARED_CONF 3258
24-67 Register Call Summary for Register TLL_SHARED_CONF 3259
24-68 TLL_CHANNEL_CONF_i 3259
24-69 Register Call Summary for Register TLL_CHANNEL_CONF_i 3262
24-70 ULPI_VENDOR_ID_LO_i 3262
24-71 Register Call Summary for Register ULPI_VENDOR_ID_LO_i 3262
24-72 ULPI_VENDOR_ID_HI_i 3262
24-73 Register Call Summary for Register ULPI_VENDOR_ID_HI_i 3262
24-74 ULPI_PRODUCT_ID_LO_i 3262
24-75 Register Call Summary for Register ULPI_PRODUCT_ID_LO_i 3263
24-76 ULPI_PRODUCT_ID_HI_i 3263
24-77 Register Call Summary for Register ULPI_PRODUCT_ID_HI_i 3263
24-78 ULPI_FUNCTION_CTRL_i 3263
24-79 Register Call Summary for Register ULPI_FUNCTION_CTRL_i 3264
24-80 ULPI_FUNCTION_CTRL_SET_i 3264
24-81 Register Call Summary for Register ULPI_FUNCTION_CTRL_SET_i 3265
24-82 ULPI_FUNCTION_CTRL_CLR_i 3265
24-83 Register Call Summary for Register ULPI_FUNCTION_CTRL_CLR_i 3265
24-84 ULPI_INTERFACE_CTRL_i 3265
24-85 Register Call Summary for Register ULPI_INTERFACE_CTRL_i 3266
24-86 ULPI_INTERFACE_CTRL_SET_i 3266
24-87 Register Call Summary for Register ULPI_INTERFACE_CTRL_SET_i 3267
24-88 ULPI_INTERFACE_CTRL_CLR_i 3267
24-89 Register Call Summary for Register ULPI_INTERFACE_CTRL_CLR_i 3268
24-90 ULPI_OTG_CTRL_i 3268
24-91 Register Call Summary for Register ULPI_OTG_CTRL_i 3269
24-92 ULPI_OTG_CTRL_SET_i 3269
24-93 Register Call Summary for Register ULPI_OTG_CTRL_SET_i 3270
24-94 ULPI_OTG_CTRL_CLR_i 3270
24-95 Register Call Summary for Register ULPI_OTG_CTRL_CLR_i 3271
24-96 ULPI_USB_INT_EN_RISE_i 3271
24-97 Register Call Summary for Register ULPI_USB_INT_EN_RISE_i 3271
24-98 ULPI_USB_INT_EN_RISE_SET_i 3272
24-99 Register Call Summary for Register ULPI_USB_INT_EN_RISE_SET_i 3272
24-100 ULPI_USB_INT_EN_RISE_CLR_i 3273
24-101 Register Call Summary for Register ULPI_USB_INT_EN_RISE_CLR_i 3273
24-102 ULPI_USB_INT_EN_FALL_i 3274
24-103 Register Call Summary for Register ULPI_USB_INT_EN_FALL_i 3274
24-104 ULPI_USB_INT_EN_FALL_SET_i 3274
24-105 Register Call Summary for Register ULPI_USB_INT_EN_FALL_SET_i 3275
24-106 ULPI_USB_INT_EN_FALL_CLR_i 3275
24-107 Register Call Summary for Register ULPI_USB_INT_EN_FALL_CLR_i 3276
24-108 ULPI_USB_INT_STATUS_i 3276
24-109 Register Call Summary for Register ULPI_USB_INT_STATUS_i 3277
24-110 ULPI_USB_INT_LATCH_i 3277
24-111 Register Call Summary for Register ULPI_USB_INT_LATCH_i 3278
24-112 ULPI_DEBUG_i 3278
24-113 Register Call Summary for Register ULPI_DEBUG_i 3278
24-114 ULPI_SCRATCH_REGISTER_i 3278

24-115 Register Call Summary for Register ULPI_SCRATCH_REGISTER_i 3278
24-116 ULPI_SCRATCH_REGISTER_SET_i 3279
24-117 Register Call Summary for Register ULPI_SCRATCH_REGISTER_SET_i 3279
24-118 ULPI_SCRATCH_REGISTER_CLR_i 3279
24-119 Register Call Summary for Register ULPI_SCRATCH_REGISTER_CLR_i 3279
24-120 ULPI_EXTENDED_SET_ACCESS_i 3279
24-121 Register Call Summary for Register ULPI_EXTENDED_SET_ACCESS_i 3280
24-122 ULPI_UTMI_VCONTROL_EN_i 3280
24-123 Register Call Summary for Register ULPI_UTMI_VCONTROL_EN_i 3280
24-124 ULPI_UTMI_VCONTROL_EN_SET_i 3280
24-125 Register Call Summary for Register ULPI_UTMI_VCONTROL_EN_SET_i 3281
24-126 ULPI_UTMI_VCONTROL_EN_CLR_i 3281
24-127 Register Call Summary for Register ULPI_UTMI_VCONTROL_EN_CLR_i 3282
24-128 ULPI_UTMI_VCONTROL_STATUS_i 3282
24-129 Register Call Summary for Register ULPI_UTMI_VCONTROL_STATUS_i 3282
24-130 ULPI_UTMI_VCONTROL_LATCH_i 3282
24-131 Register Call Summary for Register ULPI_UTMI_VCONTROL_LATCH_i 3283
24-132 ULPI_UTMI_VSTATUS_i 3283
24-133 Register Call Summary for Register ULPI_UTMI_VSTATUS_i 3283
24-134 ULPI_UTMI_VSTATUS_SET_i 3283
24-135 Register Call Summary for Register ULPI_UTMI_VSTATUS_SET_i 3283
24-136 ULPI_UTMI_VSTATUS_CLR_i 3284
24-137 Register Call Summary for Register ULPI_UTMI_VSTATUS_CLR_i 3284
24-138 ULPI_USB_INT_LATCH_NOCLR_i 3284
24-139 Register Call Summary for Register ULPI_USB_INT_LATCH_NOCLR_i 3285
24-140 ULPI_VENDOR_INT_EN_i 3285
24-141 Register Call Summary for Register ULPI_VENDOR_INT_EN_i 3285
24-142 ULPI_VENDOR_INT_EN_SET_i 3285
24-143 Register Call Summary for Register ULPI_VENDOR_INT_EN_SET_i 3285
24-144 ULPI_VENDOR_INT_EN_CLR_i 3286
24-145 Register Call Summary for Register ULPI_VENDOR_INT_EN_CLR_i 3286
24-146 ULPI_VENDOR_INT_STATUS_i 3286
24-147 Register Call Summary for Register ULPI_VENDOR_INT_STATUS_i 3286
24-148 ULPI_VENDOR_INT_LATCH_i 3287
24-149 Register Call Summary for Register ULPI_VENDOR_INT_LATCH_i 3287
24-150 UHH_REVISION 3287
24-151 Register Call Summary for Register UHH_REVISION 3287
24-152 UHH_SYSCONFIG 3288
24-153 Register Call Summary for Register UHH_SYSCONFIG 3288
24-154 UHH_SYSSTATUS 3289
24-155 Register Call Summary for Register UHH_SYSSTATUS 3289
24-156 UHH_HOSTCONFIG 3289
24-157 Register Call Summary for Register UHH_HOSTCONFIG 3290
24-158 UHH_DEBUG_CSR 3291
24-159 Register Call Summary for Register UHH_DEBUG_CSR 3291
24-160 HCREVISION 3292
24-161 Register Call Summary for Register HCREVISION 3292
24-162 HCCONTROL 3292
24-163 Register Call Summary for Register HCCONTROL 3293
24-164 HCCOMMANDSTATUS 3293
24-165 Register Call Summary for Register HCCOMMANDSTATUS 3294

24-166 HCINTERRUPTSTATUS 3294
24-167 Register Call Summary for Register HCINTERRUPTSTATUS 3295
24-168 HCINTERRUPTENABLE 3295
24-169 Register Call Summary for Register HCINTERRUPTENABLE 3296
24-170 HCINTERRUPTDISABLE 3296
24-171 Register Call Summary for Register HCINTERRUPTDISABLE 3297
24-172 HCHCCA 3297
24-173 Register Call Summary for Register HCHCCA 3297
24-174 HCPERIODCURRENTED 3297
24-175 Register Call Summary for Register HCPERIODCURRENTED 3298
24-176 HCCONTROLHEADED 3298
24-177 Register Call Summary for Register HCCONTROLHEADED 3298
24-178 HCCONTROLCURRENTED 3298
24-179 Register Call Summary for Register HCCONTROLCURRENTED 3298
24-180 HCBULKHEADED 3299
24-181 Register Call Summary for Register HCBULKHEADED 3299
24-182 HCBULKCURRENTED 3299
24-183 Register Call Summary for Register HCBULKCURRENTED 3299
24-184 HCDONEHEAD 3299
24-185 Register Call Summary for Register HCDONEHEAD 3300
24-186 HCFMINTERVAL 3300
24-187 Register Call Summary for Register HCFMINTERVAL 3300
24-188 HCFMREMAINING 3300
24-189 Register Call Summary for Register HCFMREMAINING 3301
24-190 HCFMNUMBER 3301
24-191 Register Call Summary for Register HCFMNUMBER 3301
24-192 HCPERIODICSTART 3301
24-193 Register Call Summary for Register HCPERIODICSTART 3301
24-194 HCLSTHRESHOLD 3302
24-195 Register Call Summary for Register HCLSTHRESHOLD 3302
24-196 HCRHDESCRIPTORA 3302
24-197 Register Call Summary for Register HCRHDESCRIPTORA 3303
24-198 HCRHDESCRIPTORB 3303
24-199 Register Call Summary for Register HCRHDESCRIPTORB 3303
24-200 HCRHSTATUS 3303
24-201 Register Call Summary for Register HCRHSTATUS 3304
24-202 HCRHPORTSTATUS_1 3304
24-203 Register Call Summary for Register HCRHPORTSTATUS_1 3305
24-204 HCRHPORTSTATUS_2 3306
24-205 Register Call Summary for Register HCRHPORTSTATUS_2 3307
24-206 HCRHPORTSTATUS_3 3307
24-207 Register Call Summary for Register HCRHPORTSTATUS_3 3308
24-208 HCCAPBASE 3309
24-209 Register Call Summary for Register HCCAPBASE 3309
24-210 HCSPARAMS 3309
24-211 Register Call Summary for Register HCSPARAMS 3310
24-212 HCCPARAMS 3310
24-213 Register Call Summary for Register HCCPARAMS 3311
24-214 USBCMD 3311
24-215 Register Call Summary for Register USBCMD 3313
24-216 USBSTS 3313

24-217 Register Call Summary for Register USBSTS 3314
24-218 USBINTR 3314
24-219 Register Call Summary for Register USBINTR 3315
24-220 FRINDEX 3315
24-221 Register Call Summary for Register FRINDEX 3315
24-222 CTRLDSSEGMENT 3315
24-223 Register Call Summary for Register CTRLDSSEGMENT 3315
24-224 PERIODICLISTBASE 3316
24-225 Register Call Summary for Register PERIODICLISTBASE 3316
24-226 ASYNCLISTADDR 3316
24-227 Register Call Summary for Register ASYNCLISTADDR 3316
24-228 CONFIGFLAG 3316
24-229 Register Call Summary for Register CONFIGFLAG 3317
24-230 PORTSC_i 3317
24-231 Register Call Summary for Register PORTSC_i 3319
24-232 INSNREG00 3319
24-233 Register Call Summary for Register INSNREG00 3319
24-234 INSNREG01 3320
24-235 Register Call Summary for Register INSNREG01 3320
24-236 INSNREG02 3320
24-237 Register Call Summary for Register INSNREG02 3320
24-238 INSNREG03 3320
24-239 Register Call Summary for Register INSNREG03 3321
24-240 INSNREG04 3321
24-241 Register Call Summary for Register INSNREG04 3321
24-242 INSNREG05_UTMI 3321
24-243 Register Call Summary for Register INSNREG05_UTMI 3322
24-244 INSNREG05_ULPI 3322
24-245 Register Call Summary for Register INSNREG05_ULPI 3323
25-1 General-Purpose Interface Functional Pins Description 3330
25-2 Clocks 3332
25-3 Interrupts 3335
25-4 Wake-Up Signals 3336
25-5 GPIO Channel Description 3336
25-6 Instance Summary 3348
25-7 GPIO1 to GPIO3 Register Summary 3348
25-8 GPIO4 to GPIO6 Register Summary 3349
25-9 GPIO_REVISION 3350
25-10 Register Call Summary for Register GPIO_REVISION 3350
25-11 GPIO_SYSCONFIG 3350
25-12 Register Call Summary for Register GPIO_SYSCONFIG 3351
25-13 GPIO_SYSSTATUS 3351
25-14 Register Call Summary for Register GPIO_SYSSTATUS 3352
25-15 GPIO_IRQSTATUS1 3352
25-16 Register Call Summary for Register GPIO_IRQSTATUS1 3352
25-17 GPIO_IRQENABLE1 3353
25-18 Register Call Summary for Register GPIO_IRQENABLE1 3353
25-19 GPIO_WAKEUPENABLE 3354
25-20 Register Call Summary for Register GPIO_WAKEUPENABLE 3354
25-21 GPIO_IRQSTATUS2 3354
25-22 Register Call Summary for Register GPIO_IRQSTATUS2 3355

25-23 GPIO_IRQENABLE2 ... 3355
25-24 Register Call Summary for Register GPIO_IRQENABLE2 ... 3355
25-25 GPIO_CTRL ... 3356
25-26 Register Call Summary for Register GPIO_CTRL ... 3356
25-27 GPIO_OE ... 3357
25-28 Register Call Summary for Register GPIO_OE ... 3357
25-29 GPIO_DATAIN ... 3357
25-30 Register Call Summary for Register GPIO_DATAIN ... 3357
25-31 GPIO_DATAOUT ... 3358
25-32 Register Call Summary for Register GPIO_DATAOUT ... 3358
25-33 GPIO_LEVELDETECT0 ... 3359
25-34 Register Call Summary for Register GPIO_LEVELDETECT0 ... 3359
25-35 GPIO_LEVELDETECT1 ... 3359
25-36 Register Call Summary for Register GPIO_LEVELDETECT1 ... 3359
25-37 GPIO_RISINGDETECT ... 3360
25-38 Register Call Summary for Register GPIO_RISINGDETECT ... 3360
25-39 GPIO_FALLINGDETECT ... 3360
25-40 Register Call Summary for Register GPIO_FALLINGDETECT ... 3361
25-41 GPIO_DEBOUNCENABLE ... 3361
25-42 Register Call Summary for Register GPIO_DEBOUNCENABLE ... 3361
25-43 GPIO_DEBOUNCINGTIME ... 3361
25-44 Register Call Summary for Register GPIO_DEBOUNCINGTIME ... 3362
25-45 GPIO_CLEARIRQENABLE1 ... 3362
25-46 Register Call Summary for Register GPIO_CLEARIRQENABLE1 ... 3362
25-47 GPIO_SETIRQENABLE1 ... 3363
25-48 Register Call Summary for Register GPIO_SETIRQENABLE1 ... 3363
25-49 GPIO_CLEARIRQENABLE2 ... 3363
25-50 Register Call Summary for Register GPIO_CLEARIRQENABLE2 ... 3363
25-51 GPIO_SETIRQENABLE2 ... 3364
25-52 Register Call Summary for Register GPIO_SETIRQENABLE2 ... 3364
25-53 GPIO_CLEARWKUENA ... 3364
25-54 Register Call Summary for Register GPIO_CLEARWKUENA ... 3364
25-55 GPIO_SETWKUENA ... 3365
25-56 Register Call Summary for Register GPIO_SETWKUENA ... 3365
25-57 GPIO_CLEARDATAOUT ... 3365
25-58 Register Call Summary for Register GPIO_CLEARDATAOUT ... 3366
25-59 GPIO_SETDATAOUT ... 3366
25-60 Register Call Summary for Register GPIO_SETDATAOUT ... 3366
26-1 Power Pins ... 3370
26-2 Mapping for Input Sources ... 3373
26-3 Memory Booting Configuration Pins After POR ... 3375
26-4 Peripheral Booting Configuration Pins After POR ... 3376
26-5 Booting Configuration Pins After a Warm Reset ... 3377
26-6 Pin Multiplexing According to Boot Peripheral ... 3379
26-7 ROM Exception Vectors ... 3381
26-8 Dead Loops ... 3382
26-9 Tracing Data ... 3383
26-10 RAM Exception Vectors ... 3384
26-11 ROM Code Default Clock Settings ... 3387
26-12 Software Booting Configuration Structure ... 3388
26-13 ASIC ID Structure ... 3390

26-14 Boot Messages 3391
26-15 Device Descriptor 3393
26-16 Device-Qualifier Descriptor 3394
26-17 Configuration Descriptor 3394
26-18 Other Speed Configuration Descriptor 3395
26-19 Interface Descriptor 3395
26-20 BULK IN Endpoint Descriptor 3395
26-21 BULK OUT Endpoint Descriptor 3395
26-22 Language ID String Descriptor 3396
26-23 Manufacturer ID String Descriptor 3396
26-24 Product ID String Descriptor 3396
26-25 Configuration String Descriptor 3396
26-26 Interface String Descriptor 3396
26-27 Customized Descriptor Parameters 3397
26-28 Standard Device Requests Supported 3398
26-29 Blocks and Sectors Searched on Non-XIP Memories 3401
26-30 XIP Timing Parameters 3402
26-31 NAND Timing Parameters 3403
26-32 Supported NAND Devices 3404
26-33 Fourth NAND ID Data Byte 3405
26-34 ID2 Byte Description 3409
26-35 Bad Block Mark Locations in NAND Spare Areas 3410
26-36 Master Boot Record Structure 3420
26-37 Partition Table Entry 3420
26-38 FAT Directory Entry 3423
26-39 FAT Entry Description 3424
26-40 CH TOC Item 3426
26-41 CHSETTINGS 3426
26-42 CHRAM 3427
26-43 CHFLASH 3429
26-44 CHMMCSD CH 3429
26-45 GP Device Software Image 3430
26-46 Booting Parameter Structure 3430
26-47 Tracing Vector 3431
26-48 CONTROL_SAVE_RESTORE_MEM Field Definitions 3434
26-49 PRCM Register Organization in the SCM Block 3435
26-50 SDRC Register Organization in the SCM Block 3435
26-51 Debug POR Signals 3436
26-52 Debugger Address Space 3438

Read This First

Survey

Help us meet your expectations:

We are always looking at ways to develop our service and improve our quality to fit your needs. Please take a few minutes to complete our short questionnaire by:

- Providing general suggestions
  or
- Rating a document and describing its critical points

http://www.ti.com/csdocsurvey (password: 123survey)

Thank you.

If You Need Assistance. . .

If you want to . . .	Do this . . .
Request more information about Texas Instruments Digital Signal Processing (DSP) products	Call the CRC(1) hotline: **(800) 336–5236** Or write to: Texas Instruments Incorporated Market Communications Manager, MS 736 P.O. Box 1443 Houston, Texas 77251–1443
Order Texas Instruments documentation	Call the CRC(1) hotline: **(800) 336–5236**

(1) Texas Instruments Customer Response Center

About This Manual

FCC Warning

This equipment is intended for use in a laboratory test environment only. It generates, uses, and can radiate radio frequency energy and has not been tested for compliance with the limits of computing devices pursuant to subpart J of part 15 of FCC rules, which are designed to provide reasonable protection against radio frequency interference. Operation of this equipment in other environments may cause interference with radio communications, in which case the user at his own expense will be required to take whatever measures may be required to correct this interference.

Information About Cautions and Warnings

This book may contain cautions and warnings.

CAUTION

This is an example of a caution statement.

A caution statement describes a situation that could potentially damage your software or equipment.

WARNING

This is an example of a warning statement.

A warning statement describes a situation that could potentially cause harm to you.

The information in a caution or a warning is provided for your protection. Please read each caution and warning carefully.

Register, Field, and Bit Calls

The naming convention applied for a call consists of:

- For a register call: *<Module name>.<Register name>*; for example: UART.UASR
- For a bit field call:
  - *<Module name>.<Register name>[End:Start] <Field name> field*; for example, UART.UASR[4:0] SPEED bit field
  - *<Field name> field <Module name>.<Register name>[End:Start]*; for example, SPEED bit field UART.UASR[4:0]
- For a bit call:
  - *<Module name>.<Register name>[pos] <Bit name> bit*; for example, UART.UASR[5] BIT_BY_CHAR bit
  - *<Bit name> bit <Module name>.<Register name>[pos]*; for example, BIT_BY_CHAR bit UART.UASR[5]

To help the reader navigate the document, each register call is hyperlinked to its register description in the register manual section. After each register description, a table summarizes all hyperlinked register calls.

To navigate in the PDF documents, see Acrobat Reader Tips.

Coding Rules

The programming models or code listings follow the rules:

Type	Definition	Example
File	Starts with the module name	PRCM_test1.c MCBSP1_init.h
Variable	Global variables are prefixed by "g_" Pointers are prefixed by "p" Global pointers are prefixed by "g_p"	g_SDMA_LogicalChan pAddrCounter g_pSDMA_LogicalChan
Function	Starts with the module name	PRCM_SetupClocks() ArmIntC_MaskInterrupts()
Typedef	Ends with "_t"	PRCM_Struct_t
Definition	Starts with the module name and is followed by the register name	#define SMS_ERR_TYPE *((volatileUint32*)0x680080F4) #define MCBSP2_RCR1_REG *((volatile Uint32*)0x4807401C)
Enumeration	Starta with the module name	Typedef enum DMA_Mode_Label { INPUT_MODE OUTPUT_MODE } DMA_Mode_t;

Flow Chart Rules

Flow charts follow the following rules:

Shape	Name	Definition
	Process	Any computational steps or processing function of a program; defined operation(s) causing change in value, form, or location of information
	Decision	A decision or switching type operation that determines which of a number of alternate paths is followed
	Predefined process or sub-process	One or more named operations or program steps specified in a subroutine or another set of flow charts
	Data or I/O	General I/O function; information available for processing (input) or recording of processed information (output)
	Terminator	Terminal point in a flow chart: start, stop, halt, delay, or interrupt; may show exit from a closed subroutine
	Annotation	Additional descriptive clarification, comment
	On page connector (reference)	Exit to, or entry from, another part of chart in the same page
	Off page connector (reference)	The flow continues on a different page.
	Summing Junction	Logical AND
	Or	Logical OR
	Parallel mode (ISO)	Beginning or end of two or more simultaneous operations
	Flow Line	Lines indicate the sequence of steps and the direction of flow.

Acrobat Reader Tips

Acrobat includes two methods to search for words in a PDF:

- The Find toolbar provides a basic set of options to locate a word in the current PDF.
- The Search window lista words or partial words that match your text in the current PDF.

These guidelines apply to Acrobat Reader 5.x, 6.0, and 7.0.

For more information on Acrobat Reader search features, see the Adobe Reader Help.

To search for words in a document using the Find dialog box:

1. Open the document.
2. To display the Find toolbar, right-click in the toolbar area and select Find.
3. In the Find box, type the word, words, or partial words for which you want to search.
4. From the Find Options menu, select options as desired.
5. To view each search result, click the Find toolbar, the Find Previous button, or the Find Next button to go backward or forward through the document.

To search for words in a document using the Search PDF window:

1. Open the document.
2. Click the Search button on the File toolbar or right-click on your document and select Search.
3. Type the word, words, or part of a word for which you want to search.
4. Click Search.
5. The results appear in page order and, if applicable, show a few words of context. Each result displays an icon to identify the type of occurrence. All other searchable areas display the Search Result icon.
6. To display the page that contains a search result, click an item in the Results list. The occurrence is highlighted.
7. To navigate to the next result, choose Edit > Search Results > Next Result (or Ctrl+G).
8. To navigate to the previous result, choose Edit > Search Results > Previous Result (or Shift+Ctrl+G).

Navigate through your previous view

To retrace your path within an Adobe PDF document:

- For the previous view: Choose View > Go To > Previous View or Alt+Left Arrow.
- For the next view: Choose View > Go To > Next View or Alt+Right Arrow. The Next View command is available only if you have chosen Previous View.

  If you view the PDF document in a browser, use options on the Navigation toolbar to move between views.

  - Right-click the toolbar area, and then choose *Navigation*.
  - Click the Go To Previous View button or the Go To Next View button.

Note: This navigation tip is useful to return to your previous view after clicking on a register call hyperlink.

OMAP34xx Disclaimer

All programming models and use cases presented in this manual are provided for educative purposes only and may differ from or be optimized for your applications.

All OMAP peripheral devices presented in this manual are provided for illustration purposes and may be different from those in your system.

Trademarks

OMAP, TMS320DMC64x, C64x, M-Shield and FlatLink3G are trademarks of Texas Instruments Incorporated.

ARM, CORTEX, JAZELLE, and THUMB are registered trademarks of ARM Limited.

ETM, ETB, ARM9, CoreSight and Neon are trademarks of ARM Limited.

Bluetooth is a registered trademark of Bluetooth SIG, Inc. and is licensed to Texas Instruments.

Memory Stick is a registered trademark of Sony Corporation, and Memory Stick PRO is a trademark of Sony Corporation.

HDQ is a trademark of Benchmarq.

1-Wire is a registered trademark of Dallas Semiconductor.

Windows is a trademark of Microsoft Corporation in the United States and other countries.

USSE and POWERVR are trademarks or registered trademarks of Imagination Technologies Ltd.

Mentor Graphics is a registered trademark of Mentor Graphics Corporation or its affiliated companies in the United States and other countries.

SonicsMX, Sonics3220 are trademarks or registered trademarks of Sonics, Inc.

Foveon X3 is a registered trademarks of Foveon, Inc.

Super CCD Honeycom is a registered trademark of Fuji Photo Film Co., Ltd.

Linux is a registered trademark of Linus Torvalds.

Symbian and all Symbian based trademarks and logos are trademarks of Symbian Software Limited.

Synopsys is a registered trademarks of Synopsys, Inc.

MIPI is a registered trademark of the Mobile Industry Processor Interface (MIPI) Alliance.

OneNAND is a trademark of SAMSUNG.

All other trademarks are the property of their respective owners.

History

The following table summarizes the Pubic OMAP34xx TRM versions.

Version	Literature Number	Date	Notes
G	SWPU114	March 2008	See (1)
H	SWPU114	April 2008	See (2)
I	SWPU114	June 2008	See (3)
J	SWPU114	July 2008	See (4)
K	SWPU114	August 2008	See (5)

(1) Public Version of *OMAP34xx Technical Reference Manual* - version G (SWPU114G) initial release.

(2) Public Version of *OMAP34xx Technical Reference Manual* - version H (SWPU114H)
- Chapter 4: Power, Reset, and Clock Management
- Chapter 5: Interconnect
- Chapter 7: System Control Module
- Chapter 8: Memory Management Units
- Chapter 11: Memory Subsystem
- Chapter 12: Camera ISP
- Chapter 14: IVA2.2 Subsystem
- Chapter 15: Display Subsystem
- Chapter 19: Multichannel SPI
- Chapter 20: HDQ/1-Wire
- Chapter 22: MMC/SD/SDIO Card Interface
- Chapter 24: USB
- Chapter 26: Initialization

(3) Public Version of *OMAP34xx Technical Reference Manual Silicon Revision 3.0* - version I (SWPU114I)
- Chapter 4: Power Reset, and Clock Management
- Chapter 5: Interconnect
- Chapter 7: System Control Module
- Chapter 9: Direct Management Access
- Chapter 10: Interrupt Controller
- Chapter 11: Memory Subsystem
- Chapter 12: Camera ISP
- Chapter 13: 2D/3D Graphics Accelerator
- Chapter 14: IVA2.2 Subsystem
- Chapter 15: Display Subsystem
- Chapter 17: UART/IrDA/CIR
- Chapter 21: Multi-Channel Buffered Serial Port
- Chapter 24: High-Speed USB Host Subsystem and High-Speed USG OTG Controller
- Chapter 25: General-Purpose Interface

(4) Public Version of *OMAP34xx Technical Reference Manual Silicon Revision 3.0* - version J (SWPU114J)
- Chapter 4: Power Reset, and Clock Management
- Chapter 7: System Control Module
- Chapter 9: Direct Management Access
- Chapter 11: Memory Subsystem
- Chapter 12: Camera ISP
- Chapter 15: Display Subsystem
- Chapter 18: I2C
- Chapter 21: Multi-Channel Buffered Serial Port
- Chapter 22: MMC/SD/SDIO Card Interface
- Chapter 24: High-Speed USB Host Subsystem and High-Speed USG OTG Controller
- Chapter 26: Initialization

(5) Public Version of *OMAP34xx Technical Reference Manual Silicon Revision 3.0* - version K (SWPU114K)
- Chapter 1: Introduction
- Chapter 4: Power Reset and Clock Management
- Chapter 6: Interprocessor Communication
- Chapter 7: System Control Module
- Chapter 9: DMA
- Chapter 10: Interrupt Controller
- Chapter 11: Memory Subsystem
- Chapter 12: Camera ISP
- Chapter 14: IVA2.2 Subsystem
- Chapter 15: Display Subsystem
- Chapter 17: UART/IrDA/CIR
- Chapter 18: I^2C
- Chapter 21: McBSP
- Chapter 22: MMC/SD/SDIO Card Interface
- Chapter 24: High-Speed USB Host Subsystem and High-Speed USB OTG Controller
- Chapter 25: General-Purpose Interface
- Chapter 26: Initialization

Version	Literature Number	Date	Notes
L	SWPU114	September 2008	See (6)
M	SWPU114	October 2008	See (7)
N	SWPU114	November 2008	See (8)
O	SWPU114	January 2009	See (9)
P	SWPU114	February 2009	See (10)
Q	SWPU114	March 2009	See (11)

(6) Public Version of *OMAP34xx Technical Reference Manual Silicon Revision 3.0* - version L (SWPU114L)
- Chapter 4: Power Reset and Clock Management
- Chapter 5: Interconnect
- Chapter 7: System Control Module
- Chapter 9: DMA
- Chapter 10: Interrupt Controller
- Chapter 11: Memory Subsystem
- Chapter 15: Display Subsystem
- Chapter 16: Timers
- Chapter 17: UART/IrDA/CIR
- Chapter 18: I^2C
- Chapter 19: Multichannel SPI
- Chapter 20: HDQ/1-Wire
- Chapter 21: McBSP
- Chapter 22: MMC/SD/SDIO Card Interface
- Chapter 24: High-Speed USB Host Subsystem and High-Speed USB OTG Controller
- Chapter 26: Initialization

(7) Public Version of *OMAP34xx Technical Reference Manual* - version M (SWPU114M)
- Chapter 4: Power, Reset, and Clock Management
- Chapter 7: System Control Module
- Chapter 10: Interrupt Controller
- Chapter 11: Memory Subsystem
- Chapter 12: Camera ISP
- Chapter 14: IVA2.2 Subsystem
- Chapter 15: Display Subsystem
- Chapter 17: UART/IrDA/CIR
- Chapter 20: HDQ/1-Wire
- Chapter 26: Initialization

(8) Public Version of *OMAP34xx Technical Reference Manual* - version N (SWPU114N)
- Chapter 1: Introduction
- Chapter 4: Power Reset and Clock Management
- Chapter 7: System Control Module
- Chapter 12: Camera ISP
- Chapter 14: IVA2.2 Subsystem
- Chapter 15: Display Subsystem
- Chapter 18: I^2C
- Chapter 21: McBSP
- Chapter 24: High-Speed USB Host Subsystem and High-Speed USB OTG Controller
- Chapter 26: General-Purpose Interface

(9) Public Version of *OMAP34xx Technical Reference Manual* - version O (SWPU114O).
- Chapter 4: Power Reset and Clock Management
- Chapter 7: System Control Module
- Chapter 12: Camera ISP
- Chapter 15: Display Subsystem
- Chapter 16: Timers
- Chapter 18: I^2C
- Chapter 22: MMC/SD/SDIO Card Interface
- Chapter 26: Initialization
- Appendix A: Glossary

(10) Public Version of *OMAP34xx Technical Reference Manual* - version P (SWPU114P).
- Chapter 4: Power Reset and Clock Management
- Chapter 7: System Control Module
- Chapter 11: Memory Subsystem
- Chapter 12: Camera ISP
- Chapter 21: McBSP

(11) Public Version of *OMAP34xx Technical Reference Manual* - version Q (SWPU114Q).
- Chapter 1: Introduction
- Chapter 3: MPU Subsystem
- Chapter 11: Memory Subsystem
- Chapter 12: Camera ISP
- Chapter 15: Display Subsystem
- Chapter 21: McBSP

Version	Literature Number	Date	Notes
R	SWPU114	April 2009	See (12)
S	SWPU114	May 2009	See (13)
T	SWPU114	June 2009	See (14)
U	SWPU114	August 2009	See (15)
V	SWPU114	September 2009	See (16)
W	SWPU114	September 2009	See (17)

(12) Public Version of *OMAP34xx Technical Reference Manual* - version R (SWPU114R).
- Chapter 1: Introduction
- Chapter 4: Power, Reset, and Clock Managment
- Chapter 12: Camera ISP
- Chapter 16: Timers
- Chapter 18: I^2C
- Chapter 21: McBSP
- Chapter 26: Initialization

(13) Public Version of *OMAP34xx Technical Reference Manual* - version S (SWPU114S).
- Chapter 4: Power, Reset, and Clock Managment
- Chapter 7: System Control Module
- Chapter 12: Camera ISP
- Chapter 15: Display Subsystem
- Chapter 17: UART/IrDA/CIR
- Chapter 18: I^2C
- Chapter 19: Multichannel SPI
- Chapter 21: McBSP
- Chapter 24: High-Speed USB Host Subsystem and High-Speed USB OTG Controller

(14) Public Version of *OMAP34xx Technical Reference Manual* - version T (SWPU114T)
- Chapter 1: Introduction
- Chapter 4: Power, Reset, and Clock Management
- Chapter 5: Interconnect
- Chapter 7: System Control Module
- Chapter 9: Direct Memory Access
- Chapter 11: Memory Subsystem
- Chapter 12: Camera Subsystem
- Chapter 13: 2D/3D Graphics Accelerator
- Chapter 15: Display Subsystem
- Chapter 18: I^2C
- Chapter 19: Multichannel SPI
- Chapter 21: Multichannel Buffered Serial Port
- Chapter 22: MMC/SD/SDIO Card Interface
- Appendix A: Glossary

(15) Public Version of *OMAP34xx Technical Reference Manual* - version U (SWPU114U)
- Chapter 1: Introduction
- Chapter 4: Power, Reset, and Clock Management
- Chapter 7: System Control Module
- Chapter 8: Memory Management Units
- Chapter 11: Memory Subsystem
- Chapter 12: Camera Subsystem
- Chapter 14: IVA2.2 Subsystem
- Chapter 15: Display Subsystem
- Chapter 19: Multichannel SPI
- Chapter 26: Initialization

(16) Public Version of *OMAP34xx Technical Reference Manual* - version V (SWPU114V)
- Chapter 4: Power, Reset, and Clock Management
- Chapter 7: System Control Module
- Chapter 11: Memory Subsystem
- Chapter 12: Camera Subsystem
- Chapter 15: Display Subsystem
- Chapter 17: UART/IrDA/CIR
- Chapter 19: Multichannel SPI
- Chapter 24: High-Speed USB Host Subsystem and High-Speed USB OTG Controller

(17) Public Version of *OMAP34xx Technical Reference Manual* - version W (SWPU114W)
- Chapter 1: Introduction
- Chapter 3: MPU Subsystem
- Chapter 4: Power, Reset, and Clock Management
- Chapter 12: Camera Subsystem
- Chapter 15: Display Subsystem
- Chapter 24: High-Speed USB Host Subsystem and High-Speed USB OTG Controller
- Chapter 27: Debug and Emulation

Version	Literature Number	Date	Notes
X	SWPU114	November 2009	See (18)
Y	SWPU114	December 2009	See (19)
Z	SWPU114	January 2010	See (20)
*	SWPU223	February 2010	See (21)

(18) Public Version of *OMAP34xx Technical Reference Manual* - version X (SWPU114X)
- Chapter 3: MPU Subsystem
- Chapter 4: Power, Reset, and Clock Management
- Chapter 11: Memory Subsystem
- Chapter 12: Camera Subsystem
- Chapter 26: Initialization

(19) Public Version of *OMAP34xx Technical Reference Manual* - version Y (SWPU114Y)
- Chapter 1: Introduction
- Chapter 4: Power, Reset, and Clock Management
- Chapter 5: Interconnect
- Chapter 7: System Control Module
- Chapter 9: Direct Memory Access
- Chapter 12: Camera Subsystem
- Chapter 13: 2D/3D Graphics Accelerator
- Chapter 18: I^2C
- Chapter 24: High-Speed USB Host Subsystem and High-Speed USB OTG Controller
- Chapter 26: Initialization

(20) Public Version of *OMAP34xx Technical Reference Manual* - version Z (SWPU114Z)
- Chapter 4: Power, Reset, and Clock Management
- Chapter 5: Interconnect
- Chapter 7: System Control Module
- Chapter 12: Camera Subsystem
- Chapter 15: Display Subsystem
- Chapter 21:Multichannel Buffered Serial Port
- Chapter 26: Initialization

(21) Public Version of *OMAP34xx Technical Reference Manual* - version * (SWPU223).
- Chapter 4: Power, Reset, and Clock Management
- Chapter 11: Memory Subsystem
- Chapter 12: Camera Image Signal Processor
- Chapter 14: IVA2.2 Subsystem
- Chapter 15: Display Subsystem
- Chapter 17: UART/IrDA/CIR
- Chapter 26: Initialization

Introduction

This chapter introduces the features, supporting subsystems, and architecture of the OMAP34xx high-performance multimedia devices.

Note: This document is strictly for wireless/cellular software developers using OMAP34xx application processors, which are not available for the broad market through authorized distributors.

The OMAP35x application processors should be used for all nonhandset/cellular applications. For more information, see www.ti.com/omap35x

Topic **Page**

1.1 Overview 176
1.2 Environment 177
1.3 Description 179
1.4 Package-On-Package Concept 184
1.5 OMAP34xx Family 186

1.1 Overview

The OMAP34xx high-performance, multimedia application device is based on the enhanced OMAP™ 3 architecture and is integrated on TI's advanced 65-nm process technology.

The OMAP3 architecture is configured with different sets of features in different tier devices. This TRM details all of the features available in the high-tier devices, that is, OMAP3440 and 3430 devices. Some features are not available in the lower-tier devices. For more information, see Section 1.5, *OMAP34xx Family*.

The architecture is designed to provide best-in-class video, image, and graphics processing sufficient to support the following:

- Streaming video
- 2D/3D mobile gaming
- Video conferencing
- High-resolution still image
- Video capture in 2.5G wireless terminals, 3G wireless terminals, and rich multimedia-featured handsets, and high-performance personal digital assistants (PDAs)

The device supports high-level operating systems (OSs), such as:

- Windows CE
- Symbian OS™
- Linux

This OMAP device includes state-of-the-art power-management techniques required for high-performance mobile products.

The following subsystems are part of the device:

- Microprocessor unit (MPU) subsystem based on the public ARM Cortex™-A8 microprocessor
- IVA2.2 subsystem with a C64X digital signal processor (DSP) core
- POWERVR™ SGX530 subsystem for 2D and 3D graphics acceleration to support display and gaming effects
- Camera image signal processor (ISP) that supports multiple formats and interfacing options connected to a wide variety of image sensors
- Display subsystem with a wide variety of features for multiple concurrent image manipulation, and a programmable interface supporting a wide variety of displays. The display subsystem also supports NTSC/PAL video out.
- Level 3 (L3) and level 4 (L4) interconnects that provide high-bandwidth data transfers for multiple initiators to the internal and external memory controllers and to on-chip peripherals

The device also offers:

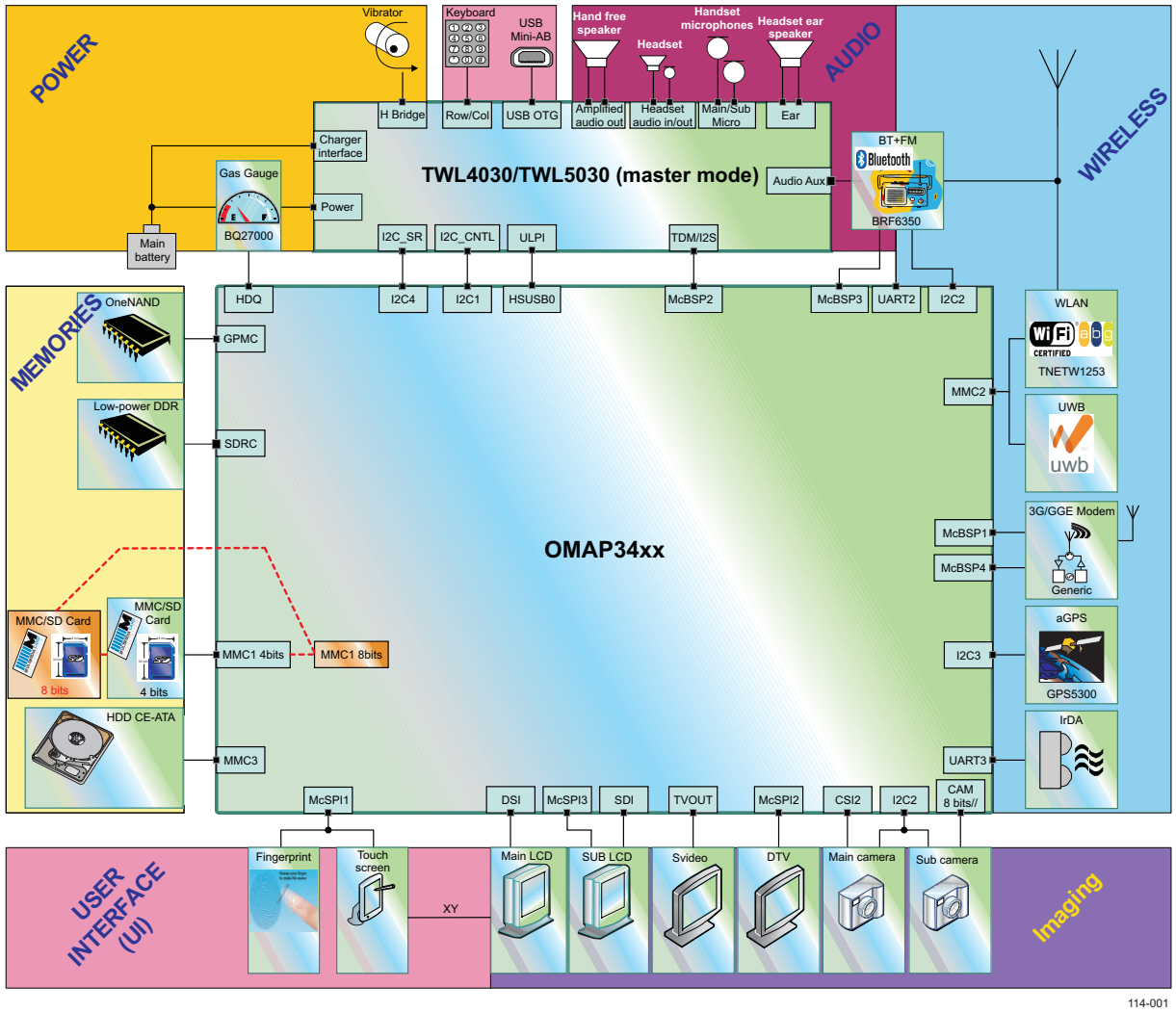
- A comprehensive power and clock-management scheme that enables high-performance, low-power operation, and ultralow-power standby features.
- Connectivity for TI 2.5G or 3G modem chipsets for a complete terminal solution
- Memory stacking feature using the package-on-package (POP) implementation (see Section 1.4, *Package-on-Package Concept*)

1.2 Environment

This section provides an overview of the OMAP environment. The device is associated with a power integrated circuit (IC). Texas Instruments provides a global solution with the TWL4030 or TWL5030 devices.

Figure 1-1 provides an overview of a nonexhaustive environment for the high-tier OMAP34xx device.

Figure 1-1. OMAP34xx High-Tier Environment



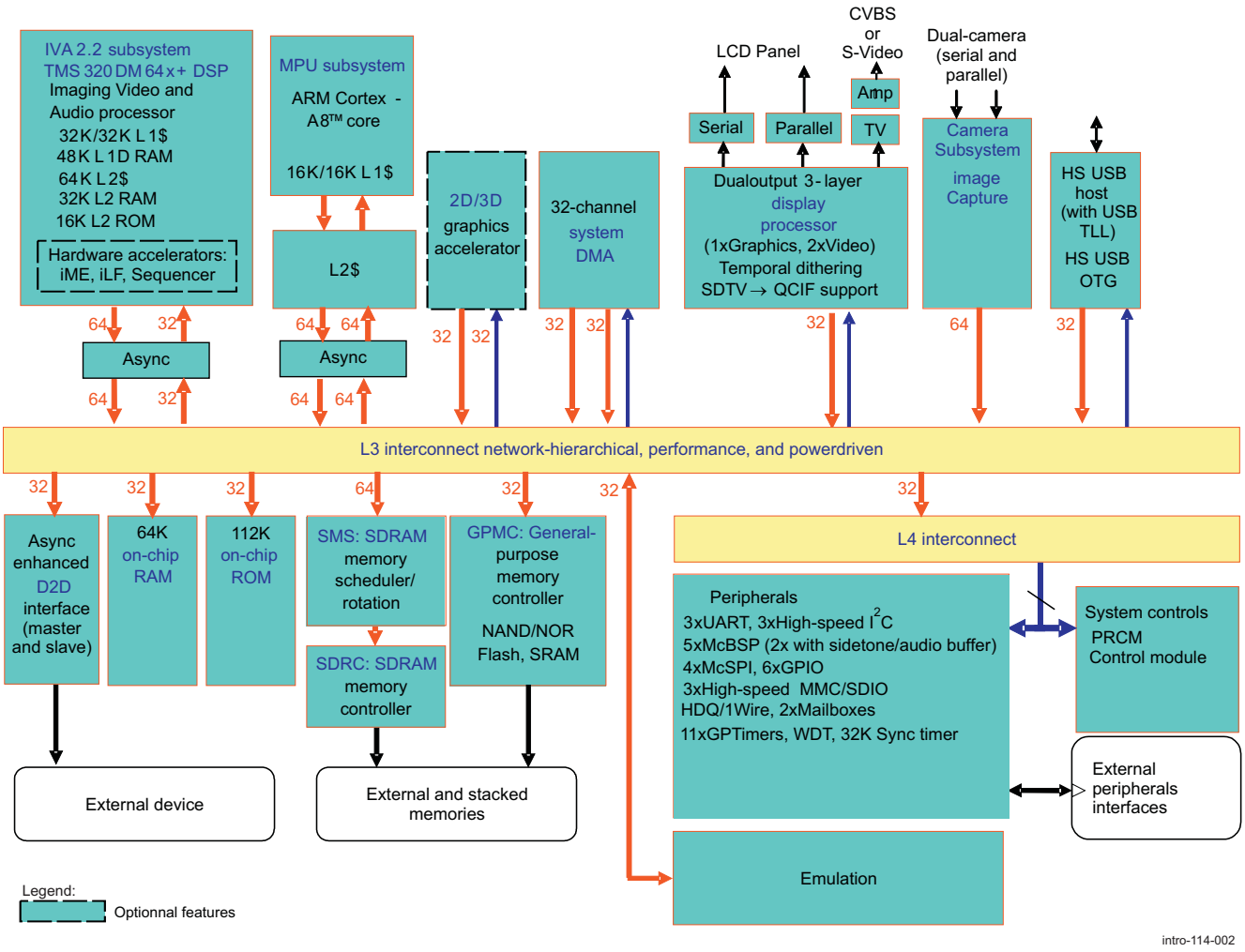
Note: Some features are not available on lower-tier devices. See Section 1.5, *OMAP34xx Family*, for more information on available features.

1.3 Description

The device is offered in the 515-ball, 12 x 12 mm, POP 0.5-mm (top) and 0.4-mm (bottom) ball pitch package. Some balls are available at the top of the device to allow memory stacking. (For more information, see Section 1.4, *Package-on-Package Concept*.)

Figure 1-2 shows the block diagram.

Figure 1-2. OMAP34xx Block Diagram



Note: Some features are not available on lower-tier devices. See Section 1.5, *OMAP34xx Family*, for more information on available features.

1.3.1 MPU Subsystem

The MPU subsystem integrates the following modules

- ARM subchip
  - Public ARM Cortex-A8 core
  - ARM Version 7 ISA™: Standard ARM instruction set + Thumb-2, Jazelle RCT Java accelerator, and media extensions
  - Neon™ SIMD coprocessor (VFP lite + media streaming instructions)
  - Cache memories
    - Level 1: 16KB instruction and 16KB data—4-way set associative cache, 64 bytes/line

  - Level 2: Up to 256KB. See Section 1.5, *OMAP34xx Family*.
- Interrupt controller (MPU INTC) of 96 synchronous interrupt lines
- Asynchronous interface with core logic
- Debug, trace, and emulation features: ICE-Crusher, ETM™, ETB™ modules.

1.3.2 IVA2.2 Subsystem

The device includes a high-performance imaging video and audio (IVA2.2) accelerator based on the Texas Instruments TMS320DMC64x+™ VLIW DSP core.

The IVA2.2 subsystem includes the following main features:

- 32-bit fixed-point media processor
- Very long instruction word (VLIW) architecture based on the programmable enhanced version of C64x™ DSP core
- Eight instructions/cycle, eight execution units
  - Optimized instruction set for video and imaging processing
  - Eight 8 x 8 or 16 x 16 multiply accumulate (MAC) per cycle
  - Eight slave asynchronous die (SAD) per cycle
  - Eight interpolations (a + b + 1)>>1 per cycle
  - Two (32-bit x 32-bit > 64-bit) multiply operations per cycle
- Low-power processor and megacell
  - Dynamically mixed 32-bit and 16-bit instruction sets
  - Software pipelined loop (SPLOOP) instruction buffer
  - Separate power domain
  - Supported multiple power-down states
- Two-level memory subsystem hierarchy
  - L1P (program)
    - 32KB direct-mapped cache—32-byte cache line, configurable as cache or memory mapped (Possible values are: 0KB cache/32KB memory, 4KB/28KB, 8KB/24KB, 16KB/16KB, or 32KB/0KB)
  - L1D (data)
    - 32KB 2-way set associative cache—4-byte cache line, configurable as cache or memory mapped (Possible values are: 0KB cache/32KB memory, 4KB/28KB, 8KB/24KB, 16KB/16KB, or 32KB/0KB)
    - 48KB memory-mapped SRAM
  - L2 (program and data)
    - 64KB 4-way set associative cache—128-byte cache line, configurable as cache or memory mapped (Possible values are: 0KB cache/64KB memory, 32KB/32KB, or 64KB/0KB)
    - 32KB memory-mapped SRAM
    - 16KB ROM
- Video hardware accelerators
  - Improved motion estimation (iME) dedicated hardware
  - Improved loop filtering (iLF) dedicated hardware
  - Video dedicated sequencer
  - Video local interconnect
  - Local level 2 (L2) memory interface/arbiter
- Private direct memory access (DMA) controller:
  - 128 logical channels
  - 1D/2D addressing
  - Chaining capability
  - Fully pipelined, two 64-bit read ports, two 64-bit write ports
  - Single access 32-byte or 64-byte incrementing bursts

- Level 1 (L1) interrupt controller (INTC)
- Local IVA2.2 digital phase-locked loop (DPLL) supplying the IVA2.2 subsystem clocking
- 32-entry memory management unit (MMU) for seamless integration in high-level OS environment
- IVA2.2 system interfaces
  - 64-bit L3 port shared for external memory accesses
    - Multithreaded link shared by DSP core and DMA accesses
    - Interface with the L3 interconnect that can be synchronous or asynchronous for clock decoupling between IVA2.2 and L3 interconnect
    - Incrementing burst support
    - Critical line first to reduce line fetch latency to the processor
  - Host port interface (HPI) for MMU programming and access to the IVA2.2 internal memories. Can be synchronous or asynchronous
  - System interfaces: clocking, power management
- C-friendly environment (state-of-the-art C compiler for VLIW architecture)
- Supports Texas Instruments low-overhead DSP-BIOS operating system

1.3.3 On-Chip Memory

On-chip memory configuration offers memory resources for program and data storage:

- 112KB ROM
- 64KB single-access static random access memory (SRAM)

1.3.4 External Memory Interfaces

The device includes two external memory interfaces supporting the stacking of a multichip memory package using the generic POP interface:

- General-purpose memory controller (GPMC)
  - NOR flash, NAND flash (with ECC Hamming code calculation), SRAM and Pseudo-SRAM asynchronous and synchronous protocols
  - Flexible asynchronous protocol control for external ASIC or peripheral interfacing
  - 16-bit data, up to 8 chip-selects (CSs)
  - 128M-byte addressable per chip-select, 1G-byte total address space
  - Nonmultiplexed device with limited address (2K bytes)
- SDRAM controller (SDRC)
  - Mobile single data rate (M-SDR) SDRAM and low-power double data rate (LPDDR) SDRAM
  - 16-bit or 32-bit data, 2 chip-selects, configurations for a maximum of 1 G-byte address space per chip-select
  - Work in conjunction with the SDRAM memory scheduler (SMS) companion module

1.3.5 DMA Controllers

The device embeds one generic DMA controller, the system DMA (sDMA) controller used for memory-to-memory, memory-to-peripheral, and peripheral-to-memory transfers:

- One read port, one write port
- 32 prioritizable logical channels
- 96 hardware requests
- 256 x 32-bit FIFO dynamically allocable between active channels

The device also embeds three dedicated DMA controllers: enhanced DMA (EDMA), which is embedded in the IVA2.2 subsystem, display DMA, and USB HS DMA.

1.3.6 Multimedia

The device uses the following multimedia accelerators for display and gaming effects as well as high-end imaging and video applications:

Note: The SGX subsystem is an instantiation by Texas Instruments of the POWERVR™ SGX530 core from Imagination Technologies Ltd.

This document contains materials that are ©2003-2007 Imagination Technologies Ltd.

POWERVR and USSE are trademarks or registered trademarks of Imagination Technologies Ltd.

- 2D and 3D graphics accelerator (POWERVR™ SGX530)
  - 2D and 3D graphics and video codecs supported on common hardware
  - Tile-based architecture
  - USSE™ multithreaded engine incorporating pixel and vertex shader functionality reducing die area
  - Advanced shader feature set in excess of Microsoft VS3.0, PS3.0, and OpenGL2.0
  - Industry standard API support Direct3D mobile, OpenGL ES 1.1 and 2.0, OpenVG 1.0.1, OpenMax
  - Fine-grained task switching, load balancing, and power management
  - Programmable high-quality image anti-aliasing
  - Advanced geometry DMA driven operation for minimum CPU interaction
  - Fully virtualized memory addressing for OS operation in a unified memory architecture
  - Advanced and standard 2D operations (e.g, vector graphics, Block Level Transfers, Raster Operations, etc.)
  - Programmable support GP-GPU functions

Note: Multimedia accelerators are not available on all devices. See Section 1.5, *OMAP34xx Family*, for more information on available features.

- Camera interface
  - Supports most of the raw image sensors available in the market
  - Serial interface compatible with the MIPI® CSI1 specification (CSIb) and the MIPI CSI2 specification (CSIa)
  - 12-bit parallel interface supported
  - Pixel clock up to 83 MHz
  - Embedded DMA controller in CSI2 receiver
- Display interface
  - Display controller
  - Color and monochrome displays up to 2048 x 2048 x 24-bpp resolution
  - 256 x 24-bit entries palette in red, green, blue (RGB)
  - Picture-in-picture (overlay), color-space conversion, rotation, color-phase rotation, and resizing support
  - Remote frame buffer interface
  - Liquid-crystal display (LCD) pixel interfaces (MIPI DPI 1.0) and LCD bus interfaces (MIPI DBI 1.0) supported
  - NTSC/PAL video encoder outputs with integrated digital-to-analog converters (DACs) output are supported on CVBS and S-video TV analog output signals
  - Serial display interface implements high-speed differential output buffers to support FlatLink3G™and MIPI DSI 1.0 formats
  - Support of HDMI via external bridge
  - Embedded DMA controller

1.3.7 Comprehensive Power Management

- Clock and reset generation and distribution
- Wake-up event management
- Dynamic voltage and frequency scaling
- Dynamic power switching
- Static leakage management

1.3.8 Peripherals

The device supports a comprehensive set of peripherals to provide flexible and high-speed interfacing and on-chip programming resources:

- Universal asynchronous receiver/transmitter (UART) 1/2
  Two general serial communication interfaces
- UART 3
  UART + IrDA SIR up to FIR + TV remote control interface (CIR)
- Multichannel buffered serial port 1 (McBSP1), McBSP2, McBSP3, McBSP4, and McBSP5
  Three general purpose (McBSP1, McBSP4, and McBSP5) and two audio-loopback capable (McBSP2 and McBSP3 associated with two sidetone modules) multichannel-buffered serial interfaces
- I2C1, I2C2, and I2C3
  Three master/slave I^2C high-speed standard interfaces
- HDQ/1-Wire
  Benchmark HDQ and Dallas Semiconductor 1-Wire protocol interfaces
- Multichannel serial port interface 1 (McSPI1), McSPI2, McSPI3, and McSPI4
  Four McSPI controllers
- High-speed multiport USB host
  High-speed USB host that is used for (local) interprocessor communication (that is, from OMAP to modems) using the TLL mode
- High-speed USB OTG
  High-speed controller that offers high-speed data transactions (up to 480 Mbps) on a USB port (ULPI 12 bits) with embedded DMA controller
- High-speed MMC/SD/SDIO 1/2/3
  Three interface controllers for HS MMC/SD/SDIO standards
- GP timers
  Eleven general-purpose timers
- Watchdog timers (WDTs)
  Two watchdog timers
- 32-kHz synchronization timer
  32-kHz clock timer
- General-purpose input/output (GPIO)
  Six 32-bit, general-purpose, input/output (I/O) controllers
- Mailbox
  MPU/IVA2.2 interprocessor communications (six in stacked mode, two in stand-alone mode)
  ICR (only in stacked mode)
- Control module
  I/O multiplexing and chip-configuration control

1.4 Package-On-Package Concept

The OMAP die uses flip-chip technology. The OMAP34xx POP device supports memory stacking using a POP implementation.

The POP device provides a generic POP memory interface to support multiple stacked package configurations, including the flash multichip package, depending on customer needs.

The stacked memory package is directly connected to the two memory interfaces (GPMC and SDRC) of the POP device through the POP interface present at the top. (For more information on the interconnect between the stacked memory package and the POP device, see Chapter 11, *Memory Subsystem*)

Figure 1-3 shows the concept of the POP solution, and Figure 1-4 shows stacked memory package on the POP device.

Figure 1-3. POP Concept

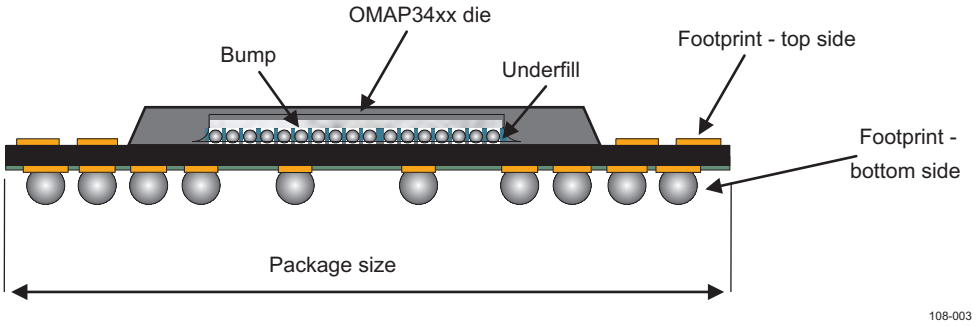
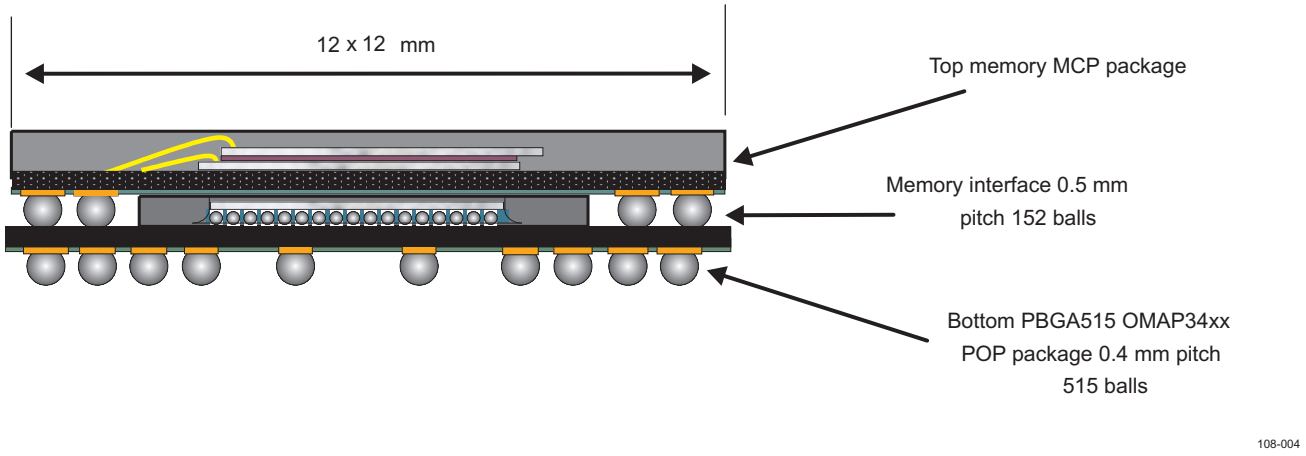


Figure 1-4. Stacked Memory Package on the POP Device



The memory interfaces should be correctly configured based on the memory package used with the POP device.

Table 1-1 summarizes the supported configurations with the generic POP interface.

Note: For other types of memory, the traditional GPMC and SDRC interfaces can be used through the board.

Table 1-1. Summary of Memories Supported by the POP Interface

Generic POP Interface Features Set	SDRC Interface	GPMC Interface
Type of memory supported	MDDR	NOR flash asynchronous and synchronous burst flash NAND flash "CE don't care" One NAND on CS0 and CS1 NOR flash address/data nonmultiplexed is not supported.
Number of chip-selects	2	2
Maximum density per chip-select	2G bits	1G NOR flash
Maximum size per interface	4G bits	2G bits NOR flash
Interface width	X 32 bits	X 16 bits
I/O voltage	1.8 V LVCMOS	1.8 V LVCMOS

For more information on the memory interface configuration, see Chapter 11, *Memory Subsystem*.

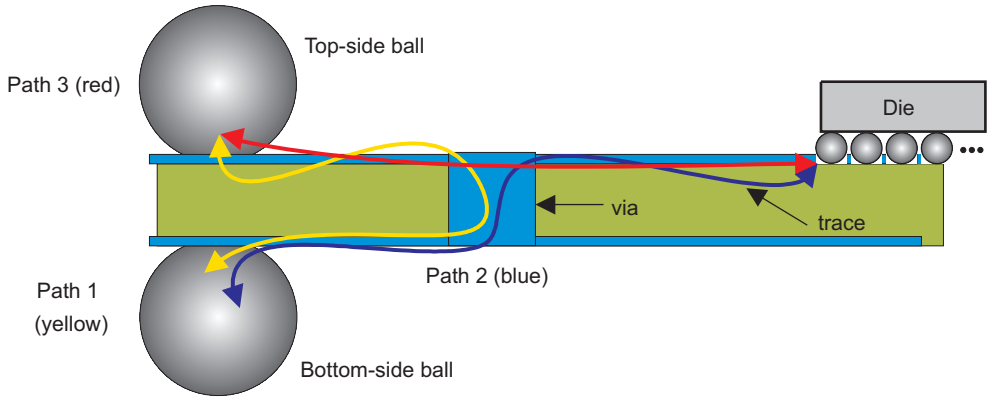
The POP device includes feedthroughs in addition to the GPMC and SDRC interfaces positioned at the top of the package. Up to 47 feedthroughs are defined from the bottom ball-grid array (BGA) to the top (or POP) interface to support different memory combinations.

The feedthroughs either provide power to a top memory device or provide specific memory signals (for example, control and/or address signals) from the bottom BGA to the POP interface.

Note: It is possible to monitor the DDR SDRAM temperature if the memory multichip package allows the temperature sensing option. A feedthrough is used to lower the temperature sensing dedicated signal to make a connection with a GPIO. For more information on DDR SDRAM temperature sensing management, see Chapter 11, *Memory Subsystem*, and Chapter 25, *General-Purpose Interface*.

Figure 1-5 shows the implementation of feedthroughs on the POP and the different paths between the bottom and the top of the package.

Figure 1-5. Stacked Memory Package on the POP Device



(1) Path 1: feedthrough only; Path 2: all but feedthrough; Path 3: POP interface (SDRC + subset GPMC)

(2) Any signal available on the POP interface to a top-side ball is also available to a bottom-side ball.

1.5 OMAP34xx Family

1.5.1 Device Features

The OMAP34xx family is composed of different devices:

- OMAP3440-720
- OMAP3430
- OMAP3420

Table 1-2 summarizes the features of OMAP34xx devices:

Table 1-2. Table of Features

Feature	OMAP3430/OMAP3440-720	OMAP3420
SGX	Yes	Yes
All IVA2 video hardware accelerators(1)	Yes	Yes
MPU L2 cache size	256KB	256KB
ARM frequency	*This information is not available in public domain.*	
IVA2 frequency	*This information is not available in public domain.*	

(1) See Section 1.3.2 for definition of video hardware accelerators.

To retrieve available features of an OMAP34xx device, see the CONTROL_FEATURE_OMAP_STATUS and CONTROL_PRODUCTION_ID registers.

Table 1-3. CONTROL_FEATURE_OMAP_STATUS

Address Offset	0x0000 01DC		
Physical Address	0x4800 244C	**Instance**	GENERAL
Description	This register shows the feature status of the device.		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																	SGX		IVA2_HW	MPU_L2_CACHESIZE		ARM_MHZ		IVA2_MHZ		RESERVED						

Bits	Field Name	Description	Type	OMAP3430/ OMAP3440-720 Device Reset Value	OMAP3420 Device Reset Value
31:15	RESERVED	RESERVED	R	0x00000	0x00000
14:13	SGX	0x0 = SGX available with full speed clock 0x1 = SGX available with half speed clock 0x2 = Not available 0x3 = Reserved	R	0x0	0x0
12	IVA2_HW	0x0 = Video Hardware accelerators available 0x1 = Video Hardware accelerators not available	R	0x0	0x0
11:10	MPU_L2_CACHESIZE	0x0 = 0KB 0x1 = 64KB 0x2 = 128KB 0x3 = 256KB	R	0x3	0x3
9:8	ARM_MHZ	*This information is not available in public domain.*	R	0x0	0x0
7:6	IVA2_MHZ	*This information is not available in public domain.*	R	0x0	0x0
5:0	RESERVED	RESERVED	R	0x-	0x-

Table 1-4. Register Call Summary for Register CONTROL_FEATURE_OMAP_STATUS

OMAP34xx Family

- Device Features: [0]

Table 1-5. CONTROL_PRODUCTION_ID

Address Offset	0x0000 0000		
Physical Address	0x4830 A210	**Instance**	GENERAL
Description	This register shows the device type and some options availability.		
Type	R		

127	...	...	...	...	...	...	...	...	...	...	72	71	...	64	63	...	...	...	...	...	...	9	8	7	6	5	4	3	2	1	0
RESERVED												DEVICE_TYPE			RESERVED								RESERVED	RESERVED							

Bits	Field Name	Description	Type	OMAP3430/ OMAP3440-720 Device Reset Value	OMAP3420 Device Reset Value
127:72	RESERVED	RESERVED	R	0x-	0x-
71:64	DEVICE_TYPE	0xF0 = GP device Other values = Reserved	R	0x-	0x-
63:9	RESERVED	RESERVED	R	0x-	0x-
8	RESERVED	This information is not available in public domain.	R	0x-	0x-
7:0	RESERVED	RESERVED	R	0x-	0x-

Table 1-6. Register Call Summary for Register CONTROL_PRODUCTION_ID

OMAP34xx Family

- Device Features: [0]
- Device Identification: [1] [2] [3]

1.5.2 Device Identification

The identification registers include the CONTROL_IDCODE, CONTROL_PRODUCTION_ID and CONTROL_DIE_ID data registers. These registers are read only accessed ports that are programmed into eFuses FARM FROM.

Table 1-7 and Table 1-8 describe the identification registers.

See Chapter 2, *Memory Mapping*, for the memory space address of the test chip-level TAP device.

Table 1-7. Device Identification Registers

Register Name	Physical Address	Size
CONTROL.CONTROL_IDCODE[31:0]	0x4830 A204	32
CONTROL.CONTROL_PRODUCTION_ID[127:0]	0x4830 A208	128
CONTROL.CONTROL_DIE_ID[127:0]	0x4830 A218	128

The device type and some options can be read in the CONTROL.CONTROL_PRODUCTION_ID register. See Table 1-5.

The silicon type can be read in the HAWKEYE bit field value of the CONTROL.CONTROL_IDCODE register. The silicon revision can be read in the VERSION bit field value of the CONTROL.CONTROL_IDCODE register.

Table 1-8. CONTROL_IDCODE Register Definition

Field	Bits	Value	Comment
CONTROL.CONTROL_IDCODE [31:28]	VERSION	See Table 1-10.	Revision number
CONTROL.CONTROL_IDCODE [27:12]	HAWKEYE	See Table 1-9.	Hawkeye number
CONTROL.CONTROL_IDCODE [11:1]	TI_IDM	See (1).	Manufacturer identity (TI)
CONTROL.CONTROL_IDCODE [0]	--	0x1	Always set to 1

(1) TI internal data

The Hawkeye number is hardcoded in the design. Table 1-9 lists the Hawkeye number values, and Table 1-10 lists the revision number values.

Table 1-9. Hawkeye Number Value

Silicon Type	Field	Value
OMAP34xx ES1.0	CONTROL.CONTROL_IDCODE[27:12]	0xB6D6
OMAP34xx ES2.0	CONTROL.CONTROL_IDCODE[27:12]	0xB7AE
OMAP34xx ES2.1	CONTROL.CONTROL_IDCODE[27:12]	0xB7AE
OMAP34xx ES3.0	CONTROL.CONTROL_IDCODE[27:12]	0xB7AE
OMAP34xx ES3.1	CONTROL.CONTROL_IDCODE[27:12]	0xB7AE
OMAP34xx ES3.1.2	CONTROL.CONTROL_IDCODE[27:12]	0xB7AE

Table 1-10. Revision Number Value

Silicon Type	Field	Value
ES1.0	CONTROL.CONTROL_IDCODE[31:28]	0000
ES2.0	CONTROL.CONTROL_IDCODE[31:28]	0001
ES2.1	CONTROL.CONTROL_IDCODE[31:28]	0010
ES3.0	CONTROL.CONTROL_IDCODE[31:28]	0011
ES3.1	CONTROL.CONTROL_IDCODE[31:28]	0100
ES3.1.2	CONTROL.CONTROL_IDCODE[31:28]	0111

The CONTROL.CONTROL_IDCODE value is 0x0B6D 602F for OMAP34xx ES1.0.

The CONTROL.CONTROL_IDCODE value is 0x1B7A E02F for OMAP34xx ES2.0.

The CONTROL.CONTROL_IDCODE value is 0x2B7A E02F for OMAP34xx ES2.1.

The CONTROL.CONTROL_IDCODE value is 0x3B7A E02F for OMAP34xx ES3.0.

The CONTROL.CONTROL_IDCODE value is 0x4B7A E02F for OMAP34xx ES3.1.

The CONTROL.CONTROL_IDCODE value is 0x7B7A E02F for OMAP34xx ES3.1.2.

Table 1-11. CONTROL_PRODUCTION_ID Register Silicon Type Identification

Silicon Type	Field	Value
ES1.0	CONTROL.CONTROL_PRODUCTION_ID[127:96]	0xCAFE0000
ES2.0	CONTROL.CONTROL_PRODUCTION_ID[127:96]	0xCAFEB7AE
ES2.1	CONTROL.CONTROL_PRODUCTION_ID[127:96]	0xCAFEB7AE
ES3.0	CONTROL.CONTROL_PRODUCTION_ID[127:96]	0xCAFEB7AE

Table 1-12. CONTROL_DIE_ID

Field	Bits	Value
DIE_ID[127:0]	RESERVED	Single identifier

1.5.3 General Recommendations Relative to Unavailable Features/Modules

This TRM details all features available in high-tier devices. Some features are not available for lower-tier devices. For unavailable features, use the following recommandations:

- Memory mapping: Memory area of unavailable modules and features are RESERVED, read is undefined, and write can lead to unpredictable behavior.
- Interrupt controllers: Ensure that interrupts of unavailable modules and features are masked in MPU/IVA subsystems.
- DMA: Ensure that DMA requests of unavailable modules and features are masked in DMA subsystems.
- SCM: Unavailable module and feature pins are not functional.
- PRCM: For power management and power-saving consideration, ensure that power domains of unavailable features/modules are switched off and clocks are cut off.

- Interconnect: To flag potential interconnect outstanding commands, the time-out of target agents attached to unavailable modules can be enabled with the lowest setting.

Memory Mapping

This chapter describes memory mapping in the OMAP34xx multimedia device.

Note: This chapter gives information about all modules and features in the high-tier device. To check availability of modules and features, see Section 1.5, *OMAP34xx Family*. In unavailable modules and features, the memory area is reserved, read is undefined, and write can lead to unpredictable behavior.

Topic **Page**

2.1 **Introduction** 192
2.2 **Global Memory Space Mapping** 194
2.3 **L3 and L4 Memory Space Mapping** 197
2.4 **IVA2.2 Subsystem Memory Space Mapping** 206

2.1 Introduction

The microprocessor unit (MPU) has a 32-bit address port, allowing it to handle a 4-GB space divided into several regions, depending on the target type.

The memory map is composed of a memory space (general-purpose memory controller [GPMC], synchronous dynamic random access memory [SDRAM] controller [SDRC], etc.), register space (L3 and L4 interconnects), and dedicated spaces (image and video accelerator [IVA2.2] subsystem, SGX, etc.), all of which are shared among the initiators (for example, the microprocessor [MPU] subsystem or the IVA2.2 subsystem).

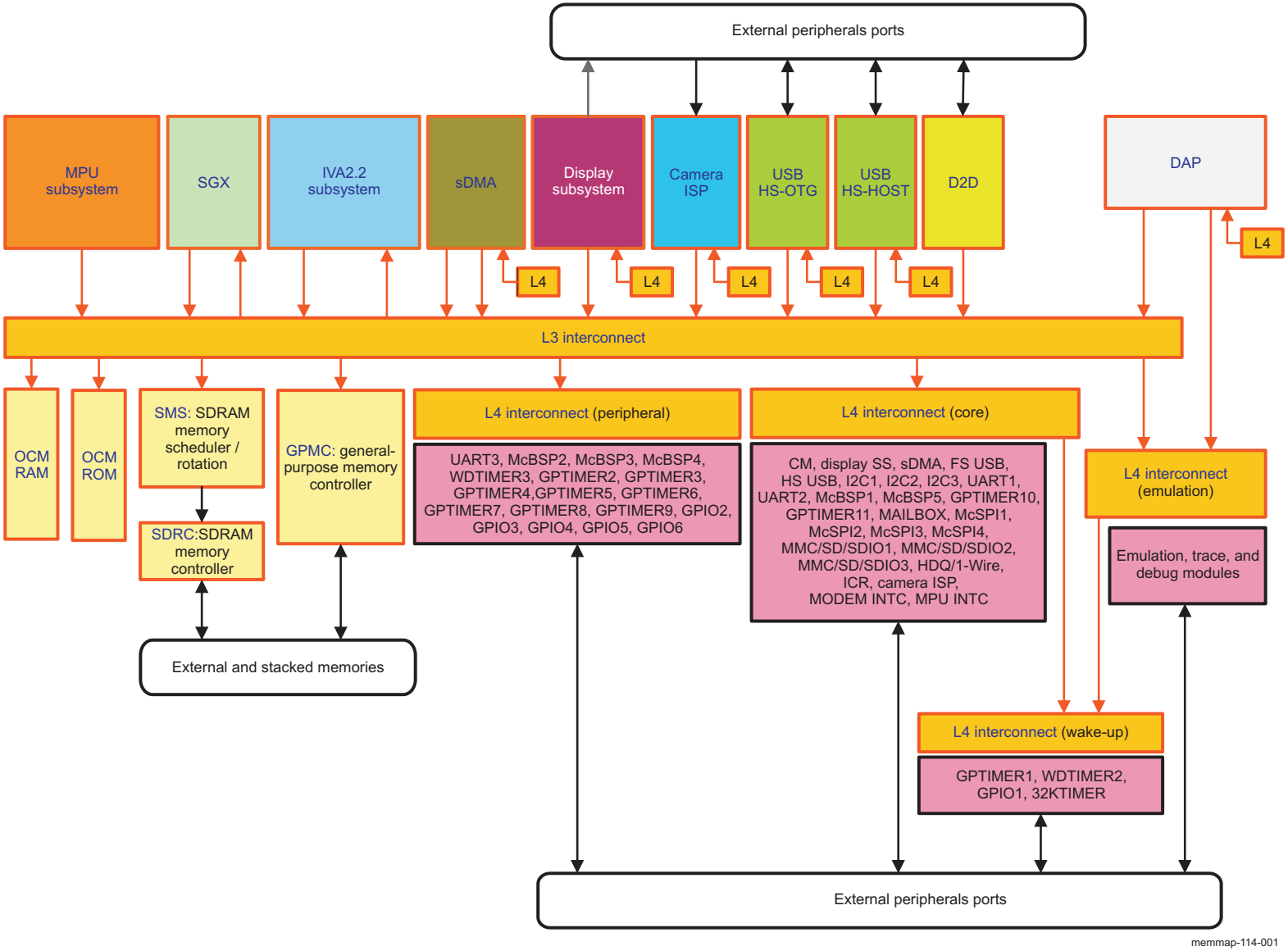
The GPMC and SDRC are dedicated to memory connection. The GPMC is used for NOR/NAND flash and SRAM memories. The SDRC is used for SDRAM memories, such as regular SDR-SDRAM (single data rate), regular JEDEC DDR1 memory (double data rate), low-power SDR-SDRAM, and mobile DDR-SDRAM. For more information, see Chapter 11, *Memory Subsystem*.

The L3 interconnect allows the sharing of resources, such as peripherals and external or on-chip memories, among all the initiators of the platform. The L4 interconnects control access to the peripherals.

Transfers between initiators and targets across the platform are physically conditioned by the chip interconnect and can be logically conditioned by firewalls. For more information about the intercommunication (L3 and L4 interconnects) and protection mechanisms implemented in the device, see Chapter 5, *Interconnect*.

Figure 2-1 shows the interconnect of the device and the main modules and subsystems in the platform.

Figure 2-1. Interconnect Overview



2.2 Global Memory Space Mapping

This section provides a global view of the memory mapping and describes the boot, GPMC, SDRC, and virtual rotated frame buffer (VRFB) memory spaces.

The system memory mapping is flexible, with two levels of granularity for target address space allocation:

- Level 1: Four quarters are labeled Q0, Q1, Q2, and Q3. Each quarter corresponds to a 1GB address space (total address space is 4GB).
- Level 2: Each quarter is divided into eight blocks of 128MB, with target spaces mapped in the blocks.

This organization allows all target spaces to be decoded based on the five most-significant bits (MSBs) of the 32-bit address ([31:27]).

- **Boot space**

  The system has a 1MB boot space in the on-chip boot ROM or on the GPMC memory space.

  When booting from the on-chip ROM with the appropriate external sys_boot5 pin configuration, the 1MB memory space is redirected to the on-chip boot ROM memory address space [0x4000 0000 – 0x400F FFFF].

  When booting from the GPMC with the appropriate external sys_boot5 pin configuration, the memory space is part of the GPMC memory space.

  For more information about sys_boot5 pin configuration, see Chapter 11, *Memory Subsystem*, and Chapter 26, *Initialization*.
- **GPMC space**

  Eight independent GPMC chip-selects (gpmc_ncs0 to gpmc_ncs7) are available in the first quarter (Q0) of the addressing space to access NOR/NAND flash and SRAM memories. The chip-selects have a programmable start address and programmable size (16MB, 32MB, 64MB, or 128MB) in a total memory space of 1GB.
- **SDRC space**

  Two SDRC chip-selects (sdrc_ncs0 and sdrc_ncs1) are available on the third quarter (Q2) of the addressing space to access SDRAM memories. The chip-selects have a programmable size (64, 128, 256, or 512MB) in a total memory space of 1GB.

  The base address of the chip-select 0 (sdrc_ncs0) memory space is always 0x8000 0000. The base address of the chip-select 1 (sdrc_ncs1) memory space is programmable. The default value after reset is 0xA000 0000.
- **VRFB space**

  The SDRC-SMS virtual memory space is a different memory space used to access a subset of the SDRC memory space through the rotation engine. The virtual address space size is 768MB split into two parts: The first 256MB part is in the second quarter (Q1) of the memory; the second 512MB part is in the fourth quarter (Q3) of the memory.

For more information about boot, GPMC, SDRC, and VRFB, see Chapter 11, *Memory Subsystem*.

This section describes all modules and features in the high-tier device. To check availability of modules and features, see Section 1.5, *OMAP34xx Family*. In unavailable modules and features, the memory area is reserved, read is undefined, and write can lead to unpredictable behavior.

Table 2-1 describes the global memory space mapping.

Table 2-1. Global Memory Space Mapping

QUARTER	Device Name	Start Address (HEX)	End Address (HEX)	Size	Description
Q0 (1GB)	**Boot space[1] GPMC**			**1MB 1GB or 1GB-1MB**	
	GPMC	0x0000 0000	0x3FFF FFFF	1GB	8/16 Ex[2]/R/W
Q1 (1GB)	**On-Chip memory**			**128MB**	**ROM/SRAM address space**
	Boot ROM internal[1]	0x4000 0000	0x4001 3FFF	80KB	Reserved for boot code Not accessible after boot
		0x4001 4000	0x4001 BFFF	32KB	32-bit Ex[2]/R
	Reserved	0x4001 C000	0x400F FFFF	912KB	Reserved
	Reserved	0x4010 0000	0x401F FFFF	1MB	Reserved
	SRAM internal	0x4020 0000	0x4020 FFFF	64KB	32-bit Ex[2]/RW
	Reserved	0x4021 0000	0x4024 FFFF	256KB	Reserved
	Reserved	0x4025 0000	0x47FF FFFF	128,704KB	Reserved
	L4 interconnects			**128MB**	**All system peripherals**
	L4-Core	0x4800 0000	0x48FF FFFF	16MB	See Table 2-3.
	(L4-Wakeup)[3]	(0x4830 0000)	(0x4833 FFFF)	(256KB)	(See Table 2-4.)
	L4-Per	0x4900 0000	0x490F FFFF	1MB	See Table 2-5.
	Reserved	0x4910 0000	0x4FFF FFFF	111MB	Reserved
	SGX			**64MB**	**Graphic accelerator slave port**
	SGX	0x5000 0000	0x5000 FFFF	64KB	Graphic accelerator slave port
	Reserved	0x5001 0000	0x53FF FFFF	65,472KB	Reserved
	L4 Emulation			**64MB**	**Emulation**
	L4-Emu	0x5400 0000	0x547F FFFF	8MB	See Table 2-6.
	Reserved	0x5480 0000	0x57FF FFFF	56MB	Reserved
	Reserved			**64MB**	**Reserved**
	Reserved	0x5800 0000	0x5BFF 0FFF	64MB	Reserved
	IVA2.2 Subsystem			**64MB**	**IVA2.2 subsystem**
	IVA2.2 Subsystem	0x5C00 0000	0x5EFF FFFF	48MB	IVA2.2 subsystem. See Table 2-8.
	Reserved	0x5F00 0000	0x5FFF FFFF	16MB	Reserved
	Reserved			**128MB**	**Reserved**
	Reserved	0x6000 0000	0x67FF FFFF	128MB	Reserved
	L3 Interconnect			**128MB**	**Control registers**
	L3 control registers	0x6800 0000	0x68FF FFFF	16MB	See Table 2-2.
	Reserved	0x6900 0000	0x6BFF FFFF	48MB	Reserved
	SMS registers	0x6C00 0000	0x6CFF FFFF	16MB	Configuration registers SMS address space 2
	SDRC registers	0x6D00 0000	0x6DFF FFFF	16MB	Configuration registers SMS address space 3
	GPMC registers	0x6E00 0000	0x6EFF FFFF	16MB	Configuration registers GPMC address space 1
	Reserved	0x6F00 0000	0x6FFF FFFF	16MB	Reserved

(1) Boot space location depends on the external sys_boot5 pin configuration.
(2) Executable
(3) Peripherals connected to the L4-Wakeup interconnect are accessed through the L4-Core interconnect.

Table 2-1. Global Memory Space Mapping (continued)

QUARTER	Device Name	Start Address (HEX)	End Address (HEX)	Size	Description
	SDRC/SMS			**256MB**	**SDRC/SMS**
	SDRC/SMS virtual Address space #0	0x7000 0000	0x7FFF FFFF	256MB	SDRC-SMS virtual address space 0
Q2 (1GB)	**SDRC/SMS**			**1GB**	**SDRAM main address space (SMS)**
	CS0 – SDRAM[4]	0x8000 0000	0x9FFF FFFF	512MB	SDRC/SMS
	CS1 – SDRAM[4]	0xA000 0000	0xBFFF FFFF	512MB	SDRC/SMS
Q3 (1GB)	**Reserved**			**512MB**	**Reserved**
	Reserved	0xC000 0000	0xDFFF FFFF	512MB	Reserved for future use
	SDRC/SMS			**512MB**	**SDRC/SMS**
	SDRC/SMS virtual Address space 1	0xE000 0000	0xFFFF FFFF	512MB	SDRC-SMS virtual address space 1

(4) Chip-select 0 and chip-select 1 spaces are configurable in the 1GB SDRC/SMS space.

2.3 L3 and L4 Memory Space Mapping

The memory space system is hierarchical: L1, L2, L3, and L4.

L1 and L2 are memories in in the MPU and the IVA2.2 subsystems.

The chip-level interconnect, which consists of one L3 and four L4s, enables communication among all modules and subsystems.

L3 handles many types of data transfers, including data exchange with system on-chip/external memories.

The four L4s handle transfers with peripherals, but are in four distinct power domains: the L4-Core, L4-Wakeup, L4-Per, and L4-Emu interconnects, which are in the CORE, WKUP, PER, and EMU power domains, respectively.

For more information about the interconnect, see Chapter 5, *Interconnect*.

The following sections describe the register mapping of the L3 and L4 interconnects. The software configures these registers.

2.3.1 L3 Memory Space Mapping

The L3 interconnect control registers are mapped in a 16MB space and allow the configuration of the L3 interconnect parameters.

The L3 default settings are fully functional and enable all possible functional data paths. However, the interconnect parameters can be changed to accommodate expectations.

Accesses to the L3 interconnect can be configured on a per-module basis using the internal L3 registers, which are grouped into five register block types:

- IA: Initiator agent configuration registers
- TA: Target agent configuration registers
- RT: Register target (global configuration registers)
- PM: Protection mechanism (firewalls) configuration registers
- SI: Global sideband signal configuration registers

For more information, see Chapter 5, *Interconnect*.

This section describes all modules and features in the high-tier device. To check availability of modules and features, see Section 1.5, *OMAP34xx Family*. In unavailable modules and features, the memory area is reserved, read is undefined, and write can lead to unpredictable behavior.

Table 2-2 describes the mapping of the L3 interconnect control registers.

Table 2-2. L3 Control Register Mapping

Device Name	Start Address (Hex)	End Address (Hex)	Size (KB)	Description
L3 RT	0x6800 0000	0x6800 03FF	1	L3 configuration registers
L3 SI	0x6800 0400	0x6800 07FF	1	Sideband signal configuration
Reserved	0x6800 0800	0x6800 13FF	3	Reserved
MPU SS IA	0x6800 1400	0x6800 17FF	1	MPU subsystem instruction port agent configuration
IVA2.2 SS IA	0x6800 1800	0x6800 1BFF	1	IVA2.2 subsystem initiator port agent configuration
SGX SS IA	0x6800 1C00	0x6800 1FFF	1	SGX subsystem initiator port agent configuration
SMS TA	0x6800 2000	0x6800 23FF	1	SMS target port agent configuration
GPMC TA	0x6800 2400	0x6800 27FF	1	GPMC target port agent configuration
OCM RAM TA	0x6800 2800	0x6800 2BFF	1	OCM RAM target port agent configuration
OCM ROM TA	0x6800 2C00	0x6800 2FFF	1	OCM ROM target port agent configuration

Table 2-2. L3 Control Register Mapping (continued)

Device Name	Start Address (Hex)	End Address (Hex)	Size (KB)	Description
D2D IA	0x6800 3000	0x6800 33FF	1	Die-to-die (D2D) initiator port agent configuration
D2D TA	0x6800 3400	0x6800 37FF	1	D2D target port agent configuration.
Reserved	0x6800 3800	0x6800 3FFF	2	Reserved
HS USB HOST IA	0x6800 4000	0x6800 43FF	1	HS USB HOST initiator port agent configuration
HS USB OTG IA	0x6800 4400	0x6800 47FF	1	HS USB OTG initiator port agent configuration
Reserved	0x6800 4800	0x6800 4BFF	1	Reserved
sDMA RD IA	0x6800 4C00	0x6800 4FFF	1	System DMA (sDMA) RD initiator port agent configuration
sDMA WR IA	0x6800 5000	0x6800 53FF	1	sDMA WR initiator port agent configuration
Display SS IA	0x6800 5400	0x6800 57FF	1	Display subsystem initiator port agent configuration
CAMERA ISP IA	0x6800 5800	0x6800 5BFF	1	Camera ISP initiator port agent configuration
DAP IA	0x6800 5C00	0x6800 5FFF	1	Debug access port initiator port agent configuration
IVA2.2 SS TA	0x6800 6000	0x6800 63FF	1	IVA2.2 subsystem target port agent configuration
SGX SS TA	0x6800 6400	0x6800 67FF	1	SGX subsystem target port agent configuration
L4-Core TA	0x6800 6800	0x6800 6BFF	1	L4-Core target port agent configuration
L4-Per TA	0x6800 6C00	0x6800 6FFF	1	L4-Per target port agent configuration
Reserved	0x6800 7000	0x6800 FFFF	36	Reserved
RT PM	0x6801 0000	0x6801 03FF	1	Register target port protection
Reserved	0x6801 0400	0x6801 23FF	8	Reserved
GPMC PM	0x6801 2400	0x6801 27FF	1	GPMC target port protection
OCM RAM PM	0x6801 2800	0x6801 2BFF	1	OCM RAM target port protection
OCM ROM PM	0x6801 2C00	0x6801 2FFF	1	OCM ROM target port protection
D2D PM	0x6801 3000	0x6801 33FF	1	D2D target port protection
Reserved	0x6801 3400	0x6801 3FFF	3	Reserved
IVA2.2 PM	0x6801 4000	0x6801 43FF	1	IVA2.2 subsystem target port protection
Reserved	0x6801 4400	0x68FF FFFF	16,303	Reserved

2.3.2 L4 Memory Space Mapping

The device contains four L4 interconnects: L4-Core, L4-Wakeup, L4-Per, and L4-Emu.

As with the L3 interconnect, the L4 interconnects can be configured to tune the access depending on the characteristics of each module.

For more information about the L4 interconnect, see Chapter 5, *Interconnect*.

2.3.2.1 L4-Core Memory Space Mapping

The L4-Core interconnect is a 16MB space composed of the L4-Core interconnect configuration registers and the module registers.

Table 2-3 describes the mapping of the registers for the L4-Core interconnect.

Note: All memory spaces described as modules provide direct access to module registers outside the L4-Core interconnect. All other accesses are internal to the L4-Core interconnect.

This section describes all modules and features in the high-tier device. To check availability of modules and features, see Section 1.5, *OMAP34xx Family*. In unavailable modules and features, the memory area is reserved, read is undefined, and write can lead to unpredictable behavior.

Table 2-3. L4-Core Memory Space Mapping (1)

Device Name	Start Address (Hex)	End Address (Hex)	Size	Description
L4-Core	0x4800 0000	0x48FF FFFF	16MB	
Reserved	0x4800 0000	0x4800 1FFF	8KB	Reserved
System control module (SCM)	0x4800 2000	0x4800 2FFF	4KB	Module
	0x4800 3000	0x4800 3FFF	4KB	L4 interconnect
Clock manager • DPLL • Clock manager	0x4800 4000	0x4800 5FFF	8KB	Module region A
	0x4800 6000	0x4800 67FF	2KB	Module region B
	0x4800 6800	0x4800 6FFF	2KB	Reserved
	0x4800 7000	0x4800 7FFF	4KB	L4 interconnect
Reserved	0x4800 8000	0x4802 3FFF	112KB	Reserved
Reserved	0x4802 4000	0x4802 4FFF	4KB	Reserved
	0x4802 5000	0x4802 5FFF	4KB	Reserved
Reserved	0x4802 6000	0x4803 FFFF	104KB	Reserved
L4-Core configuration	0x4804 0000	0x4804 07FF	2KB	Address/protection (AP)
	0x4804 0800	0x4804 0FFF	2KB	Initiator port (IP)
	0x4804 1000	0x4804 1FFF	4KB	Link agent (LA)
Reserved	0x4804 2000	0x4804 FBFF	55KB	Reserved
Display subsystem • DSI • Display subsystem top • Display controller • RFBI • Video encoder	0x4804 FBFF	0x4804 FFFF	1KB	DSI
	0x4805 0000	0x4805 03FF	1KB	Display subsystem top
	0x4805 0400	0x4805 07FF	1KB	Display controller
	0x4805 0800	0x4805 0BFF	1KB	RFBI
	0x4805 0C00	0x4805 0FFF	1KB	Video encoder
	0x4805 1000	0x4805 1FFF	4KB	L4 interconnect
Reserved	0x4805 2000	0x4805 5FFF	16KB	Reserved
sDMA	0x4805 6000	0x4805 6FFF	4KB	Module
	0x4805 7000	0x4805 7FFF	4KB	L4 interconnect
Reserved	0x4805 8000	0x4805 FFFF	32KB	Reserved
I2C3	0x4806 0000	0x4806 0FFF	4KB	Module
	0x4806 1000	0x4806 1FFF	4KB	L4 interconnect
USBTLL	0x4806 2000	0x4806 2FFF	4KB	Module
	0x4806 3000	0x4806 3FFF	4KB	L4 interconnect
HS USB HOST	0x4806 4000	0x4806 4FFF	4KB	Module
	0x4806 5000	0x4806 5FFF	4KB	L4 interconnect
Reserved	0x4806 6000	0x4806 9FFF	16KB	Reserved
UART1	0x4806 A000	0x4806 AFFF	4KB	Module
	0x4806 B000	0x4806 BFFF	4KB	L4 interconnect
UART2	0x4806 C000	0x4806 CFFF	4KB	Module
	0x4806 D000	0x4806 DFFF	4KB	L4 interconnect
Reserved	0x4806 E000	0x4806 FFFF	8KB	Reserved

(1) The registers mapped in this range are shadow registers of the first 2KB region A [0x4800 4000 - 0x4800 47FF]. Region A and region B share the same port.

Table 2-3. L4-Core Memory Space Mapping (continued)

Device Name	Start Address (Hex)	End Address (Hex)	Size	Description
I2C1	0x4807 0000	0x4807 0FFF	4KB	Module
	0x4807 1000	0x4807 1FFF	4KB	L4 interconnect
I2C2	0x4807 2000	0x4807 2FFF	4KB	Module
	0x4807 3000	0x4807 3FFF	4KB	L4 interconnect
McBSP1 (Digital baseband data)	0x4807 4000	0x4807 4FFF	4KB	Module
	0x4807 5000	0x4807 5FFF	4KB	L4 interconnect
Reserved	0x4807 6000	0x4808 5FFF	64KB	Reserved
GPTIMER10	0x4808 6000	0x4808 6FFF	4KB	Module
	0x4808 7000	0x4808 7FFF	4KB	L4 interconnect
GPTIMER11	0x4808 8000	0x4808 8FFF	4KB	Module
	0x4808 9000	0x4808 9FFF	4KB	L4 interconnect
Reserved	0x4808 A000	0x4808 AFFF	4KB	Reserved
	0x4808 B000	0x4808 BFFF	4KB	Reserved
Reserved	0x4808 C000	0x4809 3FFF	32KB	Reserved
MAILBOX	0x4809 4000	0x4809 4FFF	4KB	Module
	0x4809 5000	0x4809 5FFF	4KB	L4 interconnect
McBSP5 (MIDI data)	0x4809 6000	0x4809 6FFF	4KB	Module
	0x4809 7000	0x4809 7FFF	4KB	L4 interconnect
McSPI1	0x4809 8000	0x4809 8FFF	4KB	Module
	0x4809 9000	0x4809 9FFF	4KB	L4 interconnect
McSPI2	0x4809 A000	0x4809 AFFF	4KB	Module
	0x4809 B000	0x4809 BFFF	4KB	L4 interconnect
MMC/SD/SDIO1	0x4809 C000	0x4809 CFFF	4KB	Module
	0x4809 D000	0x4809 DFFF	4KB	L4 interconnect
Reserved	0x4809 E000	0x4809 EFFF	4KB	Reserved
	0x4809 F000	0x4809 FFFF	4KB	Reserved
Reserved	0x480A 0000	0x480A AFFF	44KB	Reserved
HS USB OTG	0x480A B000	0x480A BFFF	4KB	Module
	0x480A C000	0x480A CFFF	4KB	L4 interconnect
MMC/SD/SDIO3	0x480A D000	0x480A DFFF	4KB	Module
	0x480A E000	0x480A EFFF	4KB	L4 interconnect
Reserved	0x480A F000	0x480A FFFF	4KB	Reserved
Reserved	0x480B 0000	0x480B 0FFF	4KB	Reserved
	0x480B 1000	0x480B 1FFF	4KB	Reserved
HDQ/1-wire	0x480B 2000	0x480B 2FFF	4KB	Module
	0x480B 3000	0x480B 3FFF	4KB	L4 interconnect
MMC/SD/SDIO2	0x480B 4000	0x480B 4FFF	4KB	Module
	0x480B 5000	0x480B 5FFF	4KB	L4 interconnect
ICR MPU Port (chassis mode only)	0x480B 6000	0x480B 6FFF	4KB	Module
	0x480B 7000	0x480B 7FFF	4KB	L4 interconnect
McSPI3	0x480B 8000	0x480B 8FFF	4KB	Module
	0x480B 9000	0x480B 9FFF	4KB	L4 interconnect
McSPI4	0x480B A000	0x480B AFFF	4KB	Module
	0x480B B000	0x480B BFFF	4KB	L4 interconnect

Table 2-3. L4-Core Memory Space Mapping (continued)

Device Name	Start Address (Hex)	End Address (Hex)	Size	Description
Camera ISP	0x480B C000	0x480B FFFF	16KB	Camera ISP
	0x480C 0000	0x480C 0FFF	4KB	L4 interconnect
Reserved	0x480C 1000	0x480C CFFF	48KB	Reserved
ICR modem port (chassis mode only)	0x480C D000	0x480C DFFF	4KB	Module
	0x480C E000	0x480C EFFF	4KB	L4 interconnect
Reserved	0x480C F000	0x482F FFFF	2208KB	Reserved
L4-Wakeup interconnect (region A)	0x4830 0000	0x4830 9FFF	40KB	Nonshared device mapping
Control module ID code	0x4830 A000	0x4830 AFFF	4KB	See Table 2-4
	0x4830 B000	0x4830 BFFF	4KB	L4 interconnect
L4-Wakeup interconnect (Region B)	0x4830 C000	0x4833 FFFF	208KB	See Table 2-4
	0x4834 0000	0x4834 0FFF	4KB	L4 interconnect
Reserved	0x4834 1000	0x48FF EFFF	13,052KB	Reserved

2.3.2.2 L4-Wakeup Memory Space Mapping

The L4-Wakeup interconnect is a 256KB space composed of the L4-Wakeup interconnect configuration registers and the module registers.

Table 2-4 describes the mapping of the registers for the L4-Wakeup interconnect.

Note: All memory spaces described as modules provide direct access to module registers outside the L4-Wakeup interconnect. All other accesses are internal to the L4-Wakeup interconnect.

Table 2-4. L4-Wakeup Memory Space Mapping

Device Name	Start Address (Hex)	End Address (Hex)	Size (KB)	Description
L4-Wakeup	0x4830 0000	0x4833 FFFF	256	
Reserved	0x4830 0000	0x4830 5FFF	24	Reserved
Power and reset manager • Power manager • Reset manager	0x4830 6000	0x4830 7FFF	8	Module region A
	0x4830 8000	0x4830 87FF	2	Module region B(1)
	0x4830 8800	0x4830 8FFF	2	Reserved
	0x4830 9000	0x4830 9FFF	4	L4 interconnect
Reserved	0x4830 A000	0x4830 FFFF	24	Reserved
GPIO1	0x4831 0000	0x4831 0FFF	4	Module
	0x4831 1000	0x4831 1FFF	4	L4 interconnect
Reserved	0x4831 2000	0x4831 3FFF	8	Reserved
WDTIMER2	0x4831 4000	0x4831 4FFF	4	Module
	0x4831 5000	0x4831 5FFF	4	L4 interconnect
Reserved	0x4831 6000	0x4831 7FFF	8	Reserved
GPTIMER1	0x4831 8000	0x4831 8FFF	4	Module
	0x4831 9000	0x4831 9FFF	4	L4 interconnect
Reserved	0x4831 A000	0x4831 FFFF	24	Reserved
32KTIMER	0x4832 0000	0x4832 0FFF	4	Module
	0x4832 1000	0x4832 1FFF	4	L4 interconnect
Reserved	0x4832 2000	0x4832 7FFF	24	Reseved

(1) The registers mapped in this range are shadow registers of the first 2KB region A [0x4830 6000 - 0x4830 67FF]. Regions A and B share the same port.

Table 2-4. L4-Wakeup Memory Space Mapping (continued)

Device Name	Start Address (Hex)	End Address (Hex)	Size (KB)	Description
L4-Wakeup configuration	0x4832 8000	0x4832 87FF	2	Address/protection (AP)
	0x4832 8800	0x4832 8FFF	2	Initiator port (IP) L4-Core
	0x4832 9000	0x4832 9FFF	4	Link agent (LA)
	0x4832 A000	0x4832 A7FF	2	Initiator port (IP) L4-Emu
Reserved	0x4832 A800	0x4833 FFFF	86	Reserved

2.3.2.3 L4-Peripheral Memory Space Mapping

The L4-Per interconnect is a 1MB space composed of the L4-Per interconnect configuration registers and the module registers.

Table 2-5 describes the mapping of the registers for the L4-Per interconnect.

Note: All memory spaces described as modules provide direct access to the module registers outside the L4-Per interconnect. All other accesses are internal to the L4-Per interconnect.

Table 2-5. L4-Peripheral Memory Space Mapping

Device Name	Start Address (Hex)	End Address (Hex)	Size	Description
L4-Per	0x4900 0000	0x490F FFFF	1MB	
L4-Per configuration	0x4900 0000	0x4900 07FF	2KB	AP
	0x4900 0800	0x4900 0FFF	2KB	IP
	0x4900 1000	0x4900 1FFF	4KB	LA
Reserved	0x4900 2000	0x4901 FFFF	120KB	Reserved
UART3 (Infrared)	0x4902 0000	0x4902 0FFF	4KB	Module
	0x4902 1000	0x4902 1FFF	4KB	L4 interconnect
McBSP2 (Audio for codec)	0x4902 2000	0x4902 2FFF	4KB	Module
	0x4902 3000	0x4902 3FFF	4KB	L4 interconnect
McBSP3 (Bluetooth voice data)	0x4902 4000	0x4902 4FFF	4KB	Module
	0x4902 5000	0x4902 5FFF	4KB	L4 interconnect
McBSP4 (Digital baseband voice data)	0x4902 6000	0x4902 6FFF	4KB	Module
	0x4902 7000	0x4902 7FFF	4KB	L4 interconnect
McBSP2 (Sidetone)	0x4902 8000	0x4902 8FFF	4KB	Module
	0x4902 9000	0x4902 9FFF	4KB	L4 interconnect
McBSP3 (Sidetone)	0x4902 A000	0x4902 AFFF	4KB	Module
	0x4902 B000	0x4902 BFFF	4KB	L4 interconnect
Reserved	0x4902 C000	0x4902 FFFF	16KB	Reserved
WDTIMER3	0x4903 0000	0x4903 0FFF	4KB	Module
	0x4903 1000	0x4903 1FFF	4KB	L4 interconnect
GPTIMER2	0x4903 2000	0x4903 2FFF	4KB	Module
	0x4903 3000	0x4903 3FFF	4KB	L4 interconnect
GPTIMER3	0x4903 4000	0x4903 4FFF	4KB	Module
	0x4903 5000	0x4903 5FFF	4KB	L4 interconnect
GPTIMER4	0x4903 6000	0x4903 6FFF	4KB	Module
	0x4903 7000	0x4903 7FFF	4KB	L4 interconnect

Table 2-5. L4-Peripheral Memory Space Mapping (continued)

Device Name	Start Address (Hex)	End Address (Hex)	Size	Description
GPTIMER5	0x4903 8000	0x4903 8FFF	4KB	Module
	0x4903 9000	0x4903 9FFF	4KB	L4 interconnect
GPTIMER6	0x4903 A000	0x4903 AFFF	4KB	Module
	0x4903 B000	0x4903 BFFF	4KB	L4 interconnect
GPTIMER7	0x4903 C000	0x4903 CFFF	4KB	Module
	0x4903 D000	0x4903 DFFF	4KB	L4 interconnect
GPTIMER8	0x4903 E000	0x4903 EFFF	4KB	Module
	0x4903 F000	0x4903 FFFF	4KB	L4 interconnect
GPTIMER9	0x4904 0000	0x4904 0FFF	4KB	Module
	0x4904 1000	0x4904 1FFF	4KB	L4 interconnect
Reserved	0x4904 2000	0x4904 FFFF	56KB	Reserved
GPIO2	0x4905 0000	0x4905 0FFF	4KB	Module
	0x4905 1000	0x4905 1FFF	4KB	L4 interconnect
GPIO3	0x4905 2000	0x4905 2FFF	4KB	Module
	0x4905 3000	0x4905 3FFF	4KB	L4 interconnect
GPIO4	0x4905 4000	0x4905 4FFF	4KB	Module
	0x4905 5000	0x4905 5FFF	4KB	L4 interconnect
GPIO5	0x4905 6000	0x4905 6FFF	4KB	Module
	0x4905 7000	0x4905 7FFF	4KB	L4 interconnect
GPIO6	0x4905 8000	0x4905 8FFF	4KB	Module
	0x4905 9000	0x4905 9FFF	4KB	L4 interconnect
Reserved	0x4905 A000	0x490F FFFF	664KB	Reserved

2.3.2.4 L4-Emulation Memory Space Mapping

The L4-Emu interconnect is an 8MB space composed of the L4-Emu interconnect configuration registers and module registers.

Table 2-6 describes the mapping of the registers for the L4-Emu interconnect.

Note: All memory spaces described as modules provide direct access to the module registers outside the L4-Emu interconnect. All other accesses are internal to the L4-Emu interconnect.

Table 2-6. L4-Emulation Memory Space Mapping

Device Name	Start Address (Hex)	End Address (Hex)	Size	Description
L4-Emu	0x5400 0000	0x547F FFFF	8MB	
Reserved	0x5400 0000	0x5400 3FFF	16KB	Reserved
Reserved	0x5400 4000	0x5400 5FFF	8KB	Reserved
L4-Emu configuration	0x5400 6000	0x5400 67FF	2KB	AP
	0x5400 6800	0x5400 6FFF	2KB	IP L4-Core
	0x5400 7000	0x5400 7FFF	4KB	LA
	0x5400 8000	0x5400 87FF	2KB	IP DAP
Reserved	0x5400 8800	0x5400 FFFF	30KB	Reserved
MPU emulation	0x5401 0000	0x5401 7FFF	16KB	Module
	0x5401 8000	0x5401 7FFF	4KB	L4 interconnect

Table 2-6. L4-Emulation Memory Space Mapping (continued)

Device Name	Start Address (Hex)	End Address (Hex)	Size	Description
TPIU	0x5401 9000	0x5401 9FFF	4KB	Module
	0x5401 A000	0x5401 AFFF	4KB	L4 interconnect
ETB	0x5401 B000	0x5401 BFFF	4KB	Module
	0x5401 C000	0x5401 CFFF	4KB	L4 interconnect
DAPCTL	0x5401 D000	0x5401 DFFF	4KB	Module
	0x5401 E000	0x5401 EFFF	4KB	L4 interconnect
SDTI	0x5401 F000	0x5401 FFFF	4KB	L4 interconnect
	0x5402 0000	0x544F FFFF	4992KB	Reserved
	0x5450 0000	0x5450 FFFF	4KB	SDTI module (configuration)
	0x5451 0000	0x545F FFFF	1984KB	Reserved
	0x5460 0000	0x546F FFFF	1MB	SDTI module (window)
Reserved	0x5470 0000	0x5470 5FFF	24KB	Reserved
Power and reset manager • Power manager • Reset manager (WKUP domain[1])	0x5470 6000	0x5470 7FFF	8KB	Module region A
	0x5470 8000	0x5470 87FF	2KB	Module region B[2]
	0x5470 8800	0x5470 8FFF	2KB	Reserved
	0x5470 9000	0x5470 9FFF	4KB	L4 interconnect
Reserved	0x5470 A000	0x5470 FFFF	24KB	Reserved
GPIO1 (WKUP domain[1])	0x5471 0000	0x5471 0FFF	4KB	Module
	0x5471 1000	0x5471 1FFF	4KB	L4 interconnect
Reserved	0x5471 2000	0x5471 3FFF	8KB	Reserved
WDTIMER2 (WKUP domain[1])	0x5471 4000	0x5471 4FFF	4KB	Module
	0x5471 5000	0x5471 5FFF	4KB	L4 interconnect
Reserved	0x5471 6000	0x5471 7FFF	8KB	Reserved
GPTIMER1 (WKUP domain[1])	0x5471 8000	0x5471 8FFF	4KB	Module
	0x5471 9000	0x5471 9FFF	4KB	L4 interconnect
Reserved	0x5471 A000	0x5471 FFFF	24KB	Reserved
32KTIMER (WKUP domain[1])	0x5472 0000	0x5472 0FFF	4KB	Module
	0x5472 1000	0x5472 1FFF	4KB	L4 interconnect
Reserved	0x5472 2000	0x5472 7FFF	24KB	Reserved
L4-Wakeup configuration (WKUP domain[1])	0x5472 8000	0x5472 87FF	2KB	Address/protection (AP)
	0x5472 8800	0x5472 8FFF	2KB	Initiator port (IP) L4-Core
	0x5472 9000	0x5472 9FFF	4KB	Link agent (LA)
	0x5472 A000	0x5472 A7FF	2KB	Initiator port (IP) L4-Emu
Reserved	0x5472 A800	0x547F FFFF	854KB	Reserved

(1) These modules are accessed through the L4-Wakeup interconnect (for emulation only).
(2) The registers mapped in this range are shadow registers of the first 2-KB region A [0x5470 6000 - 0x5470 67FF]. Regions A and B share the same port.

2.3.3 Register Access Restrictions

This section describes all modules and features in the high-tier device. To check availability of modules and features, see Section 1.5, *OMAP34xx Family*. In unavailable modules and features, the memory area is reserved, read is undefined, and write can lead to unpredictable behavior.

Table 2-7 gives the supported data access widths per module.

Table 2-7. Register Access Restrictions

Module	Allowed Access (Bits)
MPU subsystem	8/16/32
IVA2.2 subsystem	32
SGX	32
Camera ISP	8/16/32
Display subsystem	32
GPMC	8/16/32
SMS	8/16/32
SDRC	8/16/32
sDMA	8/16/32
HS USB HOST	32
USBTLL	32
USB - ULPI and UTMI registers	8
HS USB OTG	32
L3 interconnect	8/16/32
L4-Wakeup interconnect	8/16/32
L4-Core interconnect	8/16/32
Clock manager	32
Power and reset manager	32
System control module	8/16/32
ICR (chassis mode only)	32
32KTIMER	16/32
GPIO	8/16/32
GPTIMER	16/32
WDTIMER	16/32
I2C	8/16
HDQ/1-wire	32
McBSP	32
Sidetone	8/16/32
McSPI	8/16/32
UART	8/16/32
MMC/SD/SDIO	32
MAILBOX	8/16/32
MPU INTC	16/32
MODEM INTC (chassis mode only)	16/32

2.4 IVA2.2 Subsystem Memory Space Mapping

This section describes the hardware accelerators of IVA2.2. To check availability of these modules, see Section 1.5, *OMAP34xx Family*. In unavailable modules and features, the memory area is reserved, read is undefined, and write can lead to unpredictable behavior.

The device includes the high-performance Texas Instruments IVA2.2. For more information, see Chapter 14, *IVA2.2 Subsystem*.

This section describes how the IVA2.2 internal memories and registers are accessed through the L3 interconnect and by the IVA2.2 internal initiators (the digital signal processor [DSP] and the enhanced direct memory access [EDMA]).

Three views of the IVA2.2 subsystem memory space mapping are provided:

- L3 interconnect view: External view (subsystem memories and configuration registers) as seen by the MPU subsystem and most of the initiators of the platform through the L3 interconnect
- IVA2.2 DSP view: Internal view as seen by the DSP
- IVA2.2 EDMA view: Internal view as seen by the EDMA

Note: The IVA2.2 subsystem also contains a local interconnect with its own memory space mapping that can be accessed only by the DSP and the video accelerator and sequencer in the IVA2.2 subsystem. For more information about this video accelerator/sequencer local interconnect and its memory space mapping, see Chapter 14, *IVA2.2 Subsystem*.

2.4.1 IVA2.2 Subsystem Internal Memory and Cache Allocation

2.4.1.1 IVA2.2 Subsystem Memory Hierarchy

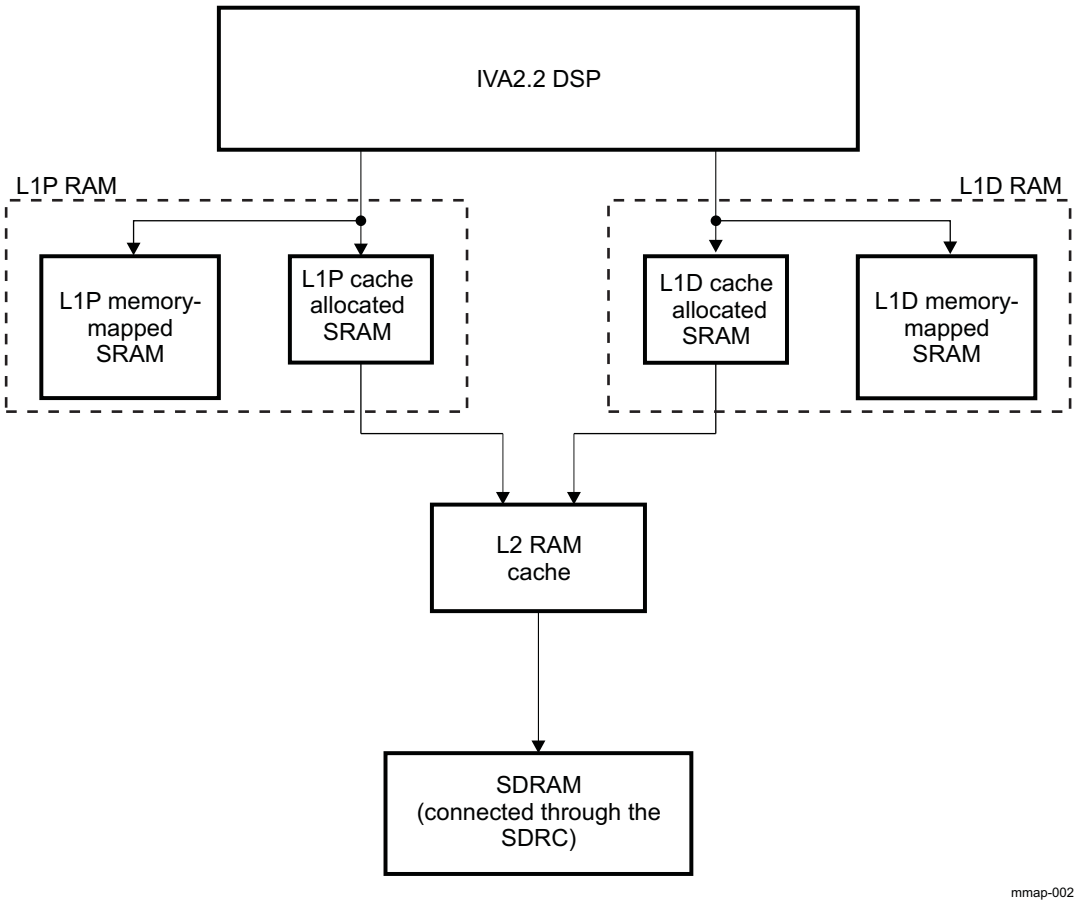
The IVA2.2 subsystem includes the following memory features:

- L1P (program)
  - 32KB configurable: Memory-mapped (default after reset) or direct-mapped cache—32-byte cache line
- L1D (data)
  - 32KB configurable: Memory-mapped (default after reset) or 2-way set associative cache—64-byte cache line
  - 48KB memory-mapped
- L2 (program and data)
  - 64KB configurable: Memory-mapped (default after reset) or 2-way set associative cache—128-byte cache line
  - 32KB memory-mapped
  - 16KB ROM

The local memories can be used as cache RAMs or memory-mapped RAMs, depending on the configuration of the different memory controllers in the IVA2.2 subsystem.

Figure 2-2 shows the memory hierarchy of the IVA2.2 subsystem.

Figure 2-2. IVA2.2 Subsystem Memory Hierarchy



2.4.1.2 IVA2.2 Cache Allocation

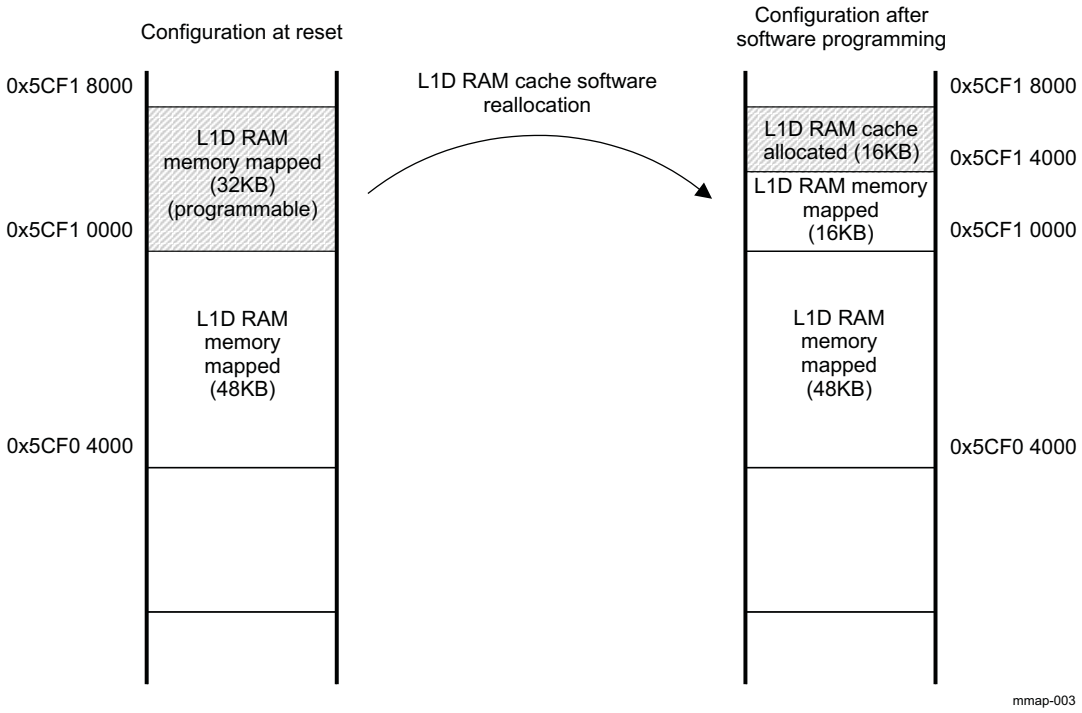
After reset, the L1P RAM is used as a 32KB memory-mapped RAM. The L1P RAM can be programmed in the C64x+ DSP program memory controller to allocate 0 (default), 4, 8, 16, or 32KB to cache. When 32KB are allocated to cache, there is no more memory-mapped L1P.

After reset, the L1D RAM is used as an 80KB memory-mapped RAM. The L1D RAM can be programmed in the C64x+ DSP data memory controller to allocate 0 (default), 4, 8, 16, or 32KB to cache. When 32KB are allocated to cache, 48KB are still allocated to the memory-mapped L1D.

After reset, the L2 is used as a 96KB memory-mapped RAM. The L2 can be programmed to allocate 0 (default), 32, or 64KB to cache. When 64KB are allocated to cache, 32KB are still allocated to memory-mapped L2.

Figure 2-3 is an example of the L1D RAM cache allocation, where 16KB are allocated to cache.

Figure 2-3. L1D RAM Cache Allocation Example (L3 Interconnect View)



2.4.2 DSP Access to L2 Memories

2.4.2.1 DSP Access to L2 ROM

The IVA2.2 subsystem contains 16KB of L2 ROM. The L2 ROM provides boot code.

When the L1P cache is configured to be inactive (default configuration), DSP program fetch accesses to the L2 ROM are accomplished directly, and thus suffer the L2 latency.

When the L1P cache is configured to be active, DSP program fetch accesses to L2 ROM are always serviced by the L1P cache controller, which partly hides the L2 latency.

2.4.2.2 DSP Access to L2 RAM

The IVA2.2 contains 96KB of L2 RAM. The L2 RAM can be configured to allocate up to 64KB to the L2 cache.

When the L1P and L1D caches are configured to be inactive (default configuration), DSP accesses to the L2 RAM are accomplished directly and thus suffer the L2 latency.

When the L1P and L1D RAM are configured to be active, DSP program accesses to L2 RAM are always serviced by the L1P cache controller (if code fetch) or the L1D cache controller (if data access), which partly hides the L2 latency

2.4.3 DSP and EDMA Access to Memories and Peripherals

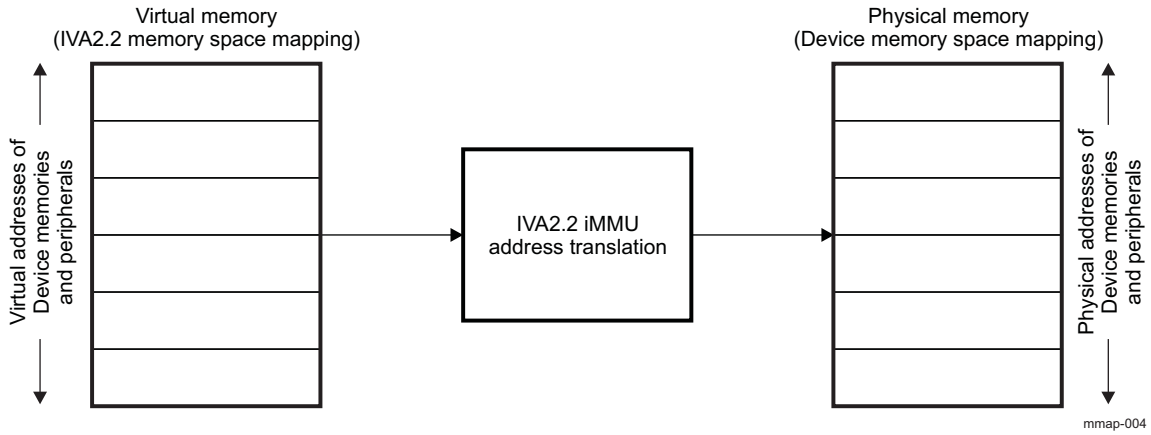
The IVA2.2 DSP and EDMA access the memories and peripherals using virtual addressing. This lets the DSP and EDMA access memories and peripherals in the same contiguous view, even when the memory is physically segmented. Table 2-9 and Table 2-10 give the address range where the DSP and the EDMA can access the memories and peripherals, respectively.

The IVA2.2 memory management unit (IVA2.2 iMMU) handles the virtual-to-physical address translation based on the software configuration (typically under control of the MPU subsystem).

Virtual addresses are issued by the initiator to the IVA2.2 iMMU. Using the translation look-aside buffer (TLB), the MMU translates the initiator virtual addresses into real physical addresses.

Figure 2-4 shows the relationship among physical addresses, virtual addresses, and the IVA2.2 iMMU.

Figure 2-4. IVA2.2 iMMU Address Translation



For more information about the MMU, see Chapter 8, *Memory Management Units*.

2.4.4 L3 Interconnect View of the IVA2.2 Subsystem Memory Space

Table 2-8 lists the IVA2.2 subsystem memory space mapping from the perspective of the MPU subsystem through the L3 interconnect.

Table 2-8. L3 Interconnect View of the IVA2.2 Subsystem Memory Space

Region Name	Start Address (Hex)	End Address (Hex)	Size	Description
Address space 0	**0x5C00 0000**	**0x5CFF FFFF**	**16MB**	
Reserved	0x5C00 0000	0x5C7D FFFF	8064KB	Reserved
L2 ROM	0x5C7E 0000	0x5C7E 3FFF	16KB	IVA2.2 internal memories
Reserved	0x5C7E 4000	0x5C7F 7FFF	80KB	Reserved
L2 RAM	0x5C7F 8000	0x5C7F FFFF	32KB	IVA2.2 internal memories
L2 RAM (cache)	0x5C80 0000	0x5C80 FFFF	64KB	IVA2.2 internal memories
Reserved	0x5C81 0000	0x5CDF FFFF	6080KB	Reserved
L1P RAM (cache)	0x5CE0 0000	0x5CE0 7FFF	32KB	IVA2.2 internal memories
Reserved	0x5CE0 8000	0x5CF0 3FFF	1008KB	Reserved
L1D RAM	0x5CF0 4000	0x5CF0 FFFF	48KB	IVA2.2 internal memories
L1D RAM (cache)	0x5CF1 0000	0x5CF1 7FFF	32KB	IVA2.2 internal memories
Reserved	0x5CF1 8000	0x5CFF FFFF	928KB	Reserved
Address space 1	**0x5D00 0000**	**0x5DFF FFFF**	**16MB**	
MMU registers	0x5D00 0000	0x5D00 0FFF	4KB	IVA2.2 iMMU module
Reserved	0x5D00 1000	0x5DFF FFFF	16,380KB	Reserved
Address space 2	**0x5E00 0000**	**0x5EFF FFFF**	**16MB**	
Video coprocessor and sequencer	0x5E00 0000	0x5E0F FFFF	1MB	IVA2.2 video modules
Reserved	0x5E10 0000	0x5EFF FFFF	15MB	Reserved

2.4.5 DSP View of the IVA2.2 Subsystem Memory Space

Table 2-9 lists the IVA2.2 subsystem memory space mapping internally from the perspective of the DSP.

Table 2-9. DSP View of the IVA2.2 Subsystem Memory Space

Region Name	Start Address (Hex)	End Address (Hex)	Size (KB)	Description
Reserved	0x0000 0000	0x007D FFFF	8064	Reserved
L2 ROM	0x007E 0000	0x007E 3FFF	16	IVA2.2 internal memories
Reserved	0x007E 4000	0x007F 7FFF	80	Reserved
L2 RAM	0x007F 8000	0x007F FFFF	32	IVA2.2 internal memories
L2 RAM (cache)	0x0080 0000	0x0080 FFFF	64	IVA2.2 internal memories
Reserved	0x0081 0000	0x00DF FFFF	6080	Reserved
L1P RAM (cache)	0x00E0 0000	0x00E0 7FFF	32	IVA2.2 internal memories
Reserved	0x00E0 8000	0x00F0 3FFF	1008	Reserved
L1D RAM	0x00F0 4000	0x00F0 FFFF	48	IVA2.2 internal memories
L1D RAM (cache)	0x00F1 0000	0x00F1 7FFF	32	IVA2.2 internal memories
Reserved	0x00F1 8000	0x017F FFFF	9120	Reserved
C64x+ interrupt selector	0x0180 0000	0x0180 FFFF	64	C64x+ DSP interrupt controller
C64x+ PDC	0x0181 0000	0x0181 0FFF	4	C64x+ DSP power-down controller
C64x+ protection ID	0x0181 1000	0x0181 1FFF	4	C64x+ DSP protection ID
C64x+ revision ID	0x0181 2000	0x0181 2FFF	4	C64x+ DSP revision ID
Reserved	0x0181 3000	0x0181 FFFF	52	Reserved
C64x+ EMC	0x0182 0000	0x0182 FFFF	64	C64x+ DSP extended memory controller
Reserved	0x0183 0000	0x0183 FFFF	64	Reserved
C64x+ memory system	0x0184 0000	0x0184 FFFF	64	Memory controller control registers
Reserved	0x0185 0000	0x01BF FFFF	3776	Reserved
TPCC configuration	0x01C0 0000	0x01C0 FFFF	64	DMA transfer engine control registers
TPTC0 configuration	0x01C1 0000	0x01C1 03FF	1	DMA transfer scheduler 0 control registers
TPTC1 configuration	0x01C1 0400	0x01C1 07FF	1	DMA transfer scheduler 1 control registers
Reserved	0x01C1 0800	0x01C1 FFFF	62	Reserved
SYSC configuration	0x01C2 0000	0x01C2 0FFF	4	SYSC module control registers
WUGEN configuration	0x01C2 1000	0x01C2 1FFF	4	Wake-up generator control registers
Reserved	0x01C2 2000	0x0FFF FFFF	233,336	Reserved
Reserved	0x1000 0000	0x107D FFFF	8064	Reserved
L2 ROM(1)	0x107E 0000	0x107E 3FFF	16	IVA2.2 internal memories
Reserved	0x107E 4000	0x107F 7FFF	80	Reserved
L2 RAM(1)	0x107F 8000	0x107F FFFF	32	IVA2.2 internal memories
L2 RAM (cache)(1)	0x1080 0000	0x1080 FFFF	64	IVA2.2 internal memories
Reserved	0x1081 0000	0x10DF FFFF	6080	Reserved
L1P RAM (cache)(1)	0x10E0 0000	0x10E0 7FFF	32	IVA2.2 internal memories
Reserved	0x10E0 8000	0x10F0 3FFF	1008	Reserved
L1D RAM(1)	0x10F0 4000	0x10F0 FFFF	48	IVA2.2 internal memories
L1D RAM (cache)2(1)	0x10F1 0000	0x10F1 7FFF	32	IVA2.2 internal memories

(1) IVA2.2 internal memories are reachable in the [0x007E 0000-0x00F1 7FFF] and [0x107E 0000-0x10F1 7FFF] (aliasing) ranges.

Table 2-9. DSP View of the IVA2.2 Subsystem Memory Space (continued)

Region Name	Start Address (Hex)	End Address (Hex)	Size (KB)	Description
Reserved	0x10F1 8000	0x10FF FFFF	928	Reserved
memories and peripherals[2]	0x1100 0000	0xFFFF FFFF	3,915,776	Controlled by the IVA2.2 MMU to access memories and peripherals external to the IVA2.2 subsystem

(2) For more information, see Chapter 14, *IVA2.2 Subsystem.*

2.4.6 EDMA View of the IVA2.2 Subsystem Memory Space

Table 2-10 shows the IVA2.2 subsystem memory space mapping from the perspective of the EDMA.

Table 2-10. EDMA View of the IVA2.2 Subsystem Memory Space

Region Name	Start Address (Hex)	End Address (Hex)	Size (KB)	Description
Reserved	0x0000 0000	0x0008 FFFF	576	Reserved
SEQ CFG	0x0009 0000	0x0009 07FF	2	Video sequencer configuration registers
Reserved	0x0009 0800	0x0009 3FFF	14	Reserved
SEQ DMEM	0x0009 4000	0x0009 4FFF	4	Video sequencer data memory
Reserved	0x0009 5000	0x0009 7FFF	12	Reserved
SEQ IMEM	0x0009 8000	0x0009 9FFF	8	Video sequencer instruction memory
Reserved	0x0009 A000	0x0009 BFFF	8	Reserved
Video sysc	0x0009 C000	0x0009 CFFF	4	Video system controller
Reserved	0x0009 D000	0x0009 FFFF	12	Reserved
iME CFG	0x000A 0000	0x000A 0FFF	4	Improved motion estimation configuration registers
iLF CFG	0x000A 1000	0x000A 1FFF	4	Variable length coding decoding configuration registers
Reserved	0x000A 2000	0x000F 7FFF	344	Reserved
Local interconnect	0x000F 8000	0x000F BFFF	16	Video accelerator/sequencer local interconnect
Reserved	0x000F C000	0x000F FFFF	16	Reserved
Reserved	0x0010 0000	0x107D FFFF	269,184	Reserved
L2 ROM	0x107E 0000	0x107E 3FFF	16	IVA2.2 internal memories
Reserved	0x107E 4000	0x107F 7FFF	80	Reserved
L2 RAM	0x107F 8000	0x107F FFFF	32	IVA2.2 internal memories
L2 RAM (cache)	0x1080 0000	0x1080 FFFF	64	IVA2.2 internal memories
Reserved	0x1081 0000	0x10DF FFFF	6080	Reserved
L1P RAM (cache)	0x10E0 0000	0x10E0 7FFF	32	IVA2.2 internal memories
Reserved	0x10E0 8000	0x10F0 3FFF	1008	Reserved
L1D RAM	0x10F0 4000	0x10F0 FFFF	48	IVA2.2 internal memories
L1D RAM (cache)	0x10F1 0000	0x10F1 7FFF	32	IVA2.2 internal memories
Reserved	0x10F1 8000	0x10FF FFFF	928	Reserved
memories and peripherals[1]	0x1100 0000	0xFFFF FFFF	3,915,776	Controlled by the IVA2.2 MMU to access memories and peripherals external to the IVA2.2 subsystem

(1) For more information, see Chapter 14, *IVA2.2 Subsystem*.

MPU Subsystem

This chapter describes the microprocessor unit (MPU) subsystem in the OMAP34xx multimedia device.

Topic **Page**

3.1 MPU Subsystem Overview 214
3.2 MPU Subsystem Integration 216
3.3 MPU Subsystem Functional Description 225
3.4 MPU Subsystem Basic Programming Model 231

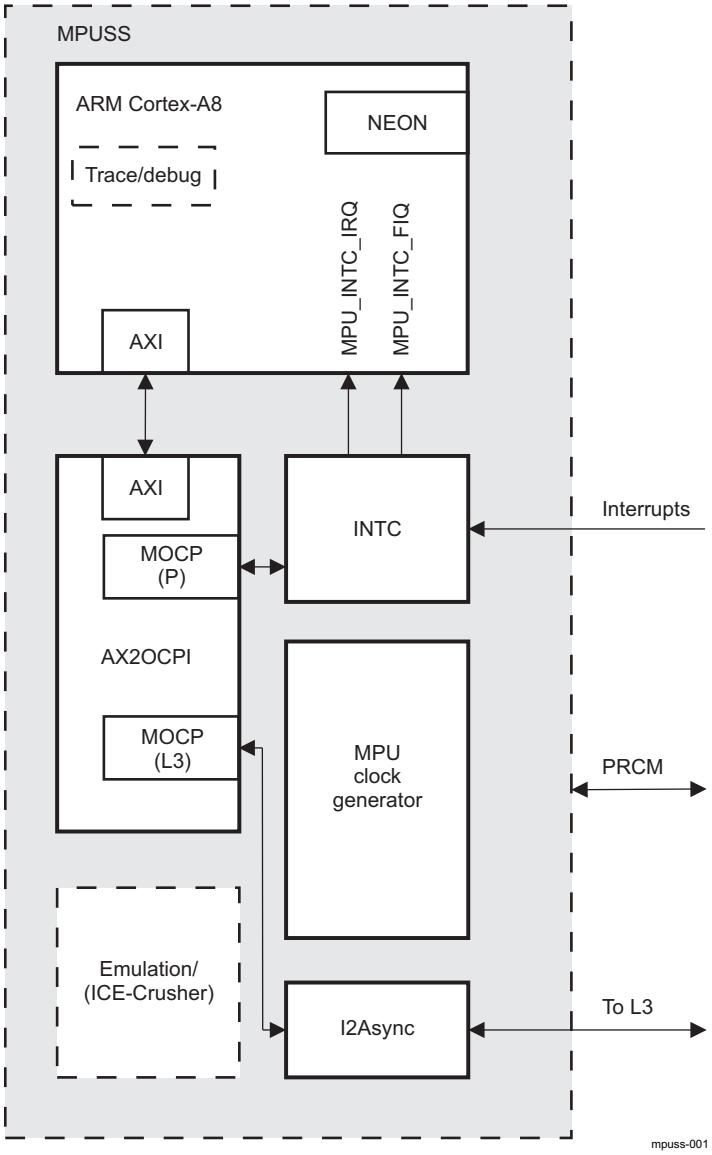
3.1 MPU Subsystem Overview

3.1.1 Introduction

The MPU subsystem of the device handles transactions among the ARM core, the L3 interconnect, and the interrupt controller (INTC).

The MPU subsystem is a hard macro that integrates the ARM subchip with additional logic for protocol conversion, emulation, interrupt handling, and debug enhancements. Figure 3-1 is a high-level block diagram of the MPU subsystem.

Figure 3-1. MPU Subsystem Overview



3.1.2 Features

The MPU subsystem integrates the following:

- ARM subchip
  - Public ARM Cortex™-A8core
    - Revision r1p3 for OMAP34xx ES3.1
    - Revision r1p7 for OMAP34xx ES3.1.2
  - ARM version 7 ISA: Standard ARM instruction set + Thumb-2™, Jazelle® RCT Java accelerator, and media extensions
  - Neon single instruction, multiple data (SIMD) coprocessor (VFP light + media streaming instructions)
  - Cache memories
    - Level 1: 16KB instruction and 16KB data caches - 4 ways associative, 64 bytes/line
    - Level 2: See Section 1.5, *OMAP34xx Family*
  - Emulation/debug
- INTC of 96 synchronous level-sensitive interrupt lines (For details, see Chapter 10, *Interrupt Controller*.)
- AXI2OCP bridge between ARM AXI bus, L3 master open-core protocol (OCP) bus, and INTC master OCP bus
- MPU clock generator: Clock generation module that generate clocks, power modes, and idle and active acknowledge signals
- Debug, trace, and emulation features: ICECrusher™, embedded trace macrocell (ETM), advanced peripheral bus (APB) modules. Cortex-A8 MPU implements an APB slave interface that allows access to ETM, ICECrusher, and debug registers.

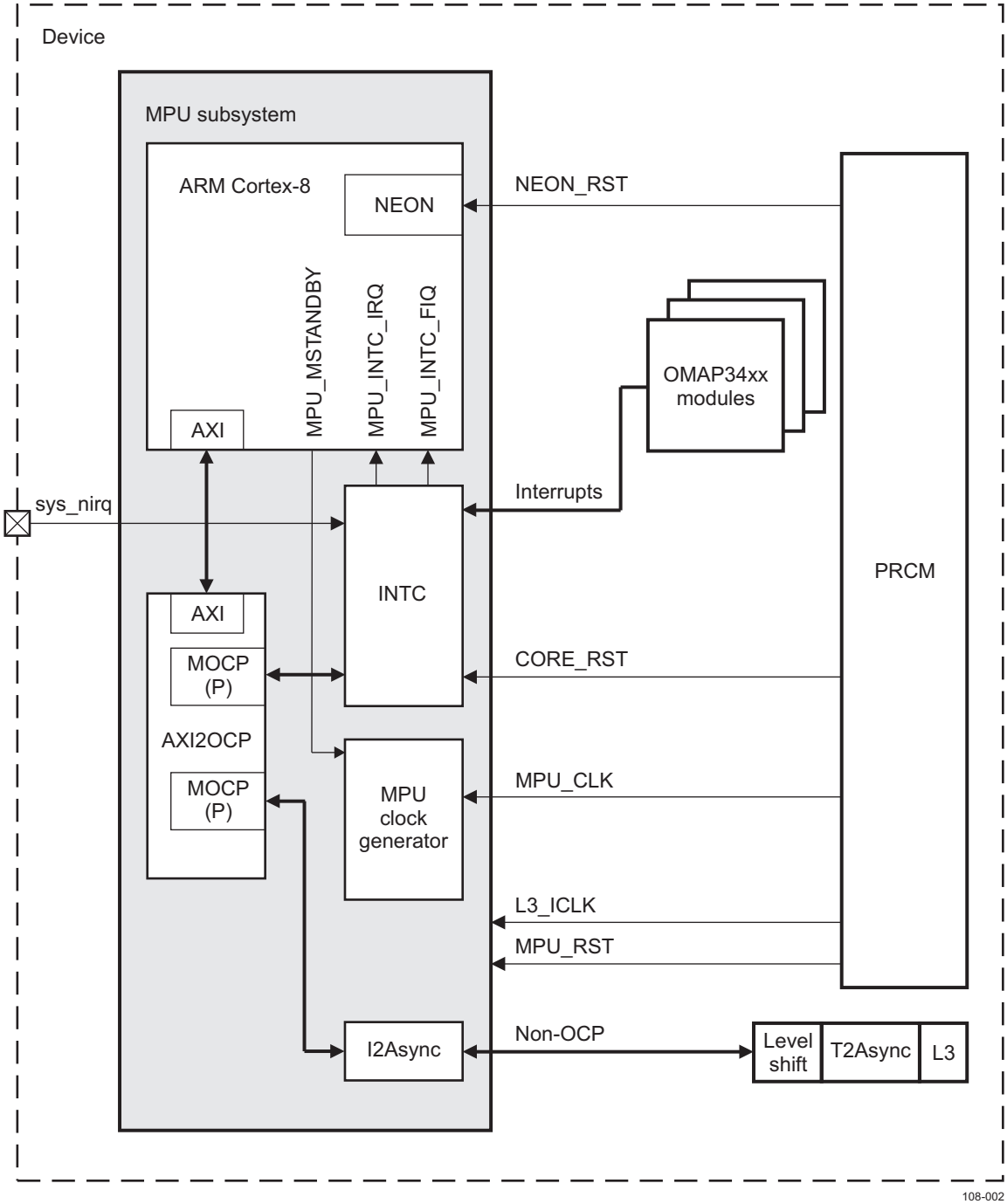
3.2 MPU Subsystem Integration

The MPU subsystem integrates the following group of submodules:

- Public ARM Cortex-A8 MPU: Provides a high processing capability, including the Neon technology for mobile multimedia acceleration. The ARM communicates through an AXI bus with the AXI2OCP bridge and receives interrupts from the MPU INTC.
- INTC: Handles module interrupts (for details, see Chapter 10, *Interrupt Controller*).
- AXI2OCP bridge: Allows communication among the ARM (AXI), the INTC (OCP), and the modules (OCP L3).
- I2Async bridge: An asynchronous bridge interface providing an asynchronous OCP-to-OCP interface. This interface is between the AXI2OCP bridge in the MPU subsystem and the T2Async bridge external to the MPU subsystem. The T2Async bridge connects to OCP L3.
- MPU clock generator: Provides clocks to internal modules of the MPU subsystem; fed by the MPU digital phase-locked loop (DPLL) of the power, reset, and clock management (PRCM) module of the device. The MPU DPLL generates the clock for the ARM Cortex-A8 CPU and the Cortex-A8 MPU subsystem logic. The power, reset, and MPU DPLL source clock are generated from the device PRCM module.

Figure 3-2 shows the signals that interface with the external modules.

Figure 3-2. MPU Subsystem Integration Overview



Note: Some debug, trace, and emulation features are implemented in the MPU subsystem. This chapter includes only clock/reset inputs and power-management aspects for these features.

3.2.1 MPU Subsystem Clock and Reset Distribution

3.2.1.1 Clock Distribution

The MPU subsystem includes a clock generator block that supplies clocks for the modules in the MPU subsystem. It is fed by the MPU_CLK clock from the PRCM module.

All major modules in the MPU subsystem are clocked at half the frequency of the ARM core. The divider of the output clock can be programmed with the PRCM.CM_CLKSEL2_PLL_MPU[4:0] MPU_DPLL_CLKOUT_DIV bit field; the frequency is relative to the ARM core. For details see Chapter 4, *Power, Reset, and Clock Management (PRCM) module* . The clock generator generates the following functional clocks:

- **ARM (ARM_FCLK):** This is the core clock. It is the base fast clock that is routed internally to the ARM logic and internal RAMs, including Neon, L2 cache, the ETM core (emulation), and the ARM core. It runs at one half the frequency of the MPU_CLK when DPLL1 is locked, and runs at the same frequency as the MPU_CLK when DPLL1 is bypassed.
- **AXI2OCP clock (AXI_FCLK):** This clock is half the frequency of the ARM clock (ARM_FCLK). The OCP interface thus performs at one half the frequency of ARM.
- **INTC functional clock (MPU_INTC_FCLK):** This clock, which is part of the INTC module, is half the frequency of the ARM clock (ARM_FCLK).
- **ICE-Crusher functional clock (ICECRUSHER_FCLK):** ICECrusher clocking operates on the APB interface, using the ARM core clocking.
- **I2Async clock (I2ASYNC_FCLK):** This clock is half the frequency of the ARM clock (ARM_FCLK). It matches the OCP interface of the AXI2OCP bridge.

Note: The second half of the asynchronous bridge (T2ASYNC) is clocked directly by the PRCM module with the core clock. T2ASYNC is not part of the MPU subsystem.

Emulation clocking: Except for the ICECrusher functional clock, which is provided by the MPU DLL, the emulation modules in the MPU subsystem are not generated by the MPU subsystem DPLL, but by an EMU DPLL. These clocks (EMU_CLOCKS) are distributed by the PRCM module, are asynchronous with the ARM core clock (ARM_FCLK) and can run at a maximum of 1/3 the ARM core clock.

Figure 3-3 shows the MPU subsystem clocking scheme.

Figure 3-3. MPU Subsystem Clocking Scheme

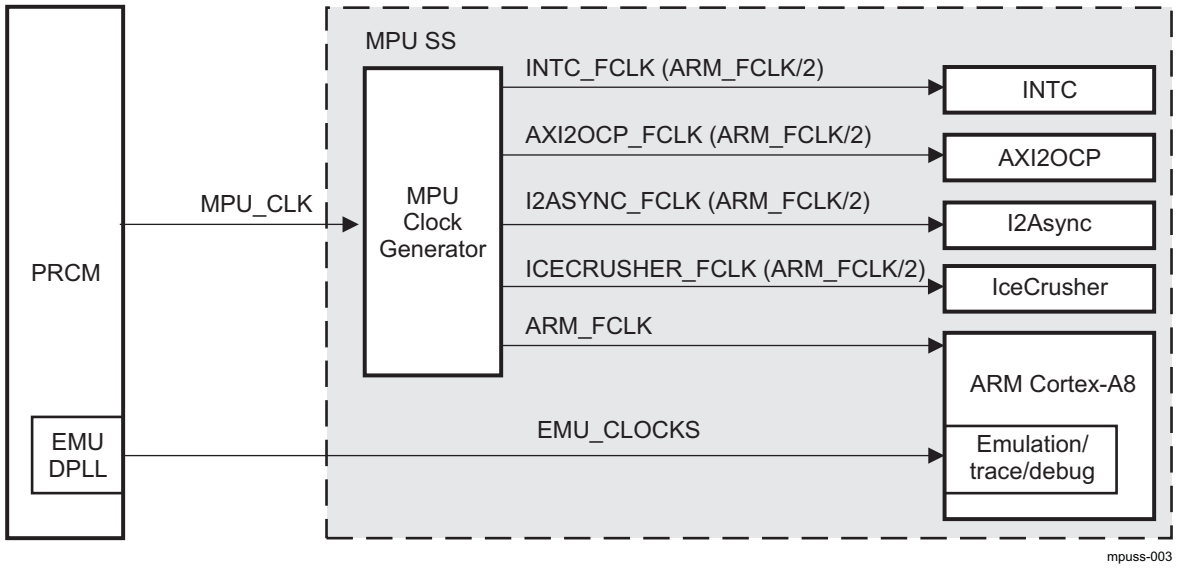


Table 3-1 lists the clocks generated in the MPU subsystem by the MPU clock generator.

Table 3-1. MPU Clock Generator Clock Signals

Signal Name	I/O	Interface	Comments
MPU_CLK	I	PRCM	MPU DPLL clock out
ARM_FCLK	O	ARM	ARM functional clock
MPU_INTC_FCLK	O	MPU INTC	MPU INTC functional clock
I2ASYNC_FCLK	O	I2Async	I2Async functional clock
AXI_FCLK	O	AXI2OCP	AXI2OCP functional clock
ICECRUSHER_FCLK	O	ICE-Crusher	ICECrusher functional clock

3.2.1.2 Reset Distribution

Resets to the MPU subsystem are provided by the PRCM module and controlled by the clock generator module. There are as many reset signals as power domains. For details about power domains, see Section 3.3.2.1. Figure 3-4 shows the reset scheme of the MPU subsystem.

Figure 3-4. MPU Subsystem Reset Scheme

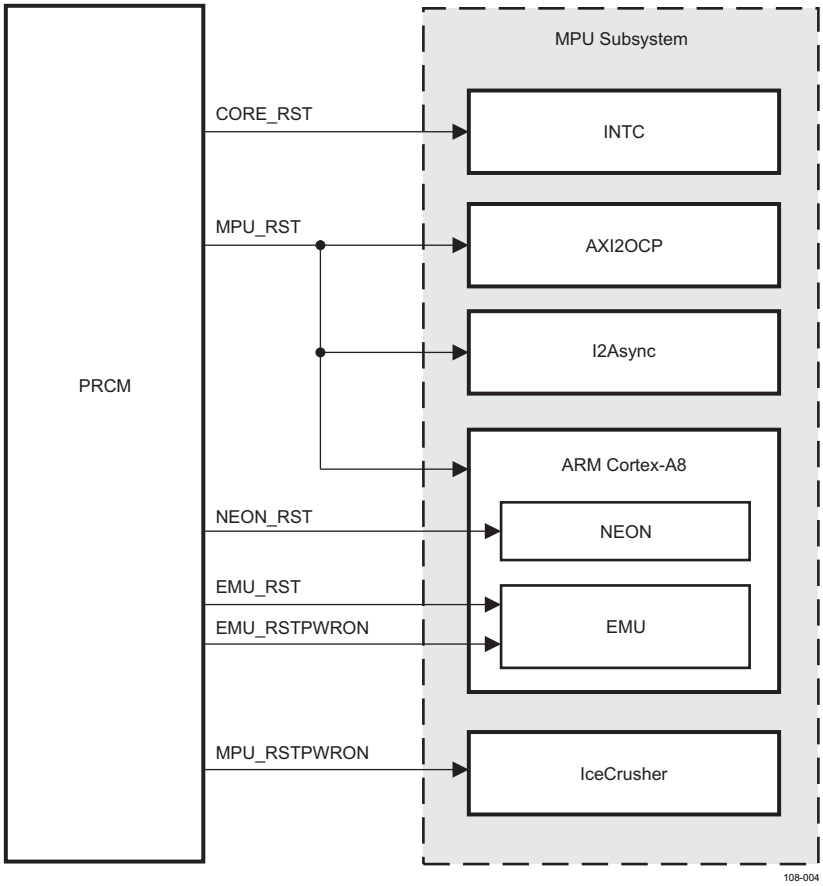


Table 3-2. MPU Subsystem Reset Signals

Signal Name	I/O	Interface	Comments
MPU_RST	I	PRCM	MPU power domain reset
Neon_RST	I	PRCM	Neon power domain reset
CORE_RST	I	PRCM	CORE power domain reset
MPU_RSTPWRON	I	PRCM	ICECrusher reset. It is active only on cold reset.
EMU_RST	I	PRCM	Emulation interconnect reset

Table 3-2. MPU Subsystem Reset Signals (continued)

EMU_RSTPWRON	I	PRCM	Emulation modules reset

For details about clocks, resets, and power domains, see Chapter 4, *Power, Reset, and Clock Management*.

3.2.2 ARM Subchip

3.2.2.1 ARM Overview

The public ARM Cortex-A8 processor incorporates the technologies available in the ARM7 architecture. These technologies include Neon for media and signal processing and Jazelle RCT for acceleration of realtime compilers, Thumb-2 technology for code density and the VFPv3 floating-point architecture.

For details, see the public ARM Cortex-A8 Technical Reference Manual.

3.2.2.2 ARM Description

3.2.2.2.1 Public ARM Cortex-A8 Instruction, Data, and Private Peripheral Port

The AXI bus interface is the main interface to the ARM system bus. It performs L2 cache fills and noncacheable accesses for instructions and data. The AXI interface supports 64-bit wide input and output data buses. It supports multiple outstanding requests on the AXI bus and a wide range of bus clock-to-core clock ratios. The bus clock is synchronous with the core clock.

See the public ARM Cortex-A8 Technical Reference Manual for a complete programming model of the transaction rules (ordering, posting, and pipeline synchronization) that are applied depending on the memory region attribute associated with the transaction destination address.

3.2.2.2.2 MPU Subsystem Features

Table 3-3 is a list of main functionalities of the ARM core supported in the MPU subsystem for the device. The MPU subsystem implements the ARM7™ instruction set architecture.

Table 3-3. ARM Core Key Features

Feature	Comment
ARM version 7 ISA	Standard ARM instruction set + Thumb-2, Jazelle RCT Java accelerator, and media extensions. Backward-compatible with previous ARM ISA versions.
L1 Icache and Dcache	16KB, 4-way, 64-byte cache line, and 128-bit interface. Note: L1 memories cannot be put in retention mode.
L2 Cache	The L2 cache and cache controller are embedded in the ARM core. For size, see Section 1.5, *OMAP34xx Family*.
TLB	Fully associative and separate ITLB with 32 entries and DTLB with 32 entries
CoreSight ETM	The CoreSight ETM is embedded in the ARM core. The 4KB buffer (ETB) exists at the device level.
Branch target address cache	512 entries
Enhanced Memory Management Unit	Mapping sizes are 4KB, 64KB, 1MB, and 16MB. ARM MMU adds extended physical address ranges.
Neon	Enhances throughput for media workloads and VFP-Lite support
Tightly coupled memory	Not present
AXI Bus	Separate 64-bit input and 64-bit output data buses. The public ARM Cortex-A8 has a single AXI bus interface. The AXI interface is shared by instruction fetches, data accesses, peripheral accesses and PLE (on-chip preload engine, previously known as DMA) accesses. DMA is available.

Table 3-3. ARM Core Key Features (continued)

Feature	Comment
Low interrupt latency	Interrupt request (IRQ) and fast interrupt request (FIQ) are directly connected to ARM and INTC is before L3.
Vectored Interrupt Controller Port	Not used
JTAG based debug	Supported through debug access port (DAP)
Trace support	Supported through trace port interface unit (TPIU)
External coprocessor	Not supported

For more information, see the public *ARM Cortex-A8 Technical Reference Manual*.

3.2.2.3 Clock, Reset, and Power Management

3.2.2.3.1 Clocks

Table 3-4 shows the ARM functional clock. For configuration, see Chapter 4, *Power, Reset, and Clock Management*.

Table 3-4. MPU Subsystem Clock Signal

Signal Name	I/O	Interface	Comments
ARM_FCLK	I	MPU Clock Generator	Functional clock

3.2.2.3.2 Reset

Table 3-5 lists the resets for the ARM. They include the power domain reset, Neon_RST, which resets the Neon module of the ARM subchip only, and MPU_RST, which resets the rest of the ARM subchip and the AXI2OCP and I2Async bridges.

Table 3-5. ARM Reset Signals

Signal Name	I/O	Interface	Comments
MPU_RST	I	PRCM	MPU power domain reset
Neon_RST	I	PRCM	Reset Neon only
EMU_RST	I	PRCM	Emulation interconnect reset
EMU_RSTPWRON	I	PRCM	Emulation modules reset

3.2.2.3.3 Power Management

For details, see Section 3.3.2.

3.2.3 AXI2OCP and I2Async Bridges

3.2.3.1 Bridges Overview

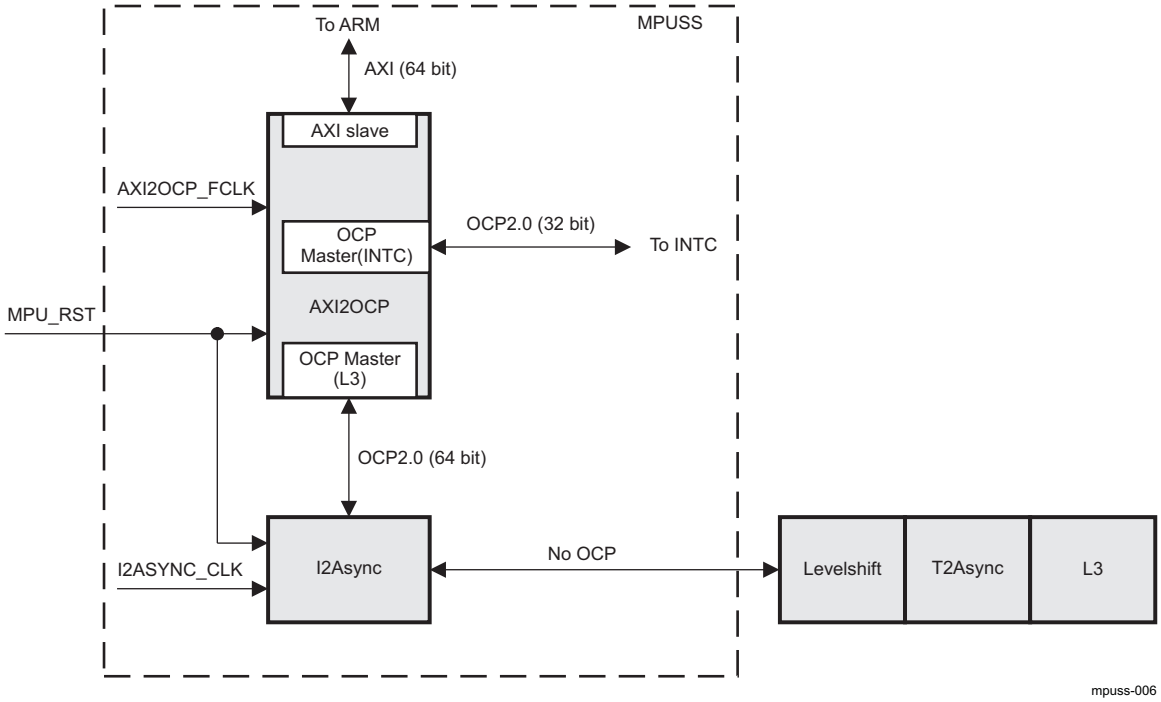
The AXI2OCP bridge connects the AXI bus on the ARM MPU to the OCP native L3 interconnect and INTC. It converts between AXI and OCP protocols and maintains a mapping of AXI tags to the OCP thread ID. This bridge is responsible for some minimal address decoding to determine where to forward requests between L3 and INTC.

Bridging to the L3 is accomplished through an asynchronous interface involving the I2Async and T2Async modules. The I2Async module in the MPU subsystem has an OCP port that is asynchronously transferred to the T2Async module and routed to the L3. T2Async is outside the MPU subsystem.

Note: The interface between I2Async and T2Async is not an OCP protocol.

Figure 3-5 is an overview of the AXI2OCP and the L3 bridges.

Figure 3-5. Bridges Overview



3.2.3.2 AXI2OCP Description

The AXI2OCP bridge key features are:

- Connection to the L3 interconnect through a 64-bit OCP port. Address bus is 32-bit.
- Connection to the INTC through a 32-bit OCP port (only single transactions are supported)
- Support of single-request-multiple-data (data handshaking) burst mode to pipeline requests to L3. Bursts with width less than 64 bits are converted as single requests on L3. This can affect system performance. The INTC does not support burst transfers
- Support of multiple outstanding requests
- Remapping of the AXI tags to five OCP threads:
  - Instruction fetch (Thread_IR)
  - Cacheable data read (Thread_CR)
  - Cacheable data write (Thread_CW)
  - Noncacheable data read (Thread_DR)
  - Noncacheable data write (Thread_DW)
- Interface to the L3 through the asynchronous bridge (I2Async)
- Emulation and support of boot-mode translation
- Translation of exclusive accesses to nonexclusive read/write in the bridge
- Boot mode address translation
- Force write nonposted support
- Debug related: Ready fail, force ready
- Interconnect attach neutralizer
- Narrow burst support
- Time-out counter for nonresponsive slaves on OCP

3.2.3.3 Clocks, Reset, and Power Management

3.2.3.3.1 Clocks

Table 3-6 lists the bridge functional clocks.

Table 3-6. Bridge Clock Signals

Signal Name	I/O	Interface	Comments
AXI2OCP_FCLK	I	MPU Clock Generator	AXI2OCP functional clock
I2ASYNC_FCLK	I	MPU Clock Generator	I2Async functional clock

3.2.3.3.2 Reset

Table 3-7 lists the bridge reset. It is a power domain reset that also resets the ARM.

Table 3-7. MPU Subsystem Reset Signal

Signal Name	I/O	Interface	Comments
MPU_RST	I	PRCM	MPU power domain reset

3.2.3.3.3 Power Management

For details, see Section 3.3.2.

3.2.4 Interrupt Controller

For information, see Chapter 10, *Interrupt Controllers*.

3.2.4.1 Clocks

Table 3-8 lists the INTC clocks.

Table 3-8. Bridge Clock Signals

Signal Name	I/O	Interface	Comments
MPU_INTC_FCLK	I	MPU Clock Generator	INTC functional clock
MPU_INTC_ICLK	I	OCP Clock	INTC interface clock

3.2.4.2 Reset

Table 3-9 lists the reset of the INTC. It is a power domain reset that resets the entire CORE power domain. For details, see Chapter 10, *Interrupt Controller* and Chapter 4, *Power, Reset, and Clock Management*.

Table 3-9. MPU Subsystem Reset Signal

Signal Name	I/O	Interface	Comments
CORE_RST	I	PRCM	CORE power domain reset

3.2.4.3 Power Management

See Chapter 10, *Interrupt Controller*, and Chapter 4, *Power, Reset, and Clock Management*.

3.3 MPU Subsystem Functional Description

3.3.1 Interrupts

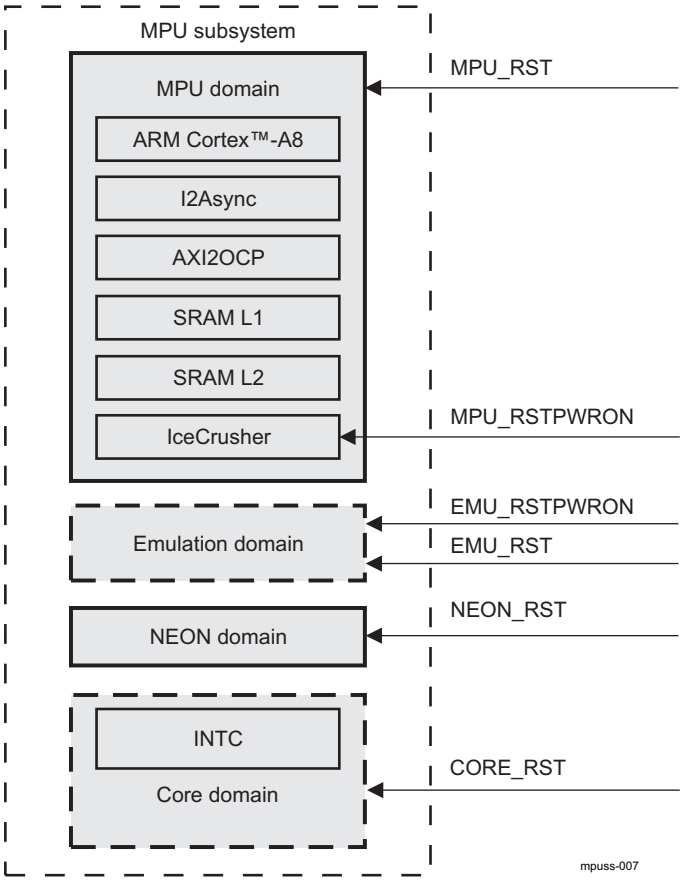
The MPU INTC is connected to the MPU through the AXI2OCP bridge. It runs at half-processor speed. The INTC prioritizes all service requests from the system peripherals and generates an IRQ or an FIQ to the MPU, depending on the INTC programming. The INTC handles only interrupts directed to the MPU subsystem. A maximum of 96 requests can be steered/prioritized as MPU FIQ or IRQ interrupt requests. For details, see Chapter 10, *Interrupt Controller*.

3.3.2 Power Management

3.3.2.1 Power Domains

The MPU subsystem is divided into five power domains controlled by the PRCM module, as shown in Figure 3-6. The EMU and CORE power domains are not fully embedded in the MPU subsystem.

Figure 3-6. MPU Subsystem Power Domain Overview



Power-management requirements at the device level govern power domains for the MPU subsystem.

The device-level power domains are directly aligned with voltage domains and thus can be represented as a cross reference to the different voltage domains. Table 3-10 shows the power domains of the MPU subsystem and the modules inside it.

Table 3-10. Overview of the MPU Subsystem Power Domain

Functional Power Domain	Physical Power Domain per System/Module
MPU subsystem domain	ARM, AXI2OCP, I2Asynch Bridge, ARM L1 and L2 periphery logic and array, ICECrusher, ETM, APB modules
MPU Neon domain	ARM Neon accelerator
CORE domain	MPU INTC
EMU domain	EMU (ETB, DAP)

Note: L1 and L2 array memories have separate control signals to the MPU subsystem; thus are directly controlled by the PRCM module.

For information about the physical power domains and the voltage domains, see Chapter 4, *Power, Reset, and Clock Management.*

3.3.2.2 Power States

Each power domain can be driven by the PRCM module in four power states, depending on the functional mode required by the user.

Table 3-11. MPU Power States

Power State	Logic Power	Memory Power	Clocks	Memory State Retention
ACTIVE	On	On or Off	On (at least one clock)	All
INACTIVE	On	On or Off	Off	All
RETENTION	On or Off	On or Off	Off (all clocks)	All or part
OFF	Off	Off	Off (all clocks)	None

For each power domain, the PRCM module manages all transitions by controlling domain clocks, domain resets, domain logic power switches, memory power switches, and memory retention. The MPU DPLL internally synchronizes the internal clocks, resets, and switches.

3.3.2.3 Power Modes

MPU DPLL power modes:

The PRCM.CM_AUTOIDLE_PLL_MPU[2:0] AUTO_MPU_DPLL bit field lets the MPU DPLL be put in an auto-idle mode when set to 0x1. In this mode, the MPU DPLL is automatically put in low power stop mode when the MPU clock is not required. It is also restarted automatically. Table 3-12 describes the power modes of the MPU DPLL in auto-idle or manual mode. The manual modes (locked and low-power bypass) can be configured by the PRCM.CM_CLKEN_PLL_MPU[2:0] EN_MPU_DPLL bit field. The status of the MPU DPLL clock activity can be checked with the PRCM.CM_IDLEST_PLL_MPU[0] ST_MPU_CLK bit.

Table 3-12. MPU DPLL Power Modes

Mode	System Input Clock	Clock Output	DPLL Power State	Condition
Locked	ON	ON	ON	Software request (manual) or MPU wakes up (auto)
Low-power bypass	ON	ON	ON	Software request (manual) or on global reset release (auto)
Stop low power	OFF	OFF	ON	MPU is RET or OFF (auto).
OFF	OFF	OFF	OFF	Device is OFF (auto).

The MPU DPLL power domain is switched off automatically by the PRCM module only when the device enters OFF mode.

MPU subsytem power modes:

The major part of the MPU subsystem belongs to the MPU power domain. Modules in this power domain can be off when the ARM processor is in OFF or standby mode. IDLE/WAKEUP control is managed by the clock generator block, but initiated by the PRCM module. The MPU standby status can be checked with the PRCM.CM_IDLEST_MPU[0] ST_MPU bit.

For the MPU to be on, the core (referred to here as the device core) power must be on.

Device power management does not allow INTC to go to OFF state when the MPU domain is on (active or one of the retention modes).

The Neon core is in independent power off mode when not in use. Enabling and disabling of Neon can be controlled by software.

CAUTION

The MPU L1 cache memory does not support retention mode, and its array switch is controlled with the MPU logic. For compliance, the L1 retention control signals exist at the PRCM boundary, but are not used. The ARM L2 can be put into retention independently of the other domains.

MPU retention modes are:

- Open switch retention (OSwR)
- Closed switch retention (CSwR)

These modes are described in Table 3-13.

Table 3-13. MPU Retention Modes

Name	Mode	ARM Logic	L1	L2
Dormant	OSwR	OFF	OFF	RET
RET	CSwR	ON	ON	RET

Table 3-14 lists the supported operational power modes. All other combinations are illegal. The ARM L2, Neon, and ETM/debug can be powered up/down independently. The APB/ATB ETM/Debug column refers to all three features: ARM emulation, trace, and debug.

Table 3-14. MPU Subsystem Operation Power Modes

Mode	MPU and ARM Core Logic	ARM L2 RAM	Neon	OMAP Core INTC	APB/ATB ETM/Debug	Comments
1	Active	Active	Active	Active	Disabled or enabled	Functional active run mode (ETM enabled mode when emulation/debug required. Production devices should have ETM disabled).
2	Active	Active	OFF	Active	Disabled or enabled	Functional active run mode. Neon disabled through software; Neon is internally clock-gated.
3	Active	RET	Active	Active	Disabled or enabled	Do not use; see .

Table 3-14. MPU Subsystem Operation Power Modes (continued)

Mode	MPU and ARM Core Logic	ARM L2 RAM	Neon	OMAP Core INTC	APB/ATB ETM/Debug	Comments
4	Active	RET	OFF	Active	Disabled or enabled	Do not use; see (1).
5	Active	OFF	Active	Active	Disabled or enabled	Active mode; L2 is off. Controlled through software to the PRCM module. L2 context save and restore required or L2 flush.
6	Active	OFF	OFF	Active	Disabled or enabled	Active mode; L2 is off. Controlled through software to the PRCM module. L2 context save and restore required or L2 flush.
7	OFF	RET	OFF	OFF	Disabled or enabled	Lowest power sleep mode (dormant mode); L2 is in retention. Controlled through software to the PRCM module. ARM core and L1 context save and restore required or L1 flush.
8	Standby	Active	Standby	Active	Disabled or enabled	Standby mode. StandbyWFI (wait for interrupt) controlled to put into standby and wakeup through interrupt.
9	Standby	Active	OFF	Active	Disabled or enabled	Standby mode. StandbyWFI controlled to put into standby and wakeup through interrupt when Neon is off.
10	Standby	RET	Standby	Active	Disabled or enabled	Standby mode (retention mode). StandbyWFI controlled to put into standby and wakeup through interrupt when L2 is in retention.
11	Standby	RET	OFF	Active	Disabled or enabled	Standby mode (retention mode). StandbyWFI controlled to put into standby and wakeup through interrupt when L2 is in retention and Neon is off.

(1) The L2 can be put in retention mode regardless of other voltage domain states. The combination of Cortex Logic active and L2 in retention mode (modes 3 and 4) is legal, but results in incorrect execution of instructions with referencing data from L2. This combination must not be used.

Table 3-14. MPU Subsystem Operation Power Modes (continued)

Mode	MPU and ARM Core Logic	ARM L2 RAM	Neon	OMAP Core INTC	APB/ATB ETM/Debug	Comments
12	Standby	OFF	Standby	Active	Disabled or enabled	Standby mode. StandbyWFI controlled to put into standby and wakeup through interrupt when L2 is off.
13	Standby	OFF	OFF	Active	Disabled or enabled	Standby mode. StandbyWFI controlled to put into standby and wakeup through interrupt when L2 and Neon are off.
14	OFF	OFF	OFF	OFF	Disabled or enabled	Power-down mode

In any mode where the MPU or Neon power domains are active, the MPU DPLL clocks must be active (modes 1, 3, and 5 require active clocks from the DPLL, while modes 7 and 8 do not).

Thus, the MPU DPLL must be in one of the following states:

- Locked
- Low-power bypass: inclk = on, clkout = on, power = on

When the MPU DPLL is not providing clocks, the MPU subsystem must be in a power mode where the MPU power domain, Neon power domain, debug power domain, and INTC power domain are in standby, RETENTION, or OFF state. The states of the MPU DPLL can be:

- Locked
- STOP low power
- OFF

CAUTION

The L2 can be put into retention mode regardless of other voltage domain states. The combination of ARM Logic active and L2 in retention mode (modes 3 and 4) is legal, but results in incorrect execution of instructions with referencing data from L2. This combination must not be used.

3.3.2.4 Transitions

Table 3-15 describes allowable transitions from power modes described in Table 3-14. For example, a transition from mode 13 to mode 4 is allowed, but the reverse is not true, because the L2 RET to OFF is illegal.

Any mode change that requires state saving or flush must be serialized. For example, L2 RET does not require state saving, so it can happen at the same time as power-down Neon. L2 off and Neon off cannot happen at the same time, because L2 flush and Neon state saving must be serialized. Standby mode can enter from active mode only by executing the WFI instruction.

Table 3-15. Power Mode Allowable Transitions

	To Power Mode													
From	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**
1	Y	Y	Y	Y	Y			Y						Y
2	Y	Y	Y	Y		Y			Y		Y			Y
3	Y	Y	Y	Y			Y			Y				
4	Y	Y	Y	Y			Y				Y			
5	Y		Y	Y	Y	Y	Y			Y		Y		Y
6		Y	Y	Y	Y	Y	Y				Y		Y	Y
7	Y	Y	Y	Y			Y							
8	Y		Y					Y		Y				
9	Y	Y	Y	Y					Y		Y			
10	Y		Y					Y		Y				
11	Y	Y	Y	Y					Y		Y			
12	Y		Y		Y							Y		
13	Y	Y	Y	**Y**	Y	Y			Y		Y		Y	
14	Y	Y	Y	Y	Y	Y	Y							Y

For more information about clocks, reset, power management, and wake-up events for the MPU subsystem, see Chapter 4, *Power, Reset, and Clock Management*.

3.4 MPU Subsystem Basic Programming Model

For detailed descriptions of registers used for MPU configuration, see Chapter 4, *Power, Reset, and Clock Management*, and Chapter 10, *Interrupt Controller*.

3.4.1 Clock Control

For clock configuration settings, see Chapter 4, *Power, Reset, and Clock Management*.

3.4.2 MPU Power Mode Transitions

The following subsections describe transitions of different power modes for the MPU power domain:

- Basic power on reset
- MPU to standby mode
- MPU out of standby mode
- MPU power on from a powered-off state

3.4.2.1 Basic Power-On Reset

The following power-on reset (POR) sequence applies to initial power-up and wakeup from device off mode:

1. Reset DPLL, supply reference clock, program the MPU DPLL in applicable DPLL mode to generate clocks for MPU subsystem modules. This is controlled solely by the PRCM module.
2. Reset the INTC (CORE_RST) and the MPU subsystem modules (MPU_RST). The clocks must be active during MPU reset and CORE reset.

3.4.2.2 MPU to Standby Mode

The following MPU to standby mode sequence applies to initial power-up and wakeup from device off mode:

1. The ARM core initiates entering standby through software only (CP15 - WFI).
2. MPU modules requested internally by MPU subsystem to enter idle, after ARM core standby detected.
3. MPU is in standby output asserted for the PRCM module (all outputs ensured to be at reset values).
4. The PRCM module requests the INTC to enter idle mode. Acknowledge from INTC goes to the PRCM module.
5. The PRCM module starts to shut down clocks through DPLL programming.

Note: The INTC SWAKEUP output is a pure hardware signal to the PRCM module for the status of its IDLE request and IDLE acknowledge handshake.

Note: In debug mode, ICECrusher can prevent the MPU subsystem from entering IDLE mode.

3.4.2.3 MPU Out of Standby Mode

The following MPU out of standby mode sequence of operation applies to initial power-up and wakeup from device off mode:

1. PRCM must start clocks through DPLL programming.
2. Detect active clocking through status output of DPLL.
3. Initiate an interrupt through the INTC to wake up the ARM core from STANDBYWFI mode.

3.4.2.4 MPU Power-On from a Powered-Off State

1. To minimize the peaking of current during power up, DPLL power on, MPU power on, Neon power on, and Core power on (INTC) should follow the ordered sequence per power switch daisy chain.

Note: Before the MPU can be reset, the CORE power domain must be on, and reset, with the DPLL clocks on.

2. Follow the reset sequence as described in Section 3.4.2.1, *Basic Power-On Reset*.

3.4.3 Neon Power Mode Transition

Because of the hardware sleep dependency between Neon and the MPU domain, when the Neon power domain transition is configured to automatic hardware-supervised mode (the CM_CLKSTCTRL_Neon[1:0] CLKTRCTRL_Neon bit field is set to 0x3), it cannot transition to idle mode unless the MPU goes to standby mode. The MPU domain must also be configured in automatic hardware-supervised mode (the CM_CLKSTCTRL_MPU[1:0] CLKTRCTRL_MPU bit field must be set to 0x3) for the Neon power domain transition to occur.

3.4.4 ARM Programming Model

For the complete programming model, see the public *ARM Cortex-A8 Technical Reference Manual*.

Power Reset and Clock Management

This chapter describes power, reset, and clock management in the OMAP34xx multimedia device.

Note: This chapter gives information about all modules and features in the high-tier device. See Section 1.5, *OMAP34xx Family*, to check availability of modules and features. For power-management saving consideration, ensure that power domains of unavailable features and modules are switched off and clocks are cut off.

Topic **Page**

4.1 Introduction to Power Management 234
4.2 PRCM Overview 244
4.3 PRCM Environment 247
4.4 PRCM Integration 251
4.5 PRCM Reset Manager Functional Description 256
4.6 PRCM Power Manager Functional Description 289
4.7 PRCM Clock Manager Functional Description 298
4.8 PRCM Idle and Wake-Up Management 356
4.9 PRCM Interrupts 372
4.10 PRCM Voltage Management Functional Description 373
4.11 PRCM Off-Mode Management 383
4.12 PRCM Basic Programming Model 389
4.13 PRCM Use Cases and Tips 427
4.14 PRCM Register Manual 430

4.1 Introduction to Power Management

This introduction contains the following information:

- Requirement and goal of power management in mobile devices
- State-of-the-art power-management techniques for maximizing battery life for mobile devices
- Essential architectural building blocks for power management
- Overview of the device power-management architecture

4.1.1 Goal of Power Management

Power management (efficient use of the available limited battery resources of a mobile device) is one of the most important design aspects of any mobile system. It imposes strong control over limited available power resources to ensure that they function for the longest possible time.

The device architecture ensures maximum performance for user satisfaction (audio/video support) while offering versatile power-management techniques for maximum design flexibility, depending on application requirements.

4.1.2 Power-Management Techniques

The following sections describe the state-of-the-art power-management techniques supported by the device.

Note: The values in Figure 4-1 through Figure 4-3, which show power-management techniques, are hypothetical only. They do not represent valid test results on the device.

4.1.2.1 Dynamic Voltage and Frequency Scaling

Dynamic voltage and frequency scaling (DVFS) reduces the device voltage and frequency by minimizing the idle time of the system. The DVFS technique uses dynamic selection of the optimal operating frequency and voltage to let a task be performed in the required amount of time. This reduces the active power consumption (power consumed while executing a task) of the device while still meeting task requirements.

Figure 4-1 compares energy consumption with and without DVFS.

Figure 4-1. Comparison of Energy Consumption With/Without DVFS

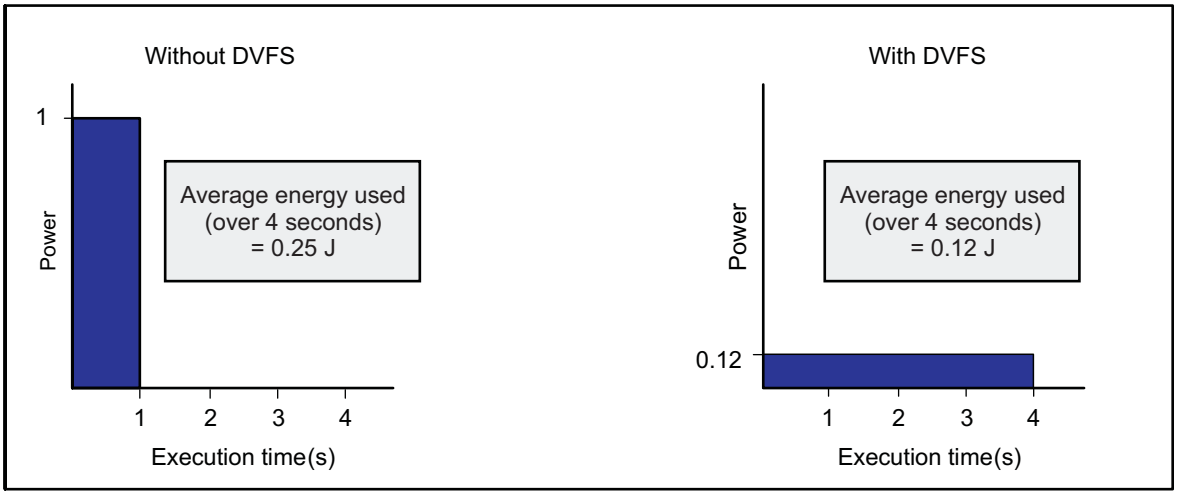


Figure 4-1 shows the DVFS technique by comparing a process executed at maximum frequency and operating voltage without applying DVFS to the same process executed at optimal frequency and voltage using DVFS, based on the task requirements. If a task that must terminate in 4 seconds is performed at maximum operating frequency (left side of the figure), it terminates in 1 second, and the remaining 3 seconds are spent in idle mode. With DVFS (right side of the figure), the operating frequency is reduced to an optimal level; the task takes the full 4 seconds to complete, but power consumption is reduced. In addition, voltage can be further reduced to save power so dynamic and leakage power consumption are reduced.

DVFS requires control over the clock frequency and the operating voltage of the device elements. By intelligently switching the individual elements of the device to their optimal operating points, it is possible to minimize the power consumption of the device for a given task.

For practical reasons related to device making (flow, tools), DVFS can be used for only a few discrete steps, not over a continuum of voltage and frequency values. Each step, or operating performance point (OPP), is composed of a voltage (V) and frequency (F) pair. For an OPP, the frequency corresponds to the maximum frequency allowed at a voltage, or reciprocally, the voltage corresponds to the minimum voltage allowed for a frequency.

When applying DVFS, a processor or system always runs at the lowest OPP that meets the performance requirement at a given time. The user determines the optimal OPP for a given task and then switches to that OPP to save power.

4.1.2.2 Dynamic Power Switching

Like DVFS, dynamic power switching (DPS) is a power-management technique that reduces active power consumption of a device. However, whereas DVFS reduces dynamic and leakage power consumption, DPS reduces only leakage power consumption, at the expense of a slight overhead in dynamic power consumption.

With DPS, the system switches dynamically between high- and low-consumption system power modes during system active time. When DPS is applied, a processor or system runs at the highest OPP (maximum frequency and voltage) to complete its tasks quickly, followed by an automatic switch to a low-power mode for minimum power consumption. DPS is useful when a real-time application is waiting for an event. The system can switch to a low-power system mode if the wake-up latency conditions allow it.

This technique consists of maximizing the idle period of the system to reduce its power consumption.

Figure 4-2 compares energy consumption with and without DPS.

Figure 4-2. Comparison of Energy Consumed With/Without DPS

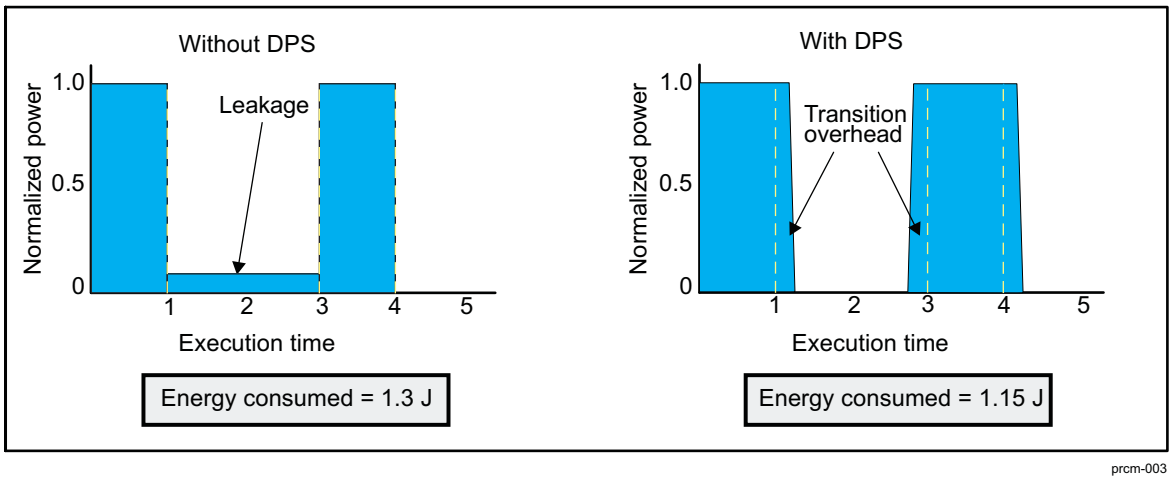


Figure 4-2 compares the power consumption behavior for the same device operation without DPS (left side of the figure) and with DPS (right side of the figure). When operating without DPS, the device has a

constant leakage current in idle mode. By using DPS, the system reduces the leakage current to zero. However, the transitions between system power modes can require storing the information before entering a low-power inactive state and restoring the information after a wake-up event (see Figure 4-2). This results in additional dynamic power consumption, referred to as the transition overhead, (see Figure 4-2). Transition overhead must be considered for a DPS operation.

For efficient deployment of DPS techniques, it is necessary to dynamically predict the performance requirement of the applications running on the processor. The DPS controller must account for the overhead of wake-up latencies related to domain switching and ensure that they do not significantly affect the performance of the device. Even with transition overhead, however, the user can identify an optimal idle-time limit after which the DPS is useful for dynamic power-saving.

4.1.2.3 Standby Leakage Management

Standby leakage management (SLM) is a power-management technique that reduces standby power consumption by reducing power leakage.

With SLM, the device switches to low-power system modes automatically or in response to user requests during system standby (in situations when no application is started and system activity is negligible or limited).

When applying SLM, the system remains in the lowest static power mode compatible with the system response time requirement.

This technique trades static power consumption for wake-up latency.

4.1.2.4 DPS Versus SLM

DPS and SLM are similar concepts: they consist of switching the system between high- and low-power consumption modes. However, their operating timescales differ, principally in the latency allowed for mode transitions.

DPS is generally used in an applicative context (tasks are started). Therefore, mode transitions are related to system performance requirements or the processor load. DPS transition latencies must be small (typically between 10 µs and 100 µs) compared to the time constraints or deadlines of the application so that they do not degrade application performance. DPS requires performance prediction to ensure that transition latencies do not deteriorate device performance to the point that real-time application deadlines are missed or the user experience degrades too much for an interactive application.

SLM is not used in an applicative context (no task started). Mode transitions are more related to system responsiveness, and the transition latencies must be small compared to user sensitivity so that they do not degrade the user experience. For SLM, transition latencies are typically 1 ms to 10 ms or more.

DPS and SLM also differ in the type of wake-up event used to exit low-power idle mode. For DPS, wake-up events are application-related (timer, DMA request, peripheral interrupt, etc.); for SLM, wake-up events are user-related (touch screen, key pressed, peripheral connections, etc.).

4.1.2.5 Combining Power-Management Techniques

The DVFS, DPS and SLM, power-management techniques have different features and are most effective when used under specific device operating conditions. Hence, the best active power saving is obtained by combining the techniques. For a given operating state, one or more of the power-saving techniques can be applied to ensure optimal operation with maximum power saving.

When medium application performance is required, or when application performance requirements vary, DVFS can be applied. The voltage and frequency can be scaled to match the closest OPP that meets the performance requirement.

When application performance requirements fall between two OPPs, or when required application performance is below the lowest performance OPP, DPS can be applied to switch to low-power mode.

When combining DVFS and DPS, the operating frequency must not be scaled to match the performance requirement without scaling the voltage. Lower operating frequency increases task completion time and reduces idle time. This prevents DPS or reduces its efficiency (DPS becomes more effective as idle time increases). Unless DPS cannot be applied for other reasons, for a given operating point of DVFS, the operating frequency must always be set to the maximum allowed at a given voltage. This ensures optimal process completion time and application of DPS.

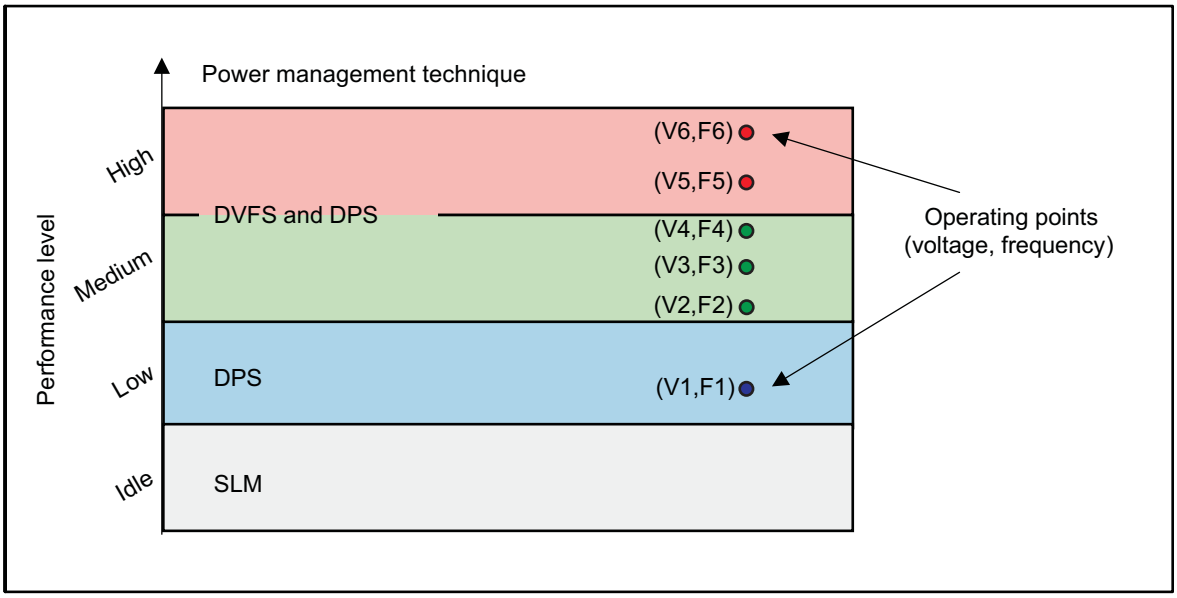
If DPS cannot be applied in a given context, scaling the frequency while keeping the voltage constant does not save energy. It does, however, reduce peak power consumption. This can have a positive effect on temperature dissipation and battery life.

In situations where no applications are running and performance requirement drops to zero, SLM must be used.

Figure 4-3 compares combinations of the power-management techniques.

Note: The OPPs shown in Figure 4-3 are for explanation only. They do not correspond to validated OPPs of the device.

Figure 4-3. Performance Level and Applied Power-Management Techniques



4.1.3 Architectural Blocks for Power Management

The device supports the power-management techniques through three architectural blocks: the power, clock, and voltage domains. A domain is a group of modules or subsections of the device that share a common entity (clock, voltage, or power switching).

4.1.3.1 Clock Domain

A clock domain is a group of modules fed by the same gated clock signal (see Figure 4-4). By gating the clock to each domain, it is possible to cut a clock to a group of inactive modules to lower their active power consumption. Thus, a clock domain allows control of dynamic power consumption by the device.

Figure 4-4. Generic Clock Domain

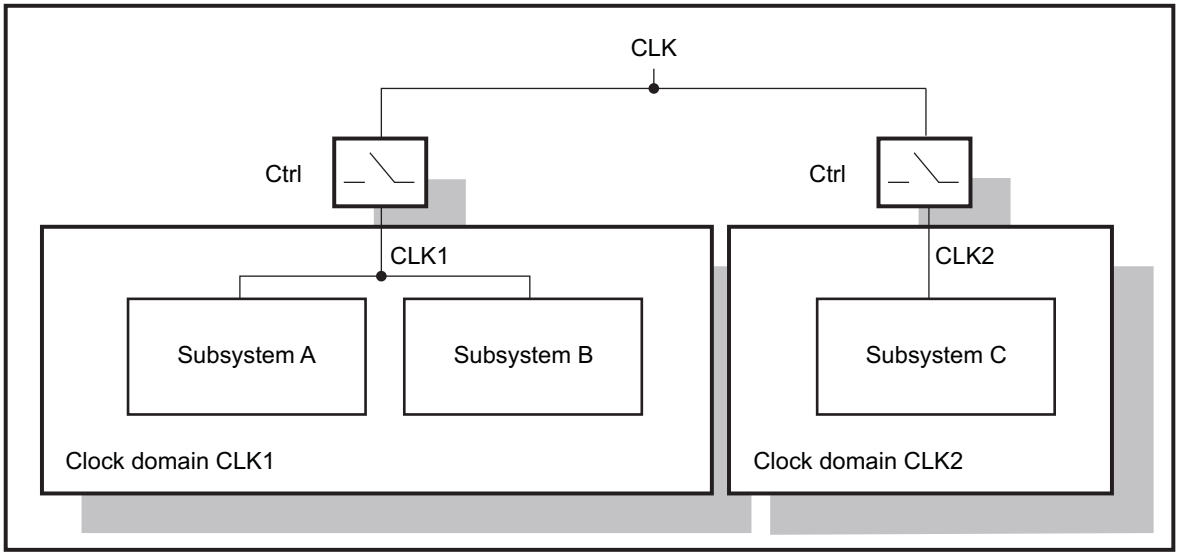


Table 4-1 lists the two possible states of the clock domain.

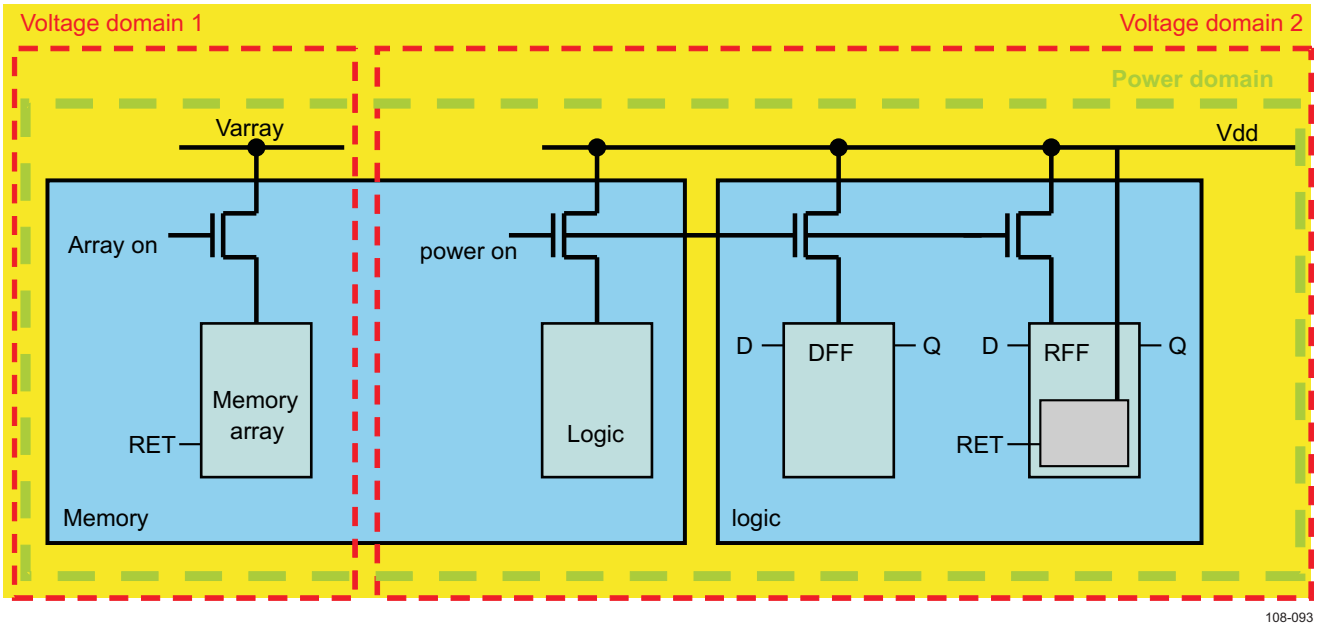
Table 4-1. States of a Clock Domain

State	Description
Active	The domain clock is running.
Idle	The domain clock is stopped or gated.

4.1.3.2 Power Domain

A power domain is a section of the device with independent and dedicated power rails (see Figure 4-5). A power domain can be turned on/off without affecting the other parts of the device.

Figure 4-5. Generic Power Domain



To minimize device power consumption, the modules are placed in power domains. All power domains embed some logic and some memory. The memory contains a memory array powered by a dedicated voltage rail (Varray + arrayon switch) and some logic powered by the same voltage rail as the power domain logic (VDD + power on switch).

Note: The behavior of the Varray, arrayon, and power on switches can be selected by software (PWSTCTRL registers) hardwired; once the selection is made, the switches are handled automatically by the PRCM module.

The power domain logic for the CORE and the PER power domains can be split between retention flip flops (RFF) or nonretention flip flops (DFF). All other power domains embed only DFF.

A power domain can be in several power domain states: On, inactive, open/closed switch retention (OSWR, CSWR), or off.

- On or inactive (inactive is same as on but with all clocks cut): Power on is set to 1; Vdd is provided to the logic (DFF and RFF) and to the memory logic. All the logic is fully working. Depending on software settings, Varray can keep its active value or be lowered and the arrayon switch can be open or closed (depending on the power domain, this can be hardwired). As a consequence, the memory content can be accessible, retained, or lost.
- Closed-switch retention (CSWR): Power on is set to 1, Vdd is provided to the logic (DFF and RFF) and to the memory logic, but Vdd can be lowered to its retention value. The logic is not functional, but is retained. Depending on software settings, Varray can keep its active value (if needed by another memory array in another power domain), or be lowered, and the arrayon switch can be open or closed (depending on the power domain, this can be hardwired). As a consequence, the memory content can be retained or lost.
- Open-switch retention (OSWR) (CORE and PER power domains only): Power on is set to 0, Vdd, which can be lowered to its retention value, is provided only to the RFF logic. Only the RFF logic is retained. The DFF logic is lost and reset on wakeup. Depending on software settings, Varray can keep its active value (if needed by another memory array in another power domain), or be lowered, and the arrayon switch can be open or closed (depending on the power domain, this can be hardwired). As a consequence, the memory content can be retained or lost.
- Off: Power on is set to 0, Vdd is usually cut, all the logic (DFF and RFF) is lost, except for the context that has been saved in the scratchpad memory of the WKUP power domain (always on). Varray can keep its active value (if needed by another memory array sharing the same Varray voltage rail in another power domain), be lowered (if all other memory arrays sharing the same Varray voltage rail are set to be in retention), or be cut (0 V) when the entire device is in off mode. The arrayon switch is open and the memory content is lost.

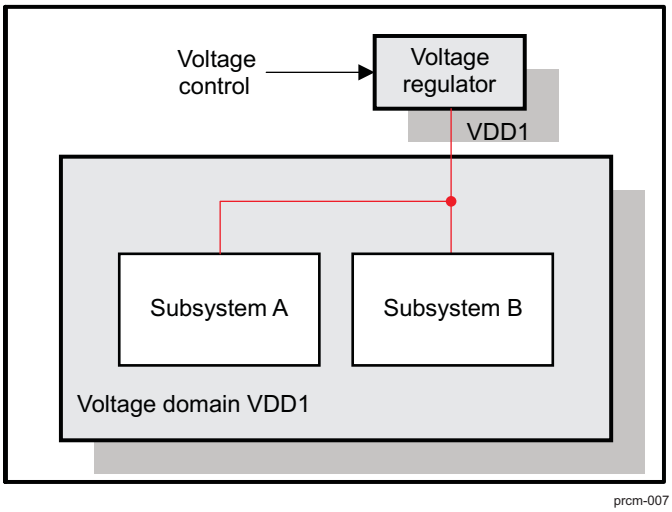
The retention state is useful for quickly switching to low-power idle mode without losing the context, and then quickly switching back to active state when necessary. In retention state, power consumption is less than in normal operating mode.

4.1.3.3 Voltage Domain

A voltage domain is a group of modules supplied by the same voltage regulator (embedded or external). The power consumption of this group can be controlled by regulating its voltage independently.

Figure 4-6 shows the voltage domain.

Figure 4-6. Generic Voltage Domain



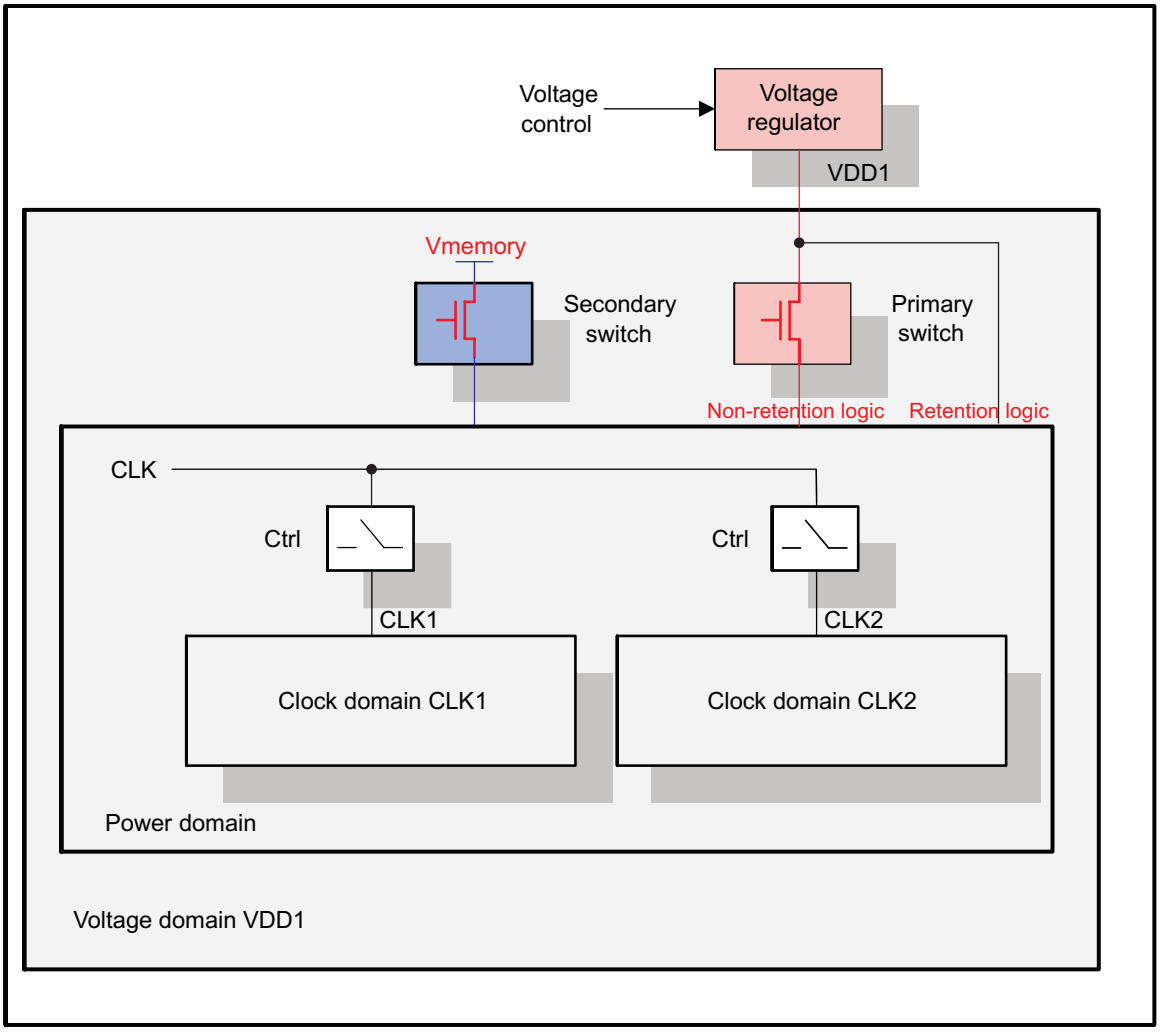
By partitioning the device into independent voltage domains, it is possible to assign different operating voltages to the different modules. The independent voltage control allows voltage scaling of device subsections to ensure that each module operates at the optimal operating voltage level based on the application performance requirements. When all modules in a voltage domain are inactive, the domain voltage can be lowered to reduce power consumption and then be switched back to normal operating voltage when a wake-up event is received.

4.1.4 Device Power-Management Architecture

The device architecture integrates the power-management architectural blocks for power-management support. It is composed of scalable/switchable voltage domains (their voltage can be controlled) and switchable power domains. Figure 4-7 shows the general hierarchical architecture scheme of the voltage, power, and clock domains.

A clock domain is a subset of a power domain. This avoids unnecessary dependencies between power domains caused by clocks covering multiple power domains. Each clock in a power domain, with an independent gating control, is a separate clock domain.

Figure 4-7. Voltage, Power, and Clock Domain Hierarchical Architecture



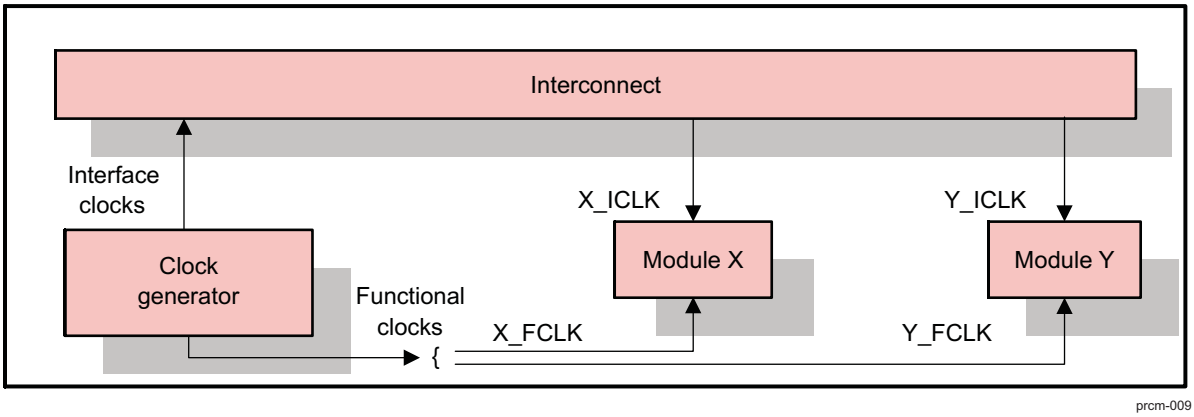
Each power domain is supplied by one or more voltage domains that can be scaled down or switched off to save power. Internal memory and logic can be put in standby (retention) mode independently. In this state, memory is not operational, but the content is retained with minimized leakage. This feature lets power consumption be reduced when the device is in sleep mode while maintaining memory contents for fast context restore. The logic and memory standby feature is software-controllable.

The OMAP34xx supports a clock distribution and control architecture, which is described in the following sections.

4.1.4.1 Module Interface and Functional Clocks

The clocks delivered to the modules in the device are divided into two categories: interface clocks and functional clocks (see Figure 4-8).

Figure 4-8. Functional and Interface Clocks



Interface clocks have the following characteristics:

- They ensure proper communication between any module/subsystem and the interconnect.
- In most cases, they supply the module interface and registers.
- A typical module has one interface clock, but it can have several.
- They are synchronous across the entire device, because the interconnect fabric is fully synchronous.
- Interface clock management is done at the device level.
- From the standpoint of the PRCM module, an interface clock is distributed through the device interconnect and is identified with an _ICLK suffix.

Functional clocks have the following characteristics:

- They supply the functional part of a module or a subsystem.
- Each module can have several functional clocks, or none. A module may or may not require its functional clocks to be active (nonidle).
- Several modules can share the same functional clock signals, but the branches of the clock tree are not balanced among the modules.
- From the PRCM module standpoint, a functional clock is directly distributed to the related modules through a dedicated clock tree. It is identified with an _FCLK suffix.

Note: At the module level, the interface clocks are always fed by the interface clock outputs of the PRCM module. The functional clocks are fed by a PRCM functional clock output or a PRCM interface clock output. In the latter case, the functional and interface clocks of the module inherit capabilities (autoidle features) from the PRCM interface clock (see Section 4.7, *Clock Manager Functional Description*).

4.1.4.2 Autoidle Clock Control

The device supports an autoidle clock control scheme for the module interface clocks. With this control scheme, the PRCM module can automatically activate and deactivate the interface clock of any device module, depending on its operating mode. This scheme executes under hardware control and is transparent to the software. This scheme identifies two module types in the device: the target and the initiator modules, or subsystems.

Note: The functional clocks do not have the autoidle clock scheme, and the software must gate the functional clock of each module when it is not needed.

Initiator

An initiator can generate bus transactions (read, write, etc.) toward targets. It is considered active when it generates transactions. If it enters standby mode, it stops generating transactions. Because most initiators also support a target port for configuration, they are both targets and initiators. Some examples of initiators are processors, direct memory access (DMA), and memory management unit (MMU).

Target

A target is a passive module that can process bus transactions (read/write to memory-mapped registers). It is considered active when its interface clocks and some or all of its functional clocks are available so it can accept incoming transactions. A target can be put in idle mode by the PRCM module, and in this mode, its interface clock can be gated at any time. An idle module can still receive its functional clocks and generate interrupts and DMA requests. It can also generate asynchronous wake-up requests, if it is wakeup-capable.

Active, Idle, and Standby Modes

The PRCM module handles automatic clock control differently for initiator and target modules.

For the initiator module, the following hardware handshake scheme is employed:

1. The initiator, when switching from active to idle mode, signals its status to the PRCM module.
2. The PRCM module cuts off the interface clock to the initiator module.
3. When the initiator module must reactivate, it signals the PRCM module, which reactivates its functional and interface clocks.

For the target module, the following hardware handshake scheme is employed:

1. When the PRCM module confirms that the target module satisfies the idle conditions, it signals the target module.
2. The target module acknowledges the idle request of the PRCM module, depending on its idle mode internal settings (for details about idle mode settings, see the chapter in the technical reference manual for the corresponding device module):
   - If the module is set to smart-idle mode, it terminates its current operations, then acknowledges the idle request to the PRCM module.
   - If the module is set to force-idle mode, it acknowledges immediately, regardless of its state. Because pending transfers, interrupts, and DMA requests can be lost, special software care must be taken.
   - If the module is set to no-idle mode, it does not acknowledge the idle request. This forces the PRCM module to maintain the clock active.
3. The PRCM module cuts off the module clocks.
4. The target module can be awakened by the PRCM module when its wake-up conditions are satisfied (wake-up event). It activates the module clocks, and then signals the wakeup.
5. The target module acknowledges the wake-up request.
6. Some target modules can support wake-up capability. They can explicitly request the PRCM module to activate their clock.

This automatic clock control ensures reduced dynamic power consumption of the device without any associated software overhead.

4.2 PRCM Overview

This section describes all modules and features in the high-tier device. To check availability of modules and features, see Section 1.5, *OMAP34xx Family*. To save power, ensure that power domains of unavailable features and modules are switched off and clocks are cut off.

4.2.1 Introduction

The power-management framework of the device significantly reduces dynamic power consumption and static leakage current to extend the life of the battery in the end product. This framework incorporates support for state-of-the-art power-management techniques. It ensures optimal device operation with significantly reduced power consumption. The PRCM module, which is the enhanced power-management subsystem of the device, is the central control module for the clock, reset, and power-management signals in the device.

The device supports the power-management techniques with the following features:

- Partitioning of the device into 16 voltage domains and 18 power domains
- Domain isolation that allows any combination of domain on/off states
- Clock tree with selective clock-gating conditions
- Hardware-controlled reset sequencing management
- Power, reset, and clock control hardware mechanism to manage sleep and wake-up dependencies of power domains
- Support for hardware-controlled autogating of module clocks
- Memory retention capability for preserving memory contents in low-power sleep mode
- Scalable voltage and frequency support for the processors, CORE, and peripherals for DVFS
- Support for low-power device off mode input/output (I/O) pad configuration for minimum power consumption when in off power state

The PRCM module interfaces with all the components of the device for power, clock, and reset management through power-control signals. It integrates enhanced features to let the device adapt energy consumption dynamically according to changing application and performance requirements. The innovative hardware architecture allows a substantial reduction in leakage current.

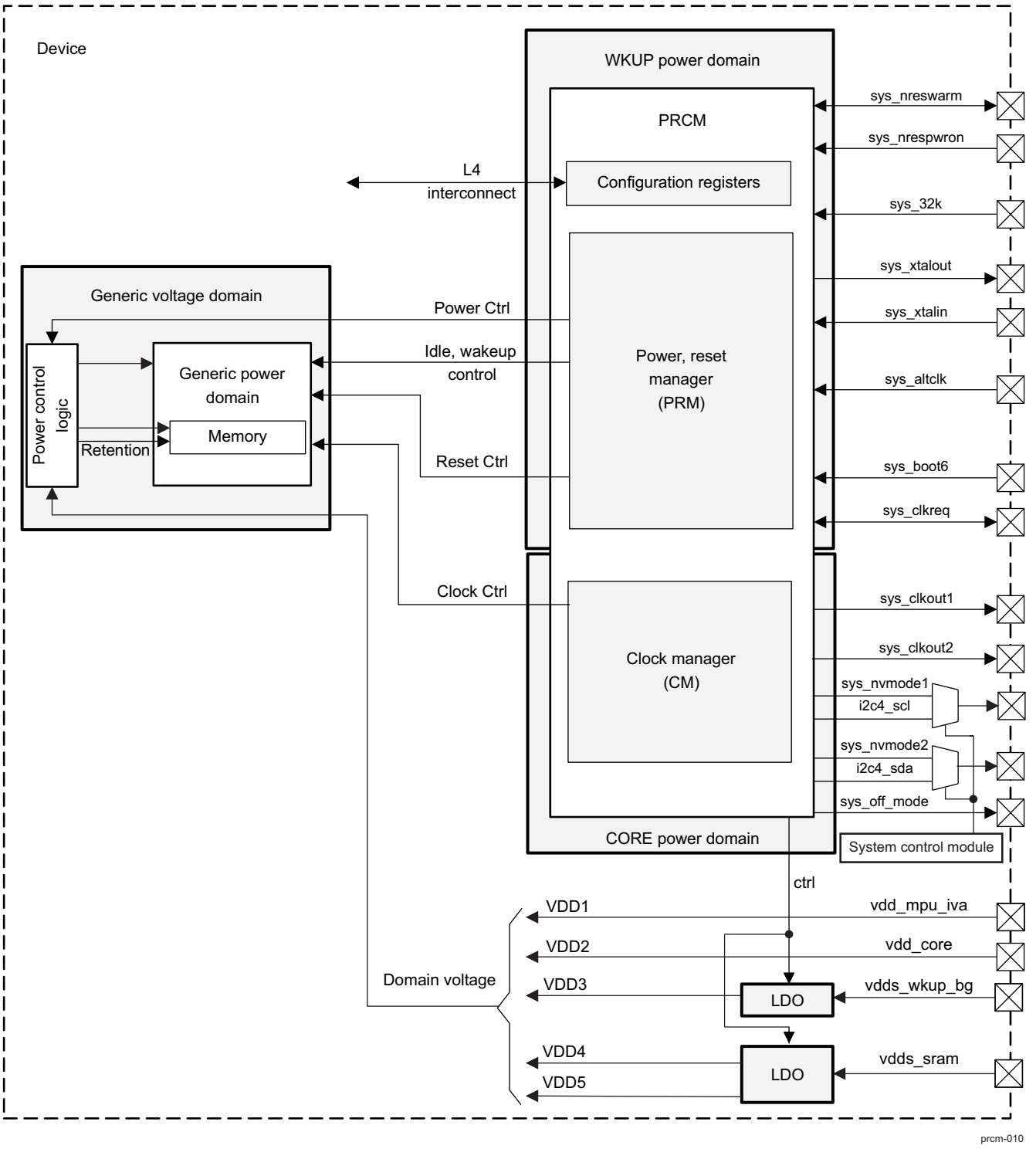
The PRCM module is composed of two main entities:

- Power reset manager (PRM): Handles power, reset, wake-up management, and system clock source control (oscillator)
- Clock manager (CM): Handles clock generation, distribution, and management

The PRCM module is fully configurable through its L4 interface port.

Figure 4-9 is an overview of the PRCM module and its internal connections with a generic power domain.

Figure 4-9. PRCM Overview



4.2.2 PRCM Features

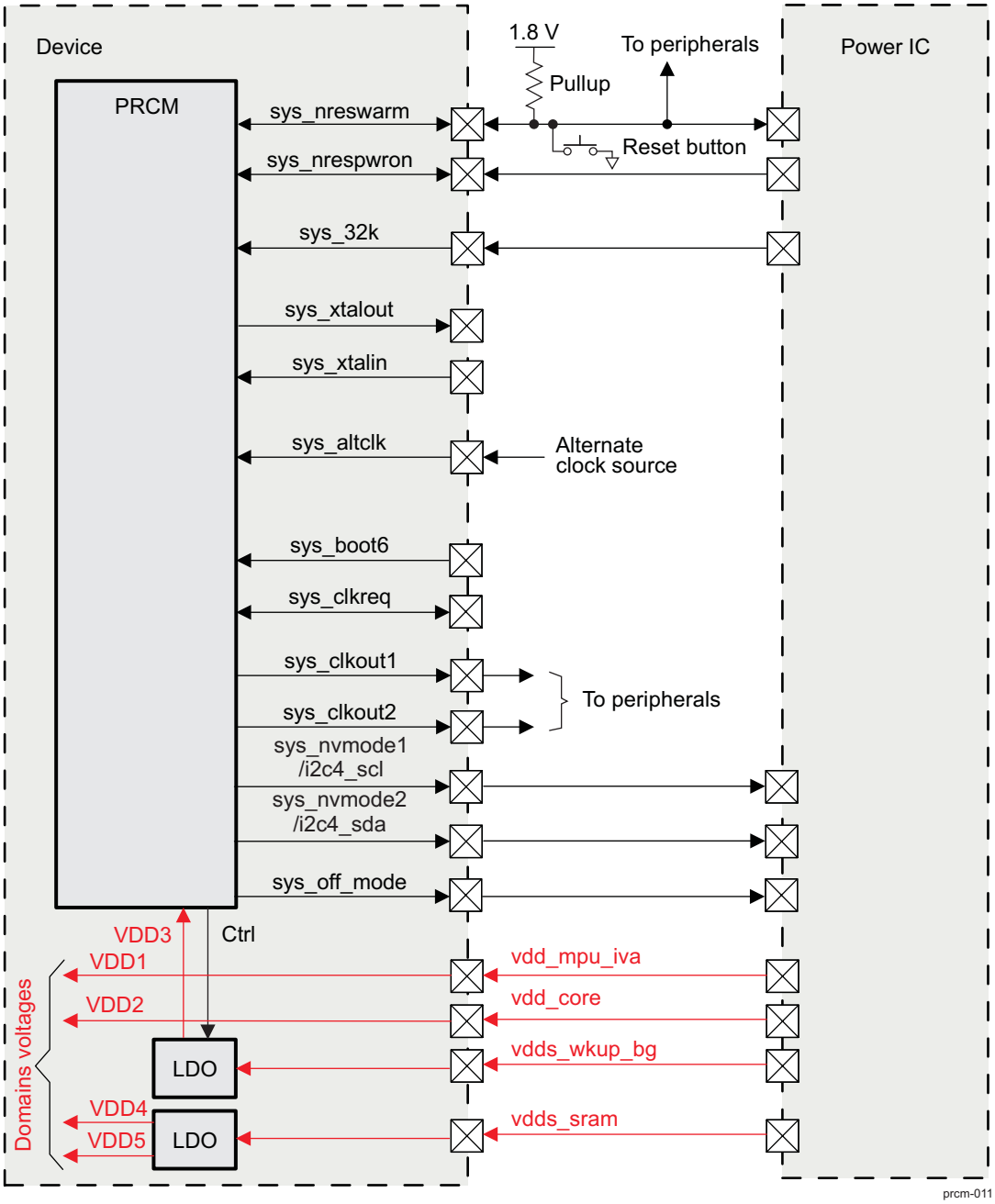
The PRCM module includes the following features:

- Managing 18 independent power domains
- Controlling two scalable and three memory-selectable voltage modes
- Handling idle/wake-up procedures
- Allowing software and partial hardware control
- Monitoring and handling wake-up events
- Controlling system clock/reset input sources
- Managing and distributing clocks and resets with high granularity
- Handling power-up sequences
- Debug and emulation features
- Controlling external supply voltage through dedicated high-speed (HS) I^2C interface
- CM implementation of RFFs to support DPS

4.3 PRCM Environment

The PRCM module receives the external reset, clock, and power signals. Figure 4-10 shows the interface of the PRCM module with external reset, clock, and power sources.

Figure 4-10. PRCM Functional External Interface (Detailed View)



Note: In the remainder of this chapter, "power IC" refers to a peripheral power source IC that is interfaced with the device. It receives power control commands (voltage scaling and power switching) from the device and provides the necessary voltages and reset signals.

The following sections describe the interfaces for external clock, reset, and power sources.

4.3.1 External Clock Signals

The device has three external clock inputs: low frequency (sys_32k), high frequency (sys_xtalin), and an optional clock (sys_altclk). The device has two configurable clock outputs: sys_clkout1 and sys_clkout2. Figure 4-11 shows the external clock signals of the PRCM module. Table 4-2 lists the external clock signals, I/Os, and module reset values.

Figure 4-11. External Clock Interface

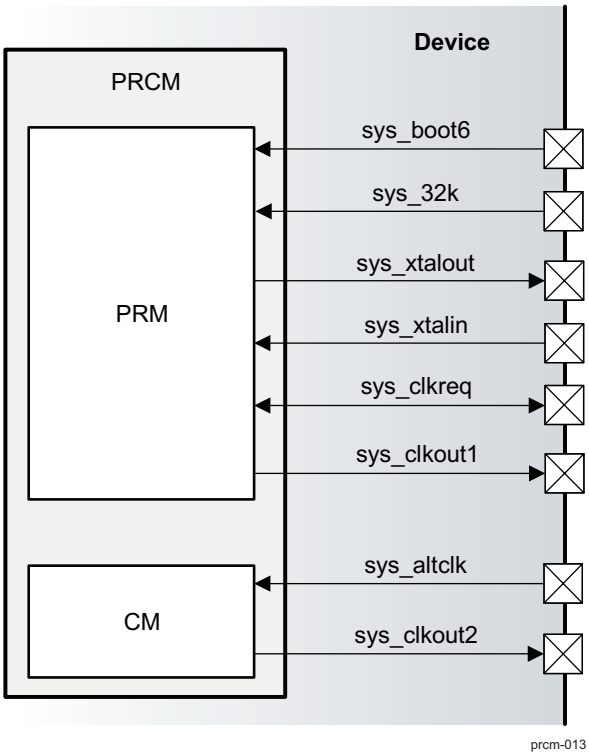


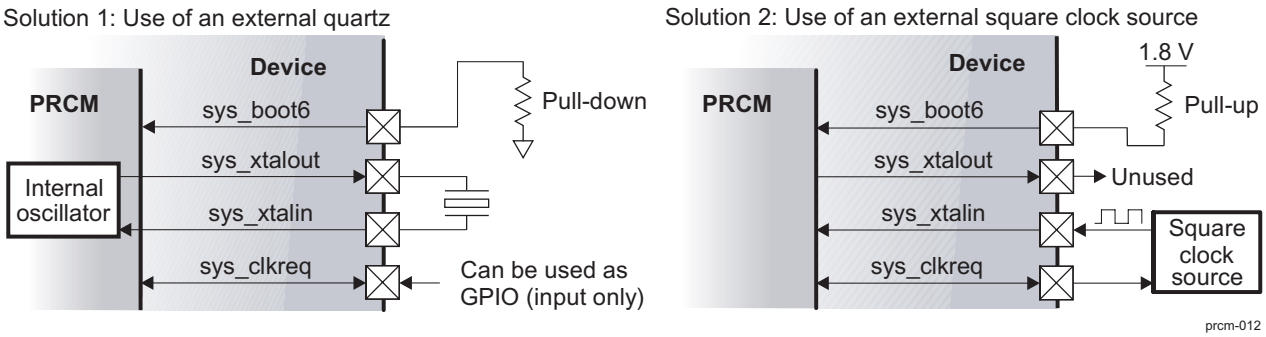
Table 4-2. External Clock Signals

Signal	I/O[1]	Description	Module Reset Value
sys_boot6	I	Boot oscillator mode control	Unknown
sys_32k	I	32-kHz clock input	Unknown
sys_xtalout	O	Oscillator output	0
sys_xtalin	I	Main input clock. Crystal oscillator clock (only at 12, 13, 16.8, or 19.2 MHz) or CMOS digital clock (at 12, 13, 16.8, 19.2, 26, or 38.4 MHz).	Unknown
sys_clkreq	I/O	Clock request to/ from device for system clock	1
sys_clkout1	O	Configurable output clock 1	0
sys_altclk	I	Alternate clock source selectable for universal serial bus (USB) (48 MHz), or NTSC/PAL (54 MHz)	Unknown
sys_clkout2	O	Configurable output clock 2	0

(1) I = Input; O = Output

Figure 4-12 shows the PRCM external clock sources.

Figure 4-12. PRCM External Clock Sources



The system clock source can be either of the following:

- Internal oscillator with crystal connected between sys_xtalin and sys_xtalout
- CMOS digital clock that arrives at the sys_xtalin pin

In the first option, the sys_clkreq signal is used in input mode to control sys_clkout1 and the internal clock oscillator. In the second option, the signal is used in output mode to request the external system clock.

An external pulldown or pullup tied on sys_boot6 determines whether the internal oscillator is used or an external clock source is supplied, respectively.

CAUTION

Only one clock source option can be used at a time.

An alternate clock input (sys_altclk) provides a precise clock source for 54 MHz, 48 MHz, or other frequencies (for example, 59 MHz or 49.04 MHz for VDAC).

For more information about external clock signals, see Section 4.7.5 , *External Clock Controls*.

4.3.2 External Reset Signals

The device supports two reset signals: power-on (sys_nrespwron) and warm reset (sys_nreswarm).

sys_nrespwron is asserted at power up to reset the full logic in the device. sys_nreswarm can be activated at any time by an external device or an external reset push-button action (see Figure 4-10) to cause a global warm reset event.

Because sys_nreswarm is bidirectional, it can also drive a reset of external devices. Any global warm reset source (for example, a push-button) causes sys_nreswarm to be driven out and maintained for a limited length of time at the boundary of the device. In this way, the device and its related peripherals are reset together.

The sys_nrespwron assertion also causes assertion of sys_nreswarm.

Figure 4-13 shows the external reset signals. Table 4-3 lists the external reset signals, I/Os, and module reset values.

Figure 4-13. External Reset Signals

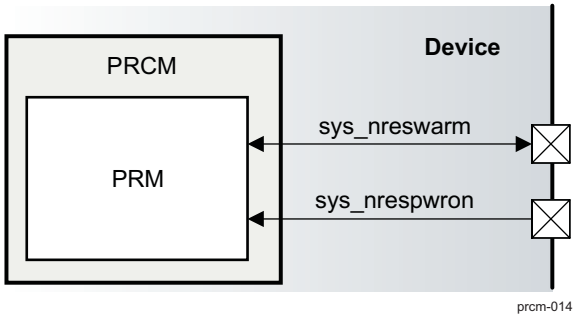


Table 4-3. External Reset Signals

Signal	I/O[1]	Description	Module Reset Value
sys_nreswarm	I/O	Warm-boot reset	1
sys_nrespwron	I	Power-on reset	Unknown

(1) I = Input; O = Output

4.3.3 External Power Signals

The power control signals let the PRCM module control the device voltage levels. The voltage level of the scalable voltage sources in the external power IC can be scaled by sending commands to the power IC through this interface. The PRCM module can also command the power IC to switch the device voltages off when the device is in off mode and activate them when it wakes up.

Figure 4-14 shows the power control interface for the external power IC. Table 4-4 lists the signals, I/Os, and module reset values for the external power IC.

Figure 4-14. Power Control Interface for External Power IC

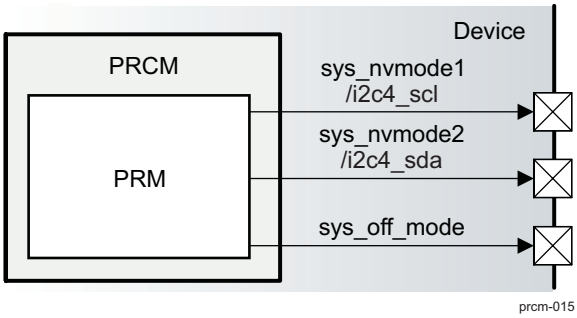


Table 4-4. Power Control Interface

Signal	I/O[1]	Description	Module Reset Value
sys_nvmode1/i2c4_scl	O	Commands the external power IC for control of VDD1 and VDD2 voltages Shared by the VMODE interface and the dedicated I^2C interface	0x1
sys_nvmode2/i2c4_sda	O	Commands the external power IC for control of VDD1 and VDD2 voltages. Shared by the VMODE interface and the dedicated I^2C interface	0x1
sys_off_mode	O	Requests the external power IC to switch the device voltage level according to the device power status	0x0

(1) I = Input; O = Output

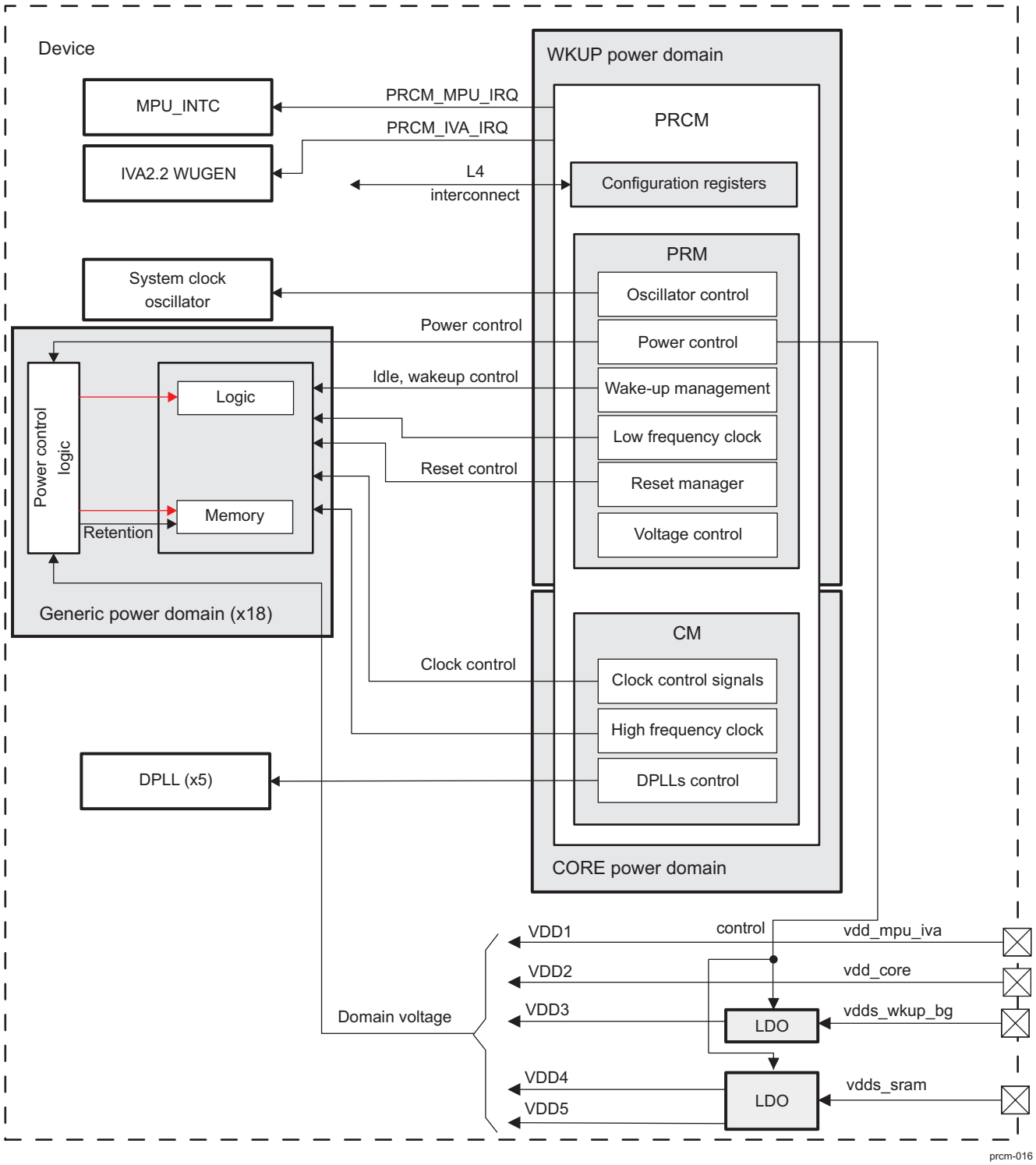
4.4 PRCM Integration

The PRCM internal registers can be accessed for configuration and controlled through the WKUP L4 interconnect. In addition to the L4 interconnect, the PRCM internal module interface contains the following:

- A set of signals for idle/wake-up control for each module
- Clocks and reset signals
- Power control signals (switches and memories) to the power domains
- Interrupts to the MPU subsystem and IVA2.2 subsystem INTCs
- Digital phase-locked loop (DPLL) control commands for recalibration and bypass
- System clock oscillator control for device-level sleep/wake-up transitions

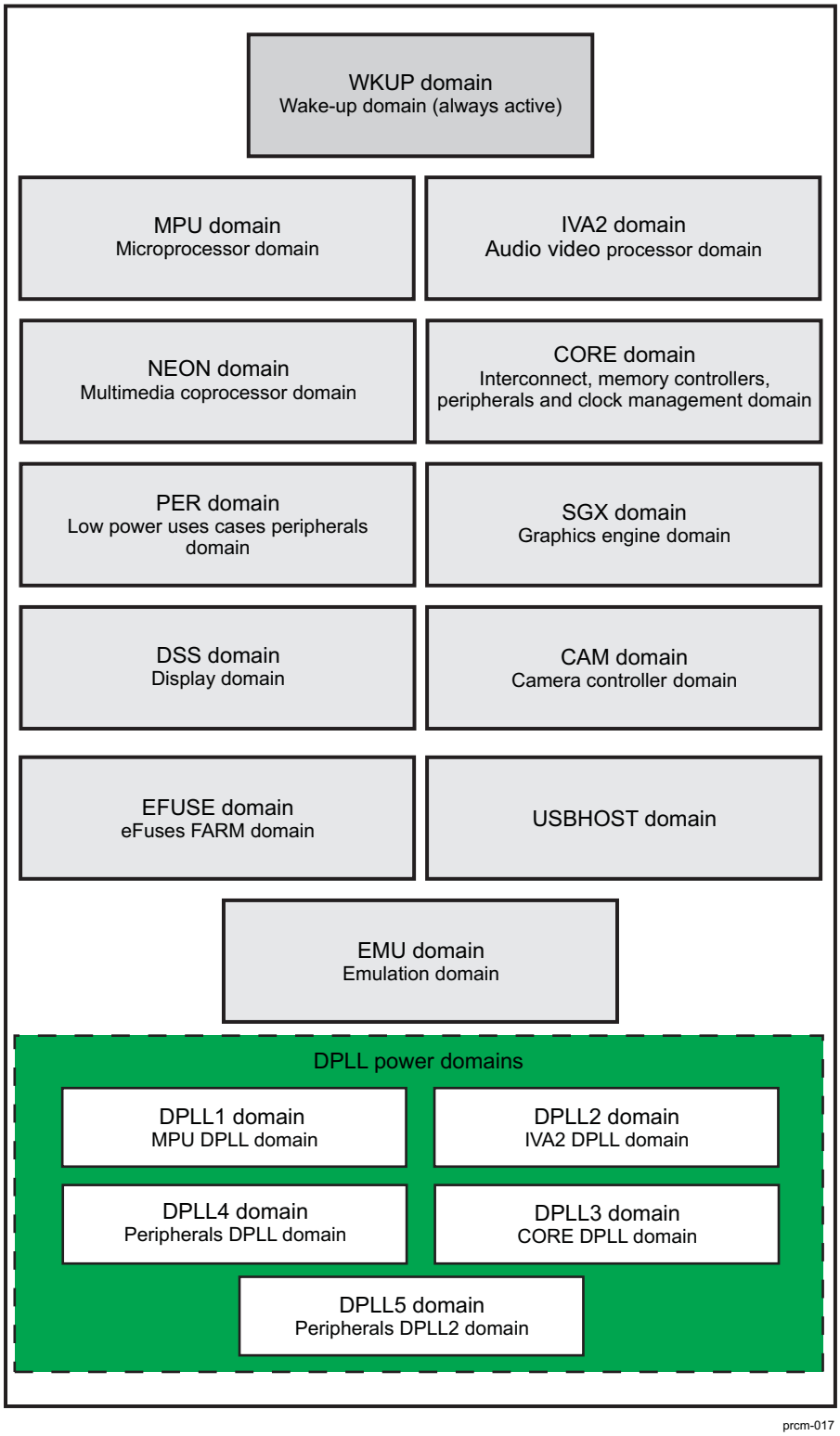
Figure 4-15 shows details of the control interface to a generic power domain.

Figure 4-15. PRCM Integration



To significantly reduce leakage in sleep modes (SLM strategy) and to optimize active power consumption (DPS strategy), the device is segmented into power domains (see Figure 4-16).

Figure 4-16. Device Power Domains



Each power domain is fed through an independent switch controlled by the PRM module. In this way, depending on the application scenario, unused parts (domains) can be switched off or put in retention state while others remain active.

4.4.1 Power-Management Scheme, Reset, and Interrupt Requests

4.4.1.1 Power Domain

The PRCM module is in two power domains (see Table 4-5).

Table 4-5. PRCM Power Domains

PRCM Subsystem Modules	Power Domain
PRM	WKUP
CM	CORE

The PRM module is in the WKUP power domain, which is continuously active. It is composed of the logic that must be permanently supplied to manage domain power-state transitions and detect wake-up events.

The CM module is in the CORE power domain, which can be activated and deactivated according to the requirements of the executing applications.

4.4.1.2 Resets

The PRM and CM modules are reset by independent reset signals (see Table 4-6).

Table 4-6. PRCM Reset Signals

PRCM Subsystem	Reset Signal
PRM	PRM_RSTPWRON
CM	CM_RSTPWRON_RET

The PRM module is reset by the cold reset signal PRM_RSTPWRON. The CM module is reset by assertion of the CM_RSTPWRON_RET signal.

The CM logic is reset on:

- Any global cold reset
- A CORE power domain transition from off to on

The PRM logic is reset on any global cold reset.

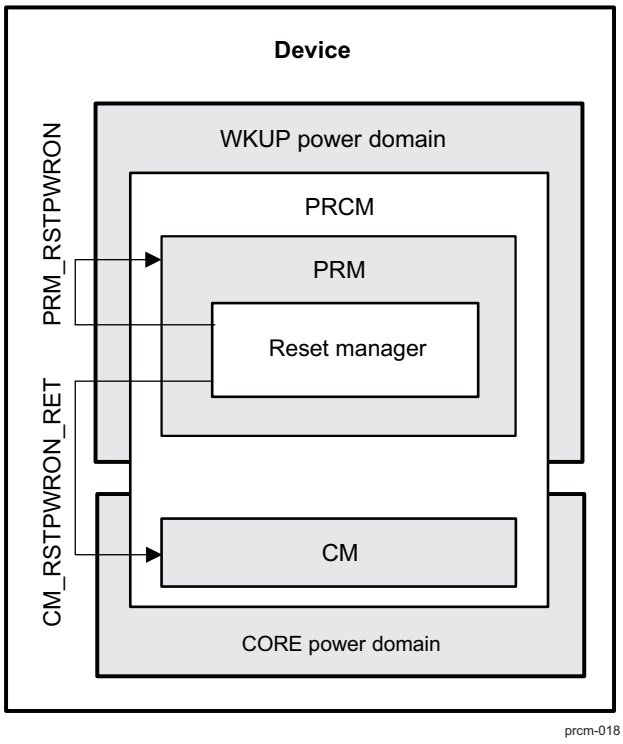
CM and PRM registers that are sensitive to a warm reset are also reset when a global warm reset occurs. However, the CM and PRM logic is not reset.

Note: At global cold reset:

- Only the device finite state-machine (FSM) in the PRM operates on the 32-kHz clock, and it is released from reset on the release of the global reset.
- PRM logic operates on the system clock and is released from reset on release of the reset PRM_RSTPWRON.

Figure 4-17 shows the PRCM reset signals.

Figure 4-17. PRCM Reset Signals



4.4.1.3 Interrupt Requests

The PRCM module can generate two interrupts:

- PRCM_MPU_IRQ: Mapped to the MPU INTC module (M_IRQ_11 interrupt line)
- PRCM_IVA_IRQ: Mapped to the IVA2.2 WUGEN module (IVA2_IRQ[12] interrupt line)

4.5 PRCM Reset Manager Functional Description

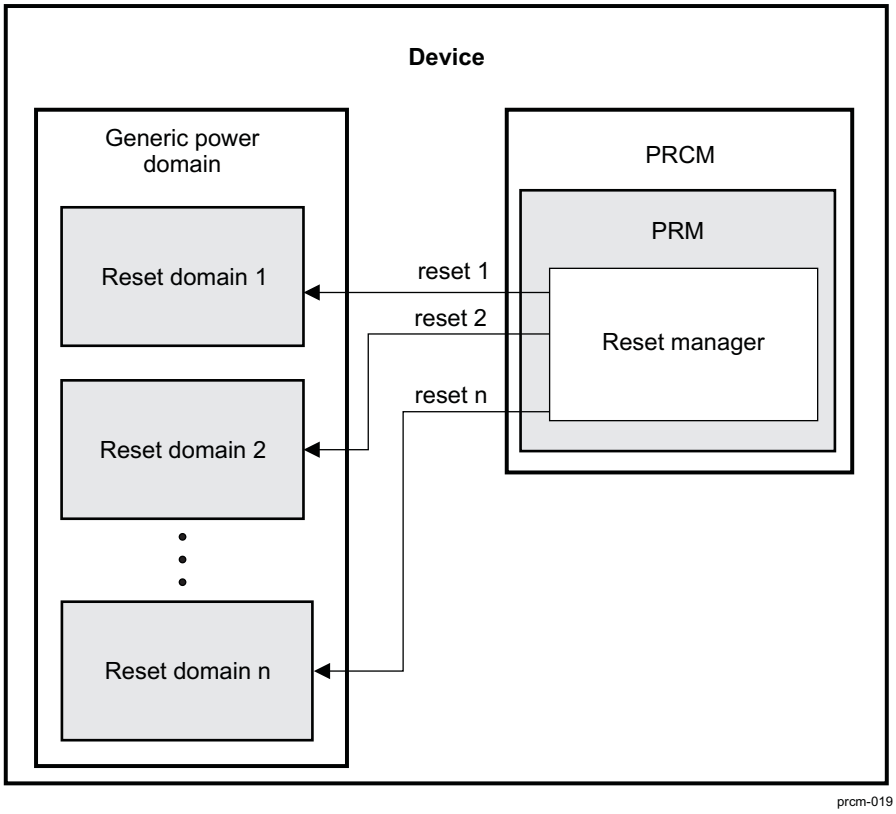
4.5.1 Overview

The PRCM module manages the reset of all modules in the device. Resets are distributed across the independent power domains (including DPLLs).

In each power domain, one or more reset domains are defined. A reset domain is defined by a unique reset signal that originates from the reset manager and is connected to one or more modules of the device. All the connected modules of the reset domain are reset simultaneously when the reset signal is asserted. Independent control of these reset domains allows sequencing the release of resets and ensures a safe reset of the entire power domain.

Figure 4-18 is an overview of the reset manager interface with a generic power domain in the device.

Figure 4-18. Reset Manager Interface With Generic Power Domain



Resets can be generated by hardware sources or software control. For a module that can be reset by software control, a software-reset bit is implemented in its <module name>_SYSCONFIG configuration register. Software reset has the same effect on the module logic as a hardware reset.

Special reset control is available to reset specific reset domains when the device operates under emulation control.

Note: All reset assertion is asynchronous, and all reset signals are active low, except for the DPLL reset signals, which are active high.

4.5.2 General Characteristics of Reset Signals

Reset signals can be categorized based on three criteria:

- Scope

- Occurrence
- Source type

4.5.2.1 Scope

A reset signal can be categorized according to its scope (the area of the device affected by that reset):

- Global reset: Affects the entire device; all modules are reset. Generally, occurs when the device powers up or an abnormal operation is detected (the eFuse bad device is detected, etc.).
- Local reset: Affects one power or reset domain. Generally, when a power domain transitions from off mode to active mode, or when a software-reset control bit for a domain is set, only the group of modules within that domain is affected.

4.5.2.2 Occurrence

A reset signal can be categorized depending on when the reset occurs:

- Cold reset: Occurs only on device power up or in certain emulation modes. The cold reset is a global reset that affects every module in the device. It usually corresponds to the initial power-on reset.
- Warm reset: Occurs when the device is in normal operating state. The warm reset is also a global reset, but it does not affect all the modules in the device. Usually, the device does not require a complete reboot on a warm reset. Several reset sources, such as the global software reset and the watchdog reset, are warm resets.

Modules that behave differently in cold reset and warm reset have two reset signals: RST and PWRON_RST. These reset signals reconstruct warm reset and cold reset in modules that require them.

For a global warm reset, the PRCM module performs the following sequence:

1. Applies a warm reset on all the modules, including modules built with RFFs.
2. Applies a warm reset on the external modem interface.
3. Drives the sys_nreswarm reset output low and holds it for a specified length of time (programmed in the PRCM.PRM_RSTTIME[7:0] RSTTIME1 bit field).
4. All ongoing transactions on the voltage control I^2C interface (I2C4 module) are aborted, and the external power IC is expected to return to nominal voltage values on receiving a sys_nreswarm reset.
5. All power domains that are in off power mode are switched to on power mode.

A global warm reset does not apply to the following modules of the device:

- SDRC
- System control module (SCM) (I/O control)
- Part of PRM and CM registers (see note below)
- 32-kHz synchronization timer
- DPLL3 (see Section 4.5.9.2)
- Emulation modules
- eFuse farm

Note: For information about the PRCM registers affected by the global warm reset, see the register mapping summary tables in Section 4.14, *PRCM Register Manual*.

4.5.2.3 Source Type

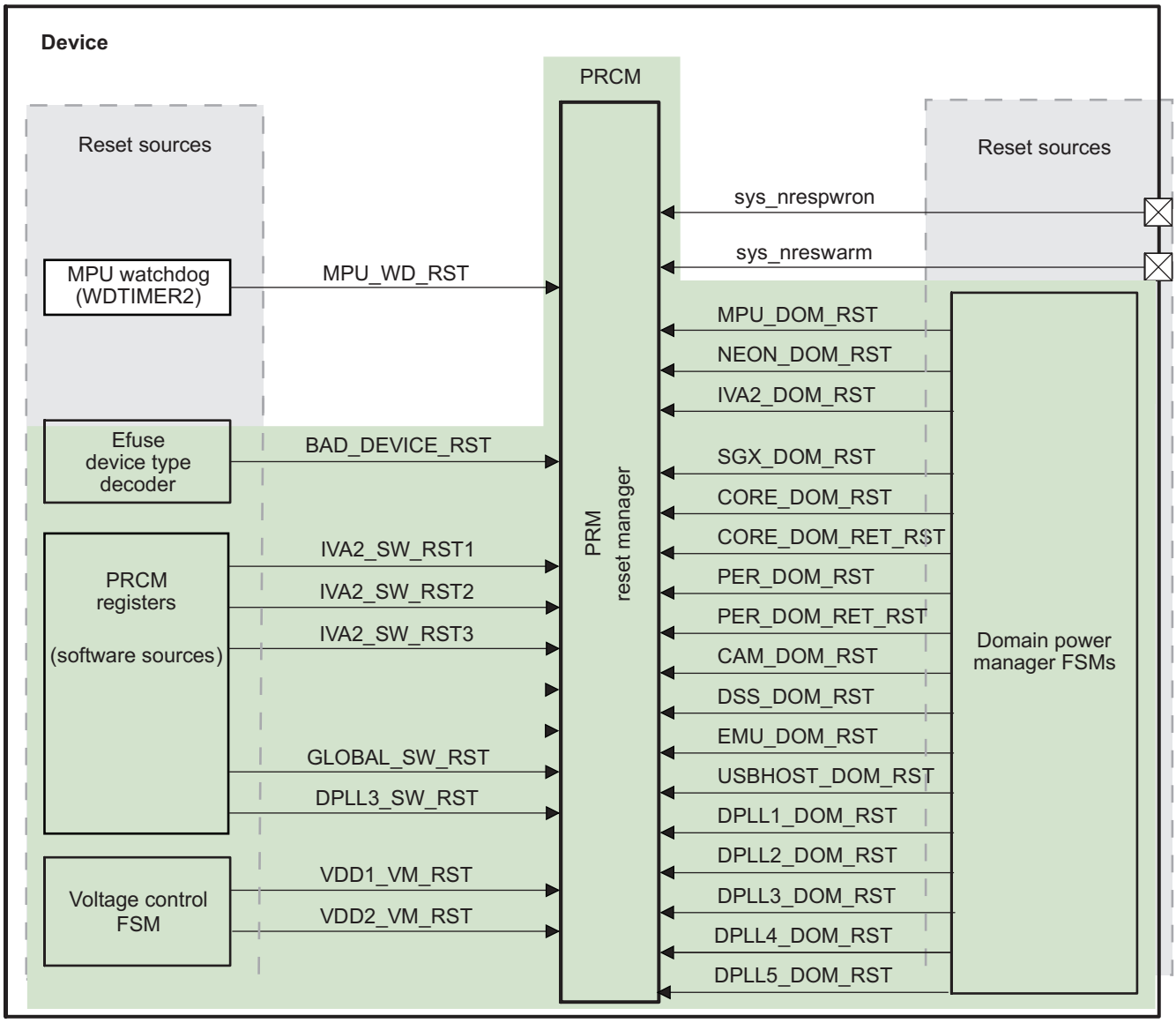
A reset can be categorized depending on whether it is software-controlled or hardware-triggered:

- Software reset: Triggered by setting a bit in a configuration register of the PRCM module
- Hardware reset: Triggered by a signal from a hardware module inside or outside the PRCM module

4.5.3 Reset Sources

Figure 4-19 is an overview of the reset sources.

Figure 4-19. Reset Sources Overview



*The green region in the figure represents the boundary of the PRCM.

prcm-020

4.5.3.1 Global Reset Sources

Table 4-7 lists the global reset sources of the device. The global reset source signals received by the reset manager trigger the reset of all the device modules. For all hardware reset signals, the source of the reset is identified; for the software reset signals, the reset triggering bit is identified.

Table 4-7. Global Reset Sources

Type[(1)]	Name	Source/Control	Description
H/C	sys_nrespwron	Input pin	The entire device is reset on power up.

(1) H = Hardware reset, S = Software reset, C = Cold reset, W = Warm reset

Table 4-7. Global Reset Sources (continued)

Type[1]	Name	Source/Control	Description
H/C	BAD_DEVICE_RST	PRCM	Asserted when during the power-up sequence the device is identified as bad, after reading eFuses
H/W	sys_nreswarm	Bidirectional pin	External hardware warm reset
H/W	MPU_WD_RST	WDTIMER2	MPU watchdog timer overflow reset
S/W	GLOBAL_SW_RST	PRCM.PRM_RSTCTRL[1] RST_GS	Global software reset
H/W	VDD1_VM_RST	PRCM	Asserted by the voltage manager FSMs when no response from the power IC is received during wake-up transition from retention or off mode
H/W	VDD2_VM_RST	PRCM	
S/W	DPLL3_SW_RST	PRCM.PRM_RSTCTRL[2] RST_DPLL3	Local cold reset for DPLL3 and a global cold reset to the device

4.5.3.2 Local Reset Sources

Table 4-8 lists the local reset sources of the device. A local reset source signal received by the reset manager resets only some of the device modules.

Table 4-8. Local Reset Sources

Type[1]	Name	Source/Control	Description
H/C	CORE_DOM_RET_RST	PRCM	Asserted only for a power domain state transition from off to active state
H/C	USB_DOM_RET_RST	PRCM	
H/C	PER_DOM_RET_RST	PRCM	
H/C	MPU_DOM_RST	PRCM	Asserted for any power domain transition from off or retention state to active state
H/C	IVA2_DOM_RST	PRCM	
H/C	NEON_DOM_RST	PRCM	
H/C	SGX_DOM_RST	PRCM	
H/C	CORE_DOM_RST	PRCM	
H/C	PER_DOM_RST	PRCM	
H/C	CAM_DOM_RST	PRCM	
H/C	DSS_DOM_RST	PRCM	
H/C	DPLL1_DOM_RST	PRCM	
H/C	DPLL2_DOM_RST	PRCM	
H/C	DPLL3_DOM_RST	PRCM	
H/C	DPLL4_DOM_RST	PRCM	
H/C	DPLL5_DOM_RST	PRCM	
S/W	IVA2_SW_RST1	PRCM.RM_RSTCTRL_IVA2[0] RST1_IVA2	IVA2.2: DSP reset control
S/W	IVA2_SW_RST2	PRCM.RM_RSTCTRL_IVA2[1] RST2_IVA2	IVA2.2: MMU reset control and video sequencer hardware accelerator reset control
S/W	IVA2_SW_RST3	PRCM.RM_RSTCTRL_IVA2[2] RST3_IVA2	Video sequencer reset control

(1) H = Hardware reset, S = Software reset, C = Cold reset, W = Warm reset

Notes:

- For power domains with <domain name>_DOM_RST and <domain name>_DOM_RET_RST, the reset sources are asserted together when the domain transitions from off to on power state, whereas only <domain name>_DOM_RET_RST is asserted on a global or local warm reset.
- Because the modem reset signals are not supported in the device stand-alone configuration, they are not discussed in this section.

4.5.4 Reset Distribution

Each power domain can contain one power-on reset (RSTPWRON) and one or more reset (RST) signals. These signals behave as follows:

- On any global or local cold reset, RST and RSTPWRON are asserted.
- On any global or local warm reset, only RST is asserted.

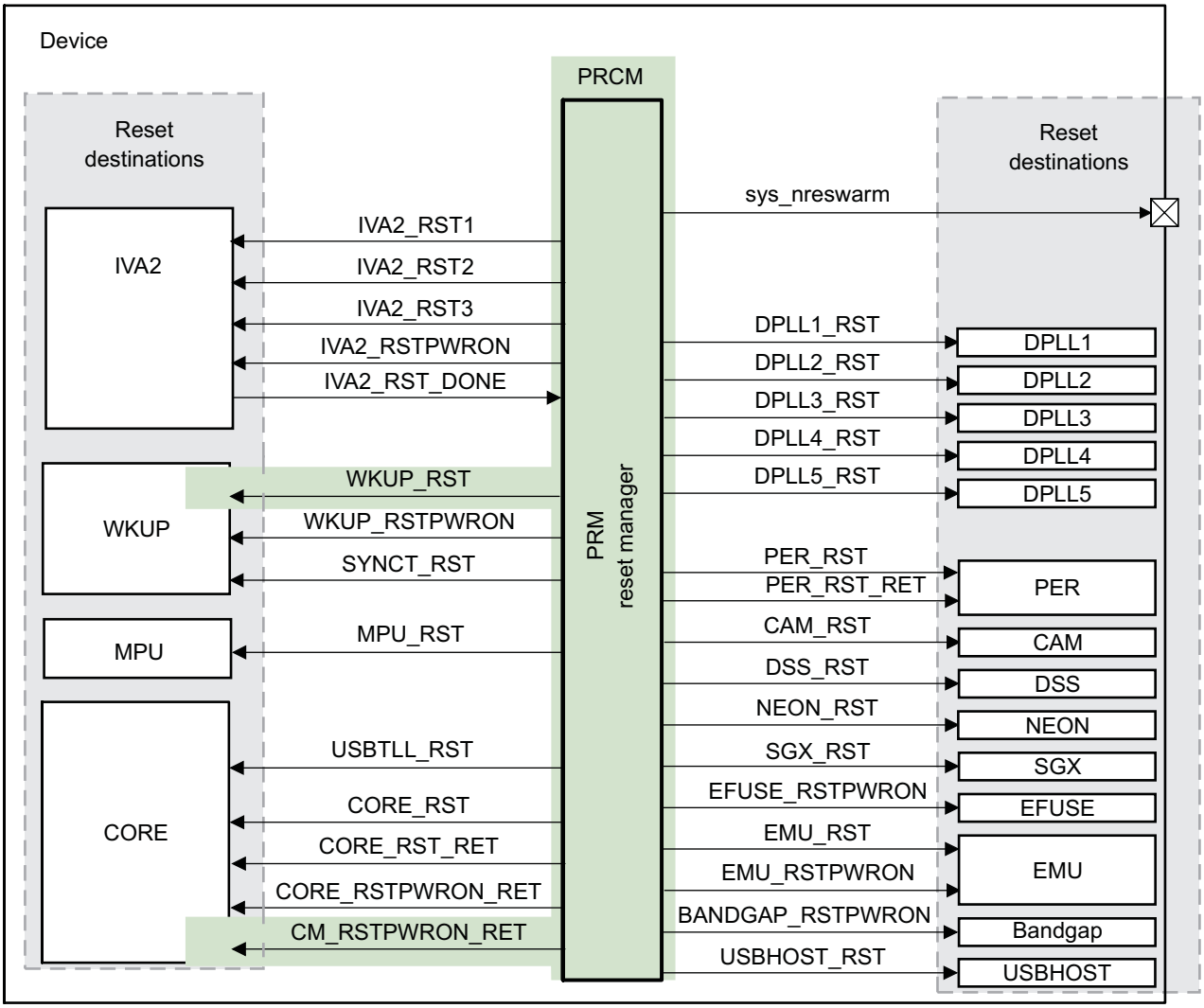
The CORE power domain receives two additional retention logic reset signals: retention reset (RST_RET) and power-on retention reset (RSTPWRON_RET). These signals behave as follows:

- On any global cold reset or wakeup from off state to active state, RST_RET and RSTPWRON_RET are asserted.
- On any global warm reset, only RST_RET is asserted.
- On wakeup from retention state, these signals are not asserted.

The IVA2 power domain outputs the IVA2_RST_DONE signal, which handles the synchronous reset scheme of the IVA2.2 subsystem.

Figure 4-20 shows the reset distribution among the power domains.

Figure 4-20. Reset Destination Overview



*The green region in the figure represents the boundary of the PRCM

prcm-021

4.5.5 Power Domain Reset Descriptions

4.5.5.1 MPU Power Domain

The MPU power domain has one reset output signal (see Table 4-9).

Table 4-9. MPU Power Domain Reset Signal

Name	I/O[1]	Source/Destination[2]	Reset Domain
MPU_RST	I	PRM	Resets the MPU processor core and the asynchronous bridge in the MPU power domain

(1) I = Input; O = Output
(2) Source for an input signal and destination for an output signal

4.5.5.2 NEON Power Domain

The NEON power domain has one reset input signal (see Table 4-10).

Table 4-10. NEON Power Domain Reset Signal

Name	I/O[1]	Source	Reset Domain
NEON_RST	I	PRM	Resets the NEON coprocessor

(1) I = Input; O = Output

4.5.5.3 IVA2 Power Domain

The IVA2 power domain has four inputs and one output reset signal (see Table 4-11).

Table 4-11. IVA2 Power Domain Reset Signals

Name	I/O[1]	Source/Destination[2]	Reset Domain
IVA2_RST1	I	PRM	Resets the IVA2.2 DSP and part of the two asynchronous bridges in the IVA2 power domain
IVA2_RST2	I	PRM	Resets the IVA2.2 MMU
IVA2_RST3	I	PRM	Resets the video sequencer module
IVA2_RSTPWRON	I	PRM	Performs a power-on reset on the IVA2.2 subsystem. Active on a cold reset only.
IVA2_RSTDONE	O	PRM	Release condition of the IVA_RST1 and RST2. Generated by the IVA2.2 subsystem at the end of the initialization sequence.

(1) I = Input; O = Output
(2) Source for an input signal and destination for an output signal

4.5.5.4 CORE Power Domain

The CORE power domain has eight inputs and one output reset signal (see Table 4-12).

Table 4-12. CORE Power Domain Reset Signals

Name	I/O[1]	Source/Destination[2]	Reset Domain
CORE_RST	I	PRM	Resets parts of the three asynchronous bridges, MPU INTC, IVA2.2 WUGEN, interconnects, ICR, modem INTC, SAD2D, mailboxes, GPMC, OCM, UART[1,2], HDQ, HS USB, I2C[1..3], McBSP 1 and 5, McSPI [1..3], MMC[1..3], GPTIMER[10,11]
CORE_RST_RET	I	PRM	Resets part of the SDRC, SDMA, SMS, MPU INTC, and IVA2 WUGEN
CORE_RSTPWRON_RET	I	PRM	Resets part of the SDRC and SCM
CM_RSTPWRON_RET	I	PRM	Resets the clock manager
USBTLL_RST	I	PRM	Resets the USB TLL asynchronously

(1) I = Input; O = Output
(2) Source for an input signal and destination for an output signal

The CM logic is reset on:

- Any global cold reset
- A CORE power domain transition from off to on

Because the CM logic is not reset in this case, CM registers that are sensitive to a warm reset must also be reset synchronously with the L4 clock when a global warm reset occurs.

4.5.5.5 DSS Power Domain

The DSS power domain has one reset input signal (see Table 4-13).

Table 4-13. DSS Power Domain Reset Signal

Name	I/O[1]	Source	Reset Domain
DSS_RST	I	PRM	Resets the entire display subsystem

(1) I = Input; O = Output

4.5.5.6 CAM Power Domain

The CAM power domain has one reset input signal (see Table 4-14).

Table 4-14. CAM Power Domain Reset Signal

Name	I/O[1]	Source	Reset Domain
CAM_RST	I	PRM	Resets the entire camera subsystem

(1) I = Input; O = Output

4.5.5.7 USBHOST Power Domain

The USBHOST power domain has one reset input signal (see Table 4-15).

Table 4-15. USBHOST Power Domain Reset Signal

Name	I/O[1]	Source	Reset Domain
USBHOST_RST	I	PRM	Resets the entire HS USB host subsystem

(1) I = Input; O = Output

4.5.5.8 SGX Power Domain

The SGX power domain has one reset input signal (see Table 4-16).

Table 4-16. SGX Power Domain Reset Signal

Name	I/O[1]	Source	Reset Domain
SGX_RST	I	PRM	Resets the entire SGX subsystem

(1) I = Input, O = Output

4.5.5.9 WKUP Power Domain

The WKUP power domain has three reset input signals and two reset output signals (see Table 4-17).

Table 4-17. WKUP Power Domain Reset Signals

Name	I/O[1]	Source/Destination[2]	Reset Domain
WKUP_RST	I	PRM	Resets the GPTIMER1, WDTIMER2, GPIO 1
WKUP_RSTPWRON	I	PRM	Resets the wake-up control module
MPU_WD_RST	O	PRM	Global warm reset for PRM. Generated by WDTIMER2.

(1) I = Input; O = Output
(2) Source for an input signal and destination for an output signal

The PRM logic is reset on any global cold reset. Because the PRM logic is not reset in this case, PRM registers that are sensitive to a warm reset must also be reset synchronously with the system clock when a global warm reset occurs.

4.5.5.10 PER Power Domain

The PER power domain has two reset input signals (see Table 4-18).

Table 4-18. PER Power Domain Reset Signal

Name	I/O[1]	Source	Reset Domain
PER_RST	I	PRCM	Resets the UART3, McBSP[2,3,4], GPTIMER[2,..,9], WDTIMER3 modules
PER_RST_RET	I	PRCM	Resets the GPIO [2,..,6] modules

(1) I = Input; O = Output

4.5.5.11 DPLL Power Domains

The DPLL power domains for DPLL1, DPLL2, DPLL3, DPLL4, and DPLL5 each have one reset input signal.

Table 4-19. DPLL Power Domain Reset Signals

Name	I/O[1]	Source	Reset Domain
DPLL1_RSTPWRON	I	PRCM	Resets DPLL1
DPLL2_RSTPWRON	I	PRCM	Resets DPLL2
DPLL3_RSTPWRON	I	PRCM	Resets DPLL3
DPLL4_RSTPWRON	I	PRCM	Resets DPLL4
DPLL5_RSTPWRON	I	PRCM	Resets DPLL5

(1) I = Input; O = Output

They are asserted for any type of global cold reset.

4.5.5.12 EFUSE Power Domain

The EFUSE power domain has one reset input signal (see Table 4-20).

Table 4-20. EFUSE Power Domain Reset Signal

Name	I/O[1]	Source	Reset Domain
EFUSE_RSTPWRON	I	PRCM	Resets the eFuse controller

(1) I = Input; O = Output

This signal is asserted for any type of global cold reset and when the device wakes up from off mode.

4.5.5.13 BANDGAP Logic

The BANDGAP logic has one reset input signal (see Table 4-21).

Table 4-21. BANDGAP Logic Reset Signal

Name	I/O[1]	Source	Reset Domain
BANDGAP_RSTPWRON	I	PRCM	Resets the BANDGAP logic

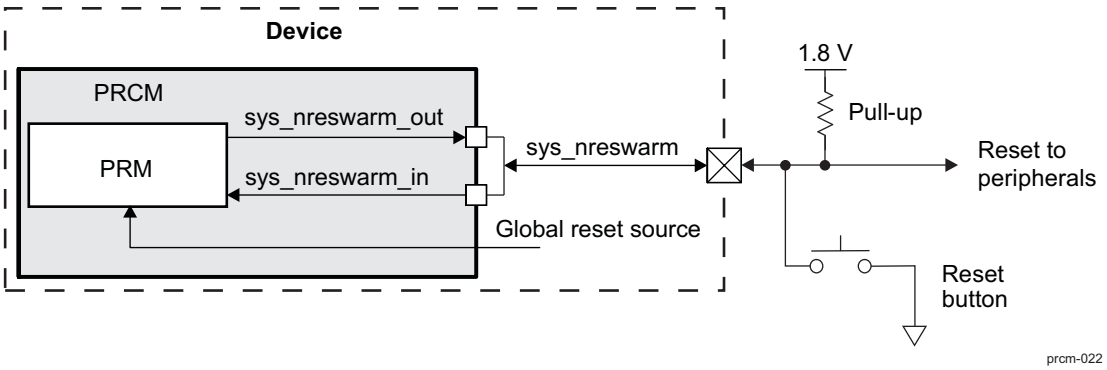
(1) I = Input; O = Output

This signal is asserted for any type of global cold reset and when the device wakes up from off mode.

4.5.5.14 External Warm Reset Assertion

Figure 4-21 shows the external warm reset interface.

Figure 4-21. External Warm Reset Interface



Any global reset source (internal or external) causes sys_nreswarm_out to be driven and maintained at the boundary of the device for at least the amount of time configured in the PRCM.PRM_RSTTIME[7:0] RSTTIME1 bit field. This ensures that the device and its related peripherals are reset together.

Note: Because the system warm-reset output is implemented on a bidirectional pad, any input pulse on sys_nreswarm causes a global warm reset.

4.5.6 Reset Logging

A reset of the device is logged in two ways. First, dedicated registers in the PRCM module (the RM_RSTST_power domain> and PRM_RSTST registers) log the reset source. Second, the SCM logs the device reset activity in dedicated registers.

4.5.6.1 PRCM Reset Logging Mechanism

The reset status registers (RM_RSTST_power domain> and PRM_RSTST) are reset asynchronously on assertion of a global cold reset. However, a reset status bit is always logged when the reset is released to the domain.

For this reason, after the assertion of a global cold reset, the reset status register is cleared to 0. When the domain reset is released, the register bit to log the global cold reset (the RM_RSTST_<power domain> [0] GLOBALCOLD_RST bit) is updated to 1. For the same reason, the reset status register of domains released from reset by software is updated only when software releases the domain reset.

The assertion of a global cold reset prevents logging any other source of reset until after the release of the domain reset. This is valid in the following situations:

- A source of reset other than global cold reset is asserted before, during, or after the active period of a global cold source of reset and before the release of the domain reset signal.
- A source of reset other than global cold reset was asserted and then released, but a global cold reset source was asserted before the release of the domain reset signal.

4.5.6.2 SCM Reset Logging

The PRM exports reset the activity status signals to the SCM. The SCM uses these signals to log a reset status in the SCM.CONTROL_SEC_STATUS register. The reset activity status signal is asserted high when any source of reset to the power domain is active.

It also provides reset status for the following global reset signals:

- Global cold reset
- Global warm reset

There is one reset activity status signal for each of the following power domains:

- CAM
- CORE
- DSS
- EMU
- SGX
- IVA2
- MPU
- NEON
- PER
- USBHOST

These signals are asserted high on assertion of any source of reset on the domain and logged. For information about the SCM, see Chapter 7, *System Control Module*.

4.5.7 Reset Management Overview

Reset management in the device is a 2-layered structure composed of multiple instances of the reset manager. In the first layer, a top-level reset manager, called the device reset manager, handles all sources of global reset (cold and warm). It provides reset managers for the second layer, called local reset managers, and the global reset and global power-on reset signals.

Each power domain has a minimum of one local reset manager. The local reset manager also receives resets, such as the software reset and domain power transition reset, from the local reset source for the power domain.

Figure 4-22 through Figure 4-25 provide an overview of reset management in the device. They do not provide reset sequencing between the reset managers.

Note: The power domain must be ready (the domain clocks must be active) before its reset is released.

Figure 4-22. Device Reset Manager Overview

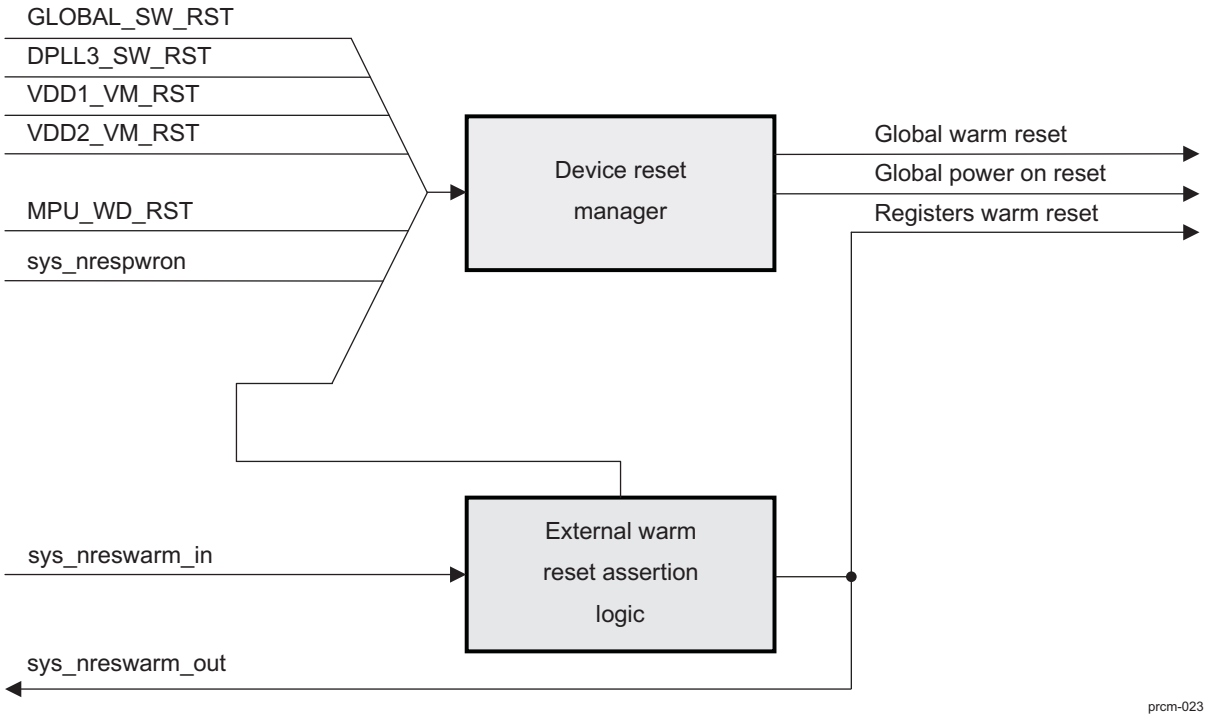


Figure 4-23. Power Domain Reset Management: Part 1

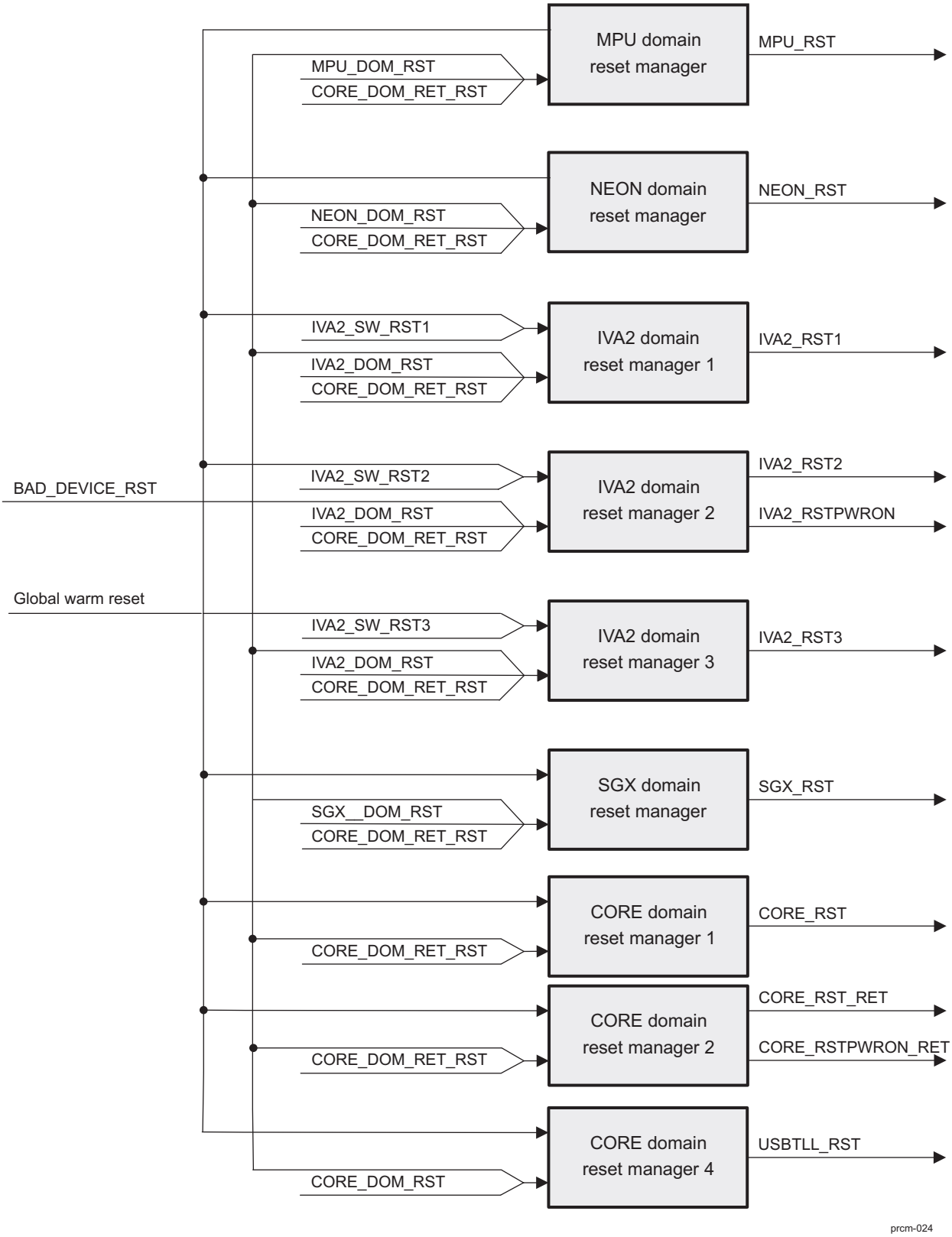


Figure 4-24. Power Domain Reset Management: Part 2

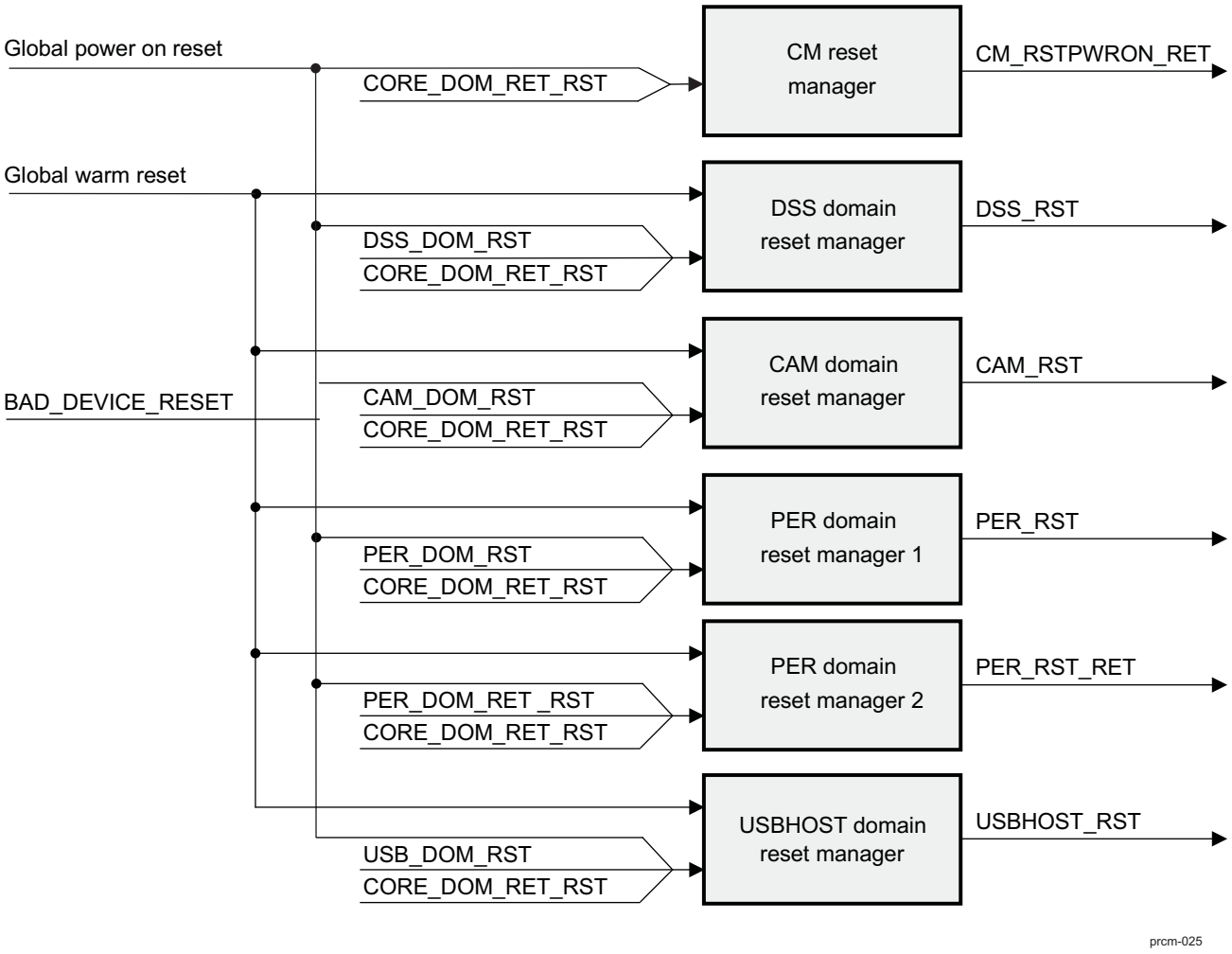
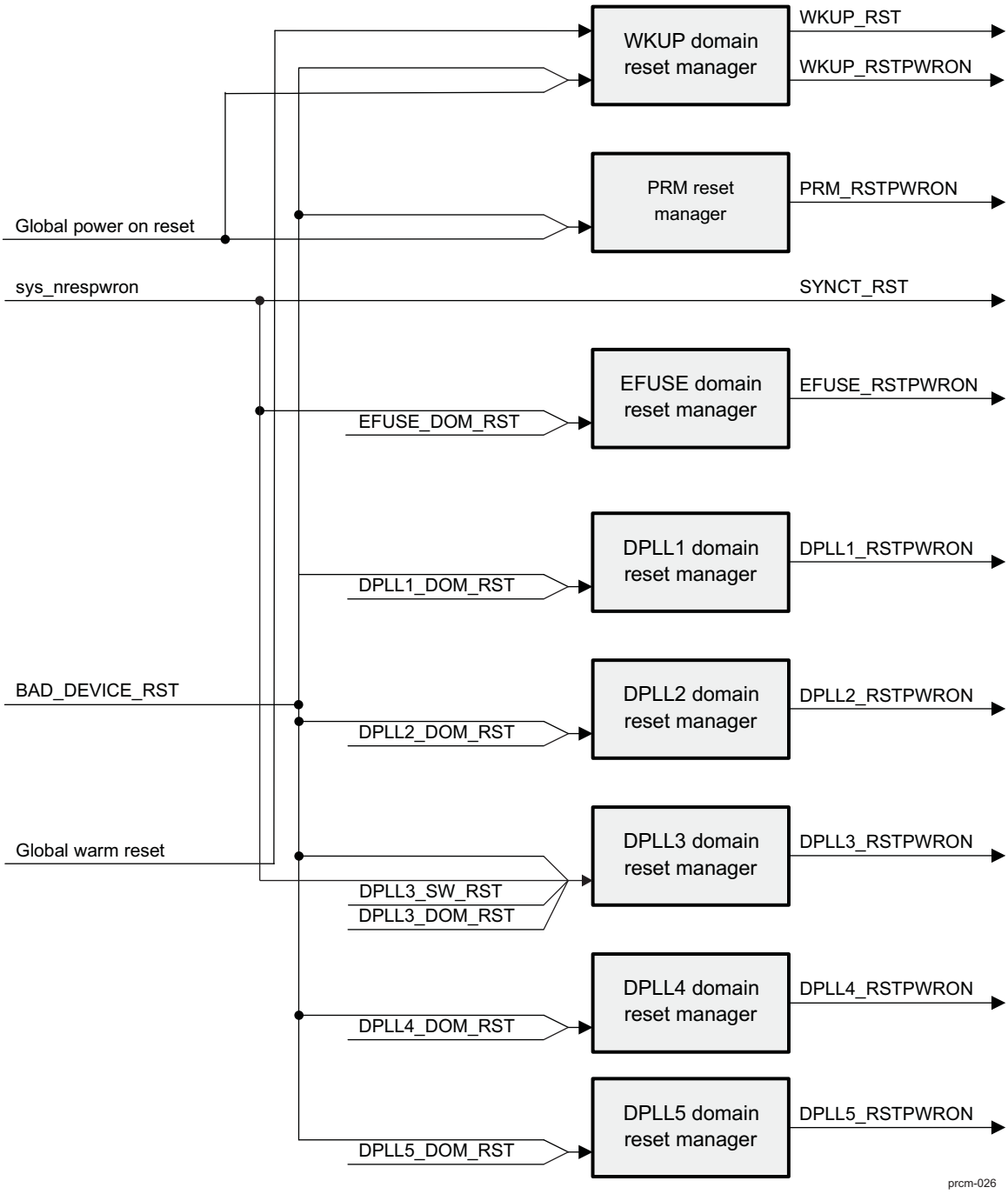


Figure 4-25. Power Domain Reset Management: Part 3



4.5.8 Reset Summary

Table 4-22 and Table 4-23 summarize the sources of global and local resets and their actions on the reset signals.

Table 4-22. Global Reset Summary[1]

Domain Resets		Reset Sources						
Power Domain	Signal	Cold Reset		Warm Reset				
		sys_nres pweron	DPLL3_ SW_RST	sys_nres warm_in	MPU_WD_ RST	GLOBAL_ SW_RST	VDD1_VM_ RST	VDD2_VM_ RST
MPU	MPU_RST							
NEON	NEON_RST							
IVA2	IVA2_RST1							
	IVA2_RST2							
	IVA2_RST3							
	IVA2_RSTPWRON							
SGX	SGX_RST							
CORE	CORE_RST							
	CORE_RSTPWRON							
	CORE_RST_RET							
	CORE_RSTPWRON_RET							
	CM_RSTPWRON_RET							
	USBTLL_RST							
WKUP	WKUP_RST							
	SYNCT_RST							
PER	PER_RST							
	PER_RST_RET							
DSS	DSS_RST							
CAM	CAM_RST							
USBHOST	USBHOST_RST							
DPLL1	DPLL1_RSTPWRON							
DPLL2	DPLL2_RSTPWRON							
DPLL3	DPLL3_RSTPWRON							
DPLL4	DPLL4_RSTPWRON							
DPLL5	DPLL5_RSTPWRON							
EFUSE	EFUSE_RSTPWRON							
BANDGAP	BANDGAP_RSTPWRON							
Device pad (output)	sys_nreswarm_out							

(1) The shaded blocks identify the power domain reset signals triggered as a result of the reset source signal (at the head of the column).

Table 4-23. Local Reset Summary[(1)]

Domain Resets		Reset Sources						
Power Domain	Signal	Cold Reset				Warm Reset		
		CORE_DOM_RET_RST	PER_DOM_RET_RST	DPLL3_SW_RST	BAD_DEVICE_RESET	IVA2_SW_RST1	IVA2_SW_RST2	IVA2_SW_RST3
MPU	MPU_RST	■			■			
NEON	NEON_RST	■			■			
IVA2	IVA2_RST1	■			■	■		
	IVA2_RST2	■			■		■	
	IVA2_RST3	■			■			■
	IVA2_RSTPWRON	■			■			
SGX	SGX_RST	■			■			
CORE	CORE_RST	■			■			
	CORE_RST_RET	■			■			
	CORE_RSTPWRON_RET	■			■			
	CM_RSTPWRON_RET				■			
	USBTLL_RST	■			■			
WKUP	WKUP_RST				■			
	SYNCT_RST							
PER	PER_RST	■	■		■			
	PER_RST_RET	■	■		■			
DSS	DSS_RST	■			■			
CAM	CAM_RST	■			■			
USBHOST	USBHOST_RST	■			■			
DPLL1	DPLL1_RSTPWRON	■			■			
DPLL2	DPLL2_RSTPWRON	■			■			
DPLL3	DPLL3_RSTPWRON	■		■	■			
DPLL4	DPLL4_RSTPWRON	■			■			
DPLL5	DPLL5_RSTPWRON	■			■			
EFUSE	EFUSE_RSTPWRON							
BANDGAP	BANDGAP_RSTPWRON							
Device pad (output)	sys_nreswarm_out			■	■			

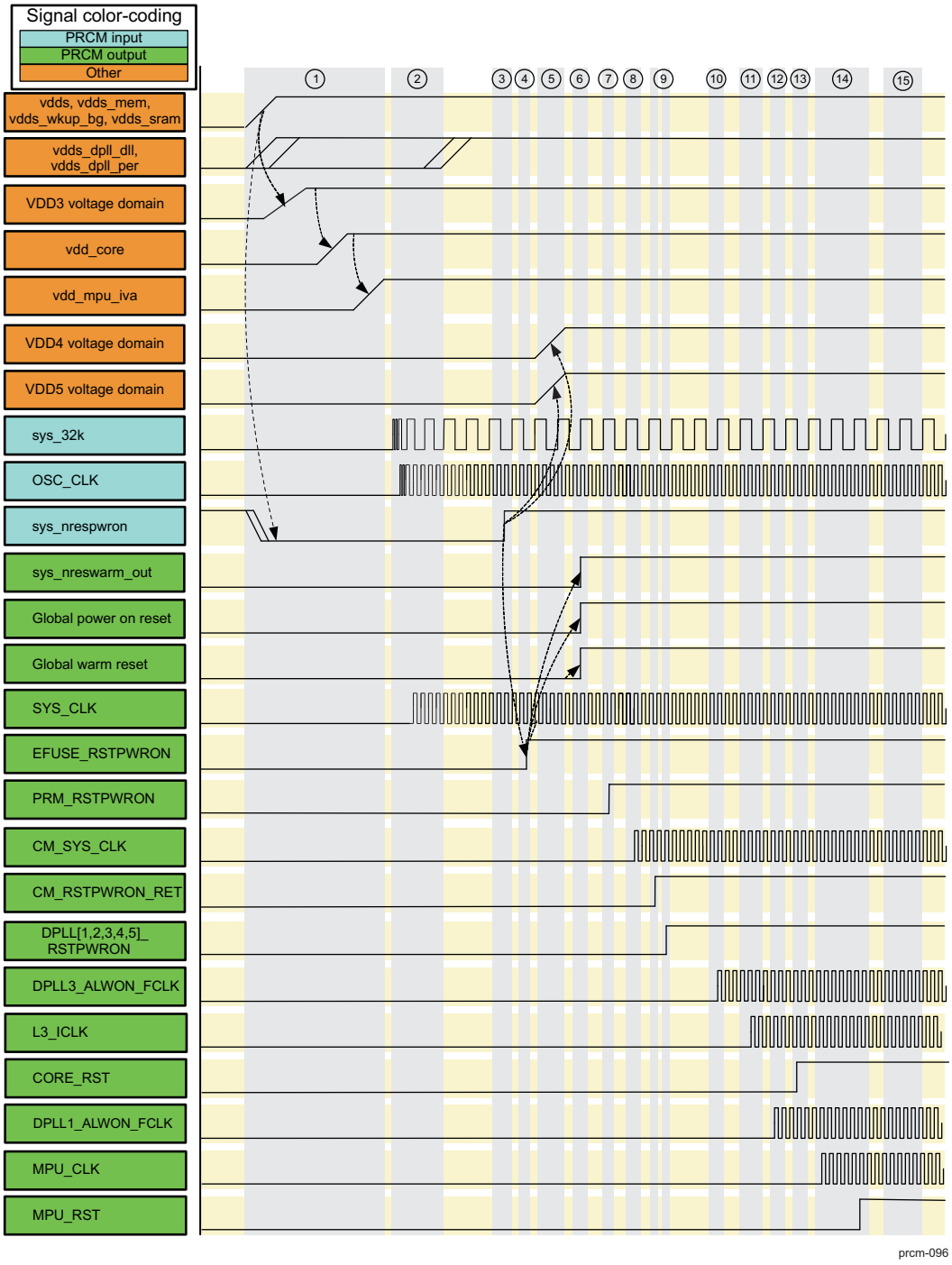
(1) The shaded blocks identify the power domain reset signals triggered as a result of the reset source signal (at the head of the column).

4.5.9 Reset Sequences

4.5.9.1 Power-Up Sequence

Figure 4-26 shows the power-up sequence.

Figure 4-26. Power-Up Sequence



The power-up sequence shown in Figure 4-26 is:

1. Voltages ramp up and global power-on reset is asserted by the external power IC:
   - vdds, vdds_mem, vdds_wkup_bg, vdds_sram voltage rails are ramped up.

- sys_nrespwron is asserted (set to low).
- vdds_dpll_dll and vdds_dpll_per voltage rails are ramped up before release of sys_nrespwron (transition to high).
- The VDD3 voltage domain LDO (WKUP power domain) tracks vdds_wkup voltage rail and automatically ramps up.
- After VDD3 voltage domain stabilization, the power IC ramps up VDD2 voltage domain.
- After VDD2 voltage domain stabalization, the power IC ramps up VDD1 voltage domain.

2. Source clock stabilizes:
   - The 32-kHz input clock and the system clock oscillator stabilize.
   - The device reset manager holds the entire device under reset.
   - The system clock (SYS_CLK) is on.
3. The global power-on reset sys_nrespwron released (to high) when vdds_dpll_dll and vdds_dpll_per voltage rails are stabilized.
4. The eFuse farm reset is released.
5. Internal memory LDOs ramp up:
   - Processor memory LDO (VDD4 voltage domain) is ramped up.
   - CORE memory LDO (VDD5 voltage domain) is ramped up.
   - The PRCM module waits for internal memory LDO stabilization.
6. Global resets are released. Global power-on reset and global warm reset are extended (remain asserted) on release of the external power-on reset until the following conditions are met:
   - Voltages are stable in the processor power domains, CORE power domain, and WKUP power domain.
   - System clock is stable.
   - Internal memory LDO is stable.
   - Device reset manager counter overflows (set up by the PRCM.PRM_RSTTIME[7:0] RSTTIME1 bit field).
   - Hardware conditions, such as eFuse farm ready, are set.
7. The PRM_RSTPWRON reset is released.
8. The CM_SYS_CLK clock starts running (gating logic is enabled).
9. The CM_RSTPWRON_RET reset is released.
10. The reset of the DPLLs is released.
11. The DPLL3_ALWON_FCLK clock starts to run (gating logic is enabled).
12. The L3_ICLK clock starts to run.
13. The DPLL1_ALWON_FCLK clock starts to run (logic controlling the clock-gating conditions element is clocked and the clock is requested).
14. The CORE_RST reset is released.
15. The MPU_CLK clock starts to run.
16. The MPU boots.

Notes:

- The IVA2 power domain is held under reset after power up by assertion of the software source of reset.
- Power domains such as DSS, CAM, SGX, and NEON are held under reset after power up until the MPU software enables the power domain interface clocks.

4.5.9.2 Global Warm Reset Sequence

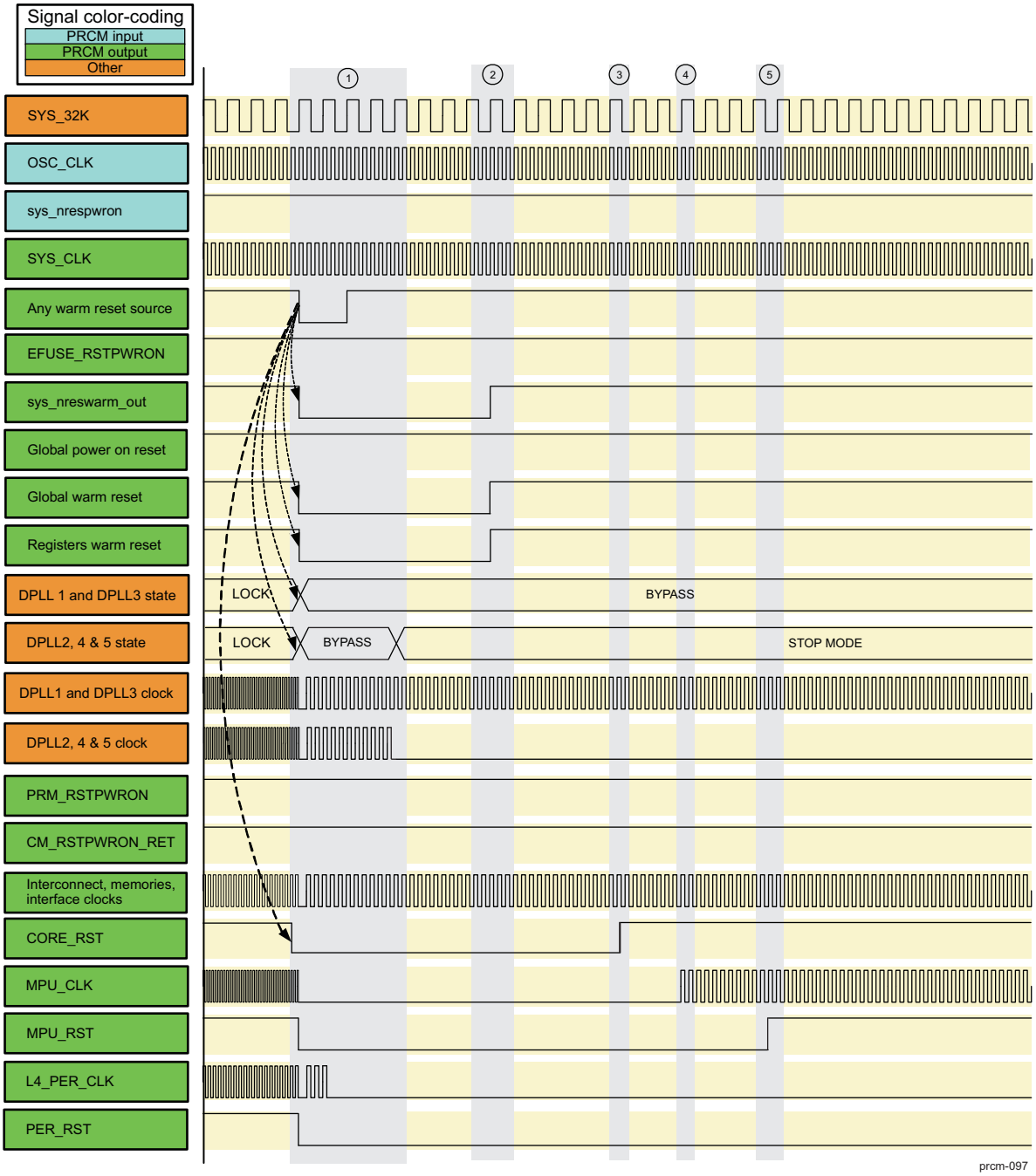
This section describes the global reset sequence.

The assumptions are:

- vdds, vdds_mem, vdds_dpll_pll, vdds_dpll_per, vdds_wkup_bg and vdds_sram power rails are regulated at 1.8 V.
- The VDD2 voltage domain (vdd_core power rail) is regulated at 1.0 V.
- The VDD1 voltage domain (vdd_mpu_iva power rail) is regulated at 1.2 V.
- The VDD3, VDD4, and VDD5 voltage domains are regulated at 1.2 V.
- The system is running:
  - Resets are released.
  - CORE DPLL and processor DPLL are locked.

Figure 4-27 shows the global warm reset sequence.

Figure 4-27. Warm Reset Sequence



The steps of a global warm reset sequence are as follows:

1. On assertion of the warm reset source:
   - The device reset manager resets part of the device by asserting the global warm reset.
   - The external warm reset (sys_nreswarm_out) is asserted.
   - All the power domain warm resets are asserted.
   - DPLL1 and DPLL3 transit to bypass mode. DPLL2, DPLL4, and DPLL5 transit to stop mode. The system clock, SYS_CLK, continues running at system clock frequency.
   - The registers sensitive to a warm reset are synchronously reset (PRM and CM register sets).
   - The CM cuts all the clocks not requested, according to the register reset value settings.

2. The global warm reset is released and extended after release of the warm reset source until the following conditions are met:
   - Device reset manager counter overflows (set up by the PRCM.PRM_RSTTIME[7:0] RSTTIME1 bit field).
   - Voltage domains (VDD1, VDD2, VDD4 and VDD5) are stable.

   Note: Voltage stabilization is an additional condition if voltage scaling was performed before the assertion of the warm reset.
3. The CORE domain is released from reset (warm sensitive modules in CORE power domain).
4. The MPU_CLK clock is running.
5. The MPU power domain is released from reset. The MPU boots.

Notes:
- The IVA2 power domain is held under reset after global warm reset by assertion of the software source of the reset.
- Power domains such as PER, DSS, CAM, SGX, and NEON are held under reset after global warm reset until the MPU software enables their interface clock.

4.5.9.3 IVA2.2 Subsystem Power-Up Sequence

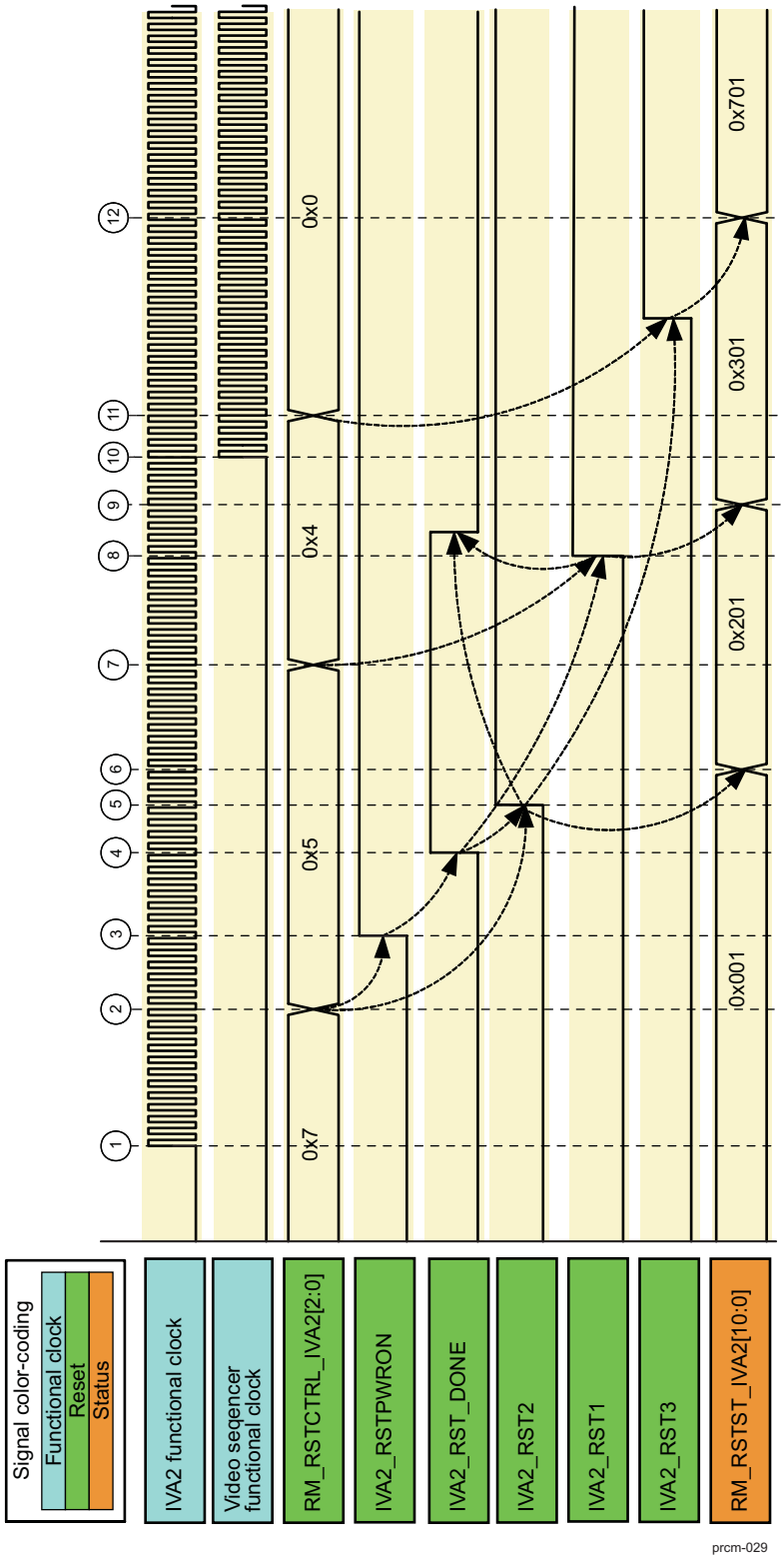
This section describes the power-up reset sequence and timing relationships of the IVA2.2 subsystem.

The assumptions are:

- The MPU is running.
- All sources of reset to the IVA2.2 subsystem are released except for the software sources of reset.

Figure 4-28 shows the IVA2.2 power-up sequence.

Figure 4-28. IVA2.2 Subsystem Power-Up Reset Sequence



The sequence is:

1. Software enables the IVA2.2 subsystem clock.

2. Software clears the PRCM.RM_RSTCTRL_IVA2[1] RST2_IVA2 bit. The PRM module releases the IVA2_RSTPWRON reset signal when reset manager 2 in the IVA2 power domain times out.
3. On release of IVA2_RSTPWRON, the IVA2.2 subsystem performs an initialization sequence.
4. The IVA2.2 subsystem asserts the IVA2_RST_DONE signal when initialization completes.
5. The PRM module releases the IVA2_RST2 reset signal.
6. The PRCM.RM_RSTST_IVA2[9] IVA2_SW_RST2 status bit is updated accordingly on release of the IVA2_RST2 reset signal. The MPU software can now program the MMU or download the DSP code.
7. Software clears the PRCM.RM_RSTCTRL_IVA2[0] RST1_IVA2 bit. The PRM module waits for reset manager 1 in the IVA2 power domain to time out.
8. The PRM module releases the IVA2_RST1 reset signal. The DSP boots.
9. The PRCM.RM_RSTST_IVA2[8] IVA2_SW_RST1 status bit is updated accordingly on release of the IVA2_RST1 reset signal.
10. DSP software enables the video sequencer (SEQ) clock.
11. DSP software clears the PRCM.RM_RSTCTRL_IVA2[2] RST3_IVA2 bit. The PRM waits for reset manager 3 in the IVA2 power domain to time out.
12. After reset manager 3 times out, the PRM can release the IVA2_RST3 reset signal. The SEQ boots.
13. The PRCM.RM_RSTST_IVA2[10] IVA2_SW_RST3 bit is updated accordingly on release of the IVA2_RST3 reset signal.

4.5.9.4 IVA2 Software Reset Sequence

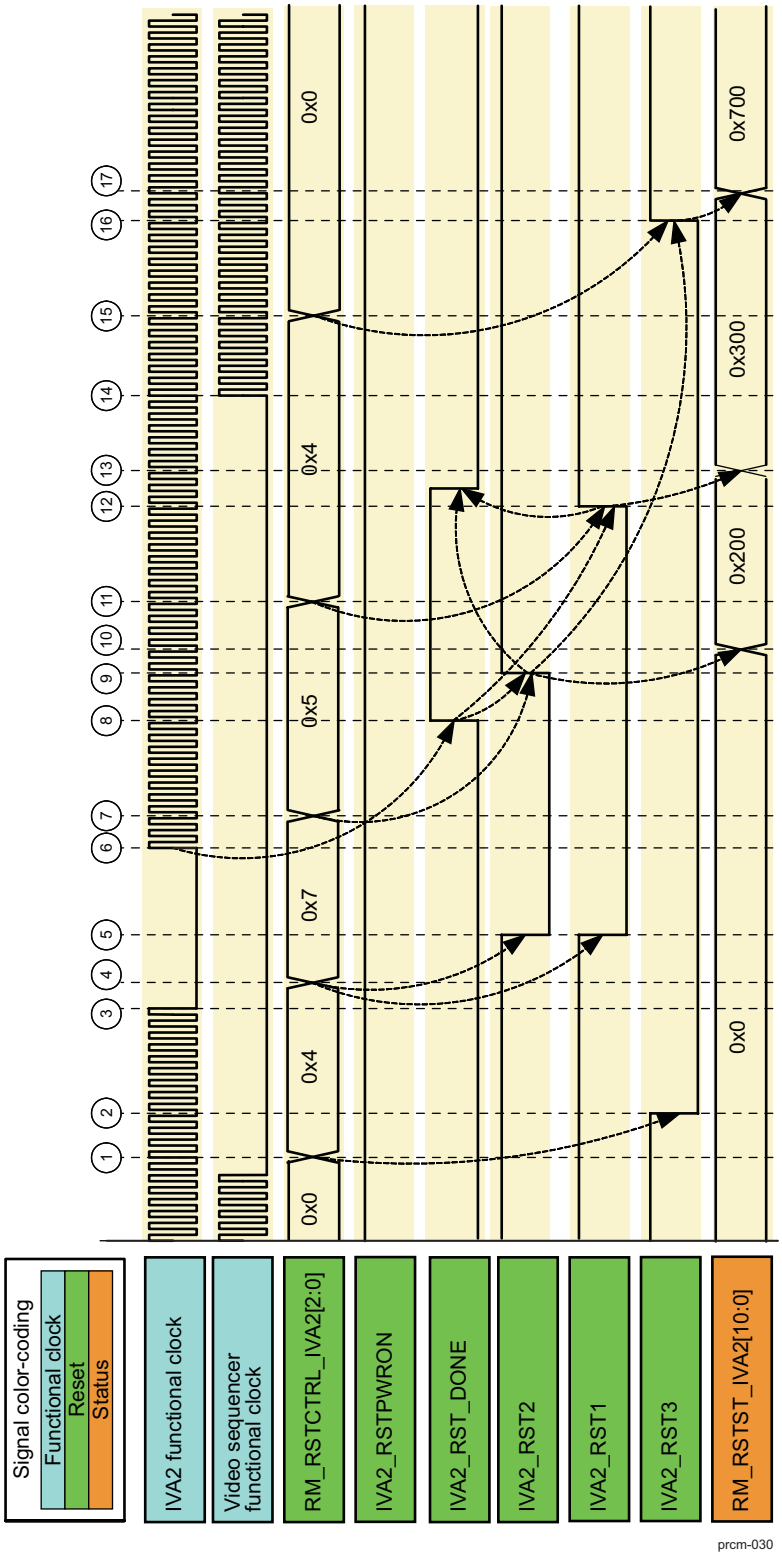
This section describes the software reset sequence and timing relationships of the IVA2.2 subsystem.

The assumptions are:

- The MPU is running.
- All sources of reset to the IVA2.2 subsystem are released.
- The software ensures that the IVA2.2 subsystem software sources of reset are not asserted while the IVA2 power domain clocks are running.
- The software clears the previous reset status.

Figure 4-29 shows the IVA2 software reset sequence.

Figure 4-29. IVA2 Software Reset Sequence



The sequence is:

1. DSP software puts the SEQ in idle, and can safely assert the IVA2 power domain software reset.
2. The PRM module asserts the IVA2_RST3 reset asynchronously.

3. The DSP goes to idle, and the CM gates the IVA_CLK clock.
4. MPU software asserts the IVA2.2 subsystem software reset.
5. The PRM asserts all remaining IVA2 power domain warm resets asynchronously.
6. MPU software enables the IVA2 power domain clock. A partial initialization sequence is performed.
7. MPU software clears the PRCM.RM_RSTCTRL_IVA2[1] RST2_IVA2 bit.
8. The IVA2.2 subsystem asserts the IVA2_RST_DONE signal when initialization completes.
9. The PRM module releases the IVA2_RST2 reset signal when reset manager 2 in the IVA2 power domain times out.
10. The PRCM.RM_RSTST_IVA2[9] IVA2_SW_RST2 status bit is updated accordingly on release of the IVA2_RST2 reset signal. MPU software now programs the MMU or downloads the DSP code.
11. Software clears the PRCM.RM_RSTCTRL_IVA2[0] RST1_IVA2 bit. The PRM module waits for reset manager 1 in the IVA2 power domain to time out.
12. After reset manager 1 times out, the PRM module releases the IVA2_RST1 reset signal. The DSP boots.
13. The PRCM.RM_RSTST_IVA2[8] IVA2_SW_RST1 status bit is updated accordingly on release of the IVA2_RST1 reset signal.
14. DSP software enables the SEQ clock.
15. DSP software clears the PRCM.RM_RSTCTRL_IVA2[2] RST3_IVA2 bit. The PRM module waits for reset manager 3 in the IVA2 power domain to time out.
16. The PRM module releases the IVA2_RST3 reset signal. The SEQ boots.
17. The PRCM.RM_RSTST_IVA2[10] IVA2_SW_RST3 status bit is updated accordingly on release of the IVA2_RST3 reset signal.

4.5.9.5 IVA2 Global Warm Reset Sequence

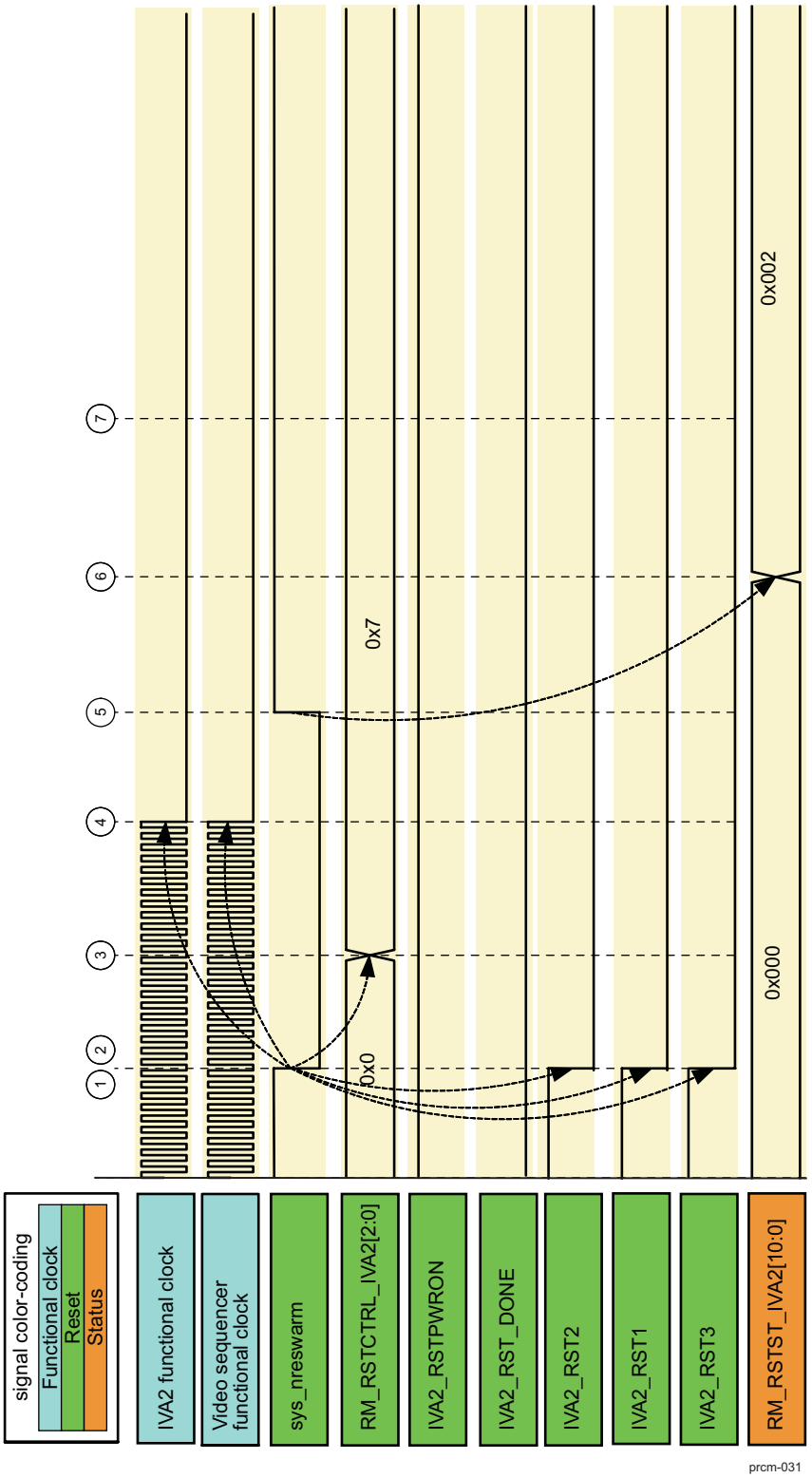
This section describes the reset sequence of the IVA2.2 subsystem and timing relationships on a global warm reset.

The assumptions are:

- The MPU is running.
- All sources of reset to the IVA2.2 subsystem are released.
- Software clears the previous reset status.

Figure 4-30 shows the IVA2 global warm reset sequence.

Figure 4-30. IVA2 Global Warm Reset Sequence



The sequence is:

1. A global warm reset source is asserted (see sys_nreswarm in Figure 4-30).
2. The PRM asserts asynchronously all IVA2 power domain warm reset signals (and all other warm reset signals).

3. The PRCM.RM_RSTCTRL_IVA2 register is reset to its default value. As a result, the IVA2 power domain warm reset signals stay asserted on release of the global warm source of reset.
4. DPLL2 is held under its reset configuration; the IVA2 power domain clocks are stopped.
5. The sys_nreswarm global warm source of reset is deasserted.
6. The PRCM.RM_RSTST_IVA2 status register is updated accordingly when the device reset manager times out.
7. The MPU boots and the MPU software sequence, shown in Section 4.5.9.4, from point 6, starts.

4.5.9.6 IVA2 Power Domain Wake-Up Cold Reset Sequence

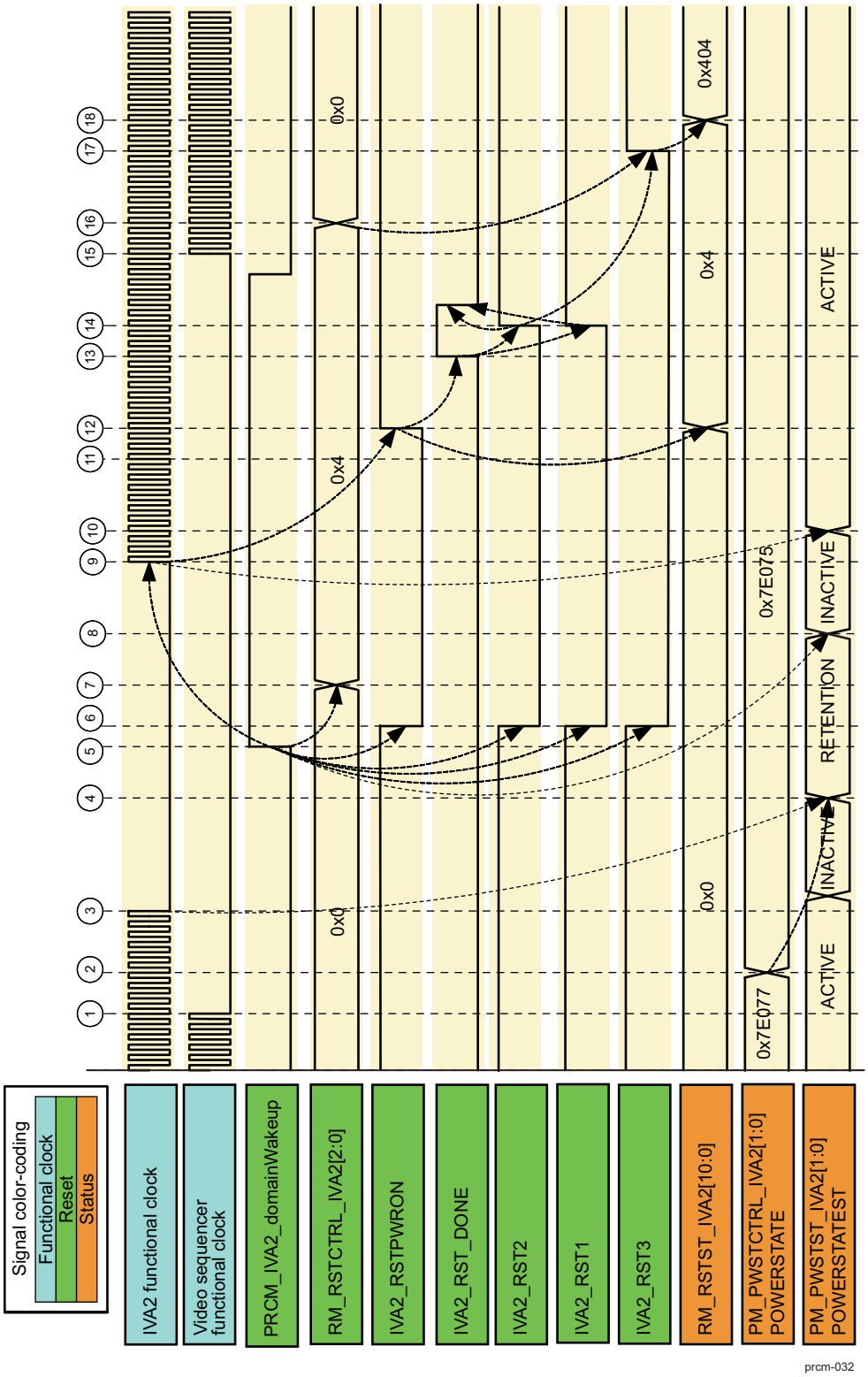
This section describes the cold reset sequence of the IVA2.2 subsystem when the IVA2 power domain transitions from retention to on power state.

The assumptions are:

- The MPU is running.
- All sources of reset to the IVA2 are released.
- The software ensures that the IVA2 power domain software sources of reset are not asserted while the IVA2 clocks are running.
- The software clears the previous reset status.

Figure 4-31 shows the IVA2 power domain wake-up cold reset sequence.

Figure 4-31. IVA2 Power Domain Power Transition Reset Sequence



The sequence is:

1. DSP software puts the SEQ in inactive state.
2. MPU software programs the IVA2 power domain to go to retention power state during the next sleep transition.

3. The DSP goes to idle mode, and the CM gates the IVA2 clock. The IVA2 power domain enters inactive power state.
4. The IVA2 power domain enters retention power state.
5. The IVA2 power domain is awakened.
6. The PRM asserts the cold reset IVA_RSTPWRON and all IVA2 warm resets asynchronously.
7. The PRCM.RM_RSTCTRL_IVA2[2] RST3_IVA2 bit is automatically reset to its reset value.
8. The IVA2 power domain is inactive.
9. IVA2 clocks are automatically reenabled. The IVA2 power domain enters active power state.
10. The PRM releases the IVA2_RSTPWRON reset signal when reset manager 2 in the IVA2 power domain times out.
11. On release of IVA2_RSTPWRON, the IVA2 performs an initialization sequence.
12. The PRCM.RM_RSTST_IVA2 register is updated accordingly on release of the IVA2_RSTPWRON reset signal.
13. The IVA2.2 subsystem asserts the IVA2_RSTDONE signal when initialization completes.
14. The PRM module releases the IVA2_RST1 and IVA2_RST2 reset signals. The DSP boots in autonomous mode.
15. DSP software enables the SEQ clock.
16. DSP software clears the PRCM.RM_RSTCTRL_IVA2[2] RST3_IVA2 bit. The PRM module waits for reset manager 3 in the IVA2 power domain to time out.
17. After reset manager 3 times out, the PRM module releases the IVA2_RST3 reset signal. The SEQ boots.
18. The PRCM.RM_RSTST_IVA2[10] IVA2_SW_RST3 bit is updated accordingly on release of the IVA2_RST3 reset signal.

There are two alternate sequences:

- The DSP is held under reset when exiting retention power state. This is done when the MPU software writes 1 to the PRCM.RM_RSTCTRL_IVA2[0] RST1_IVA2 bit. In this case, the MPU software must clear this bit to 0 to reboot the DSP.
- The DSP and MMU are held under reset when exiting retention power state (the MPU software writes 1 to the PRCM.RM_RSTCTRL_IVA2[0] RST1_IVA2 and PRCM.RM_RSTCTRL_IVA2[1] RST2_IVA2 bits). In this case, the MPU software must first clear the PRCM.RM_RSTCTRL_IVA2[1] RST2_IVA2 bit to allow the release of the reset IVA_RSTPWRON and subsequently (after initialization) the release of IVA2_RST2. Only after the MPU software reprograms the MMU or downloads the DSP code can it clear the PRCM.RM_RSTCTRL_IVA2[0] RST1_IVA2 bit field to boot the DSP.

Notes:

- Although the IVA2.2 WUGEN module and DPLL2 are part of the IVA2.2 subsystem, they are also part of the CORE power domain and the DPLL2 power domain, respectively. They are reset independently from the IVA2 power domain.
- The IVA2_RSTPWRON reset is released by software by programming the PRCM.RM_RSTCTRL_IVA2[1] RST2_IVA2 bit. Setting the PRCM.RM_RSTCTRL_IVA2[1] RST2_IVA2 bit does not assert IVA2_RSTPWRON.
- IVA2_RST3 is asserted when the IVA2 power domain transitions from off or retention state to on state. The corresponding bit in the PRCM.RM_RSTCTRL_IVA2 register is automatically set to 1 during this transition.
- The release of the IVA2_RST3 reset is stalled as long as the IVA2_RST2 reset is not released.
- The release of the IVA2_RST2 and IVA2_RST1 resets causes IVA2 to deassert the IVA2_RSTDONE signal.

4.6 PRCM Power Manager Functional Description

4.6.1 Overview

4.6.1.1 Introduction

To efficiently manage leakage and reduce dynamic power consumption, the device supports several power-management techniques:

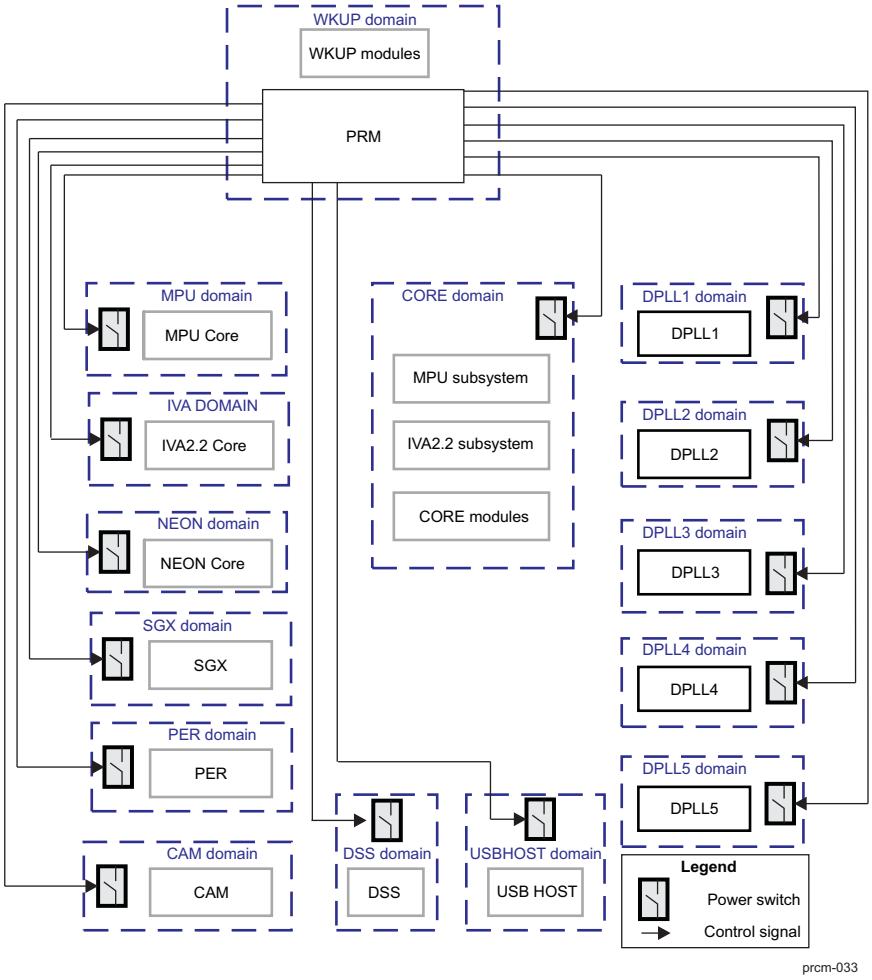
- Dynamic clock-gating conditions
- DVFS
- DPS
- SLM

These power-management techniques require software control and specific hardware features. The hardware features are fully controlled by the PRM part of the PRCM module. These features are:

- Device partitioning into power domains
- Device partitioning into several voltage domains communicating through level shifters
- Logic power-switch control
- RFF control
- Memory power-switch control
- Embedded LDOs (SRAMs, wakeup, emulation) control
- I/O off-mode control
- Level shifter control
- External power IC control

Figure 4-32 is an overview of PRM power management in the device.

Figure 4-32. Power Control Overview



4.6.1.2 Device Partitioning

To substantially reduce leakage in sleep modes (SLM strategy) and to optimize active power-consumption savings (DPS strategy), the device is segmented into power domains:

- MPU: Application processor
- NEON: Multimedia coprocessor
- IVA2: DSP
- SGX: Graphics engine
- CORE: Interconnect, memory controllers, peripherals, and clock management
- DSS: Display subsystem
- CAM: Camera controller
- PER: Low-power use case peripherals
- WKUP: Wakeup (always active)
- USBHOST: USB HOST power domain
- EMU: Emulation
- EFUSE: eFuse farm
- DPLL1: MPU DPLL
- DPLL2: IVA2.2 DPLL
- DPLL3: CORE DPLL
- DPLL4: Peripherals DPLL

- DPLL5: Peripherals DPLL2

Each power domain receives power through an independent switch controlled by the PRM module. In this way, depending on the application scenario, unused power domains can be switched off or put in retention state while others remain active.

Table 4-24 lists the device modules in relation to the power domains.

Table 4-24. Power Domain Modules

Power Domain	Modules
MPU	MPU core
	ICECrusher CS
	MPU async bridge (master)
NEON	NEON coprocessor
IVA2	IVA2.2 DSP
	IVA2 async bridge 1 (master)
	IVA2 async bridge 2 (slave)
	Video sequencer (SEQ)
SGX	SGX subsystem
CORE	GPMC
	USB TLL
	GPTIMER[10, 11]
	HDQ/1-Wire
	HS USB
	I2C[1, 2, 3]
	ICR
	IVA2.2 async bridge 1 (slave)
	IVA2.2 async bridge 2 (master)
	IVA2.2 WUGEN
	MAILBOXES
	McBSP[1, 5]
	McSPI[1, 2, 3, 4]
	MMC/SD/SDIO[1, 2, 3]
	MPU async bridge (slave)
	MPU INTC
	OCM_RAM
	OCM_ROM
	SCM
	SDRC
	SMS
	L3 interconnect
	UART[1, 2]
	SDMA
	Temperature sensor (x2)]
	L4-Core interconnect
DSS	Display subsystem
	Video DAC
CAM	Camera subsystem
PER	UART3
	WDT3
	McBSP[2..4]

Table 4-24. Power Domain Modules (continued)

Power Domain	Modules
	GPIO[2..6]
	GPTIMER[2..9]
	L4-Per interconnect
WKUP	GPIO1
	GPTIMER1
	WDT2
	L4-Wake-up interconnect
USBHOST	HS USB host subsystem
EMU	CWT
	DAP-APB
	ETB
	ICEPICK
	TRACEPORT
	L4-EMU interconnect
EFUSE	eFuse farm
DPLL1	MPU DPLL
DPLL2	IVA2.2 DPLL
DPLL3	CORE DPLL
DPLL4	Peripherals DPLL
DPLL5	Peripherals DPLL2

4.6.1.3 Memory and Logic Power Management

Device power domains can have separate control for logic and memory. This depends on the power domain implementation in the device (for details, see Section 4.6.2, *Power Domain Implementation*).

Generally, separate switches allow separate management of the logic and memory power states in a power domain.

Memory banks in the MPU (L2 cache), IVA2 (L1, L1 flat, L2, L2 flat), CORE (CORE memory banks 1 to 6), SGX, CAM, DSS, USBHOST, and EMU power domain memories have their own voltage and power control, independent of the logic. However, the user must ensure that memory states are programmed consistently with the logic state.

MPU L1 cache memory does not have an independent control and is supplied with the MPU logic.

CORE memory banks 3,4, 5, and 6 are associated to the domain power state and controlled with the domain logic.

MPU and IVA2 memories are supplied by a common LDO (VDD4). CORE, SGX, CAM, DSS, USBHOST, and EMU memories are supplied by another LDO (VDD5).

4.6.1.4 Power Domain States

Each power domain can be driven by the PRM to four different states, depending on the functional mode required by the user. Power domain states are hardware-defined and depend on the domain clock activity and domain memory/logic switch states.

Table 4-25 defines the power states a power domain can reach in the device.

Table 4-25. Power Domain States

Power State	Power		Clocks
	Logic	Memory	
Active	On	On, retention, or off	On (at least one)
Inactive	On	On, retention, or off	Off (all clocks)
Retention (CSWR)	On	Retention or off	Off (all clocks)
Retention (OSWR)	Off/RFF		
Off	Off	Off	Off (all clocks)

Notes:

- Closed-switch retention (CSWR): Full domain logic supply is maintained, and supply voltage can be dropped to minimize leakage.
- Open-switch retention (OSWR): Full domain logic is switched off, but the context is saved for modules that embed RFFs.
- Throughout the rest of this chapter, when reference is made only to retention state without explicitly stating whether it is CSWR or OSWR, then the description applies to the two retention states.

In CSWR and OSWR, memories are put in retention or can also be switched off.

4.6.1.5 Power State Transitions

For each power domain, the PRM manages all transitions controlling domain clocks, domain resets, domain logic power switches, memory power switches, and memory retention.

As previously mentioned, a power domain is characterized by different power states: Active, inactive, retention, and off.

A transition is a passage from one state to another.

Two types of power state transitions are possible:

- Sleep: Moving from a higher consumption power state (active) to a lower consumption power state (inactive, retention, or off)
- Wake-up: Moving directly from a lower consumption power state (inactive, retention, or off) to the active power state

For instance, a power domain with initiators and target modules in standby and idle modes operates an active-to-inactive transition. It is then ready to operate an inactive-to-retention or off transition, depending on the requirements.

The device supports eight transitions:

- Active to inactive
- Inactive to active
- Active to retention
- Retention to active
- Active to off
- Off to active

4.6.1.6 Device Power Modes

A device power mode is a specific functional combination of the power states of all the power domains of the device.

Unlike power domain states (see Section 4.6.1.4), device power modes are not hardware-defined; they are defined by software as relevant combinations of the power domain states.

There are two types of device power modes:

- Active: Any valid combination of power domain states in which one or several domains are still active, regardless of whether any software is running
- Standby: Any valid combination of power domain states in which all the domains are in inactive, retention, or off state

Note: All power domain states are hardware-supported, thereby preventing electrical damage to the device. However, software ensures that the power domain states are set correctly, to result in a valid device power mode and ensure correct functioning.

4.6.1.7 Isolation Between Power Domains

When switching a power domain from one state to another, the PRCM module automatically manages domain output isolation to prevent electrical damage.

This mechanism is hardware-supervised and does not require software action.

4.6.2 Power Domain Implementation

4.6.2.1 Device Power Domains

Table 4-26 summarizes the power mode implementation in the device and the control granularity for each power and memory domain.

Table 4-26. Domain Power Control Summary

Functional Domain or Subsystem	Power Domain	PRCM Power Control Type
MPU	MPU	Logic on-off
	NEON	Logic on-off
	CORE	Logic on-off
	EMU	Logic on-off
	L1 cache SRAM	Logic on-off
	L2 cache SRAM	On-off
		BB retention
IVA2.2	IVA2	Logic on-off
		Memory tags retention
	CORE	Logic on-off
		Logic retention
	L1 cache SRAM	On-off
		BB retention
	L1 flat cache SRAM	On-off
		BB retention
	L2 cache SRAM	On-off
		BB retention
	L2 flat cache SRAM	On-off
		BB retention
	Misc SRAM	Logic on-off
All MPU and IVA2.2 memories		LDO retention
GRAPHICS	SGX	Logic on-off
	Misc SRAM	Logic on-off
DISPLAY	DSS	Logic on-off
	Misc SRAM	Logic on-off
CAMERA	CAM	Logic on-off

Table 4-26. Domain Power Control Summary (continued)

Functional Domain or Subsystem	Power Domain	PRCM Power Control Type
	Misc SRAM	Logic on-off
USBHOST	USB	Logic on-off
	Misc SRAM	Logic on-off
CORE (interconnect, memory controllers, peripherals)	CORE	Logic on-off Logic retention
	SRAM bank 1	On-off BB retention
	SRAM bank 2	On-off BB retention
	OSWR SRAM 3	Logic on-off BB retention
	CSWR SRAM bank 4	On-off BB retention
PERIPHERALS	PER	Logic on-off Logic retention
WKUP	WKUP	none
EMULATION	EMU	Logic on-off
	ETB SRAM	On-off BB retention
EFUSE	eFuse farm	none
MPU DPLL	DPLL1	Logic on-off
IVA2.2 DPLL	DPLL2	Logic on-off
CORE DPLL	DPLL3	Logic on-off
Peripherals DPLL	DPLL4	Logic on-off
Peripherals DPLL2	DPLL5	Logic on-off

Notes:

- BB: Back-bias or local memory retention
- OSWR: Open-switch retention
- CSWR: Closed-switch retention
- LDO retention: All memories are in BB mode and the memory array is dropped down.
- Logic on-off: Memory is controlled with the logic (and does not support retention).
- OSWR SRAM bank 3: Corresponds to memories that must be retained when the CORE OSWR mode is activated (the CORE context is maintained only by keeping RFFs alive). These memories are scratchpad, SDMA, and SMS.
- CSWR SRAM bank 4: Corresponds to memories that are retained only in case of CSWR (all logic is maintained with reduced voltage). These memories are USB and MAILBOXES.

The CORE power domain has six memory blocks. Of these six, the power state of only two memory blocks, SRAM bank 1 and SRAM bank 2, can be configured vis-a-vis the domain power state through software setting of the corresponding bit fields in the PRCM.PM_PWSTCTRL_CORE register. The power states of the rest of the memory banks are automatically controlled by the PRCM module (hardware-controlled) and no software setting is possible.

Memory bank 5 of the CORE power domain is used for the USB TLL module save and restore operations. (see Section 4.8.6). Similarly, the memory bank 6 is for the system control module save and restore feature (see Chapter 7, *System Control Module*).

Note: Before initiating a domain sleep transition (ON to OFF) on the MPU and CORE power domains, the CORE memory banks must be configured to automatically switch to ON power state when the CORE power domain wakes up (OFF to ON). This is required for correct booting.

This is done by setting the PM_PWSTCTRL_CORE[19:18] MEM2ONSTATE bit field to 0x3 and the PM_PWSTCTRL_CORE[17:16] MEM1ONSTATE bit field to 0x3.

4.6.2.2 Power Domain Memory Status

The PRCM module logs the memory status in the SCM.CONTROL_SEC_STATUS register. The memory status is logged for CORE power domain memory banks 1 and 2 (OCM RAM) and for the L1 and L2 caches of the MPU.

Two statuses are logged for each memory bank:

- Memory is destroyed: The memory array is powered down.
- Memory is not accessible: The memory array is powered down or is in retention.

For information about the SCM, see Chapter 7, *System Control Module*.

4.6.2.3 Power Domain State Transition Rules

As previously described, a power domain can reach different states. These states are linked to the power applied to the domain and to the domain clock activity.

Any power state transition always depends on and starts with power domain clock activity control. Any active clock in a power domain sets the domain as active (see Section 4.6.2.1). Therefore, a sleep transition always starts by shutting down the power domain clocks. For more information about power domain clock controls, see Section 4.7, *Clock Manager Functional Description*.

When all clocks are shut down in a power domain, transitions from inactive-to-off or inactive-to-retention can occur.

Similarly, when a wake-up transition occurs between a lower-power domain state and a higher one, the transition to active state becomes effective when the required clocks are restarted.

Section 4.8, *Idle and Wake-Up Management*, describes sleep and wake-up transitions.

4.6.2.4 Power Domain Dependencies

Besides the inner conditions to operate a state transition on a single power domain, the device offers hardwired and software-programmable dependencies between the power domains.

Two kinds of dependencies exist:

- Sleep dependencies: Used to start a sleep transition on a power domain only if the related power domains are in mute mode (not requesting any service from the linked power domain)
- Wake-up dependencies: Used to initiate a wake-up transition on a power domain as a consequence of a linked power domain wake-up transition

Two dedicated sets of registers (PRCM.CM_SLEEPDEP_<domain> and PRCM.PM_WKDEP_<domain>) allow the setting of programmable dependencies.

Section 4.8.5, *Sleep and Wake-Up Dependencies*, summarizes the possible dependency combinations between power domains.

4.6.2.5 Power Domain Controls

4.6.2.5.1 Power Domain Hardware Control

Device power domain implementation depends on hardwired dependencies, both sleep and wake-up transitions, between power domains. This means a single power domain transitions from one power state to another depending on the states and transitions of the linked domains.

For details about the dependencies between power domains, see Section 4.8, *Idle and Wake-Up Management*, and Section 4.12, *Basic Programming Model*.

4.6.2.5.2 Power Domain Software Controls

If all conditions are met to initiate a power domain state transition (all the modules are idle/standby and the related clocks are shut down), the PRCM module automatically manages the transition according to the following settings:

- Dependencies setting: The PRCM.CM_SLEEPDEP_<domain> registers and PRCM.PM_WKDEP_<domain> registers are used to set/clear programmable dependencies between power domains.

  Not all dependencies are programmable by software; some are hardwired and thus do not allow any control. For more information about the dependencies between power domains, see Section 4.8, *Idle and Wake-Up Management*, and Section 4.12, *Basic Programming Model*.
- Power state transitions setting: The transitions of each power domain state can be controlled by software through a set of dedicated status registers (PRCM.PM_PWSTCTRL_<domain_name>) that allow the configuration of the power state that the domain enters after completing a transition (PRCM.PM_PWSTCTRL_<domain_name>[1:0] POWERSTATE bit field). These status registers are used to check the current state of the logic and memories in a power domain and to learn about any ongoing state transition.

  Also, if the power domain offers different settings for logic and memory, the PRCM.PM_PWSTCTRL_<domain_name> registers allow selection of the state to which the logic and memory go when the power domain is put in retention. They also allow selection of the state to which the memory will go when the power domain is put into on state.

  These features depend on each power domain implementation. For details, see Section 4.12, *Basic Programming Model*, and Section 4.14, *PRCM Registers Manual*.
- Event generator: Three registers (PRCM.PM_EVGENCTRL_MPU, PRCM.PM_EVGENONTIM_MPU, and PRCM.PM_EVGENOFFTIM_MPU) let the MPU power domain be switched between on and off (or placed in inactive mode). The PRCM.PM_EVGENONTIM_MPU and PRCM.PM_EVGENOFFTIM_MPU registers set the durations of the on and off modes, respectively.

  For details, see Section 4.12, *Basic Programming Model*, and Section 4.14,*PRCM Registers Manual*.

4.7 PRCM Clock Manager Functional Description

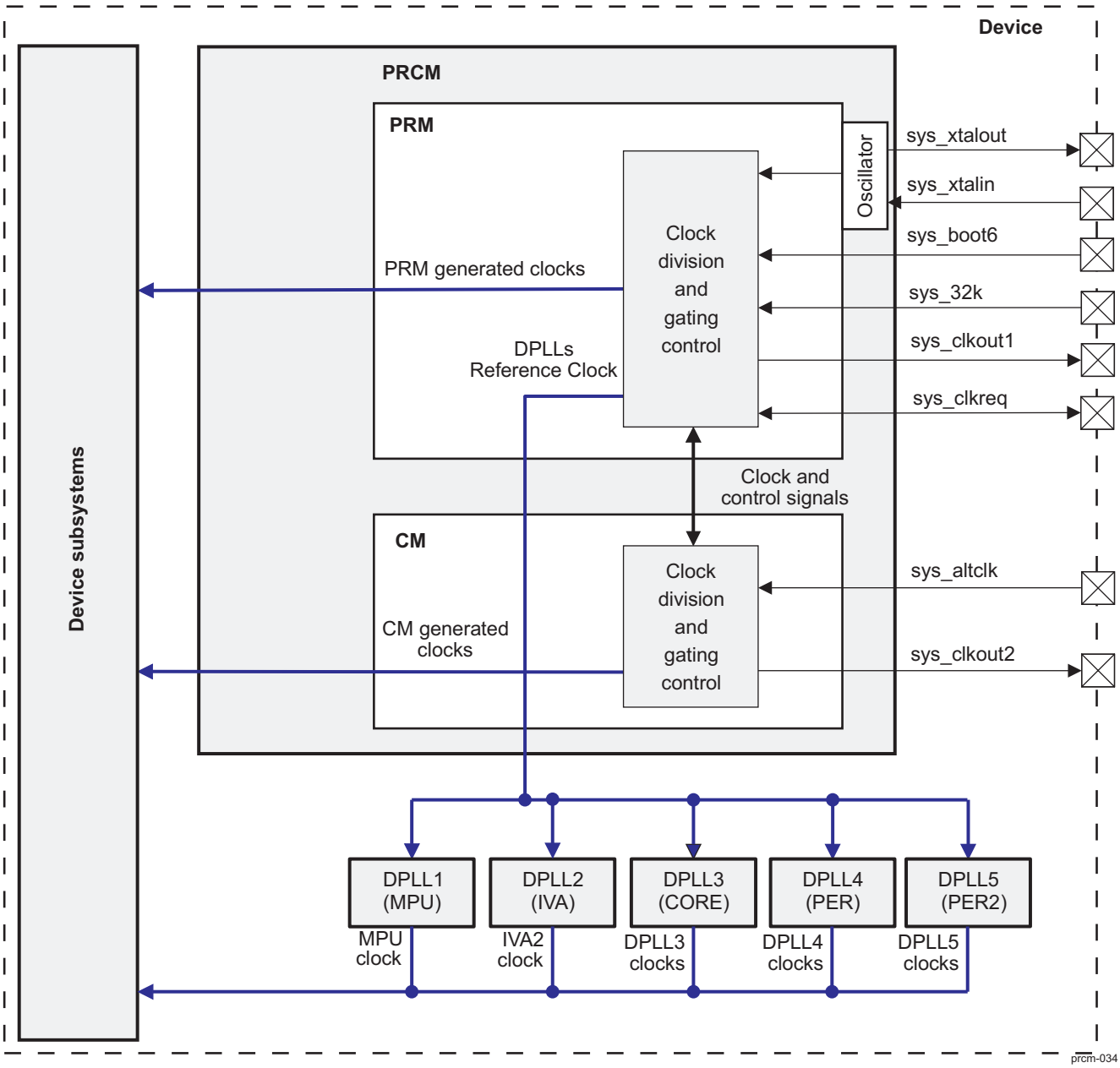
This section describes all modules and features in the high-tier device. To check availability of modules and features, see Section 1.5, *OMAP34xx Family*. To save power, ensure that power domains of unavailable features and modules are switched off and clocks are cut off.

4.7.1 Overview

The PRCM module provides control for clock generation, division, distribution, synchronization, and gating. It distributes the clock sources to all modules in the device.

The device-level clock generation is handled by an internal oscillator (the system clock oscillator) and DPLLs; clock division and gating are handled by the PRM and the CM sections of the PRCM module. Figure 4-33 shows the high-level clock-management scheme in the device.

Figure 4-33. PRCM Clock Manager Overview



4.7.1.1 Interface and Functional Clocks

The PRCM module propagates two kinds of clocks:

- Interface clock: Ensures proper communication between any module and the system interconnects (level 3 [L3] or level 4 [L4]). In most cases, the interface clock supplies the interface and registers of the module. For some modules, the interface clock is also used as the functional clock.
- Functional clock: Supplies the functional part of a module or subsystem. In some cases, a module or subsystem can require several functional clocks.

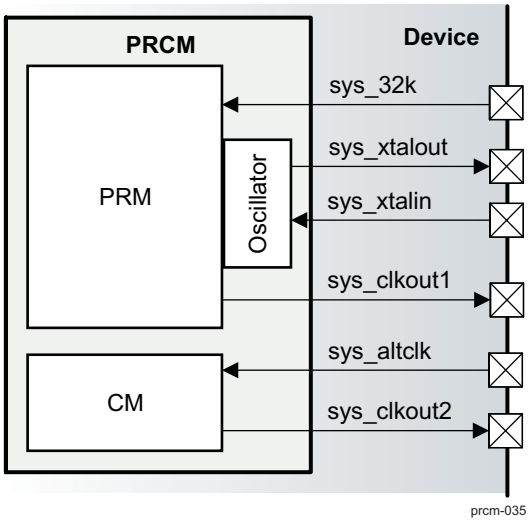
To be operational, a module requires a functional clock(s); to communicate with other modules, it requires an interface clock. For example, the functional clock of a general-purpose timer (GPTIMER) must be active for it to run, but its interface clock can be turned off.

A module can use one or more optional functional clocks. Because an optional functional clock is used only for specific features of the module, it can be shut down without stopping the module activity (for example, 54 MHz for the DSS power domain and 96 MHz for the CAM power domain).

4.7.2 External Clock I/Os

Figure 4-34 shows the external clock I/Os of the device.

Figure 4-34. External Clock I/O



4.7.2.1 External Clock Inputs

4.7.2.1.1 32-kHz Always-On Clock

The sys_32k input pin supplies the 32-kHz always-on clock (32K_FCLK clock). This clock is used for low-frequency operation (timers, debouncing, etc.). It also supplies the WKUP power domain for operation in the lowest power mode.

4.7.2.1.2 High-Frequency System Clock

The high-frequency system clock (SYS_CLK) is supplied to the device from an external clock source through the sys_xtalin input pin or is generated internally by a local system clock crystal oscillator. In the latter case, a crystal is connected between the sys_xtalout and sys_xtalin device pins. The sys_boot[6] pin sets the oscillator operating mode (see Figure 4-12).

The source system clock can be 12, 13, 16.8, 19.2, 26, or 38.4 MHz and can be divided by 2 to provide the standard system clock frequencies (using internal system clock divider configuration in the PRCM.PRM_CLKSRC_CTRL[7:6] SYSCLKDIV bit field). It supplies the reference to the DPLLs and is also used by several modules. The system clock is activated in the device after the device power-on reset is released by the reset manager in the PRCM module.

The source-clock selection register (the PRCM.PRM_CLKSEL [2:0] SYS_CLKIN_SEL bit field) is set by the software to identify the input frequency of the system clock.

Table 4-27 lists the system clock input configurations.

Table 4-27. System Clock Input Configurations

Input Source	Device Pin Mapping	Description
Quartz	sys_xtalin and sys_xtalout	Internal oscillator is enabled.
Square clock (1.8-V CMOS signal)	sys_xtalin (sys_xtalout is not connected)	Internal oscillator is bypassed.

Note: An external pullup or pulldown tied on the sys_boot6 input pin of the device determines whether the internal oscillator is used (oscillator mode) or an external clock source is supplied to the sys_xtalin input pin (bypass mode).

4.7.2.1.3 Alternate Clock

The sys_altclk pin can provide an alternate 54-MHz, 48-MHz, or any other frequency clock.

4.7.2.2 External Clock Outputs

The device can output two clocks externally:

- sys_clkout1 can output an oscillator clock (12, 13, 16.8, 19.2, 26, or 38.4 MHz) at any time. The output oscillator clock can be controlled by software or externally using sys_clkreq control. When the device is in off state, sys_clkreq can be asserted to enable the oscillator and activate sys_clkout1 without waking up the device. The off state polarity of sys_clkout1 is programmable.
- sys_clkout2 can output CM_SYS_CLK (12, 13, 16.8, 19.2, 26, or 38.4 MHz), CORE_CLK (CORE DPLL output), or 96-MHz or 54-MHz clocks. It can be divided by 2, 4, 8, or 16, and its off state polarity is programmable. This output is active only when the CORE power domain is active. Also, the selected source clock must be enabled by software. Enabling sys_clkout2 does not automatically request the required source clock.

4.7.2.3 Summary

Table 4-28 summarizes the external clock I/O.

Table 4-28. External Clock I/Os

Name	I/O[1]	Source/Destination	Description
sys_xtalin	I	Oscillator	Main input clock. Crystal oscillator clock (only at 12, 13, 16.8, or 19.2 MHz) or CMOS digital clock (at 12, 13, 16.8, 19.2, 26, or 38.4 MHz).
sys_xtalout	O	Oscillator	Output of oscillator
sys_clkout1	O	PRCM	Configurable output clock 1
sys_clkout2	O	PRCM	Configurable output clock 2
sys_32k	I	PRCM	32-kHz clock input
sys_altclk	I	PRCM	Alternate selectable clock source for UARTs or NTSC/PAL. It can be 54 MHz, 48 MHz, or any frequency.

(1) I = Input, O = Output

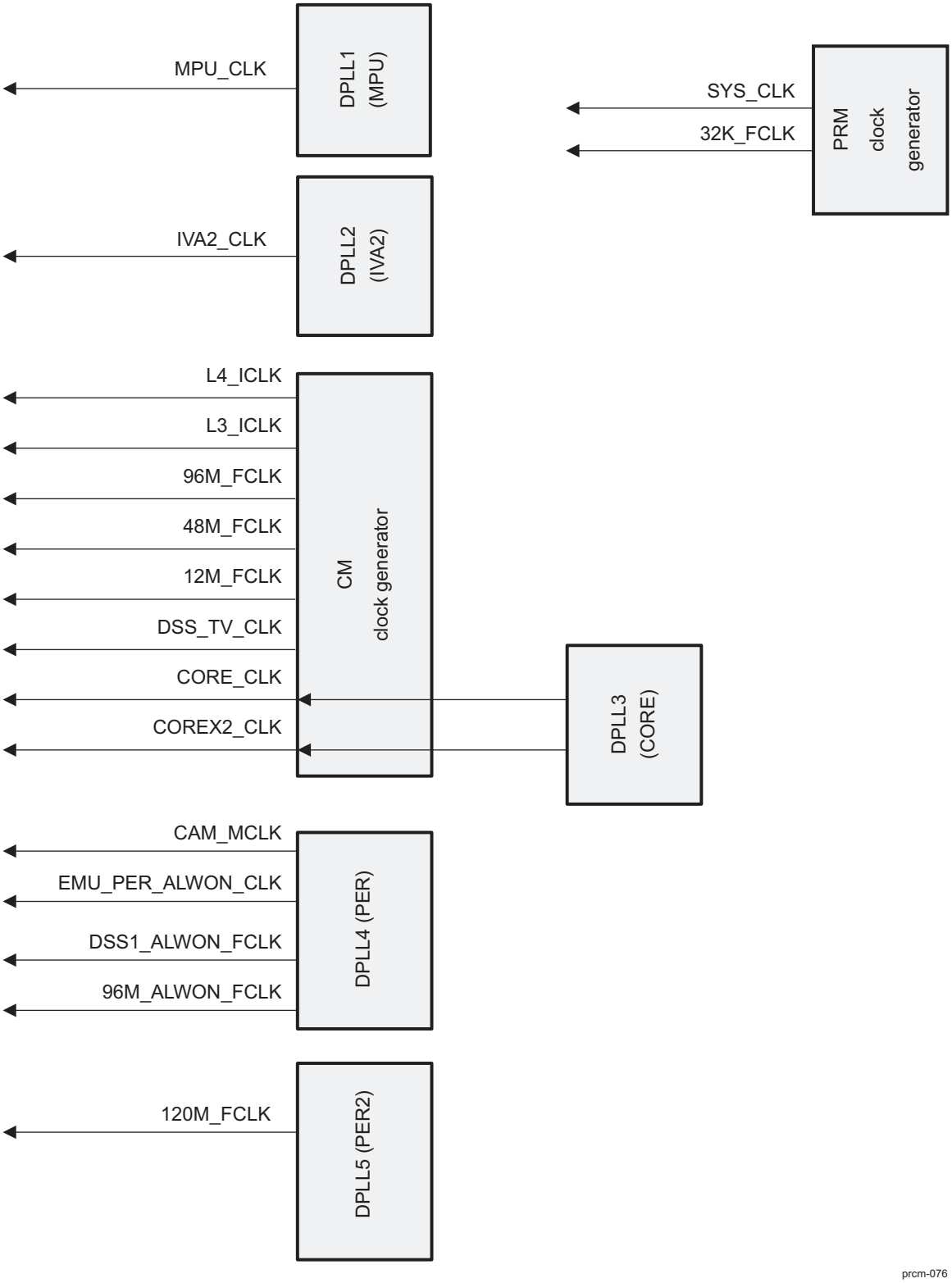
4.7.3 Internal Clock Generation

The device generates internal clocks from following sources:

- PRM
- CM
- DPLLs

Figure 4-35 shows the internal clock-generation scheme of the device.

Figure 4-35. Internal Clock Sources



4.7.3.1 PRM

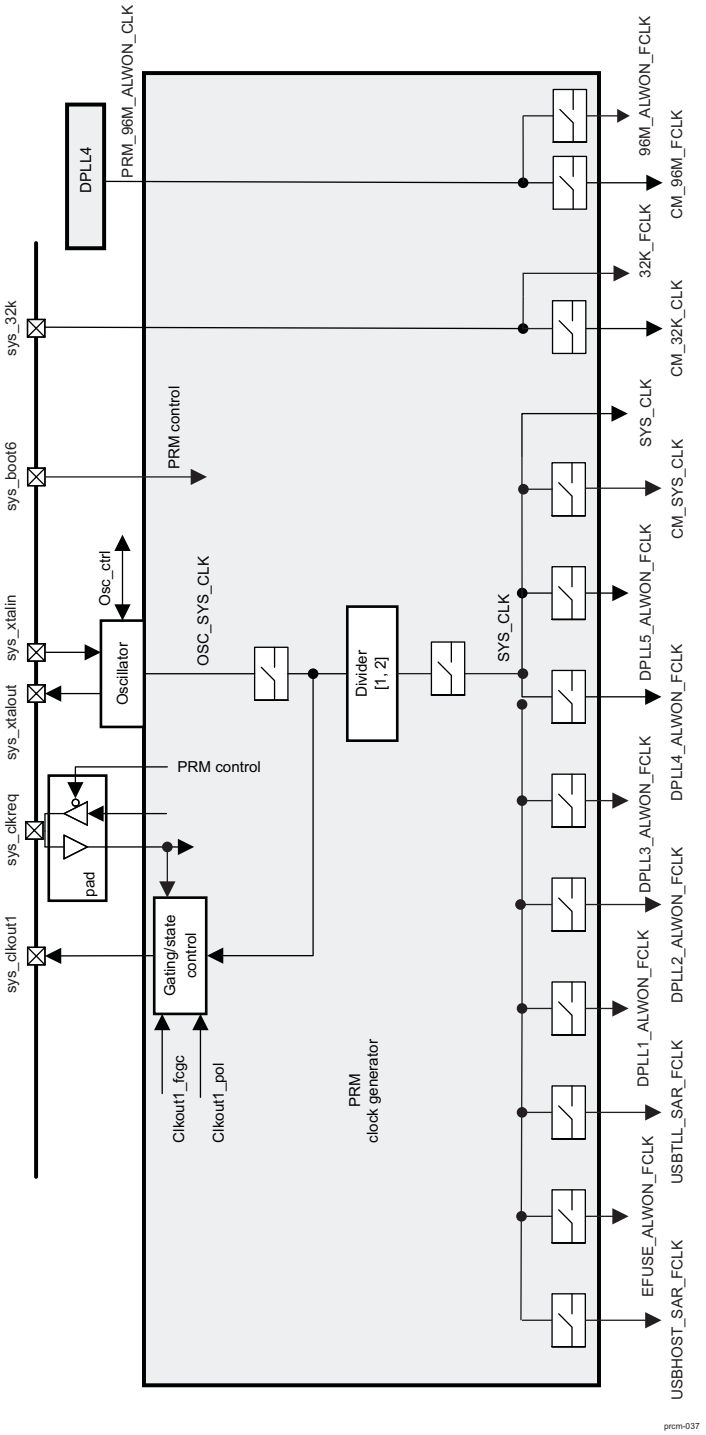
The PRM resides in the WKUP power domain. It handles the generation of the 32-kHz low-frequency clocks and the high-frequency system clocks from the SYS_CLK. It also manages the clock oscillator and the external clock output sys_clkout1.

SYS_CLK is generated by the internal oscillator or supplied as the external clock signal on the sys_xtalin pin. It supplies most of the clocks in the device. Some of the device clocks sourced by SYS_CLK are always powered (clocks are present even when the CORE power domain is in off state). SYS_CLK is also the source of the WKUP power domain interface clocks.

It also handles the gating and distribution of the 96-MHz clock from DPLL4 to the CM and PER power domain modules.

Figure 4-36 is a functional overview of the PRM. The other clocks in the figure are explained in the following sections.

Figure 4-36. PRM Clock Generator



4.7.3.2 CM

The CM clock generator generates interface clocks and peripheral functional clocks for most of the modules. It also controls DPLL3, DPLL4, and the external peripheral clock output sys_clkout2 (see Figure 4-37).

The CM is in the CORE power domain, which can be powered off. When the CORE power domain is powered off, the clocks are not available, and their off state is latched. The DPLL controls are also latched. The RFF architecture ensures that the full CM setting is saved when the CORE power domain goes to retention state, and is transparently restored when it reactivates.

DPLL3 receives SYS_CLK from the PRM module and generates CORE_CLK through the CM. CORE_CLK is the source for the interface clocks (L3 and L4) and the functional clock . The L3 and L4 interface clocks supply the device interconnects and all module interface clocks. The L4 clock is divided for USB (full-speed clock limitation at 50 MHz) and to supply the reset managers (PRM) in the WKUP power domain. The clocks derived from CORE_CLK are fully balanced over the device.

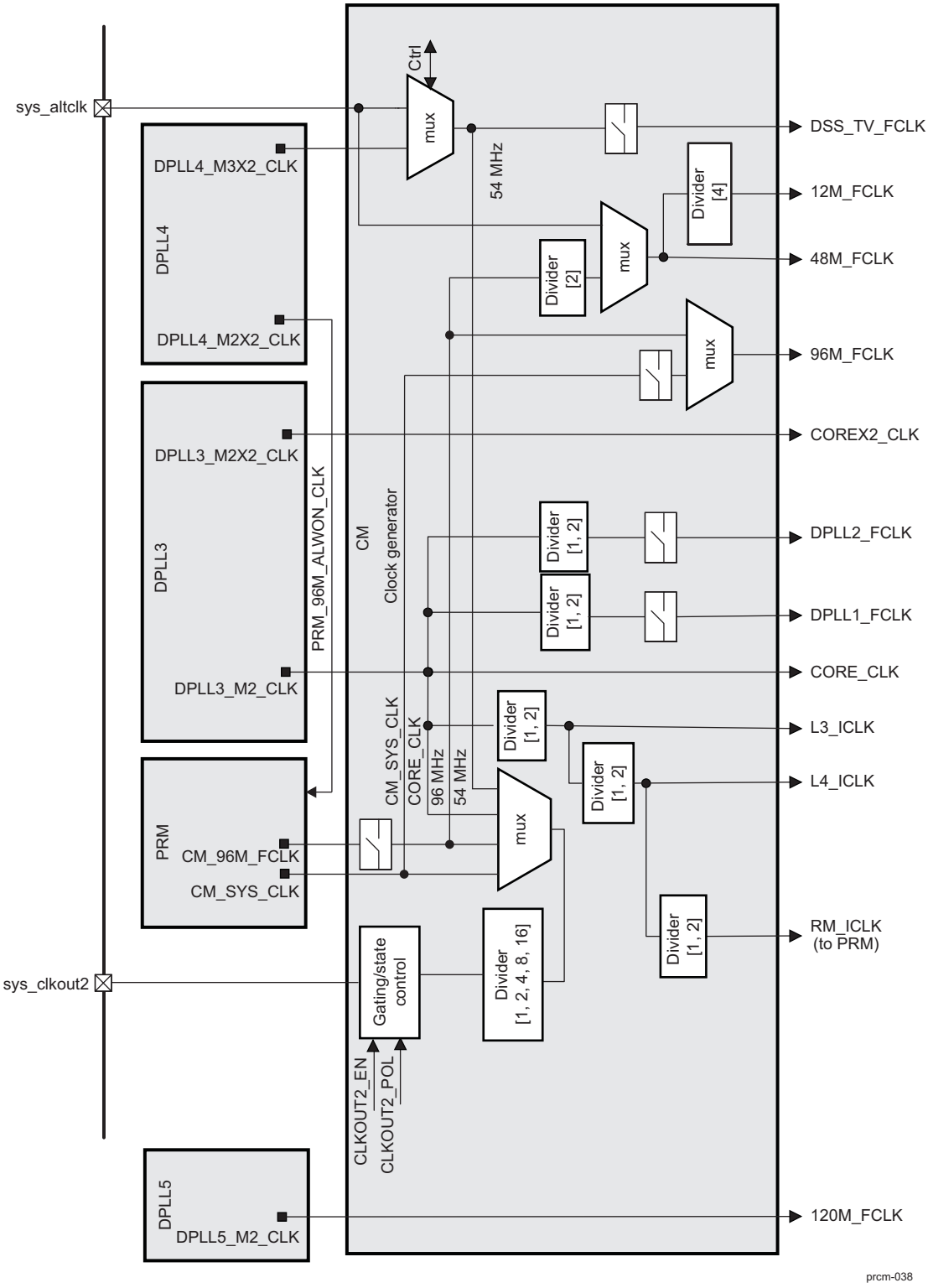
The 96M_FCLK, 48M_FCLK, and 12M_FCLK clocks are functional unbalanced clocks for a number of modules in the CORE and PER power domains.

The functional 96-MHz clock path can be bypassed with SYS_CLK to allow a peripheral such as I^2C to function while the DPLL4 is not powered on. The default configuration after initial power on is bypassed with the system clock. Software must switch to the DPLL-generated clock after programming the system settings.

The 96-MHz clock input from the PRM to the CM (CM_96M_FCLK) is internally gated by the CM.

Figure 4-37 is the functional overview of the CM clock generator.

Figure 4-37. CM Clock Generator Functional Overview



4.7.3.3 DPLLs

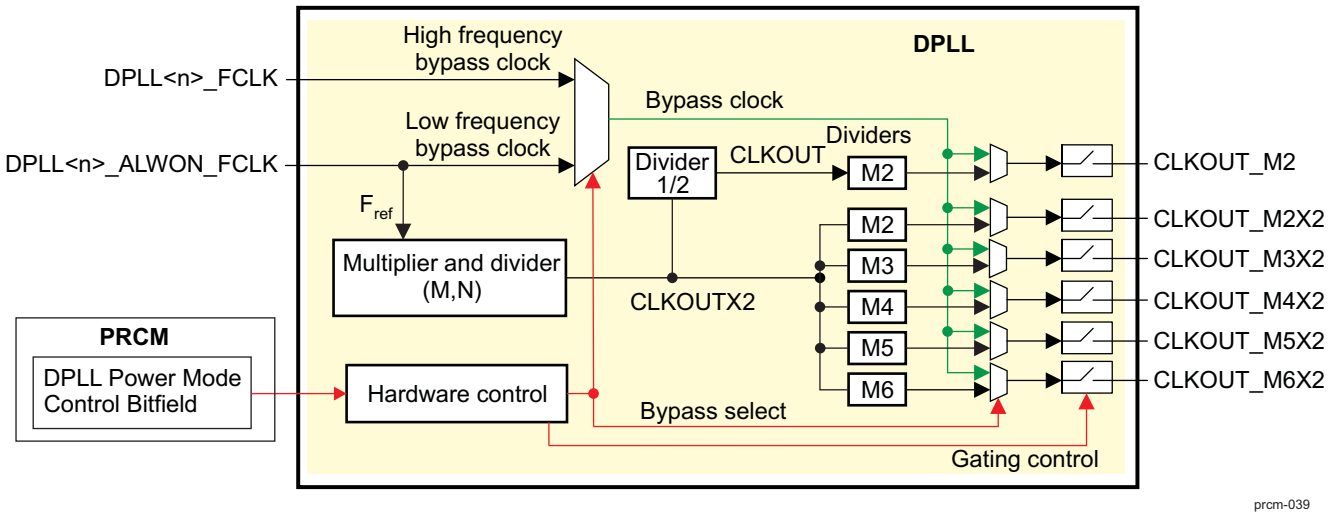
To generate high-frequency clocks, the device supports five on-chip DPLLs controlled directly by the PRCM module:

- DPLL1 (MPU)
- DPLL2 (IVA2)
- DPLL3 (CORE)
- DPLL4 (PER)
- DPLL5 (PER2)

Note: This chapter discusses only DPLL1 to DPLL5, because they are directly controlled by the PRCM module. Chapter 15, *Display Subsystem*, discusses the DPLLs in the display subsystem.

Figure 4-38 shows the functional architecture of a generic DPLL.

Figure 4-38. Generic DPLL Functional Diagram



Depending on its hardware configuration, the DPLL can receive one or two clock inputs.

When the DPLL has two clock inputs, it uses one as the reference clock (F_{ref}) to generate the high-frequency clock; the second one serves as the bypass clock when the DPLL is in bypass mode (not locked and generating the high-frequency clock). For example, DPLL1 and DPLL2 receive the high-frequency bypass clock from the DPLL3 output, and the reference clock from the PRM module.

When the DPLL has only one clock input, it uses that clock input as the reference clock and bypass clock. For example, DPLL3, DPLL4, and DPLL5 receive only one input clock from PRM module, and it is used as the reference and the bypass clock.

It internally generates two main clocks according to the following equations:

- CLKOUTX2 = (F_{ref} x 2 x M)/(N+1)
- CLKOUT = CLKOUTX2/2

where M is an 11-bit multiplier and N is a 7-bit divider.

Note: When M is set to 0 or 1, the DPLL is forced to bypass mode.

The internal clocks (CLKOUT and CLKOUTX2) of the DPLL can then be used to generate six independent output clocks:

- CLKOUT_M2 = CLKOUT/M2
- CLKOUT_M2X2 = CLKOUTX2/M2

- CLKOUT_M3X2 = CLKOUTX2/M3
- CLKOUT_M4X2 = CLKOUTX2/M4
- CLKOUT_M5X2 = CLKOUTX2/M5
- CLKOUT_M6X2 = CLKOUTX2/M6

where M2, M3, M4, M5, and M6 are additional dividers for the DPLL-synthesized clock.

The output clock frequencies defined by these equations are generated by the DPLL only when it is locked. When the DPLL is in bypass mode, however, all clock outputs run at the bypass clock frequency. The bypass clock can be a high-frequency bypass clock (only for DPLL1 and DPLL2) or the low-frequency reference clock.

The DPLL also provides an independent clock-gating signal for each of the six output clocks. The PRCM module provides the DPLL with a clock-gating control signal, and the DPLL returns a clock activity status signal indicating whether the output clock is effectively gated or running.

For an explanation of the DPLL multiplier, divider settings, and gating controls, see Section 4.7.6, *DPLL Control*.

Each clock-generating DPLL of the device has the following features:

- Independent power domain
- Control by the CM
- Fed by always-on SYS_CLK with independent gating control for the SYS_CLK
- Analog part supplied by a dedicated power supply (VDDPLL and VDDADAC at 1.8 V) and an embedded LDO to eliminate 1-MHz noise
- Up to six independent output dividers for simultaneous generation of multiple output clocks with different frequencies

4.7.3.3.1 DPLL1 (MPU) and DPLL2 (IVA2)

DPLL1 and DPLL2 are in the MPU and IVA2.2 subsystems, respectively. The DPLLs supply the source clocks for their respective subsystems, which use the source clocks to internally generate all subsystem clocks.

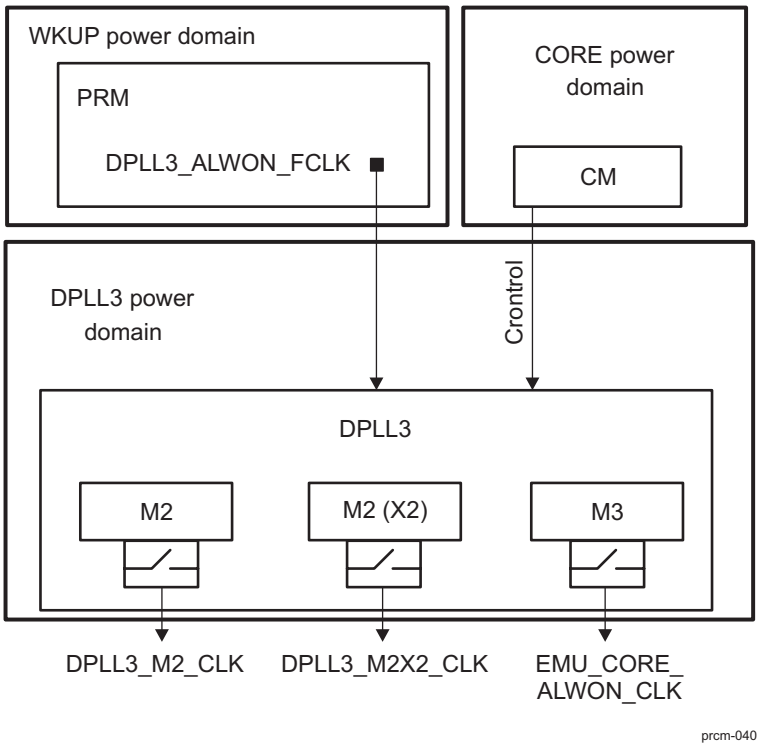
DPLL1 and DPLL2 each use a reference clock (DPLL1_ALWON_FCLK and DPLL2_ALWON_FCLK, respectively) to produce their synthesized clocks. They receive these reference clocks from the PRM module, and their high-frequency bypass clocks (DPLL1_FCLK and DPLL2_FCLK, respectively) from the CM. The reference clock is SYS_CLK, and the high-frequency bypass clock is CORE_CLK.

To save on DPLL power consumption by the processors, the high-frequency bypass input clock from DPLL3 (CORE) is used when the DPLLs are set to bypass mode (statically, or dynamically during relock time) or when the processors are not required to run faster than at L3 clock speed. This use of the high-frequency bypass input clock also optimizes the performance of the DPLLs during frequency scaling.

4.7.3.3.2 DPLL3 (CORE)

Figure 4-39 is a block diagram of DPLL3.

Figure 4-39. DPLL3 Clocks

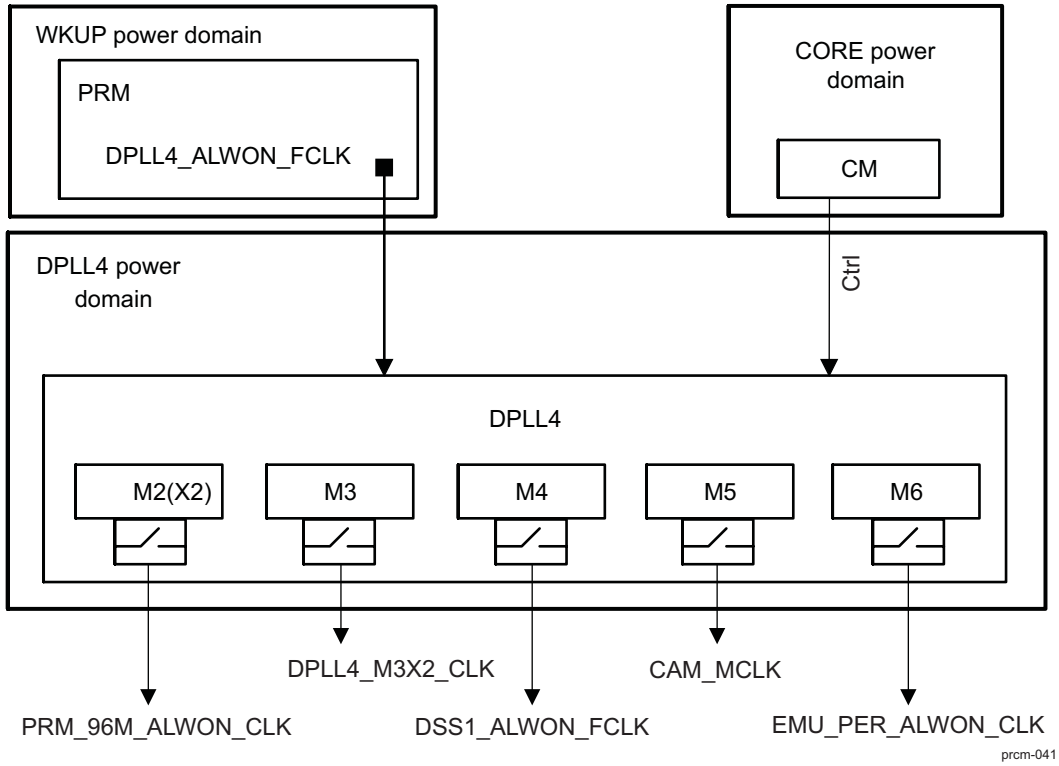


DPLL3 receives its reference clock (DPLL3_ALWON_FCLK), which is the SYS_CLK, from the PRM. DPLL3 does not receive a high-frequency bypass clock, and it uses the reference clock as the low-frequency bypass clock. DPLL3 supplies the source clock for all interfaces and a few functional clocks for the device modules. It is also the source of the emulation trace clock. While the CORE power domain is on, the output of DPLL3 can be used as HS bypass clock input to DPLL1 and DPLL2.

4.7.3.3.3 DPLL4 (Peripherals)

Figure 4-40 is a block diagram of DPLL4.

Figure 4-40. DPLL4 Clocks



DPLL4 receives its reference clock (DPLL4_ALWON_FCLK), which is the SYS_CLK, from the PRM. DPLL4 does not receive a high-frequency bypass clock, and it uses the reference clock as the low-frequency bypass clock. DPLL4 generates clocks for the peripherals, supplying five clock sources:

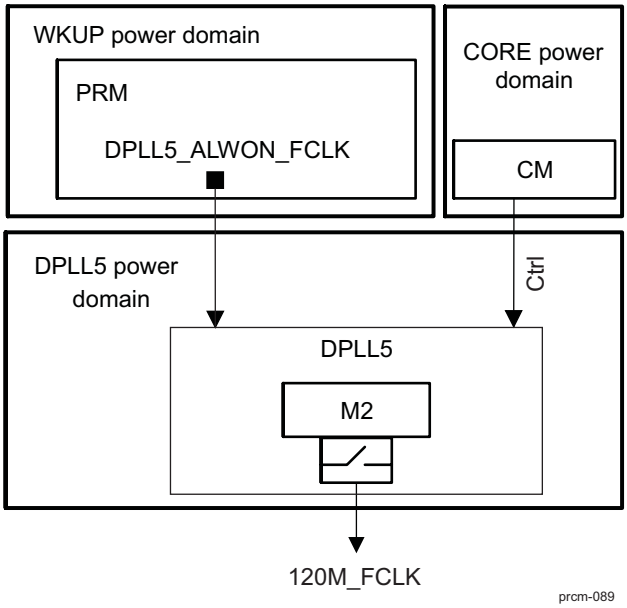
- 96-MHz always-on source clock for the PRM
- 54-MHz to TV DAC
- Display functional clock
- Camera sensor clock
- Emulation trace clock

The clock outputs to the DSS, PER, and EMU power domains are always on.

4.7.3.3.4 DPLL5 (Peripherals)

Figure 4-41 is a block diagram of DPLL5.

Figure 4-41. DPLL5 Clocks



DPLL5 receives its reference clock (DPLL5_ALWON_FCLK), which is the SYS_CLK, from the PRM. DPLL5 does not receive a high-frequency bypass clock, and it uses the reference clock as the low-frequency bypass clock. DPLL5 generates clocks for the peripherals, supplying five clock sources:

- 120-MHz functional clock to the peripheral domain modules

4.7.3.3.5 DPLL Clock Summary

Table 4-29 summarizes the use of the divided output clocks of the five DPLLs in the device.

Table 4-29. DPLL Output Clocks

	CLKOUT_M2	CLKOUT_M2X2	CLKOUT_M3X2	CLKOUT_M4X2	CLKOUT_M5X2	CLKOUT_M6X2
DPLL1		X[1]				
DPLL2	X					
DPLL3	X	X	X			
DPLL4		X	X	X	X	X
DPLL5	X					

(1) X represents the DPLL clock output used.

4.7.3.4 Summary

Table 4-30 summarizes the source clocks in the device.

Table 4-30. Source-Clock Summary

Clock	External/Internal Source	Clock Generator	Description
32K_FCLK	sys_32k (input pin)	PRM	
SYS_CLK	Oscillator	PRM	System clock. Serves as primary source clock of the device. Also used as functional and interface clock for PRM.
DSS_TV_CLK	DPLL4/sys_altclk (input pin)	CM	DSS TV clock
120M_FCLK	DPLL5	CM	
96M_FCLK	DPLL4	CM	
48M_FCLK	DPLL4/sys_altclk (input pin)	CM	
12M_FCLK	DPLL4/sys_altclk (input pin)	CM	
96M_ALWON_CLK		DPLL4	Direct from DPLL4
CORE_CLK	DPLL3	CM	DPLL3 clock output frequency
COREX2_CLK	DPLL3	CM	DPLL3 clock output frequency x 2
L3_ICLK	DPLL3	CM	L3 interconnect interface clock
L4_ICLK	DPLL3	CM	L4 interconnect interface clock
MPU_CLK		DPLL1	MPU subsystem source clock
IVA2_ CLK		DPLL2	IVA2.2 subsystem source clock

4.7.4 Clock Distribution

4.7.4.1 Power Domain Clock Distribution

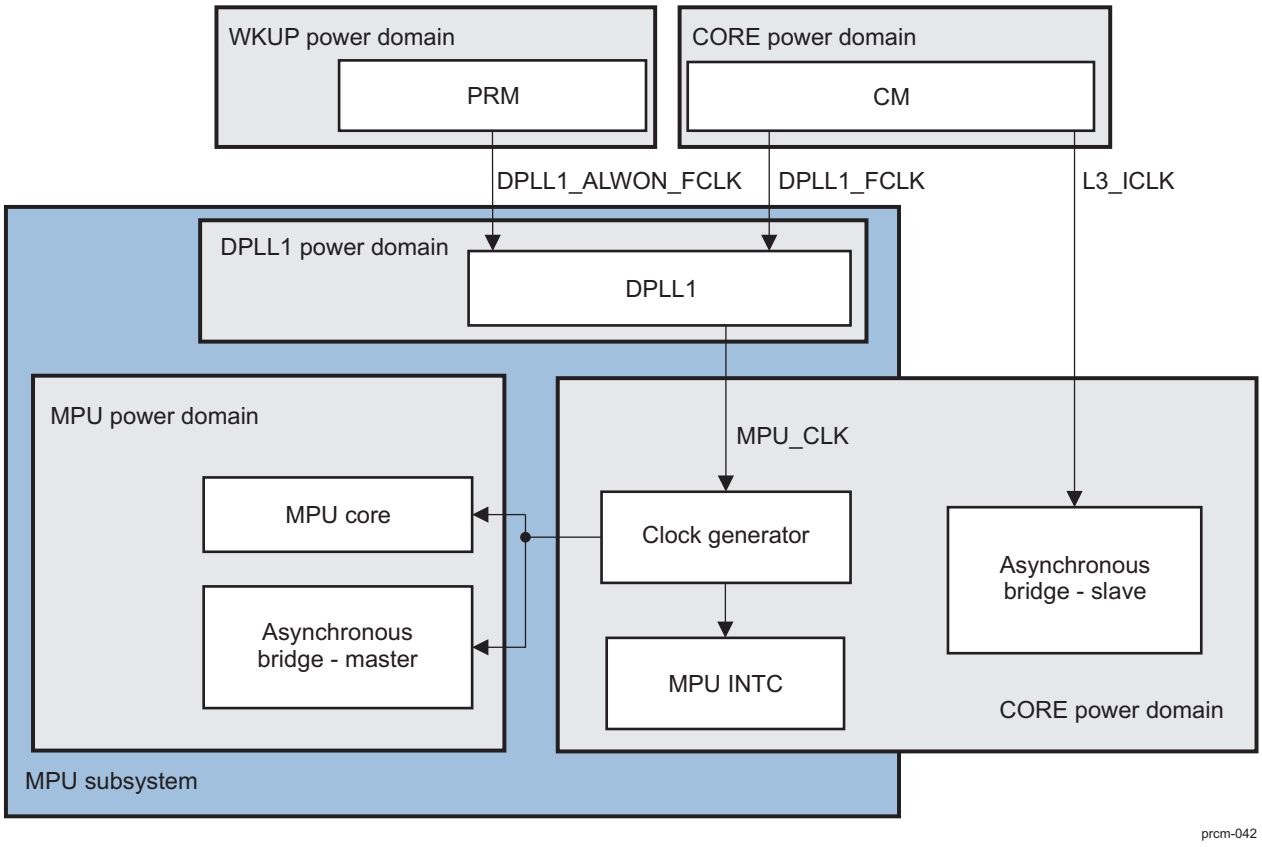
This section describes the PRCM clock distribution over device power domains.

4.7.4.1.1 MPU Power Domain

The PRCM module does not directly provide any clock to the MPU power domain. It feeds only DPLL1, which generates MPU_CLK. All clocks are then locally generated by the clock generator in the MPU subsystem.

Figure 4-42 shows the clocking scheme in the MPU power domain.

Figure 4-42. MPU Power Domain Clocking Scheme



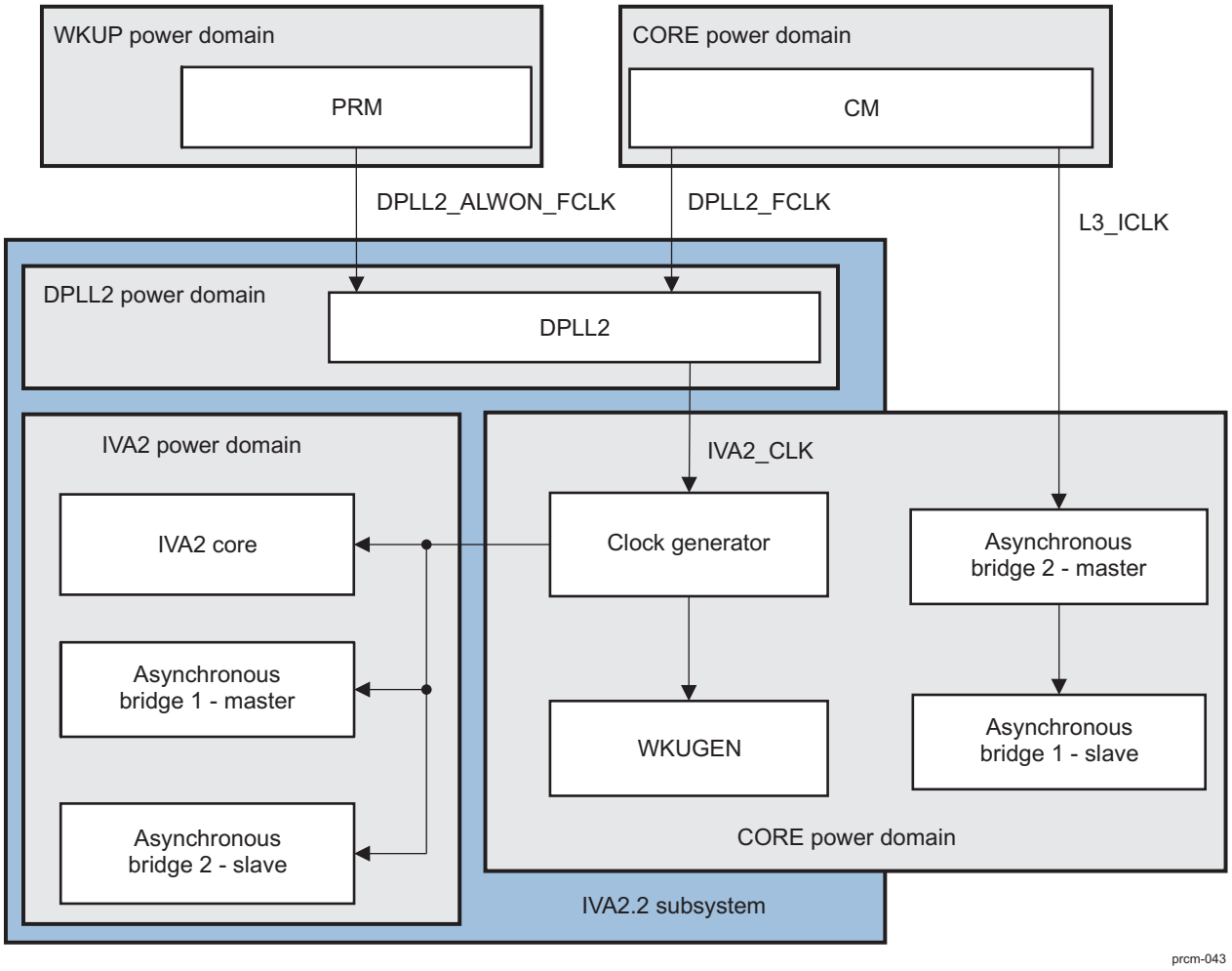
Note: The MPU_CLK is divided by 2 in the MPU subsystem to generate ARM_FCLK. This divider is active only when the DPLL is locked. (For information about ARM_FCL, see Chapter 3, *MPU Subsystem*).

4.7.4.1.2 IVA2 Power Domain

The PRCM module does not directly provide any clock to the IVA2 power domain. It feeds only DPLL2, which generates IVA2_CLK. All clocks are then locally generated by a clock generator in the IVA2.2 subsystem.

Figure 4-43 shows the clocking scheme in the IVA2 power domain.

Figure 4-43. IVA2 Power Domain Clocking Scheme



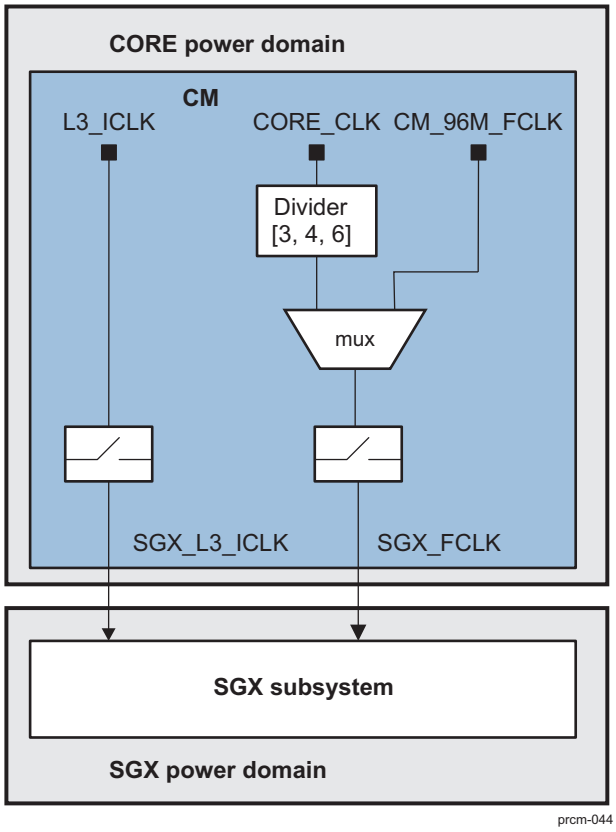
4.7.4.1.3 SGX Power Domain

This section describes all modules and features in the high-tier device. To check availability of modules and features, see Section 1.5, *OMAP34xx Family*. To save power, ensure that power domains of unavailable features and modules are switched off and clocks are cut off.

The SGX subsystem interface clock is sourced by the L3 clock. The functional clock source can be selected between CORE_CLK and CM_96M_FCLK. When the functional clock source is CORE_CLK, its frequency can be divided (by 3, 4, or 6).

Figure 4-44 shows the clocking scheme in the SGX power domain.

Figure 4-44. SGX Power Domain Clocking Scheme



4.7.4.1.4 CORE Power Domain

The CORE power domain has L3- and L4-derived clock domains.

The CORE power domain receives several functional clocks (12-, 48-, 96-MHz, system, and 32-kHz) that feed its peripherals and modules, with following exception:

- The McBSP 1 and McBSP 5 modules can be clocked by CORE_96M_FCLK from the CM or from an external clock, MCBSP_CLKS. The SCM manages the selection between the two sources. For more information about the SCM, see Chapter 7, *System Control Module*.

Figure 4-45 through Figure 4-47 show the clock signals and their relationships in the CORE power domain.

Figure 4-45. CORE Clock Signals: Part 1

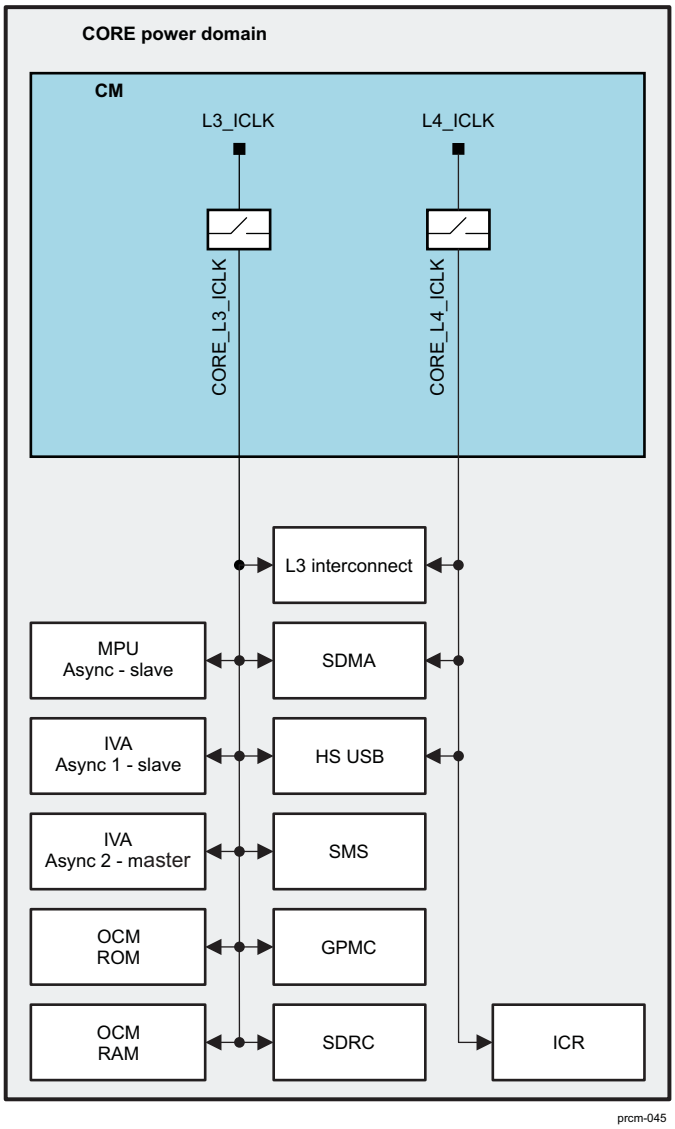


Figure 4-46. CORE Clock Signals: Part 2

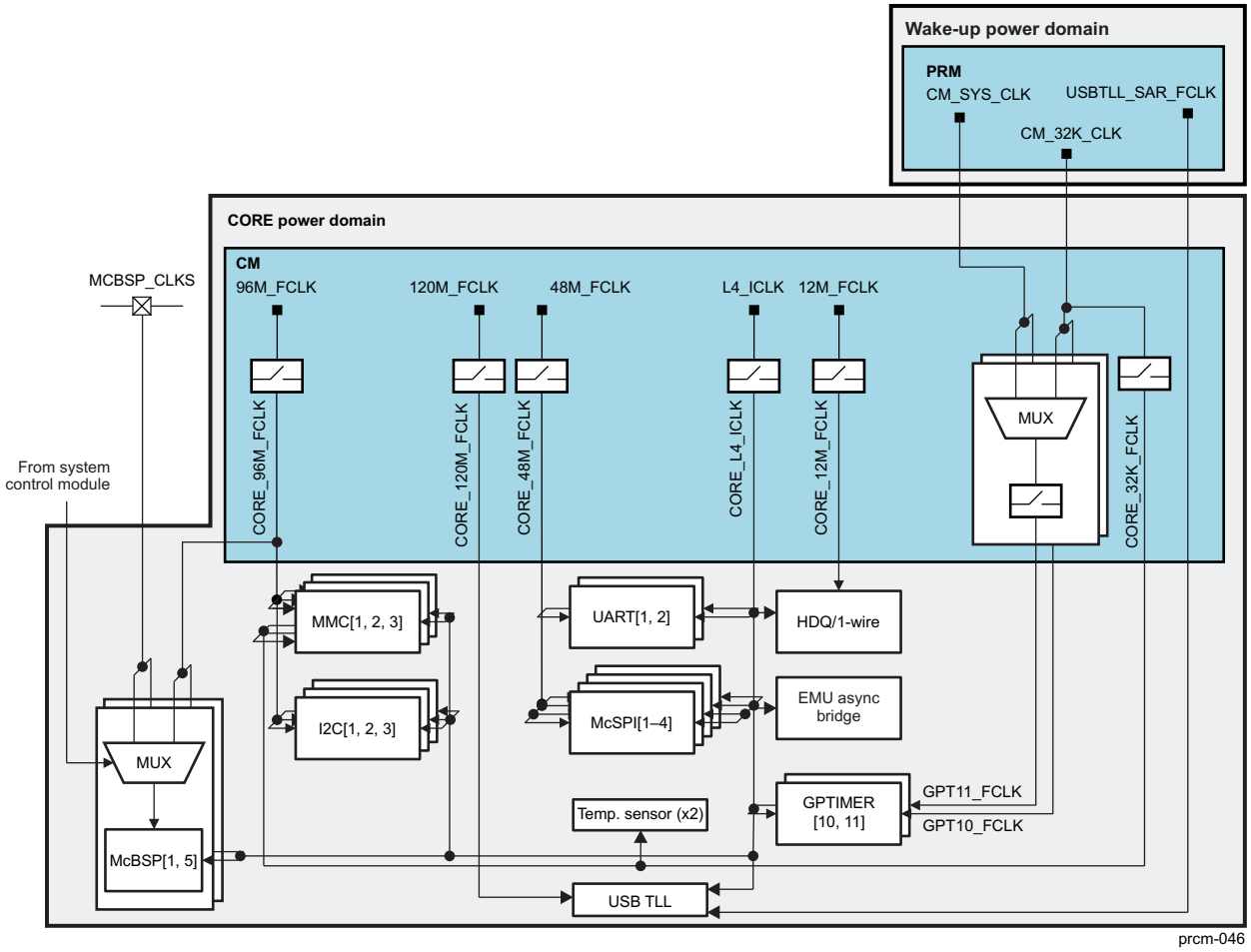
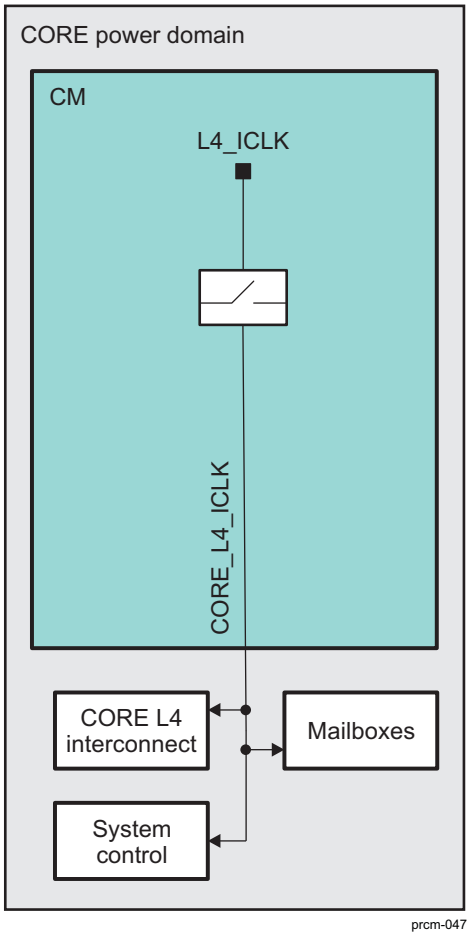


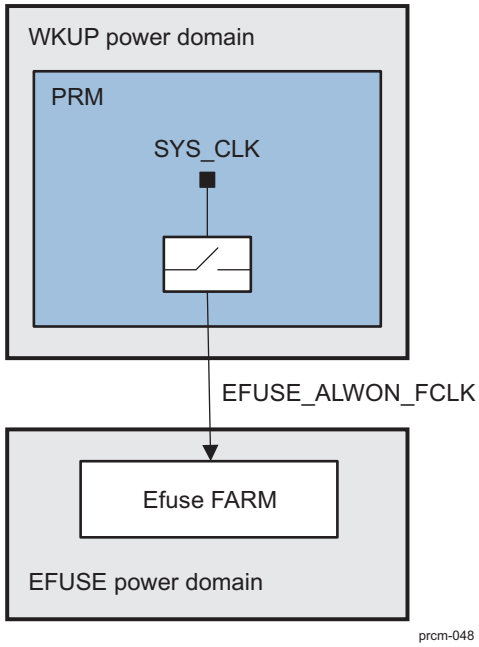
Figure 4-47. CORE Clock Signals: Part 3



4.7.4.1.5 EFUSE Power Domain

Figure 4-48 shows the clock signals and their relationships in the EFUSE power domain.

Figure 4-48. EFUSE Clock Signals



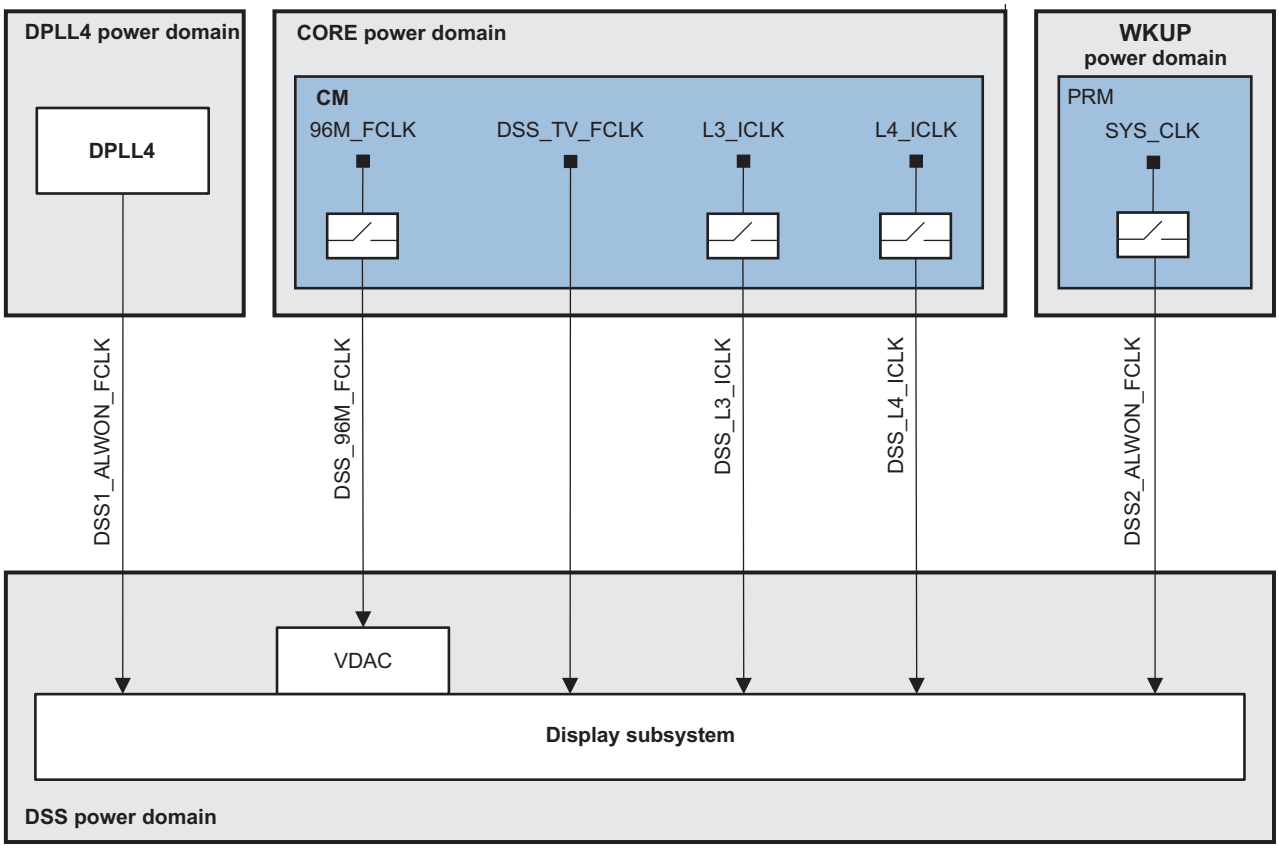
The EFUSE power domain lets the eFuse configuration be saved even when the CORE power domain is off. The EFUSE sense procedure is performed at device power up and on device wakeup from off mode. During this procedure, the PRM enables the EFUSE_ALWON_FCLK clock.

4.7.4.1.6 DSS Power Domain

This section describes all modules and features in the high-tier device. To check availability of modules and features, see Section 1.5, *OMAP34xx Family*. To save power, ensure that power domains of unavailable features and modules are switched off and clocks are cut off.

Figure 4-49 shows the clock signals and their relationships in the DSS power domain.

Figure 4-49. DSS Clock Signals



The DSS subsystem interface is clocked with the L3 and L4 clocks. It receives four functional clocks:

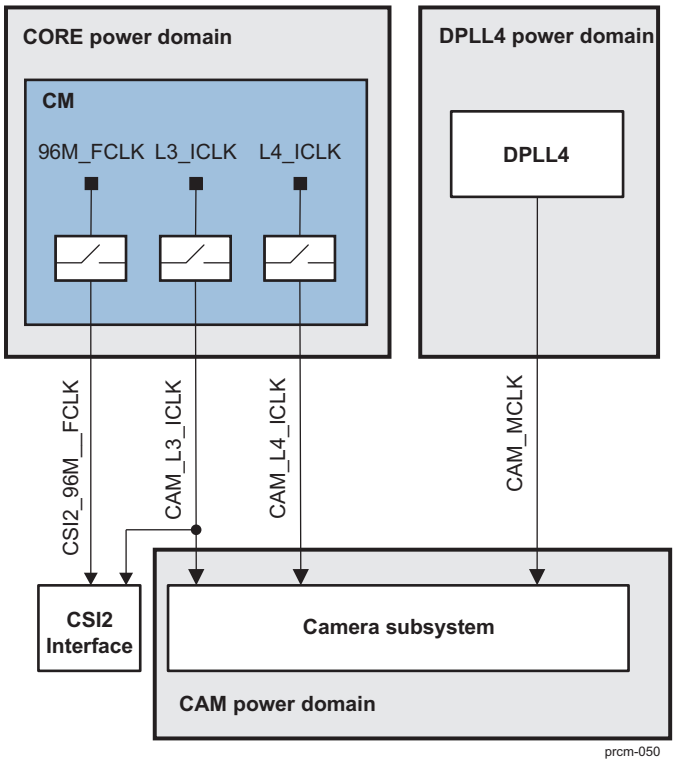
- DSS1_ALWON_FCLK: Issued from DPLL4. Its frequency can be a division by 1 to 16 of the frequency of the DPLL4 synthesized clock.
- DSS2_ALWON_FCLK: The gated SYS_CLK. Used mainly for display in low-power refresh modes.
- DSS_96M_FCLK: Required when TV output is activated
- DSS_TV_FCLK: Required when TV output is activated

4.7.4.1.7 CAM Power Domain

This section describes all modules and features in the high-tier device. To check availability of modules and features, see Section 1.5, *OMAP34xx Family*. To save power, ensure that power domains of unavailable features and modules are switched off and clocks are cut off.

Figure 4-50 shows the clock signals and their relationships in the CAM power domain.

Figure 4-50. CAM Clock Signals

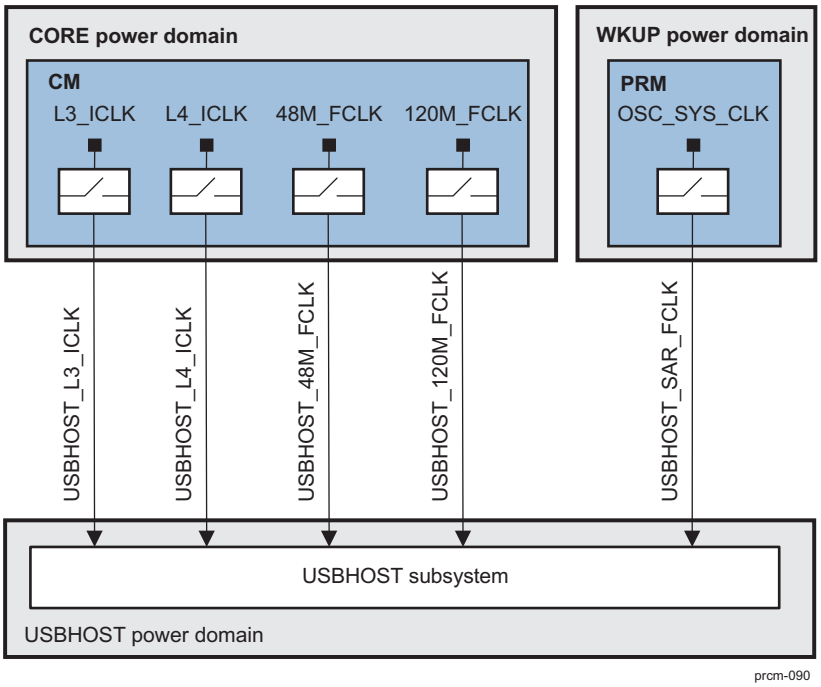


The camera subsystem interface is clocked with the L3 and L4 clocks (CAM_L3_ICLK and CAM_L4_ICLK, respectively). CAM_L3_ICLK is also used as the main functional clock. The functional clock (CAM_MCLK) is provided by DPLL4 to supply the external sensor.

4.7.4.1.8 USBHOST Power Domain

Figure 4-51 shows the clock signals and their relationships in the USBHOST power domain.

Figure 4-51. USBHOST Clock Signals



The HS USB host subsystem interface is clocked with the L3 and L4 clocks (USBHOST_L3_ICLK and USBHOST_L4_ICLK, respectively).

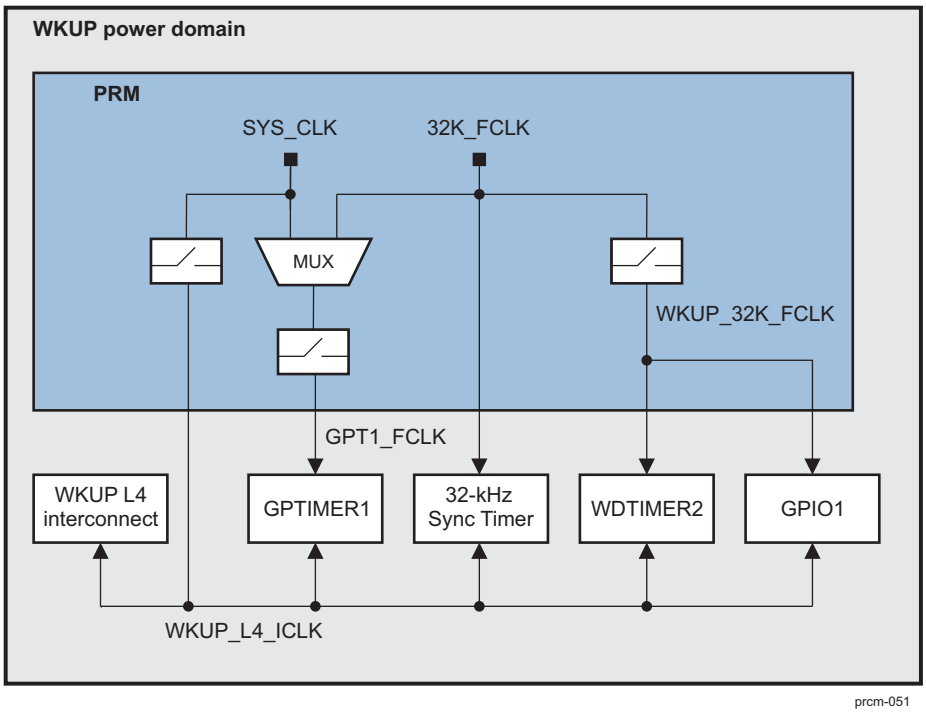
The HS USB host subsystem requires two functional clocks (USBHOST_120M_FCLK and USBHOST_48M_FCLK) that may or may not be requested simultaneously. Therefore, they are gated independently based on the configuration of the CM_FCLKEN_USBHOST[0] EN_USBHOST1 and CM_FCLKEN_USBHOST[1] EN_USBHOST2 bits.

The HS USB host subsystem gets an additional functional clock from the PRM (USBHOST_SAR_FCLK). It is dedicated to the save-and-restore mechanism and is automatically gated/enabled by the PRM, based on the HS USB host save-and-restore bit configuration (the PM_PWSTCTRL_USBHOST[4] SAVEANDRESTORE bit) and on the USBHOST power domain state transitions.

4.7.4.1.9 WKUP Power Domain

Figure 4-52 shows the clock signals and their relationships in the WKUP power domain.

Figure 4-52. WKUP Clock Signals



All clocks in the WKUP power domain are generated by the PRM. The functional clock GPT1_FCLK of GPTIMER1 can be selected as SYS_CLK or 32K_FCLK. The 32-kHz sync timer, WDTIMER2, and GPIO1 receive 32K_FCLK as their functional clock. This is the low-frequency always-on clock.

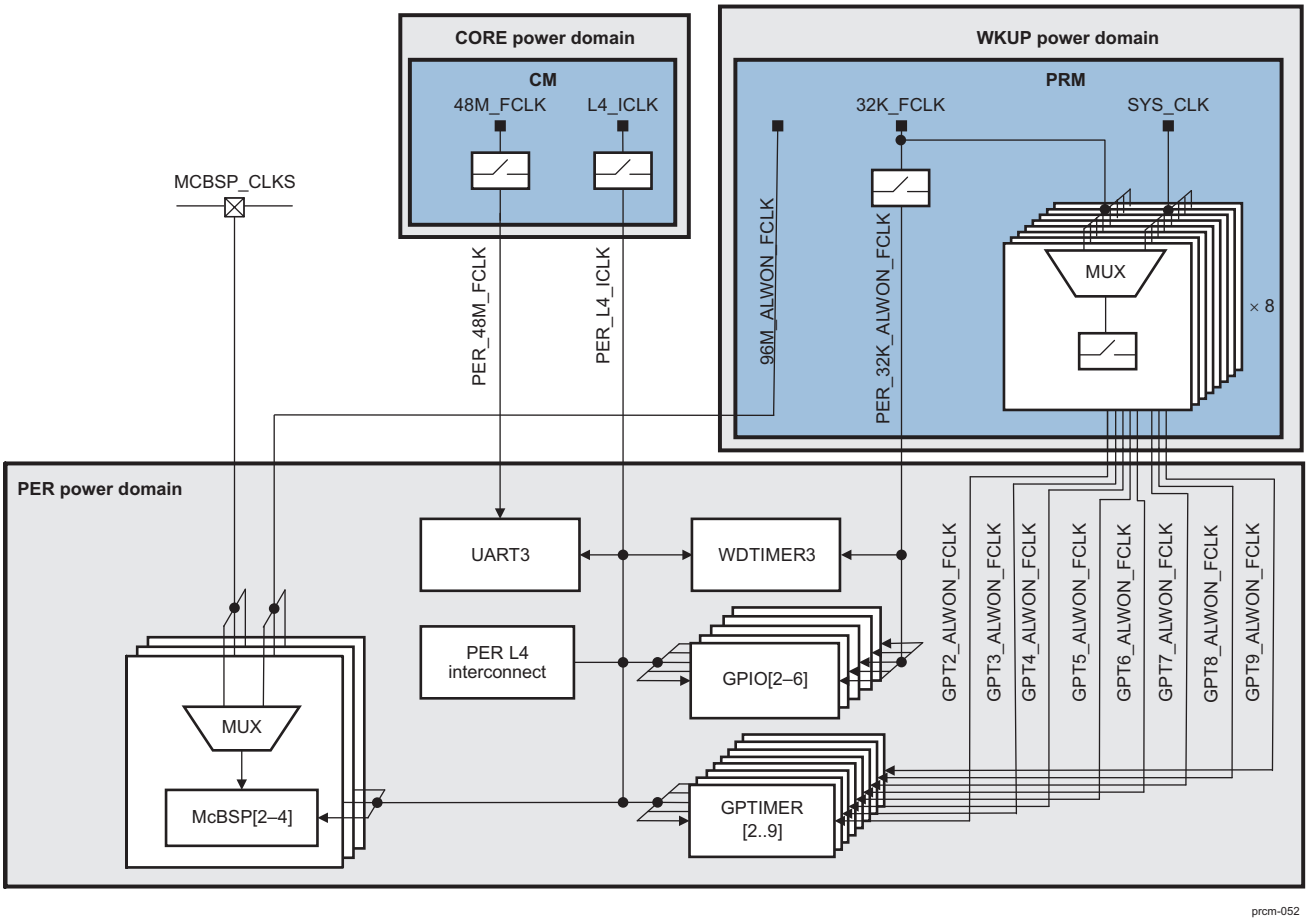
The voltage controller module in the PRM receives SYS_CLK as its functional clock. The dedicated I2C4 module implemented in the voltage controller uses the same functional clock (SYS_CLK).

The PRM receives SYS_CLK as the L4 interface clock. For all other modules of the WKUP power domain, the L4 interface clock WKUP_L4_ICLK is derived from SYS_CLK. Communication between the WKUP power domain and CORE L4 interconnects is asynchronous.

4.7.4.1.10 PER Power Domain

Figure 4-53 shows the clock signals and their relationships in the PER power domain.

Figure 4-53. PER Clock Signals



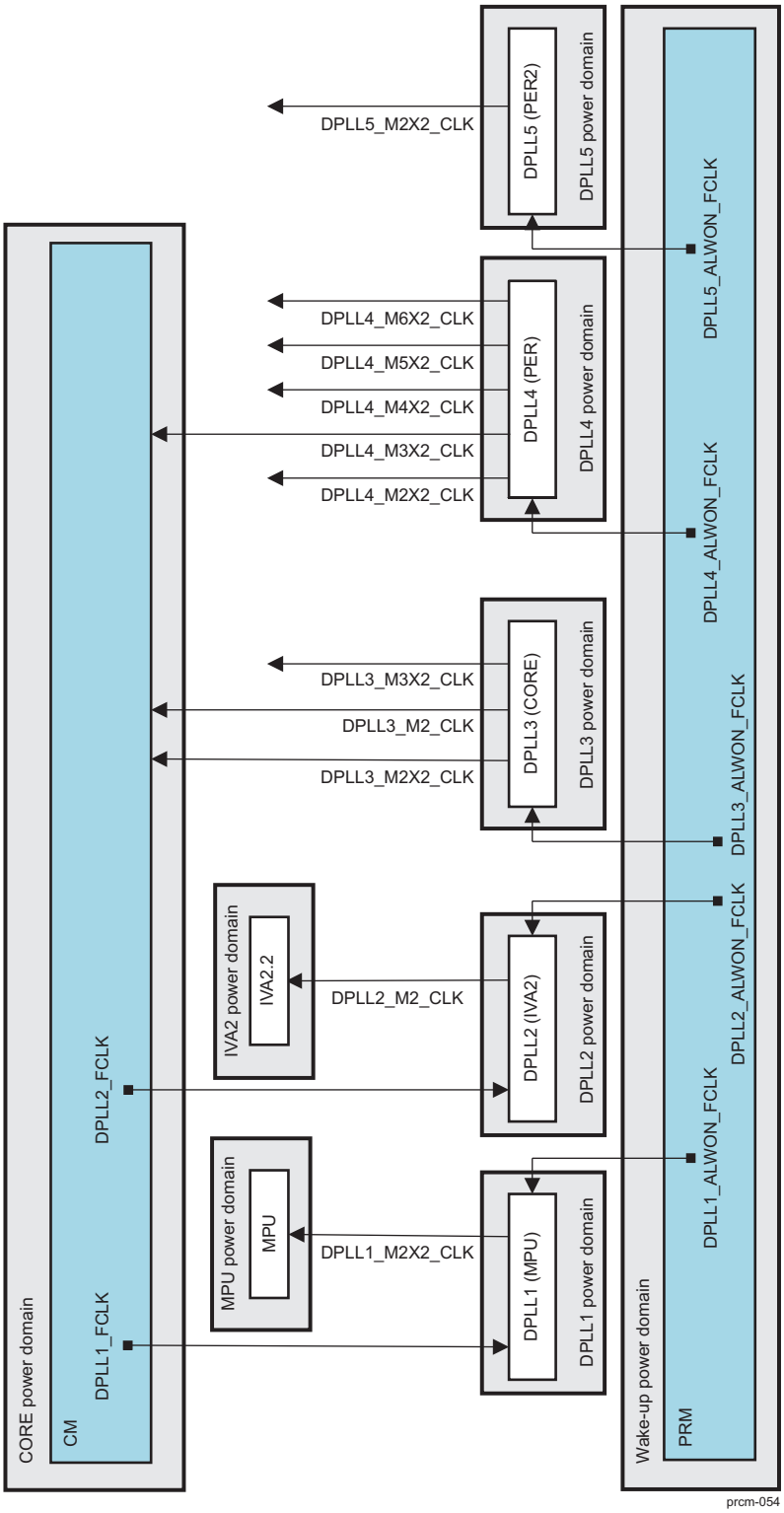
The PER power domain receives several functional clocks (48M_FCLK, 96M_ALWON_FCLK, SYS_CLK, and 32K_FCLK) that feed its peripherals and modules. All the functional clocks (except 48M_FCLK) are permanently supplied so that the peripherals can be used during low-power scenarios, even when the CORE power domain is off. Figure 4-53 shows the clock distribution scheme in the PER power domain.

The McBSP 2, 3, and 4 modules can be clocked by a clock from PRM (PER_96M_FCLK) or from an external clock (MCBSP_CLKS). This clock must be permanently buffered from the pad to the PER power domain. The device SCM manages the selection between the two (see Chapter 7, *System Control Module*).

4.7.4.1.11 DPLL Domains

The PRCM module provides clock sources for the five DPLLs, as shown in Figure 4-54.

Figure 4-54. DPLL Clock Signals



The PRCM module also manages clock-gating control for these DPLL outputs.

The MPU and IVA2 processors use clock outputs locally in their respective subsystems. The CM uses the DPLL3 clock outputs to generate all interface clocks and some functional clocks. The CM uses two of the five clocks generated by DPLL4 and one clock output of DPLL5 to generate functional clocks for the peripheral domain modules. The remaining three clocks of DPLL4 are propagated to their corresponding modules.

4.7.4.2 Clock Distribution Summary

4.7.4.2.1 Power Domain Source Clocks

Table 4-31 summarizes clock distribution for each power domain. The clock type indicates whether a clock is supplied when the CM (CORE power domain) is off. "Normal" means that the clock is gated when the power domain is off, while the always-on clock remains active even when the CORE power domain is off.

Table 4-31. Clock Distribution

Power Domain	Clock	Generator	Type	Destination
MPU	MPU_CLK	DPLL1	Normal	MPU subsystem
NEON				NEON
IVA2	IVA2_CLK	DPLL2	Normal	IVA2.2 subsystem
SGX	SGX_FCLK	CM	Normal	SGX subsystem
	SGX_L3_ICLK	CM	Normal	
CORE	CORE_ 120M_FCLK	CM	Normal	USB TLL
	CORE_ 96M_FCLK	CM	Normal	McBSP[1,5], MMC[1,2,3], I2C[1,2,3]
	CORE_48M_FCLK	CM	Normal	UART[1,2], McSPI[1..4]
	CORE_12M_FCLK	CM	Normal	HDQ
	GPT10_FCLK	CM	Normal	GPTIMER10
	GPT11_FCLK	CM	Normal	GPTIMER11
	USBTLL_SAR_FCLK	PRM	Normal	USB TLL
	CM_32K_CLK	PRM	Normal	Temperature sensor (x2), MMC[1,2,3]
	CORE_L3_ICLK	CM	Normal	L3 interconnect, SDMA, MPU async bridge (slave), IVA2 async bridge 1 (slave), IVA2 async bridge 2 (master), HS USB, SMS, GPMC, OCM ROM, SDRC, OCM RAM, CORE L3 interconnect
	CORE_L4_ICLK	CM	Normal	L3 interconnect, SDMA, HS USB, ICR, McBSP[1,5], MMC[1,2], I2C[1..3], GPTIMER [10,11], UART[1,2], McSPI[1..4], HDQ, CORE L4 interconnect, MAILBOXES, SCM
DSS	DSS_TV_FCLK	CM	Normal	DSS, VDAC
	DSS_96M_FCLK	CM	Normal	VDAC
	DSS1_ALWON_FCLK	DPLL4	Always-on	DSS
	DSS2_ALWON_FCLK	PRM	Always-on	
	DSS_L3_ICLK	CM	Normal	
	DSS_L4_ICLK	CM	Normal	
CAM	CAM_MCLK	DPLL4	Normal	Camera subsystem
	CAM_L3_ICLK	CM	Normal	Camera subsystem, CSI2 interface
	CAM_L4_ICLK	CM	Normal	Camera subsystem
	CSi2_96M_FCLK	CM	Normal	CSI2 interface
PER	96M_ALWON_FCLK	PRM	Always-on	McBSP[2..4]
	PER_48M_FCLK	CM	Normal	UART3

Table 4-31. Clock Distribution (continued)

Power Domain	Clock	Generator	Type	Destination
	PER_32K_ALWON_FCLK	CM	Always-on	WDTIMER3, GPIO[2..6],
	GPT2_ALWON_FCLK	PRM	Always-on	GPTIMER2
	GPT3_ALWON_FCLK	PRM	Always-on	GPTIMER3
	GPT4_ALWON_FCLK	PRM	Always-on	GPTIMER4
	GPT5_ALWON_FCLK	PRM	Always-on	GPTIMER5
	GPT6_ALWON_FCLK	PRM	Always-on	GPTIMER6
	GPT7_ALWON_FCLK	PRM	Always-on	GPTIMER7
	GPT8_ALWON_FCLK	PRM	Always-on	GPTIMER8
	GPT9_ALWON_FCLK	PRM	Always-on	GPTIMER9
	PER_L4_ICLK	CM	Normal	UART3, PER L4 interconnect, WDTIMER3, GPIO[2..6], GPTIMER[2..9], McBSP[2..4]
WKUP	WKUP_32K_FCLK	PRM	Always-on	WDTIMER2, GPIO1
	32K_FCLK	PRM	Always-on	32-kHz sync timer
	GPT1_FCLK	PRM	Always-on	GPTIMER1
	WKUP_L4_ICLK	PRM	Normal	WKUP L4 interconnect, GPTIMER1, 32-kHz sync timer, GPIO1, WDTIMER2
EFUSE	EFUSE_ALWON_FCLK	PRM	Always-on	eFuse farm
DPLL1	DPLL1_ALWON_FCLK	PRM	Always-on	DPLL1
	DPLL1_FCLK	CM	Normal	
DPLL2	DPLL2_ALWON_FCLK	PRM	Always-on	DPLL2
	DPLL2_FCLK	CM	Normal	
DPLL3	DPLL3_ALWON_FCLK	PRM	Always-on	DPLL3
DPLL4	DPLL4_ALWON_FCLK	PRM	Always-on	DPLL4
DPLL5	DPLL5_ALWON_FCLK	PRM	Always-on	DPLL5

Notes:

- Modules supplied by the L3 interface clock only:
  - MPU asynchronous bridge
  - IVA2.2 asynchronous bridges
  - All memory controllers (OCM ROM, OCM RAM, SDRC, SMS, and GPMC)
- Modules that require L3 and L4 clocks:
  - SDMA
  - HS USB
- Modules fed by the L4 clock:
  - SCM
  - Mailboxes
  - ICR
  - Modem INTC
  - All peripherals (McBSP1, McBSP5, MMC1, MMC2, , I2C1, I2C2, I2C3, McSPI1, McSPI2, McSPI3, McSPI4, UART1, UART2, HDQ, GPTIMER10, and GPTIMER11)

4.7.4.2.2 Peripheral Module Clocks

Table 4-32 lists the peripherals, DSS, and CAM functional clock frequency requirements. These frequencies must be operational over the full VDD2 voltage range.

Table 4-32. Peripheral Module Functional Clock Frequencies

Module	Functional Clock	Frequency
MMC-SDIO[1,2,3]	96M_FCLK	96 MHz
McBSP[1, 5]	96M_FCLK	96 MHz
CAM	CAM_MCLK	216 MHz
McSPI[1..4]	CORE_48M_FCLK	48 MHz
UART[1..3]		
Display subsystem	DSS1_ALWON_FCLK	Up to 173 MHz
	DSS2_ALWON_FCLK	System clock
	DSS_96M_FCLK	96 MHz
	DSS_TV_FCLK	54 MHz
I2C[1..3]	CORE_96M_FCLK	96 MHz
HDQ	CORE_12M_FCLK	12 MHz
GPTIMER1	GPT1_FCLK	32-kHz (p) or system clock
GPTIMER[2..9]	GPTn_ALWON_FCLK	32-kHz (p) or system clock
GPTIMER[10, 11]	GPTn_FCLK	32-kHz or system clock
ICR		
WDTIMER2	WKUP_32K_FCLK	32 kHz
WDTIMER3	PER_32K_ALWON_FCLK	32 kHz
GPIO1	WKUP_32K_FCLK	32 kHz
GPIO[2-6]	PER_32K_ALWON_FCLK	32 kHz
32-kHz sync timer	32K_FCLK	32 kHz (p)
Bandgap/temp sensor	32K_FCLK	32 kHz (p)
System control	CORE_L4_ICLK	L4_ICLK

4.7.5 External Clock Controls

Because the use of sys_32k and sys_altclk is described in Section 4.7.3.1, *PRM*, and Section 4.7.3.2, *CM*, these clock signals are not discussed here. This section discusses the remaining external clock signals.

4.7.5.1 Clock Request (sys_clkreq) Control

The system clock request signal sys_clkreq is bidirectional.

In bypass mode in the system clock oscillator (see Section 4.7.5.2), it is an output signal driven by the device to request an external clock. In this case, the output buffer is driven as long as the system clock is requested by the device; otherwise, it remains in high impedance. In this way, other external peripherals can share the same clock request signal with the device.

If the PRM_POLCTRL[1] CLKREQ_POL bit = 1, the software must configure the SCM to select the internal pulldown on the sys_clkreq pad, or an external pulldown is connected to the pad.

If the PRM_POLCTRL[1] CLKREQ_POL bit = 0, the internal pullup on the sys_clkreq pad, or an external pull-up, is connected to the pad.

In master mode in the system clock oscillator (see Section 4.7.5.2), sys_clkreq is an input.

If the PRM_POLCTRL[1] CLKREQ_POL bit = 1, the software must configure the SCM to select the internal pulldown on the sys_clkreq pad, or an external pulldown is connected to the pad.

If the PRM_POLCTRL[1] CLKREQ_POL bit = 0, the internal pullup on the sys_clkreq pad, or an external pullup, is connected to the pad.

The PRCM.PRM_POLCTRL[1] CLKREQ_POL bit allows software control over the polarity of sys_clkreq. This software setting takes effect when the clock is requested by the device, and also when the clock request is driven externally. The output buffer is driven directly by this register when the clock request comes from OMAP.

Table 4-33 describes the bidirectional control of the sys_clkreq pad.

Table 4-33. sys_clkreq Pad Direction Control

Oscillator Mode	Sys_boot 6	Internal Clock Request (Always Active-High)	External Clock Request (Note: polarity depends on CLKREQ_POL)	Sys_clkreq Direction	Description
Master mode	0	0	0(1)	Input (output buffer in Hi-Z) **Note**: Input is not driven from outside of OMAP in this case.	The clock is not requested internally (by OMAP3430) or externally (external device/peripheral).
	0	0	1(1)	Input (output buffer in Hi-Z)	The clock is requested externally.
	0	1	0(1)	Output	The clock is requested internally.
	0	1	1(1)	Output **Note**: The pad is driven both by OMAP and from outside of OMAP in this case.	The clock is requested internally and externally.
Bypass mode	1	0	0(1)	Input (output buffer in Hi-Z) **Note**: Input is not driven from outside of OMAP in this case).	The clock is not requested internally or externally.
	1	0	1(1)	Input (output buffer in Hi-Z)	The clock is requested externally.
	1	1	0(1)	Output	The clock is requested internally.
	1	1	1(1)	Output **Note**: The pad is driven by OMAP and from outside of OMAP in this case.	The clock is requested internally and externally.

(1) Case when PRM_POLCTRL.CLKREQ_POL = 1 (sys_clkreq active high). If PRM_POLCTRL.CLKREQ_POL = 0 (sys_clkreq active low), these values should be inverted in Table 4-33.

4.7.5.2 System Clock Oscillator Control

Depending on the hardware configuration, the device can receive the system clock from an external source or generate it locally using the internal system clock crystal oscillator. Thus, the device oscillator has two possible operating modes:

- Master (oscillator enable) mode: The oscillator is enabled and connected to an external quartz. It provides the system clock to the device. The oscillator is activated on a device wakeup or on an external clock request. It is set to the power-down (off) state when the device switches to off mode (see Table 4-35), except when the external system clock request is active on sys_clkreq pin.
- Bypass (oscillator inactive) mode: The system clock is supplied by an external device and the oscillator is always set in bypass mode. The oscillator is insensitive to the external system clock request on the sys_clkreq pin.

Note: An external pullup or pulldown tied on the sys_boot6 input pin of the device determines whether the oscillator is in master or bypass mode. See Section 4.3.1, *External Clock Signals*.

When operating in master mode, the device receives an external clock request (sys_clkreq) and provides the system clock to external peripherals through the sys_clkout1 pin; in bypass mode, the device generates a clock request to the external clock source to request the system clock.

The selected mode of the oscillator can be read from the PRCM.PRM_CLKSRC_CTRL[1:0] SYSCLKSEL bit field.

Regardless of oscillator mode, the oscillator can be powered down when the device enters inactive, retention or off mode, unless an external system clock request is active (from the sys_clkreq pin). This setting is configured in the PRCM.PRM_CLKSRC_CTRL[4:3] AUTOEXTCLKMODE bit field. Table 4-35 lists the four possible operation modes of the system clock.

Table 4-34. System Clock Operation Modes

AUTOEXTCLKMODE	System Clock Mode	Oscillator Mode	Description
0x0	Always-active mode	Master	The oscillator is kept active even when the clock is not requested by the device internally (all device clocks are inactive) or externally (the sysclkreq input signal is not asserted).
		Bypass	The sys_clkreq output signal is permanently asserted by the device, regardless of its internal clock states (active or inactive).
0x1	Off when device in inactive, retention, or off state	Master	The oscillator is switched to off mode when the device is in inactive, retention, or off mode and the sys_clkreq input signal is not asserted.
		Bypass	The sys_clkreq output signal is deasserted when the device is in idle, retention, or off mode.
0x2	Off when device in retention or off state	Master	The oscillator is switched to off mode when the device is in retention or off mode and the sys_clkreq input signal is not asserted.
		Bypass	The sys_clkreq output signal is deasserted when the device is in retention or off mode.
0x3	Off when device in off state	Master	The oscillator is switched to off mode when the device is in off mode and the sys_clkreq input signal is not asserted.
		Bypass	The sys_clkreq output signal is deasserted when the device is in off mode.

To exit power-down mode, the oscillator requires a device wakeup or an external clock request.

The device allows configuring of the system clock stabalization time to ensure a stable system clock in the device. This delay is configured in the PRCM.PRM_CLKSETUP[15:0] SETUP_TIME bit field.

Figure 4-55 shows the system clock oscillator controls in the device.

Figure 4-55. System Clock Oscillator Controls

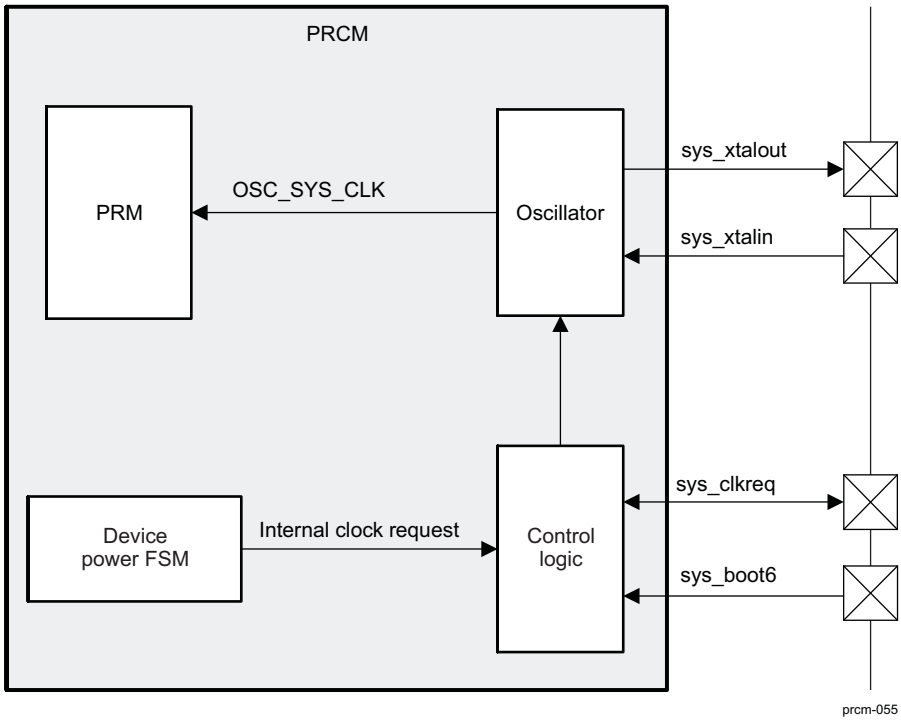


Table 4-35 lists the oscillator controls.

Table 4-35. Oscillator Controls

Oscillator Mode	Internal Clock Request[1]	External Clock Request[1]	Oscillator State	sys_clkreq Pad Direction	sys_clkreq[1]	Description
Master	Not asserted	Not asserted	Off	Input and output	Not asserted	System clock not requested internally (within the device) or externally (by an external device or peripheral)
	Not asserted	Asserted	Active	Input	Asserted	System clock is requested externally only.
	Asserted	Not asserted	Active	Output	Asserted	System clock is requested internally only.
	Asserted	Asserted	Active	Input and output (driven internally by device and externally by peripheral)	Asserted	System clock is requested internally and externally.
Bypass	Not asserted	x[2]	Bypass	Input and output (when external request is not asserted) or input (when external request is asserted)	External clock request state	System clock is not requested internally (sys_clkreq input has no effect).
	Asserted	x[2]	Bypass	Output (when only internal request is asserted) or input and output (when external and internal request are asserted)	Asserted	System clock is requested internally (sys_clkreq input has no effect).

(1)
- If the PRCM.PRM_POLCTRL[1] CLKREQ_POL is set to active high (0x1), Asserted = 1, and Not asserted = 0.
- If the PRCM.PRM_POLCTRL[1] CLKREQ_POL is set to active low (0x0), Asserted = 0, and Not asserted = 1.

(2) x indicates that the signal may be asserted or not asserted.

4.7.5.3 External Output Clock1 (sys_clkout1) Control

The sys_clkout1 clock is active if the oscillator clock (OSC_SYS_CLK) is active (stable) and an external system clock request is active. It can be gated by programming the PRCM.PRM_CLKOUT_CTRL[7] CLKOUT_EN bit. The polarity of the sys_clkout1 signal, when the clock is gated, is controllable by programming the PRCM.PRM_POLCTRL[2] CLKOUT_POL bit.

When the device is in standby mode, SYS_CLK and sys_clkout1 are disabled. in this case, reactivation of sys_clkout1 depends on the oscillator mode:

- Oscillator in active mode (sys_boot6 is 0): The sys_clkout1 clock can be reactivated (after oscillator stabilization), provided its gating was enabled by programming the PRCM.PRM_CLKOUT_CTRL[7] CLKOUT_EN bit and asserting an external clock request. This activation does not generate a device wake-up event; an external clock request activates only the internal SYS_CLK oscillator and sys_clkout1.
- Oscillator in bypass mode (sys_boot6 is 1): The sys_clkout1 clock can be reactivated only after the device wakes up (on any wake-up event) and SYS_CLK is active. When the device is active, SYS_CLK is running and sys_clkout1 is enabled as soon as requested by software.

4.7.5.4 External Output Clock2 (sys_clkout2) Control

A second output clock, sys_clkout2, is generated with a frequency that can be the source-clock frequency divided by 1, 2, 4, 8, or 16. Its source clock can be CORE_CLK, CM_SYS_CLK, 96 MHz, or 54 MHz. Unlike sys_clkout1, this second external clock is not active when the device is in off power mode. Also, the selected source clock must be enabled by software. Enabling sys_clkout2 does not automatically request the required source clock. The polarity of the sys_clkout2 signal, when the clock is gated, is controllable by programming the PRCM.CM_POLCTRL[0] CLKOUT2_POL bit.

4.7.6 DPLL Control

The PRCM module allows the configuration of the output clock frequencies of the DPLLs by setting their multipliers and dividers. It also allows control of the operating mode of the DPLLs and automatic recalibration mode.

4.7.6.1 DPLL Multiplier and Divider Factors

DPLL clock outputs are set by programming the corresponding multiplier and divider factors M, N, M2, M3, M4, M5, and M6. Table 4-36 lists the register bit fields for configuration of the multiplier and divider factors for the DPLLs.

Table 4-36. DPLL Multiplier and Divider Factors

	DPLL1	DPLL2	DPLL3	DPLL4	DPLL5
M	PRCM.CM_CLKSEL1_ PLL_MPU[18:8] MPU_ DPLL_MULT	PRCM. CM_CLKSEL1_PLL_ IVA2[18:8] IVA2_ DPLL_MULT	PRCM.CM_CLKSEL1_ PLL[26:16] CORE_DPLL_MULT	PRCM.CM_CLKSEL2_ PLL[18:8] PERIPH_ DPLL_MULT	PRCM.CM_CLKSEL4_ PLL[18:8] PERIPH2_ DPLL_MULT
N	PRCM.CM_CLKSEL1_ PLL_MPU[6:0] MPU_ DPLL_DIV	PRCM. CM_CLKSEL1_PLL_ IVA2[6:0] IVA2_DPLL_DIV	PRCM.CM_CLKSEL1_ PLL[14:8] CORE_DPLL_DIV	PRCM.CM_CLKSEL2_ PLL[6:0] PERIPH_ DPLL_DIV	PRCM.CM_CLKSEL4_ PLL[6:0] PERIPH2_ DPLL_DIV
M2	PRCM.CM_CLKSEL2_ PLL_MPU[4:0] MPU_ DPLL_CLKOUT_DIV	PRCM. CM_CLKSEL2_PLL_ IVA2[4:0] IVA2_DPLL_ CLKOUT_DIV	PRCM.CM_CLKSEL1_ PLL[31:27] CORE_DPLL_ CLKOUT_DIV	PRCM.CM_CLKSEL3_ PLL[4:0] DIV_96M	PRCM.CM_CLKSEL5_ PLL[4:0] DIV_120M
M3	Not used	Not used	PRCM.CM_CLKSEL1_ EMU[20:16] DIV_DPLL3	PRCM.CM_CLKSEL_ DSS[12:8] CLKSEL_TV	Not used
M4	Not used	Not used	Not used	PRCM.CM_CLKSEL_ DSS[4:0] CLKSEL_DSS1	Not used
M5	Not used	Not used	Not used	PRCM.CM_CLKSEL_ CAM[4:0] CLKSEL_CAM	Not used
M6	Not used	Not used	Not used	PRCM.CM_CLKSEL1_ EMU[28:24] DIV_DPLL4	Not used

4.7.6.2 DPLL Modes

DPLL supports several power modes (see Table 4-37). Each mode results in a tradeoff between power savings and relock time.

Table 4-37. DPLL Power Modes

Mode	Clock Input	Clock Output	DPLL Power State	Power Consumption	Latency
Locked	On	Lock frequency	ON	Maximum	N/A
Low-power bypass	On	Bypass frequency	ON	Less than locked	Same as low-power stop
Fast-relock bypass	On	Bypass frequency	ON	Less than locked	Less than low-power bypass
Low-power stop	On	Bypass frequency	ON	Less than locked	Same as low-power bypass
MN bypass	On	Bypass frequency	ON	Less than locked	Maximum
Off	Off	Off	Off	Minimum	Maximum

A DPLL power mode can be achieved on a software request (manual) and/or automatically (automatic), depending on the specific hardware conditions. After a device power-on reset, the DPLL can be kept in low-power stop mode (DPLL2 , DPLL4, and DPLL5) or MN bypass mode (DPLL1 and DPLL3).

A DPLL can switch from one mode to the other as a result of the following:

- Software-programmed transition only (manual): The software configures a dedicated register for the next desired DPLL mode. It must ensure that the transition can be performed based on the activity on the device.
- Combined software-programmed and hardware-conditions-based transition (automatic): The PRCM module triggers the transition when the software requests it (by configuring the registers) and the hardware conditions are satisfied. When the hardware conditions are no longer met, the PRCM module triggers the return transition.

For automatic transition, automatic mode must be enabled by programming the PRCM.CM_AUTOIDLE_PLL or the PRCM.CM_AUTOIDLE_PLL<processor_name> registers.

Table 4-38 describes the manual and automatic control of the DPLL power modes by the PRCM module.

Table 4-38. DPLL Power Mode Support

Mode	DPLL1	DPLL2	DPLL3	DPLL4	DPLL5
Locked	Software request (manual) or MPU wakes up (automatic).	Software request (manual) or IVA2.2 wakes up (automatic).	Software request (manual) or CORE wakes up (automatic).	Software request (manual) or at least one peripheral clock is used (automatic).	Software request (manual) or at least one peripheral clock is used (automatic).
Low-power bypass	Software request (manual)	Software request (manual)	Software request (manual) or all interface clocks are gated (automatic).	N/A	N/A
Fast-relock bypass	N/A	N/A	Software request (manual)	N/A	N/A
Low-power stop	MPU is idle (automatic).	Software request (manual) or IVA2.2 is idle (automatic) or on global reset release (automatic).	Device is idle (automatic).	(Default state) Software request (manual) or all functional clocks from DPLL are unused or on global reset release (automatic).	(Default state) Software request (manual) or all functional clocks from DPLL (120-MHz clock) are unused or on global reset release (automatic).

Table 4-38. DPLL Power Mode Support (continued)

Mode	DPLL1	DPLL2	DPLL3	DPLL4	DPLL5
MN bypass	Global reset (automatic)	N/A	Global reset (automatic)	N/A	N/A
Off	Device off (automatic)	Device off (automatic)	Device off (automatic)	Device off (automatic)	Device off (automatic)

Note: DPLL1 and DPLL3 cannot be manually forced to switch to Low-Power Stop mode from any other power mode. They must be in Locked state with automatic transition to Low-Power Stop mode configured and the hardware condition for the transition (identified in Table 4-38) must be satisfied, in order to switch to the Low-Power Stop mode.

Table 4-39 lists the bit fields that must be programmed for manual and automatic mode control of the five DPLLs.

Table 4-39. DPLL Power Mode Control

Mode	Manual Control	Auto Control
DPLL1	PRCM.CM_CLKEN_PLL_MPU[2:0] EN_MPU_DPLL	PRCM.CM_AUTOIDLE_PLL_MPU[2:0] AUTO_MPU_DPLL
DPLL2	PRCM.CM_CLKEN_PLL_IVA2[2:0] EN_IVA2_DPLL	PRCM.CM_AUTOIDLE_PLL_IVA2[2:0] AUTO_IVA2_DPLL
DPLL3	PRCM.CM_CLKEN_PLL[2:0] EN_CORE_DPLL	PRCM.CM_AUTOIDLE_PLL[2:0] AUTO_CORE_DPLL
DPLL4	PRCM.CM_CLKEN_PLL[18:16] EN_PERIPH_DPLL	PRCM.CM_AUTOIDLE_PLL[5:3] AUTO_PERIPH_DPLL
DPLL5	PRCM.CM_CLKEN2_PLL[2:0] EN_PERIPH2_DPLL	PRCM.CM_AUTOIDLE2_PLL[2:0] AUTO_PERIPH2_DPLL

Note: The DPLL automatically enters locked mode on a power domain wakeup only if the DPLL is locked before the sleep transition and one of the automatic modes is enabled.

4.7.6.3 DPLL Low-Power Mode

The DPLL can operate in a low-power mode by reducing the operating frequency range. This reduces the power consumption of the DPLL. In this mode, however, there is a period and phase jitter effect.

The DPLL can enter this mode only if the targeted lock frequency of the DPLL is less than 600 MHz. This implies locking or relocking the DPLL to a new targeted locked-frequency when entering or exiting low-power mode. Software must ensure that the DPLL lock frequency does not exceed 600 MHz in low-power mode.

Software can enable/disable automatic switching of the DPLL between normal mode and low-power mode. The new mode is effective only after the DPLL is relocked. Low-power mode control is considered only during the following transitions:

- Bypass mode to lock
- Stop mode to lock
- Lock to relock

Table 4-40 lists the bit fields that must be programmed for manual control of the five DPLLs.

Table 4-40. LP Mode Control

Mode	Manual Control
DPLL1	PRCM.CM_CLKEN_PLL_MPU[10] EN_MPU_DPLL_LPMODE
DPLL2	PRCM.CM_CLKEN_PLL_IVA2[10] EN_IVA2_DPLL_LPMODE

Table 4-40. LP Mode Control (continued)

Mode	Manual Control
DPLL3	PRCM.CM_CLKEN_PLL[10] EN_CORE_DPLL_LPMODE
DPLL4	PRCM.CM_CLKEN_PLL[26] EN_PERIPH_DPLL_LPMODE
DPLL5	PRCM.CM_CLKEN2_PLL[10] EN_PERIPH2_DPLL_LPMODE

4.7.6.4 DPLL Clock Path Power Down

DPLL3 and DPLL4 can power down the CLKOUTX2 path. A small section of logic is powered down, because the M2 post divider is shared with the CLKOUT path, which remains functional.

The HSDIVIDER can power down each CLKOUTn path (with n in the range of 3 to 6) independently, therefore allowing further power savings. The clock output path is also powered down when the DPLL is in stop mode, regardless of the software setting.

Software must ensure correct sequencing of the control. To avoid a glitch at the output, activate this control when the clock is not required, and when the output clock is gated. Conversely, ensure a delay between deactivation and reactivation of the clock by using the power-down control.

Table 4-41 lists the bit fields and the corresponding clock outputs of the DPLLs.

Table 4-41. Clock Path Power-Down Control

DPLL	Control Bit Field	Clock Path
DPLL4	PRCM.CM_CLKEN_PLL[27] PWRDN_96M	96-MHz clock output (DPLL4 output M2X2)
	PRCM.CM_CLKEN_PLL[28] PWRDN_TV	DSS TV clock output (DPLL4 output M3X2)
	PRCM.CM_CLKEN_PLL[29] PWRDN_DSS1	DSS1 clock output (DPLL4 output M4X2)
	PRCM.CM_CLKEN_PLL[30] PWRDN_CAM	CAM clock output (DPLL4 output M5X2)
	PRCM.CM_CLKEN_PLL[31] PWRDN_EMU_PERIPH	EMU_PERIPH clock output (DPLL4 output M6X2)
DPLL3	PRCM.CM_CLKEN_PLL[12] PWRDN_EMU_CORE	EMU_CORE clock output (DPLL3 output M3X2)

4.7.6.5 Recalibration

A lock sequence occurs during an initial lock or during a relock following a new multiplier or divider value. Each time the DPLL is reset or performs a lock sequence, it performs a recalibration of the output frequency, based on voltage and temperature conditions. By compensating for voltage and temperature changes within a certain range, calibration allows the lock frequency to remain steady. If the voltage or temperature drifts outside the acceptable range, the DPLL asserts a recalibration flag.

For example, a large temperature drift can cause the DPLL to lose its lock and require recalibration. When the DPLL locks at a temperature within the 080 degrees Celsius range, the maximum temperature drift is approximately 55 degrees Celsius. When DPLL starts at a negative temperature, the maximum temperature drift is higher.

If the DPLL locks at 30 degrees Celsius, the temperature can change by 60 degrees Celsius (from 30 to +90 degrees Celsius) and the DPLL does not lose the lock. However, for temperatures above the 60 degrees Celsius range, the DPLL may need to be relocked. A new relock sequence reinitializes the starting temperature.

This compensation mechanism is active only while the DPLL is locked. When the DPLL is in off or bypass mode (low-power or fast-relock), it does not assert the recalibration flag. If the voltage or temperature exceeds the drift limits while the DPLL is not locked, and then the DPLL tries to relock, the DPLL fails to lock within the normal delay and recalibrates automatically before eventually locking. The only difference from a standard relock is the delay.

The DPLL can automatically start recalibration when the recalibration flag is asserted, or recalibration can be managed by the software. The mode of operation is selected by configuring the corresponding registers in the PRCM module (see Table 4-42). The software or manual control mode is selected by default.

Note: Automatic recalibration of the DPLL can start at any time. While relocking, the DPLL switches to bypass mode. For modules that are sensitive to frequency change while operating, this can introduce operational instability. For example, the SDRC is sensitive to a frequency change on DPLL3 because its embedded DLL relocks on a frequency change. Any access during this DLL relock period can be corrupted. It is important; therefore, to stall SDRC access during DPLL recalibration.

To let the software recalibrate the DPLL at the correct time depending on the device activity, the PRCM module can generate a wake-up event on the MPU power domain, followed by an interrupt on the MPU when the DPLL recalibration flag is asserted.

Table 4-42 summarizes software programming control over the DPLL recalibration feature.

Table 4-42. DPLL Recalibration Controls

DPLL	Software Control	Description
DPLL1 (MPU)	PRCM.CM_CLKEN_PLL_MPU[3] MPU_DPLL_DRIFTGUARD	Enable/disable the MPU DPLL automatic recalibration feature.
	PRCM.PRM_IRQENABLE_MPU[7] MPU_DPLL_RECAL_EN	Enable/disable the MPU DPLL recalibration interrupt to MPU.
	PRCM.PRM_IRQSTATUS_MPU[7] MPU_DPLL_ST	Status of the MPU DPLL recalibration interrupt
DPLL2 (IVA2)	PRCM.CM_CLKEN_PLL_IVA2[3] IVA2_DPLL_DRIFTGUARD	Enable/disable the IVA2 DPLL automatic recalibration feature.
	PRCM.PRM_IRQENABLE_MPU[8] IVA2_DPLL_RECAL_EN	Enable/disable the IVA2 DPLL recalibration interrupt to MPU.
	PRCM.PRM_IRQSTATUS_MPU[8] IVA2_DPLL_ST	Status of the IVA2 DPLL recalibration interrupt to MPU
	PRCM.PRM_IRQENABLE_IVA2[2] IVA2_DPLL_RECAL_EN	Enable/disable the IVA2 DPLL recalibration interrupt to IVA2.2.
	PRCM.PRM_IRQSTATUS_IVA2[2] IVA2_DPLL_ST	Status of the IVA2 DPLL recalibration interrupt to IVA2.2
DPLL3 (CORE)	PRCM.CM_CLKEN_PLL[3] EN_CORE_DPLL_DRIFTGUARD	Enable/disable the CORE DPLL automatic recalibration feature.
	PRCM.PRM_IRQENABLE_MPU[5] CORE_DPLL_RECAL	Enable/disable the CORE DPLL recalibration interrupt to MPU.
	PRCM.PRM_IRQSTATUS_MPU[5] CORE_DPLL_ST	Status of the CORE DPLL recalibration interrupt
DPLL4 (PER)	PRCM.CM_CLKEN_PLL[19] EN_PERIPH_DPLL_DRIFTGUARD	Enable/disable the PER DPLL automatic recalibration feature.
	PRCM.PRM_IRQENABLE_MPU[6] PERIPH_DPLL_RECAL	Enable/disable the PER DPLL recalibration interrupt to MPU.
	PRCM.PRM_IRQSTATUS_MPU[6] PERIPH_DPLL_ST	Status of the PER DPLL recalibration interrupt
DPLL5 (PER2)	PRCM.CM_CLKEN2_PLL[3] EN_PERIPH2_DPLL_DRIFTGUARD	Enable/disable the PER DPLL2 automatic recalibration feature.
	PRCM.PRM_IRQENABLE_MPU[25] SND_PERIPH_DPLL_RECAL_EN	Enable/disable the PER DPLL2 recalibration interrupt to MPU.
	PRCM.PRM_IRQSTATUS_MPU[25] SND_PERIPH_DPLL_ST	Status of the PER DPLL2 recalibration interrupt

Note: DPLL recalibration is not necessary in real use (specified operating voltage and temperature range).

4.7.6.6 DPLL Programming Sequence

The DPLL programming sequence follows:

1. Set the M and N values for the desired CLKOUT frequency (see Section 4.7.6.1).

2. Set the corresponding output dividers (M2, M3, M4, M5, and M6) (see Section 4.7.6.1).
3. Enable/disable the autorecalibration feature (see Section 4.7.6.5).
4. Enable/disable the autoidle feature (see Section 4.7.6.2).
5. Mask/unmask the interrupt to the MPU (and the DPLL2 interrupt to IVA2) (see Section 4.7.6.5).
6. Enable the DPLL lock mode (see Section 4.7.6.2).

4.7.7 Internal Clock Controls

This section describes the software and hardware controls of the internal source clocks. Figure 4-56 through Figure 4-69 list the clock controls. The Source selection/division column lists the PRCM register bits that select or divide the clocks. The Software control column lists the PRCM register bits that enable/disable the clocks. The Hardware control column lists the hardware conditions required to effectively gate the clocks.

In the Hardware control column, the boxes labeled CL, GS, GC, and HC indicate specific information about the hardware clock controls:

- CL (combinational logic): The functional or interface clock is required by more than one module across more than one power domain. The gating control is the OR combination of all the domain clock requests. If any module of this clock domain requests the clock, the clock is not gated.
- GS (gating selection): The clock is selectable among several possible source clocks for a module. The gating control depends on the software programming of the CM_CLKSEL_<domain_name> type of register. The clock request of the module or domain must be set by the CM_CLKSEL bit.
- GC (gating control): The functional/interface clock is required by a single module across the power domain. The gating control depends only on the software programming of the FCLKEN/ICLKEN bit. For the interface clock, the enable bit is effective only if autoidle mode is not used. If autoidle mode is used, the gating control also depends on the state of the power domain.
- HC (hardware control): A specific rule not covered by CL, GS, or GC.

Note: Because the PRCM module must receive hardware acknowledgement from the different modules before it can gate the clock, the clock is not gated immediately after the software requests clock-gating conditions.

4.7.7.1 PRM Source-Clock Controls

Figure 4-56 shows the common source-clock controls for the PRM.

Figure 4-56. Common PRM Source-Clock Controls

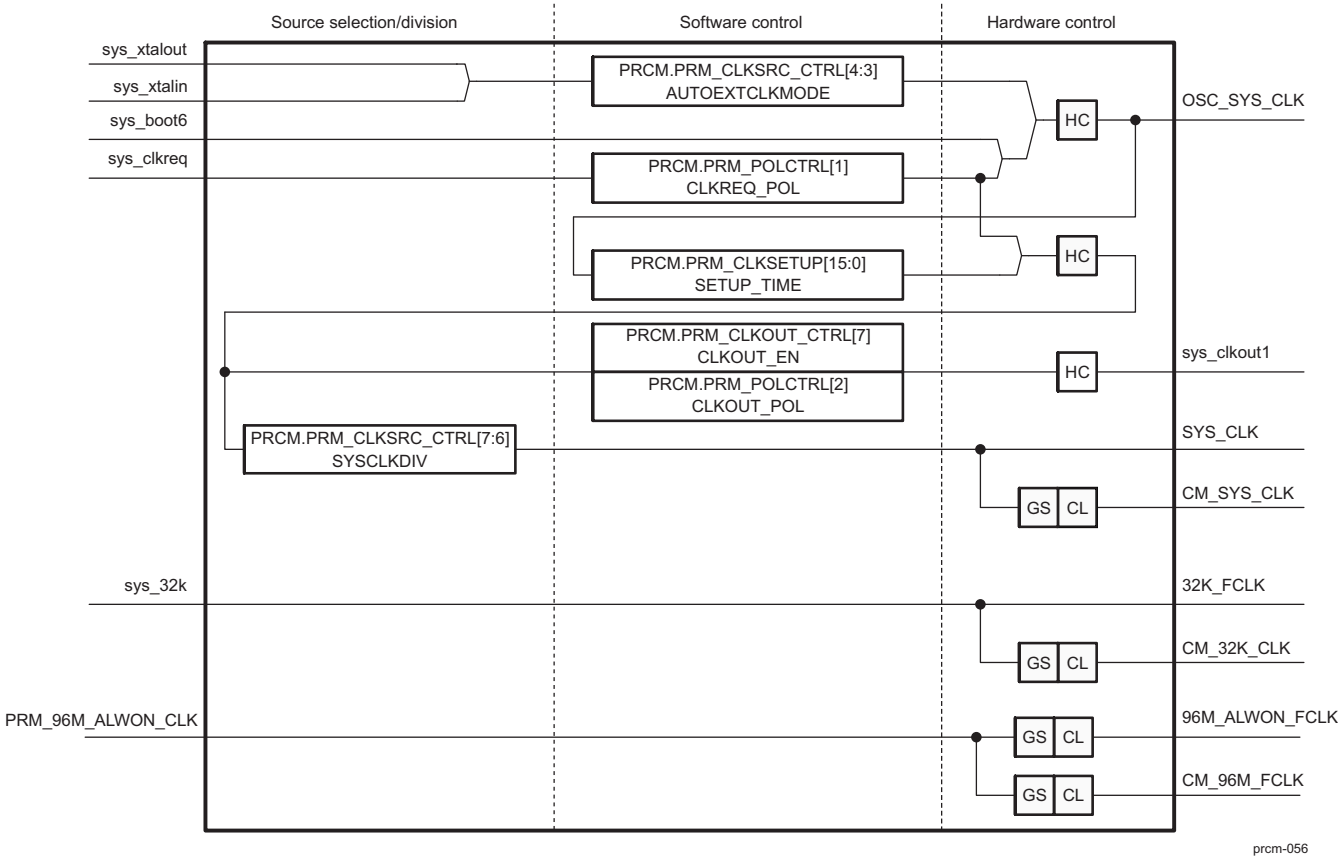


Table 4-43 shows the common source-clock gating controls for the PRM.

Table 4-43. Common PRM Source-Clock Gating Controls

Clock Name	Reset	Clock-Gating Control	Gating Description
OSC_SYS_CLK	Running	PRCM.PRM_CLKSRC_CTRL[4:3] AUTOEXTCLKMODE and device power state and sys_clkreq	Gated when the oscillator is programmed to power down with the device sleep/retention/off transition
sys_clkout1	Running	PRCM.PRM_CLKOUT_CTRL[7] CLKOUT_EN and PRCM.PRM_POLCTRL[2] CLKOUT_POL and sys_clkreq	Active when OSC_SYS_CLK is active, sys_clkout1 is enabled, and sys_clkreq is asserted
SYS_CLK	Running	Activated after clksetup_count_overflow	Active when OSC_SYS_CLK is active and the SYS_CLK setup time is up
CM_SYS_CLK	Running	PRCM.CM_CLKSEL_CORE[6] CLKSEL_GPT10, PRCM.CM_CLKSEL_CORE[7] CLKSEL_GPT11, and depends on the clock-gating conditions of GPT10_FCLK and GPT11_FCLK	Active if it is the source clock of the GPT10_FCLK or GPT11_FCLK and the functional clock is active
32K_FCLK	Running	None	Always-active clock from the sys_32k input pin
CM_32K_CLK	Running	PRCM.CM_CLKSEL_CORE[6] CLKSEL_GPT10, PRCM.CM_CLKSEL_CORE[7] CLKSEL_GPT11, and depends on the clock-gating conditions of GPT10_FCLK and GPT11_FCLK	Active if it is the source clock of the GPT10_FCLK or GPT11_FCLK and the functional clock is active

Table 4-43. Common PRM Source-Clock Gating Controls (continued)

Clock Name	Reset	Clock-Gating Control	Gating Description
CM_96M_FCLK	Gated	CM_CLKSEL1_PLL[3] SOURCE_48M bit cleared, and depends on the clock-gating condition of 96M_FCLK, 48M_FCLK, and 12M_FCLK	Active if the derived clocks (96M_FCLK, 48M_FCLK, and 12M_FCLK) are active
96M_ALWON_FCLK	Gated	CM_FCLKEN_PER[0] EN_MCBSP2, CM_FCLKEN_PER[1] EN_MCBSP3, and CM_FCLKEN_PER[2] EN_MCBSP4	Gated when none of the three McBSPs [2..4] have their functional clock enable requested

The oscillator output clock (OSC_SYS_CLK) is gated when the PRCM.PRM_CLKSRC_CTRL[4:3] AUTOEXTCLKMODE bit field is programmed to power down the oscillator when the device enters retention or off mode. In this condition, all the clock trees in the device must be gated, and the four DPLLs (DPLL1, DPLL2, DPLL3, and DPLL4) must enter stop mode before this transition can occur.

SYS_CLK is gated under the same conditions as the oscillator output clock, but it is enabled only after the oscillator stabilizes. Oscillator stabilization is determined by a counter overflow configured in the PRCM.PRM_CLKSETUP[15:0] SETUP_TIME bit field.

The sys_clkreq active condition is described in Section 4.7.5, *External Clock Control*.

4.7.7.2 CM Source-Clock Controls

Figure 4-57 shows the common source-clock controls for the CM.

Figure 4-57. Common CM Source-Clock Controls

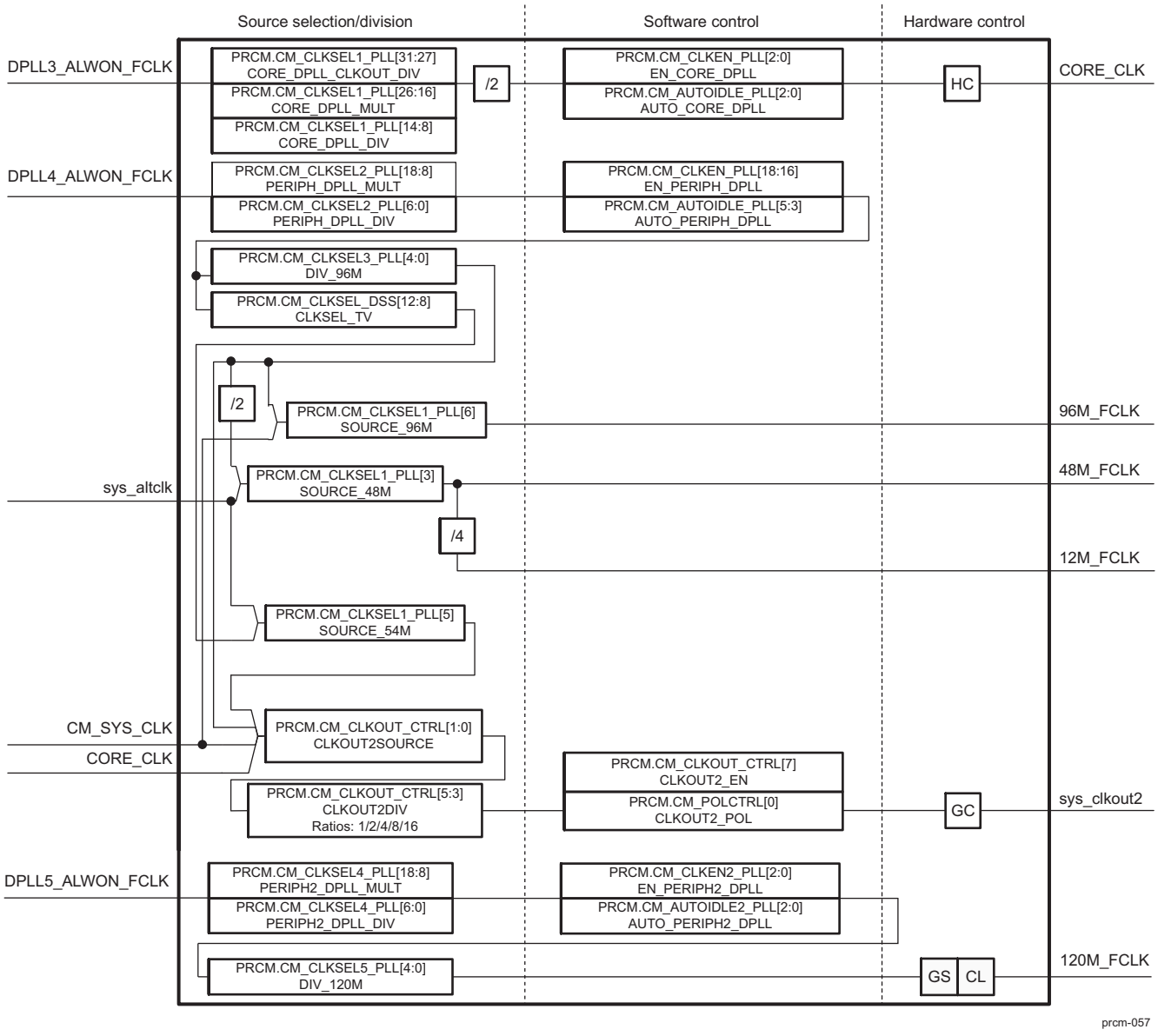


Table 4-44 shows the common source-clock gating controls for the CM.

Table 4-44. Common CM Source-Clock Gating Controls

Clock Name	Reset	Clock-Gating Control	Gating Description
CORE_CLK	Running	Depends on the clock-gating conditions of: L3_ICLK and L4_ICLK	Gated when all interface clocks of the different modules of the device are gated
96M_FCLK	Stopped	Depends on the clock-gating conditions of CORE_96M_FCLK and DSS_96M_FCLK	If the dependent clocks are active, the clock is active.
48M_FCLK	Stopped	Depends on the clock-gating conditions of CORE_12M_FCLK, PER_48M_FCLK and USBHOST_48M_FCLK	If the dependent clocks are active, the clock is active.
12M_FCLK	Stopped	Depends on the clock-gating conditions of CORE_12M_FCLK	If the dependent clock is active, the clock is active.
sys_clkout2	Stopped	PRCM.CM_CLKOUT_CTRL[7] CLKOUT2_EN	Active if enabled
DPLL4_M2_CLK	Stopped	Depends on the clock-gating conditions of: 96M_FCLK	If the dependent clocks are active, the clock is active.
DPLL4_M3_CLK	Stopped	CM_CLKSEL1_PLL.SOURCE_54M and depends on the clock-gating conditions of: DPLL4_M2_CLK and DSS_TV_CLK	If the dependent clocks are active, the clock is active. DSS_TV_CLK is a dependent clock if set by the register configuration.
120M_FCLK	Stopped	CM_FCLKEN_USBHOST[1] EN_USBHOST2, CM_FCLKEN3_CORE[2] EN_USBTLL	If any of the dependent clocks (CORE_120M_FCLK or USBHOST_12M_FCLK) is active, the clock is active.

4.7.7.3 Common Interface Clock Controls

Figure 4-58 shows the clock controls for the common interface.

Figure 4-58. Common Interface Clock Controls

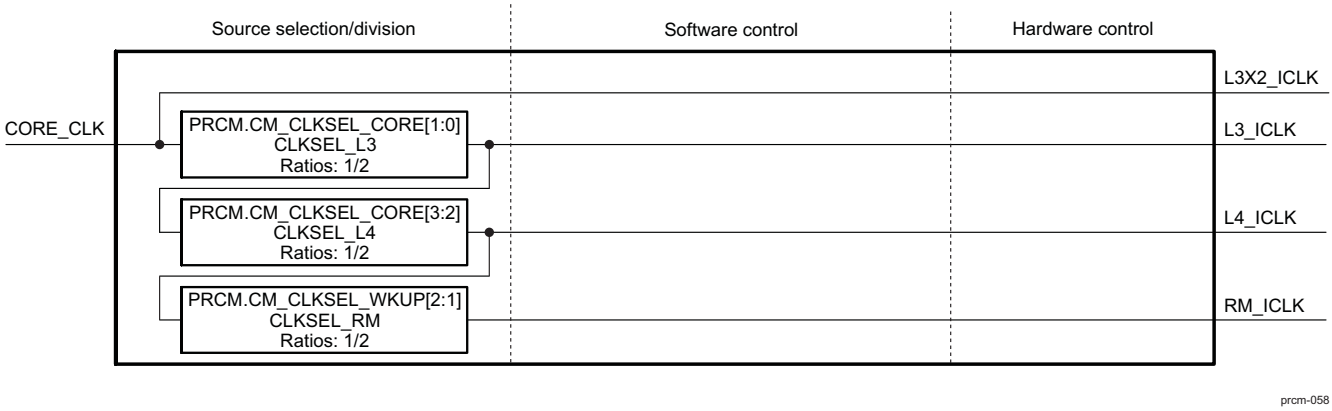


Table 4-45 shows the clock-gating controls for the common interface.

Table 4-45. Common Interface Clock-Gating Controls

Clock Name	Reset	Clock-Gating Control	Gating Description
L3X2_ICLK	Running	CORE_CLK gating conditions	Depends on the gating conditions of the CORE_CLK
L3_ICLK	Running	Depends on the clock-gating conditions of GFX_L3_ICLK, CORE_L3_ICLK, and CAM_L3_ICLK	Gated when all L3 interface clocks of the different modules of the device are gated
L4_ICLK	Running	Depends on the clock-gating conditions of CORE_L4_ICLK, CAM_L4_ICLK, DSS_L4_ICLK, PER_L4_ICLK, and WKUP_L4_ICLK	Gated when all L4 interface clocks of the different modules of the device are gated
RM_ICLK	Running	None	Gated with source clock (CORE_CLK)

4.7.7.4 DPLL Source-Clock Controls

Figure 4-59 shows the clock controls for the DPLL power domain.

Figure 4-59. DPLL Power Domain Clock Controls

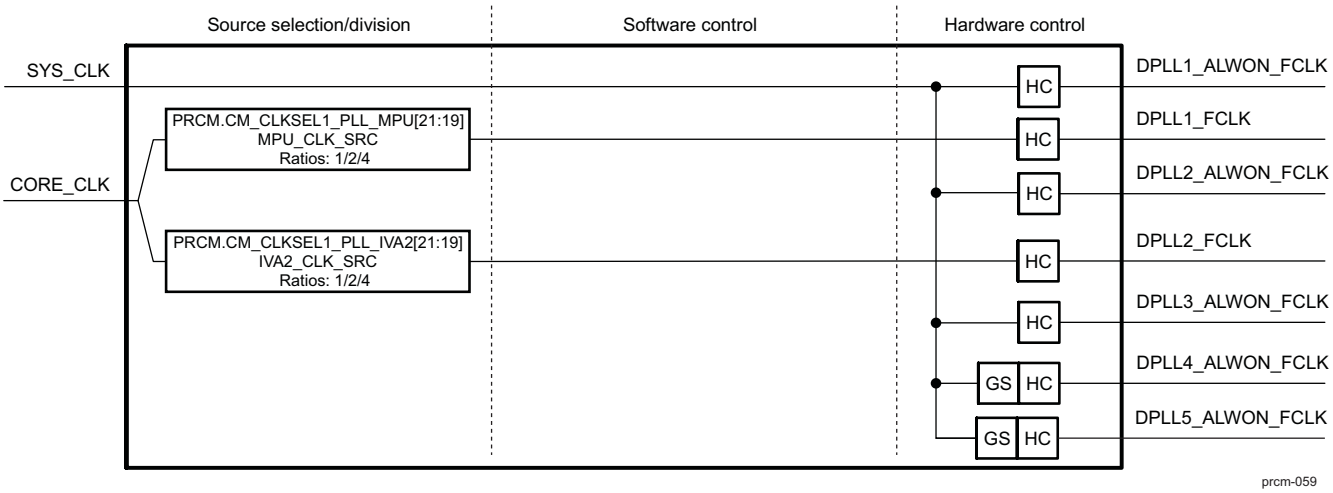


Table 4-46 shows the clock-gating controls for the DPLL power domain.

Table 4-46. DPLL Power Domain Clock-Gating Controls

Clock Name	Reset	Clock-Gating Control	Gating Description
DPLL1_ALWON_FCLK	Running	PRCM.CM_AUTOIDLE_PLL_MPU[2:0] AUTO_MPU_DPLL, PRCM.CM_CLKEN_PLL_MPU[2:0] EN_MPU_DPLL, and MPU domain power state	Gated if the DPLL is set to automatic active control and enabled in lock mode while the MPU power domain goes into retention or off mode. Also gated if DPLL is set to low-power bypass mode.
DPLL1_FCLK	Stopped		
DPLL2_ALWON_FCLK	Stopped	PRCM.CM_AUTOIDLE_PLL_IVA2[2:0] AUTO_IVA2_DPLL, PRCM.CM_CLKEN_PLL_IVA2[2 :0] EN_IVA2_DPLL, and IVA2 power domain power state	Gated if the DPLL is set to automatic active control and enabled in lock mode while the IVA2 power domain goes into retention or off mode. Also gated if DPLL is set to low-power stop or bypass mode.
DPLL2_FCLK	Stopped		
DPLL3_ALWON_FCLK	Running	PRCM.CM_AUTOIDLE_PLL[2:0] AUTO_CORE_DPLL, PRCM.CM_CLKEN_PLL[2:0] EN_CORE_DPLL, and CORE domain power clocks state	Gated if the DPLL is set to automatic active control and enabled in lock mode while the CORE power domain is idle. Also gated if DPLL is set to low-power or fast-relock bypass mode.
DPLL4_ALWON_FCLK	Stopped	PRCM.CM_AUTOIDLE_PLL[5:3] AUTO_PERIPH_DPLL, PRCM.CM_CLKEN_PLL[18:16] EN_PERIPH_DPLL, and depends on the clock-gating conditions of DPLL4_M2_CLK	Gated if the DPLL is set to automatic active control and enabled in lock mode while its dependent clock is inactive. Also gated if DPLL is set to low-power stop mode.
DPLL5_ALWON_FCLK	Stopped	PRCM.CM_AUTOIDLE2_PLL[2:0] AUTO_PERIPH2_DPLL, PRCM.CM_CLKEN2_PLL[2:0] EN_PERIPH2_DPLL, and depends on the clock-gating conditions of DPLL5_M2_CLK	Gated if the DPLL is set to automatic active control and enabled in lock mode while its dependent clock is inactive. Also gated if DPLL is set to low-power stop mode.

4.7.7.5 SGX Power Domain Clock Controls

This section describes all modules and features in the high-tier device. To check availability of modules and features, see Section 1.5, *OMAP34xx Family*. To save power, ensure that power domains of unavailable features and modules are switched off and clocks are cut off.

Figure 4-60 shows the clock controls for the SGX power domain.

Figure 4-60. SGX Power Domain Clock Controls

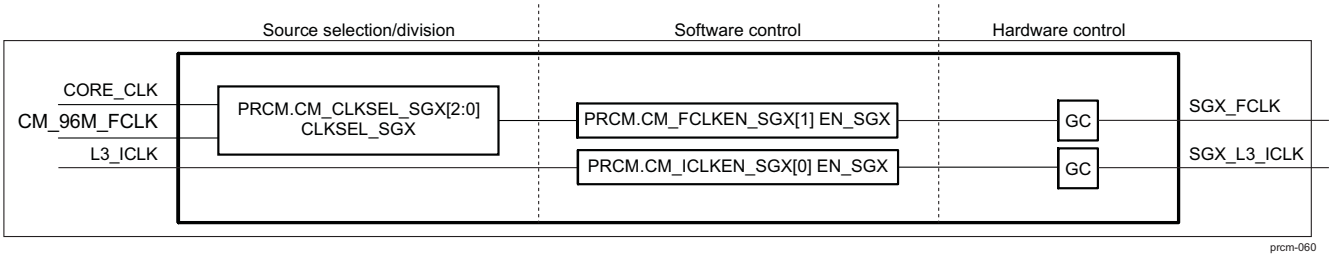


Table 4-47 lists the clock-gating controls for the SGX power domain.

Table 4-47. SGX Power Domain Clock-Gating Controls

Clock Name	Reset	Clock-Gating Control	Gating Description
SGX_FCLK	Stopped	PRCM.CM_FCLKEN_SGX[1] EN_SGX	Gated when the enable bit is set to 0
SGX_ICLK	Stopped	PRCM.CM_ICLKEN_SGX[1] EN_SGX	Gated when the enable bit is set to 0

Note: SGX clock control is available only on OMAP3430.

4.7.7.6 CORE Power Domain Clock Controls

Figure 4-61 through Figure 4-62 show the clock controls for the CORE power domain.

Figure 4-61. CORE Power Domain Clock Controls: Part 1

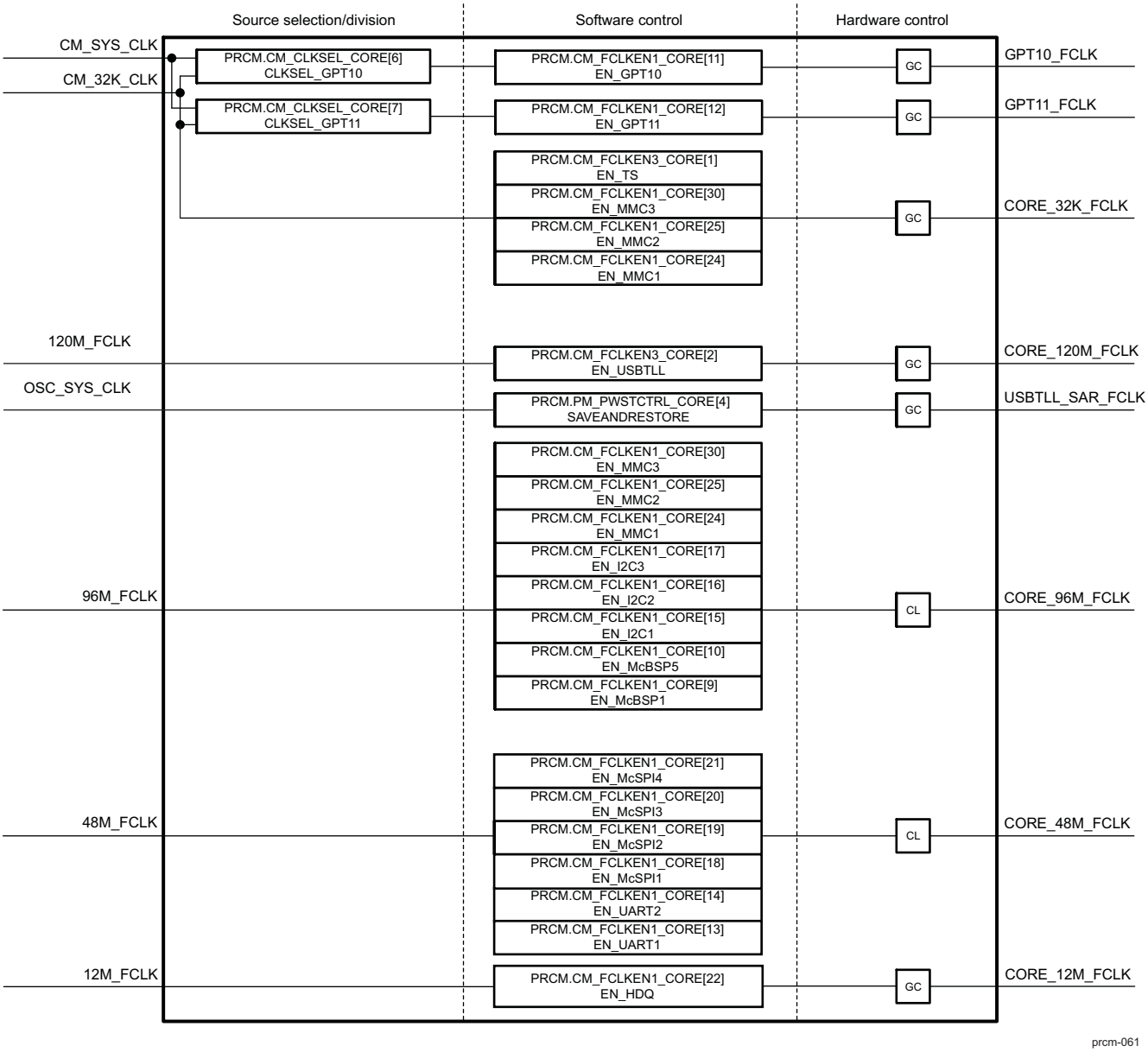


Figure 4-62. CORE Power Domain Clock Controls: Part 2

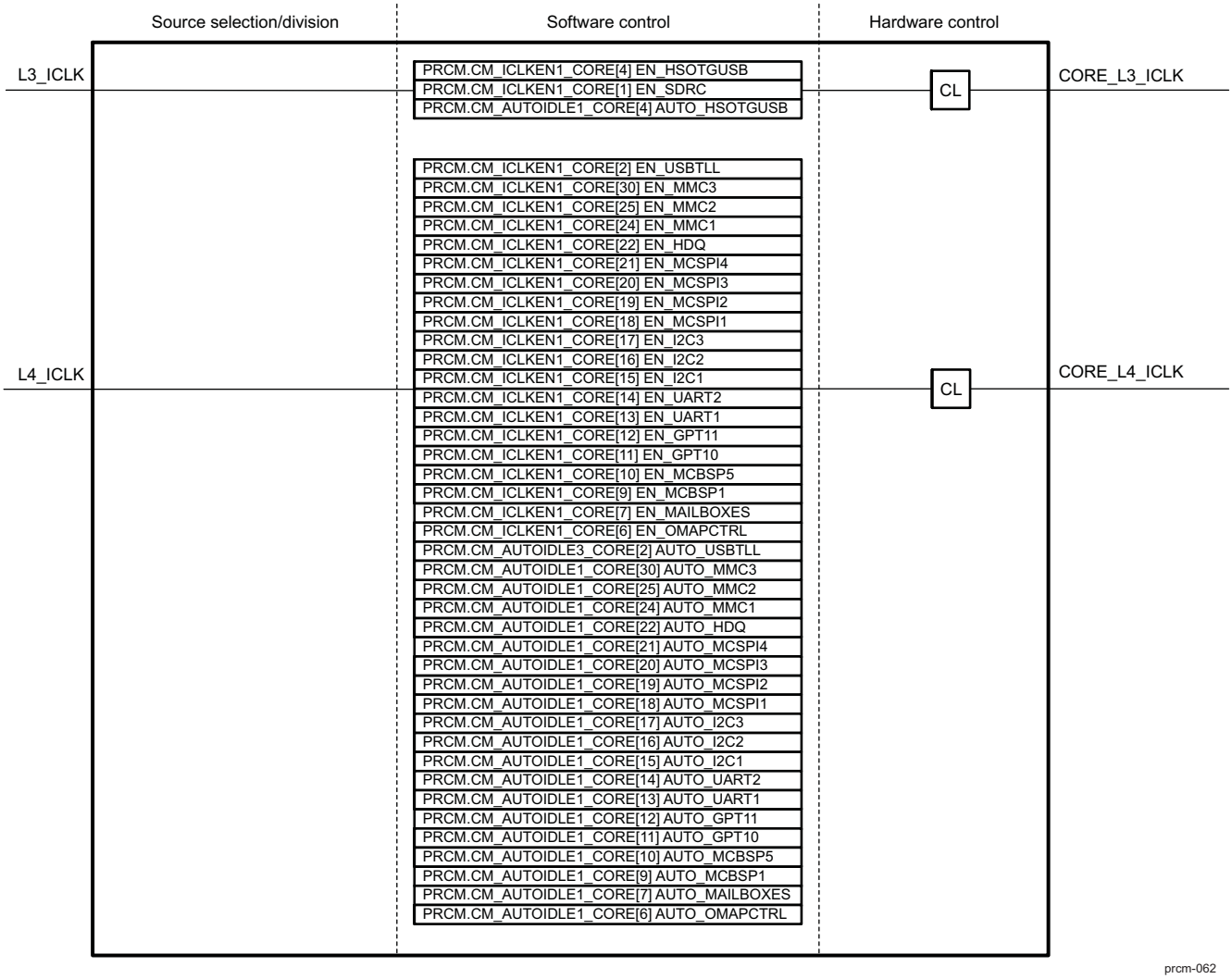


Table 4-48 lists the clock-gating controls for the CORE power domain.

Table 4-48. CORE Power Domain Clock-Gating Controls

Clock Name	Reset	Clock-Gating Control	Gating Description
GPT10_FCLK	Stopped	PRCM.CM_FCLKEN1_CORE[11] EN_GPT10	Gated when the enable bit is set to 0
GPT11_FCLK	Stopped	PRCM.CM_FCLKEN1_CORE[12] EN_GPT11	Gated when the enable bit is set to 0
CORE_96M_FCLK	Stopped	McBSP[1..5] input clock source select (in SCM) and PRCM.CM_FCLKEN1_CORE (MMC[1-2], McBSP[1, 5], I2C[1-3])	Gated when the enable bits of the module functional clock are set to 0. (The McBSPs can have MCBSP_CLKS as an alternate functional clock.)
CORE_48M_FCLK	Stopped	PRCM.CM_FCLKEN1_CORE (UART[1-2], McSPI[1-4])	Gated when the functional clock enable bits of the module are set to 0
CORE_12M_FCLK	Stopped	PRCM.CM_FCLKEN1_CORE[22] EN_HDQ	Gated when the enable bit is set to 0

Table 4-48. CORE Power Domain Clock-Gating Controls (continued)

Clock Name	Reset	Clock-Gating Control	Gating Description
CORE_L3_ICLK	Running	PRCM.CM_ICLKEN1_CORE EN_(SDRC, HSOTGUSB), PRCM.CM_AUTOIDLE1_CORE4] AUTO_HSOTGUSB	Gated when: • All enable bits are set to 0. • The enable-autoidle bit pair is set to 1, the remaining enable bits are set to 0, and the clock is not requested by any module.
CORE_L4_ICLK	Running	PRCM.CM_ICLKEN1_CORE (MMC[1..2], HDQ, MCSPI[1-4], I2C[1-3], UART[1,2], GPT[10,11], MCBSP[1,5], MAILBOXES, OMAPCTRL) and PRCM.CM_AUTOIDLE1_CORE (MMC[1..2], HDQ, MCSPI[1-4], I2C[1-3], UART[1,2], GPT[10,11], MCBSP[1,5], MAILBOXES, OMAPCTRL)	Gated when: • All enable bits are set to 0. • All enable-autoidle bit pairs are set to 1, and the clock is not requested by any module.
CORE_32K_FCLK	Stopped	CM_FCLKEN1_CORE[24] EN_MMC1, CM_FCLKEN1_CORE[25] EN_MMC2, CM_FCLKEN1_CORE[30] EN_MMC3, CM_FCLKEN3_CORE[1] EN_TS	Gated when all enable bits are set to 0
USBTLL_SAR_FCLK	Stopped	CORE power domain power state and PM_PWSTCTRL_CORE[4] SAVEANDRESTORE	Gated when the save-restore bit is set to 0, or when the CORE power domain is in off state after the save operation completes or in on state after the restore operation completes
CORE_120M_FCLK	Stopped	CM_FCLKEN3_CORE[0] EN_USBTLL and DPLL5 operating mode	Gated when the enable bit is set to 0, or the DPLL5 is in stop or bypass mode

4.7.7.7 EFUSE Power Domain Clock Controls

Figure 4-63 shows the clock controls for the EFUSE power domain. Table 4-49 lists the clock-gating control for the EFUSE power domain.

Figure 4-63. EFUSE Power Domain Clock Controls

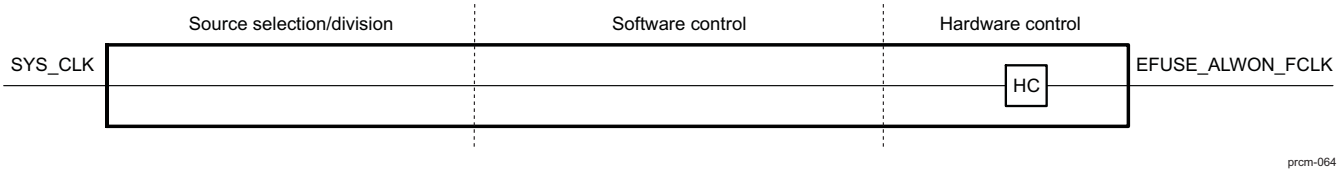


Table 4-49. EFUSE Power Domain Clock-Gating Control

Clock Name	Reset	Clock-Gating Control	Gating Description
EFUSE_ALWON_FCLK	Running	None	Active when VDD1 and VDD2 are switched on and eFuse-ready hardware signal is released

4.7.7.8 DSS Power Domain Clock Controls

This section describes all modules and features in the high-tier device. To check availability of modules and features, see Section 1.5, *OMAP34xx Family*. To save power, ensure that power domains of unavailable features and modules are switched off and clocks are cut off.

Figure 4-64 shows the clock controls for the DSS power domain. Table 4-50 lists the clock-gating controls for the DSS power domain.

Figure 4-64. DSS Power Domain Clock Controls

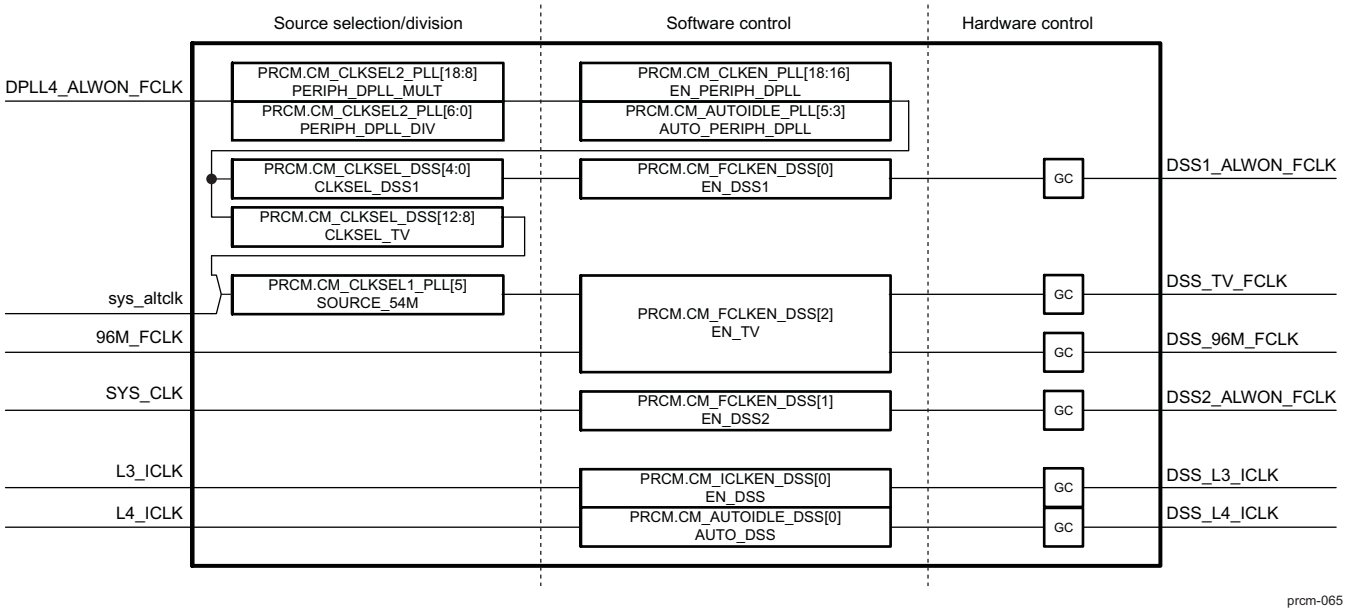


Table 4-50. DSS Power Domain Clock-Gating Controls

Clock Name	Reset	Clock-Gating Control	Gating Description
DSS1_ALWON_FCLK	Stopped	PRCM.CM_FCLKEN_DSS[0] EN_DSS1	Gated when the enable bit is set to 0
DSS2_ALWON_FCLK	Stopped	PRCM.CM_FCLKEN_DSS[1] EN_DSS2	Gated when the enable bit is set to 0
DSS_TV_FCLK	Stopped	PRCM.CM_FCLKEN_DSS[2] EN_TV	Gated when the enable bit is set to 0
DSS_96M_FCLK	Stopped		
DSS_L3_ICLK	Stopped	PRCM. CM_ICLKEN_DSS[0] EN_DSS, PRCM.CM_AUTOIDLE_DSS[0] AUTO_DSS	Gated when: • Enable bit is set to 0. • Enable-autoidle bit pair is set to 1, and the clock is not requested by any module.
DSS_L4_ICLK	Stopped		

4.7.7.9 CAM Power Domain Clock Controls

This section describes all modules and features in the high-tier device. To check availability of modules and features, see Section 1.5, *OMAP34xx Family*. To save power, ensure that power domains of unavailable features and modules are switched off and clocks are cut off.

Figure 4-65 shows the clock controls for the CAM power domain. Table 4-51 lists the clock-gating controls for the CAM power domain.

Figure 4-65. CAM Power Domain Clock Controls

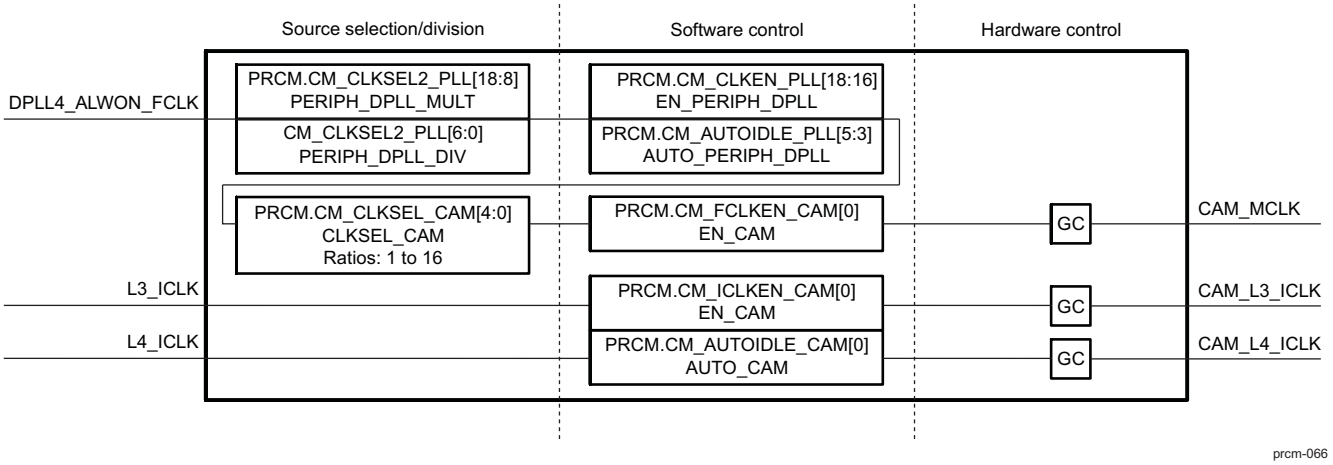


Table 4-51. CAM Power Domain Clock-Gating Controls

Clock Name	Reset	Clock-Gating Control	Gating Description
CAM_MCLK	Stopped	PRCM.CM_FCLKEN_CAM[0] EN_CAM	Gated when the enable bit is set to 0
CAM_L3_ICLK	Stopped	PRCM.CM_ICLKEN_CAM[0] EN_CAM, PRCM.CM_AUTOIDLE_CAM[0] AUTO_CAM	Gated when: • Enable bit is set to 0. • Enable-autoidle bit pair is set to 1, and the clock is not requested by subsystem.
CAM_L4_ICLK	Stopped		

4.7.7.10 USBHOST Power Domain Clock Controls

Figure 4-66 shows the clock controls for the USBHOST power domain. Table 4-52 lists the clock-gating controls for the USBHOST power domain.

Figure 4-66. USBHOST Power Domain Clock Controls

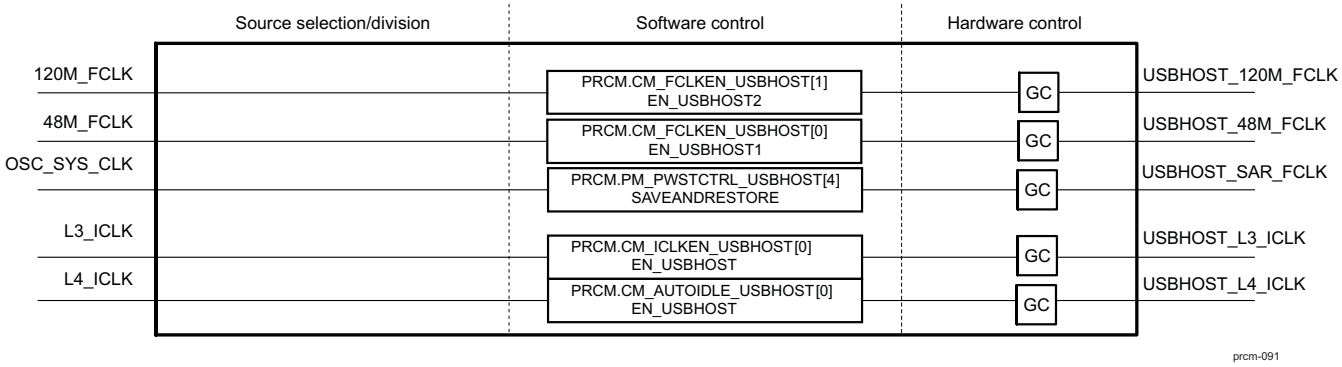


Table 4-52. USBHOST Power Domain Clock-Gating Controls

Clock Name	Reset	Clock-Gating Control	Gating Description
USBHOST_48M_FCLK	Stopped	PRCM.CM_FCLKEN_USBHOST[0] EN_USBHOST1	Gated when the enable bit is set to 0
USBHOST_120M_FCLK	Stopped	PRCM.CM_FCLKEN_USBHOST[1] EN_USBHOST2	Gated when the enable bit is set to 0

Table 4-52. USBHOST Power Domain Clock-Gating Controls (continued)

Clock Name	Reset	Clock-Gating Control	Gating Description
USBHOST_L3_ICLK	Stopped	PRCM.CM_ICLKEN_USBHOST[0] EN_USBHOST, PRCM.CM_AUTOIDLE_USBHOST[0] AUTO_USBHOST	Gated when: • Enable bit is set to 0. • Enable-autoidle bit pair is set to 1, and the clock is not requested by subsystem.
USBHOST_L4_ICLK	Stopped		
USBHOST_SAR_FCLK	Stopped	PRCM.PM_PWSTCTRL_USBHOST[4] SAVEANDRESTORE	Gated when the save-restore bit is set to 0, or when the power domain is in off state after the save operation completes or in on state after the restore operation completes.

4.7.7.11 WKUP Power Domain Clock Controls

Figure 4-67 shows the clock controls for the WKUP power domain. Table 4-53 lists the clock-gating controls for the WKUP power domain.

Figure 4-67. WKUP Power Domain Clock Controls

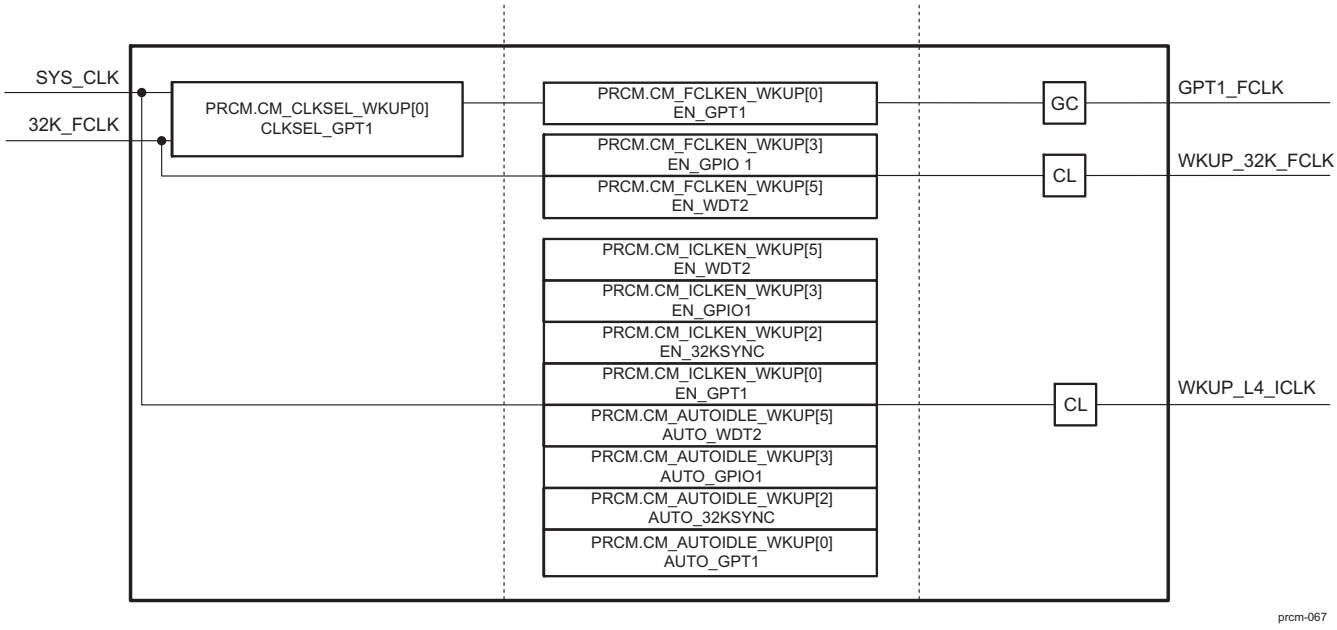


Table 4-53. WKUP Power Domain Clock-Gating Controls

Clock Name	Reset	Clock-Gating Control	Gating Description
GPT1_FCLK	Stopped	PRCM.CM_FCLKEN_WKUP[0] EN_GPT1	Gated when the enable bit is set to 0
WKUP_32K_FCLK	Stopped	PRCM.CM_FCLKEN_WKUP[3] GPIO1 and PRCM.CM_FCLKEN_WKUP[5] WDT2	Gated when the enable bits are set to 0
WKUP_L4_ICLK	Running	PRCM.CM_ICLKEN_WKUP EN_(WDT2, GPIO1, 32KSYNC, GPTIMER1), PRCM.CM_AUTOIDLE_WKUP AUTO_(WDT2, GPIO1, 32KSYNC, and GPTIMER1)	Gated when: • All enable bits are set to 0. • All enable-autoidle bit pairs are set to 1, and the clock is not requested by any module.

4.7.7.12 PER Power Domain Clock Controls

Figure 4-68 and Figure 4-69 show the clock controls for the PER power domain.

Figure 4-68. PER Power Domain Clock Controls: Part 1

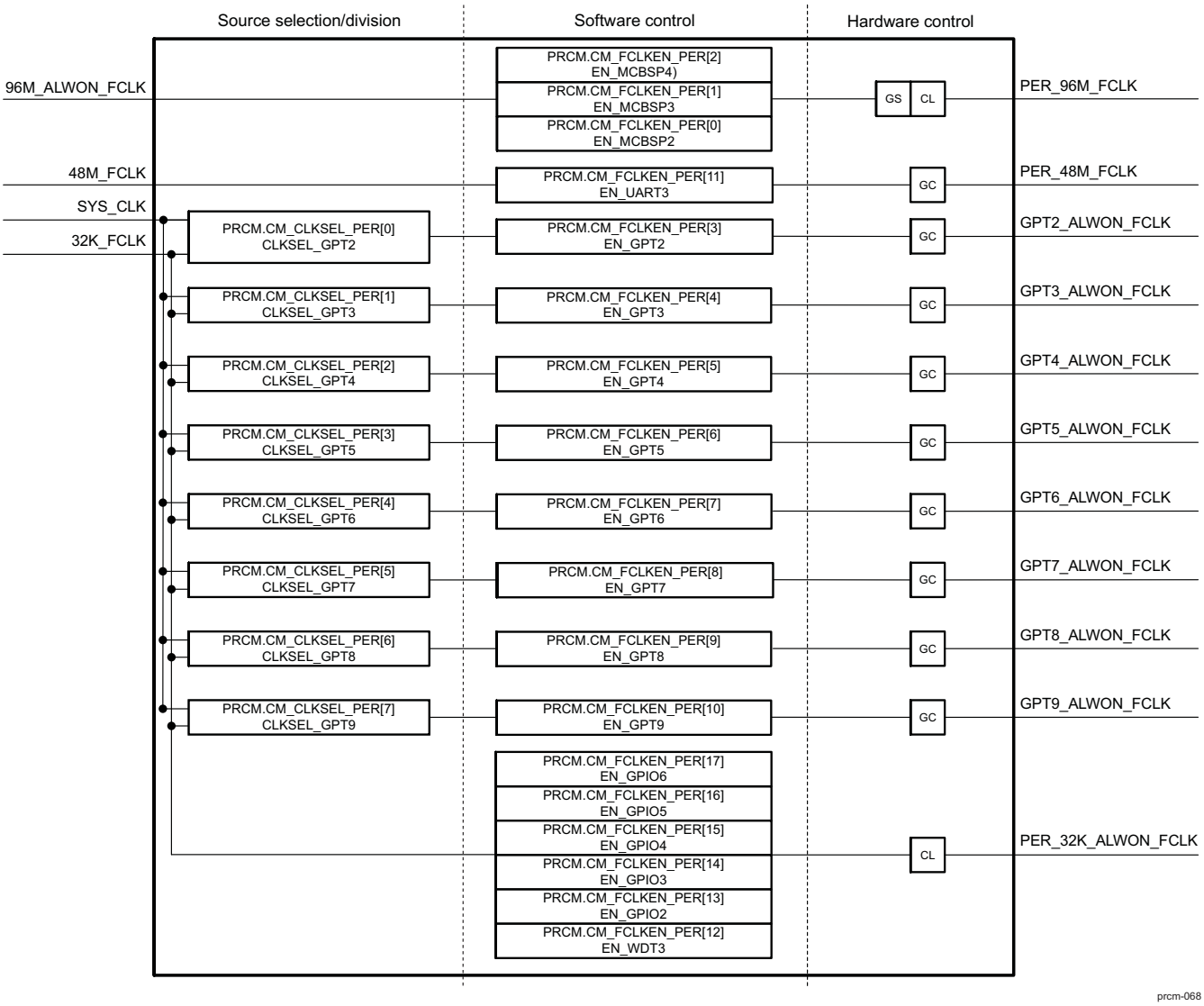


Figure 4-69. PER Power Domain Clock Controls: Part 2

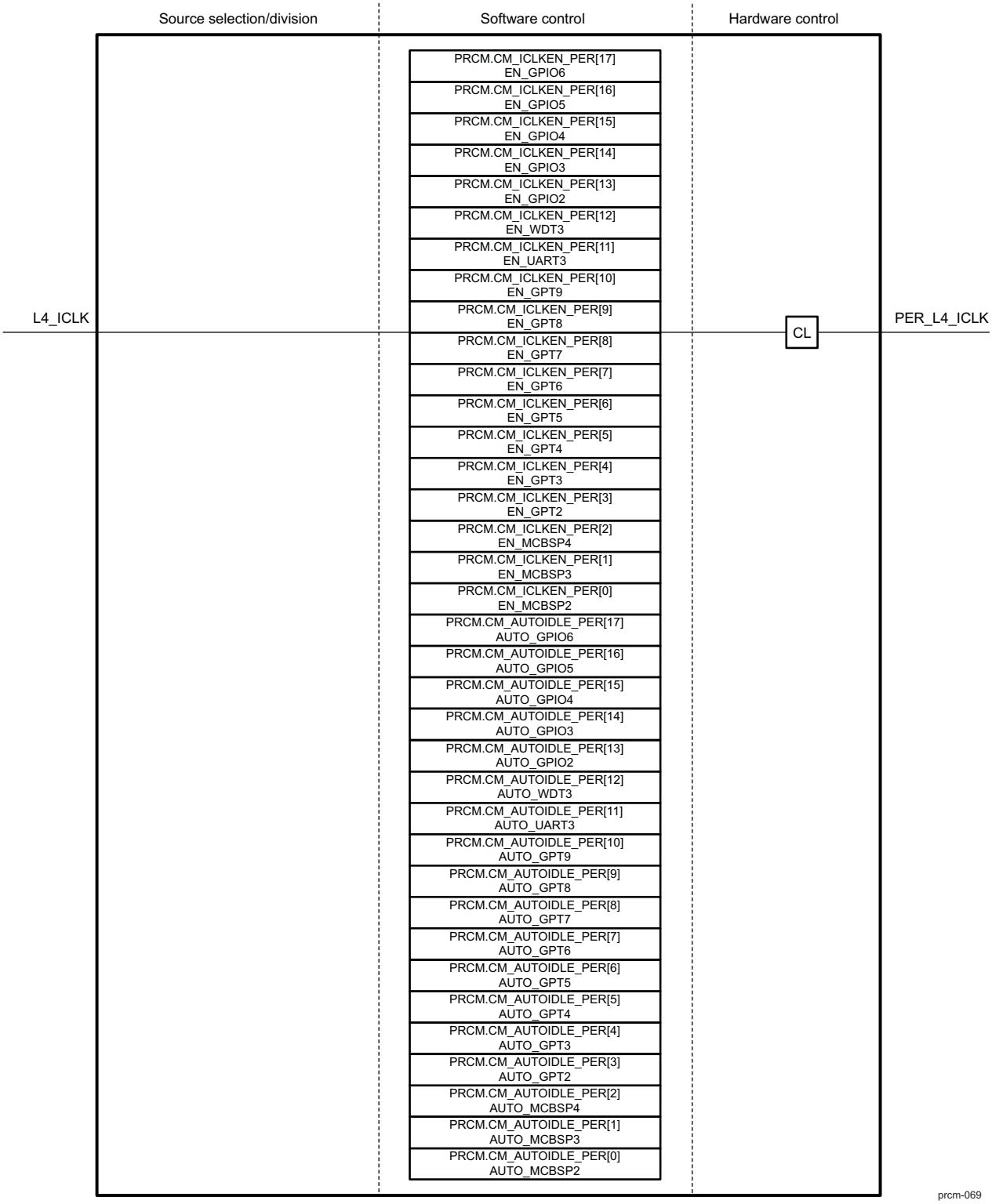


Table 4-54 lists the clock-gating controls for the PER power domain.

Table 4-54. PER Power Domain Clock-Gating Controls

Clock Name	Reset	Clock-Gating Control	Gating Description
PER_48M_FCLK	Stopped	PRCM.CM_FCLKEN_PER[11] EN_UART3	Gated when the enable bit is set to 0
PER_96M_FCLK	Stopped	McBSP[2..4] input clock source select (in SCM) and PRCM.CM_FCLKEN_PER EN_MCBSP[2-4] and the DPLL4 operating mode	Gated when the enable bits of the module functional clock are set to 0 (the McBSPs can have MCBSP_CLKS as an alternate functional clock) or DPLL4 is in stop or bypass mode
MCBSP_CLKS	Stopped	See Chapter 7, *System Control Module*.	
PER_32K_ALWON_ FCLK	Stopped	PRCM.CM_FCLKEN_PER EN_GPIO[2-6] and PRCM.CM_FCLKEN_PER[12] EN_WDT3	Gated when all the enable bits are set to 0
GPT2_ALWON_FCLK	Stopped	PRCM.CM_FCLKEN_PER[3] EN_GPT2	Gated when the enable bit is set to 0
GPT3_ALWON_FCLK	Stopped	PRCM.CM_FCLKEN_PER[4] EN_GPT3	Gated when the enable bit is set to 0
GPT4_ALWON_FCLK	Stopped	PRCM.CM_FCLKEN_PER[5] EN_GPT4	Gated when the enable bit is set to 0
GPT5_ALWON_FCLK	Stopped	PRCM.CM_FCLKEN_PER[6] EN_GPT5	Gated when the enable bit is set to 0
GPT6_ALWON_FCLK	Stopped	PRCM.CM_FCLKEN_PER[7] EN_GPT6	Gated when the enable bit is set to 0
GPT7_ALWON_FCLK	Stopped	PRCM.CM_FCLKEN_PER[8] EN_GPT7	Gated when the enable bit is set to 0
GPT8_ALWON_FCLK	Stopped	PRCM.CM_FCLKEN_PER[9] EN_GPT8	Gated when the enable bit is set to 0
GPT9_ALWON_FCLK	Stopped	PRCM.CM_FCLKEN_PER[10] EN_GPT9	Gated when the enable bit is set to 0
PER_L4_ICLK	Stopped	PRCM.CM_ICLKEN_PER EN_(GPIO[2..6], WDT3, UART3, GPT[2..9], MCBSP[2..4]) and PRCM.CM_AUTOIDLE_PER AUTO_(GPIO[2..6], WDT3, UART3, GPT[2..9], and MCBSP[2..4])	Gated when: • All enable bits are set to 0. • All enable-autoidle bit pairs are set to 1, and the clock is not requested by any module.
96M_ALWON_FCLK	Stopped	None	Always-on clock

For audio applications, the McBSP functional clocks are provided externally by the MCBSP_CLKS pin. This clock must be permanently supplied, to let McBSPs function when the CORE power domain is in the off power state. If the external clock input is selected as the functional clock, the PER power domain sleep transition is prevented.

4.7.8 Clock Configurations

The device supports several clock configurations. A clock configuration is a consistent set of divider ratios programmed into the PRCM module to obtain a certain combination of clock speed to match the performance requirement.

In the device, the MPU and IVA2.2 processors are connected to the interconnects through asynchronous bridges. The functional frequency of these processors can be configured independently of their interface clock frequency.

Therefore, the clock configurations of the device are split into two sections: one for device processor clocks and one for device interface clocks.

An OPP of the device can be defined as a pair of device operating voltage and corresponding frequency. The device processors are in the VDD1 voltage domain and the interface clocks are generated by the CM clock generator in the VDD2 voltage domain. Hence, the OPPs of the processor clocks are identified for the VDD1 voltage levels, independent of the interface clocks, which are associated with the VDD2 voltage levels.

CAUTION

Clock configuration frequencies depend on the device operating voltage values.

4.7.8.1 Processor Clock Configurations

The processor OPPs are identified as the pair of VDD1 operating voltage level and processor clock frequency.

Six generic processor OPPs can be defined as:

1. OPP6 (VDD1 = v4, (MPU_CLK = $f_{mpu}5$, IVA2_CLK = $f_{iva}5$))
2. OPP5 (VDD1 = v4, (MPU_CLK = $f_{mpu}4$, IVA2_CLK = $f_{iva}4$))
3. OPP4 (VDD1 = v3, (MPU_CLK = $f_{mpu}3$, IVA2_CLK = $f_{iva}3$))
4. OPP3 (VDD1 = v2 , (MPU_CLK = $f_{mpu}2$, IVA2_CLK = $f_{iva}2$))
5. OPP2 (VDD1 = v1 , (MPU_CLK = $f_{mpu}1$, IVA2_CLK = $f_{iva}1$))
6. OPP1 (VDD1 = v0 , (MPU_CLK = $f_{mpu}0$, IVA2_CLK = $f_{iva}0$))

where v4 > v3 > v2 > v1 > v0,

$f_{mpu}5 > f_{mpu}4 > f_{mpu}3 > f_{mpu}2 > f_{mpu}1 > f_{mpu}0$,

$f_{iva}5 > f_{iva}4 > f_{iva}3 > f_{iva}2 > f_{iva}1 > f_{iva}0$ and

f_{mpu} may not be equal to f_{iva}

The clock configuration for the MPU and the IVA applies to the following clocks of the device:

- DPLL1 (MPU DPLL) synthesized clock frequency (CLKOUTX2) configured by setting the M and N parameters of the DPLL
- DPLL1 (MPU DPLL) output clock frequency (MPU_CLK) configured by setting the M2 parameter of the DPLL

Note: The MPU_CLK is divided by 2 in the MPU subsystem to generate the ARM_FCLK. This divider is active only when the DPLL is locked. (For information about ARM_FCLK, see Chapter 3, *MPU Subsystem*.)

- DPLL2 (IVA2 DPLL) synthesized clock frequency (CLKOUT) configured by setting the M and N parameters of the DPLL
- DPLL2 (IVA2 DPLL) output clock frequency (IVA2_CLK) configured by setting the M2 parameter of the DPLL

The frequency of the processor clocks (MPU_FCLK and IVA2_FCLK) must be configured according to the selected OPP (the clock frequency associated with the operating voltage VDD1). Table 4-55 identifies the clocks of the processors, their source clocks, and the configuration register bit fields.

Table 4-55. Processor Clock Configuration Controls

Module	Clocks	Reference Clock	Multiplier (Factors)	Divider (Factors)	Configuration Bits
DPLL1	CLKOUTX2	SYS_CLK	M (0 ... 2047)		PRCM.CM_CLKSEL1_PLL_MPU[18:8] MPU_DPLL_MULT
				N (0 ... 127)	PRCM.CM_CLKSEL1_PLL_MPU[6:0] MPU_DPLL_DIV
	MPU_CLK (1)	CLKOUTX2		M2	PRCM.CM_CLKSEL2_PLL_MPU[4:0] MPU_DPLL_CLKOUT_DIV
DPLL2	CLKOUT	SYS_CLK	M (0 ... 2047)		PRCM.CM_CLKSEL1_PLL_IVA2[18:8] IVA2_DPLL_MULT
				N (0 ... 127)	PRCM.CM_CLKSEL1_PLL_IVA2[6:0] IVA2_DPLL_DIV
	IVA2_CLK	CLKOUT		M2 (1 ... 16)	PRCM.CM_CLKSEL2_PLL_IVA2[4:0] IVA2_DPLL_CLKOUT_DIV

(1) The MPU_CLK is divided by 2 in the MPU subsystem to generate the ARM_FCLK. This divider is active wonly hen the DPLL is locked. For information about ARM_FCLK, see Chapter 3, *MPU Subsystem*.

Table 4-56. Processor Clock Configurations

DPLL State	BYPASS	LOCKED
MPU_CLK	DPLL1_FCLK (bypass clock from DPLL3)	(SYS_CLK * M * 2)/([N+1] * M2)
ARM_FCLK	MPU_CLK	MPU_CLK/2
MPU subsystem internal module clocks	ARM_FCLK/2	ARM_FCLK/2

4.7.8.2 Interface and Peripheral Functional Clock Configurations

The interface clock OPPs are identified as the pair of VDD2 operating voltage level and the device interface clocks frequencies.

The DPLL3 (CORE DPLL) generates the CORE_CLK, which serves as the source clock for the L3_ICLK and L4_ICLK interface clocks of the device. The CORE_CLOCK is also used by the DPLL1 and DPLL2 as the bypass clock. The L3_ICLK is supplied to the SGX module as SGX_L3_ICLK and is used as its functional clock. The L4_ICLK is used by RM_L4_CLK as its source clock.

The interface and peripheral functional clock frequencies can be configured according to the device performance requirements for the OPP.

DPLL3 synthesized clock frequencies are configured as:

- $f_{CLKOUT} = (f_{SYS_CLK} \times M)/(N+1)$
- $f_{CLKOUTX2} = f_{CLKOUT} \times 2$

DPLL3 output clock frequencies are configured as:

- $f_{CORE_CLK} = f_{CLKOUT}/M2$
- $f_{COREX2_CLK} = f_{CLKOUTX2}/M2$

L3_ICLK, L4_ICLK and RM_L4_ICLK frequencies are configured as:

- $f_{L3_ICLK} = f_{CORE_CLK}/DIV_L3$
- $f_{L4_ICLK} = f_{L3_ICLK}/DIV_L4$
- $f_{RM_L4_ICLK} = f_{L4_ICLK}/DIV_RM$

Table 4-57 identifies the interface clocks, their reference clocks, and the control bits for configuration of the interface clock frequencies.

Table 4-57. Interface Clock Configuration Controls

Module	Clock	Reference Clock	Multiplier (Factor)	Divider (Factor)	Configuration Bits
DPLL3	CLKOUT	SYS_CLK	M (0 ... 2047)		PRCM.CM_CLKSEL1_PLL[26:16] CORE_DPLL_MULT
	CLKOUTX2			N (0 ... 127)	PRCM.CM_CLKSEL1_PLL[14:8] CORE_DPLL_ DIV
CORE_CLK	CORE_CLK	CLKOUT		M2 (1 ... 31)	PRCM. CM_CLKSEL1_PLL[31 :27] CORE_DPLL_CLKOUT_DIV
	COREX2_CLK	CLKOUTX2			
L3 interconnect	L3_ICLK	CORE_CLK		DIV_L3 (1 ... 2)	PRCM. CM_CLKSEL_CORE[1:0] CLKSEL_L3
L4 interconnect	L4_ICLK	L3_ICLK		DIV_L4 (1 ... 2)	PRCM.CM_CLKSEL_CORE[3:2] CLKSEL_L4
RM clock	RM_L4_ICLK	L4_ICLK		DIV_RM (1 ... 2)	PRCM.CM_CLKSEL_WKUP[2:1] CLKSEL_RM

SGX_FCLK, DPLL1_FCLK (bypass mode) and DPLL2_FCLK (bypass mode) frequencies are configured as:

- $f_{SGX_FCLK} = f_{CORE_CLK}/DIV_SGX$
- $f_{SGX_FCLK} = f_{CM_96M_FCLK}$
- $f_{DPLL1_FCLK} = f_{CORE_CLK}/DIV_DPLL1$
- $f_{DPLL2_FCLK} = f_{CORE_CLK}/DIV_DPLL2$

Table 4-58 identifies the functional clocks, their reference clocks, and the control bits for configuration of the functional clock frequencies.

Table 4-58. Functional Clock Configuration Controls

Module	Clock	Reference Clock	Divider (Factor)	Configuration Bits
SGX	SGX_FCLK	CORE_CLK CM_96M_FCLK	DIV_SGX (3, 4, 6)	PRCM.CM_CLKSEL_SGX[2:0] CLKSEL_SGX
MPU HS bypass	DPLL1_FCLK	CORE_CLK	DIV_DPLL1 (1, 2, 4)	PRCM.CM_CLKSEL1_PLL_MPU[20:19] MPU_CLK_SRC
IVA2 HS bypass	DPLL2_FCLK	CORE_CLK	DIV_DPLL2 (1, 2, 4)	PRCM.CM_CLKSEL1_PLL_IVA2[20:19] IVA2_CLK_SRC

Note: SGX clock control is available only on OMAP3430.

The rest of the functional clocks are issued from DPLL4 and DPLL5 and remain invariable, regardless of the interface clock configuration. In all clock configurations, any divider ratio to generate functional clocks is applicable, provided it complies with the maximum frequency specification.

CAUTION

Before any transition to OPP1 or OPP2, set the DPLL2 bypass clock to CORE_CLK/4 in the CM_CLKSEL1_PLL_IVA2[20:19] IVA2_CLK_SRC bit field. During relock phase, the DSP clock is 83 MHz. This has no effect on performance because DPLL2 relock time is only a few µs.

Before any transition to OPP1 for DPLL1, set the DPLL1 bypass clock to CORE_CLK/4 in the CM_CLKSEL1_PLL_MPU[20:19] MPU_CLK_SRC bit field. Otherwise, set the DPLL1 bypass clock to CORE_CLK/2.

4.8 PRCM Idle and Wake-Up Management

4.8.1 Overview

When a group of modules belonging to a clock domain in a power domain does not require a clock (interface or functional), the PRCM module can be programmed to automatically cut the clock to these modules, thereby reducing their power consumption. The PRCM module can then switch the power domain to low-power retention or off mode to ensure minimum power consumption. When all clocks in a domain are cut, the domain is idle.

Similarly, when a module belonging to a power domain in low-power idle mode is required to switch to active mode, the PRCM module switches on the power to the entire power domain and activates the necessary clock signals to the module. This is a wake-up transition. Generally, a wake-up event triggers the wake-up transition.

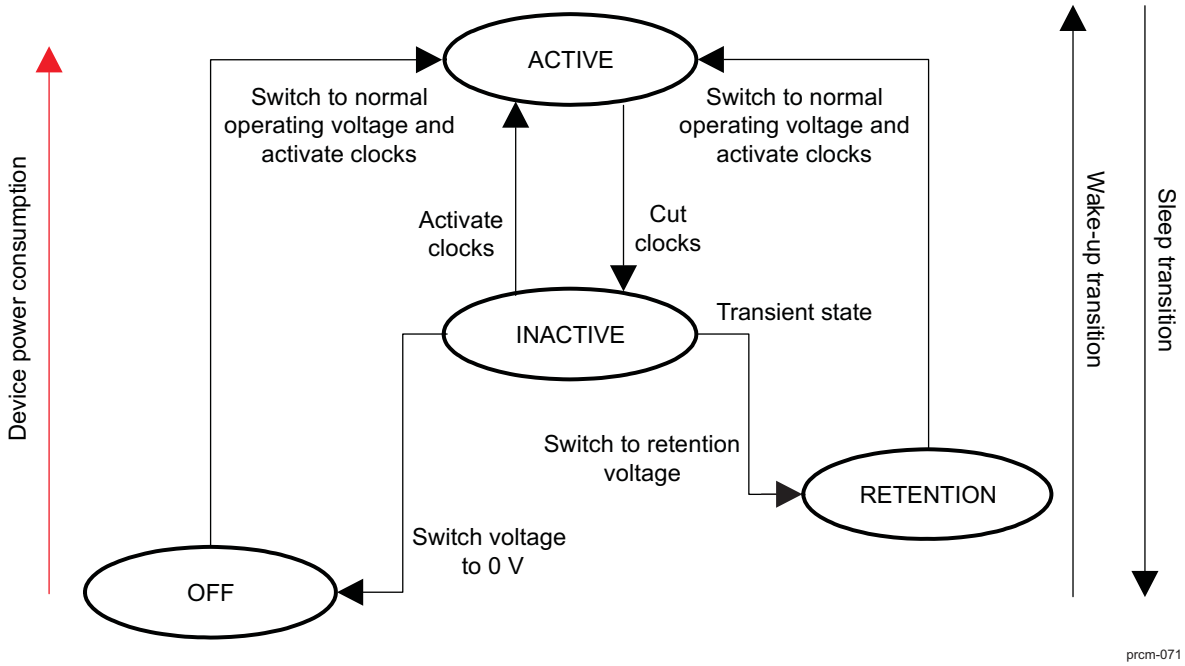
A sleep/wake-up dependency can be defined between power domains. A sleep dependency ensures that a power domain does not make a sleep transition unless all the dependent power domains are in idle mode and do not require the power domain. Similarly, a wake-up dependency ensures that a power domain wakes up when any of its dependent power domains wakes up.

The PRCM module automatically handles the sequence clock-gating conditions and power switching for each power domain, based on the configured dependencies between the domains and the clock-control bits of the modules.

Figure 4-70 shows the sleep/wake-up transition of the power domains. The inactive state is a transient power state of a power domain, while moving from active to retention or off power state. In inactive state, all the domain clocks are gated. A power domain cannot switch from off to retention or retentiON to OFF power state without passing to the active power state.

A low-power state (inactive, retention, or off) means less power consumption. However, wake-up latency increases as the power domain switches to inactive, retention, or off power state.

Figure 4-70. Power Domain Sleep/Wake-Up Transition



Functionally, the PRCM module is composed of a single global control block and a power-control block for each power domain. The domain power-control block handles power switching, wake-up events, and clock-gating control for the domain. The domain power controllers communicate with each other for power sequencing, sleep dependencies, and wake-up dependencies. All these blocks interact with the global control block that handles reset management, clock generation and distribution, and all PRCM registers.

Figure 4-71. Device Power Reset and Clock Controllers

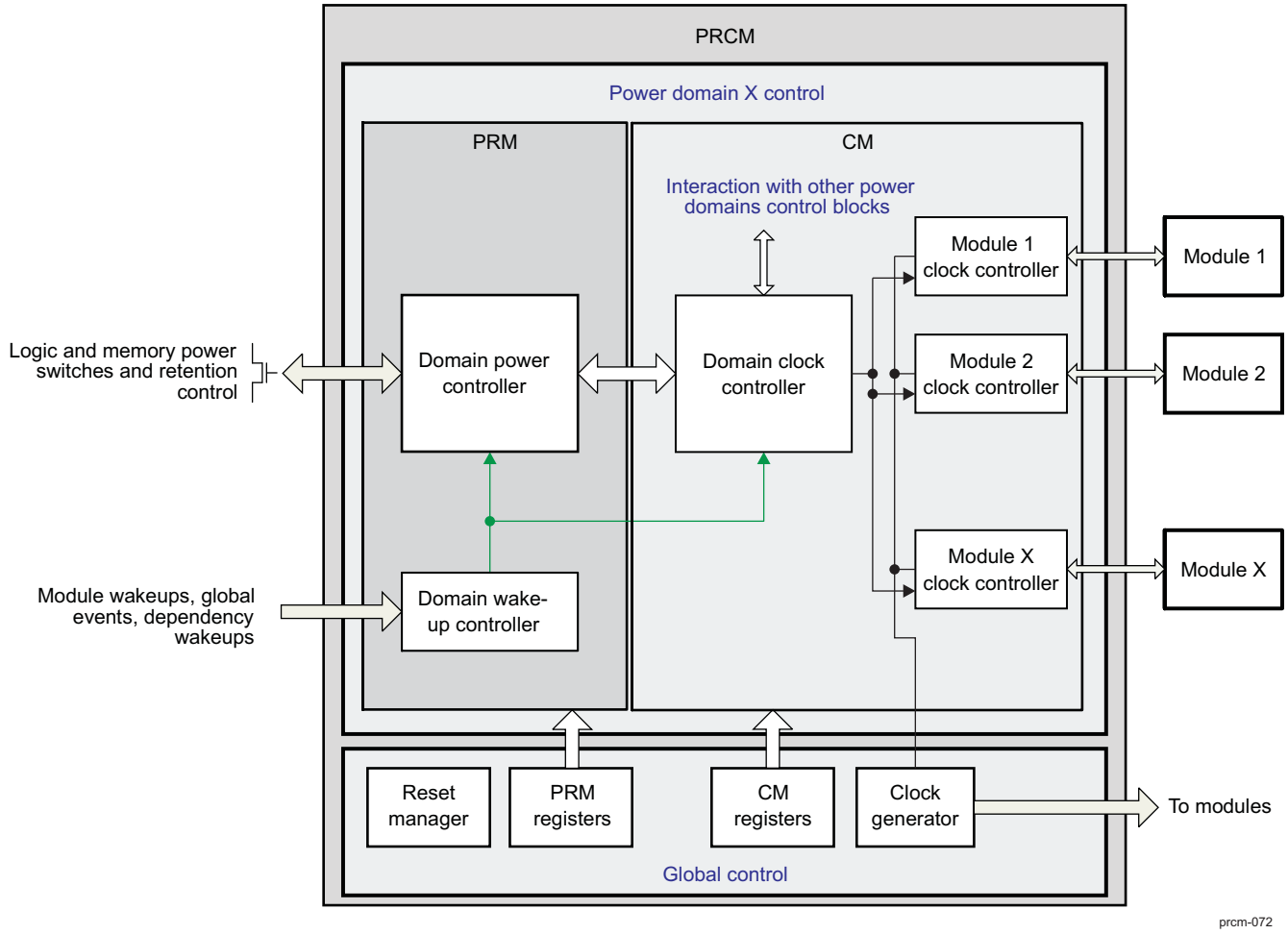


Figure 4-71 shows the functions in one power domain block:

- A module clock controller (for example, module X clock controller) uses a hardware handshake protocol to communicate directly with the module. It controls the idle transition of the target module and responds to the standby requests of the initiator module (for details, see Section 4.1.4.2, *Autoidle Clock Control*). Depending on the mode, the clock controller sends clock commands (enable/disable) to the clock generator. The registers that control the functional and interface clocks in the module are directly mapped to the module clock controller.
- The domain clock controller gathers information from the following:
  - All module clock controllers of the power domain
  - The domain power controller
  - The domain wake-up controller
  - The other power domain control blocks

  The domain clock controller informs the domain power controller when all conditions for the domain power transition are met (domain clocks are idle, no initiator is requesting service, no wake-up event is pending) to allow a domain power transition. On a wake-up event, the domain clock controller starts the module domain clocks, if power is present.
- The domain power controller communicates with the domain clock controller, the domain wake-up controller, and the domain power controller.
- The domain wake-up controller gathers all events that can wake up the power domain. Some events are active only when the domain is on (these switch on the clocks, when required); others also allow domain power up. These events can be internal (coming from the modules in the power domain, such as general-purpose [GP] timer time-out), external (coming from another power domain control block

dependency), or global (voltage stabilization).

- The reset manager globally controls all resets in the device. It gathers information from all power domain control blocks to sequence power and clocks, and resets the activation of each domain.
- The clock generator generates and distributes clocks over the device, depending on requests from all module clock controllers and on other global conditions.

4.8.2 Sleep Transition

The PRCM module can initiate a domain sleep transition on a power domain only if the domain meets the following conditions:

- All initiator modules are idle (they have completed their activity and idled themselves through software requests).
- All target modules are idle (automatically when all initiators are in standby mode or on software request).
- All sleep dependencies with other domains are met (can be set by software).

When the sleep conditions are met (based on the settings of the PRCM.PRM_PWSTCTRL_<power domain>[1,0] POWERSTATE bit field), the PRCM module performs the actions described in Table 4-59 automatically or when instructed by the software.

Table 4-59. Power-State-Related Sleep Transition Actions

Power Domain State	Action
ON	All functional and interface clocks in the power domain are shut down when the sleep conditions are met. The power domain is idle and is not functional.
Retention	The power domain is idle and part or all of the logic and memory of the domain is switched to retention mode.
Off	The domain is idled and all the logic and memory in the domain are switched off.

The power domain state transition can be set to automatic (hardware controlled) or software-controlled by setting the PRCM. CM_CLKSTCTRL_<power domain> CLKTRCTRL_<power domain> bit field.

4.8.3 Wakeup

A wake-up event switches on two domains:

- A power domain (logic and associated memories) that is in inactive state
- A clock domain (including related clock sources, such as a DPLL)

If a domain is already on and the clock domain is idle, only the clocks are reactivated on a wake-up event.

To wake up, a processor requires an interrupt associated with the wake-up event. If its power domain is to transition from off or retention state to active, it must be reset.

There are three types of wake-up events:

- Global: Generated on a particular device event (device wakeup, voltage transition completed, DPLL recalibration, etc.). Used mainly to wake up the MPU domain
- Module: Functional wake-up event issued from a module, which wakes up the domain where the module resides. It can also directly wake up the MPU or the IVA2.2 processor, depending on software settings in the PRCM.PM_MPUGRPSEL_<power domain> and PRCM.PM_IVA2GRPSEL_<power domain> registers.
- Dependency: A power domain can wake up on the wakeup of another power domain. The dependency is software-controllable by configuring the PRCM.PM_WKDEP_<power domain> register.

4.8.4 Device Wake-Up Events

This section summarizes the wake-up events for each power domain. Table 4-60 through Table 4-70 list the wake-up events, related control registers, and MPU and IVA2.2 interrupts.

Two registers, PRCM.PRM_IRQENABLE_MPU and PRCM.PRM_IRQENABLE_IVA2, enable the MPU and IVA2.2 interrupts. In some cases, they can also enable the wake-up feature associated with the interrupt (DPLL recalibration requests, voltage controller and processors errors, and device wake-up event).

Notes:

- The PRCM module can be configured to generate an interrupt to the MPU or the IVA2 subsystem as a result of a wake-up event from the CORE, WKUP, and PER power domain modules to the MPU and the IVA2. However, these modules can also directly interrupt the MPU and the IVA2 subsystems. To avoid a double interrupt (from the PRCM module and one of these modules) as the result of a single event (wakeup), one interrupt must be masked when the other is unmasked. For more information about the interrupt capability of a module, see the module chapter.
- The UART, GPIO, and McSPI modules generate an asynchronous wake-up event (their interface and functional clocks can be gated during the sleep period). However, because the GPTIMERs generate a synchronous wake-up event, they require their functional clock to be active during the sleep period; their interface clock can be gated. McBSP modules can generate a synchronous or asynchronous wake-up event, based on the mode configurations. For more information about the interrupt capability of a module, see the module chapter.
- The ability of a module to generate a wake-up event depends on the power state of the power domain in which the module resides. If the power domain is inactive (the functional and interface clocks of the domain are gated), only the asynchronous wake-up modules can wake up the power domain. If only the interface clocks in the domain are gated, synchronous and asynchronous wake-up modules can generate a wake-up event to activate the interface clock. Similarly, if the power domain is in retention or off power state (the domain logic is nonfunctional), no wake-up event can be generated by any module in that power domain.

Table 4-60. MPU Power Domain Wake-Up Events

Internal Wake-Up Events	Source Module	PRCM Software Control	Wake-Up Event	Interrupt/Type	Interrupt to
IVA2, CORE, DSS, and PER domain dependency	PRCM	PM_WKDEP_MPU	Yes	No	N/A
Peripherals wake-up events	Peripherals	PM_MPUGRPSEL1_CORE	Yes	MPU peripheral group event occurred.	MPU
		PM_MPUGRPSEL3_CORE			
		PM_MPUGRPSEL_WKUP			
		PM_MPUGRPSEL_PER			
Event generator on, off time	PRCM	PM_EVGENCTRL_MPU	Yes	Event generator on, off	MPU
Forced wake-up transition	PRCM	CM_CLKSTCTRL_IVA2	No	Wake-up transition is complete (IVA2, NEON, SGX, USBHOST, DSS, CAM, PER, EMU domains).	MPU
		CM_CLKSTCTRL_NEON			
		CM_CLKSTCTRL_SGX			
		CM_CLKSTCTRL_DSS			
		CM_CLKSTCTRL_CAM			
		CM_CLKSTCTRL_PER			
		CM_CLKSTCTRL_USBHOST			
		CM_CLKSTCTRL_EMU			

Table 4-60. MPU Power Domain Wake-Up Events (continued)

Internal Wake-Up Events	Source Module	PRCM Software Control	Wake-Up Event	Interrupt/Type	Interrupt to
Forced sleep transition	PRCM	CM_CLKSTCTRL_IVA2	No	Sleep transition is complete (IVA2, NEON, SGX, USBHOST, DSS, CAM, PER, EMU domains).	MPU
		CM_CLKSTCTRL_NEON			
		CM_CLKSTCTRL_SGX			
		CM_CLKSTCTRL_DSS			
		CM_CLKSTCTRL_CAM			
		CM_CLKSTCTRL_PER			
		CM_CLKSTCTRL_USBHOST			
		CM_CLKSTCTRL_EMU			
DPLL1 recalibration request	PRCM	N/A	Yes	MPU DPLL recalibration event	MPU
DPLL2 recalibration request	PRCM	N/A	Yes	IVA2 DPLL recalibration event	MPU, IVA2
DPLL3 recalibration request	PRCM	N/A	Yes	CORE DPLL recalibration event	MPU
DPLL4 recalibration request	PRCM	N/A	Yes	Peripheral DPLL recalibration event	MPU
DPLL5 recalibration request	PRCM	N/A	Yes	Peripheral DPLL2 recalibration event	MPU
Voltage controller error	PRCM	N/A	Yes	Voltage controller error status (I^2C frame not acknowledged)	MPU
Device wake-up event	PRCM	N/A	Yes	Any PAD wake-up event when CORE domain is off	MPU

Table 4-61. NEON Power Domain Wake-Up Events

Internal Wake-Up Events	Source Module	PRCM Software Control	Wake-Up Event	Interrupt/Type	Interrupt to
MPU domain dependency	PRCM	PM_WKDEP_NEON	Yes	No	N/A
Forced wake-up transition	PRCM	CM_CLKSTCTRL_NEON	Yes	Wake-up transition is complete.	MPU

Table 4-62. IVA2 Power Domain Wake-Up Events

Internal Wake-Up Events	Source Module	PRCM Software Control	Wake-Up Event	Interrupt/Type	Interrupt to
MPU domain dependency	PRCM	PM_WKDEP_IVA2	Yes	No	N/A
CORE domain dependency	PRCM	PM_WKDEP_IVA2	Yes	No	N/A
DSS domain dependency	PRCM	PM_WKDEP_IVA2	Yes	No	N/A
PER domain dependency	PRCM	PM_WKDEP_IVA2	Yes	No	N/A
WKUP domain dependency	PRCM	PM_WKDEP_IVA2	Yes	No	N/A
Peripheral wake-up events	Peripherals	PM_IVA2GRPSEL1_CORE	Yes	IVA2.2 peripheral group event occurred .	IVA
		PM_IVA2GRPSEL3_CORE			
		PM_IVA2GRPSEL_PER			

Table 4-62. IVA2 Power Domain Wake-Up Events (continued)

Internal Wake-Up Events	Source Module	PRCM Software Control	Wake-Up Event	Interrupt/Type	Interrupt to
Forced wake-up transition	PRCM	CM_CLKSTCTRL_IVA2	Yes	Wake-up transition is complete.	IVA, MPU

Table 4-63. SGX Power Domain Wake-Up Events

Internal Wake-Up Events	Source Module	PRCM Software Control	Wake-Up Event	Interrupt/Type	Interrupt to
MPU domain dependency	PRCM	PM_WKDEP_SGX	Yes	No	N/A
IVA2 domain dependency	PRCM	PM_WKDEP_SGX	Yes	No	N/A
WKUP domain dependency	PRCM	PM_WKDEP_SGX	Yes	No	N/A
Forced wake-up transition	PRCM	CM_CLKSTCTRL_SGX	Yes	Wake-up transition is complete.	MPU

Table 4-64. CORE Power Domain Wake-Up Events

Internal Wake-Up Events	Source Module	PRCM Software Control	Wake-Up Event	Interrupt/Type	Interrupt to
MPU domain dependency	PRCM	Hardware set (always-enabled)	Yes	No	N/A
IVA2 domain dependency	PRCM		Yes	No	N/A
CAM domain dependency	PRCM		Yes	No	N/A
DSS domain dependency	PRCM		Yes	No	N/A
USBHOST domain dependency	PRCM		Yes	No	N/A
PER domain dependency	PRCM		Yes	No	N/A
SGX domain dependency	PRCM		Yes	No	N/A
WKUP domain dependency	PRCM		Yes	No	N/A
HS USB OTG wakeup	HS USB OTG	PM_processor>GRPSEL1_ CORE, PM_WKEN1_CORE	Yes	No	N/A
McBSP1 wakeup	McBSP 1		Yes	No	N/A
McBSP5 wakeup	McBSP 5		Yes	No	N/A
GPTIMER10 wakeup	GPTIMER 10		Yes	No	N/A
GPTIMER11 wakeup	GPTIMER 11		Yes	No	N/A
UART1 wakeup	UART 1		Yes	No	N/A
UART2 wakeup	UART 2		Yes	No	N/A
McSPI1 wakeup	McSP 1		Yes	No	N/A
McSPI2 wakeup	McSP 2		Yes	No	N/A
McSPI3 wakeup	McSP 3		Yes	No	N/A
McSPI4 wakeup	McSP 4		Yes	No	N/A
MMC1 wakeup	MMC 1		Yes	No	N/A
MMC2 wakeup	MMC 2		Yes	No	N/A
MMC3 wakeup	MMC 3		Yes	No	N/A

Table 4-64. CORE Power Domain Wake-Up Events (continued)

Internal Wake-Up Events	Source Module	PRCM Software Control	Wake-Up Event	Interrupt/Type	Interrupt to
USBTLL wakeup	USBTLL		Yes	No	N/A
Device wake-up event	PRCM	N/A	Yes	Any PAD wake-up event when CORE domain is off	MPU

Table 4-65. DSS Power Domain Wake-Up Events

Internal Wake-Up Events	Source Module	PRCM Software Control	Wake-Up Event	Interrupt/Type	Interrupt to
MPU domain dependency	PRCM	PM_WKDEP_DSS register	Yes	No	N/A
IVA2 domain dependency	PRCM	PM_WKDEP_DSS register	Yes	No	N/A
WKUP domain dependency	PRCM	PM_WKDEP_DSS register	Yes	No	N/A
Forced transition state wakeup	PRCM	CM_CLKSTCTRL_DSS register	Yes	Wake-up transition is complete.	MPU

Table 4-66. CAM Power Domain Wake-Up Events

Internal Wake-Up Events	Source Module	PRCM Software Control	Wake-Up Event	Interrupt/Type	Interrupt to
MPU domain dependency	PRCM	PM_WKDEP_CAM register	Yes	No	N/A
IVA2 domain dependency	PRCM	PM_WKDEP_CAM register	Yes	No	N/A
WKUP domain dependency	PRCM	PM_WKDEP_CAM register	Yes	No	N/A
Forced transition state wakeup	PRCM	CM_CLKSTCTRL_CAM register	Yes	Wake-up transition is complete.	MPU

Table 4-67. USBHOST Power Domain Wake-Up Events

Internal Wake-Up Events	Source Module	PRCM Software Control	Wake-Up Event	Interrupt/Type	Interrupt to
MPU domain dependency	PRCM	PM_WKDEP_USBHOST register	Yes	No	N/A
CORE domain dependency	PRCM	PM_WKDEP_USBHOST register	Yes	No	N/A
IVA2 domain dependency	PRCM	PM_WKDEP_USBHOST register	Yes	No	N/A
WKUP domain dependency	PRCM	PM_WKDEP_USBHOST register	Yes	No	N/A
USBHOST wake-up	HS USB Host	PM_processor>GRPSEL_ USBHOST, PM_WKEN_USBHOST	Yes	No	N/A
Forced transition state wakeup	PRCM	CM_CLKSTCTRL_USBHOST register	Yes	Wake-up transition is complete.	MPU

Table 4-68. PER Power Domain Wake-Up Events

Internal Wake-Up Events	Source Module	PRCM Software Control	Wake-Up Event	Interrupt/Type	Interrupt to
MPU domain dependency	PRCM	PM_WKDEP_PER register	Yes	No	N/A

Table 4-68. PER Power Domain Wake-Up Events (continued)

Internal Wake-Up Events	Source Module	PRCM Software Control	Wake-Up Event	Interrupt/Type	Interrupt to
IVA2 domain dependency	PRCM	PM_WKDEP_PER register	Yes	No	N/A
CORE domain dependency	PRCM	PM_WKDEP_PER register	Yes	No	N/A
WKUP domain dependency	PRCM	PM_WKDEP_PER register	Yes	No	N/A
McBSP2 wakeup	McBSP 2	PM_processor>GRPSEL1_ PER, PM_WKEN_PER	Yes	No	N/A
McBSP3 wakeup	McBSP 3		Yes	No	N/A
McBSP4 wakeup	McBSP 4		Yes	No	N/A
GPTIMER2 wakeup	GPTIMER2		Yes	No	N/A
GPTIMER3 wakeup	GPTIMER3		Yes	No	N/A
GPTIMER4 wakeup	GPTIMER4		Yes	No	N/A
GPTIMER5 wakeup	GPTIMER5		Yes	No	N/A
GPTIMER6 wakeup	GPTIMER6		Yes	No	N/A
GPTIMER7 wakeup	GPTIMER7		Yes	No	N/A
GPTIMER8 wakeup	GPTIMER 8		Yes	No	N/A
GPTIMER9 wakeup	GPTIMER9		Yes	No	N/A
UART3 wakeup	UART3		Yes	No	N/A
GPIO2 wakeup	GPIO 2		Yes	No	N/A
GPIO3 wakeup	GPIO 3		Yes	No	N/A
GPIO4 wakeup	GPIO 4		Yes	No	N/A
GPIO5 wakeup	GPIO 5		Yes	No	N/A
GPIO6 wakeup	GPIO 6		Yes	No	N/A
Forced transition state wakeup	PRCM	CM_CLKSTCTRL_PER register	Yes	Wake-up transition is complete.	MPU

Table 4-69. EMU Power Domain Wake-Up Events

Internal Wake-Up Events	Source Module	PRCM Software Control	Wake-Up Event	Interrupt/Type	Interrupt to
EMU wakeup	ICEPick-C	N/A	Yes	No	N/A
Forced transition state wakeup	PRCM	CM_CLKSTCTRL_EMU register	Yes	Wake-up transition is complete.	MPU

Table 4-70. WKUP Power Domain Wake-Up Events

Internal Wake-Up Events	Source Module	PRCM Software Control	Wake-Up Event	Interrupt/Type	Interrupt to
Device wakeup	PRCM	PM_WKEN_WKUP	Yes	No	N/A

Table 4-70. WKUP Power Domain Wake-Up Events (continued)

Internal Wake-Up Events	Source Module	PRCM Software Control	Wake-Up Event	Interrupt/Type	Interrupt to
GPTIMER1 wakeup	GPTIMER1		Yes	No	N/A
GPIO1 wakeup	GPIO 1	PM_WKEN_WKUP	Yes	No	N/A

Note: Some of the 32 I/O pins for the GPIO1 in the WKUP power domain are connected to the device I/O pad logic in the CORE power domain (VDD2) and to the reset to the device I/O pad logic in the WKUP power domain (VDD3). As a result, when the CORE power domain is off, the corresponding I/O pins of the GPIO1 cannot generate a wake-up event. For details, see Chapter 25, *General-Purpose Interface Module*.

4.8.5 Sleep and Wake-Up Dependencies

4.8.5.1 Sleep Dependencies

The (clock activity) dependencies between power domains are implemented to manage their sleep and wake-up transitions to ensure stable operation of the device.

A power domain sleep transition is achieved when all its clock domains (that is, functional and interface) are gated. For the gating of a clock domain, the following conditions must be satisfied:

- All initiator modules in the clock domain are in standby mode, that is, they cannot initiate any new transitions.
- All target modules in the clock domain are in idle mode and have no pending transitions.
- If the power domain depends on another power domain (that is, has a sleep dependency), the clock domains of the other power domain must be muted.

A clock domain is said to be muted when all its public initiator modules (that is, the initiator modules of the clock domain that can generate interconnect transactions towards targets outside the clock domain) are in standby mode, and the domain cannot initiate new interconnect transactions toward other domains.

Table 4-71 describes the mute conditions for the clock domains.

Table 4-71. Clock Domain Mute Conditions

Clock Domain Name	Clock Domain Composition	Mute Conditions
MPU	MPU power domain and MPU INTC in CORE power domain	MPU in stand-by, MPU INTC idle
IVA2	IVA2 power domain and WUGEN in CORE power domain	IVA2.2 in stand-by, WUGEN idle
SGX	SGX power domain	SGX in stand-by
CAM	CAM power domain	CAM in stand-by
DSS	DSS power domain	DSS in stand-by
PER	PER power domain	N/A (No public Initiator)
CORE_L3	L3 interconnect, L3 targets/initiators in CORE power domain	sDMA in standby
CORE_L4	L4 interconnect, L4 targets in CORE power domain	N/A (No public Initiator)
USBHOST	USBHOST power domain	USBHOST in stand-by
WKUP	WKUP power domain	N/A (No public Initiator)

A sleep dependency prevents the PRCM module from gating the clocks to a power domain (for sleep transition to inactive, retention, or off state from on state) if a dependent power domain is active (its clocks are active). A power domain can depend on several other power domains.

A power domain sleep dependency is programmed by setting the PRCM.CM_SLEEPDEP_<domain> register.

Example:

The PER domain has a programmable sleep dependency with the MPU and IVA2 power domains, and a hardwired sleep dependency with the CORE power domain. Therefore, if all software dependencies are enabled, the PER domain can go to idle mode only if the MPU, IVA2, and CORE-L3 clock domains are idle. If all software dependencies are disabled, the PER domain can go to idle mode, provided that the CORE-L3 clock domain is muted (no access to the PER domain can be pending).

Table 4-72 summarizes the programmable and hardwired sleep dependencies among the domains.

Note: The first row of the table identifies the dependent power domains for a power domain listed in the first column. Therefore, to identify the sleep dependency of power domain X, identify domain X in the first column, then follow its row to its sleep dependency with another power domain (identified in the top row).

Table 4-72. Sleep Dependencies

		Sleep Dependency											
Power Domain	**Clock Domain**	**MPU**	**NEON**	**IVA2**	**SGX**	**CAM**	**DSS**	**PER**	**CORE_ L3**	**CORE_ L4**	**CORE_ CM**	**USB HOST**	**WKUP**
MPU	MPU	n/a	0	0	0	0	0	0	0	0	0	0	0
NEON	MPU	1	n/a	0	0	0	0	0	0	0	0	0	0
IVA2	IVA2	0	0	n/a	0	0	0	0	0	0	0	0	0
SGX	SGX	RW	0	0	n/a	0	0	0	0	0	0	0	0
CAM	CAM	RW	0	0	0	n/a	0	0	0	0	0	0	0
DSS	DSS	RW	0	RW	0	0	n/a	0	RW	0	0	0	0
PER	PER	RW	0	RW	0	0	0	n/a	1	0	0	0	0
CORE	CORE_L3	1	0	1	1	1	1	1	n/a	1	0	1	1
	CORE_L4	1	0	1	0	0	0	0	1	n/a	0	0	0
	CORE_CM	1	1	1	1	1	1	1	1	1	n/a	1	1
USBHOST	USBHOST	RW	0	RW	0			0	0	0	0	n/a	0
WKUP	WKUP	1	0	1	0	0	0	0	1	0	0	0	n/a

Notes:

No software control (hardwired values):	0	Does not depend on
	1	Depends on
Software controllable	RW	Read and write
	n/a	Not applicable

4.8.5.2 Wake-Up Dependencies

A wake-up dependency allows a power domain to wake up (from off, retention or inactive state to on state) when another power domain wakes up (from off, retention or inactive state to on state). For example, power domain one (PD1) provides a service to power domain two (PD2), which creates a dependency between the two domains. When the dependent power domain (PD2) wakes up, it signals PD1 with a broadcasted wake-up event, causing PD1 to wake up.

A broadcasted wake-up event can originate from one of two sources:

- PD2 internal wake-up event (typically, a peripheral wake-up event)
- Abortion of the mute mode (at least one initiator in the domain exits standby mode) of PD2 when both the sleep dependency and wake-up dependency with PD1 are enabled.

Note: The domain wake-up dependency is a non-transitive property.

If a power domain PD1 has wake-up dependency with power domain PD2, i.e., PRCM.PM_WKDEP_PD1. EN_PD2 bit is set to 1 and if PD2 has wake-up dependency with power domain PD3, i.e., PRCM.PM_WKDEP_PD2. EN_PD3 bit is set to 1.

However, if PD1does not have a direct wake-up dependency with PD3, i.e., PRCM.PM_WKDEP_PD1. EN_PD3 bit is set to 0. Then if the PD1 and PD2 are in inactive/retention or off power state and PD3 is inactive and is woken-up by a wake-up event, the PD2 is also woken-up but not the PD1.

Because a power domain can depend on several other power domains, it can broadcast the wake-up signal to each power domain on which it depends.

A power domain wake-up dependency can either be hardwired (set in the hardware) or programmable through software by configuring the PM_WKDEP_<domain> register for that power domain. Continuing the example of PD2 as dependent on PD1, setting the PM_WKDEP_PD1.PD2 bit means that when PD2 wakes up, PD1 also wakes up.

The MPU power domain can be wakened only by the IVA2, DSS, USBHOST, PER, CORE-L3, CORE-L4, and WKUP power domains. Similarly, the IVA2 power domain can be wakened only by the MPU, DSS, USBHOST, PER, CORE-L3, CORE-L4, and WKUP power domains. All wake-up dependencies of the MPU and the IVA2 power domain are software-configurable.

For the processor power domains (MPU and the IVA2 power domain), when the wake-up dependency with other power domains is software-programmable (that is, with the USBHOST, PER, CORE, and WKUP power domains), two registers may need be configured: PM_WKDEP_<domain> and PM_<processor>GROUPSEL_<domain>.

The PM_WKDEP_<domain> register serves only to enable/disable the global wake-up dependency of the processor power domain on these three power domains. The PM_<processor>GROUPSEL_<domain> register must be configured to enable the particular wake-up event in these domains that will wake up the processor domain. Thus, the global wake-up dependency for a dependent domain must be enabled so that a wake-up event can occur, if it has been enabled in the corresponding GROUPSEL register.

For example, if the wake-up dependency of the MPU power domain is to be enabled for a wake-up event from the GPIO2 module of the PER power domain, the following configuration is required:

- PRCM.PM_MPUGRPSEL_PER[13] GRPSEL_GPIO2
- PRCM.PM_WKDEP_MPU[7] EN_PER

Table 4-73 summarizes the programmable and hardwired wake-up dependencies among the domains.

Table 4-73. Wake-Up Dependencies

		Wake-Up Dependency											
Power Domain	**Clock Domain**	**MPU**	**NEON**	**IVA2**	**SGX**	**CAM**	**DSS**	**USB HOST**	**PER**	**CORE_ L3**	**CORE_ L4**	**CORE_ CM**	**WKUP**
MPU	MPU	N/A(1)	0(2)	RW(3)	0	0	RW	RW**(4)	RW*(5)	RW*	RW*	0	RW**
NEON	MPU	RW	N/A	0	0	0	0	0	0	0	0	0	0
IVA2	IVA2	RW	0	N/A	0	0	RW	RW**	RW*	RW*	RW*	0	RW*
SGX	SGX	RW	0	RW	N/A	0	0	0	0	0	0	0	RW
CAM	CAM	RW	0	RW	0	N/A	0	0	0	0	0	0	RW
DSS	DSS	RW	0	RW	0	0	N/A	0	0	0	0	0	RW
USBHOST	USBHOST	RW	0	RW	0	0	0	N/A	0	RW	0	0	RW
PER	PER	RW	0	RW	0	0	0	0	N/A	RW	0	0	RW
CORE	CORE_L3	1	0	1	0	0	1	1	1	N/A	1	0	1
	CORE_L4	1	0	1	0	0	0	0	0	0	N/A	0	1
	CORE_CM	1	0	1	0	0	1	1	1	1	1	N/A	1
WKUP	WKUP	1	0	1	0	0	0	1	1	1	1	0	N/A

(1) N/A: Not applicable
(2) 0: No wake-up dependency applicable; 1: Hardware wake-up dependency: Dependency always enabled
(3) RW: Software wake-up dependency: Dependency by PM_WKDEP_<> register
(4) RW**: Software wake-up dependency: Dependency by PM_<>GRPSEL_<> registers
(5) RW*: Software wake-up dependency: Dependency by PM_WKDEP_<> and PM_<>GRPSEL_<> registers

4.8.6 USBHOST/USBTLL Save-and-Restore Management

Both USBHOST and USBTLL support a save-and-restore mechanism. When the device enters into off mode (that is, all power domains, except the WKUP power domain, are off), the user may want to save the USBHOST context and restore it when exiting off mode. The save-and-restore mechanism for the HS USB Host is enabled by setting the PRCM.PM_PWSTCTRL_USBHOST[4] SAVEANDRESTORE bit; for the USBTLL, it is configured by the PRCM.PM_PWSTCTRL_CORE[4] SAVEANDRESTORE bit. The save mechanism is initiated as the power domain transitions from active to off state (or to OSWR state for the USBTLL), whereas the restore mechanism is initiated as the power domain transitions from off to active power state.

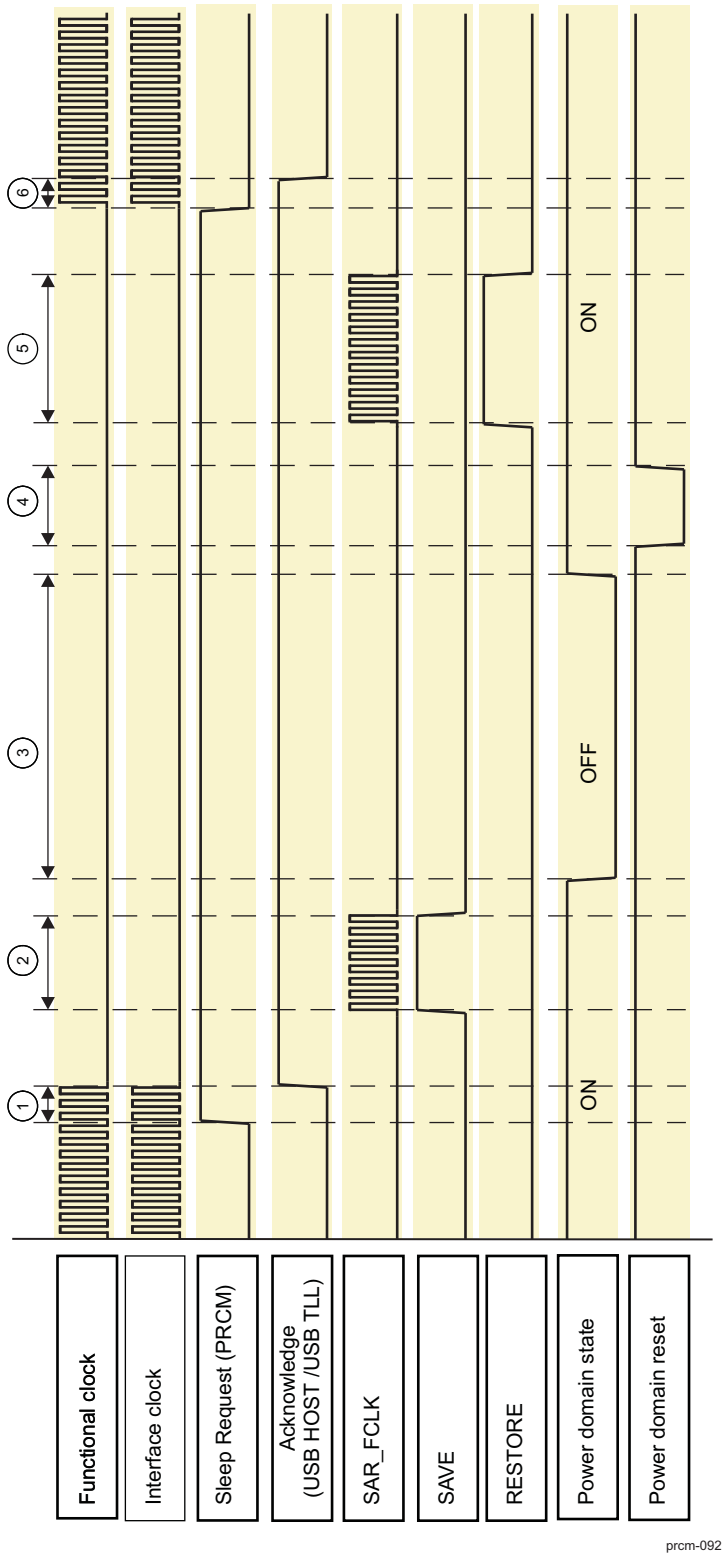
Figure 4-72 shows the generic save-and-restore sequence applicable to both the USBHOST and the USBTLL. The sequence follows:

1. When the PRCM module intends to initiate a sleep transition over the power domain (the USBHOST power domain for the HS USB Host subsystem and the CORE power domain for the USBTLL module), it sends a sleep request to the power domain. When the domain modules acknowledge the sleep request, the functional and interface clocks to the modules are gated.
2. The PRCM module enables the SAR_FCLK functional clock and initiates the save sequence for the module. SAR_FCLK is gated when the save sequence completes.
3. The PRCM module switches the power domain state to off power state. When a wake-up event occurs, the PRCM module switches on the power domain.
4. The power domain local reset is asserted and then released by the reset manager.
5. The PRCM module enables the SAR_FCLK functional clock, and then initiates the restore sequence for the module. SAR_FCLK is gated when the restore sequence completes.
6. The PRCM module enables the functional and interface clocks to the modules and releases the sleep request. The modules acknowledge, and the wake-up sequence completes.

Note: The PRCM sleep request and the module acknowledge signals are part of a hardware communication interface to ensure the correct sleep and wake-up sequence.

Figure 4-72 shows the save-and-restore sequence. It does not provide the exact timing delays between the switching of the signals and serves only to highlight the sequence of events.

Figure 4-72. Save-and-Restore Sequence



4.8.6.1 USBHOST SAR Sequences

4.8.6.1.1 Save Sequence on Sleep Transition

A precondition to the save sequence on sleep transition is that the save-and-restore mechanism is enabled (the PRCM.PM_PWSTCTRL_USBHOST[4] SAVEANDRESTORE bit is set to 1). The sequence is initiated when the power domain switches from ON to OFF power state.

1. The PRCM module activates USBHOST_SAR_FCLK and starts the save sequence.
2. When the USBHOST power domain is idled (all clocks of the power domain are gated), the PRM initiates the power domain transition to OFF power state by setting the PRCM.PM_PWSTCTRL_USBHOST[1:0] POWERSTATE bit field to 0x0.
3. When the save sequence completes, USBHOST_SAR_FCLK is gated.
4. The power domain sleep transition to OFF power state completes.

4.8.6.1.2 Restore Sequence on Wake-Up Transition

A precondition to the restore sequence on wake-up transition is that the save-and-restore mechanism is enabled (the PRCM.PM_PWSTCTRL_USBHOST[4] SAVEANDRESTORE bit is set to 1). The sequence is initiated when the power domain switches from off to on power state.

1. The PRCM switches the power domain to ON power state.
2. The PRCM asserts the USBHOST domain reset. The functional reset is released locally when the functional clocks are back.
3. The PRCM releases the USBHOST domain reset.
4. The PRCM module activates USBHOST_SAR_FCLK and starts the restore sequence.
5. When the restore sequence completes, USBHOST_SAR_FCLK is gated.
6. The power domain wake-up transition to ON power state completes.

4.8.6.2 USB TLL SAR Sequences

4.8.6.2.1 Save Sequence on Sleep Transition

A precondition to the save sequence on sleep transition is that the save-and-restore mechanism is enabled (the PRCM.PM_PWSTCTRL_CORE[4] SAVEANDRESTORE bit is set to 1). The sequence is initiated when the power domain switches from ON to Open Switch Retention (OSWR) or OFF power state.

1. When the CORE power domain is idled (all clocks of the power domain are gated), the PRM initiates the power domain transition to OFF or OSWR power state by setting the PRCM.PM_PWSTCTRL_CORE[1:0] POWERSTATE bit field to 0x0 or 0x1.
2. The PRCM module activates USBTTL_SAR_FCLK and starts the save sequence.
3. When the save sequence completes, USBTTL_SAR_FCLK is gated.
4. The power domain sleep transition to OFF or OSWR power state completes.

4.8.6.2.2 Restore Sequence on Wake-Up Transition

A precondition to the restore sequence on wake-up transition is that the save-and-restore mechanism is enabled (the PRCM.PM_PWSTCTRL_CORE[4] SAVEANDRESTORE bit is set to 1). The sequence is initiated when the power domain switches from OFF or OSWR to ON power state.

1. 1. PRCM switches the power domain to ON power state.
2. Assert CORE domain reset (must override the functional stall conditions of the reset manager). The functional reset is released locally when the functional clocks are back.
3. Release CORE domain reset.
4. The PRCM module activates the USBTLL_SAR_FCLK and starts the restore sequence.
5. When the restore sequence completes, USBTLL_SAR_FCLK is gated.
6. The power domain wake-up transition to ON power state completes.

4.9 PRCM Interrupts

Table 4-74 lists the interrupts from the PRCM module to the MPU and the IVA2.

Table 4-74. Interrupt Descriptions

Interrupt Name	Mapping	Description
PRCM_MPU_IRQ	M_IRQ_11	To MPU interrupt controller (MPU INTC)
PRCM_IVA_IRQ	IVA2_IRQ[12]	To IVA2.2 wake-up controller (IVA2.2 WUGEN)

Table 4-75 and Table 4-76 summarize the event flags (in PRM_IRQSTATUS_MPU and PRM_IRQSTATUS_IVA2 registers) and mask (in PRM_IRQENABLE_MPU and PRM_IRQENABLE_IVA2 registers) related to the events generating PRCM interrupts to the MPU and the IVA2.

Table 4-75. MPU Interrupt Event Descriptions

Event Flag	Event Mask	Map to	Automatic MPU Domain Wake-Up Event	Description
PRM_IRQSTATUS_ MPU[0] WKUP_ST	PRM_IRQENABLE_ MPU[0] WKUP_EN	PRCM_MPU_IRQ	No	MPU peripheral group wakeup
PRM_IRQSTATUS_ MPU[2] EVGENON_ST	PRM_IRQENABLE_ MPU[2] EVGENON_EN	PRCM_MPU_IRQ	No	Event generator end of on time
PRM_IRQSTATUS_ MPU[3] EVGENOFF_ST	PRM_IRQENABLE_ MPU[3] EVGENOFF_EN	PRCM_MPU_IRQ	No	Event generator end of off time
PRM_IRQSTATUS_ MPU[4] TRANSITION_ST	PRM_IRQENABLE_ MPU[4] TRANSITION_EN	PRCM_MPU_IRQ	No	A sleep or wake-up transition completion event for IVA2, NEON, SGX, DSS, CAM, PER, EMU, USBHOST power domains
PRM_IRQSTATUS_ MPU[5] CORE_DPLL_ST	PRM_IRQENABLE_ MPU[5] CORE_DPLL_ RECAL_EN	PRCM_MPU_IRQ	Yes	DPLL3 recalibration event
PRM_IRQSTATUS_ MPU[6] PERIPH_DPLL_ST	PRM_IRQENABLE_ MPU[6] PERIPH_DPLL_ RECAL_EN	PRCM_MPU_IRQ	Yes	DPLL4 recalibration event
PRM_IRQSTATUS_ MPU[7] MPU_DPLL_ST	PRM_IRQENABLE_ MPU[7] MPU_DPLL_ RECAL_EN	PRCM_MPU_IRQ	Yes	DPLL2 recalibration event
PRM_IRQSTATUS_ MPU[8] IVA2_DPLL_ST	PRM_IRQENABLE_ MPU[8] IVA2_DPLL_ RECAL_EN	PRCM_MPU_IRQ	Yes	DPLL1 recalibration event
PRM_IRQSTATUS_ MPU[9] IO_ST	PRM_IRQENABLE_ MPU[9] IO_EN	PRCM_MPU_IRQ	No	I/O pads wake-up event
PRM_IRQSTATUS_ MPU[22] VC_ SAERR_ST	PRM_IRQENABLE_ MPU[22] VC_ SAERR_EN	PRCM_MPU_IRQ	Yes	Slave address in an I^2C frame sent by the voltage controller not acknowledged by the power IC device
PRM_IRQSTATUS_ MPU[23] VC_ RAERR_ST	PRM_IRQENABLE_ MPU[23] VC_ RAERR_EN	PRCM_MPU_IRQ	Yes	Register address in an I^2C frame sent by the voltage controller not acknowledged by the power IC device

Table 4-75. MPU Interrupt Event Descriptions (continued)

Event Flag	Event Mask	Map to	Automatic MPU Domain Wake-Up Event	Description
PRM_IRQSTATUS_MPU[24] VC_TIMEOUTERR_ST	PRM_IRQENABLE_MPU[24] VC_TIMEOUTERR_EN	PRCM_MPU_IRQ	Yes	Last byte of an I^2C frame issued from the bypass port or from the voltage manager FSM ports sent by the voltage controller not acknowledged by the power IC device
PRM_IRQSTATUS_MPU[25] SND_PERIPH_DPLL_RECAL_ST	PRM_IRQENABLE_MPU[25] SND_PERIPH_DPLL_RECAL_EN	PRCM_MPU_IRQ	Yes	DPLL5 recalibration event

Table 4-76. IVA2 Interrupt Event Descriptions

Event Flag	Event Mask	Map to	IVA2 Domain Wake-Up Event	Description
PRM_IRQSTATUS_IVA2[0] WKUP_ST	PRM_IRQENABLE_IVA2[0] WKUP_EN	PRCM_IVA_IRQ	No	IVA2 peripheral group wakeup
PRM_IRQSTATUS_IVA2[1] FORCEWKUP_ST	PRM_IRQENABLE_IVA2[1] FORCEWKUP_EN	PRCM_IVA_IRQ	No	IVA2 power domain forced wake-up transition completion
PRM_IRQSTATUS_IVA2[2] IVA2_DPLL_ST	PRM_IRQENABLE_IVA2[2] IVA2_DPLL_RECAL_EN	PRCM_IVA_IRQ	Yes	DPLL2 recalibration required

Note: The software must first read the event flag to determine the cause of the interrupt and then write 1 to it to clear the flag.

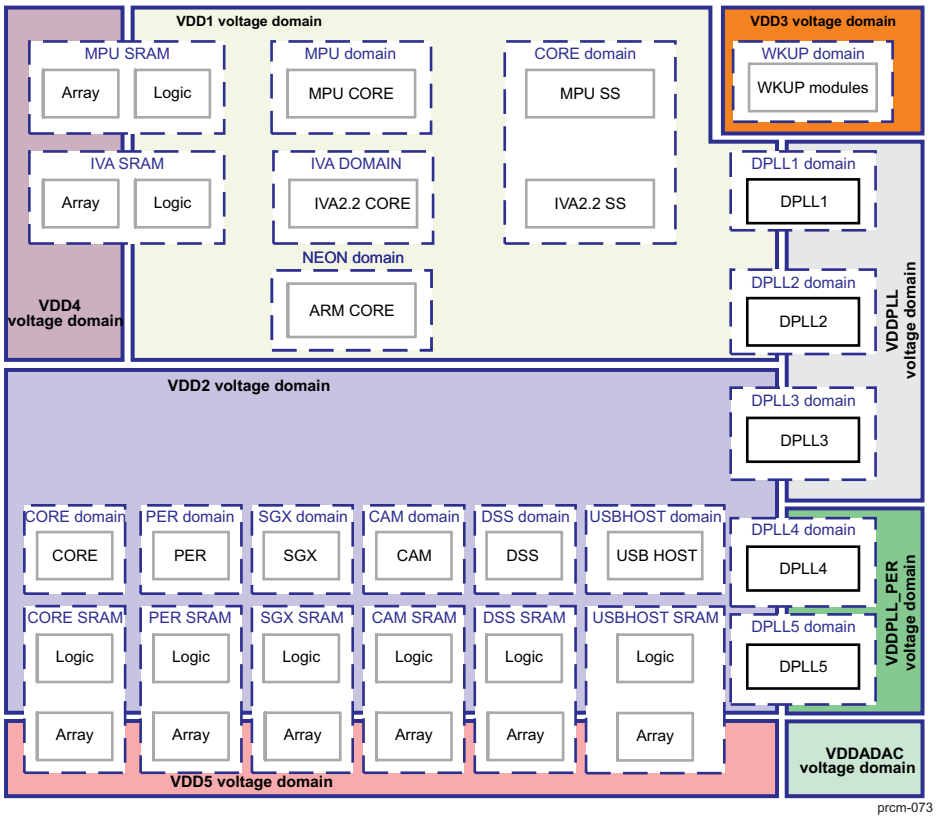
4.10 PRCM Voltage Management Functional Description

This section describes the voltage domains and voltage control architecture. It also explains the interactions between the device and the external power IC.

4.10.1 Overview

Figure 4-73 is an overview of the device voltage domains.

Figure 4-73. Overview of Device Voltage Domains



The device is split into voltage domains:

- Three voltage domains for logic (VDD1, VDD2, and VDD3)
- Two voltage domains for memory (VDD4 and VDD5)
- Two voltage domains for PLLs and analog cells (VDDPLL and VDDPLL_PER)
- Voltage domain for the I/Os

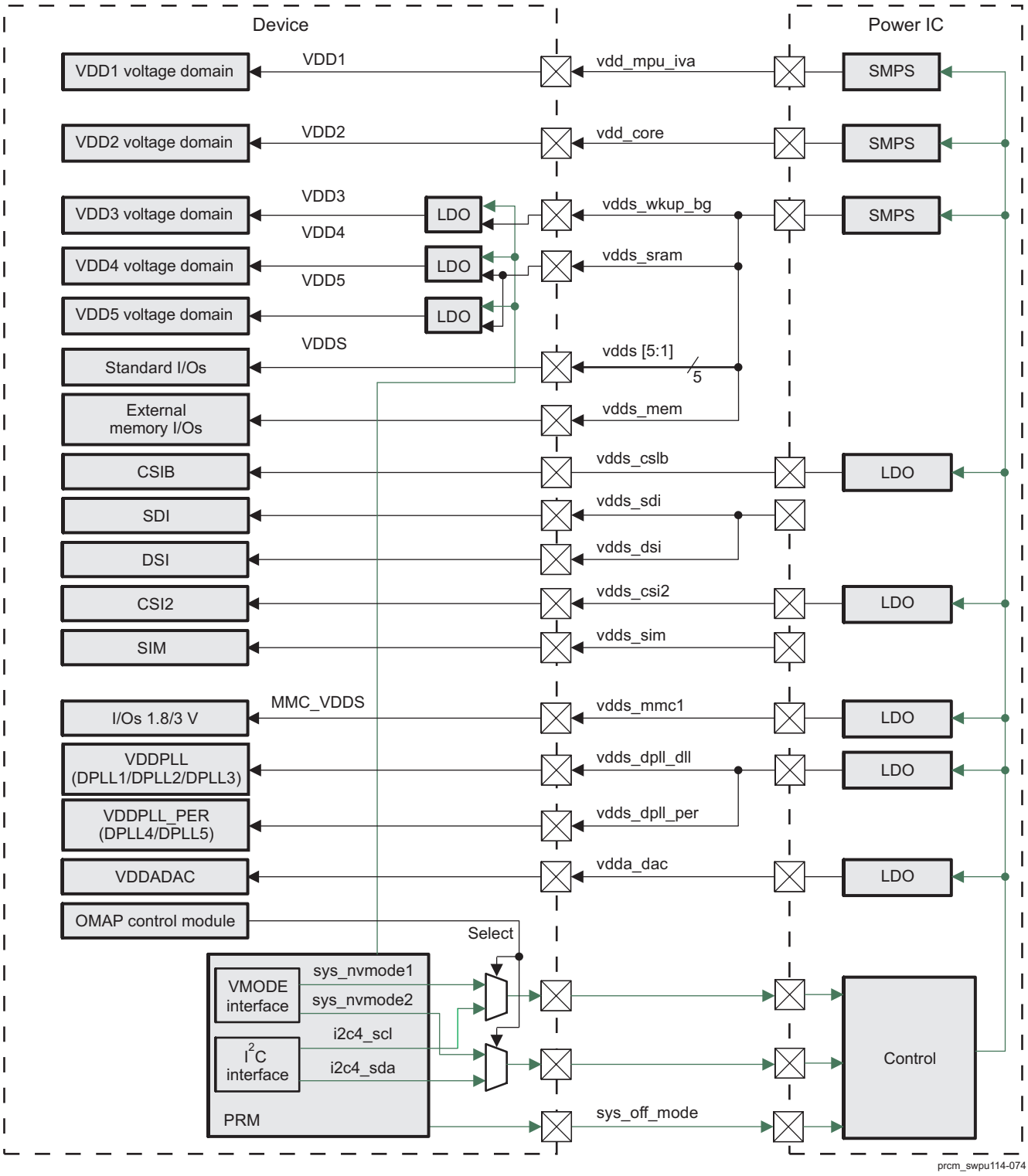
This partition of the voltage domains ensures independent voltage control of each voltage domain through dedicated SMPS or LDO. Functionally, however, voltage control of a voltage domain may depend on the voltage level of another voltage domain; for example, VDD1 voltage-level control may depend on the VDD2 voltage level. Figure 4-73 shows the voltage dependency between domains.

Figure 4-74 is an overview of the device voltage sources and their controls. The PRM directly manages the following voltage sources:

- VDD1: Processor voltage
- VDD2: CORE voltage
- VDD3: Wake-up voltage
- VDD4: Processor SRAM voltage
- VDD5: CORE SRAM voltage

The remaining voltage sources (VDDS, VDDPLL, and MMC-VDDS) are either controlled directly by the external device or software-controlled through an I^2C interface independent of the PRM. Figure 4-74 is an overview of PRCM voltage control.

Figure 4-74. Overview of Device Voltage Distribution



Note: Memory I/Os (1.8 V) are not supplied with the other I/Os. They come directly from the power IC.

4.10.2 Voltage Domains

Table 4-77 summarizes, for each voltage domain, the voltage level corresponding to its OPP when its power domains are on and the voltage level when its power domains are in retention or off power state. The exact values depend on the silicon; for details, see the data manual performance addendum.

Table 4-77. Voltage Domain Controls Summary

Voltage Domain (Power Rail)	Operating Point	Control	Supplied Modules
VDD1 voltage domain (vdd_mpu_iva)	Off	HW and SW	MPU subsystem, IVA2.2 subsystem, digital part of DPLL1 and DPLL2
	Retention		
	OPP1		
	OPP2		
	OPP3		
	OPP4		
	OPP5		
	OPP6 (1)		
VDD2 voltage domain (vdd_core)	Off	HW and SW	Most modules, interconnects, peripherals.
	Retention		digital part of DPLL3, DPLL4, and DPLL5
	OPP1		
	OPP2		
	OPP3		
VDD3 voltage domain (vdds_wkup_bg)	Low-power mode	HW	WKUP and EMU domains
	Normal mode		
	Emulation mode		
VDD4 voltage domain (vdds_sram)	All memories off	HW	Processor memories
	All memories in retention		
	VDD1 is in OPP1, OPP2, or OPP3		
	VDD1 in OPP4 or OPP5		
VDD5 voltage domain (vdds_sram)	Off	HW	CORE, SGX, CAM, DSS, PER, USBHOST, and EMU memories
	Retention		
	VDD2 is in OPP1, OPP2, or OPP3		
VDDS voltage domain (vdds)	Always-on	None	I/Os
VDDADAC voltage domain (vdda_dac)	Software-driven only(2)	SW	Video DAC
VDDPLL voltage domain (vdds_dpll_dll)	Always-on(2)	None	Analog part of DPLL1, DPLL2, and DPLL3
VDDPLL_PER voltage domain (vdds_dpll_per)	Always-on(2)	None	Analog part of DPLL4 and DPLL5
External memory I/Os (vdds_mem)	Always-on (2)	None	External memory I/Os
CSIB (vdds_csib)	Always-on(2)	None	CSIB balls
SDI (vdds_sdi)	Always-on(2)	None	SDI balls
DSI (vdds_dsi)	Always-on(2)	None	DSI balls
CSI2 (vdds_csi2)	Always-on(2)	None	CSI2 balls

(1) OPP6 is available for OMAP3440-720.
(2) Identifies voltage domains that can be switched to 0.0 V when the interface/module is not in use.

Table 4-77. Voltage Domain Controls Summary (continued)

Voltage Domain (Power Rail)	Operating Point	Control	Supplied Modules
SIM (vdds_sim)	Always-on[2]	None	SIM
MMC_VDDS voltage domain (vdds_mmc1)	Always-on[2]	None	MMC1

Notes:

- Logic retention (RFFs and some memory registers OSWR) is implemented only for parts of the CORE power domain. Full device logic retention is also feasible by keeping all power domains on and lowering VDD1 and VDD2 to minimum OPP (CSWR). This consumes more power than the first option, but allows for a fast restart.
- When the user programs all memories in retention, VDD4 and VDD5 are automatically set to RETENTION voltage level by the PRM, which allows the memory arrays to be retained.

4.10.3 Voltage Domain Dependencies

Table 4-78 and Table 4-79 summarize the hardware conditions and the software setting required to switch the VDD1 and VDD2 to their respective low-power states.

Table 4-78. VDD1 Voltage Domain Dependencies

VDD1 Domain State	Power Domains State	DPLL State	sys_clk	Software Control[1]	VDD2	VDD3	VDD4	VDD5
SLEEP	Software controlled power domains of VDD1 are in inactive, retention, or off state.	DPLL1 and DPLL2 are in Stop mode.	N/A	AUTO_SLEEP = 1 AUTO_RET = 0 AUTO_OFF = 0	ON or SLEEP	ON or Overdriven	ON or retention	ON or retention
Retention	Software controlled power domains of VDD1 are in retention or off state and of VDD2 are in INACTIVE, retention or off state.	DPLL1 and DPLL2 are in Stop mode.	N/A	AUTO_SLEEP = 0 AUTO_RET = 1 AUTO_OFF = 0	ON or retention	ON or Overdriven	Retention	ON or retention
Off	Software controllable power domains of VDD1 and VDD2 are in off state.	All DPLLs are in Stop mode.	No longer required	AUTO_SLEEP = 0 AUTO_RET = 0 AUTO_OFF = 1	Off	SLEEP	Off	Off

(1) It is mandatory to program the AUTO_SLEEP, AUTO_RET, and AUTO_OFF bit fields in an exclusive manner.

Table 4-79. VDD2 Voltage Domain Dependencies

VDD2 Domain State	Power Domains State	DPLL State	sys_clk	Software Control[1]	VDD1	VDD3	VDD4	VDD5
SLEEP	Software controlled power domains of VDD1 and VDD2 are in inactive, retention or off state.	All DPLLs are in Stop mode.	No longer required	AUTO_SLEEP = 1 AUTO_RET = 0 AUTO_OFF = 0	SLEEP	ON or Overdriven	ON or retention	ON or retention
Retention	Software controlled power domains of VDD1 are in inactive, retention or off state and of VDD2 are in retention or off state.	All DPLLs are in Stop mode.	No longer required	AUTO_SLEEP = 0 AUTO_RET = 1 AUTO_OFF = 0	ON or retention	ON or Overdriven	ON or retention	Retention
Off	Software controllable power domains of VDD1 and VDD2 are in off state.	All DPLLs are in Stop mode.	No longer required	AUTO_SLEEP = 0 AUTO_RET = 0 AUTO_OFF = 1	Off	SLEEP	Off	Off

(1) It is mandatory to program the AUTO_SLEEP, AUTO_RET, and AUTO_OFF bit fields in an exclusive manner.

Notes:

- In voltage domain SLEEP state, the current load decreases because of reduced activity level; however, the voltage level remains that of on state.
- In voltage domain overdriven state, the domain voltage is raised above the nominal operating voltage to boost performance; however, power consumption also increases.
- AUTO_SLEEP, AUTO_RET and AUTO_OFF, in the Software Control column, refer to the PRM_VOLTCTRL[0] AUTO_SLEEP, PRM_VOLTCTRL[1] AUTO_RET and PRM_VOLTCTRL[2] AUTO_OFF bits.

Table 4-80. Remaining Voltage Domain Dependencies

Voltage	Condition
VDDS	No dependency with other device voltages
External memory I/Os	
VDDADAC	
VDDPLL	
VDDPLL_PER	
CSIB	
SDI	

Table 4-80. Remaining Voltage Domain Dependencies (continued)

Voltage	Condition
DSI	
CSI2	
SIM	
MMC_VDDS	

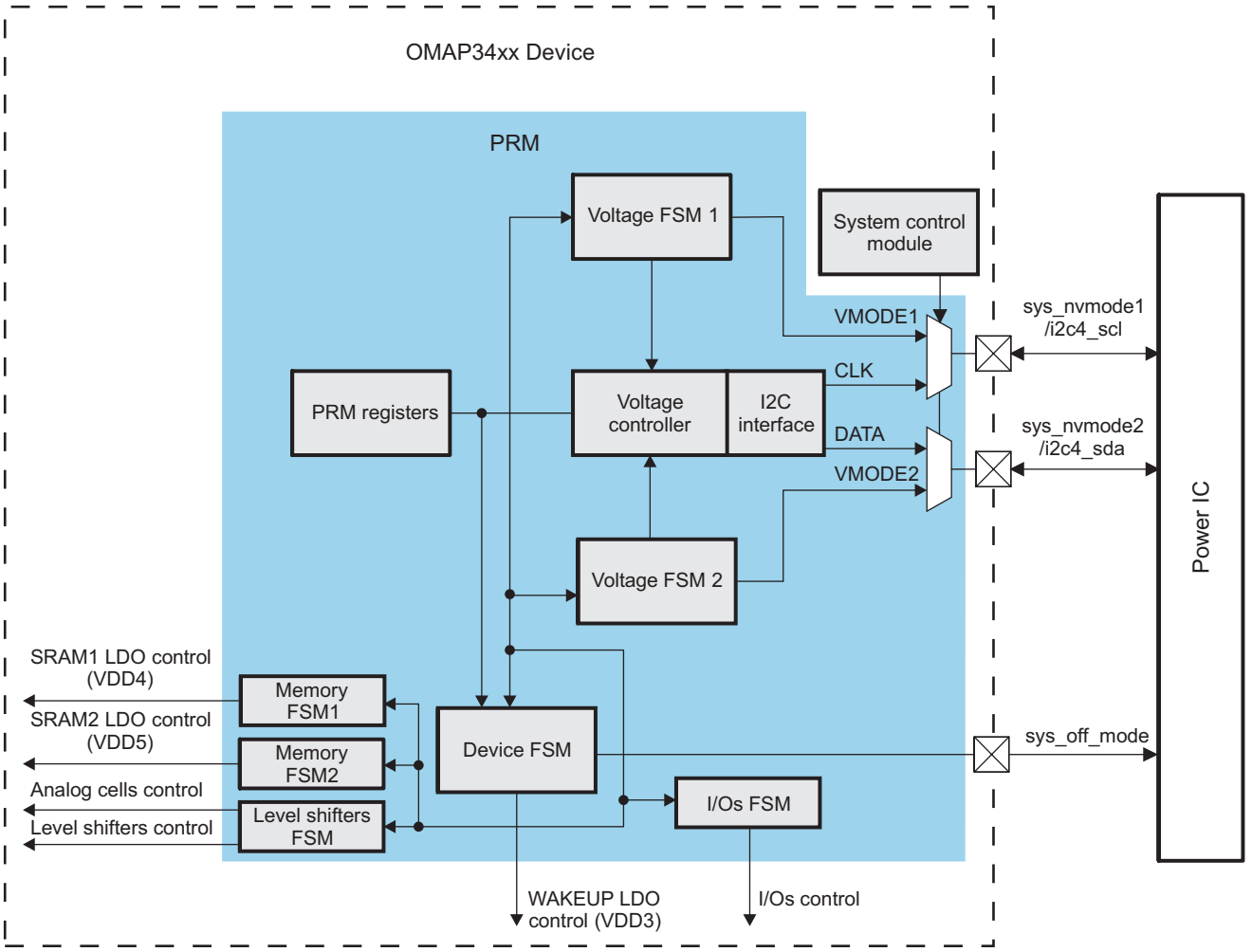
4.10.4 Voltage-Control Architecture

The PRM is split over several blocks that manage the different voltage sources.

- VDD1 and VDD2 voltages are managed by multiple control sources:
  - Direct software control of VDD1 and VDD2 is also possible by writing to the registers of the voltage controller, which then sends the voltage commands to the power IC through the dedicated I^2C interface.
  - Two voltage finite state-machines (FSM 1 and FSM 2) can also manage the VDD1 and VDD2 voltages either by sending commands to the voltage controller (I^2C mode) or by sending the sys_nvmode1 and sys_nvmode2 control signals (direct-control mode). The FSMs control voltage-level switching, based on the power state of the device.
- The FSMs in the PRM also control SRAM and wake-up LDOs, sleep mode for analog cells, and level shifters.
- A device FSM sequences the memory, wakeup, voltage, and I/O FSMs during the device off, sleep, and wake-up transitions.

Figure 4-75 shows the architecture for PRM voltage control.

Figure 4-75. PRM Voltage Control Architecture



* The green region in the figure represents the boundary of the PRM

prcm_swpu114-075

CAUTION

It is highly recommended to use the embedded dedicated I2C4 instead of VMODE, which is a legacy mode. The I2C4 provides greater flexibility and higher efficiency in terms of power optimization.

4.10.5 VDD1 and VDD2 Control

The main power-supply sources, VDD1 and VDD2, can be controlled using mutually exclusive modes specified in the SCM:

- Direct control with VMODE signals or I2C interface
- Dedicated I^2C control

Because both control modes are multiplexed on the same device pins, direct control signals can be used alternately with I^2C control signals. The muxes are managed by the SCM.

4.10.5.1 Direct Control With VMODE Signals

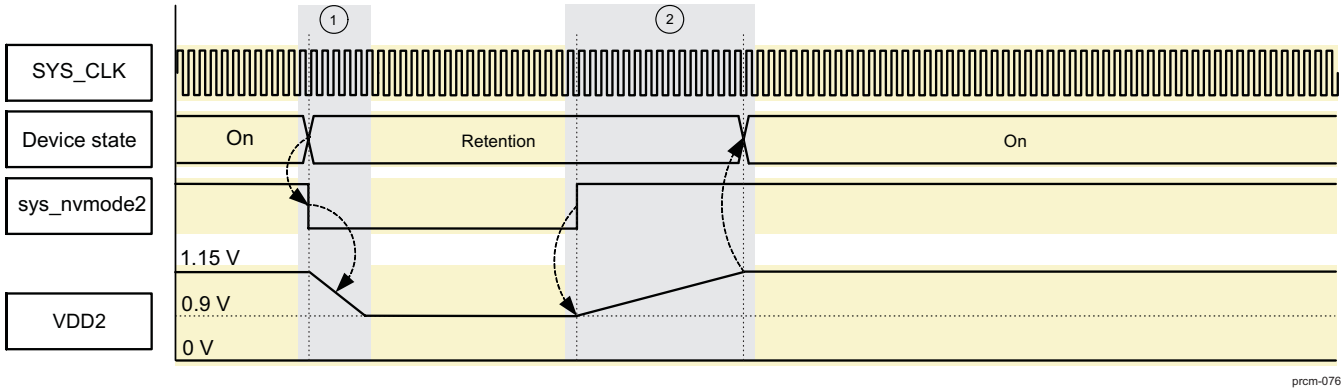
In direct-control mode, the VDD1 and VDD2 voltage sources can be controlled by the voltage FSMs in the PRM to trigger a voltage transition based on the power domain states. The voltage FSMs use a VMODE interface to send voltage commands through the sys_nvmode1 and sys_nvmode2 pins of the device to the external power IC. The simple voltage-control commands of the VMODE interface can be used to set two voltage values (VDD1 and VDD2) per voltage domain .

The sys_nvmode[1,2] signals request new voltage levels to the external SMPS. The polarity of the sys_nvmode[1,2] signals from the PRM is configured by the PRCM.PRM_POLCTRL[0] EXTVOL_POL bit, which is active-low by default. When the signal goes low, a lower voltage is supplied, and when it goes high, a higher voltage is supplied.

The VMODE voltage control is enabled by setting the PRM.PRM_VOLTCTRL[4] SEL_VMODE bit, and is triggered when the device switches between low-power (retention or off) mode and normal (on) mode.

Figure 4-76 is an example of a VDD2 voltage transition from an operational voltage at 1.15 V (OPP3) to the lowest functional voltage of the device (0.9 V). In this example, the PRCM module is programmed to automatically lower sys_nvmode2 when the device enters sleep mode. The power IC is programmed to ramp the VDD2 voltage down to 0.9 V when sys_nvmode2 is deasserted and to ramp up the voltage to 1.15 V when sys_nvmode2 is asserted. The PRCM module must also wait for the VDD2 regulator to stabilize when sys_nvmode2 is deasserted and asserted. This stabilization time is programmable by software using the PRCM.PRM_VOLTSETUP1 register.

Figure 4-76. Voltage Transition Controlled by sys_nvmode2



Internally, the PRM manipulates two logical voltage levels (L0 and L1) for each sys_nvmode[1,2] signal. One level is set when all the power domains in the voltage domain are in retention or off mode, while the other is set when a power domain is on. The power IC must be configured to switch to low voltage when the associated sys_nvmode[1,2] signal is received, and vice versa.

For example, if L0 is the lower voltage and an active-low polarity (default setting) on the sys_nvmode[1,2] signal:

- L0 logical level sys_nvmode[1,2] = 1 VDD = 1.15 V
- L1 logical level sys_nvmode[1,2] = 0 VDD = 0.9 V

The polarity of the logical voltage-level signal from the PRM can be controlled by the PRCM.PRM_POLCTRL[0] EXTVOL_POL bit. It is active-low by default (L0 = 1, L1 = 0).

4.10.5.2 Direct Voltage Control With I^2C Interface

PRM can send VDD1 and VDD2 sleep commands to the power IC through the dedicated I^2C. This allows the power IC to activate sleep mode (the voltage regulator switches in sleep mode, where voltage is maintained but only small load is supported). This also allows external power IC to reduce its power consumption.

4.10.5.3 Voltage Controller and Dedicated I²C Interface

The central modules for the I²C control mode are the voltage controller and the I²C interface. The voltage controller receives voltage control commands from input ports:

- Voltage FSM 1 and 2 ports: The voltage FSMs send commands to the voltage controller when the device enters retention or off power state, and also when the device wakes up.
- PRM register port: The PRM registers give direct software control over the voltage levels of the VDD1 and VDD2 voltage domains.

An arbitration scheme in the voltage controller manages concurrent requests on multiple ports.

Externally, the voltage controller interfaces to a power IC through a dedicated I²C interface (I2C4). To reduce the latency of voltage changes, the voltage controller is configurable to run in HS I²C mode.

Each internal port has a handshake to indicate when the I²C frame that results from the request on that port is acknowledged by the external power IC.

4.10.6 Analog Cells, LDOs, and Level Shifter Controls

In addition to the VDD1 and VDD2 voltage controls, the PRM handles the following operations automatically, based on hardware conditions, without any software control:

- Reduces SRAM LDOs voltage when all memories are in retention
- Reduces wake-up LDO voltage when the device enters off mode (wake-up leakage reduction)
- Increases wake-up LDO voltage when emulation trace is active
- Actively isolates level shifters during VDD1 and VDD2 removal
- Activates sleep mode in all analog cells when the device enters off mode

4.10.6.1 SRAM Voltage Control

The two embedded SRAM LDOs supply regulated voltage (VDD4 or VDD5) to memory banks. The processor memory LDO (VDD4) has three reference voltages, while the CORE memory LDO (VDD5) supports two:

- 1.2 V is the normal voltage reference, used for processors OPP1, OPP2, and OPP3.
- Tracking between 1.2 V and 1.35 V: The processor SRAM LDO (VDD4) tracks and follows VDD1 voltage when it exceeds OPP3 (1.2 V) nominal voltage. VDD1 (up to 1.35 V) is the overdrive voltage reference when processors operate at OPP4 and OPP5. This option is not supported by the LDO (VDD5) of the CORE memory.
- 1.0 V is set when all memory banks belonging to the LDO are in retention state. This allows reducing the memory arrays to 0.6 V to minimize leakage.

When not used (all memories are off), the LDO is shut down. These modes are managed automatically by hardware (PRM).

4.10.6.2 Wake-Up and Emulation Voltage Control

The embedded wake-up LDO (VDD3) supplies both WKUP and EMU power domains. It is permanently active and feeds the WKUP power domain continuously. An embedded switch allows the PRM to control power to the EMU power domain. This switch is closed on a software request command when a debug session starts, or automatically on JTAG plug detection.

This LDO has three reference voltages:

- 1.2 V is the normal voltage reference, used in device active mode.
- 1.35 V is the overdrive voltage reference used when emulation is activated and MPU emulation trace is required.
- 1.0 V is set when the device is in low-power (off) mode, to minimize leakage.

These modes are managed automatically by hardware.

4.11 PRCM Off-Mode Management

4.11.1 Overview

The CORE power domain can be switched from ON to OFF or retention state.

When the CORE power domain is inactive, the clock manager does not generate an interface clock, and device interconnects are inactive; therefore, the device is effectively in off mode. In retention state, however, the logic in the CORE domain is retained, and the device wake-up latency is small compared to the latency when the CORE power domain is in off state. In waking from the CORE domain off state, the CORE logic configuration must be loaded from a scratchpad memory in the WKUP power domain.

Note: Although some modules in the device can be kept active while the CORE power domain is off or in retention, this is not recommended.

When the device is put in off mode (the CORE power domain is in retention or off state), the device voltage domains can be switched off to minimize leakage currents. However, reactivity to wake-up events must be ensured.

The device supports a unique off-mode management scheme that allows deactivation of all modules, isolation of their outputs, and configuration of a dedicated off mode on the I/O pads of the device, to listen to wake-up events. A wake-up event on any I/O pad is detected by the PRM (in WKUP always-on power domain), which can then wake up the device.

4.11.2 Device Off-Mode Configuration

4.11.2.1 Overview

Any I/O pad of the device can be configured to generate a wake-up event when the device is in off mode. The off-mode scheme is based on the following components of the device:

- I/O pads
- SCM

Figure 4-77 is an overview of the I/O pad off-mode scheme. In this mode, the I/O pads of the device form a daisy chain. The I/O pad logic at the two ends of the chain is connected to the PRM.

When a wake-up event (WUEVT) occurs on an enabled I/O pad in the chain, the event is propagated through the chain to the PRM. Multiple I/O pads can receive a wake-up event simultaneously; however, the PRM receives only the OR output for all the enabled I/O pads. When the device wakes up, the MPU can determine all sources of the current wake-up event logged into the corresponding CONTROL.CONTROL_PADCONF_<IOpad>[15] WAKEUPEVENT0 or CONTROL.CONTROL_PADCONF_<IOpad>[31] WAKEUPEVENT1 bit in the SCM .

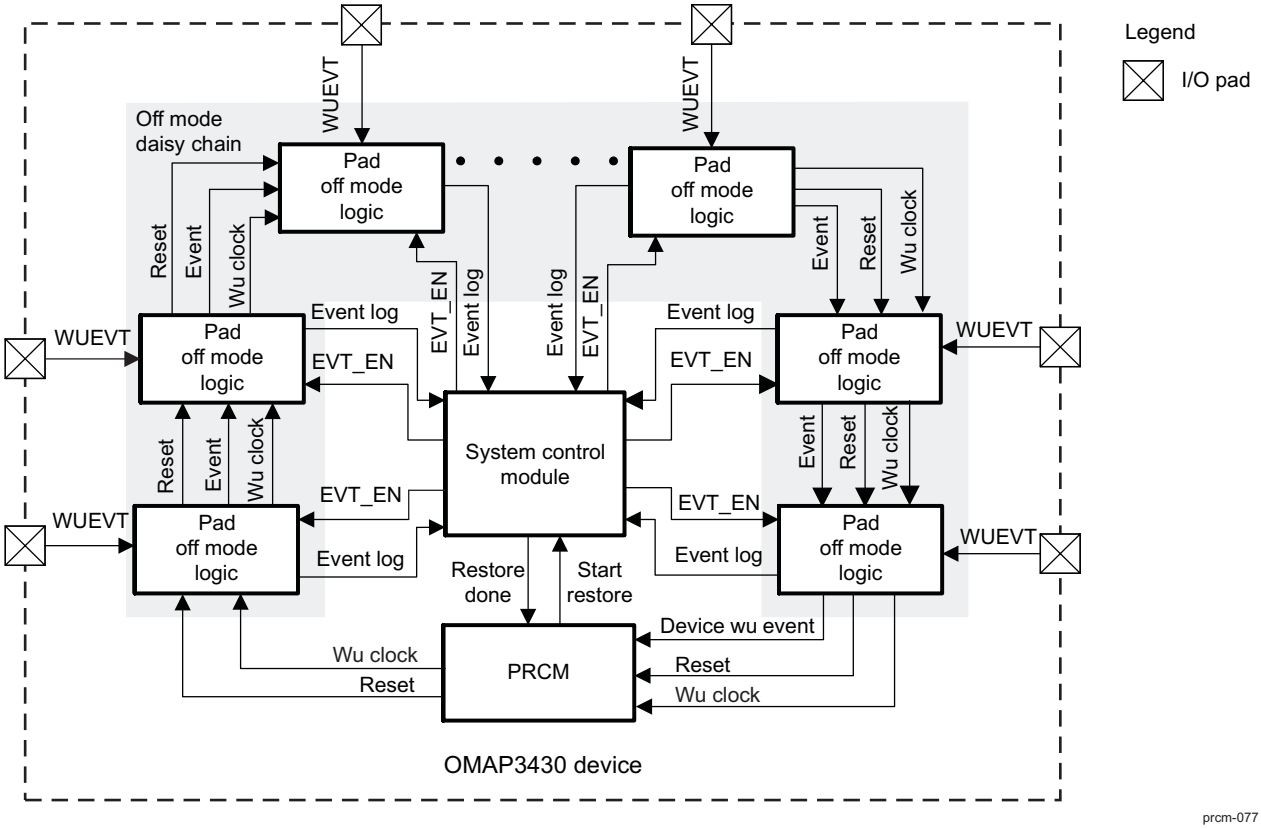
The I/O pad wake-up scheme must be enabled globally by setting the PRCM.PM_WKEN_WKUP[8] EN_IO bit and the PRCM.PM_WKEN_WKUP[16] EN_IO_CHAIN bit. The wake-up event from each I/O pad of the device can be individually enabled/disabled (EVT_EN signal) by writing to the CONTROL.CONTROL_PADCONF_<IOpad>[14] WAKEUPENABLE0 or CONTROL.CONTROL_PADCONF_<IOpad>[30] WAKEUPENABLE1 bit in the SCM.

For information about the SCM, see Chapter 7, *System Control Module*.

Note: As explained previously (see Section 4.8, *Idle and Wake-Up Management*), module-specific wake-up events other than the I/O pad wake-up scheme can wake up the device from off mode. For example, the GPIO pads in the WKUP power domain can also wake up the device.

Note: For the wake-up features of the GPIO pads , see Chapter 25, *General-Purpose Interface*.

Figure 4-77. Device Off-Mode Control Overview



4.11.2.2 I/O Wake-Up Mechanism

There are two ways to capture an I/O event. The expected way is to use GPIO modules. However, these modules may be powered-off when there is a need to capture an I/O event. The second way to capture an I/O event is through an I/O wake-up scheme controlled by the I/O daisy chain.

Only one GPIO module is in the WKUP domain, which is not switchable, and the other GPIO modules are in the PER domain, which is switchable. Therefore, the I/O wake-up scheme for the corresponding I/O must be enabled before transitioning the PER domain to a nonfunctional state. This typically happens when the device is programmed to enter OFF state.

During off mode, only six pins of the GPIO module in the WAKEUP domain are wake-up capable; therefore, the I/O wake-up scheme must be enabled for the other 26 I/Os before transitioning to OFF state.

The I/O wake-up scheme can be enabled when the PER domain is in ACTIVE state.

Software must enable and disable the I/O wake-up scheme and ensure an overlap window between the GPIO wake-up capability and the I/O wake-up capability. Software must manage this overlap window.

The I/O wake-up scheme in each pad is enabled through the I/O daisy chain:

- The I/O wake-up capability is enabled by programming the corresponding register (CONTROL.CONTROL_PADCONF_X) in the control module.
- The global I/O wake-up enable bit is enabled by programming a dedicated register (PRCM.PM_WKEN_WKUP[8] EN_IO) in the PRCM module.
- The I/O wake-up scheme is enabled by triggering the I/O daisy chain control (Wu clock) by programming a dedicated register (PRCM.PM_WKEN_WKUP[16] EN_IO_CHAIN) in the PRCM module.

Software must wait for the I/O daisy chain to complete before it transitions the PER domain to a nonfunctional state. This is done by polling a dedicated status bit in the PRCM module (PRCM.PM_WKST_WKUP[16] ST_IO_CHAIN). This status bit must be cleared by software when the bit is read to 1.

The I/O wake-up scheme in each pad is disabled through the I/O daisy chain:

- The I/O wake-up scheme is disabled by programming the dedicated register (PRCM.PM_WKEN_WKUP[16] EN_IO_CHAIN) in the PRCM module.
- The global I/O wake-up enable bit is disabled by programming a dedicated register (PRCM.PM_WKEN_WKUP[8] EN_IO) in the PRCM module.
- The I/O wake-up scheme is disabled by triggering the I/O daisy chain control (Wu clock) by writing 1 in a dedicated register (PRCM.PM_WKST_WKUP[8] ST_IO) in the PRCM

  This bit is also used to log the source of the I/O wakeup. Writing 1 in this register clears this bit to 0.

4.11.3 CORE Power Domain Off-Mode Sequences

The I/O pad wake-up scheme can be activated to detect wake-up events when the CORE power domain is switched to inactive, retention, or off state. The off state sequences for the CORE power domain include the sequence for going from on state to retention or off power state (while enabling the I/O pad wake-up scheme), and then switching back to the on power state (while disabling the I/O pad wake-up scheme) on an I/O wake-up event.

4.11.3.1 Sleep Sequences (Transition From On to Retention/Off)

When the CORE power domain is in retention state, the logic is switched off, its context is saved by the RFFs in the modules, and the memory blocks are retained. In off state, however, the logic and memories are switched off, and the RFFs are not saved.

Note: If the PER power domain is kept on while the CORE power domain is in retention state, all I/O wake-up enable (EVT_EN signals) signals to the I/O pads related to the PER power domain inputs must be disabled (by clearing the CONTROL.CONTROL_PADCONF_<IOpad>[14] WAKEUPENABLE bit in the SCM). in this case, wake-up events are generated by the PER domain and not through the daisy chain.

For information about the SCM, see Chapter 7, *System Control Module*.

The following sequence is used to go from on to retention or off state:

1. The software sets the PRCM.PM_WKEN_WKUP[8] EN_IO bit and the PRCM.PM_WKEN_WKUP[16] EN_IO_CHAIN bit to enable the I/O pad wake-up scheme.
2. The MPU initiates the sleep sequence. When all conditions are met, the CORE power domain clocks are shut down. At this stage, most of the pads are inactive, but some, such as the PER and DSS power domains, can stay active.
3. The PRM initializes and resets the I/O wake-up detection scheme and saves all RFFs. Its CORE power domain output is isolated from the I/O pads.
4. The PRM switches the CORE power domain to retention/off power state. The I/O configuration is saved on wake-up power domain registers; see Section 7.4.4.4.1, *Save-and-Restore Mechanism*, in Chapter 7, *System Control Module*, for more information.
5. The PRM waits for a wake-up event from the daisy chain. Other possible wake-up sources can be a wake-up event from a module in the wake-up power domain or from a global warm reset.

Note: When the CORE power domain is in retention state, VDD2 voltage must be at retention or on voltage level to maintain stable CORE power domain output values to the I/Os.

4.11.3.2 Wake-Up Sequences (Transition From Retention/Off to On)

The PRM detects a wake-up event from the daisy chain. Several wake-up events can occur simultaneously, but PRM only sees the ORing of them. Depending on whether the CORE power domain is in retention or off state, the following sequence is followed to wake up the CORE and MPU power domains:

1. When the CORE power domain wakes up from off/retention power state, reset is asserted for both the MPU and the CORE power domains.
2. The MPU and the CORE power domains are switched to ON power state.
3. The domain clocks of the MPU and the CORE power domains are restarted.
4. If the CORE power domain was in the off power state, its reset is released before the MPU power domain.
   - The hardware restores the I/O pad configuration and some configuration registers of the control module. This configuration has been saved in the scratchpad memory (by the control module save procedure initiated by software control prior to the transition to off power state).
   - The PRCM module switches the I/O pad configurations to the normal restored configuration (The I/O pad isolation is released).
   - The reset is released for MPU power domain.
5. If the CORE power domain was in the retention power state, the reset is released for both the MPU and the CORE power domains at the same time.
6. When the MPU boots, the software accesses the SCM to read the wake-up event source (identified by the CONTROL.CONTROL_PADCONF_< IOpad>[15] WAKEUPEVENT bits in the SCM corresponding to all enabled pads).
7. The software disables the wake-up daisy chain by clearing the PRCM.PM_WKEN_WKUP[8] EN_IO bit and the PRCM.PM_WKEN_WKUP[16] EN_IO_CHAIN bit.

Note: A daisy-chain wake-up event always causes the MPU to restart and boot. Other independent wake-up events activated in off mode (for example, a GPIO wake-up event) always cause the CORE power domain to be activated. The MPU is wakened if a wake-up dependency has been set by the user, or if the modem generates an interrupt to the MPU.

4.11.4 Device Off-Mode Sequences

Device off-mode sequencing requires preliminary settings that can be done during device initialization. The following actions are performed once and remain valid for all device off/on transitions:

1. Configure valid off and active pad configuration for each pad by programming the SCM CONTROL.CONTROL_PADCONF_< IOpad> register.
2. Save the active pad configuration by asserting the CONTROL.CONTROL_PADCONF_OFF[1] STARTSAVE bit in the SCM. This process can be repeated each time the active pad configuration of the device is changed.
3. Set the SCM to smart-idle mode by configuring the CONTROL. CONTROL_SYSCONFIG[4:3] IDLEMODE bit field. This ensures that the module clocks remain active during the configuration save procedure.
4. Disable all the modules running on system clock or switch them on the 32 kHz clock when required.
5. Configure the voltage regulator setup times (the PRCM.PRM_VOLTSETUP1[15:0] SETUP_TIME1 and PRCM.PRM_VOLTSETUP1[31:16] SETUP_TIME2 bit fields).
6. Set sys_off_mode deassertion time in the PRCM.PRM_VOLTOFFSET[15:0] OFFSET_TIME bit field.
7. Configure the system clock oscillator setup time in the PRCM.PRM_CLKSETUP[15:0] SETUP_TIME bit field.

For information about the SCM, see Chapter 7, *System Control Module*.

4.11.4.1 Sleep Sequences

The sleep sequences exemplify the two types of device transitions from ON to OFF mode:

- Device off-mode transition without using the SYS_OFF_MODE signal

- Device off-mode transition using only the SYS_OFF_MODE signal

4.11.4.1.1 Device Off-Mode Transition Without Using the SYS_OFF_MODE Signal

Each time the device enters I/O-pad-off configuration mode, the following sequence must be performed:

1. Software sets the PRCM.PM_WKEN_WKUP[8] EN_IO bit and the PRCM.PM_WKEN_WKUP[16] EN_IO_CHAIN bit to enable the I/O wake-up scheme.
2. The MPU initiates the sleep sequence. When all sleep conditions are met, all domain clocks are shut down. At this stage, all output pads are inactive and static.
3. The PRM configures the I/O daisy-chain for the wake-up detection scheme.
4. The CORE domain output is isolated.

   When conditions for entering off mode are met, the PRM proceeds with the sequence.
5. It switches off all domains, including the CORE power domain, and then switches from active mode pad configuratiON to OFF mode pad configuration.
6. It isolates the pads before the removal of VDD1 and VDD2.
7. It shuts down all analog cells (DPLL, DLL).
8. It switches to off all the DPLL power domains .
9. It sends the off command for VDD1 to the voltage controller
10. It shuts down the SRAM LDOs.
11. The daisy chain is now ready to detect I/O pad wake-up events.
12. The PRM sends the off command for VDD1 to the voltage controller. Through I2C4, the voltage controller requests the power IC to set VDD1 to the voltage corresponding to the PRM_VC_CMD_VAL_0[7:0] OFF bit field. (A legacy mode exists where vmode pins can be used instead of I2C4, depending on the PRM_VOLTCTRL[4] SEL_VMODE bit. In this case, the PRM asserts sys_nvmode1 and the power IC updates VDD1 to its previously programmed value.)
13. Upon reception of the I2C4 transaction acknowledge, and after waiting for the VDD1 regulator setup time counter to expire (the PRM_VOLTSETUP1[15:0] SETUPTIME1 bit field), the PRM sends the off command for VDD2 to the voltage controller. Through I2C4, the voltage controller requests the power IC to set VDD2 to the voltage corresponding to PRM_VC_CMD_VAL_1[7:0] OFF bit field. (If vmode legacy mode is used, the PRM asserts sys_nvmode2 and the power IC restores VDD2 to its previously programmed value.)
14. Upon reception of the I2C4 transaction acknowledge, and after waiting for the VDD2 regulator setup time counter to expire (the PRM_VOLTSETUP1[31:16] SETUPTIME2 bit field), the PRM gates the internal system clock.
15. It releases sys_clkreq and disables the system clock oscillator (if used).
16. The PRM ramps down the wake-up LDO to 1 V.
17. It waits for a wake-up event from the daisy chain or from an internal event (timer) from the WKUP domain.

Note: If the I/O pad toggles when the I/O daisy-chain is already configured for the wake-up detection scheme, and before OFF mode transition occurs, a not real wake-up event is logged in its PADCONF register.

4.11.4.1.2 Device Off-Mode Transition Using Only the SYS_OFF_MODE Signal

Depending on the external power IC capability and the user setting, VDD1 and VDD2 voltage control may use the direct sys_off_mode signal and not send the commands through the I^2C interface.

in this case, the initial sequence for device off-mode transition is the same as in Section 4.11.4.1.1, *Device Off-Mode Transition Without Using the SYS_OFF_MODE Signal*, up to Step 8. After this, the following steps are executed:

1. It shuts down the SRAM LDOs.
2. It asserts sys_off_mode. At this point, VDD1 and VDD2 are shut down.
3. It ramps down the wake-up LDO to 1 V.

4. It releases CLKREQ and disables the system clock oscillator (if used)
5. It waits for a wake-up event from the daisy chain.

4.11.4.2 Wake-Up Sequences

Note: vdds_dpll_per and vdds_dpll_dll can be ramped up any time before VDD1 ramps up.

4.11.4.2.1 Device Wakeup from Off Mode Without Using the SYS_OFF_MODE Signal

On detection of a wake-up event from the daisy chain, the PRM performs the following steps:

1. It enables the system clock oscillator, asserts sys_clkreq, and ramps up the wake-up LDO.
2. It waits for the oscillator set-up time and the wake-up LDO stabilization time.
3. It sends the on command for VDD2 to the voltage controller. Through I2C4, the voltage controller requests the power IC to set VDD2 to set the voltage corresponding to the PRM_VC_CMD_VAL_1[31:24] ON bit field. (A legacy mode exists where vmode pins can be used instead of I2C4, depending on the PRM_VOLTCTRL[4] SEL_VMODE bit. In this case, the voltage controller deasserts sys_nvmode2 and the power IC restores VDD2 to its previously programmed value.) At this point, VDD2 ramps up (the reset is not yet asserted on VDD2 logic).
4. It waits for the VDD2 regulator setup time counter to expire (PRM_VOLTSETUP1[31:16] SETUPTIME2).
5. It sends the on command for VDD1 to the voltage controller. Through I2C4, the voltage controller requests the power IC to set VDD1 to set the voltage corresponding to the PRM_VC_CMD_VAL_0[31:24] ON bit field. (If vmode legacy mode is used, the voltage controller deasserts sys_nvmode1 and the power IC restores VDD1 to its previously programmed value). At this point, VDD1 ramps up (the reset is not yet asserted on VDD1 logic).
6. It waits for the VDD1 regulator setup time counter to expire (PRM_VOLTSETUP1[15:0] SETUPTIME1).
7. It restarts the memory LDOs.
8. It powers up all analog cells on the VDD2 domain and waits for the settling time of the LDO and analog cells (internal counter).
9. The following steps can occur in parallel:
   - The DPLL, CORE, and MPU power domains are powered up.
   - The reset on the eFuse controller, DPLL, CORE, , and MPU power domains is asserted.
10. When the CORE, MPU, DPLLs are on, the PRM releases the eFuse controller reset, and the eFuse scan starts.
11. When the scan completes, the DPLL resets are released. The DPLLs are in bypass and the clocks start flowing (DPLL1 and DPLL3).
12. When the CORE and MPU domains are on, the PRM releases their corresponding domain output isolations.
13. When the CORE power domain is on and DPLL3 is in bypass, the reset timer for the CORE power domain starts.
14. When the MPU power domain is on and DPLL1 is in bypass, the reset timer for the MPU power domain starts.
15. When the CORE reset time expires, the CORE domain reset is released.
16. When the CORE domain exits reset and the SCM context and I/O configuration are restored from ScratchPad, memory starts.
17. When the PRM receives the acknowledge from the SCM, it releases the MPU and CORE domain output isolations and the I/O isolation.
18. The PRM switches the I/O pad configuration from off mode configuration to active mode configuration.
19. When the MPU reset time expires and pad configuration returns to normal mode, the MPU domain reset is released.
20. When the MPU boots, the software accesses the control module to read the wake-up event source.
21. The software disables the I/O wake-up daisy chain by clearing the PRCM.PM_WKEN_WKUP[8] EN_IO bit and the PRCM.PM_WKEN_WKUP[16] EN_IO_CHAIN bit.

4.11.4.2.2 Device Wakeup From Off Mode Using Only the SYS_OFF_MODE Signal

Depending on the external power IC capability and the user setting, VDD1 and VDD2 voltage control use the direct sys_off_mode signal and do not send commands through the I^2C interface.

in this case, the first part of the sequence in Section 4.11.4.2.1 is as follows.

Upon detection of a wakeup from the daisy chain, the PRM performs the following steps:

1. It enables the system clock oscillator and asserts sys_clkreq.
2. It ramps up the wake-up LDO.
3. It releases sys_off_mode.
4. It waits for the system clock oscillator setup time.
5. It waits for VDD1 and VDD2 setup time. The two counts are started in parallel.
6. When both timers expire, the following steps happen in parallel:
   - Reset is asserted on the MPU and CORE power domains.
   - The MPU and CORE power domains are powered up.
   - The PRM restarts the CORE and processor memory LDOs.
   - The PRM powers up all analog cells in the VDD1 and VDD2 domains.

4.12 PRCM Basic Programming Model

The PRCM module supports an extensive set of module-specific registers that allow programming control over numerous features of the clocks, resets, and power-management signals for each power domain of the device.

These registers are fully programmable and accessible by the MPU and the IVA2.2 subsystems.

Logically, the registers are grouped into five categories:

- Global
- Clock management
- Reset management
- Power management
- Voltage management

4.12.1 Global Registers

4.12.1.1 Revision Information Registers

- CM_REVISION: Indicates the CM module revision code; it is read-only
- PRM_REVISION: Indicates the PRM module revision code; it is read-only

4.12.1.2 PRCM Configuration Registers

- CM_SYSCONFIG: Holds an AUTOIDLE bit to control the CM internal clock autogating feature
- PRM_SYSCONFIG: Holds an AUTOIDLE bit to control the PRM internal clock autogating feature

4.12.1.3 Interrupt Configuration Registers

The PRM can interrupt the MPU and the IVA2.2 subsystems as a result of four events:

- PRM internal event (event generator, sleep transition, wake-up transition, voltage controller)
- Peripheral wake-up event for a peripheral with interrupt capability (GPTIMER[1..11], GPIO[1..6], McBSP[1..5], UART[1..3], HS USB OTG, I2C[1..3], McSPI[1..4], MMC[1,2])
- Module/device-level event not associated with any interrupt (DPLL recalibration, I/O wakeup)

The PRM interrupt is enabled by programming the PRM_IRQENABLE_<processor_name> register; the interrupt status can be read from the PRM_IRQSTATUS_<processor_name> register.

The device has four processor interrupt registers:

- PRM_IRQSTATUS_MPU
- PRM_IRQENABLE_MPU
- PRM_IRQSTATUS_IVA2
- PRM_IRQENABLE_IVA2

4.12.1.3.1 MPU Interrupt Event Sources

The MPU interrupt registers correspond to the interrupt sources connected to the interrupt line mapped to the MPU interrupt controller.

Multiple events can activate this interrupt line:

- MPU peripheral group wake-up event
- Event-generator module end-of-on time and end-of-off time events
- Sleep or wake-up transition (SGX, USBHOST, IVA2, PER, DSS, CAM, NEON, EMU power domains)
- DPLL1/DPLL2/DPLL3/DPLL4/DPLL5 recalibration request
- I/O pad wake-up event
- Slave address in an I^2C frame sent by the voltage controller not acknowledged by the power IC device
- Register address in an I^2C frame sent by the voltage controller not acknowledged by the power IC device
- Last byte of an I^2C frame issued from the bypass port or the voltage manager FSM ports sent by the voltage controller not acknowledged by the power IC device

The end-of-on time period and end-of-off time period events of the event generator module are sources of interrupts to the MPU processor (the corresponding bits in the PRM_IRQENABLE_MPU are set to 1). The end-of-off time period is the source of wake-up events on the MPU domain.

The MPU can force a sleep or wake-up transition on some domains (SGX, USBHOST, IVA2, PER, DSS, CAM, NEON, EMU). The PRM triggers the MPU interrupt when the domain enters a power state. The software must clear the CM_CLKSTCTRL_<domain_name> register only after getting the interrupt. If the software clears this bit before getting the interrupt, the interrupt never occurs, regardless of whether the transition occurs.

An interrupt for a peripheral with wake-up capability is enabled when the wake-up enable bit (PM_WKEN_<domain_name> register type) and group select bit (PM_<processor_name>GRPSEL_<domain_name> type of register) are set to 1.

The PRM triggers its interrupt line on the I/O daisy chain wake-up event. This wake-up event is enabled by setting the corresponding bit in the PRCM.PM_WKEN_WKUP register to 1.

The PRM triggers the MPU interrupt as long as the DPLL recalibration flag is set and the corresponding interrupt enable bit in the PRM_IRQENABLE_<processor_name> register is set to 1. The recalibration flag is set by the DPLL and remains active if the DPLL is not reinitialized.

4.12.1.3.2 MPU Interrupt Registers

4.12.1.3.2.1 PRM_IRQENABLE_MPU (MPU Interrupt Enable Register)

The MPU interrupt enable register allows independent masking/unmasking of each MPU internal interrupt source.

Note: If the following interrupts are enabled and the MPU power domain is idled, then when the event occurs, the PRCM module sets the interrupt that wakes up the power domain:

- DPLL1/DPLL2/DPLL3/DPLL4/DPLL5 recalibration event
- Voltage controller errors

4.12.1.3.2.2 PRM_IRQSTATUS_MPU (MPU Interrupt Status Register)

The MPU interrupt status register provides the status of all PRCM internal events that can generate an MPU interrupt. Software must read this register to identify the interrupt cause, and then clear the pending interrupt by setting the corresponding bit to 1.

4.12.1.3.3 IVA2.2 Interrupt Event Sources

The IVA2.2 interrupt registers correspond to the interrupt sources connected to the interrupt line mapped to the IVA2.2 interrupt controller.

Three events can activate this interrupt line:

- An IVA2.2 peripheral group wake-up event
- A force wake-up transition completion (IVA2 domain)
- A required IVA2 DPLL recalibration

The PRM also interrupts IVA2.2 if the corresponding bit in the PM_IVA2GRPSEL_WKUP register is set to 1.

The PRM triggers the interrupt line dedicated to the IVA2 processor when the MPU performs a force wake-up transition on the IVA2 domain. This interrupt is triggered under the same conditions as that of the MPU.

The PRM triggers an IVA2.2 interrupt if the DPLL recalibration flag is set and the corresponding interrupt enable bit in the PRM_IRQENABLE_<processor_name> register is set to 1. The recalibration flag is set by the DPLL and remains active if the DPLL is not reinitialized.

4.12.1.3.4 IVA2 Interrupt Registers

4.12.1.3.4.1 PRM_IRQENABLE_IVA2 (IVA2.2 Interrupt Enable Register)

The IVA2.2 interrupt enable register allows independent masking/unmasking of each of the three internal interrupt sources.

Note: If the IVA2 DPLL recalibration event interrupt is enabled and the IVA2 power domain is idled, then when the event occurs, the PRCM module sets the interrupt, thus waking up the power domain.

4.12.1.3.4.2 PRM_IRQSTATUS_IVA2 (IVA2.2 Interrupt Status Register)

The IVA2.2 interrupt status register provides the status of the three events that can generate an IVA2.2 interrupt. Software must read the register to identify the interrupt cause, and then clear the pending interrupt by setting the corresponding bit to 1.

4.12.1.4 Event Generator Control Registers

For details about the event generator module, see the public *ARM Cortex-A8* technical reference manual.

In the PRCM module, three event generator registers allow configuration of the event generator module:

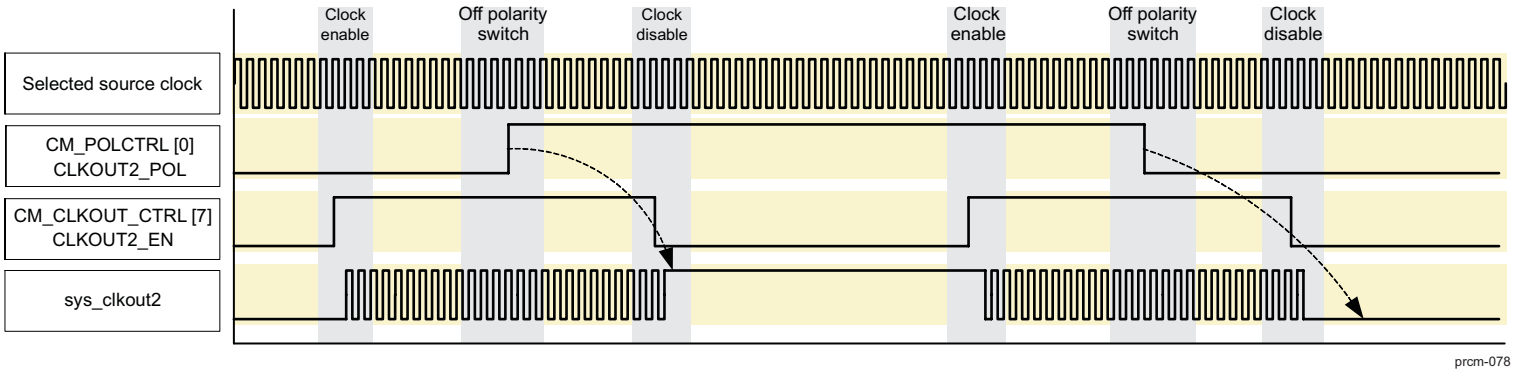
- PM_EVGENCTRL_MPU (event generator control register): Event-generator settings
- PM_EVGENONTIM_MPU (event generator on-time register): On-time duration setting
- PM_EVGENOFFTIM_MPU (event generator off-time register): Off-time duration setting

4.12.1.5 Output Signal Polarity Control Registers

4.12.1.5.1 CM_POLCTRL (CM Polarity Control Register)

The CM_POLCTRL register allows the setting of the polarity of sys_clkout2 when gated (disabled). sys_clkout2 can be gated to a low level or a high level, depending on the software programming of this register. Figure 4-78 shows the normal behavior of sys_clkout2 when gated.

Figure 4-78. sys_clkout2 Gating Polarity Control



4.12.1.5.2 PRM_POLCTRL (PRM Polarity Control Register)

The PRM polarity control register allows the setting of the polarity of the external signals controlled by the PRM. It contains the following polarity control bits:

- OFFMODE_POL: Polarity of sys_off_mode
- CLKOUT_POL: Polarity of sys_clkout1
- CLKREQ_POL: Polarity of sys_clkreq
- EXTVOL_POL: Polarity of both sys_nvmode signals: sys_nvmode1 and sys_nvmode2

At device power up, the reset values of polarity settings are OFFMODE_POL = 1 and CLKREQ_POL = 1. For details about using these signals, see Section 4.7, *Clock Manager Functional Description*.

4.12.1.6 SRAM Precharge Time Control Register

4.12.1.6.1 PRM_SRAM_PCHARGE (Voltage SRAM Precharge Counter Register)

The voltage SRAM precharge counter register allows the setting of the precharge duration of the SRAM. It has following bit field:

- PCHARGE_TIME: Number of system clock cycles defined for the SRAM precharge duration

4.12.2 Clock Management Registers

4.12.2.1 System Clock Control Registers

4.12.2.1.1 PRM_CLKSRC_CTRL (Clock Source Control Register)

The clock source control register is dedicated to the clock source controls of the device. It contains the following bit fields:

- SYSCLKSEL: Indicates the oscillator mode (oscillator/bypass) and is automatically updated at power up
- AUTOEXTCLKMODE: Enables autogating for the system clock, depending on the device power state. The clock can be configured to be automatically gated when all power domains are in either off or retention state. When the gating condition is satisfied, the internal oscillator is turned off, if it is used. Otherwise, sys_clkreq is asserted to notify the external clock source to turn off.

Note: Power consumption is reduced with the SYS_CLK off; however, wake-up latency is increased because of oscillator stabilization time after the clock is turned on again.

- SYSCLKDIV: Input divider (1, 2) of the system clock

4.12.2.1.2 PRM_CLKSETUP (Source-Clock Setup Register)

The source-clock setup register allows the setting of the setup time of the oscillator system clock, based on the number of 32-kHz clock cycles. This duration corresponds to the time required by the oscillator to stabilize before propagating the system clock.

Because the reset lasts long enough for oscillator stabilization, this register is not used at power-on reset of the device. This is ensured either by the external reset source (power IC) or by the reset extension in the PRCM module (RSTTIME0 timer). The PRM_CLKSETUP register is cleared on cold reset only.

Note: The clock setup time is implemented using the oscillator mode or the external clock mode (bypass mode). (The mode of the oscillator is selected by the SYSCLKSEL bits in the PRM_CLKSRC_CTRL register).

4.12.2.1.3 PRM_CLKSEL (Source-Clock Selection Register)

The PRCM.PRM_CKSEL[2:0] SYS_CLKIN_SEL bit field is used to select the input frequency of the system clock (12, 13, 16.8, 19.2, 26 or 38.4 MHz).

4.12.2.2 External Clock Output Control Registers

4.12.2.2.1 PRM_CLKOUT_CTRL (Clock Out Control Register)

The PRM clock out control register provides control to enable and disable the gating of the sys_clkout1 output clock.

4.12.2.2.2 CM_CLKOUT_CTRL (Clock Out Control Register)

The CM clock out control register provides control over the device output clock sys_clkout2, which can be used externally for functional or test purposes. The register allows the following:

- Selection of the source clock for sys_clkout2:
  - CORE_CLK
  - CM_SYS_CLK
  - 96-MHz clock
  - 54-MHz clock
- Dividing-down the selected source clock by 1, 2, 4, 8, or 16
- Enabling/disabling of the gating of sys_clkout2

4.12.2.3 DPLL Clock Control Registers

A set of registers controls the clock features of the five DPLLs (DPLL1, DPLL2, DPLL3, DPLL4, and DPLL5):

- CM_CLKSELn_PLL_<processor_name>
- CM_CLKSELn_PLL
- CM_CLKEN_PLL_<processor_name>
- CM_CLKEN_PLL
- CM_AUTOIDLE_PLL_<processor_name>
- CM_AUTOIDLE_PLL
- CM_IDLEST_PLL_<processor_name>
- CM_IDLEST_CKGEN
- CM_IDLEST2_CKGEN

The following sections describe the purposes of these registers.

4.12.2.3.1 CM_CLKSELn_PLL_ <processor_name> (Processor DPLL Clock Selection Register)

The processor DPLL clock selection register controls the clock configuration of DPLL1 and DPLL2, including the following features:

- The multiplier (M) and divider (N) values of the DPLL
- Selection of the fast bypass clock (CORE_CLK or CORE_CLK/2) in bypass mode
- Configuration of DPLL output clock divider values (M2 factor)

The device has four DPLL clock selection registers for DPLL1 and DPLL2:

- CM_CLKSEL1_PLL_MPU: DPLL1 multiplier, divider, and fast bypass clock selection
- CM_CLKSEL2_PLL_MPU: DPLL1 output clock divider selection
- CM_CLKSEL1_PLL_IVA2: DPLL2 multiplier, divider, and fast bypass clock selection
- CM_CLKSEL2_PLL_IVA2: DPLL2 output clock divider selection

4.12.2.3.2 CM_CLKSELn_PLL (DPLL Clock Selection Register)

The DPLL clock selection register controls the clock configuration for DPLL3, DPLL4, and DPLL5, including the following features:

- The multiplier (M) and divider (N) values of the DPLL
- Configuration of DPLL output clock divider values

The device has the following DPLL clock selection registers for DPLL3, DPLL4, and DPLL5:

- CM_CLKSEL1_PLL: DPLL3 multiplier, divider, and output clock division configuration. Source selection for the 54-MHz and 48-MHz clock (48M_FCLK) between DPLL4 output and sys_altclk.
- CM_CLKSEL2_PLL: DPLL4 multiplier and divider configuration
- CM_CLKSEL3_PLL: Divider configuration for 96-MHz clock (96M_FCLK)
- CM_CLKSEL4_PLL: DPLL5 multiplier and divider configuration
- CM_CLKSEL5_PLL: Divider configuration for 120-MHz clock (120M_FCLK)

4.12.2.3.3 CM_CLKEN_PLL_ <processor_name> (Processor DPLL Clock Enable Register)

The processor DPLL clock enable register allows the enabling or disabling of DPLL1 and DPLL2. It allows an immediate setting of the DPLL.

The device has two DPLL clock enable registers, one for DPLL1 and one for DPLL2:

- CM_CLKEN_PLL_MPU: DPLL1 (MPU)
- CM_CLKEN_PLL_IVA2: DPLL2 (IVA2)

The CM_CLKEN_PLL_<processor_name> register allows programmable control of the following DPLL features:

- Low-power stop mode (only for DPLL2)/low-power bypass mode/lock mode selection
- Automatic recalibration enable/disable
- Programmable internal frequency range of the DPLL
- The DPLL LP mode can be enabled or disabled. However, switching between LP and normal mode is effective only when the DPLL has performed a recalibration; therefore, the DPLL must lock or relock. Also, the LP mode control is considered only during the following transitions:
  - From bypass to lock
  - From stop mode to lock
  - From lock to relock

Note: DPLL1 enters its internal power state (MNBYPASS) after being released from reset or when a multiplier value of 0 or 1 is loaded into the DPLL. Therefore, even if the CM_CLKEN_PLL_MPU[2:0] EN_MPU_DPLL bit field is reset to the low-power bypass mode, DPLL1 automatically transitions to MNBYPASS mode, because the multiplier value (the CM_CLKSEL1_PLL_MPU[18:8] MPU_DPLL_MULT bit field) resets to 0.

4.12.2.3.4 CM_CLKEN_PLL (DPLL Enable Register)

The DPLL enable register allows control of DPLL3 DPLL4, and DPLL5. It allows an immediate setting of the DPLLs. This register controls the following features of the two DPLLs:

- DPLL operation mode:
  - Low-power bypass, fast-relock bypass, and lock modes for DPLL3
  - Low-power bypass and lock modes for DPLL4
- Programmable automatic recalibration
- Programmable internal frequency range for the DPLL
- Frequency ramp-up feature disable/enable with programmable step duration
- The DPLL3 output M3X2 and the DPLL4 outputs M2X2, M3X2, M4X2, M5X2, and M6X2 clock paths can be powered down. However, the setting takes effect only when the output clock is gated. It is also powered down whenever the DPLL is in stop mode, regardless of the software settings.

- The DPLL LP mode can be enabled or disabled. However, switching between LP and normal mode is effective only when the DPLL has performed a recalibration; therefore, the DPLL must lock or relock. Also, the LP mode control is considered only during the following transition:
  - From bypass to lock
  - From stop mode to lock
  - From lock to relock

4.12.2.3.5 CM_AUTOIDLE_PLL_ <processor_name> (Processor DPLL Autoidle Register)

The processor DPLL autoidle register allows the enabling/disabling of the automatic processor DPLL activity control. In automatic mode, the DPLL automatically enters low-power stop mode when the DPLL clock is not required. It is also restarted automatically.

The following are the device DPLL autocontrol registers:

- CM_AUTOIDLE_PLL_MPU: DPLL1 autoidle mode control
- CM_AUTOIDLE_PLL_IVA2: DPLL2 autoidle mode control

4.12.2.3.6 CM_AUTOIDLE_PLL (DPLL Autoidle Register)

The DPLL autoidle register allows the enabling/disabling of the automatic mode-switching control for DPLL3 and DPLL4. This automatic mode takes effect only when the DPLLs are locked.

DPLL3 can be configured to automatically switch to either low-power bypass mode or low-power stop mode when the CORE power domain clock is not required.

DPLL4 can be configured to automatically switch to low-power stop mode when the PER power domain clock is not required.

4.12.2.3.7 CM_AUTOIDLE1_PLL (DPLL5 Autoidle Register)

The DPLL5 autoidle register allows the enabling/disabling of the automatic mode-switching control. This automatic mode takes effect only when the DPLL is locked.

When enabled, DPLL5 is automatically switched to low-power stop mode whenever the 120-MHz output clock is gated.

4.12.2.3.8 CM_IDLEST_CKGEN (Source-Clock Idle-Status Register)

The source-clock idle-status register provides a status of DPLL3 and DPLL4 clock activity. It also provides the status of the functional clocks derived from DPLL4 output.

The activity status for DPLL3 and DPLL4 can be:

- DPLL is bypassed.
- DPLL is locked.

The activity status for the functional clocks can be:

- The functional 96-MHz clock is active, or not.
- The functional 48-MHz clock is active, or not.
- The functional 12-MHz clock is active, or not.
- The functional 54-MHz clock is active, or not.

The functional 96-MHz clock and all other functional clocks derived from it are active only when DPLL4 is locked and the clock is required.

The functional 48-MHz and 12-MHz clocks are qualified as active only if the clock is required and when the external alternate clock is selected as the source clock.

The functional 54-MHz clock (DSS_TV_CLK) is active only when DPLL4 is locked, selected as the source clock, and required.

4.12.2.3.9 CM_IDLEST2_CKGEN (DPLL5 Source-Clock Idle-Status Register)

The source-clock idle-status register provides the status of DPLL5 clock activity. It also provides the status of the functional clocks derived from DPLL5 output.

The activity status for DPLL5 can be:

- DPLL is bypassed.
- DPLL is locked.

The activity status for the functional clocks can be:

- The 120-MHz functional clock is active, or not.
- The output stage of the 120-MHz functional clock is active, or not.

4.12.2.3.10 CM_IDLEST_PLL_ <processor_name> (Processor DPLL Idle-Status Register)

The processor DPLL idle-status register indicates the status of the processor DPLL: whether it is in locked or bypass mode.

The device DPLL idle-status registers are:

- CM_IDLEST_PLL_MPU: DPLL1 activity status
- CM_IDLEST_PLL_IVA2: DPLL2 activity status

4.12.2.4 Power-Domain Clock Control Registers

An identical set of registers controls the clock features of the power domains in the device:

- CM_CLKSEL_<domain_name>
- CM_FCLKEN_<domain_name>
- CM_ICLKEN_<domain_name>
- CM_AUTOIDLE_<domain_name>
- CM_IDLEST_<domain_name>
- CM_CLKSTCTRL_<domain_name>
- CM_CLKSTST_<domain_name>
- CM_SLEEPDEP_<domain_name>

The following sections describe the purposes of these registers.

4.12.2.4.1 CM_CLKSEL_ <domain_name> (Clock Select Register)

The clock select register controls the selection of the module or subsystem input clock frequency (except for the processor modules). Therefore, it deals only with modules or subsystems for which the frequency is scalable or selectable (the others have fixed and nonprogrammable frequencies). Both functional and interface clocks can be scaled. In most cases, their value is a divided value from the DPLL3 (CORE) or the DPLL4 (PER) output clock.

The device includes the following clock select registers:

- CM_CLKSEL_CORE: L3 and L4 interconnects, and GPTIMER10 and 11 functional clocks
- CM_CLKSEL_CAM: Camera subsystem functional clock
- CM_CLKSEL_DSS: DSS functional clock 1 and TV functional clock
- CM_CLKSEL_PER: GPTIMER2, 3, 4, 5, 6, 7, 8, and 9 functional clocks
- CM_CLKSEL_SGX: SGX subsystem functional clock
- CM_CLKSEL_WKUP: GPTIMER1 functional clock and reset manager counter clock

The 32-kHz functional clock (32K_FCLK) is selected when the device enters off mode. The functional clock of the GPTIMER1 is selectable between the always-on 32K_FCLK and the system clock (SYS_CLK), but it is used with the 32-kHz clock.

The CM_CLKSEL_SGX register controls the SGX functional clock divider ratio, which is divided from the L3_ICLK source clock. Table 4-81 summarizes L3_ICLK ratio for different configurations of the register field.

Table 4-81. GFX Functional Clock Ratio Settings

CM_CLKSEL_SGX.CLKSEL_SGX	SGX_L3_FCLK
0x1	L3_ICLK
0x2	L3_ICLK/2
0x3	L3_ICLK/3
0x4	L3_ICLK/4

Note: SGX clock control is only available on OMAP3430.

4.12.2.4.2 CM_FCLKEN_ <domain_name> (Functional Clock Enable Register)

The functional clock enable register allows control of the functional clock activity of each module or subsystem. All module functional clocks are controllable by software, except for the MPU, interconnect, and memory subsystems, for which the clocks are automatically controlled by the PRCM.

The device has the following functional clock control registers:

- CM_FCLKEN1_CORE and CM_FCLKEN3_CORE: CORE domain peripherals set
- CM_FCLKEN_CAM: Camera subsystem
- CM_FCLKEN_DSS: DSS subsystem
- CM_FCLKEN_PER: PER domain peripherals set
- CM_FCLKEN_IVA2: IVA2.2 subsystem
- CM_FCLKEN_SGX: SGX subsystem
- CM_FCLKEN_WKUP: WKUP domain peripherals set
- CM_FCLKEN_USBHOST: HS USB Host subsystem

The software effect is immediate and direct. The functional clock is turned on as soon as the bit is set, and turned off if the bit is cleared and the clock is not required by any module. On module wakeup, the functional clock can be automatically restarted.

Note: The functional clock supplies the functional part of a module or subsystem, which is not operational without its functional clock. In some cases, a module or a subsystem may require multiple functional clocks.

4.12.2.4.3 CM_ICLKEN_ <domain_name> (Interface Clock Enable Register)

The interface clock enable register allows control of the interface clock activity of each module or subsystem. This register provides control over each module in the device.

The device has following interface clock control registers:

- CM_ICLKEN1_CORE and CM_ICLKEN3_CORE: CORE domain peripherals set
- CM_ICLKEN_CAM: Camera subsystem
- CM_ICLKEN_DSS: DSS subsystem
- CM_ICLKEN_PER: PER domain peripherals set
- CM_ICLKEN_SGX: SGX subsystem
- CM_ICLKEN_WKUP: WKUP domain peripherals set
- CM_ICLKEN_USBHOST: HS USB Host subsystem

This register has an immediate effect, causing the source of the interface clock to be effectively cut (if the appropriate bit is cleared and no module requires this clock) or activated (if the appropriate bit is set).

Independent functional and interface clock control registers provide potential power savings for each module. For example, because the configuration port and interface of the module are still active, disabling a functional clock of a module while keeping its interface clock active (leaving the module inactive but allowing access to its registers) reduces power consumption.

When the interface clock is disabled, the module cannot communicate with the reset of the device; therefore, it is in idle mode. For example, the interface clock of a peripheral can be disabled while its functional clock is active; it is then idled, from the device standpoint, while it can detect any external event. This configuration typically allows a main part of the device to go into idle mode while keeping a peripheral active and ready to wake up from an external event.

Because a module may or may not be able to function without its functional or interface clocks, power-management strategies must be adapted accordingly. This relation is programmable and is defined in the CLOCKACTIVITY bits of the module SYSCONFIG register; therefore, it requires consistent programming of the CM_FCLKEN and CM_ICLKEN registers.

Note: The interface clock ensures proper communication between any module and the interconnect (L3 or L4), in most cases supplying the module interface and registers.

4.12.2.4.4 CM_AUTOIDLE_ <domain_name> (Autoidle Register)

The autoidle register holds an AUTOIDLE bit per module that belongs to the related power domain. Each AUTOIDLE bit enables/disables automatic (hardware) gating of a module interface clock.

When AUTOIDLE and ICLKEN are set for a module, the module interface clock is managed automatically (that is, by hardware control) according to the power domain clock activity; for example, stopped before a power domain sleep transition and reenabled on wakeup. Table 4-82 lists the possible autoidle settings for the interface clock.

Table 4-82. Interface Clock Autoidle Settings

CM_AUTOIDLE.AUTO_<module>	CM_ICLKEN.EN_<module>	Interface Clock
0	0	Disabled
0	1	Enabled
1	0	Disabled
1	1	Automatic enabling/disabling

The device has the following autoidle control registers:

- CM_AUTOIDLE1_CORE and CM_AUTOIDLE3_CORE: CORE domain peripherals set
- CM_AUTOIDLE_CAM: Camera subsystem
- CM_AUTOIDLE_DSS: DSS subsystem
- CM_AUTOIDLE_PER: PER domain peripherals set
- CM_AUTOIDLE_WKUP: WKUP domain peripherals set
- CM_AUTOIDLE_USBHOST: HS USB Host subsystem

Note: For IVA2.2, and GFX modules, the automatic idle mode is always enabled, and is not software-controllable.

4.12.2.4.5 CM_IDLEST_ <domain_name> (Idle-Status Register)

The idle-status register allows checking whether a target module is in idle mode or if an initiator module is in standby mode. The software should not access a target module in idle mode. A target access in this state can lead to an error.

The idle mode of any module can depend on the configuration of the CM_FCLKEN_<domain_name> and CM_ICLKEN_<domain_name> registers, or may be controlled automatically by hardware, depending on the configuration of the CM_AUTOIDLE_<domain_name> registers.

In the case of IVA2.2 subsystem, standby mode is reached after the IVA2.2 processor has performed its idle instruction.

The device has the following idle status registers:

- CM_IDLEST_MPU: MPU subsystem
- CM_IDLEST1_CORE and CM_IDLEST3_CORE: CORE domain peripherals set
- CM_IDLEST_CAM: Camera subsystem
- CM_IDLEST_DSS: DSS
- CM_IDLEST_PER: Peripheral domain peripherals set
- CM_IDLEST_NEON: NEON subsystem
- CM_IDLEST_IVA2: IVA2.2 subsystem
- CM_IDLEST_SGX SGX subsystem
- CM_IDLEST_WKUP: WKUP domain peripherals set
- CM_IDLEST_USBHOST: HS USB Host subsystem

4.12.2.4.6 CM_CLKSTCTRL_ <domain_name> (Clock State Control Register)

The clock state control register holds a CLKTRCTRL_<clock domin> bit field for each clock domain in the power domain. It controls the hardware- and software-supervised state transitions between active and inactive states. Table 4-83 lists the clock state transition settings.

Table 4-83. Clock State Transition Settings

CM_CLKSTCTRL_<pwr domain>. CLKTRCTRL_<clk domain>	Description
0x0	The automatic hardware-supervised mode is disabled. The clocks in a clock domain cannot be cut automatically. This prevents any power transition on the power domain.
0x1	Starts software-supervised (forced) sleep transition on the domain. All clocks in the clock domain are automatically cut whenever the initiators in the modules are in standby mode.
0x2	Starts software-supervised (forced) wake-up transition on the domain. The clocks in the clock domain are restarted.
0x3	The automatic hardware-supervised mode is enabled, and the clocks in the clock domain are automatically cut whenever the modules and subsystem in the clock domain are in idle or standby mode and the domain dependencies are met.

The hardware-supervised mode and the software-supervised (forced) sleep mode are mutually exclusive. It is a software decision to program one mode or another, and the software programs the PRCM module accordingly.

The hardware-supervised mode is coupled to the sleep and wake-up dependencies programmed in the PRCM module, whereas the software-supervised mode must be independent of those dependencies. Therefore, the sleep and wake-up dependencies must be disabled before the software forces a sleep transition on a power domain; otherwise, the forced-domain will be wakened immediately because of the wake-up dependency.

The device has the following clock state control registers:

- CM_CLKSTCTRL_MPU: MPU subsystem clock domain
- CM_CLKSTCTRL_CORE: L3, L4, and D2D clock domains
- CM_CLKSTCTRL_CAM: Camera subsystem clock domain
- CM_CLKSTCTRL_DSS: DSS clock domain
- CM_CLKSTCTRL_PER: Peripherals clock domain
- CM_CLKSTCTRL_NEON: NEON clock domain
- CM_CLKSTCTRL_IVA2: IVA2.2 clock domain
- CM_CLKSTCTRL_SGX: Graphics clock domain
- CM_CLKSTCTRL_EMU: Emulation clock domain
- CM_CLKSTCTRL_USBHOST: HS USB Host clock domain

The CORE domain has three clock domains (L3, L4, and D2D); it does not have forced sleep and forced wake-up ability.

Although the sleep transition in the MPU power domain cannot be initiated by software using this register, a software-initiated forced wake-up capability exists. This can be used to wake up the MPU power domain if it does not wake up when the CORE power domain wakes up (the MPU wake-up-dependency PM_WKDEP_MPU[0] EN_CORE bit is set to 0).

If the hardware-supervised mode is enabled, the following occur:

- The MPU domain clock is automatically cut if the MPU executes the wait-for-interrupt instruction.
- The IVA2 domain clock is automatically cut when the DSP completes its idle procedure, and the IVA2.2 subsystem is ready for standby.
- The L3 domain clock is automatically cut when all initiators are in standby mode and slave ports are idled.
- The L4 domain clock is automatically cut when all peripherals and slave ports are idled.
- The DSS interface clock is automatically cut when it stops fetching data from the frame buffer (provided the MPU is also in standby mode, if sleep dependency in the MPU power domain is enabled). The display functional clock is controlled only by the CM_FCLKEN_DSS register.
- The PER domain clock is automatically cut when all initiators are in standby mode and all slave ports are in idle (provided the MPU and the DSP are in standby mode and the CORE power domain is inactive, if sleep dependency in their domains is enabled).
- The USBHOST power domain clock is automatically cut whenever the USBHOST is in standby mode (provided MPU is also in standby mode, if sleep dependency in the MPU domain is enabled, and IVA2 is also in standby mode, if sleep dependency in the IVA2 domain is enabled).
- Because of the hardwired sleep dependency between NEON and the MPU domain, NEON can go into idle only if the MPU goes into standby mode. The MPU domain must also be configured in automatic hardware supervised mode for the NEON power domain idle transition to occur.

4.12.2.4.7 CM_CLKSTST_ <domain_name> (Clock State Status Register)

The clock state status register logs the activity status of the power domain clock. This includes the activity of the interface clocks running only on the domain.

The device has following clock state status registers:

- CM_CLKSTST_MPU: MPU subsystem clock activity
- CM_CLKSTST_CORE: L3 clock domain activity and L4 clock domain activity
- CM_CLKSTST_CAM: Camera subsystem clock activity
- CM_CLKSTST_DSS: DSS clock activity
- CM_CLKSTST_PER: PER clock domain activity
- CM_CLKSTST_IVA2: IVA2.2 clock domain activity
- CM_CLKSTST_SGX: Graphics subsystem clock activity
- CM_CLKSTST_EMU: Emulation clock activity
- CM_CLKSTST_USBHOST: USBHOST clock activity

4.12.2.4.8 CM_SLEEPDEP_ <domain_name> (Sleep Dependency Control Register)

The sleep dependency control register allows the enabling or disabling of the sleep transition dependency of a power domain with respect to other power domains.

The device has following sleep dependency registers:

- CM_SLEEPDEP_CAM: CAM power domain sleep dependency with the MPU power domain
- CM_SLEEPDEP_DSS: Display power domain sleep dependency with the MPU power domain and the IVA2 power domain
- CM_SLEEPDEP_PER: PER power domain sleep dependencies with the MPU, IVA2, and CORE power domains
- CM_SLEEPDEP_SGX: SGX power domain sleep dependency with the MPU power domain

- CM_SLEEPDEP_USBHOST: USBHOST power domain sleep dependency with the MPU and IVA2 power domains

Although the CORE power domain is composed of L3 and L4 clock domains, the sleep dependency with the L3 clock domain is always enabled (not programmable by software); there is no sleep dependency between the PER power domain and the CORE_L4 clock domain.

The dependencies listed in Table 4-84 can be enabled or disabled by software.

Table 4-84. Sleep Dependency Settings

		Parent Domain		
Dependent Domain		**MPU**	**IVA2**	**CORE-L3**
CAM		Software controlled		
DSS		Software controlled	Software controlled	
PER		Software controlled	Software controlled	Always enabled
SGX		Software controlled		
USBHOST		Software controlled	Software controlled	

4.12.2.5 Domain Wake-Up Control Registers

The following modules have programmable wake-up control:

- CORE power domain:
  - USBTLL
  - HS USB OTG
  - McBSP 1 and 5
  - GPTIMER[10, 11]
  - UART[1,2]
  - I2C[1..3]
  - McSPI[1..4]
  - MMC[1,2, 3]
- PER power domain:
  - McBSP[2..4]
  - GPTIMER[2..9]
  - UART 3
  - WDTIMER3
  - GPIO[2..6]
- WKUP power domain:
  - GPTIMER1
  - GPIO1
  - I/O pad
- DSS power domain:
  - DSS subsystem
- USBHOST power domain:
  - HS USB Host subsystem

The registers in the following sections control the wake-up settings.

4.12.2.5.1 PM_WKEN_ <domain_name> (Wake-Up Enable Register)

The wake-up enable register applies only to modules that can generate wake-up events. It allows the enabling or disabling of the wakeup of the related power domain on a module or subsystem wake-up event. Each EN_<module> bit in the register enables/disables the wake-up event from the module to the PRCM module.

The device has the following wake-up enable registers:

- PM_WKEN1_CORE and PM_WKEN3_CORE: CORE domain modules wake-up control
- PM_WKEN_DSS: Display subsystem (DSS) wake-up control
- PM_WKEN_PER: Peripheral domain modules wake-up control
- PM_WKEN_WKUP: WKUP domain modules wake-up control
- PM_WKEN_USBHOST: HS USB Host subsystem wake-up control

Note: The wake-up signals issued from the MPU and IVA2.2 INTCs are nonmaskable in the PRCM module. However, they cannot be generated if the corresponding interrupts are masked in the INTCs.

4.12.2.5.2 PM_WKST_ <domain_name> (Wake-Up Status Register)

The wake-up status register logs wake-up events from all modules. Each ST_<module> bit of this register logs a given module-generated wake-up event.

The device has the following wake-up status registers:

- PM_WKST1_CORE and PM_WKST3_CORE: CORE domain modules wake-up event status
- PM_WKST_PER: Peripheral domain modules wake-up event status
- PM_WKST_WKUP: WKUP domain modules wake-up event status
- PM_WKST_USBHOST: HS USB Host subsystem wake-up control

The functional clock of the module causing the wakeup automatically restarts on the wakeup, but the software must enable the functional clock (CM_FCLKEN_<domain_name> register) before clearing the wake-up status bit.

Notes:

- Software must clear this register before a new sleep transition request; otherwise, the register prevents the sleep transition from occurring.
- If the PRM interrupt is enabled on a module wakeup, the PM_WKST_<domain_name> must be cleared before clearing the PRM_IRQSTATUS_<processor_name> interrupt status register.

4.12.2.5.3 PM_WKDEP_ <domain_name> (Wake–Up Dependency Register)

The wake-up dependency register allows the enabling/disabling of a wake-up dependency between power domains. When a domain wakes up, it can wake up another domain.

The device has the following wake-up dependency registers:

- PM_WKDEP_MPU: MPU power domain wake-up dependency with the following domains:
  - CORE
  - IVA2
  - WKUP
  - DSS
  - PER
- PM_WKDEP _DSS: DSS power domain wake-up dependency with the following domains:
  - MPU
  - IVA2
  - WKUP
- PM_WKDEP _CAM: The CAM power domain has programmable wake-up dependency with the following domains:
  - MPU
  - IVA2

  - WKUP
- PM_WKDEP _PER: The PER power domain has programmable wake-up dependency with the following domains:
  - MPU
  - IVA2
  - WKUP
  - CORE

  The PER power domain can be wakened only by a wakeup of the L3 and D2D clock domains, not by an L4 clock domain.
- PM_WKDEP _NEON: The NEON power domain has programmable wake-up dependency with the following domain:
  - MPU
- PM_WKDEP_IVA2: The IVA2 power domain has programmable wake-up dependency with the following domains:
  - MPU
  - PER
  - WKUP
  - CORE
- PM_WKDEP _SGX: The SGX power domain has programmable wake-up dependency with the following domains:
  - MPU
  - IVA2
  - WKUP
- PM_WKDEP _USBHOST: The USBHOST power domain has programmable wake-up dependency with the following domains:
  - MPU
  - IVA2
  - WKUP
  - CORE

There is no wake-up dependency control for the WKUP power domain, because it is always on.

Because the CORE power domain wake-up dependencies are not programmable, the CORE power domain always wakes up on a wake-up event on the following domains:

- MPU
- IVA2
- WKUP
- CAM
- DSS
- SGX
- PER
- USBHOST

4.12.2.5.4 PM_ <processor_name> GRPSEL_ <domain_name> (Processor Group Selection Register)

The processor group selection register allows defining the group of modules in the domain that can wake up a processor domain (MPU or IVA2). This bit is effective only if the module wake-up capability is enabled (PM_WKEN_<domain_name> register).

The device has the following processor group selection registers:

- PM_MPUGRPSEL1_CORE and PM_MPUGRPSEL3_CORE: CORE power domain modules MPU wake-up control

- PM_IVA2GRPSEL1_CORE and PM_IVA2GRPSEL3_CORE: CORE power domain modules IVA2 wake-up control
- PM_MPUGRPSEL_PER: PER power domain modules MPU wake-up control
- PM_IVA2GRPSEL_PER: PER power domain modules IVA2 wake-up control
- PM_MPUGRPSEL_WKUP: WKUP power domain modules MPU wake-up control
- PM_IVA2GRPSEL_WKUP: WKUP power domain modules IVA2 wake-up control
- PM_MPUGRPSEL_USBHOST: HS USB Host subsystem MPU wake-up control
- PM_IVA2GRPSEL_USBHOST: HS USB Host subsystem IVA2 wake-up control

Note: The wake-up capability of the DSS is always attached to the MPU group; it cannot be attached to the IVA2.2, and therefore is not programmable.

4.12.3 Reset Management Registers

4.12.3.1 Reset Control

4.12.3.1.1 PRM_RSTTIME (Reset Time Register)

The reset time register provides control over reset duration. Two durations are configurable:

- RSTTIME1: Minimum duration (by default, two cycles of the 32-kHz clock) of the global warm reset (sys_nreswarm) assertion for external devices, such as flash memories. At power-up reset, DPLLs are reset and the power domains are switched on and stabilized during this duration.
- RSTTIME2: Minimum duration (by default, 16 cycles of the RM_ICLK clock) to control power domain resets when the domain clocks are on.

For global cold and warm resets, the reset is applied for the RSTTIME1 + RSTTIME2 duration. In the other cases, when a power domain individually goes from off to active, the reset is applied only for the RSTTIME2 duration (plus the power domain wake-up duration).

RSTTIME1 and RSTTIME2 can be reprogrammed for different behavior after the initial power up.

4.12.3.1.2 RM_RSTCTRL_ <domain_name> (Reset Control Register)

The reset control register provides control over the local domain software reset.

Most device modules include an individual software reset that can be controlled using the related module SYS_CONFIG register.

The RM_RSTCTRL_<domain_name> register is used for specific subsystems or modules that do not support this local reset or that require a specific system control (the IVA2.2 subsystem).

The PRM_RSTCTRL register also handles the global software warm reset control.

The device includes the following reset control registers:

- PRM_RSTCTRL: This register allows control of the assertion of the global software reset and the DPLL3 software reset. These bits are automatically cleared. Assertion of the DPLL3 software reset triggers a device global cold reset. The reset condition of this register depends on the bit field:
  - The RST_GS bit is set on any global source of reset (warm or cold).
  - The RST_DPLL3 bit is set only on a global cold source of reset.
- RM_RSTCTRL_IVA2: IVA2 power domain software resets the IVA2.2 and SEQ subsystems. Both subsystems are held under reset after power up and are released by software. The SEQ software reset bit is reset to 1 (SEQ reset active) on an IVA2 domain power transition from off or retention to on.

4.12.3.1.3 RM_RSTST_ <domain_name> (Reset Status Register)

The reset status register logs any source that has generated a reset in the related power domain. Depending on the domain, several causes of reset can be logged:

- Global device cold reset
- Global device warm reset
- Power domain transition (off to active, and inactive to active)
- Software reset
- Processor emulation reset

The PRM_RSTST register logs the source of the global reset:

- VDD1/VDD2 voltage manager reset
- External warm reset
- MPU watchdog reset
- Global software reset and DPLL3 software reset
- Global cold reset

The device includes the following reset status registers:

- RM_RSTST_MPU
- RM_RSTST_CORE
- RM_RSTST_DSS
- RM_RSTST_CAM
- RM_RSTST_IVA2
- RM_RSTST_PER
- RM_RSTST_NEON
- RM_RSTST_SGX
- RM_RSTST_EMU
- PRM_RSTST
- RM_RSTST_USBHOST

Only the RM_RSTST_CORE, RM_RSTST_IVA2, and PRM_RSTST registers log software sources of reset.

Each bit of these registers is set on the effective release of the respective reset signal.

4.12.4 Power Management Registers

4.12.4.1 PM_PWSTCTRL_ <domain_name> (Power State Control Register)

The power state control register allows controlling the power state transition of the domain.

This register holds the following bit fields:

- POWERSTATE: Power state (ON, retention, or off) to be applied for the next transition
- LOGICRETSTATE: Logic is retained or switched off when the domain goes into retention. This feature is domain-dependent and is not necessarily programmable (read only).
- MEMRETSTATE[1 to X]: Memory block i is retained (RAM state retained) or switched off (RAM state not retained) when the domain goes into retention. This feature is domain-dependent and is not necessarily programmable (read only).
- MEMONSTATE[1 to X]: Memory block i in the domain is on (RAM active), retained (RAM inactive but state retained) or switched off (RAM state not retained) when the domain is on. This feature is domain-dependent and is not necessarily programmable (read only).
- MEMORYCHANGE: Allows updating the memory state according to latest MEMONSSTATE[1 to X] setting.

The device has the following power state control registers:

- PM_PWSTCTRL_MPU: MPU domain power management
- PM_PWSTCTRL_CORE: CORE domain power management
- PM_PWSTCTRL_SGX: SGX domain power management
- PM_PWSTCTRL_DSS: DSS domain power management
- PM_PWSTCTRL_CAM: CAM domain power management
- PM_PWSTCTRL_PER: PER domain power management
- PM_PWSTCTRL_NEON: NEON domain power management
- PM_PWSTCTRL_IVA2: IVA2 domain power management
- PM_PWSTCTRL_USBHOST: USBHOST domain power management

MPU domain power management has the following features:

- The power state is programmable to on, retention, and off states.
- Logic and L1 cache state are switchable to off and retention states when the power domain is in retention state.
- L2 cache state is switchable to off and retention states when the power domain is in retention state.
- L2 cache state is switchable to off and on states when the power domain is in on state.
- L2 cache state can be switched dynamically when the domain state is on (no power transition is required).
- The L1 cache is always on when the power domain state is on.

CORE domain power management has the following features:

- The power state is programmable to on, retention, or off states.
- Nonretained logic is switchable to off and retention states when the power domain is in retention state.
- Memory bank 1 and 2 states are switchable to off and retention states when the power domain is in retention state.
- Memory bank 1 and 2 states are switchable to off, retention, and on states when the power domain is in on state.
- Memory bank 1 and 2 states can be switched dynamically when the domain state is on (no power transition is required).

SGX domain power management has the following features:

- The power state is programmable to on, retention, and off states.
- The logic is always in retention state when the power domain is in retention state.
- The memory is always in retention state when the power domain is in retention state.
- The memory is always in on state when the power domain is in on state.

DSS domain power management has the following features:

- The power state is programmable to on, retention, and off states.
- The logic is always in retention state when the power domain is in retention state.
- The memory is always in retention state when the power domain is in retention state.
- The memory is always in on state when the power domain is in on state.

CAM domain power management has the following features:

- The power state is programmable to on, retention, and off states.
- The logic is always in retention state when the power domain is in retention state.
- The memory is always in retention state when the power domain is in retention state.
- The memory is always in on state when the power domain is in on state.

PER domain power management has the following features:

- The power state is programmable to on, retention, and off states.
- The logic is always in retention state when the power domain is in retention state.

NEON domain power management has the following features:

- The power state is programmable to on, retention, and off states.
- The logic is always in retention state when the power domain is in retention state.

IVA2 domain power management has the following features:

- The power state is programmable to on, retention, and off states.
- Logic state is switchable to off and retention states when the power domain is in retention state.
- L1 cache state is switchable to off and retention states when the power domain is in retention state.
- L1 flat memory state is switchable to off and retention states when the power domain is in retention state.
- L2 cache state is switchable to off and retention states when the power domain is in retention state.
- L2 flat memory state is switchable to off and retention states when the power domain is in retention state.
- L1 cache state is always on when the power domain is in on state.
- L1 flat memory state is always on when the power domain is in on state.
- L2 cache state is switchable to off and on states when the power domain is in on state.
- L2 flat memory state is always on when the power domain is in on state.
- L2 cache memory state can be switched dynamically to off state and back to on state when the domain state is on (no domain power transition is required).

USBHOST domain power management has the following features:

- The power state is programmable to on, retention, and off states.
- The logic is always in retention state when the power domain is in retention state.
- The memory is always in retention state when the power domain is in retention state, and the memory is always on when the power domain is in on state.

The EMU power state is not programmable. The EMU domain is always powered down when the software or hardware conditions that allow a power transition are met. Thus, the EMU domain is automatically powered down after the power-up sequence if no emulation hardware is connected.

Note: The L1 cache memory bank can be configured to be either entirely or partially cache or flat memory. This configuration is done in the IVA2.2 subsystem. The part that is configured as cache memory can be retained; however, because tag and validity bit information are not retained in the case of L1 cache, this part cannot be retrieved after wakeup. It is important to configure the L1 cache memory state with a consistent value depending on the chosen configuration when the domain is in retention state:

- If the L1 cache memory bank is entirely configured as cache memory, set the PM_PWSTCTRL_IVA2[8] SHAREDL1CACHEFLATRETSTATE bit to the same value as the PM_PWSTCTRL_IVA2[2] LOGICRETSTATE bit.
- If the L1 cache memory bank is entirely or partially configured as flat memory, set the PM_PWSTCTRL_IVA2[8] SHAREDL1CACHEFLATRETSTATE bit to the same value as the PM_PWSTCTRL_IVA2[9] L1FLATMEMRETSTATE bit.

The same mechanism applies to the L2 cache memory bank.

Because the L2 cache may not be useful in a system when the DSP frequency and interconnect frequencies are in the same range, this memory bank can be switched off when the domain is on.

4.12.4.2 PM_PWSTST_ <domain_name> (Power State Status Register)

The power state status register provides the status of the power state transition of the domain.

This register holds the following bit fields:

- POWERSTATESTATUS: Indicates the current power state of the domain (on, retention, off)
- LOGICSTATESTATUS: Indicates the current logic power state
- MEMORYSTATESTATUS: Indicates the current memory power state
- INTRANSITION: Indicates an ongoing power state transition from ON power state to inactive, off, or retention, and from inactive, off, or retention power states to on power state

The memory power state is updated by performing a dynamic memory change (the PM_PWSTCTRL_<domain_name> MEMORYCHANGE bit). Software clears the PM_PREPWST_<domain_name> register (this forces a memory change to the same value).

The device has the following power state status registers:

- PM_PWSTST_MPU: MPU domain power state status
- PM_PWSTST_CORE: CORE domain power state status
- PM_PWSTST_SGX: SGX domain power state status
- PM_PWSTST_DSS: DSS domain power state status
- PM_PWSTST_CAM: CAM domain power state status
- PM_PWSTST_PER: PER domain power state status
- PM_PWSTST_NEON: NEON domain power state status
- PM_PWSTST_EMU: EMU domain power state status
- PM_PWSTST_IVA2: IVA2 domain power state status
- PM_PWSTST_USBHOST: USBHOST domain power state status

The MPU domain power state status register indicates the current domain, logic, L1 cache (on or off), and L2 cache (on, retention, or off) power state.

The CORE domain power state status register indicates the current domain, logic (on or off), and memory banks 1 and 2 (on, retention, or off) power state.

The SGX, DSS, CAM, PER, NEON, USBHOST, and EMU domain power state status registers indicate the current domain (on, inactive, retention, or off) power state or that the power domain transition is in progress. Because the SGX memory state is not programmable and reflects the power state, the SGX power domain does not require a memory power state.

The IVA2 domain power state status register indicates the current domain (on, inactive, retention, or off), logic (on or off), L1 cache and flat memory (on, retention, or off), L2 cache and flat memory (on, retention, or off) power state status.

4.12.4.3 PM_PREPWSTST_ <domain_name> (Previous Power State Status Register)

The previous power state status register indicates the power state entered during the last sleep transition. The information in this register is useful when the domain switches back to on state (following a wake-up transition). This register must only be read when the domain power state is on.

This register has the following bit fields:

- LASTPOWERSTATEENTERED: Last domain power state entered after the last sleep transition
- LASTLOGICSTATEENTERED: Last logic power state entered after the last sleep transition
- LASTMEMORYSTATEENTERED: Last memory power state entered after the last sleep transition or before the last memory change update

The device has the following previous power state status registers:

- PM_PREPWSTST _MPU: The MPU power domain previous power state status
- PM_PREPWSTST _CORE: The CORE power domain previous power state status
- PM_PREPWSTST _SGX: The SGX power domain previous power state status
- PM_PREPWSTST _DSS: The DSS power domain previous power state status
- PM_PREPWSTST _CAM: The CAM power domain previous power state status
- PM_PREPWSTST _PER: The PER power domain previous power state status
- PM_PREPWSTST _NEON: The NEON power domain previous power state status
- PM_PREPWSTST _IVA2: The IVA2 power domain previous power state status
- PM_ PREPWSTST _USBHOST: The USBHOST power domain previous power state status

The previous power state status register for the MPU power domain indicates the previous domain, logic, L1 cache (on or off), and L2 cache (on, retention, or off) power state.

The previous power state status register for the CORE power domain indicates the previous domain, logic (on or off), and memory banks 1 and 2 (on, retention, or off) power state.

The previous power state status registers for the SGX, DSS, CAM, PER, NEON, USBHOST and EMU power domains indicate the previous domain (on, inactive, retention, or off) power state. The SGX power domain does not require a memory power state status, because the SGX memory state is not programmable and reflects the power state.

The previous power state status register for the IVA2 power domain indicates the previous domain (on, inactive, retention, or off), logic (on or off), L1 cache and flat memory (on, retention, or off), and L2 cache and flat memory (on, retention, or off) power state status.

The current memory state depends on the setting of the PM_PWSTCTRL_<domain_name> MEMONSTATE bit during the last wake-up transition or during the last memory change operation.

This register must be cleared by software by writing any value to it; this operation must be done when the domain power state is on. Clearing this register does the following:

- Resets the PM_PREPWSTST_<domain_name> LASTPOWERSTATEENTERED bit field and the PM_PREPWSTST_<domain_name> LASTLOGICSTATEENTERED bit to the on state
- Sets the bit fields corresponding to the last memory state of the power domain to the current memory state

Note: Performing a memory change (the PM_PWSTCTRL_<domain_name>[3] MEMORYCHANGE bit for MPU, IVA2, and CORE) has the same effect as clearing this register.

4.12.5 Voltage Management Registers

4.12.5.1 External Voltage Control Register Descriptions

4.12.5.1.1 PRM_VOLTSETUP (Voltage Setup Time Register)

The device has two voltage setup registers:

- PRM_VOLTSETUP1
- PRM_VOLTSETUP2

These registers allow controlling the setup time of the VDD1 and VDD2 power supplies when the device exits the off mode or when the voltage is scaled. It is a constant value that depends on the connected power IC. At power-up reset, this register is not used, because the external power supply is already stable. After boot up, the software must set the correct value to ensure proper sequences when performing voltage scaling or sleep transitions.

The PRM_VOLTSETUP1 register has the following bit fields:

- SETUPTIME1: The setup time of the VDD1 regulator. It is computed as 8 x NbCycles (number of cycles of the system clock), where NbCycles is configured in the register bit field.
- SETUPTIME2: The setup time of the VDD2 regulator. It is computed as 8 x NbCycles (number of cycles of the system clock), where NbCycles is configured in the register bit field.

The PRM_VOLTSETUP1 register is used when the device exits off/retention/sleep mode (or enters off/retention/sleep mode) and manages the sequencing of the voltage regulation steps (the PRM_VOLTCTRL[3] SEL_OFF bit is set to 0).

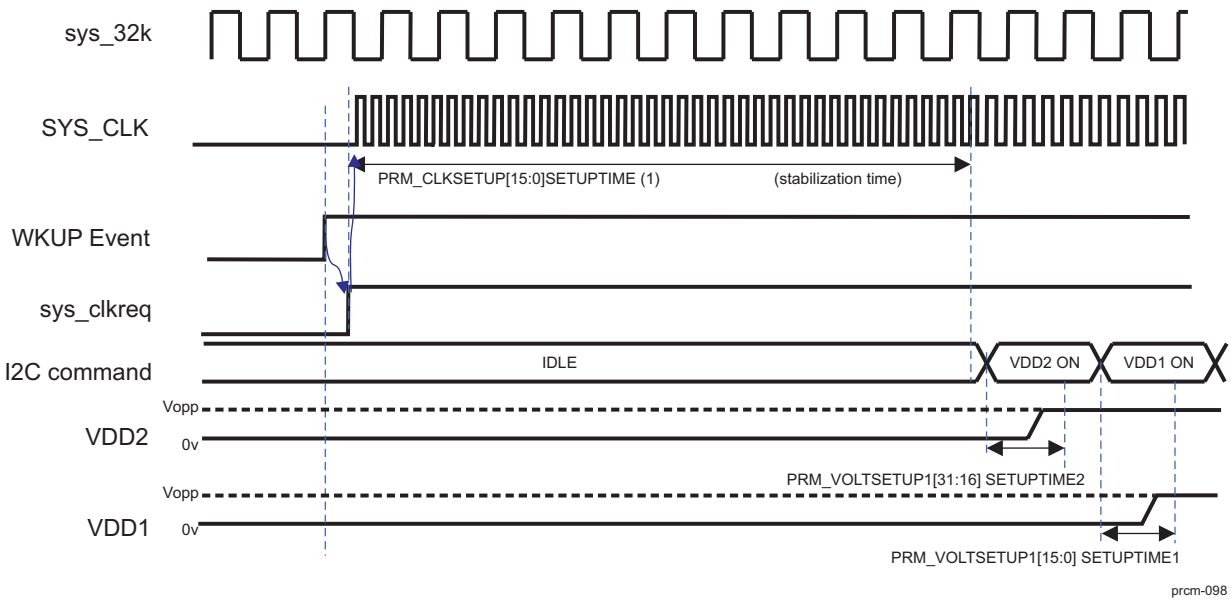
The PRM_VOLTSETUP2 register has the following bit field:

- OFFMODESETUPTIME: The number of 32-kHz clock cycles for the overall setup time of the power management IC when coming back from off mode (VDD1, VDD2, VDD_PLL,..)

The PRM_VOLTSETUP2 register is used only when the device exits off mode and an external power IC manages the sequencing of the voltage regulation steps (PRM_VOLTCTRL[3] SEL_OFF is set to 1).

Figure 4-79 shows off mode wakeup using I^2C.

Figure 4-79. Off Mode Wakeup Using I^2C



(1) PRM_VOLTSETUP1[31:16] SETUPTIME2 and PRM_VOLTSETUP1[15:0] SETUPTIME1 are not used during voltage scaling (between different OPPs).

4.12.5.1.2 PRM_VOLTOFFSET (Voltage Offset Register)

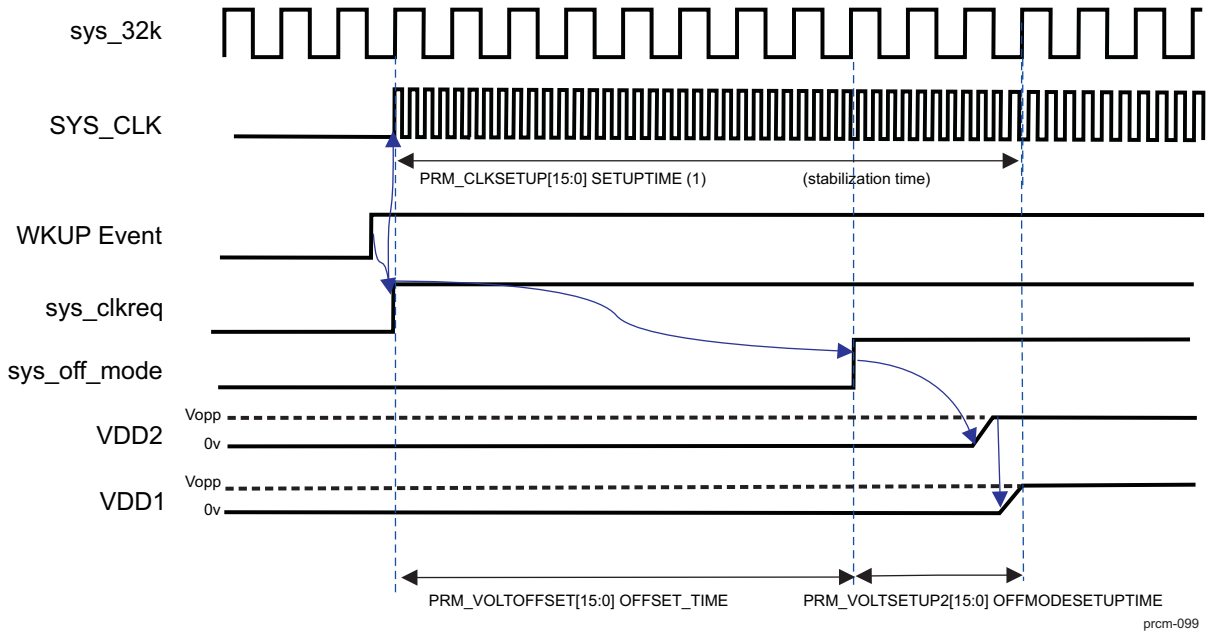
This register allows setting the offset time, which is the time duration between the assertion of sys_clkreq and the deassertion of sys_off_mode when the device is exiting off mode. This time, set as a number of 32-kHz clock cycles, is used to delay the deassertion of sys_off_mode (and as a consequence, the voltage ramping) after the clock has been requested. This is done to avoid waiting for the clock to stabilize when the voltage is already ramped and stabilized, which would induce some leakage.

Program this bit field to match the following equation: PRM_VOLTOFFSET[15:0] OFFSET_TIME = PRM_CLKSETUP[15:0] SETUPTIME – PRM_VOLTSETUP2[15:0] OFFMODESETUPTIME. This way, the voltages and clock are ready at the same time.

This register is used on device wakeup from off mode when the off sequence is supervised by the power IC.

Figure 4-80 shows off mode wakeup using SYS_OFF_MODE.

Figure 4-80. OFF Mode Wakeup Using SYS_OFF_MODE



(1) PRM_CLKSETUP[15:0] SETUPTIME is not used during device cold boot-up sequence.

4.12.5.1.3 PRM_VOLTCTRL (Voltage Source Control Register)

This register allows control of the power IC. The following are the register bit fields:

- AUTO_SLEEP: Automatically sends the sleep command when the voltage domain is in the appropriate standby mode
- AUTO_RET: Automatically sends the retention command when the voltage domain is in the appropriate standby mode
- AUTO_OFF: Automatically sends the off command when the voltage domain is in the appropriate standby mode
- SEL_OFF: Controls whether the off command is sent through the voltage controller I^2C interface or the signal sys_off_mode is asserted when entering off mode
- SEL_VMODE: Allows control of the power IC through either the voltage controller I^2C interface or the VMODE interface

Note: It is mandatory to program the AUTO_SLEEP, AUTO_RET, and AUTO_OFF bit fields in an exclusive manner.

4.12.5.2 Voltage Controller Registers

A specific set of registers allows control of the voltage controller. This register set provides programming flexibility to address different power IC device types through an I^2C interface.

The register set is composed of the following registers:

- Registers to store the addresses on the I^2C bus:
  - PRM_VC_SMPS_SA
  - PRM_VC_SMPS_VOL_RA
  - PRM_VC_SMPS_CMD_RA
- Registers to store the voltage values or voltage commands:
  - PRM_VC_CMD_VAL_0
  - PRM_VC_CMD_VAL_1
- Register to configure the VDD channel:
  - PRM_VC_CH_CONF
- Register to configure the I^2C interface:
  - PRM_VC_I2C_CFG
- Register to use the bypass command:
  - PRM_VC_BYPASS_VAL

4.12.5.2.1 PRM_VC_SMPS_SA (Voltage Controller SMPS Slave Address Register)

This register stores the I^2C slave address value of the power IC device. Two address values can be stored in case two different power IC chips are used to control the two VDD channels (VDD1 and VDD2).

4.12.5.2.2 PRM_VC_SMPS_VOL_RA (Voltage Controller SMPS Voltage Register Address Register)

This register stores the address value of the voltage configuration registers in the power IC device. Two different register address values can be configured.

4.12.5.2.3 PRM_VC_SMPS_CMD_RA (Voltage Controller SMPS Command Register Address Register)

This register stores the address value of the command configuration register in the power IC device. Two different register address values can be configured.

4.12.5.2.4 PRM_VC_CMD_VAL_0 and PRM_VC_CMD_VAL_1 (Voltage Controller Command and Voltage Value Register 0 and 1)

The voltage controller can store two sets of voltage-level commands for each of the four power states (on, low power, retention, and off) for the power IC. The voltage commands depend on the power IC device used.

4.12.5.2.5 PRM_VC_CH_CONF (Voltage Controller Channel Configuration Register)

This register consists of a set of select bits for the VDD1 and VDD2 channels, the slave address, command register address, voltage register address, and voltage-level command settings for power modes (on/low power, on/retention/off). A pair of bit fields can be configured for each of these parameters. The PRM_VC_CH_CONF register allows the allocation of either of the two bit fields (for each of these parameters) to each voltage channel (VDD1 and VDD2). For example, when PRCM.PRM_VC_CH_CONF[0] SA0 = 0x1, the voltage controller allocates PRCM.PRM_VC_SMPS_SA[22:16] SA1 as the slave address for the VDD1 channel. When PRCM.PRM_VC_CH_CONF[0] SA0 = 0x0, the voltage controller allocates PRCM.PRM_VC_SMPS_SA[6:0] SA0 as the slave address for the VDD1 channel.

4.12.5.2.6 PRM_VC_I2C_CFG (Voltage Controller I^2C Interface Configuration Register)

This register allows the setting of the I^2C interface configuration parameters:

- Master code value when I^2C interface is used in HS mode
- Enable and disable the HS mode
- Enable and disable the repeated start-operation mode
- Enable and disable the HSMASTER mode

4.12.5.2.7 PRM_VC_BYPASS_VAL (Voltage Controller Bypass Command Register)

This register is used to address directly the power IC device through the I^2C interface. It has the following bit fields:

- SLAVEADDR: Slave address value of the I^2C interface of the power IC device (similar to PRM_VC_SMPS_SA)
- REGADDR: Address of the command or voltage value register of the power IC device (similar to PRM_VC_SMPS_VOL_RA or PRM_VC_SMPS_CMD_RA)
- DATA: Data to be written to the command or voltage value register
- VALID: Voltage command transaction enable and acknowledge bit

4.12.6 Generic Programming Examples

4.12.6.1 Clock Control

The module clock management is controlled through three programmable steps:

- Enabling or disabling the functional clocks
- Enabling or disabling the interface clocks
- Enabling or disabling the automatic idle mode for the interface clocks

4.12.6.1.1 Enabling and Disabling the Functional Clocks

The flow chart in Figure 4-81 shows how to enable or disable a functional clock.

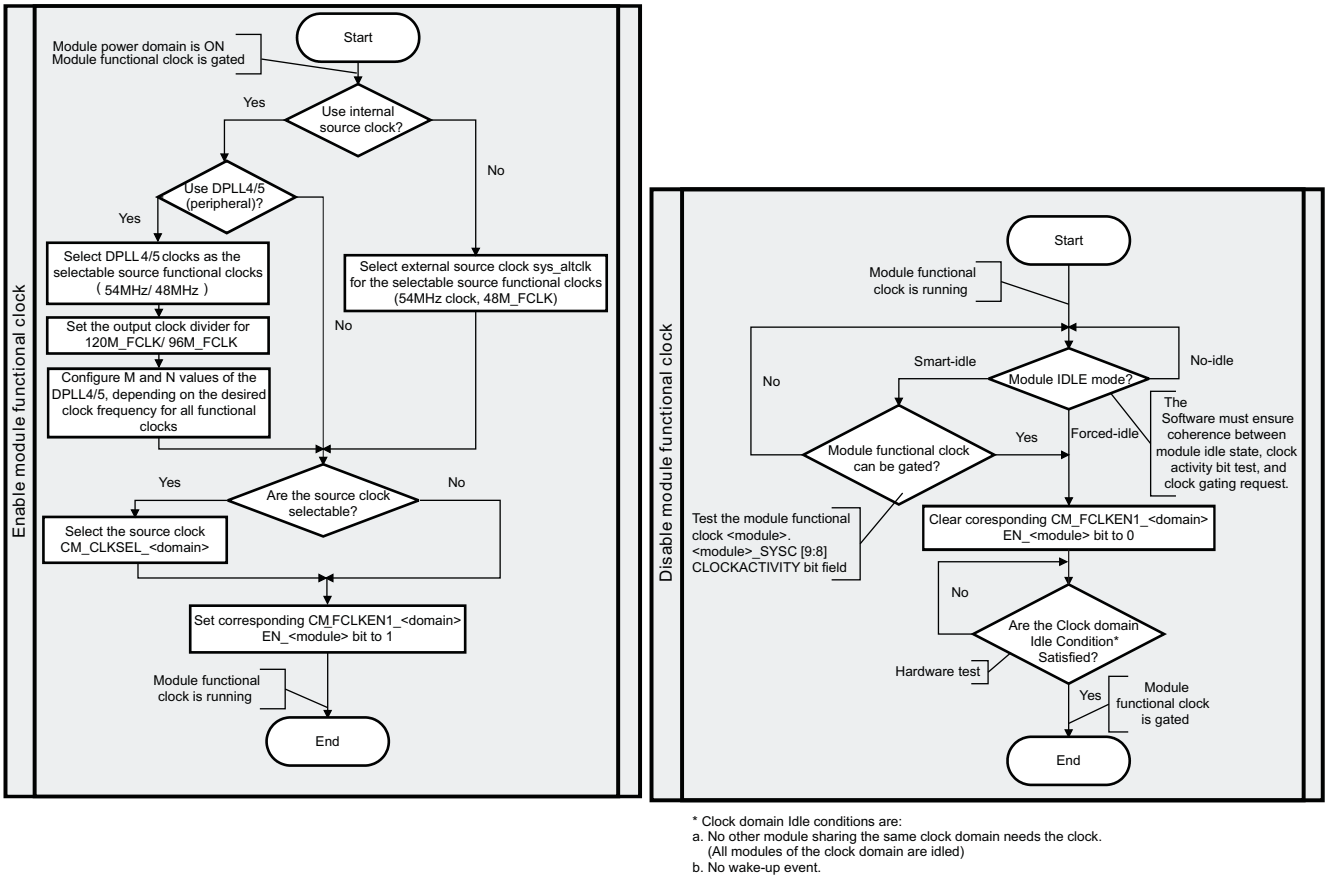
The first step before enabling a functional clock is to select the proper source clock using the corresponding clock selection register (CM_CLKSEL_<domain_name>). It can be either an external source clock or an internal clock; that is, sys_altclk or DPLL4_M3X2_CLK for the DSS_TV_FCLK functional clock of the DSS.

If the source clock is a DPLL3 or DPLL4 output clock, the DPLL multiplier, divider, and output clock ratios are set in the CM_CLKSELn_PLL register, where n is from 1 to 3. The DPLL operating mode is set in the CM_CLKEN_PLL register.

The functional clock is enabled or disabled by writing the dedicated bit in the CM_FCLKEN_<domain_name> register. This bit has a direct effect on the clock activity:

- The functional clock is turned on if the bit is enabled and the clock is not yet active.
- The functional clock is turned off if the bit is disabled and the clock is not required by any other module.

Figure 4-81. Functional Clock Basic Programming Model



* Clock domain Idle conditions are:
a. No other module sharing the same clock domain needs the clock. (All modules of the clock domain are idled)
b. No wake-up event.

prcm-085

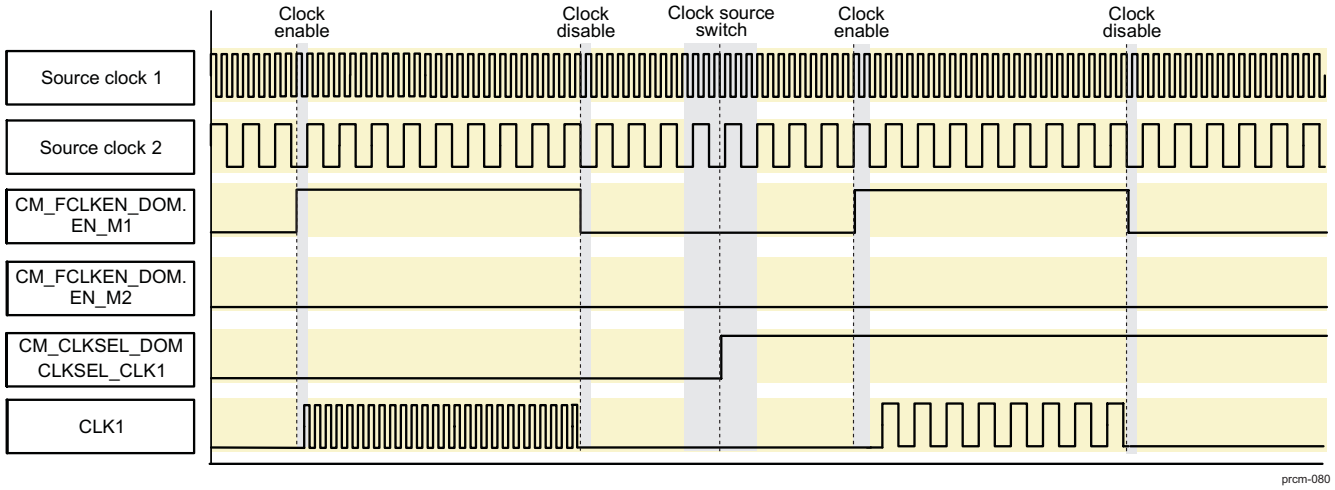
The functional clock must be disabled before switching or scaling its source clock. This means that all the modules using the particular functional clock must not be active during clock switching. To switch a source clock, perform the following sequence:

1. Disable the functional clock by setting the CM_FCLKEN_<power domain>EN_<module> bits to 0.
2. Modify the CM_CLKSEL_<power domain>CLKSEL_<clock> bits to select the new clock source or clock divider.
3. Enable the functional clock by setting the CM_FCLKEN_<power domain>EN_<module> bits to 1.

The timing diagram in Figure 4-82 is a generic example of this sequencing for a functional clock (CLK1). The source clock can be switched between source clock 1 and source clock 2 using the CM_CLKSEL_DOM.CLKSEL_CLK1 bit (source clock 1 is selected when the bit is set to 0; otherwise, source clock 2 is selected). CLK1 can be requested by two modules, M1 and M2. The CM_FCLKEN_DOM.EN_M1 and CM_FCLKEN_DOM.EN_M2 bits control the functional clock enable for the two modules.

Note: The activation or deactivation of the clock is implementation-dependent, not one cycle of the source clock, as shown in Figure 4-82.

Figure 4-82. Functional Clock Switching



The following functional clocks require this switching sequence:

- sys_clkout2
- 48M_FCLK (and all gated versions of it)
- 12M_FCLK (and all gated versions of it)
- DSS_TV_CLK
- GPTx_FCLK (with x = 1, 10, and 11)
- GPTx_ALWON_FCLK (with x = 2 up to 9)

4.12.6.1.2 Enabling and Disabling the Interface Clocks

The flow chart in Figure 4-83 shows the enable/disable sequence of the interface clock.

The first step before enabling an interface clock is to select the proper source clock using the corresponding clock selection register (CM_CLKSEL_<domain>). This register allows selection of the interconnect frequency (L3_ICLK, L4_ICLK) from among several divider ratios.

The interface clock is enabled or disabled by writing the dedicated bit in the CM_ICLKEN_<domain> register. This bit has a direct effect on the clock activity:

- The interface clock is turned on if the bit is enabled and the clock is not yet active.
- The interface clock is turned off if the bit is disabled and the clock is not required by any other module.

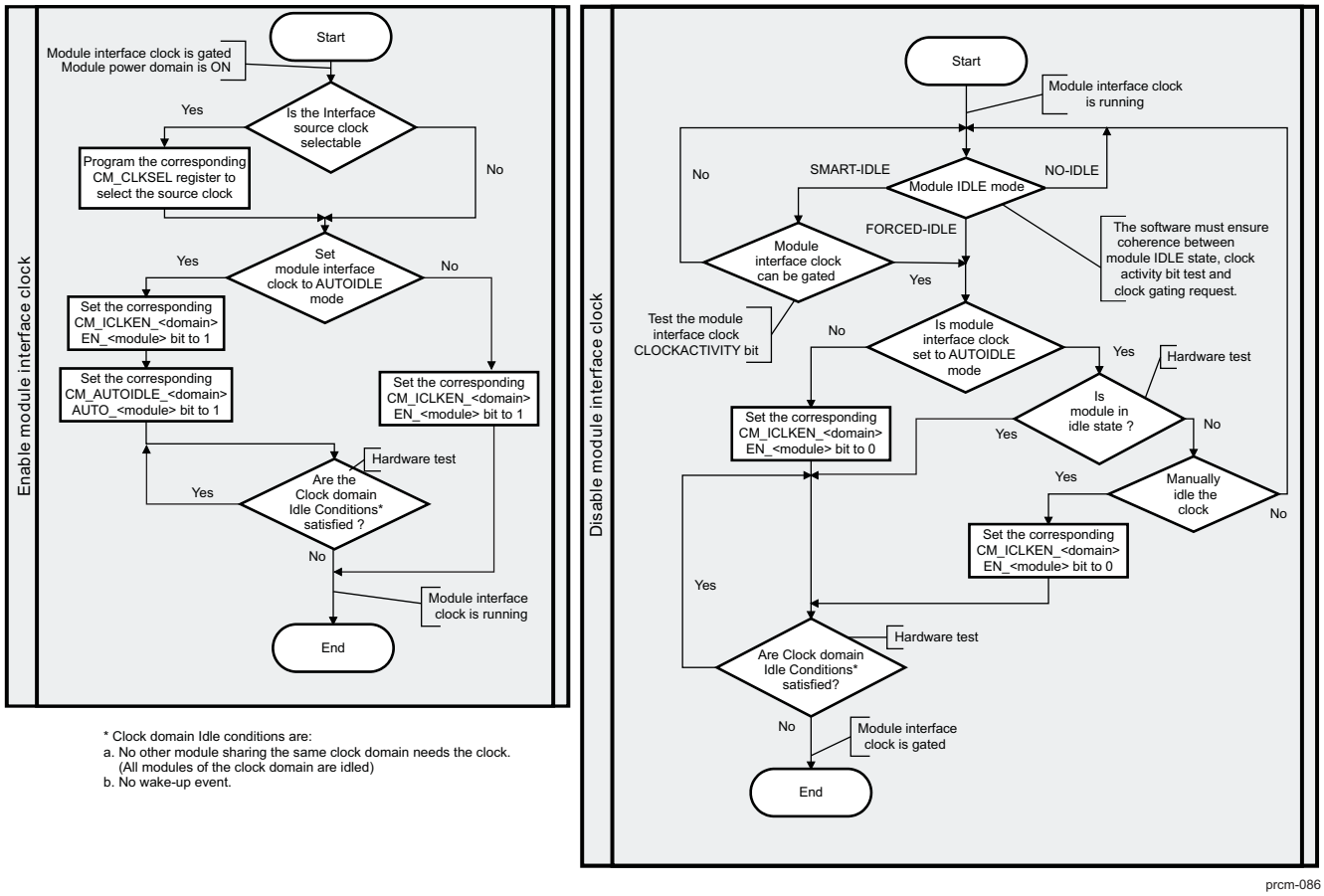
The interface clock can be automatically enabled or disabled by the PRCM module, based on hardware conditions. This automatic clock activity control mode is enabled by writing the corresponding bit in the CM_AUTOIDLE_<domain > register. It takes effect only if the interface clock is enabled (the corresponding bit in the CM_ICLKEN_<domain > is set to 1).

The hardware conditions for automatic gating (deactivation) of the clock are as follows:

- The module activity; that is, the module is inactive.
- The domain activity; that is, all modules in the domain are inactive.

The software can read the idle status register CM_IDLESTAT_<domain> at any time to know whether the module is accessible. A module is inaccessible if its idle status bit is set. Accessing an idle module generates an error (if the interface clock is still running) or a time-out (if the interface clock is cut).

Figure 4-83. Interface Clock Basic Programming Model



The frequency ratio between the CORE_CLK, the L3_ICLK, and the L4_ICLK is configured by setting the corresponding CM_CLKSEL_CORE register bit fields. This configuration must be done before switching DPLL3 to lock mode. In this way, the clock ratio is switched while DPLL3 is operating at system clock frequency, and then only DPLL3 is switched to high-frequency locked mode.

If the configuration of the interface clock needs to be changed, first put DPLL3 in bypass mode, select the new configuration, and then relock DPLL3.

Note: When performing frequency scaling, the clock division can be done directly by programming the DPLL output divider. In this case, there is no need to change the configuration of the interface clock.

4.12.6.1.3 Enabling and Disabling the Inactive State

The flow chart in Figure 4-84 shows how to put a domain into inactive state. This state is required before any power transition from on state to retention or off state.

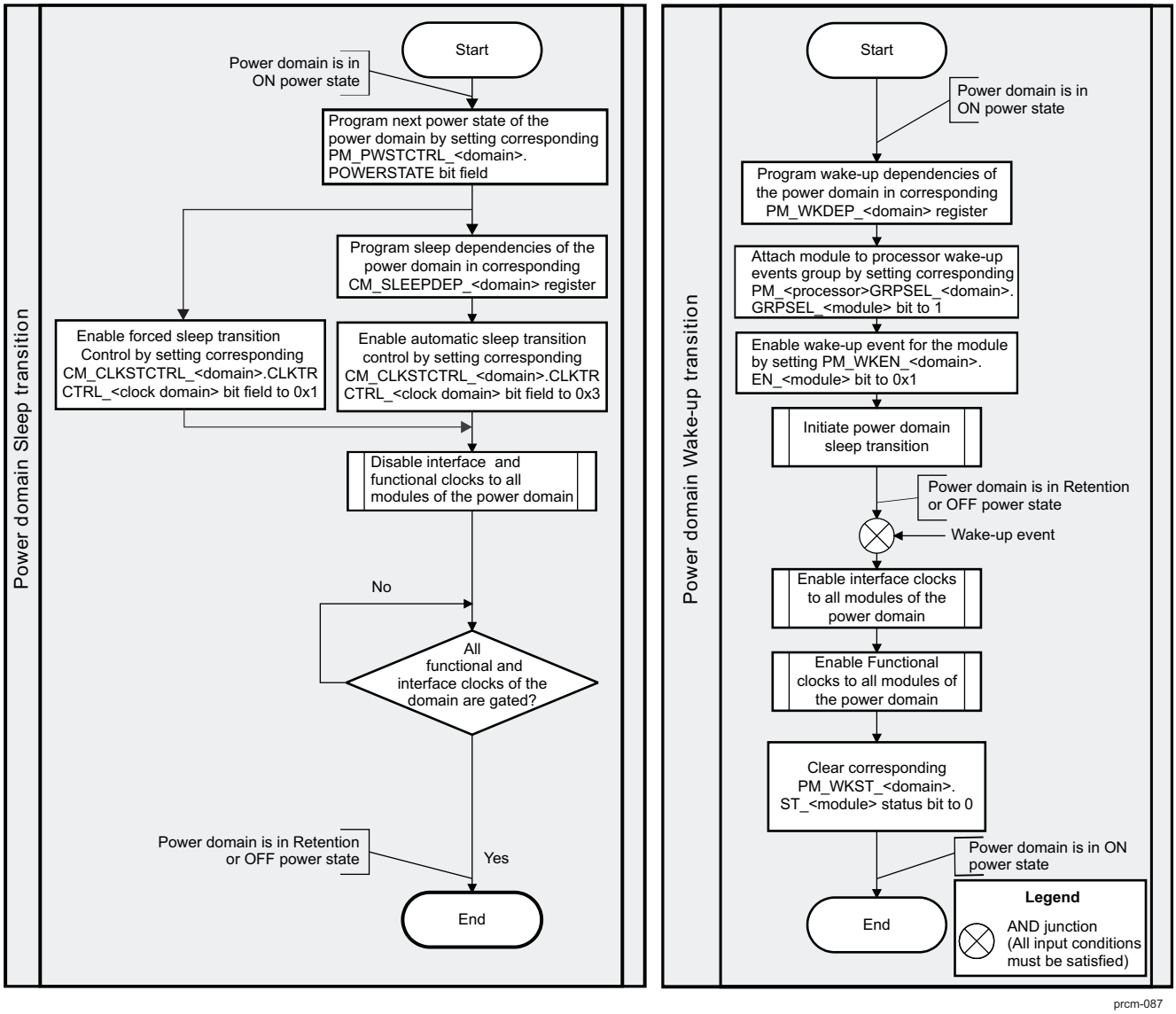
A domain is said to be in inactive state if:

- All the functional and interface clocks of the domain are gated (deactivated).
- All the initiator modules are in standby mode.
- All the dependent domains have reached their mute state. The sleep dependency between the power domains is configured by programming the CM_SLEEPDEP_<domain> register.

The domain transition from active state to inactive state is effective only if the CM_CLKSTCTRL_<domain> register is programmed for hardware-supervised state transition.

The CM_CLKSTST_<domain> register identifies whether a power domain or a clock domain within the power domain is accessible. A domain is inaccessible if its corresponding bit in the CM_CLKSTST_<domain > register is set to 1.

Figure 4-84. Domain Inactive STATE Basic Programming Model



4.12.6.1.4 Processor Clock Control

The flow chart in Figure 4-85 shows the control sequence of the processor clock.

The processor source clock is generated by the dedicated processor DPLL (DPLL1 and DPLL2). After power up, the processor DPLL is in bypass or stop mode. This means the processor clock is either the system clock or is shut off.

The first step is to program the multiplier and divider ratios of the processor DPLL. Those two values are written in the CM_CLKSEL1_PLL_<processor> register.

For frequency scaling, the processor DPLL integrates an additional divider to scale down the synthesized clock. The value of this second divider is written in the CM_CLKSEL2_PLL_<processor> register. This divider can be configured dynamically (while the processor executes instructions), and the DPLL output clock is scaled without any glitches.

The processor DPLL must be locked at a desired frequency, provided the clock frequency is higher than the system clock. It can then be set to low-power bypass mode when the high-frequency clock is not required. The DPLL operating mode (locked or bypassed) is controlled by programming the CM_CLKEN_PLL_<processor> register.

The processor clock can be switched automatically, based on hardware conditions, between the processor DPLL synthesized clock and a bypass clock. This mode is enabled by programming the corresponding mode in the CM_AUTOIDLE_PLL_<processor> register. It takes effect only if the processor DPLL is locked (the CM_CLKEN_PLL_<processor> register is set in lock mode).

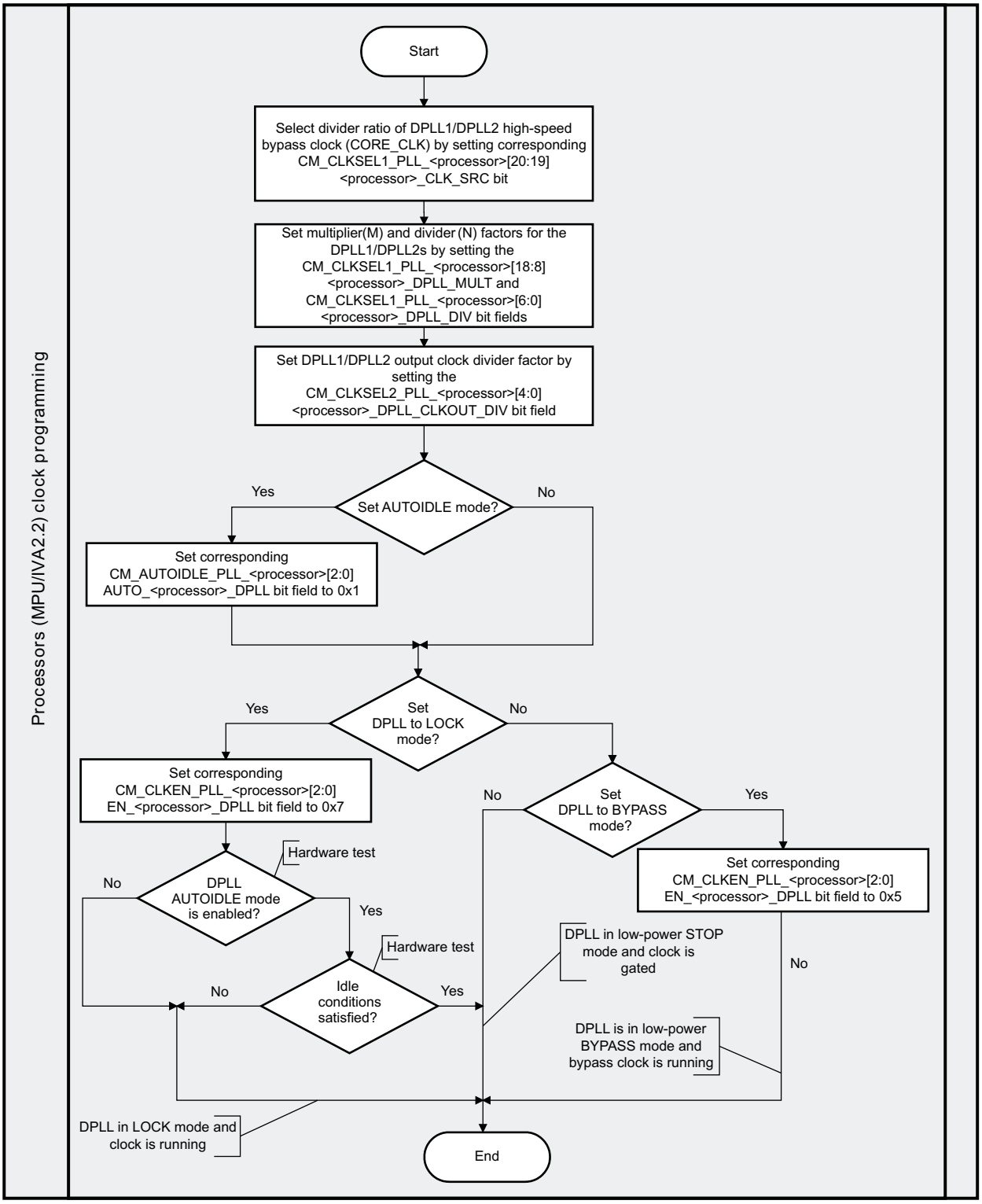
The HS bypass clock for the processor DPLL can be:

- The DPLL3 output clock (CORE_CLK)
- The DPLL3 output clock (CORE_CLK) divided by 2

The HS bypass clock is selected by programming the CM_CLKSEL1_PLL_<processor > register.

The software can read the CM_IDLEST_PLL_<processor > register at any time to determine whether the DPLL is in lock mode (DPLL output is a high-frequency synthesized clock) or bypass mode (DPLL output is not the synthesized clock; the DPLL output is the bypass clock, or the DPLL is in transitioning state).

Figure 4-85. Processor Clock Basic Programming Model



4.12.6.2 Reset Management

The reset sequence is hardware-driven. On power on, once all the reset sources have been released, the PRM holds the entire device under reset long enough to ensure the stabilization of the power IC voltages and the oscillator system clock frequency. This reset delay is programmed in the PRM_RSTTIME register.

The IVA2.2 and the modem domain resets are held active after power up. They are released by writing to the corresponding bits in the RM_RSTCTRL_<domain> register.

A domain reset status register (PRM_RSTST_domain>) identifies the source of the current reset applied to the domain. The software must clear this status bit after reset.

A global reset status register (PRM_RSTST) provides information about the global source of resets. All sources of warm reset are logged separately in this register, and all sources of cold reset are logged in a common status bit.

4.12.6.3 Wake-Up Control

The flow chart in Figure 4-86 shows the control sequence of the module wake-up event.

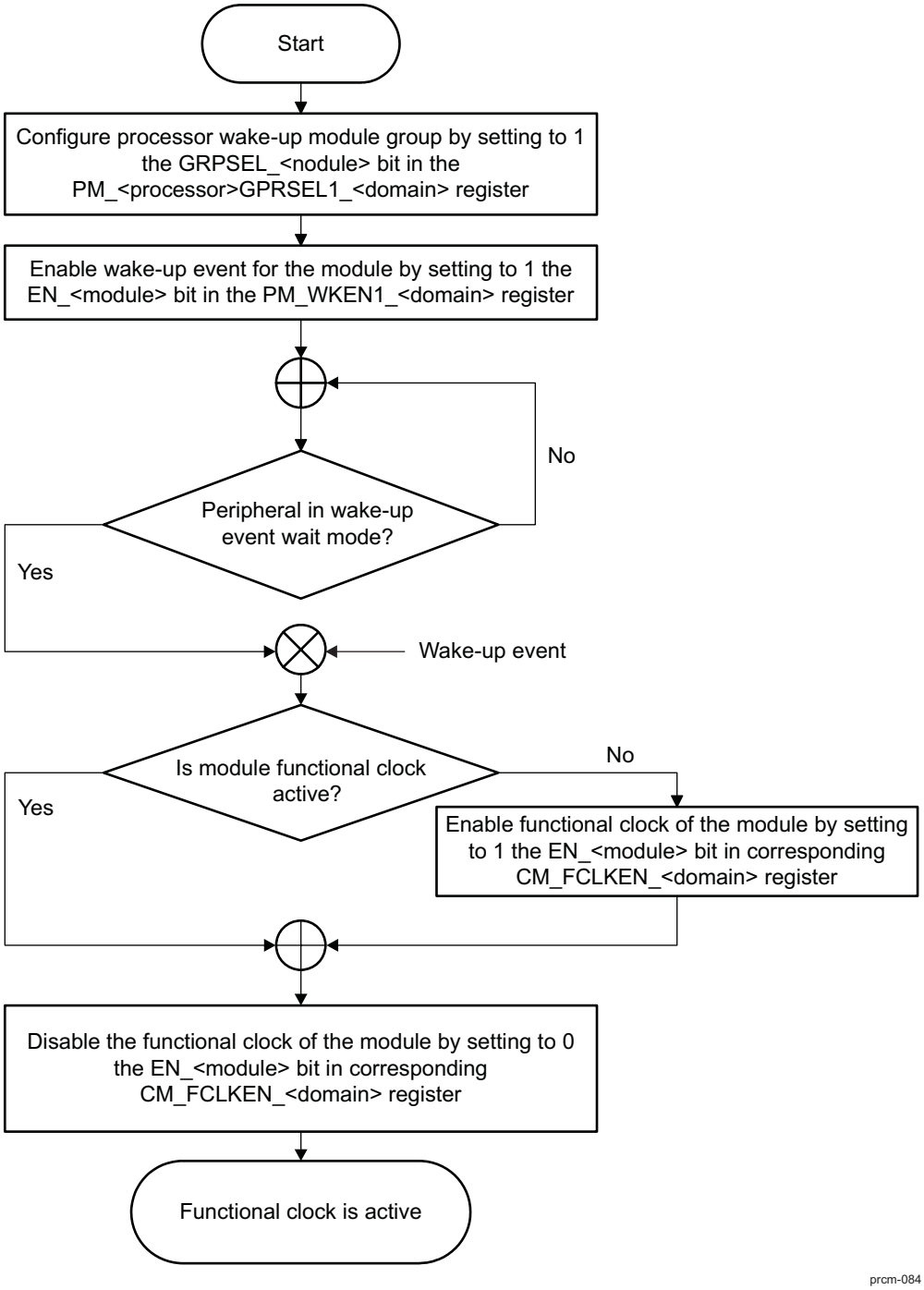
This procedure consists of the following steps:

1. Program the PRCM module to consider the wake-up event.
2. Switch to idle mode and wait for the wake-up event.
3. Wake up on the wake-up event and activate the module functional clock.
4. Acknowledge the wake-up event.

The peripheral that can generate a wake-up event must be attached to a group of wake-up event generating modules for one or both processors by programming the PM_<processor>GRPSEL register. Writing 1 to this register allows the corresponding processor to be wakened on a peripheral wake-up event, assuming that the peripheral wake-up capability has been enabled by programming the register PM_WKEN_<domain>.

After this is configured, the PRCM module initiates a wake-up procedure on receiving the peripheral wake-up event. The peripheral functional clock must be reenabled by programming the CM_FCLKEN_<domain > register, and then the wake-up event can be acknowledged by clearing the PM_WKST_<domain > register.

Figure 4-86. Wake-up Basic Programming Model

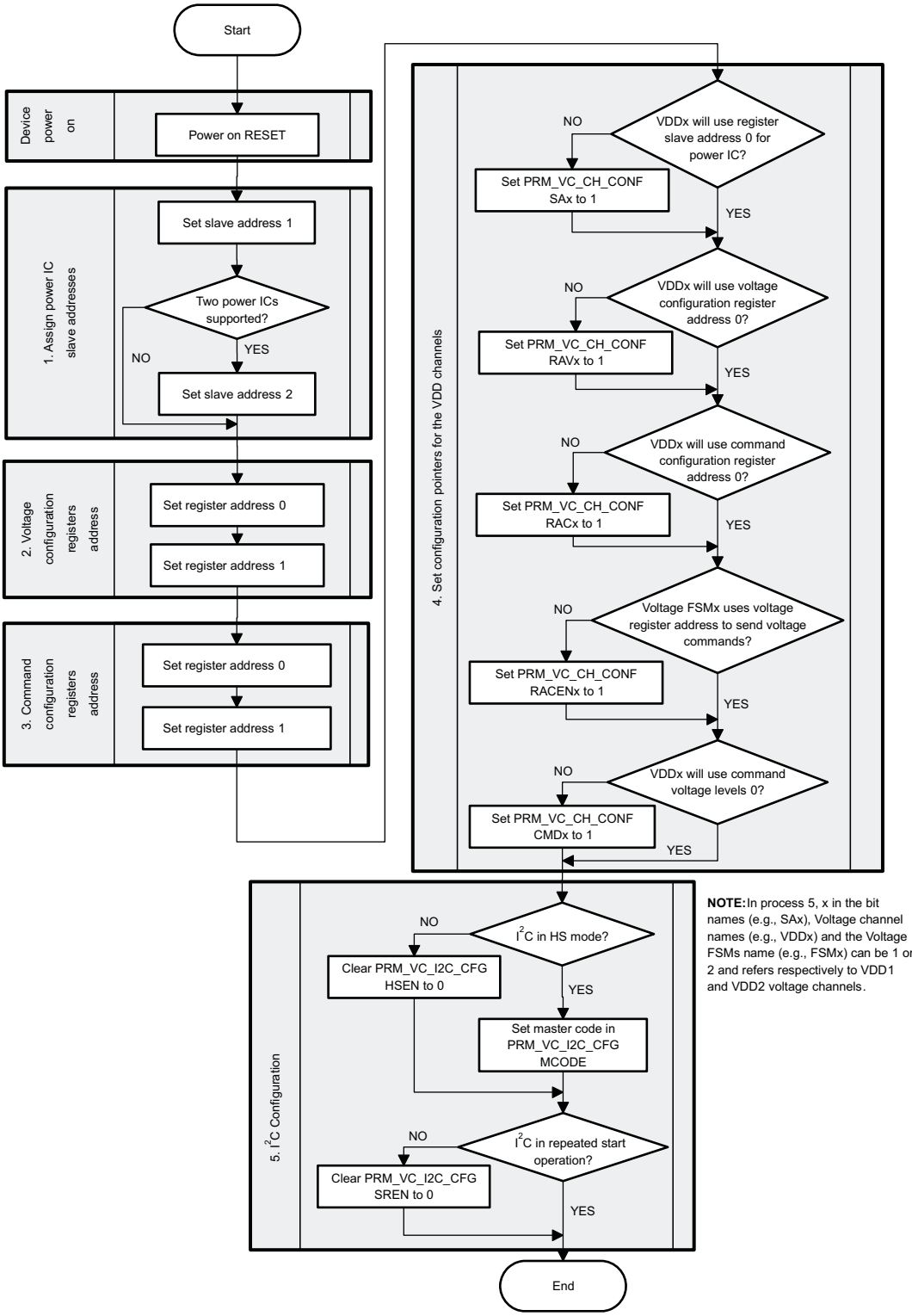


A power domain A can have a functional dependency on a power domain B. Thus, when power domain A wakes up, it may be necessary to wake up power domain B. This wake-up dependency is hardcoded for the CORE power domain, but is programmable for the other power domains through the PM_WKDEP_<domain> register.

4.12.6.4 Voltage Controller Initialization Basic Programming Model

Figure 4-87 is the flow chart for voltage controller initialization.

Figure 4-87. Voltage Controller Initialization Flow Chart



1. Assign power IC slave addresses.

To support the I²C communication with external power ICs, the slave address of the power ICs must be configured in the voltage controller registers. The voltage controller can support two slave addresses to control VDD1 and VDD2 independently, from two different power ICs. At least one valid slave address is required to communicate with a power IC.

Note: The slave address for the following depends on the configured slave address of the power IC:

- PRCM.PRM_VC_SMPS_SA[6:0] SA0
- PRCM.PRM_VC_SMPS_SA[22:16] SA1

2. Set voltage configuration register addresses.

   The addresses of the voltage configuration registers of the power ICs are set in the corresponding bit fields. The voltage controller can support addresses of two different voltage configuration registers (belonging to same or different power ICs).

PRCM.PRM_VC_SMPS_VOL_RA[7:0] VOLRA0		Depend on the characteristics of the connected power ICs.
PRCM.PRM_VC_SMPS_VOL_RA[23:16] VOLRA1		

3. Set command configuration register addresses.

   The addresses of the command configuration registers of the power ICs are set in the corresponding bit fields. The voltage controller can support addresses of two different command configuration registers (belonging to same or different power ICs).

PRCM.PRM_VC_SMPS_CMD_RA[7:0] CMDRA0		Depend on the characteristics of the connected power ICs.
PRCM.PRM_VC_SMPS_CMD_RA[23:16] CMDRA1		

4. Set configuration pointers for the VDD channels.

   The configuration pointers allow the selection of one of four configurations for each voltage channel (VDD1 and VDD2):

   - Two slave I²C interfaces (slave address)
   - Two voltage configuration registers
   - Two command configuration registers
   - Two power-mode voltage levels

PRCM.PRM_VC_CH_CONF[16] SA1		Slave address pointer (SA1 for VDD2, and SA0 for VDD1)
PRCM.PRM_VC_CH_CONF[0] SA0		
PRCM.PRM_VC_CH_CONF[17] RAV1		Voltage configuration register address pointer (RAV1 for VDD2, and RAV0 for VDD1)
PRCM.PRM_VC_CH_CONF[1] RAV0		
PRCM.PRM_VC_CH_CONF[18] RAC1		Command configuration register address pointer (RAC1 for VDD2, and RAC0 for VDD1)
PRCM.PRM_VC_CH_CONF[2] RAC0		
PRCM.PRM_VC_CH_CONF[19] RACEN1		Select voltage or command configuration register for FSM commands (RACEN1 for VDD2, and RACEN0 for VDD1)
PRCM.PRM_VC_CH_CONF[3] RACEN0		
PRCM.PRM_VC_CH_CONF[20] CMD1		Command voltage level selection pointer (CMD1 for VDD2, and CMD0 for VDD1)
PRCM.PRM_VC_CH_CONF[4] CMD0		

5. Configure I²C.

   At power-on reset, the I2C4 is in HS mode. In HS mode, a master code value must be configured for the preamble I²C HS transmission. If the external power ICs do not support I²C HS mode, the interface can be switched to fast/standard (F/S) mode. By default, a repeated-start operation for communication is enabled. If necessary, it can be disabled.

   For more information, see Chapter 18, *I²C*.

PRCM.PRM_VC_I2C_CFG[2:0] MCODE		Master code for I^2C HS preamble
PRCM.PRM_VC_I2C_CFG[3] HSEN	0x0	Switch to F/S mode.
PRCM.PRM_VC_CH_CONF[4] SREN	0x0	Disable repeated start operation.

4.12.6.5 Event Generator Programming Examples

The event generator feature allows the MPU power domain to be switched between on and off (or placed in idled mode). This is intended to implement an efficient activity modulation of the MPU power state with minimum software support.

When the event generator is activated, the PRCM module ensures that the CORE power domain always follows the MPU power domain activity.

The PRCM.PM_EVGENONTIM_MPU and PRCM.PM_EVGENOFFTIM_MPU registers of the event generator counter allow the configuration of the on and off durations (the number of system clock cycles) for the MPU power domain. The PRCM.PM_EVGENCTRL_MPU register allows the enabling/disabling of the event generator feature and control of the way on and off values are loaded in the counter.

There are three ways to load the counter with the next counting value:

- Load on update of the corresponding PRCM.PM_EVGENONTIM_MPU or PRCM.PM_EVGENOFFTIM_MPU register.
- Load the on-time value when the MPU power domain wakes up, and the off-time value when the MPU power domain starts the sleep transition (the MPU executes the WFI instruction).
- Automatically load the on-time value when the off-time value expires, and automatically load the off-time value when the on-time value expires.

When the counter times out at the end of the on/off time, it triggers the interrupt/wake-up transition, respectively. The software can use the on-time interrupt to trigger the WFI processor instruction to idle the processor clock. Similarly, the wake-up event at the end of the off time restarts the processor clock and interrupts the processor. To enable the corresponding interrupts, the MPU interrupts mask must be set in the PRCM.PRM_IRQENABLE_MPU register.

4.13 PRCM Use Cases and Tips

4.13.1 Voltage Control Using VMODE

4.13.1.1 Introduction

The PRCM module allows direct simple control of VDD1 and VDD2 through the dedicated signals sys_nvmode1 and sys_nvmode2. A single voltage value can be linked in the external power IC to a given state of sys_nvmode1 or sys_nvmode2. This allows direct voltage management (see Table 4-85).

Table 4-85. VDD1 and VDD2 Voltage Control Through VMODE

sys_nvmode Signals	Voltage Values	Comments
sys_nvmode1 is high.	VDD1 value 1	VDD1 value 1 depends on the power IC programming.
sys_nvmode1 is low.	VDD1 value 2	VDD1 value 2 depends on the power IC programming.
sys_nvmode2 is high.	VDD2 value 1	VDD2 value 1 depends on the power IC programming.
sys_nvmode2 is low.	VDD2 value 2	VDD2 value 2 depends on the power IC programming.

Note: The polarity of sys_nvmode1 and sys_nvmode2 is programmable. This means that the PRCM module sets them high or low, depending on the user settings. For this reason, the external power IC must be configured to provide a given voltage level depending on the states of the sys_nvmode*x* signals.

Assertion of sys_nvmode1 and sys_nvmode2 (low or high, depending on the chosen polarity) occurs immediately after a device sleep or wake-up transition. When the PRCM module triggers a sleep transition (ON to OFF, or ON to retention), it deasserts sys_nvmode1 and sys_nvmode2. When the PRCM module detects a device wake-up transition (retention to on, or off to on), it asserts sys_nvmode1 and sys_nvmode2.

Note: sys_nvmode1 and sys_nvmode2 are managed as one signal from the PRCM standpoint. In other words, their settings and assertion conditions are managed globally, without any granularity. The power IC software sets different values for VDD1 and VDD2, if necessary.

From the PRCM standpoint, using the VMODE signals is mutually exclusive with using the dedicated I2C4 interface to the external power IC; sys_nvmode*x* signals share the same physical interface as I2C4 at device boundary.

Using VMODE voltage control is an easy way to switch VDD1 and VDD2 voltage values upon device power state transitions. The following section gives the programming steps to achieve voltage management through VMODE signals.

4.13.1.2 Programming Sequence

4.13.1.2.1 Initialization Procedure

1. Configure sys_nvmode1 and sys_nvmode2 at the device level.

CONTROL.CONTROL_PADCONF_I2C4_SCL[15:0]	Mux mode1
CONTROL.CONTROL_PADCONF_I2C4_SCL[31:16]	Mux mode1

The correct multiplexing mode is selected at the device level to select sys_nvmode1 and sys_nvmode2 on the balls. This is done in the SCM registers, as indicated. For further details, see Chapter 7, *System Control Module*.

2. Activate VMODE control.

PRM_VOLTCTRL[4] SEL_VMODE	0x1

VMODE control is selected in the PRCM module. This also ensures the other interfaces to the power IC are correctly deactivated and managed.

3. Set the polarity of sys_nvmode1 and sys_nvmode2.

PRM_POLCTRL[0] EXTVOL_POL	0x1 => sys_nvmodex signals are active high

Depending on the polarity set, the VMODE signals remain high or low until a transition is initiated. Their states then automatically toggle to the opposite state and return when the reverse transition is initiated.

4. Set the voltage stabilization delays.

PRM_VOLTSETUP1[31:16] SETUP_TIME2	VDD2 voltage stabilization time
PRM_VOLTSETUP1[15:0] SETUP_TIME1	VDD1 voltage stabilization time

SETUP_TIME1 and SETUP_TIME2 are power IC-dependent parameters for voltage stabilization time for VDD1 and VDD2, respectively. For further details, see the power IC documentation.

5. Configure the power IC.

This step relies entirely on power IC software settings. The purpose here is to link a voltage value to both VDD1 and VDD2 depending on the state of the sys_nvmode*x* signals. For further details, see the power IC documentation.

4.13.1.2.2 VMODE Signals Toggling

1. Has a sleep/wake-up transition been initiated?

As stated previously, when VMODE voltage control is selected, the PRCM module automatically toggles sys_nvmode*x* signals and device sleep and wake-up transitions. This means the user has nothing more to control except the device transitions, as explained in this document. When a transition is initiated, sys_nvmode1 and sys_nvmode2 toggle according to their programmed polarity and alert the power IC to switch VDD1 and VDD2 to the programmed values.

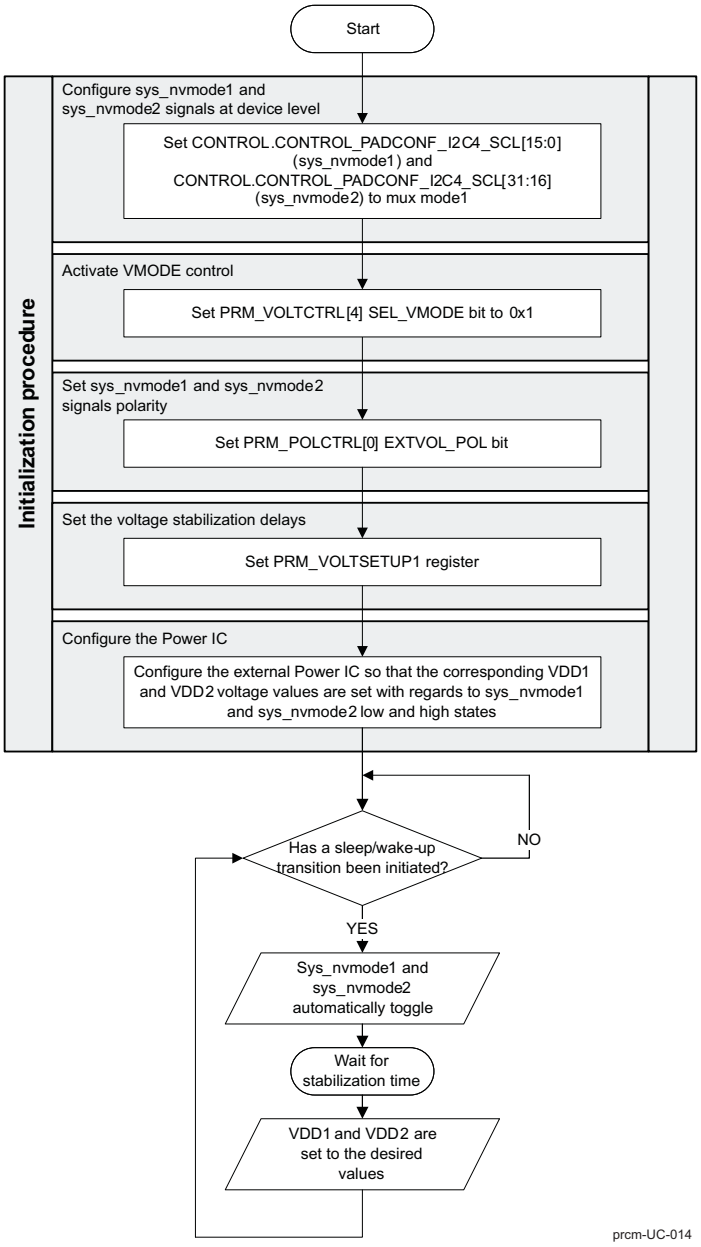
2. Wait for stabilization time.

Depending on the SETUP_TIME1 and SETUP_TIME2 settings, the PRCM module starts a countdown immediately after the VMODE signal state changes. This ensures that the voltages provided by the power IC are stable before switching the device to the desired power state.

4.13.1.2.3 Summary Flow Chart

Figure 4-88 is a flow chart of VMODE voltage control.

Figure 4-88. Voltage Control Through VMODE Flow Chart



4.14 PRCM Register Manual

This section gives information about all modules and features in the high-tier device. See Section 1.5, *OMAP34xx Family*, to check availability of modules and features. For power-management saving consideration, ensure that power domains of unavailable features and modules are switched off and clocks are cut off.

Table 4-86 lists the Instance Summary of the CM modules. Table 4-87 through Table 4-285 provide register mapping summaries of the CM registers and describe the bits in the individual registers.

Table 4-287 lists the Instance Summary of the PRM modules. Table 4-288 through Table 4-498 provide register mapping summaries of the PRM registers and describe the bits in the individual registers.

4.14.1 CM Module Registers

4.14.1.1 CM Module Summary

Table 4-86. CM Instance Summary

Module Name	Base Address (hex)	Size
IVA2_CM	0x4800 4000	8192 bytes
OCP_System_Reg_CM	0x4800 4800	8192 bytes
MPU_CM	0x4800 4900	8192 bytes
CORE_CM	0x4800 4A00	8192 bytes
SGX_CM	0x4800 4B00	8192 bytes
WKUP_CM	0x4800 4C00	8192 bytes
Clock_Control_Reg_CM	0x4800 4D00	8192 bytes
DSS_CM	0x4800 4E00	8192 bytes
CAM_CM	0x4800 4F00	8192 bytes
PER_CM	0x4800 5000	8192 bytes
EMU_CM	0x4800 5100	8192 bytes
Global_Reg_CM	0x4800 5200	8192 bytes
NEON_CM	0x4800 5300	8192 bytes
USBHOST_CM	0x4800 5400	8192 bytes

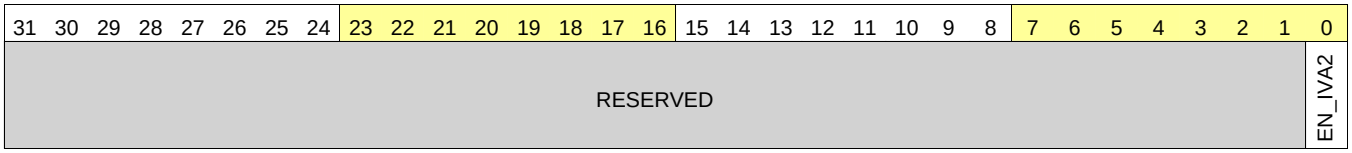
4.14.1.2 IVA2_CM Registers

Table 4-87. IVA2_CM Register Summary

Register Name	Type	Register Width (Bits)	Address Offset	Physical Address	Reset Type
CM_FCLKEN_IVA2	RW	32	0x0000 0000	0x4800 4000	W
CM_CLKEN_PLL_IVA2	RW	32	0x0000 0004	0x4800 4004	W
CM_IDLEST_IVA2	R	32	0x0000 0020	0x4800 4020	C
CM_IDLEST_PLL_IVA2	R	32	0x0000 0024	0x4800 4024	C
CM_AUTOIDLE_PLL_IVA2	RW	32	0x0000 0034	0x4800 4034	W
CM_CLKSEL1_PLL_IVA2	RW	32	0x0000 0040	0x4800 4040	W
CM_CLKSEL2_PLL_IVA2	RW	32	0x0000 0044	0x4800 4044	W
CM_CLKSTCTRL_IVA2	RW	32	0x0000 0048	0x4800 4048	W
CM_CLKSTST_IVA2	R	32	0x0000 004C	0x4800 404C	C

Table 4-88. CM_FCLKEN_IVA2

Address Offset	0x0000 0000		
Physical Address	0x4800 4000	**Instance**	IVA2_CM
Description	This register controls the IVA2 domain functional clock activity.		
Type	RW		



Bits	Field Name	Description	Type	Reset
31:1	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x00000000
0	EN_IVA2	IVA2 functional clock control 0x0: IVA2_CLK is disabled 0x1: IVA2_CLK is enabled	RW	0x0

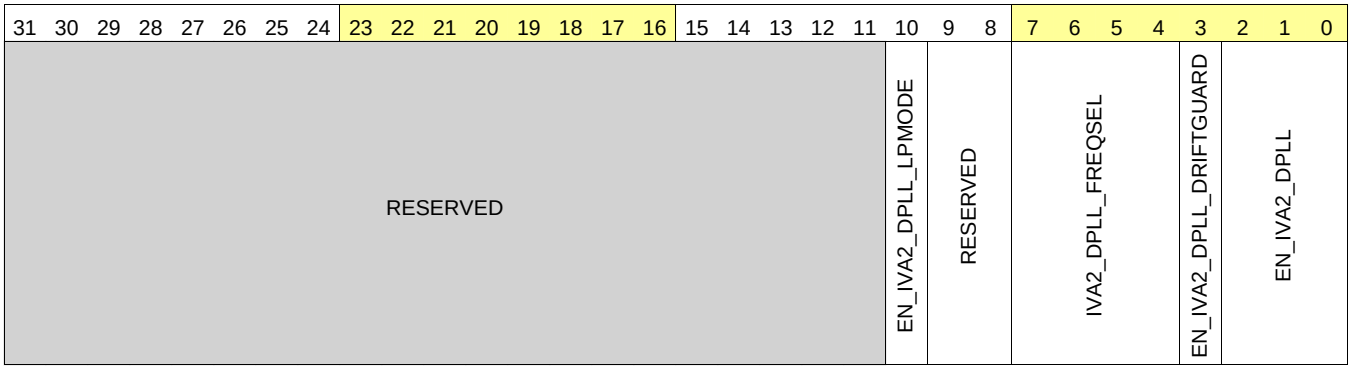
Table 4-89. Register Call Summary for Register CM_FCLKEN_IVA2

PRCM Basic Programming Model
- Power-Domain Clock Control Registers: [0]

PRCM Register Manual
- IVA2_CM Registers: [1]

Table 4-90. CM_CLKEN_PLL_IVA2

Address Offset	0x0000 0004		
Physical Address	0x4800 4004	**Instance**	IVA2_CM
Description	This register controls the IVA2 DPLL modes.		
Type	RW		



Bits	Field Name	Description	Type	Reset
31:11	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x000000
10	EN_IVA2_DPLL_LPMODE	This bit allows to enable or disable the LP mode of the IVA2 DPLL. Writting this bit to switch the mode between LP or normal mode will take effect only when the DPLL will have transition into the bypass or stop state, followed by a lock or re-lock of the DPLL. 0x0: Disables the DPLL LP mode to re-enter the normal mode at the following lock or re-lock sequence. 0x1: Enables the DPLL LP mode to enter the LP mode at the following lock or re-lock sequence.	RW	0x0

Bits	Field Name	Description	Type	Reset
9:8	RESERVED	Write 0s for future compatibility. Read returns 0.	RW	0x0
7:4	IVA2_DPLL_FREQSEL	This bit field allows selecting the proper range of the IVA2 DPLL internal frequency depending on the DPLL reference clock and the N divider. 0x3: 0.75 MHz—1.0 MHz 0x4: 1.0 MHz—1.25 MHz 0x5: 1.25 MHz—1.5 MHz 0x6: 1.5 MHz—1.75 MHz 0x7: 1.75 MHz—2.1 MHz 0xB: 7.5 MHz—10 MHz 0xC: 10 MHz—12.5 MHz 0xD: 12.5 MHz—15 MHz 0xE: 15 MHz—17.5 MHz 0xF: 17.5 MHz—21 MHz	RW	0x1
3	EN_IVA2_DPLL_DRIFTGUARD	This bit allows to enable or disable the automatic recalibration feature of the IVA2 DPLL. The IVA2 DPLL will automatically start a recalibration process upon assertion of the recal flag if this bit is set. 0x0: Disables the IVA2 DPLL automatic recalibration mode 0x1: Enables the IVA2 DPLL automatic recalibration mode	RW	0x0
2:0	EN_IVA2_DPLL	IVA2 DPLL control; Other enums: Reserved 0x1: Put the IVA2 DPLL in low power stop mode 0x5: Put the IVA2 DPLL in low power bypass mode 0x7: Enables the IVA2 DPLL in lock mode	RW	0x1

Table 4-91. Register Call Summary for Register CM_CLKEN_PLL_IVA2

PRCM Clock Manager Functional Description • DPLL Modes: [0] • DPLL Low-Power Mode: [1] • Latencies : • Recalibration: [3] • DPLL Source-Clock Controls: [4]
PRCM Basic Programming Model • DPLL Clock Control Registers: [5]
PRCM Register Manual • IVA2_CM Registers: [6]

Table 4-92. CM_IDLEST_IVA2

Address Offset	0x0000 0020		
Physical Address	0x4800 4020	**Instance**	IVA2_CM
Description	IVA2 standby status and access availability monitoring. This register is read only and automatically updated.		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																															ST_IVA2

Bits	Field Name	Description	Type	Reset
31:1	RESERVED	Read returns 0.	R	0x00000000
0	ST_IVA2	IVA2 standby status. 0x0: IVA2 sub-system is active. 0x1: IVA2 sub-system is in standby mode.	R	0x1

Table 4-93. Register Call Summary for Register CM_IDLEST_IVA2

PRCM Basic Programming Model • Power-Domain Clock Control Registers: [0]
PRCM Register Manual • IVA2_CM Registers: [1]

Table 4-94. CM_IDLEST_PLL_IVA2

Address Offset	0x0000 0024		
Physical Address	0x4800 4024	**Instance**	IVA2_CM
Description	This register allows monitoring the master clock activity. This register is read only and automatically updated.		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																															ST_IVA2_CLK

Bits	Field Name	Description	Type	Reset
31:1	RESERVED	Read returns 0.	R	0x00000000
0	ST_IVA2_CLK	IVA2_CLK activity 0x0: IVA2 DPLL is bypassed 0x1: IVA2 DPLL is locked	R	0x0

Table 4-95. Register Call Summary for Register CM_IDLEST_PLL_IVA2

PRCM Basic Programming Model • DPLL Clock Control Registers: [0]
PRCM Register Manual • IVA2_CM Registers: [1]

Table 4-96. CM_AUTOIDLE_PLL_IVA2

Address Offset	0x0000 0034		
Physical Address	0x4800 4034	**Instance**	IVA2_CM
Description	This register provides automatic control over the IVA2 DPLL activity.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																														AUTO_IVA2_DPLL	

Bits	Field Name	Description	Type	Reset
31:3	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x00000000
2:0	AUTO_IVA2_DPLL	IVA2 DPLL automatic control; Other enums: Reserved 0x0: Auto control disabled 0x1: IVA2 DPLL is automatically put in low power stop mode when the IVA2 clock is not required anymore. It is also restarted automatically.	RW	0x0

Table 4-97. Register Call Summary for Register CM_AUTOIDLE_PLL_IVA2

PRCM Clock Manager Functional Description • DPLL Modes: [0] • DPLL Source-Clock Controls: [1]
PRCM Basic Programming Model • DPLL Clock Control Registers: [2]
PRCM Register Manual • IVA2_CM Registers: [3]

Table 4-98. CM_CLKSEL1_PLL_IVA2

Address Offset	0x0000 0040		
Physical Address	0x4800 4040	**Instance**	IVA2_CM
Description	This register provides controls over the IVA2 DPLL.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED										IVA2_CLK_SRC						IVA2_DPLL_MULT								RESERVED							
																									IVA2_DPLL_DIV						

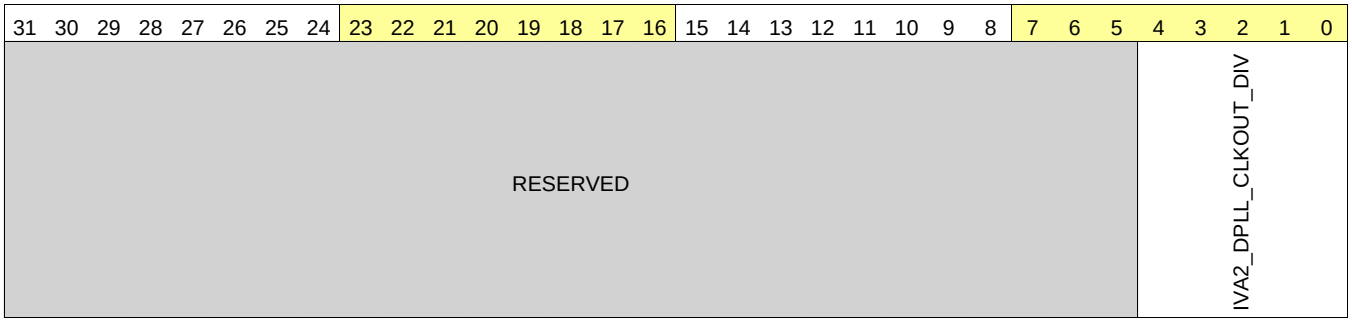
Bits	Field Name	Description	Type	Reset
31:22	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x000
21:19	IVA2_CLK_SRC	Selects the IVA2 DPLL bypass source clock; Other enums: Reserved 0x1: DPLL2_FCLK is CORE_CLK divided by 1 0x2: DPLL2_FCLK is CORE.CLK divided by 2 0x4: DPLL2_FCLK is CORE.CLK divided by 4	RW	0x1
18:8	IVA2_DPLL_MULT	IVA2 DPLL multiplier factor (0 to 2047)	RW	0x000
7	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x0
6:0	IVA2_DPLL_DIV	IVA2 DPLL divider factor (0 to 127)	RW	0x00

Table 4-99. Register Call Summary for Register CM_CLKSEL1_PLL_IVA2

PRCM Clock Manager Functional Description • Processor Clock Configurations: [0] [1] • Interface and Peripheral Functional Clock Configurations: [2] [3]
PRCM Basic Programming Model • DPLL Clock Control Registers: [4]
PRCM Use Cases and Tips • DVFS Using SmartReflex :
PRCM Register Manual • IVA2_CM Registers: [9]

Table 4-100. CM_CLKSEL2_PLL_IVA2

Address Offset	0x0000 0044		
Physical Address	0x4800 4044	**Instance**	IVA2_CM
Description	This register provides controls over the IVA2 DPLL.		
Type	RW		



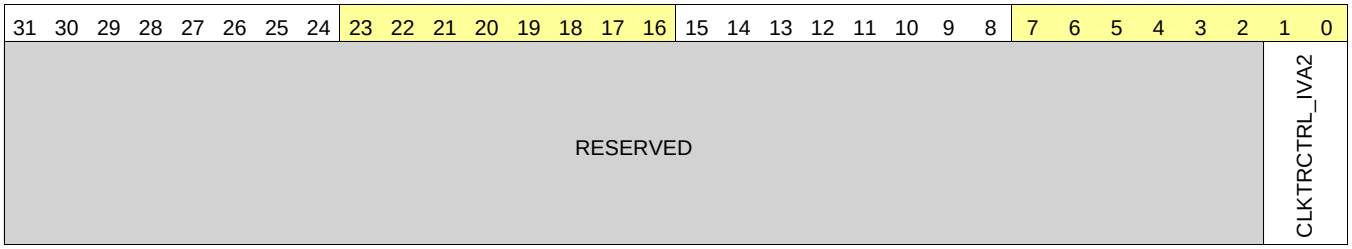
Bits	Field Name	Description	Type	Reset
31:5	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x0000000
4:0	IVA2_DPLL_CLKOUT_DIV	IVA2 DPLL output clock divider factor (1 up to 16); Other enums: Reserved 0x1: DPLL2 CLKOUTX2 divided by 1 0x2: DPLL2 CLKOUTX2 divided by 2 0x3: DPLL2 CLKOUTX2 divided by 3 0x4: DPLL2 CLKOUTX2 divided by 4 0x5: DPLL2 CLKOUTX2 divided by 5 0x6: DPLL2 CLKOUTX2 divided by 6 0x7: DPLL2 CLKOUTX2 divided by 7 0x8: DPLL2 CLKOUTX2 divided by 8 0x9: DPLL2 CLKOUTX2 divided by 9 0xA: DPLL2 CLKOUTX2 is divided by 10 0xB: DPLL2 CLKOUTX2 divided by 11 0xC: DPLL2 CLKOUTX2 divided by 12 0xD: DPLL2 CLKOUTX2 divided by 13 0xE: DPLL2 CLKOUTX2 divided by 14 0xF: DPLL2 CLKOUTX2 divided by 15 0x10: DPLL2 CLKOUTX2 divided by 16	RW	0x01

Table 4-101. Register Call Summary for Register CM_CLKSEL2_PLL_IVA2

PRCM Clock Manager Functional Description • Processor Clock Configurations: [0]
PRCM Basic Programming Model • DPLL Clock Control Registers: [1]
PRCM Register Manual • IVA2_CM Registers: [2]

Table 4-102. CM_CLKSTCTRL_IVA2

Address Offset	0x0000 0048		
Physical Address	0x4800 4048	**Instance**	IVA2_CM
Description	This register enables the domain power state transition. It controls the HW supervised domain power state transition between ACTIVE and INACTIVE states.		
Type	RW		



Bits	Field Name	Description	Type	Reset
31:2	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x00000000
1:0	CLKTRCTRL_IVA2	Controls the clock state transition of the IVA2 clock domain. 0x0: Automatic transition is disabled 0x1: Start a software supervised sleep transition on the domain 0x2: Start a software supervised wake-up transition on the domain 0x3: Automatic transition is enabled. Transition is supervised by the HardWare.	RW	0x0

Table 4-103. Register Call Summary for Register CM_CLKSTCTRL_IVA2

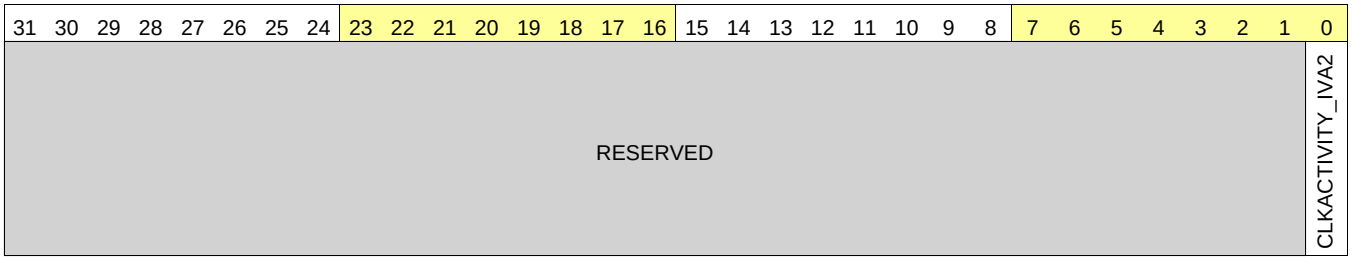
PRCM Idle and Wake-Up Management
- Device Wake-Up Events: [0] [1] [2]

PRCM Basic Programming Model
- Power-Domain Clock Control Registers: [3]

PRCM Register Manual
- IVA2_CM Registers: [4]

Table 4-104. CM_CLKSTST_IVA2

Address Offset	0x0000 004C		
Physical Address	0x4800 404C	**Instance**	IVA2_CM
Description	This register provides a status on the clock activity in the domain (IVA2 DPLL output clock).		
Type	R		



Bits	Field Name	Description	Type	Reset
31:1	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x00000000
0	CLKACTIVITY_IVA2	Clock activity status 0x0: No domain clock activity 0x1: Domain clock is active	R	0x0

Table 4-105. Register Call Summary for Register CM_CLKSTST_IVA2

PRCM Basic Programming Model • Power-Domain Clock Control Registers: [0]
PRCM Register Manual • IVA2_CM Registers: [1]

4.14.1.3 OCP_System_Reg_CM Registers

Table 4-106. OCP_System_Reg_CM Register Summary

Register Name	Type	Register Width (Bits)	Address Offset	Physical Address	Reset Type
CM_REVISION	R	32	0x0000 0000	0x4800 4800	C
CM_SYSCONFIG	RW	32	0x0000 0010	0x4800 4810	W

Table 4-107. CM_REVISION

Address Offset	0x0000 0000		
Physical Address	0x4800 4800	**Instance**	OCP_System_Reg_CM
Description	This register contains the IP revision code for the CM part of the PRCM		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																								REV							

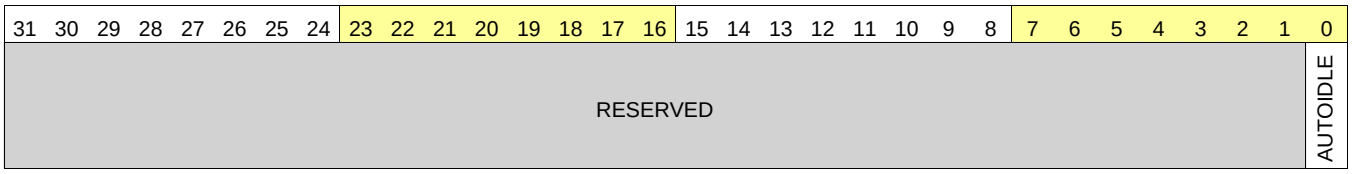
Bits	Field Name	Description	Type	Reset
31:8	RESERVED	Reads returns 0.	R	0x000000
7:0	REV	IP revision [7:4] Major revision [3:0] Minor revision Examples: 0x10 for 1.0, 0x21 for 2.1	R	0x10

Table 4-108. Register Call Summary for Register CM_REVISION

PRCM Basic Programming Model • Revision Information Registers: [0]
PRCM Register Manual • OCP_System_Reg_CM Registers: [1]

Table 4-109. CM_SYSCONFIG

Address Offset	0x0000 0010		
Physical Address	0x4800 4810	**Instance**	OCP_System_Reg_CM
Description	This register controls the various parameters of the interface clock		
Type	RW		



Bits	Field Name	Description	Type	Reset
31:1	RESERVED	Write 0s for future compatibility. Reads returns 0.	R	0x00000000
0	AUTOIDLE	Internal clock gating strategy (for the CM part of the PRCM) 0x0: Interface clock is free-running 0x1: Automatic clock gating strategy is enabled, based on the interface activity.	RW	0x1

Table 4-110. Register Call Summary for Register CM_SYSCONFIG

PRCM Basic Programming Model
- PRCM Configuration Registers: [0]

PRCM Register Manual
- OCP_System_Reg_CM Registers: [1]

4.14.1.4 MPU_CM Registers

Table 4-111. MPU_CM Register Summary

Register Name	Type	Register Width (Bits)	Address Offset	Physical Address	Reset Type
CM_CLKEN_PLL_MPU	RW	32	0x0000 0004	0x4800 4904	W
CM_IDLEST_MPU	R	32	0x0000 0020	0x4800 4920	C
CM_IDLEST_PLL_MPU	R	32	0x0000 0024	0x4800 4924	C
CM_AUTOIDLE_PLL_MPU	RW	32	0x0000 0034	0x4800 4934	W
CM_CLKSEL1_PLL_MPU	RW	32	0x0000 0040	0x4800 4940	W
CM_CLKSEL2_PLL_MPU	RW	32	0x0000 0044	0x4800 4944	W
CM_CLKSTCTRL_MPU	RW	32	0x0000 0048	0x4800 4948	W
CM_CLKSTST_MPU	R	32	0x0000 004C	0x4800 494C	C

Table 4-112. CM_CLKEN_PLL_MPU

Address Offset	0x0000 0004		
Physical Address	0x4800 4904	**Instance**	MPU_CM
Description	This register controls the DPLL1 modes.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																					EN_MPU_DPLL_LPMODE	RESERVED		MPU_DPLL_FREQSEL				EN_MPU_DPLL_DRIFTGUARD		EN_MPU_DPLL	

Bits	Field Name	Description	Type	Reset
31:11	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x000000
10	EN_MPU_DPLL_LPMODE	This bit allows to enable or disable the LP mode of the MPU DPLL. Writting this bit to switch the mode between LP or normal mode will take effect only when the DPLL will have transition into the bypass or stop state, followed by a lock or re-lock of the DPLL. 0x0: Disables the DPLL LP mode to re-enter the normal mode at the following lock or re-lock sequence. 0x1: Enables the DPLL LP mode to enter the LP mode at the following lock or re-lock sequence.	RW	0x0
9:8	RESERVED	Write 0s for future compatibility. Read returns 0.	RW	0x0
7:4	MPU_DPLL_FREQSEL	This bit field allows selecting the proper range of the DPLL1 internal frequency depending on the DPLL reference clock and the N divider. 0x3: 0.75 MHz—1.0 MHz 0x4: 1.0 MHz—1.25 MHz 0x5: 1.25 MHz—1.5 MHz 0x6: 1.5 MHz—1.75 MHz 0x7: 1.75 MHz—2.1 MHz 0xB: 7.5 MHz—10 MHz 0xC: 10 MHz—12.5 MHz 0xD: 12.5 MHz—15 MHz 0xE: 15 MHz—17.5 MHz 0xF: 17.5 MHz—21 MHz	RW	0x1
3	EN_MPU_DPLL_DRIFTGUARD	This bit allows to enable or disable the automatic recalibration feature of the MPU DPLL. The DPLL1 will automatically start a recalibration process upon assertion of the recal flag if this bit is set. 0x0: Disables the DPLL1 automatic recalibration mode 0x1: Enables the DPLL1 automatic recalibration mode	RW	0x0
2:0	EN_MPU_DPLL	DPLL1 control; Other enums: Reserved 0x5: Put the DPLL1 in low power bypass mode 0x7: Enables the DPLL1 in lock mode	RW	0x5

Table 4-113. Register Call Summary for Register CM_CLKEN_PLL_MPU

PRCM Clock Manager Functional Description
- DPLL Modes: [0]
- DPLL Low-Power Mode: [1]
- Latencies :
- Recalibration: [3]
- DPLL Source-Clock Controls: [4]

PRCM Basic Programming Model
- DPLL Clock Control Registers: [5] [6]

Table 4-113. Register Call Summary for Register CM_CLKEN_PLL_MPU (continued)

PRCM Register Manual • MPU_CM Registers: [7]

Table 4-114. CM_IDLEST_MPU

Address Offset	0x0000 0020		
Physical Address	0x4800 4920	**Instance**	MPU_CM
Description	Modules access availability monitoring. This register is read only and automatically updated.		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																															ST_MPU

Bits	Field Name	Description	Type	Reset
31:1	RESERVED	Read returns 0.	R	0x00000000
0	ST_MPU	MPU standby status. 0x0: MPU is active. 0x1: MPU is in standby mode.	R	0x1

Table 4-115. Register Call Summary for Register CM_IDLEST_MPU

PRCM Basic Programming Model • Power-Domain Clock Control Registers: [0]
PRCM Register Manual • MPU_CM Registers: [1]

Table 4-116. CM_IDLEST_PLL_MPU

Address Offset	0x0000 0024		
Physical Address	0x4800 4924	**Instance**	MPU_CM
Description	This register allows monitoring the master clock activity. This register is read only and automatically updated.		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																															ST_MPU_CLK

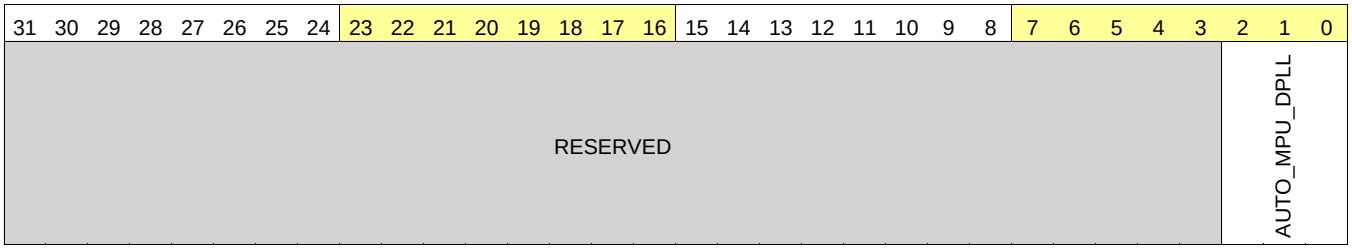
Bits	Field Name	Description	Type	Reset
31:1	RESERVED	Read returns 0.	R	0x00000000
0	ST_MPU_CLK	MPU_CLK activity 0x0: DPLL1 is bypassed 0x1: DPLL1 is locked	R	0x0

Table 4-117. Register Call Summary for Register CM_IDLEST_PLL_MPU

PRCM Basic Programming Model • DPLL Clock Control Registers: [0]
PRCM Register Manual • MPU_CM Registers: [1]

Table 4-118. CM_AUTOIDLE_PLL_MPU

Address Offset	0x0000 0034		
Physical Address	0x4800 4934	**Instance**	MPU_CM
Description	This register provides automatic control over the DPLL1 activity.		
Type	RW		



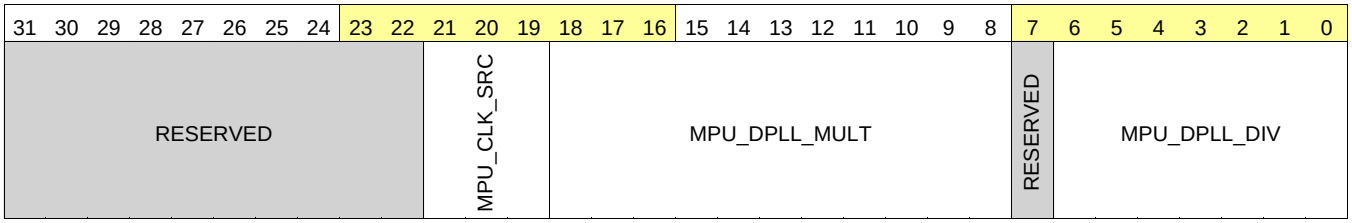
Bits	Field Name	Description	Type	Reset
31:3	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x00000000
2:0	AUTO_MPU_DPLL	DPLL1 automatic control; Other enums: Reserved 0x0: Auto control disabled 0x1: DPLL1 is automatically put in low power stop mode when the MPU clock is not required anymore. It is also restarted automatically.	RW	0x0

Table 4-119. Register Call Summary for Register CM_AUTOIDLE_PLL_MPU

PRCM Clock Manager Functional Description • DPLL Modes: [0] • DPLL Source-Clock Controls: [1]
PRCM Basic Programming Model • DPLL Clock Control Registers: [2]
PRCM Register Manual • MPU_CM Registers: [3]

Table 4-120. CM_CLKSEL1_PLL_MPU

Address Offset	0x0000 0040		
Physical Address	0x4800 4940	**Instance**	MPU_CM
Description	This register provides controls over the MPU DPLL.		
Type	RW		



Bits	Field Name	Description	Type	Reset
31:22	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x000
21:19	MPU_CLK_SRC	Selects the DPLL1 bypass source clock; Other enums: Reserved 0x1: DPLL1_FCLK is CORE_CLK divided by 1 0x2: DPLL1_FCLK is CORE_CLK divided by 2 0x4: DPLL1_FCLK is CORE_CLK divided by 4	RW	0x1
18:8	MPU_DPLL_MULT	DPLL1 multiplier factor (0 to 2047)	RW	0x000
7	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x0
6:0	MPU_DPLL_DIV	DPLL1 divider factor (0 to 127)	RW	0x00

Table 4-121. Register Call Summary for Register CM_CLKSEL1_PLL_MPU

PRCM Clock Manager Functional Description • Processor Clock Configurations: [0] [1] • Interface and Peripheral Functional Clock Configurations: [2] [3]
PRCM Basic Programming Model • DPLL Clock Control Registers: [4] [5]
PRCM Use Cases and Tips • DVFS Using SmartReflex :
PRCM Register Manual • MPU_CM Registers: [10]

Table 4-122. CM_CLKSEL2_PLL_MPU

Address Offset	0x0000 0044		
Physical Address	0x4800 4944	**Instance**	MPU_CM
Description	This register provides controls over the MPU DPLL.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																											MPU_DPLL_CLKOUT_DIV					

Bits	Field Name	Description	Type	Reset
31:5	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x0000000
4:0	MPU_DPLL_CLKOUT_DIV	DPLL1 output clock divider factor (1 up to 16); Other enums: Reserved 0x1: DPLL1 CLKOUTX2 divided by 1 0x2: DPLL1 CLKOUTX2 divided by 2 0x3: DPLL1 CLKOUTX2 divided by 3 0x4: DPLL1 CLKOUTX2 divided by 4 0x5: DPLL1 CLKOUTX2 divided by 5 0x6: DPLL1 CLKOUTX2 divided by 6 0x7: DPLL1 CLKOUTX2 divided by 7 0x8: DPLL1 CLKOUTX2 divided by 8 0x9: DPLL1 CLKOUTX2 divided by 9 0xA: DPLL1 CLKOUTX2 divided by 10 0xB: DPLL1 CLKOUTX2 divided by 11 0xC: DPLL1 CLKOUTX2 divided by 12 0xD: DPLL1 CLKOUTX2 divided by 13 0xE: DPLL1 CLKOUTX2 divided by 14 0xF: DPLL1 CLKOUTX2 divided by 15 0x10: DPLL1 CLKOUTX2 divided by 16	RW	0x01

Table 4-123. Register Call Summary for Register CM_CLKSEL2_PLL_MPU

PRCM Clock Manager Functional Description • Processor Clock Configurations: [0]
PRCM Basic Programming Model • DPLL Clock Control Registers: [1]
PRCM Register Manual • MPU_CM Registers: [2]

Table 4-124. CM_CLKSTCTRL_MPU

Address Offset	0x0000 0048		
Physical Address	0x4800 4948	**Instance**	MPU_CM
Description	This register enables the domain power state transition. It controls the HW supervised domain power state transition between ACTIVE and INACTIVE states.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																														CLKTRCTRL_MPU	

Bits	Field Name	Description	Type	Reset
31:2	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x00000000
1:0	CLKTRCTRL_MPU	Controls the clock state transition of the MPU clock domain. 0x0: Automatic transition is disabled 0x1: Reserved 0x2: Start a software supervised wake-up transition on the domain 0x3: Automatic transition is enabled. Transition is supervised by the HardWare.	RW	0x0

Table 4-125. Register Call Summary for Register CM_CLKSTCTRL_MPU

PRCM Basic Programming Model • Power-Domain Clock Control Registers: [0]
PRCM Register Manual • MPU_CM Registers: [1]

Table 4-126. CM_CLKSTST_MPU

Address Offset	0x0000 004C		
Physical Address	0x4800 494C	**Instance**	MPU_CM
Description	This register provides a status on the clock activity in the domain (MPU DPLL output clock).		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																															CLKACTIVITY_MPU

Bits	Field Name	Description	Type	Reset
31:1	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x00000000
0	CLKACTIVITY_MPU	Clock activity status 0x0: No domain clock activity 0x1: Domain clock is active	R	0x0

Table 4-127. Register Call Summary for Register CM_CLKSTST_MPU

PRCM Basic Programming Model • Power-Domain Clock Control Registers: [0]
PRCM Register Manual • MPU_CM Registers: [1]

4.14.1.5 CORE_CM Registers

Table 4-128. CORE_CM Register Summary

Register Name	Type	Register Width (Bits)	Address Offset	Physical Address	Reset Type
CM_FCLKEN1_CORE	RW	32	0x0000 0000	0x4800 4A00	W

Table 4-128. CORE_CM Register Summary (continued)

Register Name	Type	Register Width (Bits)	Address Offset	Physical Address	Reset Type
CM_FCLKEN3_CORE	RW	32	0x0000 0008	0x4800 4A08	W
CM_ICLKEN1_CORE	RW	32	0x0000 0010	0x4800 4A10	W
Reserved for non-GP devices.	RW	32	0x0000 0014	0x4800 4A14	W
CM_ICLKEN3_CORE	RW	32	0x0000 0018	0x4800 4A18	W
CM_IDLEST1_CORE	R	32	0x0000 0020	0x4800 4A20	C
Reserved for non-GP devices.	R	32	0x0000 0024	0x4800 4A24	C
CM_IDLEST3_CORE	R	32	0x0000 0028	0x4800 4A28	C
CM_AUTOIDLE1_CORE	RW	32	0x0000 0030	0x4800 4A30	W
Reserved for non-GP devices.	RW	32	0x0000 0034	0x4800 4A34	W
CM_AUTOIDLE3_CORE	RW	32	0x0000 0038	0x4800 4A38	W
CM_CLKSEL_CORE	RW	32	0x0000 0040	0x4800 4A40	W
CM_CLKSTCTRL_CORE	RW	32	0x0000 0048	0x4800 4A48	W
CM_CLKSTST_CORE	R	32	0x0000 004C	0x4800 4A4C	C

Table 4-129. CM_FCLKEN1_CORE

Address Offset	0x0000 0000		
Physical Address	0x4800 4A00	**Instance**	CORE_CM
Description	Controls the module functional clock activity.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED	EN_MMC3	RESERVED				EN_MMC2	EN_MMC1	RESERVED	EN_HDQ	EN_MCSPI4	EN_MCSPI3	EN_MCSPI2	EN_MCSPI1	EN_I2C3	EN_I2C2	EN_I2C1	EN_UART2	EN_UART1	EN_GPT11	EN_GPT10	EN_MCBSP5	EN_MCBSP1	RESERVED								RESERVED

Bits	Field Name	Description	Type	Reset
31	RESERVED	Write 0s for future compatibility. Read returns 0.	RW	0x0
30	EN_MMC3	MMC3 functional clock control. 0x0: MMC3 functional clock is disabled 0x1: MMC3 functional clock is enabled	RW	0x0
29:26	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x0
25	EN_MMC2	MMC2 functional clock control. 0x0: MMC2 functional clock is disabled 0x1: MMC2 functional clock is enabled	RW	0x0
24	EN_MMC1	MMC1 functional clock control. 0x0: MMC 1 functional clock is disabled 0x1: MMC 1 functional clock is enabled	RW	0x0
23	RESERVED	Write 0s for future compatibility. Read returns 0.	RW	0x0
22	EN_HDQ	HDQ-1 wire functional clock control. 0x0: HDQ functional clock is disabled 0x1: HDQ functional clock is enabled	RW	0x0
21	EN_MCSPI4	McSPI 4 functional clock control. 0x0: McSPI 4 functional clock is disabled 0x1: McSPI 4 functional clock is enabled	RW	0x0

Bits	Field Name	Description	Type	Reset
20	EN_MCSPI3	McSPI 3 functional clock control. 0x0: McSPI 3 functional clock is disabled 0x1: McSPI 3 functional clock is enabled	RW	0x0
19	EN_MCSPI2	McSPI 2 functional clock control. 0x0: McSPI 2 functional clock is disabled 0x1: McSPI 2 functional clock is enabled	RW	0x0
18	EN_MCSPI1	McSPI 1 functional clock control. 0x0: McSPI 1 functional clock is disabled 0x1: McSPI 1 functional clock is enabled	RW	0x0
17	EN_I2C3	I2C 3 functional clock control. 0x0: I2C 3 functional clock is disabled 0x1: I2C 3 functional clock is enabled	RW	0x0
16	EN_I2C2	I2C 2 functional clock control. 0x0: I2C 2 functional clock is disabled 0x1: I2C 2 functional clock is enabled	RW	0x0
15	EN_I2C1	I2C 1 functional clock control. 0x0: I2C 1 functional clock is disabled 0x1: I2C 1 functional clock is enabled	RW	0x0
14	EN_UART2	UART 2 functional clock control. 0x0: UART 2 functional clock is disabled 0x1: UART 2 functional clock is enabled	RW	0x0
13	EN_UART1	UART 1 functional clock control. 0x0: UART 1 functional clock is disabled 0x1: UART 1 functional clock is enabled	RW	0x0
12	EN_GPT11	GPTIMER 11 functional clock control. 0x0: GPTIMER 11 functional clock is disabled 0x1: GPTIMER 11 functional clock is enabled	RW	0x0
11	EN_GPT10	GPTIMER 10 functional clock control. 0x0: GPTIMER 10 functional clock is disabled 0x1: GPTIMER 10 functional clock is enabled	RW	0x0
10	EN_MCBSP5	McBSP 5 functional clock control. 0x0: McBSP 5 functional clock is disabled 0x1: McBSP 5 functional clock is enabled	RW	0x0
9	EN_MCBSP1	McBSP 1 functional clock control. 0x0: McBSP 1 functional clock is disabled 0x1: McBSP 1 functional clock is enabled	RW	0x0
8:1	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x00
0	RESERVED	Write 0s for future compatibility. Read returns 0.	RW	0x0

Table 4-130. Register Call Summary for Register CM_FCLKEN1_CORE

PRCM Clock Manager Functional Description • CORE Power Domain Clock Controls: [0] [1] [2] [3] [4] [5] [6] [7]
PRCM Basic Programming Model • Power-Domain Clock Control Registers: [8]
PRCM Register Manual • CORE_CM Registers: [9]

Table 4-131. CM_FCLKEN3_CORE

Address Offset	0x0000 0008		
Physical Address	0x4800 4A08	**Instance**	CORE_CM
Description	Controls the module functional clock activity.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																													EN_USBTLL	EN_TS	RESERVED

Bits	Field Name	Description	Type	Reset
31:3	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x00000000
2	EN_USBTLL	USB TLL functional clock control. 0x0: USB TLL functional clock is disabled 0x1: USB TLL functional clock is enabled	RW	0x0
1	EN_TS	Temperature Sensors functional clock control. 0x0: Temperature Sensors functional clock is disabled (for both BandGap) 0x1: Temperature Sensors functional clock is enabled (for both BandGap)	RW	0x0
0	RESERVED	Reserved for non-GP devices.	RW	0x0

Table 4-132. Register Call Summary for Register CM_FCLKEN3_CORE

PRCM Clock Manager Functional Description
- CM Source-Clock Controls: [0]
- CORE Power Domain Clock Controls: [1] [2]

PRCM Basic Programming Model
- Power-Domain Clock Control Registers: [3]

PRCM Register Manual
- CORE_CM Registers: [4]

Table 4-133. CM_ICLKEN1_CORE

Address Offset	0x0000 0010		
Physical Address	0x4800 4A10	**Instance**	CORE_CM
Description	Controls the modules interface clock activity.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED	EN_MMC3	EN_ICR	RESERVED			EN_MMC2	EN_MMC1	RESERVED	EN_HDQ	EN_MCSPI4	EN_MCSPI3	EN_MCSPI2	EN_MCSPI1	EN_I2C3	EN_I2C2	EN_I2C1	EN_UART2	EN_UART1	EN_GPT11	EN_GPT10	EN_MCBSP5	EN_MCBSP1	RESERVED	EN_MAILBOXES	EN_OMAPCTRL	RESERVED	EN_HSOTGUSB	RESERVED	RESERVED	EN_SDRC	RESERVED

Bits	Field Name	Description	Type	Reset
31	RESERVED	Read returns 0.	R	0x0
30	EN_MMC3	MMC SDIO 3 interface clock control. 0x0: MMC 3 interface clock is disabled 0x1: MMC 3 interface clock is enabled	RW	0x0
29	EN_ICR	ICR interface clock control. 0x0: ICR interface clock is disabled 0x1: ICR interface clock is enabled	RW	0x0
28:26	RESERVED	Reserved for non-GP devices.	RW	0x0
25	EN_MMC2	MMC SDIO 2 interface clock control. 0x0: MMC 2 interface clock is disabled 0x1: MMC 2 interface clock is enabled	RW	0x0
24	EN_MMC1	MMC SDIO 1 interface clock control. 0x0: MMC 1 interface clock is disabled 0x1: MMC 1 interface clock is enabled	RW	0x0
23	RESERVED	Write 0s for future compatibility. Read returns 0.	RW	0x0
22	EN_HDQ	HDQ-wire interface clock control. 0x0: HDQ interface clock is disabled 0x1: HDQ interface clock is enabled	RW	0x0
21	EN_MCSPI4	McSPI 4 interface clock control. 0x0: McSPI 4 interface clock is disabled 0x1: McSPI 4 interface clock is enabled	RW	0x0
20	EN_MCSPI3	McSPI 3 interface clock control. 0x0: McSPI 3 interface clock is disabled 0x1: McSPI 3 interface clock is enabled	RW	0x0
19	EN_MCSPI2	McSPI 2 interface clock control. 0x0: McSPI 2 interface clock is disabled 0x1: McSPI 2 interface clock is enabled	RW	0x0
18	EN_MCSPI1	McSPI 1 interface clock control. 0x0: McSPI 1 interface clock is disabled 0x1: McSPI 1 interface clock is enabled	RW	0x0
17	EN_I2C3	I2C 3 interface clock control. 0x0: I2C 3 interface clock is disabled 0x1: I2C 3 interface clock is enabled	RW	0x0
16	EN_I2C2	I2C 2 interface clock control. 0x0: I2C 2 interface clock is disabled 0x1: I2C 2 interface clock is enabled	RW	0x0
15	EN_I2C1	I2C 1 interface clock control. 0x0: I2C 1 interface clock is disabled 0x1: I2C 1 interface clock is enabled	RW	0x0
14	EN_UART2	UART 2 interface clock control. 0x0: UART 2 interface clock is disabled 0x1: UART 2 interface clock is enabled	RW	0x0
13	EN_UART1	UART 1 interface clock control. 0x0: UART 1 interface clock is disabled 0x1: UART 1 interface clock is enabled	RW	0x0
12	EN_GPT11	GPTIMER 11 interface clock control. 0x0: GPTIMER 11 interface clock is disabled 0x1: GPTIMER 11 interface clock is enabled	RW	0x0

Bits	Field Name	Description	Type	Reset
11	EN_GPT10	GPTIMER 10 interface clock control. 0x0: GPTIMER 10 interface clock is disabled 0x1: GPTIMER 10 interface clock is enabled	RW	0x0
10	EN_MCBSP5	McBSP 5 interface clock control. 0x0: McBSP 5 interface clock is disabled 0x1: McBSP 5 interface clock is enabled	RW	0x0
9	EN_MCBSP1	McBSP 1 interface clock control. 0x0: McBSP 1 interface clock is disabled 0x1: McBSP 1 interface clock is enabled	RW	0x0
8	RESERVED	Read returns 0.	R	0x0
7	EN_MAILBOXES	Mailboxes interface clock control 0x0: Maiboxes interface clock is disabled 0x1: Mailboxes interface clock is enabled	RW	0x0
6	EN_OMAPCTRL	System Control Module interface clock control 0x0: SCM interface clock is disabled 0x1: SCM interface clock is enabled	RW	0x1
5	RESERVED	Read returns 0.	R	0x0
4	EN_HSOTGUSB	HS OTG USB interface clock control. 0x0: HS OTG USB interface clock is disabled 0x1: HS OTG USB interface clock is enabled	RW	0x0
3	RESERVED	Read returns 0.	R	0x01
2	RESERVED	Read returns 0.	R	0x0
1	EN_SDRC	SDRC interface clock control. 0x0: SDRC interface clock is disabled 0x1: SDRC interface clock is enabled	RW	0x1
0	RESERVED	Read returns 0.	RW	0x0

Table 4-134. Register Call Summary for Register CM_ICLKEN1_CORE

PRCM Clock Manager Functional Description • CORE Power Domain Clock Controls: [0] [1]
PRCM Idle and Wake-Up Management • Sleep Dependencies:
PRCM Basic Programming Model • Power-Domain Clock Control Registers: [3]
PRCM Register Manual • CORE_CM Registers: [4]

Table 4-135. CM_ICLKEN3_CORE

Address Offset	0x0000 0018		
Physical Address	0x4800 4A18	**Instance**	CORE_CM
Description	Controls the module interface clock activity.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																												RESERVED	EN_USBTLL	RESERVED	

Bits	Field Name	Description	Type	Reset
31:4	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x0000000
3	RESERVED	Write 0s for future compatibility. Read returns 0.	RW	0x0
2	EN_USBTLL	USB TLL interface clock control. 0x0: USB TLL interface clock is disabled 0x1: USB TLL interface clock is enabled	RW	0x0
1:0	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x0

Table 4-136. Register Call Summary for Register CM_ICLKEN3_CORE

PRCM Basic Programming Model
- Power-Domain Clock Control Registers: [0]

PRCM Register Manual
- CORE_CM Registers: [1]

Table 4-137. CM_IDLEST1_CORE

Address Offset	0x0000 0020		
Physical Address	0x4800 4A20	**Instance**	CORE_CM
Description	CORE modules access availability monitoring. This register is read only and automatically updated.		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED	ST_MMC3	ST_ICR	RESERVED			ST_MMC2	ST_MMC1	RESERVED	ST_HDQ	ST_MCSPI4	ST_MCSPI3	ST_MCSPI2	ST_MCSPI1	ST_I2C3	ST_I2C2	ST_I2C1	ST_UART2	ST_UART1	ST_GPT11	ST_GPT10	ST_MCBSP5	ST_MCBSP1	RESERVED	ST_MAILBOXES	ST_OMAPCTRL	ST_HSOTGUSB_IDLE	ST_HSOTGUSB_STDBY	RESERVED	ST_SDMA	ST_SDRC	RESERVED

Bits	Field Name	Description	Type	Reset
31	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x1
30	ST_MMC3	MMC 3 idle status. 0x0: MMC 3 can be accessed. 0x1: MMC 3 cannot be accessed. Any access may return an error.	R	0x1
29	ST_ICR	ICR idle status. 0x0: ICR can be accessed. 0x1: ICR cannot be accessed. Any access may return an error.	R	0x1
28:26	RESERVED	Reserved for non-GP devices.	R	0x7
25	ST_MMC2	MMC 2 idle status. 0x0: MMC 2 can be accessed. 0x1: MMC 2 cannot be accessed. Any access may return an error.	R	0x1
24	ST_MMC1	MMC SDIO 1 idle status. 0x0: MMC 1 can be accessed. 0x1: MMC 1 cannot be accessed. Any access may return an error.	R	0x1
23	RESERVED	Read returns 1.	R	0x1

Bits	Field Name	Description	Type	Reset
22	ST_HDQ	HDQ-1 wire idle status. 0x0: HDQ can be accessed. 0x1: HDQ cannot be accessed. Any access may return an error.	R	0x1
21	ST_MCSPI4	McSPI 4 idle status. 0x0: McSPI 4 can be accessed. 0x1: McSPI 4 cannot be accessed. Any access may return an error.	R	0x1
20	ST_MCSPI3	McSPI 3 idle status. 0x0: McSPI 3 can be accessed. 0x1: McSPI 3 cannot be accessed. Any access may return an error.	R	0x1
19	ST_MCSPI2	McSPI 2 idle status. 0x0: McSPI 2 can be accessed. 0x1: McSPI 2 cannot be accessed. Any access may return an error.	R	0x1
18	ST_MCSPI1	McSPI 1 idle status. 0x0: McSPI 1 can be accessed. 0x1: McSPI 1 cannot be accessed. Any access may return an error.	R	0x1
17	ST_I2C3	I2C 3 idle status. 0x0: I2C 3 can be accessed. 0x1: I2C 3 cannot be accessed. Any access may return an error.	R	0x1
16	ST_I2C2	I2C 2 idle status. 0x0: I2C 2 can be accessed. 0x1: I2C 2 cannot be accessed. Any access may return an error.	R	0x1
15	ST_I2C1	I2C 1 idle status. 0x0: I2C 1 can be accessed. 0x1: I2C 1 cannot be accessed. Any access may return an error.	R	0x1
14	ST_UART2	UART 2 idle status. 0x0: UART 2 can be accessed. 0x1: UART 2 cannot be accessed. Any access may return an error.	R	0x1
13	ST_UART1	UART 1 idle status. 0x0: UART 1 can be accessed. 0x1: UART 1 cannot be accessed. Any access may return an error.	R	0x1
12	ST_GPT11	GPTIMER 11 idle status. 0x0: GPTIMER 11 can be accessed. 0x1: GPTIMER 11 cannot be accessed. Any access may return an error.	R	0x1
11	ST_GPT10	GPTIMER 10 idle status. 0x0: GPTIMER 10 can be accessed. 0x1: GPTIMER 10 cannot be accessed. Any access may return an error.	R	0x1
10	ST_MCBSP5	McBSP 5 idle status. 0x0: McBSP 5 can be accessed. 0x1: McBSP 5 cannot be accessed. Any access may return an error.	R	0x1

Bits	Field Name	Description	Type	Reset
9	ST_MCBSP1	McBSP 1 idle status. 0x0: McBSP 1 can be accessed. 0x1: McBSP 1 cannot be accessed. Any access may return an error.	R	0x1
8	RESERVED	Read undefined.	R	0x1
7	ST_MAILBOXES	Mailboxes idle status 0x0: Mailboxes can be accessed. 0x1: Mailboxes cannot be accessed. Any access may return an error.	R	0x1
6	ST_OMAPCTRL	System Control Module idle status 0x0: SCM can be accessed. 0x1: SCM cannot be accessed. Any access may return an error.	R	0x1
5	ST_HSOTGUSB_IDLE	HS OTG USB idle status. 0x0: HS OTG USB can be accessed. 0x1: HS OTG USB cannot be accessed. Any access may return an error.	R	0x1
4	ST_HSOTGUSB_STDBY	HS OTG USB standby status. 0x0: HS OTG USB is active. 0x1: HS OTG USB is in standby mode.	R	0x1
3	RESERVED	Read returns 1.	R	0x1
2	ST_SDMA	System DMA standby status. 0x0: System DMA is active. 0x1: System DMA is in standby mode.	R	0x1
1	ST_SDRC	SDRC idle status. 0x0: SDRC can be accessed. 0x1: SDRC cannot be accessed. Any access may return an error.	R	0x1
0	RESERVED	Read returns 1.	R	0x1

Table 4-138. Register Call Summary for Register CM_IDLEST1_CORE

PRCM Basic Programming Model • Power-Domain Clock Control Registers: [0]
PRCM Register Manual • CORE_CM Registers: [1]

Table 4-139. CM_IDLEST3_CORE

Address Offset	0x0000 0028		
Physical Address	0x4800 4A28	**Instance**	CORE_CM
Description	CORE modules access availability monitoring. This register is read only and automatically updated.		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																												RESERVED	ST_USBTLL	RESERVED	RESERVED

Bits	Field Name	Description	Type	Reset
31:4	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x0000000
3	RESERVED	Read returns 1.	R	0x1
2	ST_USBTLL	USB TLL idle status. 0x0: USB TLL can be accessed. 0x1: USB TLL cannot be accessed. Any access may return an error.	R	0x1
1	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x0
0	RESERVED	Reserved for non-GP devices.	R	0x1

Table 4-140. Register Call Summary for Register CM_IDLEST3_CORE

PRCM Basic Programming Model
- Power-Domain Clock Control Registers: [0]

PRCM Register Manual
- CORE_CM Registers: [1]

Table 4-141. CM_AUTOIDLE1_CORE

Address Offset	0x0000 0030		
Physical Address	0x4800 4A30	**Instance**	CORE_CM
Description	This register controls the automatic control of the CORE modules interface clock activity.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED	AUTO_MMC3	AUTO_ICR	RESERVED			AUTO_MMC2	AUTO_MMC1	RESERVED	AUTO_HDQ	AUTO_MCSPI4	AUTO_MCSPI3	AUTO_MCSPI2	AUTO_MCSPI1	AUTO_I2C3	AUTO_I2C2	AUTO_I2C1	AUTO_UART2	AUTO_UART1	AUTO_GPT11	AUTO_GPT10	AUTO_MCBSP5	AUTO_MCBSP1	RESERVED	AUTO_MAILBOXES	AUTO_OMAPCTRL	RESERVED	AUTO_HSOTGUSB	RESERVED	RESERVED	RESERVED

Bits	Field Name	Description	Type	Reset
31	RESERVED	Write 0s for future compatibility. Read returns 0.	RW	0x0
30	AUTO_MMC3	MMC SDIO 3 auto clock control. 0x0: MMC 3 interface clock is unrelated to the domain state transition. 0x1: MMC 3 interface clock is automatically enabled or disabled along with the domain state transition.	RW	0x0
29	AUTO_ICR	ICR auto clock control. 0x0: ICR interface clock is unrelated to the domain state transition. 0x1: ICR interface clock is automatically enabled or disabled along with the domain state transition.	RW	0x0
28:26	RESERVED	Reserved for non-GP devices.	RW	0x0
25	AUTO_MMC2	MMC SDIO 2 auto clock control. 0x0: MMC 2 interface clock is unrelated to the domain state transition. 0x1: MMC 2 interface clock is automatically enabled or disabled along with the domain state transition.	RW	0x0
24	AUTO_MMC1	MMC SDIO 1 auto clock control. 0x0: MMC 1 interface clock is unrelated to the domain state transition. 0x1: MMC 1 interface clock is automatically enabled or disabled along with the domain state transition.	RW	0x0

Bits	Field Name	Description	Type	Reset
23	RESERVED	Read returns 0.	RW	0x0
22	AUTO_HDQ	HDQ-1 wire auto clock control. 0x0: HDQ interface clock is unrelated to the domain state transition. 0x1: HDQ interface clock is automatically enabled or disabled along with the domain state transition.	RW	0x0
21	AUTO_MCSPI4	McSPI 4 auto clock control. 0x0: McSPI 4 interface clock is unrelated to the domain state transition. 0x1: McSPI 4 interface clock is automatically enabled or disabled along with the domain state transition.	RW	0x0
20	AUTO_MCSPI3	McSPI 3 auto clock control. 0x0: McSPI 3 interface clock is unrelated to the domain state transition. 0x1: McSPI 3 interface clock is automatically enabled or disabled along with the domain state transition.	RW	0x0
19	AUTO_MCSPI2	McSPI 2 auto clock control. 0x0: McSPI 2 interface clock is unrelated to the domain state transition. 0x1: McSPI 2 interface clock is automatically enabled or disabled along with the domain state transition.	RW	0x0
18	AUTO_MCSPI1	McSPI 1 auto clock control. 0x0: McSPI 1 interface clock is unrelated to the domain state transition. 0x1: McSPI 1 interface clock is automatically enabled or disabled along with the domain state transition.	RW	0x0
17	AUTO_I2C3	I2C 3 auto clock control. 0x0: I2C 3 interface clock is unrelated to the domain state transition. 0x1: I2C 3 interface clock is automatically enabled or disabled along with the domain state transition.	RW	0x0
16	AUTO_I2C2	I2C 2 auto clock control. 0x0: I2C 2 interface clock is unrelated to the domain state transition. 0x1: I2C 2 interface clock is automatically enabled or disabled along with the domain state transition.	RW	0x0
15	AUTO_I2C1	I2C 1 auto clock control. 0x0: I2C 1 interface clock is unrelated to the domain state transition. 0x1: I2C 1 interface clock is automatically enabled or disabled along with the domain state transition.	RW	0x0
14	AUTO_UART2	UART 2 auto clock control. 0x0: UART 2 interface clock is unrelated to the domain state transition. 0x1: UART 2 interface clock is automatically enabled or disabled along with the domain state transition.	RW	0x0
13	AUTO_UART1	UART 1 auto clock control. 0x0: UART 1 interface clock is unrelated to the domain state transition. 0x1: UART 1 interface clock is automatically enabled or disabled along with the domain state transition.	RW	0x0

Bits	Field Name	Description	Type	Reset
12	AUTO_GPT11	GPTIMER 11 auto clock control. 0x0: GPTIMER 11 interface clock is unrelated to the domain state transition. 0x1: GPTIMER 11 interface clock is automatically enabled or disabled along with the domain state transition.	RW	0x0
11	AUTO_GPT10	GPTIMER 10 auto clock control. 0x0: GPTIMER 10 interface clock is unrelated to the domain state transition. 0x1: GPTIMER 10 interface clock is automatically enabled or disabled along with the domain state transition.	RW	0x0
10	AUTO_MCBSP5	McBSP 5 auto clock control. 0x0: McBSP 5 interface clock is unrelated to the domain state transition. 0x1: McBSP 5 interface clock is automatically enabled or disabled along with the domain state transition.	RW	0x0
9	AUTO_MCBSP1	McBSP 1 auto clock control. 0x0: McBSP 1 interface clock is unrelated to the domain state transition. 0x1: McBSP 1 interface clock is automatically enabled or disabled along with the domain state transition.	RW	0x0
8	RESERVED	Read returns 0.	R	0x0
7	AUTO_MAILBOXES	Mailboxes auto clock control 0x0: Mailboxes interface clock is unrelated to the domain state transition. 0x1: Mailboxes interface clock is automatically enabled or disabled along with the domain state transition.	RW	0x0
6	AUTO_OMAPCTRL	System Control Module auto clock control 0x0: SCM interface clock is unrelated to the domain state transition. 0x1: SCM interface clock is automatically enabled or disabled along with the domain state transition.	RW	0x0
5	RESERVED	Read returns 0.	R	0x0
4	AUTO_HSOTGUSB	HS OTG USB auto clock control. 0x0: HS OTG USB interface clock is unrelated to the domain state transition. 0x1: HS OTG USB interface clock is automatically enabled or disabled along with the domain state transition.	RW	0x0
3	RESERVED	Write 0s for future compatibility. Read returns 0.	RW	0x1
2:1	RESERVED	Read returns 0.	R	0x0
0	RESERVED	Write 0s for future compatibility. Read returns 0.	RW	0x0

Table 4-142. Register Call Summary for Register CM_AUTOIDLE1_CORE

PRCM Clock Manager Functional Description • CORE Power Domain Clock Controls: [0] [1]
PRCM Basic Programming Model • Power-Domain Clock Control Registers: [2]
PRCM Register Manual • CORE_CM Registers: [3]

Table 4-143. CM_AUTOIDLE3_CORE

Address Offset	0x0000 0038		
Physical Address	0x4800 4A38	**Instance**	CORE_CM
Description	This register controls the automatic control of the CORE modules interface clock activity.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																													AUTO_USBTLL	RESERVED	

Bits	Field Name	Description	Type	Reset
31:3	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x00000000
2	AUTO_USBTLL	USB TLL auto clock control. 0x0: USB TLL interface clock is unrelated to the domain state transition. 0x1: USB TLL interface clock is automatically enabled or disabled along with the domain state transition.	RW	0x0
1:0	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x0

Table 4-144. Register Call Summary for Register CM_AUTOIDLE3_CORE

PRCM Basic Programming Model
- Power-Domain Clock Control Registers: [0]

PRCM Register Manual
- CORE_CM Registers: [1]

Table 4-145. CM_CLKSEL_CORE

Address Offset	0x0000 0040		
Physical Address	0x4800 4A40	**Instance**	CORE_CM
Description	CORE modules clock selection.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																				RESERVED				CLKSEL_GPT11	CLKSEL_GPT10	RESERVED				CLKSEL_L4		CLKSEL_L3	

Bits	Field Name	Description	Type	Reset
31:12	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x00000
11:8	RESERVED	Write the reset value, read returns reset value.	RW	0x1
7	CLKSEL_GPT11	Selects GPTIMER 11 source clock 0x0: source is CM_32K_CLK 0x1: source is CM_SYS_CLK	RW	0x0
6	CLKSEL_GPT10	Selects GPTIMER 10 source clock 0x0: source is CM_32K_CLK 0x1: source is CM_SYS_CLK	RW	0x0

Bits	Field Name	Description	Type	Reset
5:4	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x0
3:2	CLKSEL_L4	Selects Peripherals interconnect clock (L4_CLK); Other enums: Reserved 0x1: L4_CLK is L3_CLK divided by 1 (boot mode only) 0x2: L4_CLK is L3_CLK divided by 2	RW	0x1
1:0	CLKSEL_L3	Selects L3 interconnect clock (L3_CLK); Other enums: Reserved 0x1: L3_CLK is CORE_CLK divided by 1 0x2: L3_CLK is CORE_CLK divided by 2	RW	0x1

Table 4-146. Register Call Summary for Register CM_CLKSEL_CORE

PRCM Clock Manager Functional Description
- PRM Source-Clock Controls: [0] [1] [2] [3]
- Interface and Peripheral Functional Clock Configurations: [4] [5]

PRCM Basic Programming Model
- Power-Domain Clock Control Registers: [6]
- Clock Control: [7]

PRCM Register Manual
- CORE_CM Registers: [8]

Table 4-147. CM_CLKSTCTRL_CORE

Address Offset	0x0000 0048		
Physical Address	0x4800 4A48	**Instance**	CORE_CM
Description	This register enables the domain power state transition. It controls the HW supervised domain power state transition between ACTIVE and INACTIVE states.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																												CLKTRCTRL_L4		CLKTRCTRL_L3	

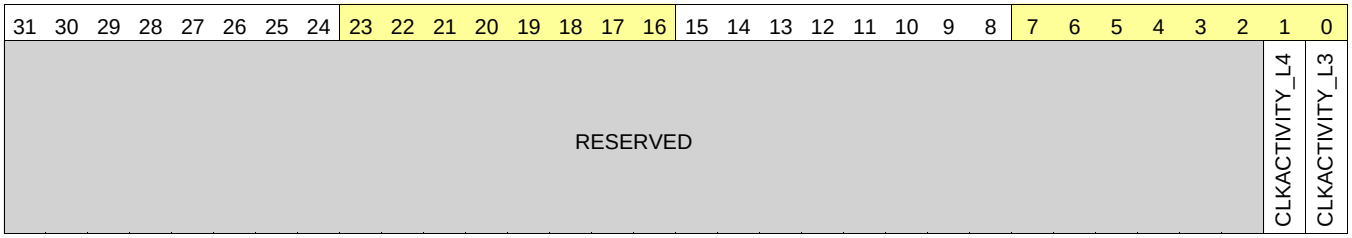
Bits	Field Name	Description	Type	Reset
31:4	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x0000000
3:2	CLKTRCTRL_L4	Controls the clock state transition of the L4 clock domain. 0x0: Automatic transition is disabled 0x1: Reserved 0x2: Reserved 0x3: Automatic transition is enabled. Transition is supervised by the HardWare.	RW	0x0
1:0	CLKTRCTRL_L3	Controls the clock state transition of the L3 clock domain. 0x0: Automatic transition is disabled 0x1: Reserved 0x2: Reserved 0x3: Automatic transition is enabled. Transition is supervised by the HardWare.	RW	0x0

Table 4-148. Register Call Summary for Register CM_CLKSTCTRL_CORE

PRCM Idle and Wake-Up Management • Sleep Dependencies:
PRCM Basic Programming Model • Power-Domain Clock Control Registers: [1]
PRCM Register Manual • CORE_CM Registers: [2]

Table 4-149. CM_CLKSTST_CORE

Address Offset	0x0000 004C		
Physical Address	0x4800 4A4C	**Instance**	CORE_CM
Description	This register provides a status on the interface clock activity in the domain.		
Type	R		



Bits	Field Name	Description	Type	Reset
31:2	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x00000000
1	CLKACTIVITY_L4	L4_ICLK interface clock activity status 0x0: No domain interface clock activity 0x1: Domain interface clock is active	R	0x0
0	CLKACTIVITY_L3	L3_ICLK interface clock activity status 0x0: No domain interface clock activity 0x1: Domain interface clock is active	R	0x0

Table 4-150. Register Call Summary for Register CM_CLKSTST_CORE

PRCM Idle and Wake-Up Management • Sleep Dependencies:
PRCM Basic Programming Model • Power-Domain Clock Control Registers: [1]
PRCM Register Manual • CORE_CM Registers: [2]

4.14.1.6 SGX_CM Registers

Table 4-151. SGX_CM Register Summary

Register Name	Type	Register Width (Bits)	Address Offset	Physical Address	Reset Type
CM_FCLKEN_SGX	RW	32	0x0000 0000	0x4800 4B00	W
CM_ICLKEN_SGX	RW	32	0x0000 0010	0x4800 4B10	W
CM_IDLEST_SGX	R	32	0x0000 0020	0x4800 4B20	C
CM_CLKSEL_SGX	RW	32	0x0000 0040	0x4800 4B40	W
CM_SLEEPDEP_SGX	RW	32	0x0000 0044	0x4800 4B44	W
CM_CLKSTCTRL_SGX	RW	32	0x0000 0048	0x4800 4B48	W

Table 4-151. SGX_CM Register Summary (continued)

Register Name	Type	Register Width (Bits)	Address Offset	Physical Address	Reset Type
CM_CLKSTST_SGX	R	32	0x0000 004C	0x4800 4B4C	C

Table 4-152. CM_FCLKEN_SGX

Address Offset	0x0000 0000		
Physical Address	0x4800 4B00	**Instance**	SGX_CM
Description	Controls the Graphic engine functional clock activity.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																														EN_SGX	RESERVED

Bits	Field Name	Description	Type	Reset
31:2	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x00000000
1	EN_SGX	SGX functional clock enable 0x0: SGX_FCLK is disabled 0x1: SGX_FCLK is enabled	RW	0x0
0	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x0

Table 4-153. Register Call Summary for Register CM_FCLKEN_SGX

PRCM Clock Manager Functional Description • SGX Power Domain Clock Controls: [0]
PRCM Basic Programming Model • Power-Domain Clock Control Registers: [1]
PRCM Register Manual • SGX_CM Registers: [2]

Table 4-154. CM_ICLKEN_SGX

Address Offset	0x0000 0010		
Physical Address	0x4800 4B10	**Instance**	SGX_CM
Description	Controls the Graphic engine interface clock activity.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																															EN_SGX

Bits	Field Name	Description	Type	Reset
31:1	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x00000000
0	EN_SGX	SGX interface clock control 0x0: SGX_L3_ICLK is disabled 0x1: SGX_L3_ICLK is enabled	RW	0x0

Table 4-155. Register Call Summary for Register CM_ICLKEN_SGX

PRCM Clock Manager Functional Description • SGX Power Domain Clock Controls: [0]
PRCM Basic Programming Model • Power-Domain Clock Control Registers: [1]
PRCM Register Manual • SGX_CM Registers: [2]

Table 4-156. CM_IDLEST_SGX

Address Offset	0x0000 0020		
Physical Address	0x4800 4B20	**Instance**	SGX_CM
Description	SGX standby status. This register is read only and automatically updated.		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																															ST_SGX

Bits	Field Name	Description	Type	Reset
31:1	RESERVED	Read returns 0	R	0x00000000
0	ST_SGX	SGX standby status. 0x0: SGX subsystem is active. 0x1: SGX subsystem is in standby mode.	R	0x1

Table 4-157. Register Call Summary for Register CM_IDLEST_SGX

PRCM Basic Programming Model • Power-Domain Clock Control Registers: [0]
PRCM Register Manual • SGX_CM Registers: [1]

Table 4-158. CM_CLKSEL_SGX

Address Offset	0x0000 0040		
Physical Address	0x4800 4B40	**Instance**	SGX_CM
Description	SGX clock selection.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																														CLKSEL_SGX	

Bits	Field Name	Description	Type	Reset
31:3	RESERVED	Write 0s for future compatibility. Read return 0.	R	0x00000000
2:0	CLKSEL_SGX	Selects SGX functional clock; Other enums: Reserved 0x0: SGX_FCLK is CORE_CLK divided by 3 0x1: SGX_FCLK is CORE_CLK divided by 4 0x2: SGX_FCLK is CORE_CLK divided by 6 0x3: SGX_FCLK clock is CM_96M_FCLK clock	RW	0x0

Table 4-159. Register Call Summary for Register CM_CLKSEL_SGX

PRCM Clock Manager Functional Description • Interface and Peripheral Functional Clock Configurations: [0]
PRCM Basic Programming Model • Power-Domain Clock Control Registers: [1] [2] [3]
PRCM Register Manual • SGX_CM Registers: [4]

Table 4-160. CM_SLEEPDEP_SGX

Address Offset	0x0000 0044		
Physical Address	0x4800 4B44	**Instance**	SGX_CM
Description	This register allows enabling or disabling the sleep transition dependency of SGX domain with respect to other domain.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																														EN_MPU	RESERVED

Bits	Field Name	Description	Type	Reset
31:2	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x00000000
1	EN_MPU	MPU domain dependency 0x0: SGX domain sleep dependency with MPU domain is disabled. 0x1: SGX domain sleep dependency with MPU domain is enabled.	RW	0x0
0	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x0

Table 4-161. Register Call Summary for Register CM_SLEEPDEP_SGX

PRCM Basic Programming Model • Power-Domain Clock Control Registers: [0]
PRCM Register Manual • SGX_CM Registers: [1]

Table 4-162. CM_CLKSTCTRL_SGX

Address Offset	0x0000 0048		
Physical Address	0x4800 4B48	**Instance**	SGX_CM
Description	This register enables the domain power state transition. It controls the HW supervised domain power state transition between ACTIVE and INACTIVE states.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																														CLKTRCTRL_SGX	

Bits	Field Name	Description	Type	Reset
31:2	RESERVED	Write 0s for future compatibility. Reads returns 0.	R	0x00000000
1:0	CLKTRCTRL_SGX	Controls the clock state transition of the SGX clock domain. 0x0: Automatic transition is disabled 0x1: Start a software supervised sleep transition on the domain 0x2: Start a software supervised wake-up transition on the domain 0x3: Automatic transition is enabled. Transition is supervised by the hardware.	RW	0x0

Table 4-163. Register Call Summary for Register CM_CLKSTCTRL_SGX

PRCM Idle and Wake-Up Management • Device Wake-Up Events: [0] [1] [2]
PRCM Basic Programming Model • Power-Domain Clock Control Registers: [3]
PRCM Register Manual • SGX_CM Registers: [4]

Table 4-164. CM_CLKSTST_SGX

Address Offset	0x0000 004C		
Physical Address	0x4800 4B4C	**Instance**	SGX_CM
Description	This register provides a status on the interface clock activity in the domain.		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																															CLKACTIVITY_SGX

Bits	Field Name	Description	Type	Reset
31:1	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x00000000
0	CLKACTIVITY_SGX	Interface clock activity status 0x0: No domain interface clock activity 0x1: Domain interface clock is active	R	0x0

Table 4-165. Register Call Summary for Register CM_CLKSTST_SGX

PRCM Basic Programming Model • Power-Domain Clock Control Registers: [0]
PRCM Register Manual • SGX_CM Registers: [1]

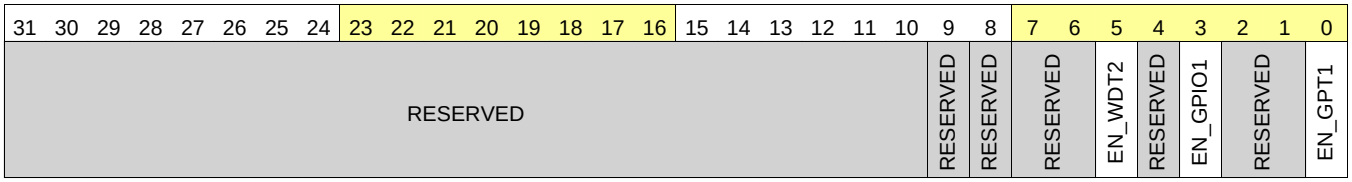
4.14.1.7 WKUP_CM Registers

Table 4-166. WKUP_CM Register Summary

Register Name	Type	Register Width (Bits)	Address Offset	Physical Address	Reset Type
CM_FCLKEN_WKUP	RW	32	0x0000 0000	0x4800 4C00	W
CM_ICLKEN_WKUP	RW	32	0x0000 0010	0x4800 4C10	W
CM_IDLEST_WKUP	R	32	0x0000 0020	0x4800 4C20	C
CM_AUTOIDLE_WKUP	RW	32	0x0000 0030	0x4800 4C30	W
CM_CLKSEL_WKUP	RW	32	0x0000 0040	0x4800 4C40	W

Table 4-167. CM_FCLKEN_WKUP

Address Offset	0x0000 0000		
Physical Address	0x4800 4C00	**Instance**	WKUP_CM
Description	Controls the modules functional clock activity.		
Type	RW		



Bits	Field Name	Description	Type	Reset
31:10	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x000000
9	RESERVED	Reserved for non-GP devices.	RW	0x0
8	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x0
7:6	RESERVED	This information is not available in the public domain.	RW	0x0
5	EN_WDT2	WDTIMER 2 functional clock control. 0x0: WDTIMER 2 functional clock is disabled 0x1: WDTIMER 2 functional clock is enabled	RW	0x0
4	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x0
3	EN_GPIO1	GPIO 1 clock control 0x0: GPIO 1 functional clock is disabled 0x1: GPIO 1 functional clock is enabled	RW	0x0
2:1	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x0
0	EN_GPT1	GPTIMER 1 clock control 0x0: GPTIMER 1 functional clock is disabled 0x1: GPTIMER 1 functional clock is enabled	RW	0x0

Table 4-168. Register Call Summary for Register CM_FCLKEN_WKUP

PRCM Clock Manager Functional Description
- WKUP Power Domain Clock Controls: [0] [1] [2]
- SMARTREFLEX Power Domain Clock Controls :

PRCM Basic Programming Model
- Power-Domain Clock Control Registers: [5]
- SmartReflex Module Initialization Basic Programming Model :

PRCM Use Cases and Tips
- DVFS Using SmartReflex :

PRCM Register Manual
- WKUP_CM Registers: [10]

Table 4-169. CM_ICLKEN_WKUP

Address Offset	0x0000 0010		
Physical Address	0x4800 4C10	**Instance**	WKUP_CM
Description	Controls the modules interface clock activity.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																						RESERVED		RESERVED		EN_WDT2	RESERVED	EN_GPIO1	EN_32KSYNC	RESERVED	EN_GPT1

Bits	Field Name	Description	Type	Reset
31:10	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x000000
9	RESERVED	Reserved for non-GP devices	RW	0x0
8:6	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x0
5	EN_WDT2	WDTIMER 2 interface clock 0x0: WDTIMER 2 interface clock is disabled 0x1: WDTIMER 2 interface clock is enabled	RW	0x0
4	RESERVED	Reserved for non-GP devices	RW	0x0
3	EN_GPIO1	GPIO 1 interface clock control 0x0: GPIO 1 interface clock is disabled 0x1: GPIO 1 interface clock is enabled	RW	0x0
2	EN_32KSYNC	32 kHz Sync Timer interface clock control 0x0: 32k Sync Timer interface clock is disabled 0x1: 32k Sync Timer interface clock is enabled	RW	0x0
1	RESERVED	Reserved for non-GP devices	RW	0x0
0	EN_GPT1	GPTIMER 1 interface clock control 0x0: GPTIMER 1 interface clock is disabled 0x1: GPTIMER 1 interface clock is enabled	RW	0x0

Table 4-170. Register Call Summary for Register CM_ICLKEN_WKUP

PRCM Clock Manager Functional Description • WKUP Power Domain Clock Controls: [0]
PRCM Basic Programming Model • Power-Domain Clock Control Registers: [1]
PRCM Register Manual • WKUP_CM Registers: [2]

Table 4-171. CM_IDLEST_WKUP

Address Offset	0x0000 0020		
Physical Address	0x4800 4C20	**Instance**	WKUP_CM
Description	WAKEUP domain modules access monitoring.This register is read only and automatically updated.		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																						RESERVED	RESERVED	RESERVED		ST_WDT2	RESERVED	ST_GPIO1	ST_32KSYNC	RESERVED	ST_GPT1

Bits	Field Name	Description	Type	Reset
31:10	RESERVED	Read returns 0.	R	0x000000
9	RESERVED	Reserved for non-GP devices	R	0x1
8	RESERVED	Read returns 0.	R	0x0
7:6	RESERVED	This information is not available in the public domain.	R	0x3
5	ST_WDT2	WDTIMER 2 idle status 0x0: WDTIMER 2 can be accessed. 0x1: WDTIMER 2 cannot be accessed. Any access may return an error.	R	0x1
4	RESERVED	Reserved for non-GP devices	R	0x1
3	ST_GPIO1	GPIO 1 idle status 0x0: GPIO 1 can be accessed. 0x1: GPIO 1 cannot be accessed. Any access may return an error.	R	0x1
2	ST_32KSYNC	32 kHz Sync Timer idle status 0x0: 32k Sync Timer can be accessed. 0x1: 32k Sync Timer cannot be accessed. Any access may return an error.	R	0x1
1	RESERVED	Reserved for non-GP devices	R	0x1
0	ST_GPT1	GPTIMER 1 idle status 0x0: GPTIMER 1 can be accessed 0x1: GPTIMER 1 cannot be accessed. Any access may return an error.	R	0x1

Table 4-172. Register Call Summary for Register CM_IDLEST_WKUP

PRCM Basic Programming Model
- Power-Domain Clock Control Registers: [0]

PRCM Register Manual
- WKUP_CM Registers: [1]

Table 4-173. CM_AUTOIDLE_WKUP

Address Offset	0x0000 0030		
Physical Address	0x4800 4C30	**Instance**	WKUP_CM
Description	This register controls the automatic control of the WAKEUP modules interface clock activity. This activity is related to CORE domain activity.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																						RESERVED		RESERVED		AUTO_WDT2	RESERVED	AUTO_GPIO1	AUTO_32KSYNC	RESERVED	AUTO_GPT1

Bits	Field Name	Description	Type	Reset
31:10	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x000000
9	RESERVED	Reserved for non-GP devices	RW	0x0
8:6	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x0
5	AUTO_WDT2	WDTIMER 2 autoidle control 0x0: WDTIMER 2 interface clock is unrelated to the domain activity. 0x1: WDTIMER 2 interface clock is automaticaly enabled or disabled according to the domain activity.	RW	0x0
4	RESERVED	Reserved for non-GP devices	RW	0x0
3	AUTO_GPIO1	GPIO 1 autoidle control 0x0: GPIO 1 interface clock is unrelated to the domain activity. 0x1: GPIO 1 interface clock is automaticaly enabled / disabled according to the domain activity.	RW	0x0
2	AUTO_32KSYNC	32 kHz Sync Timer autoidle control 0x0: 32k Sync Timer interface clock is unrelated to the domain activity. 0x1: 32k Sync Timer interface clock is automaticaly enabled or disabled according to the domain activity.	RW	0x0
1	RESERVED	Reserved for non-GP devices	RW	0x0
0	AUTO_GPT1	GPTIMER 1 autoidle control 0x0: GPTIMER 1 interface clock is unrelated to the domain activity. 0x1: GPTIMER 1 interface clock is automaticaly enabled or disabled according to the domain activity.	RW	0x0

Table 4-174. Register Call Summary for Register CM_AUTOIDLE_WKUP

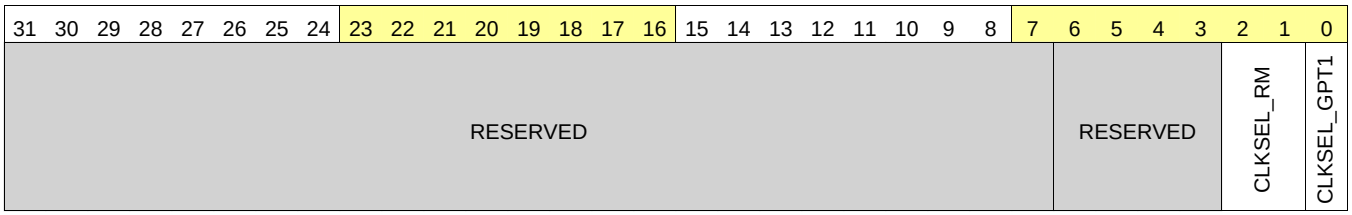
PRCM Clock Manager Functional Description
- WKUP Power Domain Clock Controls: [0]

PRCM Basic Programming Model
- Power-Domain Clock Control Registers: [1]

PRCM Register Manual
- WKUP_CM Registers: [2]

Table 4-175. CM_CLKSEL_WKUP

Address Offset	0x0000 0040		
Physical Address	0x4800 4C40	**Instance**	WKUP_CM
Description	WAKEUP domain modules source clock selection.		
Type	RW		



Bits	Field Name	Description	Type	Reset
31:7	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x0000000
6:3	RESERVED	Reserved for non-GP devices	RW	0x2
2:1	CLKSEL_RM	Selects the Reset Manager clock; Other enums: Reserved 0x1: RM_ICLK is L4_CLK divided by 1 0x2: RM_ICLK is L4_CLK divided by 2	RW	0x1
0	CLKSEL_GPT1	Selects GPTIMER 1 source clock 0x0: source is 32K_FCLK 0x1: source is SYS_CLK	RW	0x0

Table 4-176. Register Call Summary for Register CM_CLKSEL_WKUP

PRCM Clock Manager Functional Description • Interface and Peripheral Functional Clock Configurations: [0]
PRCM Basic Programming Model • Power-Domain Clock Control Registers: [1]
PRCM Register Manual • WKUP_CM Registers: [2]

4.14.1.8 Clock_Control_Reg_CM Registers

Table 4-177. Clock_Control_Reg_CM Register Summary

Register Name	Type	Register Width (Bits)	Address Offset	Physical Address	Reset Type
CM_CLKEN_PLL	RW	32	0x0000 0000	0x4800 4D00	W
CM_CLKEN2_PLL	RW	32	0x0000 0004	0x4800 4D04	W
CM_IDLEST_CKGEN	R	32	0x0000 0020	0x4800 4D20	C
CM_IDLEST2_CKGEN	R	32	0x0000 0024	0x4800 4D24	C
CM_AUTOIDLE_PLL	RW	32	0x0000 0030	0x4800 4D30	W
CM_AUTOIDLE2_PLL	RW	32	0x0000 0034	0x4800 4D34	W
CM_CLKSEL1_PLL	RW	32	0x0000 0040	0x4800 4D40	W
CM_CLKSEL2_PLL	RW	32	0x0000 0044	0x4800 4D44	W
CM_CLKSEL3_PLL	RW	32	0x0000 0048	0x4800 4D48	W
CM_CLKSEL4_PLL	RW	32	0x0000 004C	0x4800 4D4C	W
CM_CLKSEL5_PLL	RW	32	0x0000 0050	0x4800 4D50	W
CM_CLKOUT_CTRL	RW	32	0x0000 0070	0x4800 4D70	C

Table 4-178. CM_CLKEN_PLL

Address Offset	0x0000 0000		
Physical Address	0x4800 4D00	**Instance**	Clock_Control_Reg_CM
Description	This register allows controlling the DPLL3 and DPLL4 modes.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PWRDN_EMU_PERIPH	PWRDN_CAM	PWRDN_DSS1	PWRDN_TV	PWRDN_96M	EN_PERIPH_DPLL_LPMODE	RESERVED		PERIPH_DPLL_FREQSEL				EN_PERIPH_DPLL_DRIFTGUARD	EN_PERIPH_DPLL			RESERVED			PWRDN_EMU_CORE	RESERVED	EN_CORE_DPLL_LPMODE	RESERVED		CORE_DPLL_FREQSEL				EN_CORE_DPLL_DRIFTGUARD	EN_CORE_DPLL		

Bits	Field Name	Description	Type	Reset
31	PWRDN_EMU_PERIPH	This bit allows to power-down or not the DPLL4_M6X2_CLK HSDIVIDER path. 0x0: Power-up the DPLL4_M6X2_CLK HSDIVIDER path. 0x1: Power-down the DPLL4_M6X2_CLK HSDIVIDER path. Writting this bit to 1 will take effect immediatly.	RW	0x0
30	PWRDN_CAM	This bit allows to power-down or not the DPLL4_M5X2_CLK HSDIVIDER path. 0x0: Power-up the DPLL4_M5X2_CLK HSDIVIDER path. 0x1: Power-down the DPLL4_M5X2_CLK HSDIVIDER path. Writting this bit to 1 will take effect immediatly.	RW	0x0
29	PWRDN_DSS1	This bit allows to power-down or not the DPLL4_M4X2_CLK HSDIVIDER path. 0x0: Power-up the DPLL4_M4X2_CLK HSDIVIDER path. 0x1: Power-down the DPLL4_M4X2_CLK HSDIVIDER path. Writting this bit to 1 will take effect immediatly.	RW	0x0
28	PWRDN_TV	This bit allows to power-down or not the DPLL4_M3X2_CLK HSDIVIDER path. 0x0: Power-up the DPLL4_M3X2_CLK HSDIVIDER path. 0x1: Power-down the DPLL4_M3X2_CLK HSDIVIDER path. Writting this bit to 1 will take effect immediatly.	RW	0x0
27	PWRDN_96M	This bit allows to power-down or not the DPLL4_M2X2_CLK path. 0x0: Power-up the DPLL4_M2X2_CLK path. 0x1: Power-down the DPLL4_M2X2_CLK path. Writting this bit to 1 will take effect immediatly.	RW	0x0
26	EN_PERIPH_DPLL_LPMODE	This bit allows to enable or disable the LP mode of the DPLL4. Writting this bit to switch the mode between LP or normal mode will take effect only when the DPLL will have transition into the bypass or stop state, followed by a lock or re-lock of the DPLL. 0x0: Disables the DPLL LP mode to re-enter the normal mode at the following lock or re-lock sequence. 0x1: Enables the DPLL LP mode to enter the LP mode at the following lock or re-lock sequence.	RW	0x0
25:24	RESERVED	Write 0s for future compatibility. Read returns 0.	RW	0X0

Bits	Field Name	Description	Type	Reset
23:20	PERIPH_DPLL_FREQSEL	This bit field allows selecting the proper range of the DPLL4 internal frequency depending on the DPLL reference clock and the N divider. 0x3: 0.75 MHz—1.0 MHz 0x4: 1.0 MHz—1.25 MHz 0x5: 1.25 MHz—1.5 MHz 0x6: 1.5 MHz—1.75 MHz 0x7: 1.75 MHz—2.1 MHz 0xB: 7.5 MHz—10 MHz 0xC: 10 MHz—12.5 MHz 0xD: 12.5 MHz—15 MHz 0xE: 15 MHz—17.5 MHz 0xF: 17.5 MHz—21 MHz	RW	0x1
19	EN_PERIPH_DPLL_DRIFTGUARD	This bit allows to enable or disable the automatic recalibration feature of the DPLL4. The DPLL4 will automatically start a recalibration process upon assertion of the recal flag if this bit is set. 0x0: Disables the DPLL4 automatic recalibration mode 0x1: Enables the DPLL4 automatic recalibration mode	RW	0x0
18:16	EN_PERIPH_DPLL	DPLL4 control; Other enums: Reserved 0x1: Put the DPLL4 in low power stop mode 0x7: Enables the DPLL4 in lock mode	RW	0x1
15:13	RESERVED		R	0x0
12	PWRDN_EMU_CORE	This bit allows to power-down or not the DPLL3_M3X2 HSDIVIDER path. 0x0: Power-up the DPLL3_M3X2 HSDIVIDER path. 0x1: Power-down the DPLL3_M3X2 HSDIVIDER path. Writting this bit to 1 will take effect immediatly.	RW	0x0
11	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x0
10	EN_CORE_DPLL_LPMODE	This bit allows to enable or disable the LP mode of the DPLL3. Writting this bit to switch the mode between LP or normal mode will take effect only when the DPLL will have transition into the bypass or stop state, followed by a lock or re-lock of the DPLL. 0x0: Disables the DPLL LP mode to re-enter the normal mode at the following lock or re-lock sequence. 0x1: Enables the DPLL LP mode to enter the LP mode at the following lock or re-lock sequence.	RW	0x0
9:8	RESERVED	Write 0s for future compatibility. Read returns 0.	RW	0x0
7:4	CORE_DPLL_FREQSEL	This bit field allows selecting the proper range of the DPLL3 internal frequency depending on the DPLL reference clock and the N divider. 0x3: 0.75 MHz—1.0 MHz 0x4: 1.0 MHz—1.25 MHz 0x5: 1.25 MHz—1.5 MHz 0x6: 1.5 MHz—1.75 MHz 0x7: 1.75 MHz—2.1 MHz 0xB: 7.5 MHz—10 MHz 0xC: 10 MHz—12.5 MHz 0xD: 12.5 MHz—15 MHz 0xE: 15 MHz—17.5 MHz 0xF: 17.5 MHz—21 MHz	RW	0x1
3	EN_CORE_DPLL_DRIFTGUARD	This bit allows to enable or disable the automatic recalibration feature of the DPLL3. The DPLL3 will automatically start a recalibration process upon assertion of the recal flag if this bit is set. 0x0: Disables the DPLL3 automatic recalibration mode 0x1: Enables the DPLL3 automatic recalibration mode	RW	0x0

Bits	Field Name	Description	Type	Reset
2:0	EN_CORE_DPLL	DPLL3 control; Other enums: Reserved 0x5: Put the DPLL3 in low power bypass 0x6: Put the DPLL3 in fast relock bypass 0x7: Enables the DPLL3 in lock mode	RW	0x5

Table 4-179. Register Call Summary for Register CM_CLKEN_PLL

PRCM Clock Manager Functional Description
- DPLL Jitter Correction :
- DPLL Frequency Ramp-Up Delay :
- DPLL Modes: [6] [7]
- DPLL Low-Power Mode: [8] [9]
- DPLL Clock Path Power Down: [10] [11] [12] [13] [14] [15]
- Latencies :
- Recalibration: [18] [19]
- DPLL Source-Clock Controls: [20] [21]

PRCM Basic Programming Model
- DPLL Clock Control Registers: [22]
- Clock Control: [23]

PRCM Register Manual
- Clock_Control_Reg_CM Registers: [24]

Table 4-180. CM_CLKEN2_PLL

Address Offset	0x0000 0004		
Physical Address	0x4800 4D04	**Instance**	Clock_Control_Reg_CM
Description	This register controls the DPLL5 modes.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																					EN_PERIPH2_DPLL_LPMODE	RESERVED			PERIPH2_DPLL_FREQSEL				EN_PERIPH2_DPLL_DRIFTGUARD	EN_PERIPH2_DPLL		

Bits	Field Name	Description	Type	Reset
31:11	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x000000
10	EN_PERIPH2_DPLL_LPMODE	This bit allows to enable or disable the LP mode of the DPLL5. Writting this bit to switch the mode between LP or normal mode will take effect only when the DPLL will have transition into the bypass or stop state, followed by a lock or re-lock of the DPLL. 0x0: Disables the DPLL LP mode to re-enter the normal mode at the following lock or re-lock sequence. 0x1: Enables the DPLL LP mode to enter the LP mode at the following lock or re-lock sequence.	RW	0x0
9:8	RESERVED	Write 0s for future compatibility. Read returns 0.	RW	0x0

Bits	Field Name	Description	Type	Reset
7:4	PERIPH2_DPLL_FREQSEL	This bit field allows selecting the proper range of the second PERIPHERAL DPLL internal frequency depending on the DPLL reference clock and the N divider. 0x3: 0.75 MHz—1.0 MHz 0x4: 1.0 MHz—1.25 MHz 0x5: 1.25 MHz—1.5 MHz 0x6: 1.5 MHz—1.75 MHz 0x7: 1.75 MHz—2.1 MHz 0xB: 7.5 MHz—10 MHz 0xC: 10 MHz—12.5 MHz 0xD: 12.5 MHz—15 MHz 0xE: 15 MHz—17.5 MHz 0xF: 17.5 MHz—21 MHz	RW	0x1
3	EN_PERIPH2_DPLL_DRIFTGUARD	This bit allows to enable or disable the automatic recalibration feature of the DPLL5. The DPLL5 will automatically start a recalibration process upon assertion of the recal flag if this bit is set. 0x0: Disables the DPLL5 automatic recalibration mode 0x1: Enables the DPLL5 automatic recalibration mode	RW	0x0
2:0	EN_PERIPH2_DPLL	DPLL5 control; Other enums: Reserved 0x1: Put the second DPLL5 in low power stop mode 0x7: Enables the DPLL5 in lock mode	RW	0x1

Table 4-181. Register Call Summary for Register CM_CLKEN2_PLL

PRCM Clock Manager Functional Description
- DPLL Jitter Correction :
- DPLL Frequency Ramp-Up Delay :
- DPLL Modes: [2]
- DPLL Low-Power Mode: [3]
- Latencies :
- Recalibration: [5]
- DPLL Source-Clock Controls: [6]

PRCM Register Manual
- Clock_Control_Reg_CM Registers: [7]

Table 4-182. CM_IDLEST_CKGEN

Address Offset	0x0000 0020		
Physical Address	0x4800 4D20	**Instance**	Clock_Control_Reg_CM
Description	This register allows monitoring the master clock activity. This register is read only and automatically updated.		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																		ST_EMU_PERIPH_CLK	ST_CAM_CLK	ST_DSS1_CLK	ST_TV_CLK	ST_FUNC96M_CLK	ST_EMU_CORE_CLK	RESERVED		ST_54M_CLK	ST_12M_CLK	ST_48M_CLK	ST_96M_CLK	ST_PERIPH_CLK	ST_CORE_CLK

Bits	Field Name	Description	Type	Reset
31:14	RESERVED	Read returns 0.	R	0x00000
13	ST_EMU_PERIPH_CLK	Emulation clock activity at the output stage of the DPLL4 0x0: EMU_PER_ALWON_CLK is not active 0x1: EMU_PER_ALWON_CLK is active	R	0x0
12	ST_CAM_CLK	CAMERA functional clock activity at the output stage of the DPLL4 0x0: CAM_MCLK is not active 0x1: CAM_MCLK is active	R	0x0
11	ST_DSS1_CLK	DSS functional clock 1 activity at the output stage of the DPLL4 0x0: DSS1_ALWON_FCLK is not active 0x1: DSS1_ALWON_FCLK is active	R	0x0
10	ST_TV_CLK	TV clock activity at the output stage of the DPLL4 0x0: DPLL4_M3X2_CLK is not active 0x1: DPLL4_M3X2_CLK is active	R	0x0
9	ST_FUNC96M_CLK	96 MHz clock activity at the output stage of the DPLL4 0x0: DPLL4_M2X2_CLK is not active 0x1: DPLL4_M2X2_CLK is active	R	0x0
8	ST_EMU_CORE_CLK	Emulation clock activity at the output stage of the DPLL3 0x0: EMU_CORE_ALWON_CLK is not active 0x1: EMU_CORE_ALWON_CLK is active	R	0x0
7:6	RESERVED	Read returns 0.	R	0x0
5	ST_54M_CLK	Functional clock 54 MHz activity 0x0: 54MHz clock is not active 0x1: 54MHz clock is active	R	0x0
4	ST_12M_CLK	Functional clock 12 MHz activity 0x0: 12M_FCLK is not active 0x1: 12M_FCLK is active	R	0x0
3	ST_48M_CLK	Functional clock 48 MHz activity 0x0: 48M_FCLK is not active 0x1: 48M_FCLK is active	R	0x0
2	ST_96M_CLK	Functional clock 96 MHz activity 0x0: 96M_FCLK is not active 0x1: 96M_FCLK is active	R	0x0
1	ST_PERIPH_CLK	DPLL4 clock activity 0x0: DPLL4 is bypassed 0x1: DPLL4 is locked	R	0x0
0	ST_CORE_CLK	DPLL3 clock activity 0x0: DPLL3 is bypassed 0x1: DPLL3 is locked	R	0x0

Table 4-183. Register Call Summary for Register CM_IDLEST_CKGEN

PRCM Basic Programming Model • DPLL Clock Control Registers: [0]
PRCM Register Manual • Clock_Control_Reg_CM Registers: [1]

Table 4-184. CM_IDLEST2_CKGEN

Address Offset	0x0000 0024		
Physical Address	0x4800 4D24	**Instance**	Clock_Control_Reg_CM
Description	This register allows monitoring the master clock activity. This register is read only and automatically updated.		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																												ST_FUNC120M_CLK	RESERVED	ST_120M_CLK	ST_PERIPH2_CLK

Bits	Field Name	Description	Type	Reset
31:4	RESERVED	Read returns 0.	R	0x0000000
3	ST_FUNC120M_CLK	120 MHz clock activity at the output stage of the DPLL5 0x0: DPLL5_M2_CLK is not active 0x1: DPLL5_M2_CLK is active	R	0x0
2	RESERVED	Reserved for non-GP devices	R	0x0
1	ST_120M_CLK	Functional clock 120 MHz activity 0x0: 120M_FCLK is not active 0x1: 120M_FCLK is active	R	0x0
0	ST_PERIPH2_CLK	DPLL5 clock activity 0x0: DPLL5 is bypassed 0x1: DPLL5 is locked	R	0x0

Table 4-185. Register Call Summary for Register CM_IDLEST2_CKGEN

PRCM Basic Programming Model
- DPLL Clock Control Registers: [0]

PRCM Register Manual
- Clock_Control_Reg_CM Registers: [1]

Table 4-186. CM_AUTOIDLE_PLL

Address Offset	0x0000 0030		
Physical Address	0x4800 4D30	**Instance**	Clock_Control_Reg_CM
Description	This register provides automatic control over the DPLL3 and DPLL4 activity.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																										AUTO_PERIPH_DPLL				AUTO_CORE_DPLL	

Bits	Field Name	Description	Type	Reset
31:6	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x0000000
5:3	AUTO_PERIPH_DPLL	DPLL4 automatic control; Other enums: Reserved 0x0: Auto control disabled 0x1: DPLL4 is automatically put in low power stop mode when none of the 96 MHz and 54 MHz clocks are required anymore. It is also restarted automatically.	RW	0x0
2:0	AUTO_CORE_DPLL	DPLL3 automatic control; Other enums: Reserved 0x0: Auto control disabled 0x1: DPLL3 is automatically put in low power stop mode when the CORE clock is not required anymore. It is also restarted automatically. 0x5: DPLL3 is automatically put in idle bypass low power mode when the CORE clock is not required anymore. It is also restarted automatically.	RW	0x0

Table 4-187. Register Call Summary for Register CM_AUTOIDLE_PLL

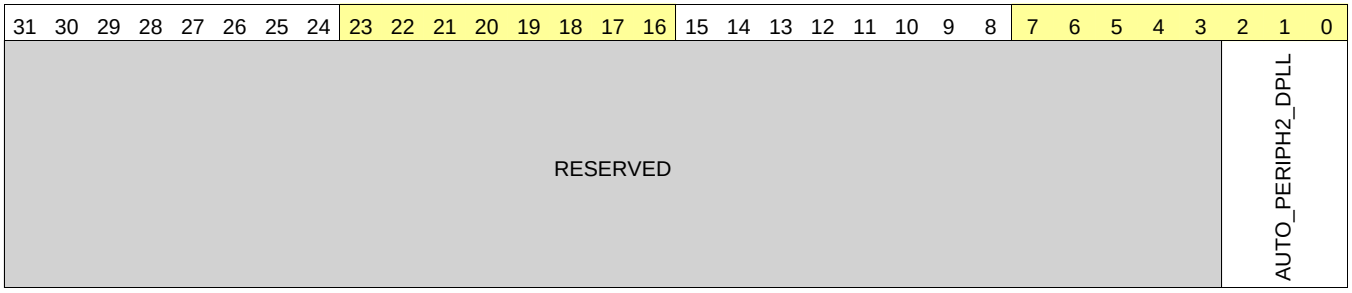
PRCM Clock Manager Functional Description
- DPLL Modes: [0] [1] [2] [3]
- DPLL Source-Clock Controls: [4] [5]

PRCM Basic Programming Model
- DPLL Clock Control Registers: [6]

PRCM Register Manual
- Clock_Control_Reg_CM Registers: [7]

Table 4-188. CM_AUTOIDLE2_PLL

Address Offset	0x0000 0034		
Physical Address	0x4800 4D34	**Instance**	Clock_Control_Reg_CM
Description	This register provides automatic control over the DPLL5 activity.		
Type	RW		



Bits	Field Name	Description	Type	Reset
31:3	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x00000000
2:0	AUTO_PERIPH2_DPLL	DPLL5 automatic control; Other enums: Reserved 0x0: Auto control disabled 0x1: DPLL5 is automatically put in low power stop mode when the 120 MHz clock is not required anymore. It is also restarted automatically.	RW	0x0

Table 4-189. Register Call Summary for Register CM_AUTOIDLE2_PLL

PRCM Clock Manager Functional Description
- DPLL Modes: [0]
- DPLL Source-Clock Controls: [1]

Table 4-189. Register Call Summary for Register CM_AUTOIDLE2_PLL (continued)

PRCM Register Manual • Clock_Control_Reg_CM Registers: [2]

Table 4-190. CM_CLKSEL1_PLL

Address Offset	0x0000 0040		
Physical Address	0x4800 4D40	**Instance**	Clock_Control_Reg_CM
Description	This register controls the selection of the master clock frequencies.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CORE_DPLL_CLKOUT_DIV								CORE_DPLL_MULT								RESERVED	CORE_DPLL_DIV							RESERVED	SOURCE_96M	SOURCE_54M	RESERVED	SOURCE_48M	RESERVED		

Bits	Field Name	Description	Type	Reset
31:27	CORE_DPLL_CLKOUT_DIV	DPLL3 output clock divider factor M2; Other enums: Reserved 0x1: DPLL3 output clock is divided by 1 0x2: DPLL3 output clock is divided by 2 0x3: DPLL3 output clock is divided by 3 0x4: DPLL3 output clock is divided by 4 0x5: DPLL3 output clock is divided by 5 0x6: DPLL3 output clock is divided by 6 0x7: DPLL3 output clock is divided by 7 0x8: DPLL3 output clock is divided by 8 0x9: DPLL3 output clock is divided by 9 0xA: DPLL3 output clock is divided by 10 0xB: DPLL3 output clock is divided by 11 0xC: DPLL3 output clock is divided by 12 0xD: DPLL3 output clock is divided by 13 0xE: DPLL3 output clock is divided by 14 0xF: DPLL3 output clock is divided by 15 0x10: DPLL3 output clock is divided by 16 0x11: DPLL3 output clock is divided by 17 0x12: DPLL3 output clock is divided by 18 0x13: DPLL3 output clock is divided by 19 0x14: DPLL3 output clock is divided by 20 0x15: DPLL3 output clock is divided by 21 0x16: DPLL3 output clock is divided by 22 0x17: DPLL3 output clock is divided by 23 0x18: DPLL3 output clock is divided by 24 0x19: DPLL3 output clock is divided by 25 0x1A: DPLL3 output clock is divided by 26 0x1B: DPLL3 output clock is divided by 27 0x1C: DPLL3 output clock is divided by 28 0x1D: DPLL3 output clock is divided by 29 0x1E: DPLL3 output clock is divided by 30 0x1F: DPLL3 output clock is divided by 31	RW	0x01
26:16	CORE_DPLL_MULT	DPLL3 multiplier factor (0 to 2047)	RW	0x000
15	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x0
14:8	CORE_DPLL_DIV	DPLL3 divider factor (0 to 127)	RW	0x00
7	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x0
6	SOURCE_96M	Selection of 96M_FCLK source 0x0: source is the CM_96M_FCLK 0x1: source is CM_SYS_CLK	RW	0x1
5	SOURCE_54M	Selection of 54MHz clock source 0x0: source is the DPLL4_M3X2_CLK 0x1: source is sys_altclk	RW	0x0
4	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x0
3	SOURCE_48M	Selection of Func_12M_clk and Func_48M_clk source 0x0: source is the CM_96M_FCLK 0x1: source is sys_altclk	RW	0x0
2:0	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x0

Table 4-191. Register Call Summary for Register CM_CLKSEL1_PLL

PRCM Clock Manager Functional Description • PRM Source-Clock Controls: [0] • CM Source-Clock Controls: [1] • Interface and Peripheral Functional Clock Configurations: [2] [3] [4]
PRCM Basic Programming Model • DPLL Clock Control Registers: [5]
PRCM Use Cases and Tips • DVFS Using SmartReflex :
PRCM Register Manual • Clock_Control_Reg_CM Registers: [8]

Table 4-192. CM_CLKSEL2_PLL

Address Offset	0x0000 0044		
Physical Address	0x4800 4D44	**Instance**	Clock_Control_Reg_CM
Description	This register controls the selection of the master clock frequencies.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								PERIPH_DPLL_MULT																RESERVED	PERIPH_DPLL_DIV						

Bits	Field Name	Description	Type	Reset
31:19	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x0000
18:8	PERIPH_DPLL_MULT	DPLL4 multiplier factor (0 to 2047)	RW	0x000
7	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x0
6:0	PERIPH_DPLL_DIV	DPLL4 divider factor (0 to 127)	RW	0x00

Table 4-193. Register Call Summary for Register CM_CLKSEL2_PLL

PRCM Basic Programming Model • DPLL Clock Control Registers: [0]
PRCM Register Manual • Clock_Control_Reg_CM Registers: [1]

Table 4-194. CM_CLKSEL3_PLL

Address Offset	0x0000 0048		
Physical Address	0x4800 4D48	**Instance**	Clock_Control_Reg_CM
Description	This register controls the selection of the master clock frequencies.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																												DIV_96M			

Bits	Field Name	Description	Type	Reset
31:5	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x0000000
4:0	DIV_96M	96 MHz clock divider factor M2 (1 up to 16); Other enums: Reserved 0x1: 96 MHz clock is DPLL4 clock divided by 1 0x2: 96 MHz clock is DPLL4 clock divided by 2 0x3: 96 MHz clock is DPLL4 clock divided by 3 0x4: 96 MHz clock is DPLL4 clock divided by 4 0x5: 96 MHz clock is DPLL4 clock divided by 5 0x6: 96 MHz clock is DPLL4 clock divided by 6 0x7: 96 MHz clock is DPLL4 clock divided by 7 0x8: 96 MHz clock is DPLL4 clock divided by 8 0x9: 96 MHz clock is DPLL4 clock divided by 9 0xA: 96 MHz clock is DPLL4 clock divided by 10 0xB: 96 MHz clock is DPLL4 clock divided by 11 0xC: 96 MHz clock is DPLL4 clock divided by 12 0xD: 96 MHz clock is DPLL4 clock divided by 13 0xE: 96 MHz clock is DPLL4 clock divided by 14 0xF: 96 MHz clock is DPLL4 clock divided by 15 0x10: 96 MHz clock is DPLL4 clock divided by 16	RW	0x01

Table 4-195. Register Call Summary for Register CM_CLKSEL3_PLL

PRCM Basic Programming Model • DPLL Clock Control Registers: [0]
PRCM Register Manual • Clock_Control_Reg_CM Registers: [1]

Table 4-196. CM_CLKSEL4_PLL

Address Offset	0x0000 004C		
Physical Address	0x4800 4D4C	**Instance**	Clock_Control_Reg_CM
Description	This register controls the selection of the master clock frequencies.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																PERIPH2_DPLL_MULT								RESERVED	PERIPH2_DPLL_DIV						

Bits	Field Name	Description	Type	Reset
31:19	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x0000
18:8	PERIPH2_DPLL_MULT	DPLL5 multiplier factor (0 to 2047)	RW	0x000
7	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x0
6:0	PERIPH2_DPLL_DIV	DPLL5 divider factor (0 to 127)	RW	0x00

Table 4-197. Register Call Summary for Register CM_CLKSEL4_PLL

PRCM Basic Programming Model • DPLL Clock Control Registers: [0]
PRCM Register Manual • Clock_Control_Reg_CM Registers: [1]

Table 4-198. CM_CLKSEL5_PLL

Address Offset	0x0000 0050		
Physical Address	0x4800 4D50	**Instance**	Clock_Control_Reg_CM
Description	This register controls the selection of the master clock frequencies.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																								DIV_120M							

Bits	Field Name	Description	Type	Reset
31:5	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x0000000
4:0	DIV_120M	120 MHz clock divider factor M2 (1 up to 16); Other enums: Reserved 0x1: 120 MHz clock is DPLL5 clock divided by 1 0x2: 120 MHz clock is DPLL5 clock divided by 2 0x3: 120 MHz clock is DPLL5 clock divided by 3 0x4: 120 MHz clock is DPLL5 clock divided by 4 0x5: 120 MHz clock is DPLL5 clock divided by 5 0x6: 120 MHz clock is DPLL5 clock divided by 6 0x7: 120 MHz clock is DPLL5 clock divided by 7 0x8: 120 MHz clock is DPLL5 clock divided by 8 0x9: 120 MHz clock is DPLL5 clock divided by 9 0xA: 120 MHz clock is DPLL5 clock divided by 10 0xB: 120 MHz clock is DPLL5 clock divided by 11 0xC: 120 MHz clock is DPLL5 clock divided by 12 0xD: 120 MHz clock is DPLL5 clock divided by 13 0xE: 120 MHz clock is DPLL5 clock divided by 14 0xF: 120 MHz clock is DPLL5 clock divided by 15 0x10: 120 MHz clock is DPLL5 clock divided by 16	RW	0x01

Table 4-199. Register Call Summary for Register CM_CLKSEL5_PLL

PRCM Basic Programming Model
- DPLL Clock Control Registers: [0]

PRCM Register Manual
- Clock_Control_Reg_CM Registers: [1]

Table 4-200. CM_CLKOUT_CTRL

Address Offset	0x0000 0070		
Physical Address	0x4800 4D70	**Instance**	Clock_Control_Reg_CM
Description	This register provides control over the SYS_CLKOUT2 output clock.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																								CLKOUT2_EN	RESERVED	CLKOUT2_DIV			RESERVED	CLKOUT2SOURCE	

Bits	Field Name	Description	Type	Reset
31:8	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x000000
7	CLKOUT2_EN	This bit controls the external output clock activity 0x0: sys_clkout2 is disabled 0x1: sys_clkout2 is enabled	RW	0x0
6	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x0
5:3	CLKOUT2_DIV	This field controls the external output clock division; Other enums: Reserved 0x0: sys_clkout2 / 1 0x1: sys_clkout2 / 2 0x2: sys_clkout2 / 4 0x3: sys_clkout2 / 8 0x4: sys_clkout2 / 16	RW	0x0
2	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x0
1:0	CLKOUT2SOURCE	This field selects the external output clock source 0x0: source is CORE_CLK 0x1: source is CM_SYS_CLK 0x2: source is CM_96M_FCLK 0x3: source is 54 MHz clock	RW	0x3

Table 4-201. Register Call Summary for Register CM_CLKOUT_CTRL

PRCM Clock Manager Functional Description • CM Source-Clock Controls: [0]
PRCM Register Manual • Clock_Control_Reg_CM Registers: [1]

4.14.1.9 DSS_CM Registers

Table 4-202. DSS_CM Register Summary

Register Name	Type	Register Width (Bits)	Address Offset	Physical Address	Reset Type
CM_FCLKEN_DSS	RW	32	0x0000 0000	0x4800 4E00	W
CM_ICLKEN_DSS	RW	32	0x0000 0010	0x4800 4E10	W
CM_IDLEST_DSS	R	32	0x0000 0020	0x4800 4E20	C
CM_AUTOIDLE_DSS	RW	32	0x0000 0030	0x4800 4E30	W
CM_CLKSEL_DSS	RW	32	0x0000 0040	0x4800 4E40	W
CM_SLEEPDEP_DSS	RW	32	0x0000 0044	0x4800 4E44	W
CM_CLKSTCTRL_DSS	RW	32	0x0000 0048	0x4800 4E48	W
CM_CLKSTST_DSS	R	32	0x0000 004C	0x4800 4E4C	C

Table 4-203. CM_FCLKEN_DSS

Address Offset	0x0000 0000		
Physical Address	0x4800 4E00	**Instance**	DSS_CM
Description	Controls the modules functional clock activity.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																													EN_TV	EN_DSS2	EN_DSS1

Bits	Field Name	Description	Type	Reset
31:3	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x00000000
2	EN_TV	DSS_TV_FCLK functional clock control 0x0: DSS_TV_FCLK is disabled 0x1: DSS_TV_FCLK is enabled	RW	0x0
1	EN_DSS2	Display Sub-System functional clock 2 control 0x0: DSS2_ALWON_FCLK is disabled 0x1: DSS2_ALWON_FCLK is enabled	RW	0x0
0	EN_DSS1	Display Sub-System functional clock 1 control 0x0: DSS1_ALWON_FCLK is disabled 0x1: DSS1_ALWON_FCLK is enabled	RW	0x0

Table 4-204. Register Call Summary for Register CM_FCLKEN_DSS

PRCM Clock Manager Functional Description • DSS Power Domain Clock Controls: [0] [1] [2]
PRCM Basic Programming Model • Power-Domain Clock Control Registers: [3] [4]
PRCM Register Manual • DSS_CM Registers: [5]

Table 4-205. CM_ICLKEN_DSS

Address Offset	0x0000 0010		
Physical Address	0x4800 4E10	**Instance**	DSS_CM
Description	Controls the modules interface clock activity.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																															EN_DSS

Bits	Field Name	Description	Type	Reset
31:1	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x00000000
0	EN_DSS	Display sub-system interface clock control 0x0: DSS_L3_ICLK and DSS_L4_ICLK are disabled 0x1: DSS_L3_ICLK and DSS_L4_ICLK are enabled	RW	0x0

Table 4-206. Register Call Summary for Register CM_ICLKEN_DSS

PRCM Clock Manager Functional Description • DSS Power Domain Clock Controls: [0]
PRCM Basic Programming Model • Power-Domain Clock Control Registers: [1]
PRCM Register Manual • DSS_CM Registers: [2]

Table 4-207. CM_IDLEST_DSS

Address Offset	0x0000 0020		
Physical Address	0x4800 4E20	**Instance**	DSS_CM
Description	Modules access availability monitoring. This register is read only and automatically updated.		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																														ST_DSS_IDLE	ST_DSS_STDBY

Bits	Field Name	Description	Type	Reset
31:2	RESERVED	Read returns 0.	R	0x00000000
1	ST_DSS_IDLE	Display Sub-System idle status. 0x0: Display Sub-System is active. 0x1: Display Sub-System is in idle mode and cannot be accessed.	R	0x1
0	ST_DSS_STDBY	Display Sub-System standby status. 0x0: Display Sub-System is active. 0x1: Display Sub-System is in standby mode.	R	0x1

Table 4-208. Register Call Summary for Register CM_IDLEST_DSS

PRCM Basic Programming Model • Power-Domain Clock Control Registers: [0]
PRCM Register Manual • DSS_CM Registers: [1]

Table 4-209. CM_AUTOIDLE_DSS

Address Offset	0x0000 0030		
Physical Address	0x4800 4E30	**Instance**	DSS_CM
Description	This register controls the automatic control of the modules interface clock activity.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																															AUTO_DSS

Bits	Field Name	Description	Type	Reset
31:1	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x00000000
0	AUTO_DSS	Display Sub-System auto clock control. 0x0: Display Sub-System interface clock is unrelated to the domain state transition. 0x1: Display Sub-System interface clock is automatically enabled or disabled along with the domain state transition.	RW	0x0

Table 4-210. Register Call Summary for Register CM_AUTOIDLE_DSS

PRCM Clock Manager Functional Description • DSS Power Domain Clock Controls: [0]
PRCM Basic Programming Model • Power-Domain Clock Control Registers: [1]
PRCM Register Manual • DSS_CM Registers: [2]

Table 4-211. CM_CLKSEL_DSS

Address Offset	0x0000 0040		
Physical Address	0x4800 4E40	**Instance**	DSS_CM
Description	Modules clock selection.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																			CLKSEL_TV					RESERVED				CLKSEL_DSS1			

Bits	Field Name	Description	Type	Reset
31:13	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x00000
12:8	CLKSEL_TV	TV functional clock divider factor DPLL4 M3 (1 up to 16); Other enums: Reserved 0x1: TV functional clock is DPLL4 clock divided by 1 0x2: TV functional clock is DPLL4 clock divided by 2 0x3: TV functional clock is DPLL4 clock divided by 3 0x4: TV functional clock is DPLL4 clock divided by 4 0x5: TV functional clock is DPLL4 clock divided by 5 0x6: TV functional clock is DPLL4 clock divided by 6 0x7: TV functional clock is DPLL4 clock divided by 7 0x8: TV functional clock is DPLL4 clock divided by 8 0x9: TV functional clock is DPLL4 clock divided by 9 0xA: TV functional clock is DPLL4 clock divided by 10 0xB: TV functional clock is DPLL4 clock divided by 11 0xC: TV functional clock is DPLL4 clock divided by 12 0xD: TV functional clock is DPLL4 clock divided by 13 0xE: TV functional clock is DPLL4 clock divided by 14 0xF: TV functional clock is DPLL4 clock divided by 15 0x10: TV functional clock is DPLL4 clock divided by 16	RW	0x10
7:5	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x0

Bits	Field Name	Description	Type	Reset
4:0	CLKSEL_DSS1	DPLL4 M4 divide factor for DSS1_ALWON_FCLK (1 up to 16); Other enums: Reserved 0x1: DSS1_ALWON_FCLK is DPLL4 clock divided by 1 0x2: DSS1_ALWON_FCLK is DPLL4 clock divided by 2 0x3: DSS1_ALWON_FCLK is DPLL4 clock divided by 3 0x4: DSS1_ALWON_FCLK is DPLL4 clock divided by 4 0x5: DSS1_ALWON_FCLK is DPLL4 clock divided by 5 0x6: DSS1_ALWON_FCLK is DPLL4 clock divided by 6 0x7: DSS1_ALWON_FCLK is DPLL4 clock divided by 7 0x8: DSS1_ALWON_FCLK is DPLL4 clock divided by 8 0x9: DSS1_ALWON_FCLK is DPLL4 clock divided by 9 0xA: DSS1_ALWON_FCLK is DPLL4 clock divided by 10 0xB: DSS1_ALWON_FCLK is DPLL4 clock divided by 11 0xC: DSS1_ALWON_FCLK is DPLL4 clock divided by 12 0xD: DSS1_ALWON_FCLK is DPLL4 clock divided by 13 0xE: DSS1_ALWON_FCLK is DPLL4 clock divided by 14 0xF: DSS1_ALWON_FCLK is DPLL4 clock divided by 15 0x10: DSS1_ALWON_FCLK is DPLL4 clock divided by 16	RW	0x10

Table 4-212. Register Call Summary for Register CM_CLKSEL_DSS

PRCM Basic Programming Model
- Power-Domain Clock Control Registers: [0]

PRCM Register Manual
- DSS_CM Registers: [1]

Table 4-213. CM_SLEEPDEP_DSS

Address Offset	0x0000 0044		
Physical Address	0x4800 4E44	**Instance**	DSS_CM
Description	This register allows enabling or disabling the sleep transition dependency of DSS domain with respect to other domain.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																													EN_IVA2	EN_MPU	EN_CORE

Bits	Field Name	Description	Type	Reset
31:3	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x00000000
2	EN_IVA2	IVA2 domain dependency 0x0: DSS domain sleep dependency with IVA2 domain is disabled. 0x1: DSS domain sleep dependency with IVA2 domain is enabled.	RW	0x0
1	EN_MPU	MPU domain dependency 0x0: DSS domain sleep dependency with MPU domain is disabled. 0x1: DSS domain sleep dependency with MPU domain is enabled.	RW	0x0
0	EN_CORE	CORE domain dependency	RW	0

Table 4-214. Register Call Summary for Register CM_SLEEPDEP_DSS

PRCM Basic Programming Model • Power-Domain Clock Control Registers: [0]
PRCM Register Manual • DSS_CM Registers: [1]

Table 4-215. CM_CLKSTCTRL_DSS

Address Offset	0x0000 0048		
Physical Address	0x4800 4E48	**Instance**	DSS_CM
Description	This register enables the domain power state transition. It controls the HW supervised domain power state transition between ACTIVE and INACTIVE states.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																														CLKTRCTRL_DSS	

Bits	Field Name	Description	Type	Reset
31:2	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x00000000
1:0	CLKTRCTRL_DSS	Controls the clock state transition of the DSS clock domain. 0x0: Automatic transition is disabled 0x1: Start a software supervised sleep transition on the domain 0x2: Start a software supervised wake-up transition on the domain 0x3: Automatic transition is enabled. Transition is supervised by the HardWare.	RW	0x0

Table 4-216. Register Call Summary for Register CM_CLKSTCTRL_DSS

PRCM Idle and Wake-Up Management • Device Wake-Up Events: [0] [1] [2]
PRCM Basic Programming Model • Power-Domain Clock Control Registers: [3]
PRCM Register Manual • DSS_CM Registers: [4]

Table 4-217. CM_CLKSTST_DSS

Address Offset	0x0000 004C		
Physical Address	0x4800 4E4C	**Instance**	DSS_CM
Description	This register provides a status on the OCP interface clock activity in the domain.		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																															CLKACTIVITY_DSS

Bits	Field Name	Description	Type	Reset
31:1	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x00000000
0	CLKACTIVITY_DSS	Interface clock activity status 0x0: No domain Interface clock activity 0x1: Domain Interface clock is active	R	0x0

Table 4-218. Register Call Summary for Register CM_CLKSTST_DSS

PRCM Basic Programming Model • Power-Domain Clock Control Registers: [0]
PRCM Register Manual • DSS_CM Registers: [1]

4.14.1.10 CAM_CM Registers

Table 4-219. CAM_CM Register Summary

Register Name	Type	Register Width (Bits)	Address Offset	Physical Address	Reset Type
CM_FCLKEN_CAM	RW	32	0x0000 0000	0x4800 4F00	W
CM_ICLKEN_CAM	RW	32	0x0000 0010	0x4800 4F10	W
CM_IDLEST_CAM	R	32	0x0000 0020	0x4800 4F20	C
CM_AUTOIDLE_CAM	RW	32	0x0000 0030	0x4800 4F30	W
CM_CLKSEL_CAM	RW	32	0x0000 0040	0x4800 4F40	W
CM_SLEEPDEP_CAM	RW	32	0x0000 0044	0x4800 4F44	W
CM_CLKSTCTRL_CAM	RW	32	0x0000 0048	0x4800 4F48	W
CM_CLKSTST_CAM	R	32	0x0000 004C	0x4800 4F4C	C

Table 4-220. CM_FCLKEN_CAM

Address Offset	0x0000 0000		
Physical Address	0x4800 4F00	**Instance**	CAM_CM
Description	Controls the modules functional clock activity.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																														EN_CSI2	EN_CAM

Bits	Field Name	Description	Type	Reset
31:2	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x00000000
1	EN_CSI2	CSI2 functional clock control (96 MHz) 0x0: CSI2_96M_FCLK is disabled 0x1: CSI2_96M_FCLK is enabled	RW	0x0
0	EN_CAM	Camera functional clock control 0x0: CAM_MCLK is disabled 0x1: CAM_MCLK is enabled	RW	0x0

Table 4-221. Register Call Summary for Register CM_FCLKEN_CAM

PRCM Clock Manager Functional Description • CAM Power Domain Clock Controls: [0]
PRCM Basic Programming Model • Power-Domain Clock Control Registers: [1]
PRCM Register Manual • CAM_CM Registers: [2]

Table 4-222. CM_ICLKEN_CAM

Address Offset	0x0000 0010		
Physical Address	0x4800 4F10	**Instance**	CAM_CM
Description	Controls the modules interface clock activity.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																															EN_CAM

Bits	Field Name	Description	Type	Reset
31:1	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x00000000
0	EN_CAM	Camera interface clock control 0x0: CAM_L3_ICK and CAM_L4_ICLK are disabled 0x1: CAM_L3_ICK and CAM_L4_ICLK are enabled	RW	0x0

Table 4-223. Register Call Summary for Register CM_ICLKEN_CAM

PRCM Clock Manager Functional Description • CAM Power Domain Clock Controls: [0]
PRCM Basic Programming Model • Power-Domain Clock Control Registers: [1]
PRCM Register Manual • CAM_CM Registers: [2]

Table 4-224. CM_IDLEST_CAM

Address Offset	0x0000 0020		
Physical Address	0x4800 4F20	**Instance**	CAM_CM
Description	Modules access availability monitoring. This register is read only and automatically updated.		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																															ST_CAM

Bits	Field Name	Description	Type	Reset
31:1	RESERVED	Read returns 0.	R	0x00000000
0	ST_CAM	Camera standby status. 0x0: Camera is active. 0x1: Camera is in standby mode.	R	0x1

Table 4-225. Register Call Summary for Register CM_IDLEST_CAM

PRCM Basic Programming Model
- Power-Domain Clock Control Registers: [0]

PRCM Register Manual
- CAM_CM Registers: [1]

Table 4-226. CM_AUTOIDLE_CAM

Address Offset	0x0000 0030		
Physical Address	0x4800 4F30	**Instance**	CAM_CM
Description	This register controls the automatic control of the modules interface clock activity.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																															AUTO_CAM

Bits	Field Name	Description	Type	Reset
31:1	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x00000000
0	AUTO_CAM	Camera auto clock control. 0x0: Camera clock is unrelated to the domain state transition. 0x1: Camera clock is automatically enabled or disabled along with the domain state transition.	RW	0x0

Table 4-227. Register Call Summary for Register CM_AUTOIDLE_CAM

PRCM Clock Manager Functional Description
- CAM Power Domain Clock Controls: [0]

PRCM Basic Programming Model
- Power-Domain Clock Control Registers: [1]

PRCM Register Manual
- CAM_CM Registers: [2]

Table 4-228. CM_CLKSEL_CAM

Address Offset	0x0000 0040		
Physical Address	0x4800 4F40	**Instance**	CAM_CM
Description	CAM module clock selection.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																														CLKSEL_CAM	

Bits	Field Name	Description	Type	Reset
31:5	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x0000000
4:0	CLKSEL_CAM	CAM_MCLK divider factor DPLL4 M5 (1 up to 16); Other enums: Reserved 0x1: CAM_MCLK is DPLL4 clock divided by 1 0x2: CAM_MCLK is DPLL4 clock divided by 2 0x3: CAM_MCLK is DPLL4 clock divided by 3 0x4: CAM_MCLK is DPLL4 clock divided by 4 0x5: CAM_MCLK is DPLL4 clock divided by 5 0x6: CAM_MCLK is DPLL4 clock divided by 6 0x7: CAM_MCLK is DPLL4 clock divided by 7 0x8: CAM_MCLK is DPLL4 clock divided by 8 0x9: CAM_MCLK is DPLL4 clock divided by 9 0xA: CAM_MCLK is DPLL4 clock divided by 10 0xB: CAM_MCLK is DPLL4 clock divided by 11 0xC: CAM_MCLK is DPLL4 clock divided by 12 0xD: CAM_MCLK is DPLL4 clock divided by 13 0xE: CAM_MCLK is DPLL4 clock divided by 14 0xF: CAM_MCLK is DPLL4 clock divided by 15 0x10: CAM_MCLK is DPLL4 clock divided by 16	RW	0x10

Table 4-229. Register Call Summary for Register CM_CLKSEL_CAM

PRCM Basic Programming Model
- Power-Domain Clock Control Registers: [0]

PRCM Register Manual
- CAM_CM Registers: [1]

Table 4-230. CM_SLEEPDEP_CAM

Address Offset	0x0000 0044		
Physical Address	0x4800 4F44	**Instance**	CAM_CM
Description	This register allows enabling or disabling the sleep transition dependency of CAM domain with respect to other domain.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																														EN_MPU	RESERVED

Bits	Field Name	Description	Type	Reset
31:2	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x00000000
1	EN_MPU	MPU domain dependency 0x0: CAM domain sleep dependency with MPU domain is disabled. 0x1: CAM domain sleep dependency with MPU domain is enabled.	RW	0x0
0	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x0

Table 4-231. Register Call Summary for Register CM_SLEEPDEP_CAM

PRCM Basic Programming Model • Power-Domain Clock Control Registers: [0]
PRCM Register Manual • CAM_CM Registers: [1]

Table 4-232. CM_CLKSTCTRL_CAM

Address Offset	0x0000 0048		
Physical Address	0x4800 4F48	**Instance**	CAM_CM
Description	This register allows to enable or disable SW and HW supervised transition between ACTIVE and INACTIVE states.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																														CLKTRCTRL_CAM	

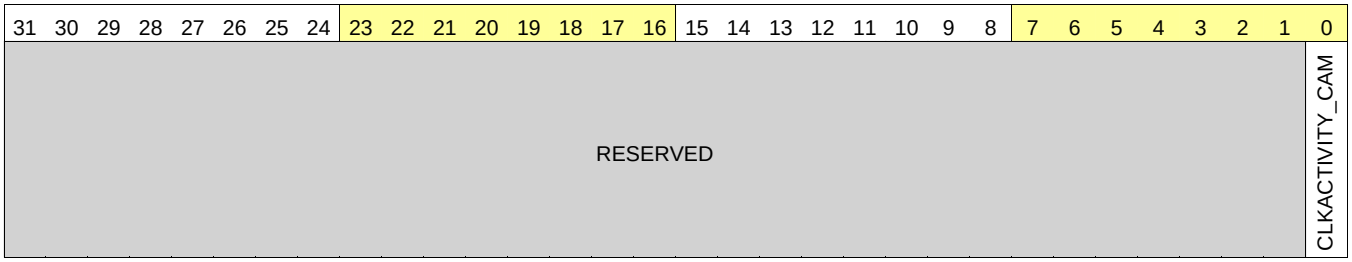
Bits	Field Name	Description	Type	Reset
31:2	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x00000000
1:0	CLKTRCTRL_CAM	Controls the clock state transition of the CAMERA clock domain. 0x0: Automatic transition is disabled 0x1: Start a software supervised sleep transition on the domain 0x2: Start a software supervised wake-up transition on the domain 0x3: Automatic transition is enabled. Transition is supervised by the hardware.	RW	0x0

Table 4-233. Register Call Summary for Register CM_CLKSTCTRL_CAM

PRCM Idle and Wake-Up Management • Device Wake-Up Events: [0] [1] [2]
PRCM Basic Programming Model • Power-Domain Clock Control Registers: [3]
PRCM Register Manual • CAM_CM Registers: [4]

Table 4-234. CM_CLKSTST_CAM

Address Offset	0x0000 004C		
Physical Address	0x4800 4F4C	**Instance**	CAM_CM
Description	This register provides a status on the OCP interface clock activity in the domain.		
Type	R		



Bits	Field Name	Description	Type	Reset
31:1	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x00000000
0	CLKACTIVITY_CAM	Interface clock activity status 0x0: No domain interface clock activity 0x1: Domain interface clock is active	R	0x0

Table 4-235. Register Call Summary for Register CM_CLKSTST_CAM

PRCM Basic Programming Model
- Power-Domain Clock Control Registers: [0]

PRCM Register Manual
- CAM_CM Registers: [1]

4.14.1.11 PER_CM Registers

Table 4-236. PER_CM Register Summary

Register Name	Type	Register Width (Bits)	Address Offset	Physical Address	Reset Type
CM_FCLKEN_PER	RW	32	0x0000 0000	0x4800 5000	W
CM_ICLKEN_PER	RW	32	0x0000 0010	0x4800 5010	W
CM_IDLEST_PER	R	32	0x0000 0020	0x4800 5020	C
CM_AUTOIDLE_PER	RW	32	0x0000 0030	0x4800 5030	W
CM_CLKSEL_PER	RW	32	0x0000 0040	0x4800 5040	W
CM_SLEEPDEP_PER	RW	32	0x0000 0044	0x4800 5044	W
CM_CLKSTCTRL_PER	RW	32	0x0000 0048	0x4800 5048	W
CM_CLKSTST_PER	R	32	0x0000 004C	0x4800 504C	C

Table 4-237. CM_FCLKEN_PER

Address Offset	0x0000 0000		
Physical Address	0x4800 5000	**Instance**	PER_CM
Description	Controls the modules functional clock activity.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED														EN_GPIO6	EN_GPIO5	EN_GPIO4	EN_GPIO3	EN_GPIO2	EN_WDT3	EN_UART3	EN_GPT9	EN_GPT8	EN_GPT7	EN_GPT6	EN_GPT5	EN_GPT4	EN_GPT3	EN_GPT2	EN_MCBSP4	EN_MCBSP3	EN_MCBSP2

Bits	Field Name	Description	Type	Reset
31:18	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x0000
17	EN_GPIO6	GPIO 6 functional clock control 0x0: GPIO 6 functional clock is disabled 0x1: GPIO 6 functional clock is enabled	RW	0x0
16	EN_GPIO5	GPIO 5 functional clock control 0x0: GPIO 5 functional clock is disabled 0x1: GPIO 5 functional clock is enabled	RW	0x0
15	EN_GPIO4	GPIO 4 functional clock control 0x0: GPIO 4 functional clock is disabled 0x1: GPIO 4 functional clock is enabled	RW	0x0
14	EN_GPIO3	GPIO 3 functional clock control 0x0: GPIO 3 functional clock is disabled 0x1: GPIO 3 functional clock is enabled	RW	0x0
13	EN_GPIO2	GPIO 2 functional clock control 0x0: GPIO 2 functional clock is disabled 0x1: GPIO 2 functional clock is enabled	RW	0x0
12	EN_WDT3	WDTIMER 3 functional clock control. 0x0: WDTIMER 3 functional clock is disabled 0x1: WDTIMER 3 functional clock is enabled	RW	0x0
11	EN_UART3	UART3 functional clock control. 0x0: UART 3 functional clock is disabled 0x1: UART 3 functional clock is enabled	RW	0x0
10	EN_GPT9	GPTIMER 9 functional clock control. 0x0: GPTIMER 9 functional clock is disabled 0x1: GPTIMER 9 functional clock is enabled	RW	0x0
9	EN_GPT8	GPTIMER 8 functional clock control. 0x0: GPTIMER 8 functional clock is disabled 0x1: GPTIMER 8 functional clock is enabled	RW	0x0
8	EN_GPT7	GPTIMER 7 functional clock control. 0x0: GPTIMER 7 functional clock is disabled 0x1: GPTIMER 7 functional clock is enabled	RW	0x0
7	EN_GPT6	GPTIMER 6 functional clock control. 0x0: GPTIMER 6 functional clock is disabled 0x1: GPTIMER 6 functional clock is enabled	RW	0x0
6	EN_GPT5	GPTIMER 5 functional clock control. 0x0: GPTIMER 5 functional clock is disabled 0x1: GPTIMER 5 functional clock is enabled	RW	0x0

Bits	Field Name	Description	Type	Reset
5	EN_GPT4	GPTIMER 4 functional clock control. 0x0: GPTIMER 4 functional clock is disabled 0x1: GPTIMER 4 functional clock is enabled	RW	0x0
4	EN_GPT3	GPTIMER 3 functional clock control. 0x0: GPTIMER 3 functional clock is disabled 0x1: GPTIMER 3 functional clock is enabled	RW	0x0
3	EN_GPT2	GPTIMER 2 functional clock control. 0x0: GPTIMER 2 functional clock is disabled 0x1: GPTIMER 2 functional clock is enabled	RW	0x0
2	EN_MCBSP4	McBSP 4 functional clock control. 0x0: McBSP 4 functional clock is disabled 0x1: McBSP 4 functional clock is enabled	RW	0x0
1	EN_MCBSP3	McBSP3 functional clock control. 0x0: McBSP 3 functional clock is disabled 0x1: McBSP 3 functional clock is enabled	RW	0x0
0	EN_MCBSP2	McBSP 2 functional clock control. 0x0: McBSP 2 functional clock is disabled 0x1: McBSP 2 functional clock is enabled	RW	0x0

Table 4-238. Register Call Summary for Register CM_FCLKEN_PER

PRCM Clock Manager Functional Description
- PRM Source-Clock Controls: [0] [1] [2]
- PER Power Domain Clock Controls: [3] [4] [5] [6] [7] [8] [9] [10] [11] [12] [13] [14]

PRCM Basic Programming Model
- Power-Domain Clock Control Registers: [15]

PRCM Register Manual
- PER_CM Registers: [16]

Table 4-239. CM_ICLKEN_PER

Address Offset	0x0000 0010		
Physical Address	0x4800 5010	**Instance**	PER_CM
Description	Controls the modules interface clock activity.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED														EN_GPIO6	EN_GPIO5	EN_GPIO4	EN_GPIO3	EN_GPIO2	EN_WDT3	EN_UART3	EN_GPT9	EN_GPT8	EN_GPT7	EN_GPT6	EN_GPT5	EN_GPT4	EN_GPT3	EN_GPT2	EN_MCBSP4	EN_MCBSP3	EN_MCBSP2

Bits	Field Name	Description	Type	Reset
31:18	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x0000
17	EN_GPIO6	GPIO 6 interface clock control 0x0: GPIO 6 interface clock is disabled 0x1: GPIO 6 interface clock is enabled	RW	0x0
16	EN_GPIO5	GPIO 5 interface clock control 0x0: GPIO 5 interface clock is disabled 0x1: GPIO 5 interface clock is enabled	RW	0x0

Bits	Field Name	Description	Type	Reset
15	EN_GPIO4	GPIO 4 interfface clock control 0x0: GPIO 4 interface clock is disabled 0x1: GPIO 4 interface clock is enabled	RW	0x0
14	EN_GPIO3	GPIO 3 interface clock control 0x0: GPIO 3 interface clock is disabled 0x1: GPIO 3 interface clock is enabled	RW	0x0
13	EN_GPIO2	GPIO 2 interface clock control 0x0: GPIO 2 interface clock is disabled 0x1: GPIO 2 interface clock is enabled	RW	0x0
12	EN_WDT3	WDTIMER 3 interface clock control. 0x0: WDTIMER 3 interface clock is disabled 0x1: WDTIMER 3 interface clock is enabled	RW	0x0
11	EN_UART3	UART3 interface clock control. 0x0: UART 3 interface clock is disabled 0x1: UART 3 interface clock is enabled	RW	0x0
10	EN_GPT9	GPTIMER 9 interface clock control. 0x0: GPTIMER 9 interface clock is disabled 0x1: GPTIMER 9 interface clock is enabled	RW	0x0
9	EN_GPT8	GPTIMER 8 interface clock control. 0x0: GPTIMER 8 interface clock is disabled 0x1: GPTIMER 8 interface clock is enabled	RW	0x0
8	EN_GPT7	GPTIMER 7 interface clock control. 0x0: GPTIMER 7 interface clock is disabled 0x1: GPTIMER 7 interface clock is enabled	RW	0x0
7	EN_GPT6	GPTIMER 6 interface clock control. 0x0: GPTIMER 6 interface clock is disabled 0x1: GPTIMER 6 interface clock is enabled	RW	0x0
6	EN_GPT5	GPTIMER 5 interface clock control. 0x0: GPTIMER 5 interface clock is disabled 0x1: GPTIMER 5 interface clock is enabled	RW	0x0
5	EN_GPT4	GPTIMER 4 interface clock control. 0x0: GPTIMER 4 interface clock is disabled 0x1: GPTIMER 4 interface clock is enabled	RW	0x0
4	EN_GPT3	GPTIMER 3 interface clock control. 0x0: GPTIMER 3 interface clock is disabled 0x1: GPTIMER 3 interface clock is enabled	RW	0x0
3	EN_GPT2	GPTIMER 2 interface clock control. 0x0: GPTIMER 2 interface clock is disabled 0x1: GPTIMER 2 interface clock is enabled	RW	0x0
2	EN_MCBSP4	McBSP 4 interface clock control. 0x0: McBSP 4 interface clock is disabled 0x1: McBSP 4 interface clock is enabled	RW	0x0
1	EN_MCBSP3	McBSP 3 interface clock control. 0x0: McBSP 3 interface clock is disabled 0x1: McBSP 3 interface clock is enabled	RW	0x0
0	EN_MCBSP2	McBSP 2 interface clock control. 0x0: McBSP 2 interface clock is disabled 0x1: McBSP 2 interface clock is enabled	RW	0x0

Table 4-240. Register Call Summary for Register CM_ICLKEN_PER

PRCM Clock Manager Functional Description • PER Power Domain Clock Controls: [0]
PRCM Basic Programming Model • Power-Domain Clock Control Registers: [1]
PRCM Register Manual • PER_CM Registers: [2]

Table 4-241. CM_IDLEST_PER

Address Offset	0x0000 0020		
Physical Address	0x4800 5020	**Instance**	PER_CM
Description	Modules access availability monitoring. This register is read only and automatically updated.		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED														ST_GPIO6	ST_GPIO5	ST_GPIO4	ST_GPIO3	ST_GPIO2	ST_WDT3	ST_UART3	ST_GPT9	ST_GPT8	ST_GPT7	ST_GPT6	ST_GPT5	ST_GPT4	ST_GPT3	ST_GPT2	ST_MCBSP4	ST_MCBSP3	ST_MCBSP2

Bits	Field Name	Description	Type	Reset
31:18	RESERVED	Read returns 0.	R	0x0000
17	ST_GPIO6	GPIO 6 idle status 0x0: GPIO 6 can be accessed. 0x1: GPIO 6 cannot be accessed. Any access may return an error.	R	0x1
16	ST_GPIO5	GPIO 5 idle status 0x0: GPIO 5 can be accessed. 0x1: GPIO 5 cannot be accessed. Any access may return an error.	R	0x1
15	ST_GPIO4	GPIO 4 idle status 0x0: GPIO 4 can be accessed. 0x1: GPIO 4 cannot be accessed. Any access may return an error.	R	0x1
14	ST_GPIO3	GPIO 3 idle status 0x0: GPIO 3 can be accessed. 0x1: GPIO 3 cannot be accessed. Any access may return an error.	R	0x1
13	ST_GPIO2	GPIO 2 idle status 0x0: GPIO 2 can be accessed. 0x1: GPIO 2 cannot be accessed. Any access may return an error.	R	0x1
12	ST_WDT3	WDTIMER 3 idle status. 0x0: WDTIMER 3 can be accessed. 0x1: WDTIMER 3 cannot be accessed. Any access may return an error.	R	0x1
11	ST_UART3	UART3 idle status. 0x0: UART 3 can be accessed. 0x1: UART 3 cannot be accessed. Any access may return an error.	R	0x1

Bits	Field Name	Description	Type	Reset
10	ST_GPT9	GPTIMER 9 idle status. 0x0: GPTIMER 9 can be accessed. 0x1: GPTIMER 9 cannot be accessed. Any access may return an error.	R	0x1
9	ST_GPT8	GPTIMER 8 idle status. 0x0: GPTIMER 8 can be accessed. 0x1: GPTIMER 8 cannot be accessed. Any access may return an error.	R	0x1
8	ST_GPT7	GPTIMER 7 idle status. 0x0: GPTIMER 7 can be accessed. 0x1: GPTIMER 7 cannot be accessed. Any access may return an error.	R	0x1
7	ST_GPT6	GPTIMER 6 idle status. 0x0: GPTIMER 6 can be accessed. 0x1: GPTIMER 6 cannot be accessed. Any access may return an error.	R	0x1
6	ST_GPT5	GPTIMER 5 idle status. 0x0: GPTIMER 5 can be accessed. 0x1: GPTIMER 5 cannot be accessed. Any access may return an error.	R	0x1
5	ST_GPT4	GPTIMER 4 idle status. 0x0: GPTIMER 4 can be accessed. 0x1: GPTIMER 4 cannot be accessed. Any access may return an error.	R	0x1
4	ST_GPT3	GPTIMER 3 idle status. 0x0: GPTIMER 3 can be accessed. 0x1: GPTIMER 3 cannot be accessed. Any access may return an error.	R	0x1
3	ST_GPT2	GPTIMER 2 idle status. 0x0: GPTIMER 2 can be accessed. 0x1: GPTIMER 2 cannot be accessed. Any access may return an error.	R	0x1
2	ST_MCBSP4	McBSP 4 idle status. 0x0: McBSP 4 can be accessed. 0x1: McBSP 4 cannot be accessed. Any access may return an error.	R	0x1
1	ST_MCBSP3	McBSP 3 idle status. 0x0: McBSP 3 can be accessed. 0x1: McBSP 3 cannot be accessed. Any access may return an error.	R	0x1
0	ST_MCBSP2	McBSP 2 idle status. 0x0: McBSP 2 can be accessed. 0x1: McBSP 2 cannot be accessed. Any access may return an error.	R	0x1

Table 4-242. Register Call Summary for Register CM_IDLEST_PER

PRCM Basic Programming Model • Power-Domain Clock Control Registers: [0]
PRCM Register Manual • PER_CM Registers: [1]

Table 4-243. CM_AUTOIDLE_PER

Address Offset	0x0000 0030		
Physical Address	0x4800 5030	**Instance**	PER_CM
Description	This register controls the automatic control of the modules interface clock activity.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED														AUTO_GPIO6	AUTO_GPIO5	AUTO_GPIO4	AUTO_GPIO3	AUTO_GPIO2	AUTO_WDT3	AUTO_UART3	AUTO_GPT9	AUTO_GPT8	AUTO_GPT7	AUTO_GPT6	AUTO_GPT5	AUTO_GPT4	AUTO_GPT3	AUTO_GPT2	AUTO_MCBSP4	AUTO_MCBSP3	AUTO_MCBSP2

Bits	Field Name	Description	Type	Reset
31:18	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x0000
17	AUTO_GPIO6	GPIO 6 auto clock control 0x0: GPIO 6 interface clock is unrelated to the domain state transition. 0x1: GPIO 6 interface clock is automatically enabled or disabled along with the domain state transition.	RW	0x0
16	AUTO_GPIO5	GPIO 5 auto clock control 0x0: GPIO 5 interface clock is unrelated to the domain state transition. 0x1: GPIO 5 interface clock is automatically enabled or disabled along with the domain state transition.	RW	0x0
15	AUTO_GPIO4	GPIO 4 auto clock control 0x0: GPIO 4 interface clock is unrelated to the domain state transition. 0x1: GPIO 4 interface clock is automatically enabled or disabled along with the domain state transition.	RW	0x0
14	AUTO_GPIO3	GPIO 3 auto clock control 0x0: GPIO 3 interface clock is unrelated to the domain state transition. 0x1: GPIO 3 interface clock is automatically enabled or disabled along with the domain state transition.	RW	0x0
13	AUTO_GPIO2	GPIO 2 auto clock control 0x0: GPIO 2 interface clock is unrelated to the domain state transition. 0x1: GPIO 2 interface clock is automatically enabled or disabled along with the domain state transition.	RW	0x0
12	AUTO_WDT3	WDTIMER 3 auto clock control. 0x0: WDTIMER 3 interface clock is unrelated to the domain state transition. 0x1: WDTIMER 3 interface clock is automatically enabled or disabled along with the domain state transition.	RW	0x0
11	AUTO_UART3	UART3 auto clock control. 0x0: UART 3 interface clock is unrelated to the domain state transition. 0x1: UART 3 interface clock is automatically enabled or disabled along with the domain state transition.	RW	0x0
10	AUTO_GPT9	GPTIMER 9 auto clock control. 0x0: GPTIMER 9 interface clock is unrelated to the domain state transition. 0x1: GPTIMER 9 interface clock is automatically enabled or disabled along with the domain state transition.	RW	0x0

Bits	Field Name	Description	Type	Reset
9	AUTO_GPT8	GPTIMER 8 auto clock control. 0x0: GPTIMER 8 interface clock is unrelated to the domain state transition. 0x1: GPTIMER 8 interface clock is automatically enabled or disabled along with the domain state transition.	RW	0x0
8	AUTO_GPT7	GPTIMER 7 auto clock control. 0x0: GPTIMER 7 interface clock is unrelated to the domain state transition. 0x1: GPTIMER 7 interface clock is automatically enabled or disabled along with the domain state transition.	RW	0x0
7	AUTO_GPT6	GPTIMER 6 auto clock control. 0x0: GPTIMER 6 interface clock is unrelated to the domain state transition. 0x1: GPTIMER 6 interface clock is automatically enabled or disabled along with the domain state transition.	RW	0x0
6	AUTO_GPT5	GPTIMER 5 auto clock control. 0x0: GPTIMER 5 interface clock is unrelated to the domain state transition. 0x1: GPTIMER 5 interface clock is automatically enabled or disabled along with the domain state transition.	RW	0x0
5	AUTO_GPT4	GPTIMER 4 auto clock control. 0x0: GPTIMER 4 interface clock is unrelated to the domain state transition. 0x1: GPTIMER 4 interface clock is automatically enabled or disabled along with the domain state transition.	RW	0x0
4	AUTO_GPT3	GPTIMER 3 auto clock control. 0x0: GPTIMER 3 interface clock is unrelated to the domain state transition. 0x1: GPTIMER 3 interface clock is automatically enabled or disabled along with the domain state transition.	RW	0x0
3	AUTO_GPT2	GPTIMER 2 auto clock control. 0x0: GPTIMER 2 interface clock is unrelated to the domain state transition. 0x1: GPTIMER 2 interface clock is automatically enabled or disabled along with the domain state transition.	RW	0x0
2	AUTO_MCBSP4	McBSP 4 auto clock control. 0x0: McBSP 4 interface clock is unrelated to the domain state transition. 0x1: McBSP 4 interface clock is automatically enabled or disabled along with the domain state transition.	RW	0x0
1	AUTO_MCBSP3	McBSP 3 auto clock control. 0x0: McBSP 3 interface clock is unrelated to the domain state transition. 0x1: McBSP 3 interface clock is automatically enabled or disabled along with the domain state transition.	RW	0x0
0	AUTO_MCBSP2	McBSP 2 auto clock control. 0x0: McBSP 2 interface clock is unrelated to the domain state transition. 0x1: McBSP 2 interface clock is automatically enabled or disabled along with the domain state transition.	RW	0x0

Table 4-244. Register Call Summary for Register CM_AUTOIDLE_PER

PRCM Clock Manager Functional Description

- PER Power Domain Clock Controls: [0]

Table 4-244. Register Call Summary for Register CM_AUTOIDLE_PER (continued)

PRCM Basic Programming Model • Power-Domain Clock Control Registers: [1]
PRCM Register Manual • PER_CM Registers: [2]

Table 4-245. CM_CLKSEL_PER

Address Offset	0x0000 0040		
Physical Address	0x4800 5040	**Instance**	PER_CM
Description	PER domain modules source clock selection.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																								CLKSEL_GPT9	CLKSEL_GPT8	CLKSEL_GPT7	CLKSEL_GPT6	CLKSEL_GPT5	CLKSEL_GPT4	CLKSEL_GPT3	CLKSEL_GPT2

Bits	Field Name	Description	Type	Reset
31:8	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x000000
7	CLKSEL_GPT9	Selects GPTIMER 9 source clock 0x0: source is 32K_FCLK 0x1: source is SYS_CLK	RW	0x0
6	CLKSEL_GPT8	Selects GPTIMER 8 source clock 0x0: source is 32K_FCLK 0x1: source is SYS_CLK	RW	0x0
5	CLKSEL_GPT7	Selects GPTIMER 7 source clock 0x0: source is 32K_FCLK 0x1: source is SYS_CLK	RW	0x0
4	CLKSEL_GPT6	Selects GPTIMER 6 source clock 0x0: source is 32K_FCLK 0x1: source is SYS_CLK	RW	0x0
3	CLKSEL_GPT5	Selects GPTIMER 5 source clock 0x0: source is 32K_FCLK 0x1: source is SYS_CLK	RW	0x0
2	CLKSEL_GPT4	Selects GPTIMER 4 source clock 0x0: source is 32K_FCLK 0x1: source is SYS_CLK	RW	0x0
1	CLKSEL_GPT3	Selects GPTIMER 3 source clock 0x0: source is 32K_FCLK 0x1: source is SYS_CLK	RW	0x0
0	CLKSEL_GPT2	Selects GPTIMER 2 source clock 0x0: source is 32K_FCLK 0x1: source is SYS_CLK	RW	0x0

Table 4-246. Register Call Summary for Register CM_CLKSEL_PER

PRCM Basic Programming Model • Power-Domain Clock Control Registers: [0]
PRCM Register Manual • PER_CM Registers: [1]

Table 4-247. CM_SLEEPDEP_PER

Address Offset	0x0000 0044		
Physical Address	0x4800 5044	**Instance**	PER_CM
Description	This register allows enabling or disabling the sleep transition dependency of PER domain with respect to other domain.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																													EN_IVA2	EN_MPU	RESERVED

Bits	Field Name	Description	Type	Reset
31:3	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x00000000
2	EN_IVA2	IVA2 domain dependency 0x0: PER domain sleep dependency with IVA2 domain is disabled. 0x1: PER domain sleep dependency with IVA2 domain is enabled.	RW	0x0
1	EN_MPU	MPU domain dependency 0x0: PER domain sleep dependency with MPU domain is disabled. 0x1: PER domain sleep dependency with MPU domain is enabled.	RW	0x0
0	RESERVED	Write 0s for future compatibility. Read returns 0.	RW	0x0

Table 4-248. Register Call Summary for Register CM_SLEEPDEP_PER

PRCM Basic Programming Model • Power-Domain Clock Control Registers: [0]
PRCM Register Manual • PER_CM Registers: [1]

Table 4-249. CM_CLKSTCTRL_PER

Address Offset	0x0000 0048		
Physical Address	0x4800 5048	**Instance**	PER_CM
Description	This register enables the domain power state transition. It controls the HW supervised domain power state transition between ACTIVE and INACTIVE states.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																														CLKTRCTRL_PER	

Bits	Field Name	Description	Type	Reset
31:2	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x00000000
1:0	CLKTRCTRL_PER	Controls the clock state transition of the PERIPHERAL clock domain. 0x0: Automatic transition is disabled 0x1: Start a software supervised sleep transition on the domain 0x2: Start a software supervised wake-up transition on the domain 0x3: Automatic transition is enabled. Transition is supervised by the HardWare.	RW	0x0

Table 4-250. Register Call Summary for Register CM_CLKSTCTRL_PER

PRCM Idle and Wake-Up Management • Device Wake-Up Events: [0] [1] [2]
PRCM Basic Programming Model • Power-Domain Clock Control Registers: [3]
PRCM Register Manual • PER_CM Registers: [4]

Table 4-251. CM_CLKSTST_PER

Address Offset	0x0000 004C		
Physical Address	0x4800 504C	**Instance**	PER_CM
Description	This register provides a status on the OCP interface clock activity in the domain.		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																															CLKACTIVITY_PER

Bits	Field Name	Description	Type	Reset
31:1	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x00000000
0	CLKACTIVITY_PER	Interface clock activity status 0x0: No domain interface clock activity 0x1: Domain interface clock is active	R	0x0

Table 4-252. Register Call Summary for Register CM_CLKSTST_PER

PRCM Basic Programming Model • Power-Domain Clock Control Registers: [0]

Table 4-252. Register Call Summary for Register CM_CLKSTST_PER (continued)

PRCM Register Manual • PER_CM Registers: [1]

4.14.1.12 EMU_CM Registers

Table 4-253. EMU_CM Register Summary

Register Name	Type	Register Width (Bits)	Address Offset	Physical Address	Reset Type
CM_CLKSEL1_EMU	RW	32	0x0000 0040	0x4800 5140	W
CM_CLKSTCTRL_EMU	RW	32	0x0000 0048	0x4800 5148	C
CM_CLKSTST_EMU	R	32	0x0000 004C	0x4800 514C	C
CM_CLKSEL2_EMU	RW	32	0x0000 0050	0x4800 5150	W
CM_CLKSEL3_EMU	RW	32	0x0000 0054	0x4800 5154	W

Table 4-254. CM_CLKSEL1_EMU

Address Offset	0x0000 0040		
Physical Address	0x4800 5140	**Instance**	EMU_CM
Description	Modules clock selection.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED				DIV_DPLL4				RESERVED			DIV_DPLL3					RESERVED		CLKSEL_TRACECLK			CLKSEL_PCLK			CLKSEL_PCLKX2			CLKSEL_ATCLK			TRACE_MUX_CTRL			MUX_CTRL	

Bits	Field Name	Description	Type	Reset
31:29	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x0
28:24	DIV_DPLL4	DPLL4 M6 clock divider factor (1 up to 16); Other enums: Reserved 0x1: EMU_PER_ALWON_CLK is DPLL4 clock divided by 1 0x2: EMU_PER_ALWON_CLK is DPLL4 clock divided by 2 0x3: EMU_PER_ALWON_CLK is DPLL4 clock divided by 3 0x4: EMU_PER_ALWON_CLK is DPLL4 clock divided by 4 0x5: EMU_PER_ALWON_CLK is DPLL4 clock divided by 5 0x6: EMU_PER_ALWON_CLK is DPLL4 clock divided by 6 0x7: EMU_PER_ALWON_CLK is DPLL4 clock divided by 7 0x8: EMU_PER_ALWON_CLK is DPLL4 clock divided by 8 0x9: EMU_PER_ALWON_CLK is DPLL4 clock divided by 9 0xA: EMU_PER_ALWON_CLK is DPLL4 clock divided by 10 0xB: EMU_PER_ALWON_CLK is DPLL4 clock divided by 11 0xC: EMU_PER_ALWON_CLK is DPLL4 clock divided by 12 0xD: EMU_PER_ALWON_CLK is DPLL4 clock divided by 13 0xE: EMU_PER_ALWON_CLK is DPLL4 clock divided by 14 0xF: EMU_PER_ALWON_CLK is DPLL4 clock divided by 15 0x10: EMU_PER_ALWON_CLK is DPLL4 clock divided by 16	RW	0x10
23:21	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x0

Bits	Field Name	Description	Type	Reset
20:16	DIV_DPLL3	DPLL3_M3X2 clock divider factor (1 up to 16); Other enums: Reserved 0x1: EMU_CORE_ALWON_CLK is DPLL3 clock divided by 1 0x2: EMU_CORE_ALWON_CLK is DPLL3 clock divided by 2 0x3: EMU_CORE_ALWON_CLK is DPLL3 clock divided by 3 0x4: EMU_CORE_ALWON_CLK is DPLL3 clock divided by 4 0x5: EMU_CORE_ALWON_CLK is DPLL3 clock divided by 5 0x6: EMU_CORE_ALWON_CLK is DPLL3 clock divided by 6 0x7: EMU_CORE_ALWON_CLK is DPLL3 clock divided by 7 0x8: EMU_CORE_ALWON_CLK is DPLL3 clock divided by 8 0x9: EMU_CORE_ALWON_CLK is DPLL3 clock divided by 9 0xA: EMU_CORE_ALWON_CLK is DPLL3 clock divided by 10 0xB: EMU_CORE_ALWON_CLK is DPLL3 clock divided by 11 0xC: EMU_CORE_ALWON_CLK is DPLL3 clock divided by 12 0xD: EMU_CORE_ALWON_CLK is DPLL3 clock divided by 13 0xE: EMU_CORE_ALWON_CLK is DPLL3 clock divided by 14 0xF: EMU_CORE_ALWON_CLK is DPLL3 clock divided by 15 0x10: EMU_CORE_ALWON_CLK is DPLL3 clock divided by 16	RW	0x10
15:14	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x0
13:11	CLKSEL_TRACECLK	Selects the TRACE clock; Other enums: Reserved 0x1: TRACECLK.FCLK is the selected TRACE source clock divided by 1 0x2: TRACECLK.FCLK is the selected TRACE source clock divided by 2 0x4: TRACECLK.FCLK is the selected TRACE source clock divided by 4	RW	0x1
10:8	CLKSEL_PCLK	Selects the PCLK clock; Other enums: Reserved 0x2: PCLK.FCLK is the selected PCLK source clock divided by 2 0x3: PCLK.FCLK is the selected PCLK source clock divided by 3 0x4: PCLK.FCLK is the selected PCLK source clock divided by 4 0x6: PCLK.FCLK is the selected PCLK source clock divided by 6	RW	0x2
7:6	CLKSEL_PCLKX2	Selects the PCLKx2 clock; Other enums: Reserved 0x1: PCLKx2.FCLK is the selected PCLK source clock divided by 1 0x2: PCLKx2.FCLK is the selected PCLK source clock divided by 2 0x3: PCLKx2.FCLK is the selected PCLK source clock divided by 3	RW	0x1

Bits	Field Name	Description	Type	Reset
5:4	CLKSEL_ATCLK	Selects the ATCLK clock; Other enums: Reserved 0x1: ATCLK.FCLK is the selected ATCLK source clock divided by 1 0x2: ATCLK.FCLK is the selected ATCLK source clock divided by 2	RW	0x1
3:2	TRACE_MUX_CTRL	Selection of TRACECLK.FCLK source clock 0x0: TRACE source clock is SYS_CLK 0x1: TRACE source clock is EMU_CORE_ALWON_CLK 0x2: TRACE source clock is EMU_PER_ALWON clock 0x3: TRACE source clock is EMU_MPU_ALWON clock	RW	0x0
1:0	MUX_CTRL	Selection of ATCLK.FCLK, PCLK.FCLK and PCLKx2.FCLK source clock 0x0: ATCLK, PCLK and PCLKx2 source clock is SYS_CLK 0x1: ATCLK, PCLK and PCLKx2 source clock is EMU_CORE_ALWON_CLK 0x2: ATCLK, PCLK and PCLKx2 source clock is EMU_PER_ALWON clock 0x3: ATCLK, PCLK and PCLKx2 source clock is EMU_MPU_ALWON_CLK	RW	0x0

Table 4-255. Register Call Summary for Register CM_CLKSEL1_EMU

PRCM Register Manual
- EMU_CM Registers: [0]

Table 4-256. CM_CLKSTCTRL_EMU

Address Offset	0x0000 0048		
Physical Address	0x4800 5148	**Instance**	EMU_CM
Description	This register allows to enable or disable SW and HW supervised transition between ACTIVE and INACTIVE states.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																														CLKTRCTRL_EMU	

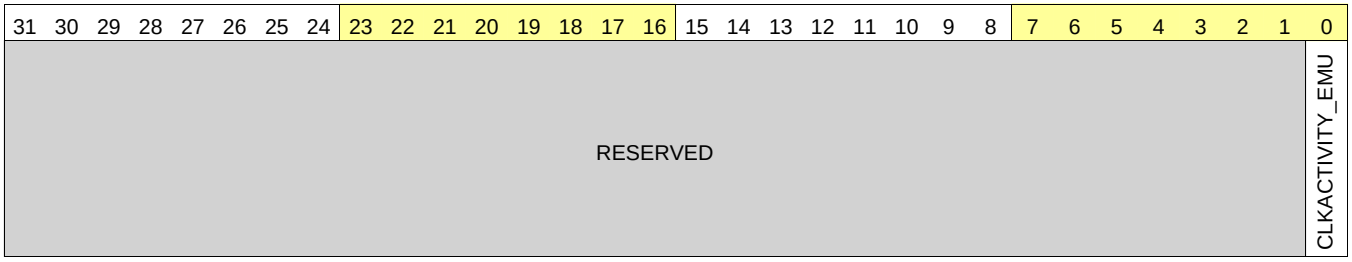
Bits	Field Name	Description	Type	Reset
31:2	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x00000000
1:0	CLKTRCTRL_EMU	Controls the clock state transition of the EMULATION clock domain. 0x0: Reserved 0x1: Start a software supervised sleep transition on the domain 0x2: Start a software supervised wake-up transition on the domain or maintain emulation domain active. (force wakeup has to be kept asserted to keep Emulation domain ON) 0x3: Automatic transition is enabled. Transition is supervised by the HardWare.	RW	0x2

Table 4-257. Register Call Summary for Register CM_CLKSTCTRL_EMU

PRCM Idle and Wake-Up Management • Device Wake-Up Events: [0] [1] [2]
PRCM Basic Programming Model • Power-Domain Clock Control Registers: [3]
PRCM Register Manual • EMU_CM Registers: [4]

Table 4-258. CM_CLKSTST_EMU

Address Offset	0x0000 004C		
Physical Address	0x4800 514C	**Instance**	EMU_CM
Description	This register provides a status on the clock activity in the domain (depends on the selected source clock).		
Type	R		



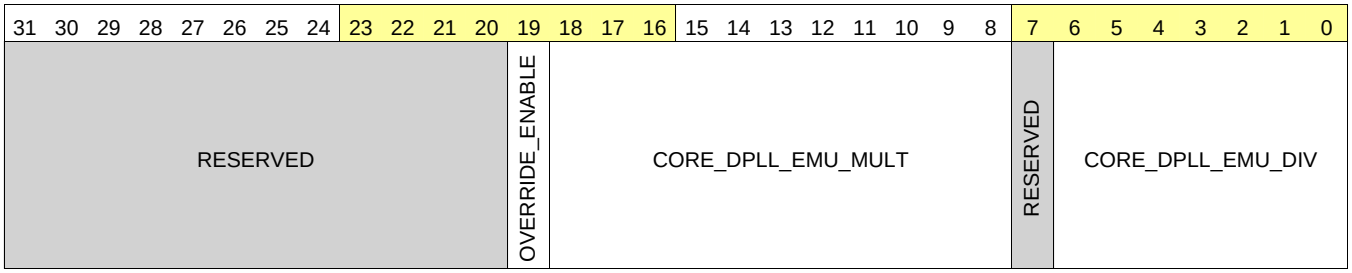
Bits	Field Name	Description	Type	Reset
31:1	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x00000000
0	CLKACTIVITY_EMU	Clock activity status (depends on the selected source clock) 0x0: No domain clock activity 0x1: Domain clock is active	R	0x0

Table 4-259. Register Call Summary for Register CM_CLKSTST_EMU

PRCM Basic Programming Model • Power-Domain Clock Control Registers: [0]
PRCM Register Manual • EMU_CM Registers: [1]

Table 4-260. CM_CLKSEL2_EMU

Address Offset	0x0000 0050		
Physical Address	0x4800 5150	**Instance**	EMU_CM
Description	This register provides override controls over the DPLL3.		
Type	RW		



Bits	Field Name	Description	Type	Reset
31:20	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x000
19	OVERRIDE_ENABLE	This bit allows to enable or disable the emulation override controls 0x0: The emulation override controls are disabled 0x1: The emulation override controls are enabled	RW	0x0
18:8	CORE_DPLL_EMU_MULT	DPLL3 override multiplier factor (0 to 2047)	RW	0x000
7	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x0
6:0	CORE_DPLL_EMU_DIV	DPLL3 override divider factor (0 to 127)	RW	0x00

Table 4-261. Register Call Summary for Register CM_CLKSEL2_EMU

PRCM Register Manual
- EMU_CM Registers: [0]

Table 4-262. CM_CLKSEL3_EMU

Address Offset	0x0000 0054		
Physical Address	0x4800 5154	**Instance**	EMU_CM
Description	This register provides override controls over the PERIPHERAL DPLL.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED												OVERRIDE_ENABLE	PERIPH_DPLL_EMU_MULT											RESERVED	PERIPH_DPLL_EMU_DIV						

Bits	Field Name	Description	Type	Reset
31:20	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x000
19	OVERRIDE_ENABLE	This bit allows to enable or disable the emulation override controls 0x0: The emulation override controls are disabled 0x1: The emulation override controls are enabled	RW	0x0
18:8	PERIPH_DPLL_EMU_MULT	DPLL4 override multiplier factor (0 to 2047)	RW	0x000
7	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x0
6:0	PERIPH_DPLL_EMU_DIV	DPLL4 override divider factor (0 to 127)	RW	0x00

Table 4-263. Register Call Summary for Register CM_CLKSEL3_EMU

PRCM Register Manual
- EMU_CM Registers: [0]

4.14.1.13 Global_Reg_CM Registers

Table 4-264. Global_Reg_CM Register Summary

Register Name	Type	Register Width (Bits)	Address Offset	Physical Address	Reset Type
CM_POLCTRL	RW	32	0x0000 009C	0x4800 529C	C

Table 4-265. CM_POLCTRL

Address Offset	0x0000 009C		
Physical Address	0x4800 529C	**Instance**	Global_Reg_CM
Description	This register allows setting the polarity of device outputs control signals.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																															CLKOUT2_POL

Bits	Field Name	Description	Type	Reset
31:1	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x00000000
0	CLKOUT2_POL	Controls the external output clock 2 polarity when disabled 0x0: sys_clkout2 is gated low when inactive 0x1: sys_clkout2 is gated high when inactive	RW	0x0

Table 4-266. Register Call Summary for Register CM_POLCTRL

PRCM Clock Manager Functional Description • External Output Clock2 (sys_clkout2) Control: [0]
PRCM Basic Programming Model • Output Signal Polarity Control Registers: [1]
PRCM Register Manual • Global_Reg_CM Registers: [2]

4.14.1.14 NEON_CM Registers

Table 4-267. NEON_CM Register Summary

Register Name	Type	Register Width (Bits)	Address Offset	Physical Address	Reset Type
CM_IDLEST_NEON	R	32	0x0000 0020	0x4800 5320	C
CM_CLKSTCTRL_NEON	RW	32	0x0000 0048	0x4800 5348	W

Table 4-268. CM_IDLEST_NEON

Address Offset	0x0000 0020		
Physical Address	0x4800 5320	**Instance**	NEON_CM
Description	Modules access availability monitoring. This register is read only and automatically updated.		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																															ST_NEON

Bits	Field Name	Description	Type	Reset
31:1	RESERVED	Read returns 0.	R	0x00000000
0	ST_NEON	NEON standby status. 0x0: NEON is active 0x1: NEON is in standby mode	R	0x1

Table 4-269. Register Call Summary for Register CM_IDLEST_NEON

PRCM Basic Programming Model • Power-Domain Clock Control Registers: [0]
PRCM Register Manual • NEON_CM Registers: [1]

Table 4-270. CM_CLKSTCTRL_NEON

Address Offset	0x0000 0048		
Physical Address	0x4800 5348	**Instance**	NEON_CM
Description	This register enables the domain power state transition. It controls the HW supervised domain power state transition between ACTIVE and INACTIVE states.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																														CLKTRCTRL_NEON	

Bits	Field Name	Description	Type	Reset
31:2	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x00000000
1:0	CLKTRCTRL_NEON	Controls the clock state transition of the NEON clock domain. 0x0: Automatic transition is disabled 0x1: Start a software supervised sleep transition on the domain 0x2: Start a software supervised wake-up transition on the domain 0x3: Automatic transition is enabled. Transition is supervised by the HardWare.	RW	0x0

Table 4-271. Register Call Summary for Register CM_CLKSTCTRL_NEON

PRCM Idle and Wake-Up Management • Device Wake-Up Events: [0] [1] [2]
PRCM Basic Programming Model • Power-Domain Clock Control Registers: [3]
PRCM Register Manual • NEON_CM Registers: [4]

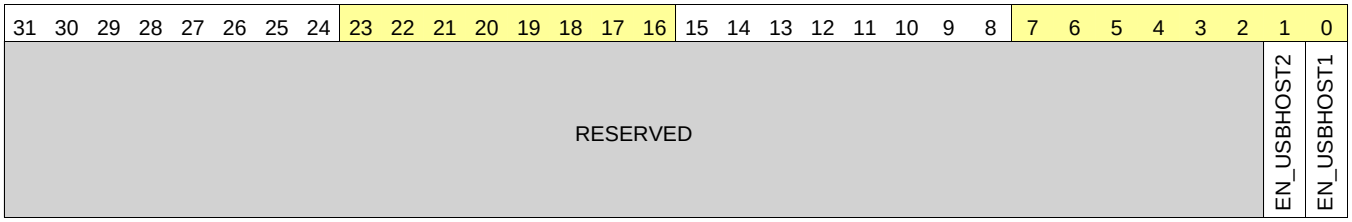
4.14.1.15 USBHOST_CM Registers

Table 4-272. USBHOST_CM Register Summary

Register Name	Type	Register Width (Bits)	Address Offset	Physical Address	Reset Type
CM_FCLKEN_USBHOST	RW	32	0x0000 0000	0x4800 5400	W
CM_ICLKEN_USBHOST	RW	32	0x0000 0010	0x4800 5410	W
CM_IDLEST_USBHOST	R	32	0x0000 0020	0x4800 5420	C
CM_AUTOIDLE_USBHOST	RW	32	0x0000 0030	0x4800 5430	W
CM_SLEEPDEP_USBHOST	RW	32	0x0000 0044	0x4800 5444	W
CM_CLKSTCTRL_USBHOST	RW	32	0x0000 0048	0x4800 5448	W
CM_CLKSTST_USBHOST	R	32	0x0000 004C	0x4800 544C	C

Table 4-273. CM_FCLKEN_USBHOST

Address Offset	0x0000 0000		
Physical Address	0x4800 5400	**Instance**	USBHOST_CM
Description	Controls the modules functional clock activity.		
Type	RW		



Bits	Field Name	Description	Type	Reset
31:2	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x00000000
1	EN_USBHOST2	USB HOST 120-MHz functional clock control 0x0: USBHOST_120M_FCLK is disabled 0x1: USBHOST_120M_FCLK is enabled	RW	0x0
0	EN_USBHOST1	USB HOST 48-MHz functional clock control 0x0: USBHOST_48M_FCLK is disabled 0x1: USBHOST_48M_FCLK is enabled	RW	0x0

Table 4-274. Register Call Summary for Register CM_FCLKEN_USBHOST

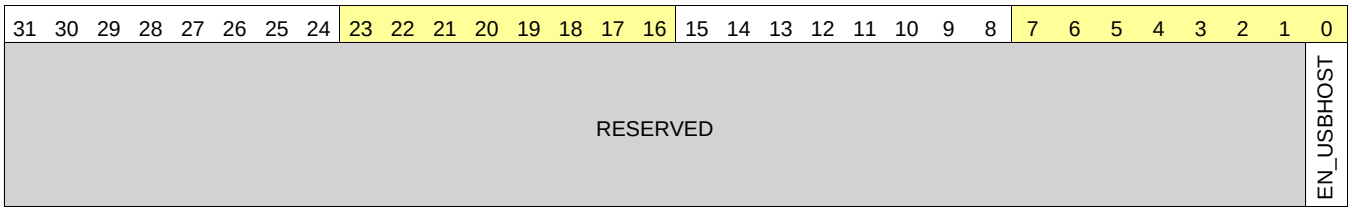
PRCM Clock Manager Functional Description
- Power Domain Clock Distribution: [0] [1]
- CM Source-Clock Controls: [2]
- USBHOST Power Domain Clock Controls: [3] [4]

PRCM Basic Programming Model
- Power-Domain Clock Control Registers: [5]

PRCM Register Manual
- USBHOST_CM Registers: [6]

Table 4-275. CM_ICLKEN_USBHOST

Address Offset	0x0000 0010		
Physical Address	0x4800 5410	**Instance**	USBHOST_CM
Description	Controls the modules interface clock activity.		
Type	RW		



Bits	Field Name	Description	Type	Reset
31:1	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x00000000
0	EN_USBHOST	USB HOST interface clock control 0x0: USB HOST interface clock is disabled 0x1: USB HOST interface clock is enabled	RW	0x0

Table 4-276. Register Call Summary for Register CM_ICLKEN_USBHOST

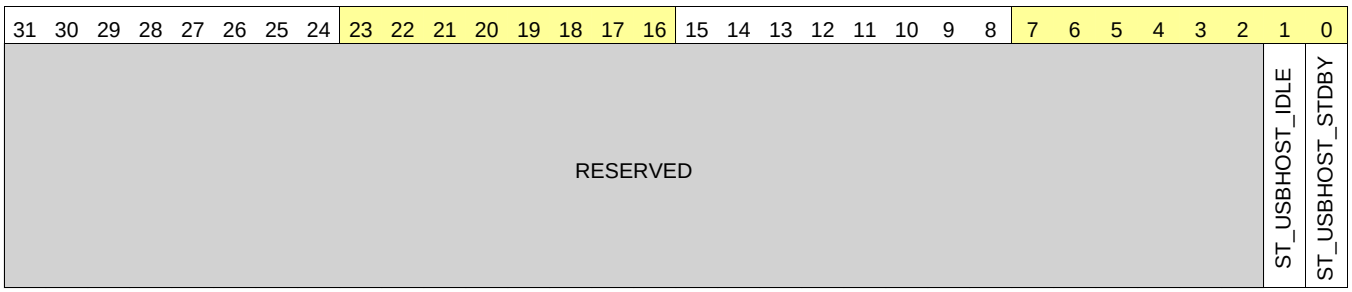
PRCM Clock Manager Functional Description
- USBHOST Power Domain Clock Controls: [0]

PRCM Basic Programming Model
- Power-Domain Clock Control Registers: [1]

PRCM Register Manual
- USBHOST_CM Registers: [2]

Table 4-277. CM_IDLEST_USBHOST

Address Offset	0x0000 0020		
Physical Address	0x4800 5420	**Instance**	USBHOST_CM
Description	Modules access availability monitoring. This register is read only and automatically updated.		
Type	R		



Bits	Field Name	Description	Type	Reset
31:2	RESERVED	Read returns 0.	R	0x00000000
1	ST_USBHOST_IDLE	USB HOST idle status. 0x0: USB HOST is active. 0x1: USB HOST is in idle mode and cannot be accessed.	R	0x1
0	ST_USBHOST_STDBY	USB HOST standby status. 0x0: USB HOST is active. 0x1: USB HOST is in standby mode.	R	0x1

Table 4-278. Register Call Summary for Register CM_IDLEST_USBHOST

PRCM Basic Programming Model • Power-Domain Clock Control Registers: [0]
PRCM Register Manual • USBHOST_CM Registers: [1]

Table 4-279. CM_AUTOIDLE_USBHOST

Address Offset	0x0000 0030		
Physical Address	0x4800 5430	**Instance**	USBHOST_CM
Description	This register controls the automatic control of the modules interface clock activity.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																															AUTO_USBHOST

Bits	Field Name	Description	Type	Reset
31:1	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x00000000
0	AUTO_USBHOST	USB HOST auto clock control. 0x0: USB HOST interface clock is unrelated to the domain state transition. 0x1: USB HOST interface clock is automatically enabled or disabled along with the domain state transition.	RW	0x0

Table 4-280. Register Call Summary for Register CM_AUTOIDLE_USBHOST

PRCM Clock Manager Functional Description • USBHOST Power Domain Clock Controls: [0]
PRCM Basic Programming Model • Power-Domain Clock Control Registers: [1]
PRCM Register Manual • USBHOST_CM Registers: [2]

Table 4-281. CM_SLEEPDEP_USBHOST

Address Offset	0x0000 0044		
Physical Address	0x4800 5444	**Instance**	USBHOST_CM
Description	This register allows enabling or disabling the sleep transition dependency of USB HOST domain with respect to other domain.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																													EN_IVA2	EN_MPU	RESERVED

Bits	Field Name	Description	Type	Reset
31:3	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x00000000
2	EN_IVA2	IVA2 domain dependency 0x0: USB HOST domain sleep dependency with IVA2 domain is disabled. 0x1: USB HOST domain sleep dependency with IVA2 domain is enabled.	RW	0x0
1	EN_MPU	MPU domain dependency 0x0: USB HOST domain sleep dependency with MPU domain is disabled. 0x1: USB HOST domain sleep dependency with MPU domain is enabled.	RW	0x0
0	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x0

Table 4-282. Register Call Summary for Register CM_SLEEPDEP_USBHOST

PRCM Basic Programming Model • Power-Domain Clock Control Registers: [0]
PRCM Register Manual • USBHOST_CM Registers: [1]

Table 4-283. CM_CLKSTCTRL_USBHOST

Address Offset	0x0000 0048		
Physical Address	0x4800 5448	**Instance**	USBHOST_CM
Description	This register enables the domain power state transition. It controls the HW supervised domain power state transition between ACTIVE and INACTIVE states.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																														CLKTRCTRL_USBHOST	

Bits	Field Name	Description	Type	Reset
31:2	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x00000000
1:0	CLKTRCTRL_USBHOST	Controls the clock state transition of the USB HOST clock domain. 0x0: Automatic transition is disabled 0x1: Start a software supervised sleep transition on the domain 0x2: Start a software supervised wake-up transition on the domain 0x3: Automatic transition is enabled. Transition is supervised by the HardWare.	RW	0x0

Table 4-284. Register Call Summary for Register CM_CLKSTCTRL_USBHOST

PRCM Idle and Wake-Up Management • Device Wake-Up Events: [0] [1] [2]

Table 4-284. Register Call Summary for Register CM_CLKSTCTRL_USBHOST (continued)

PRCM Basic Programming Model • Power-Domain Clock Control Registers: [3]
PRCM Register Manual • USBHOST_CM Registers: [4]

Table 4-285. CM_CLKSTST_USBHOST

Address Offset	0x0000 004C		
Physical Address	0x4800 544C	**Instance**	USBHOST_CM
Description	This register provides a status on the interface clock activity in the domain.		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																															CLKACTIVITY_USBHOST

Bits	Field Name	Description	Type	Reset
31:1	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x00000000
0	CLKACTIVITY_USBHOST	Interface clock activity status 0x0: No domain interface clock activity 0x1: Domain interface clock is active	R	0x0

Table 4-286. Register Call Summary for Register CM_CLKSTST_USBHOST

PRCM Basic Programming Model • Power-Domain Clock Control Registers: [0]
PRCM Register Manual • USBHOST_CM Registers: [1]

4.14.2 PRM Module Registers

This section describes the PRM module registers.Table 4-287 lists the physical address of the the PRM modules. Table 4-288 through Table 4-498 provide register mapping summaries and describe the bits in the individual registers.

4.14.2.1 PRM Module Summary

Table 4-287. PRM Instance Summary

Module Name	Base address (hex)	Size
IVA2_PRM	0x4830 6000	8192 bytes
OCP_System_Reg_PRM	0x4830 6800	8192 bytes
MPU_PRM	0x4830 6900	8192 bytes
CORE_PRM	0x4830 6A00	8192 bytes
SGX_PRM	0x4830 6B00	8192 bytes
WKUP_PRM	0x4830 6C00	8192 bytes

Table 4-287. PRM Instance Summary (continued)

Module Name	Base address (hex)	Size
Clock_Control_Reg_PRM	0x4830 6D00	8192 bytes
DSS_PRM	0x4830 6E00	8192 bytes
CAM_PRM	0x4830 6F00	8192 bytes
PER_PRM	0x4830 7000	8192 bytes
EMU_PRM	0x4830 7100	8192 bytes
Global_Reg_PRM	0x4830 7200	65536 bytes
NEON_PRM	0x4830 7300	8192 bytes
USBHOST_PRM	0x4830 7400	8192 bytes

4.14.2.2 IVA2_PRM Registers

Table 4-288. IVA2_PRM Register Summary

Register Name	Type	Register Width (Bits)	Address Offset	Physical Address	Reset Type
RM_RSTCTRL_IVA2	RW	32	0x0000 0050	0x4830 6050	C (refer to Section 4.12.3.1.2)
RM_RSTST_IVA2	RW	32	0x0000 0058	0x4830 6058	C
PM_WKDEP_IVA2	RW	32	0x0000 00C8	0x4830 60C8	W
PM_PWSTCTRL_IVA2	RW	32	0x0000 00E0	0x4830 60E0	W
PM_PWSTST_IVA2	R	32	0x0000 00E4	0x4830 60E4	C
PM_PREPWSTST_IVA2	RW	32	0x0000 00E8	0x4830 60E8	C
PRM_IRQSTATUS_IVA2	RW	32	0x0000 00F8	0x4830 60F8	W
PRM_IRQENABLE_IVA2	RW	32	0x0000 00FC	0x4830 60FC	W

Table 4-289. RM_RSTCTRL_IVA2

Address Offset	0x0000 0050		
Physical Address	0x4830 6050	**Instance**	IVA2_PRM
Description	This register controls the release of the IVA2 sub-system resets.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																													RST3_IVA2	RST2_IVA2	RST1_IVA2

Bits	Field Name	Description	Type	Reset
31:3	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x00000000
2	RST3_IVA2	Video sequencer reset control 0x0: Video sequencer reset is cleared 0x1: Resets video sequencer	RW	0x1
1	RST2_IVA2	IVA2 - MMU reset control and Video Sequencer hardware accelerator reset control 0x0: IVA2 - MMU reset and Video Sequencer hardware accelerator reset are cleared 0x1: Resets IVA2 - MMU and Video Sequencer hardware accelerator	RW	0x1

Bits	Field Name	Description	Type	Reset
0	RST1_IVA2	IVA2 - DSP reset control 0x0: IVA2 - DSP reset is cleared 0x1: Resets IVA2 - DSP	RW	0x1

Table 4-290. Register Call Summary for Register RM_RSTCTRL_IVA2

PRCM Reset Manager Functional Description
- Local Reset Sources: [0] [1] [2]
- IVA2.2 Subsystem Power-Up Sequence: [3] [4] [5]
- IVA2 Software Reset Sequence: [6] [7] [8]
- IVA2 Global Warm Reset Sequence: [9]
- IVA2 Power Domain Wake-Up Cold Reset Sequence: [10] [11] [12] [13] [14] [15] [16] [17] [18] [19]

PRCM Basic Programming Model
- Reset Control: [20]

PRCM Register Manual
- IVA2_PRM Registers: [21]

Table 4-291. RM_RSTST_IVA2

Address Offset	0x0000 0058		
Physical Address	0x4830 6058	**Instance**	IVA2_PRM
Description	This register logs the different reset sources of the IVA2 domain. Each bit is set upon release of the domain reset signal. Must be cleared by software.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																		EMULATION_SEQ_RST	EMULATION_VIDEO_HWA_RST	EMULATION_IVA2_RST	IVA2_SW_RST3	IVA2_SW_RST2	IVA2_SW_RST1	RESERVED				COREDOMAINWKUP_RST	DOMAINWKUP_RST	GLOBALWARM_RST	GLOBALCOLD_RST

Bits	Field Name	Description	Type	Reset
31:14	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x00000
13	EMULATION_SEQ_RST	Emulation reset Read 0x0: No emulation reset. Write 0x0: Status bit unchanged Read 0x1: Video Sequencer has been reset upon an emulation reset Write 0x1: Status bit is cleared to 0.	RW	0x0
12	EMULATION_VIDEO_HWA_RST	Emulation reset Read 0x0: No emulation reset. Write 0x0: Status bit unchanged Read 0x1: Video Sequencer hardware accelerator has been reset upon an emulation reset Write 0x1: Status bit is cleared to 0.	RW	0x0

Bits	Field Name	Description	Type	Reset
11	EMULATION_IVA2_RST	Emulation reset Read 0x0: No emulation reset. Write 0x0: Status bit unchanged Read 0x1: IVA2 (DSP) has been reset upon an emulation reset Write 0x1: Status bit is cleared to 0.	RW	0x0
10	IVA2_SW_RST3	IVA2-Video Sequencer software reset Read 0x0: No IVA2-Video Sequencer software reset occured. Write 0x0: Status bit unchanged Read 0x1: IVA2 domain has been reset upon IVA2-Video Sequencer software reset. Write 0x1: Status bit is cleared to 0.	RW	0x0
9	IVA2_SW_RST2	IVA2-MMU software reset Read 0x0: No IVA2-MMU software reset occured. Write 0x0: Status bit unchanged Read 0x1: IVA2 domain has been reset upon IVA2-MMU software reset. Write 0x1: Status bit is cleared to 0.	RW	0x0
8	IVA2_SW_RST1	IVA2 - DSP software reset Read 0x0: No IVA2-DSP software reset occured. Write 0x0: Status bit unchanged Read 0x1: IVA2 domain has been reset upon IVA2-DSP software reset. Write 0x1: Status bit is cleared to 0.	RW	0x0
7:4	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x0
3	COREDOMAINWKUP_RST	CORE domain wake-up reset Read 0x0: No power domain wake-up reset. Write 0x0: Status bit unchanged Read 0x1: IVA2 domain has been reset following a CORE power domain wake-up from OFF to ON. Write 0x1: Status bit is cleared to 0.	RW	0x0
2	DOMAINWKUP_RST	Power domain wake-up reset Read 0x0: No DSP domain wake-up. Write 0x0: Status bit unchanged Read 0x1: IVA2 domain has been reset following an IVA2 domain wake-up. Write 0x1: Status bit is cleared to 0.	RW	0x0
1	GLOBALWARM_RST	Global warm reset Read 0x0: No Global warm reset. Write 0x0: Status bit unchanged Read 0x1: IVA2 domain has been reset upon global warm reset. Write 0x1: Status bit is cleared to 0.	RW	0x0
0	GLOBALCOLD_RST	Global cold reset Read 0x0: No global cold reset. Write 0x0: Status bit unchanged Read 0x1: IVA2 domain has been reset upon a global cold reset Write 0x1: Status bit is cleared to 0.	RW	0x1

Table 4-292. Register Call Summary for Register RM_RSTST_IVA2

PRCM Reset Manager Functional Description

- IVA2.2 Subsystem Power-Up Sequence: [0] [1] [2]
- IVA2 Software Reset Sequence: [3] [4] [5]
- IVA2 Global Warm Reset Sequence: [6]
- IVA2 Power Domain Wake-Up Cold Reset Sequence: [7] [8]

PRCM Basic Programming Model

- Reset Control: [9] [10]

PRCM Register Manual

- IVA2_PRM Registers: [11]

Table 4-293. PM_WKDEP_IVA2

Address Offset	0x0000 00C8		
Physical Address	0x4830 60C8	**Instance**	IVA2_PRM
Description	This register allows enabling or disabling the wake-up of the IVA2 domain upon another domain wakeup events.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																								EN_PER	RESERVED	EN_DSS	EN_WKUP	RESERVED	RESERVED	EN_MPU	EN_CORE

Bits	Field Name	Description	Type	Reset
31:8	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x000000
7	EN_PER	PER domain dependency 0x0: IVA2 domain is independent of PER domain wake-up event. 0x1: IVA2 domain is woken-up upon PER domain wake-up event.	RW	0x1
6	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x0
5	EN_DSS	WAKEUP domain dependency 0x0: IVA2 domain is independent of DSS domain wake-up event. 0x1: IVA2 domain is woken-up upon DSS domain wake-up event.	RW	0x1
4	EN_WKUP	WAKEUP domain dependency 0x0: IVA2 domain is independent of WKUP domain wake-up event. 0x1: IVA2 domain is woken-up upon WKUP domain wake-up event.	RW	0x1
3	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x0
2	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x0
1	EN_MPU	MPU domain dependency 0x0: IVA2 domain is independent of MPU domain wake-up event. 0x1: IVA2 domain is woken-up upon MPU domain wake-up event.	RW	0x1
0	EN_CORE	CORE domain dependency 0x0: IVA2 domain is independent of CORE domain wake-up event. 0x1: IVA2 domain is is woken-up upon CORE domain wake-up event.	RW	0x1

Table 4-294. Register Call Summary for Register PM_WKDEP_IVA2

PRCM Idle and Wake-Up Management • Device Wake-Up Events: [0] [1] [2] [3] [4]
PRCM Basic Programming Model • Domain Wake-Up Control Registers: [5]
PRCM Register Manual • IVA2_PRM Registers: [6]

Table 4-295. PM_PWSTCTRL_IVA2

Address Offset	0x0000 00E0		
Physical Address	0x4830 60E0	**Instance**	IVA2_PRM
Description	This register controls the IVA2 domain power state transition.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								L2FLATMEMONSTATE		SHAREDL2CACHEFLATONSTATE		L1FLATMEMONSTATE		SHAREDL1CACHEFLATONSTATE		RESERVED				L2FLATMEMRETSTATE		SHAREDL2CACHEFLATRETSTATE		L1FLATMEMRETSTATE		SHAREDL1CACHEFLATRETSTATE		RESERVED				MEMORYCHANGE		LOGICRETSTATE		POWERSTATE	

Bits	Field Name	Description	Type	Reset
31:24	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x00
23:22	L2FLATMEMONSTATE	L2 Flat memory state when domain is ON; Other enums: Reserved 0x0: Reserved 0x1: Reserved 0x2: Reserved 0x3: L2 Flat memory is always ON when domain is ON.	R	0x3
21:20	SHAREDL2CACHEFLATONSTATE	Shared L2 Cache and Flat memory state when domain is ON 0x0: Shared L2 Cache and Flat memory is OFF when domain is ON. 0x1: Reserved 0x2: Reserved 0x3: Shared L2 Cache and Flat memory is ON when domain is ON.	RW	0x3
19:18	L1FLATMEMONSTATE	L1 Flat memory state when domain is ON; Other enums: Reserved 0x0: Reserved 0x1: Reserved 0x2: Reserved 0x3: L1 Flat memory is always ON when domain is ON.	R	0x3

Bits	Field Name	Description	Type	Reset
17:16	SHAREDL1CACHEFLATONSTATE	Shared L1 Cache and Flat memory state when domain is ON 0x0: Reserved 0x1: Reserved 0x2: Reserved 0x3: Shared L1 Cache and Flat memory is always ON when domain is ON.	R	0x3
15:12	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x0
11	L2FLATMEMRETSTATE	L2 Flat memory state when domain is RETENTION 0x0: L2 Flat memory is OFF when domain is in RETENTION state. 0x1: L2 Flat memory is retained when domain is in RETENTION state.	RW	0x1
10	SHAREDL2CACHEFLATRETSTATE	Shared L2 Cache and Flat memory state when domain is RETENTION 0x0: Shared L2 Cache and Flat memory is OFF when domain is in RETENTION state. 0x1: Shared L2 Cache and Flat memory is retained when domain is in RETENTION state.	RW	0x1
9	L1FLATMEMRETSTATE	L1 Flat memory state when domain is RETENTION 0x0: L1 Flat memory is OFF when domain is in RETENTION state. 0x1: L1 Flat memory is retained when domain is in RETENTION state.	RW	0x1
8	SHAREDL1CACHEFLATRETSTATE	Shared L1 Cache and Flat memory state when domain is RETENTION 0x0: Shared L1 Cache and Flat memory is OFF when domain is in RETENTION state. 0x1: Shared L1 Cache and Flat memory is retained when domain is in RETENTION state.	RW	0x1
7:4	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x0
3	MEMORYCHANGE	Memory change control in ON state 0x0: Disable memory change 0x1: Enable memory change state in ON state. This bit is automaticaly cleared when memory state is effectively changed.	RW	0x0
2	LOGICRETSTATE	Logic state when RETENTION 0x0: Logic is OFF when domain is in RETENTION state. 0x1: Logic is retained when domain is in RETENTION state.	RW	0x1
1:0	POWERSTATE	Power state control 0x0: OFF state 0x1: RETENTION state 0x2: Reserved 0x3: ON state	RW	0x3

Table 4-296. Register Call Summary for Register PM_PWSTCTRL_IVA2

PRCM Basic Programming Model • PM_PWSTCTRL_ <domain_name> (Power State Control Register): [0] [1] [2] [3] [4]
PRCM Register Manual • IVA2_PRM Registers: [5]

Table 4-297. PM_PWSTST_IVA2

Address Offset	0x0000 00E4		
Physical Address	0x4830 60E4	**Instance**	IVA2_PRM
Description	This register provides a status on the power state transition of the IVA2 domain.		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED											INTRANSITION	RESERVED								L2FLATMEMSTATEST		SHAREDL2CACHEFLATSTATEST		L1FLATMEMSTATEST		SHAREDL1CACHEFLATSTATEST		RESERVED		LOGICSTATEST		POWERSTATEST	

Bits	Field Name	Description	Type	Reset
31:21	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x000
20	INTRANSITION	Domain transition status 0x0: No transition 0x1: IVA2 power domain transition is in progress.	R	0x0
19:12	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x00
11:10	L2FLATMEMSTATEST	L2 Flat memory state status 0x0: L2 Flat memory is OFF 0x1: L2 Flat memory is in RETENTION 0x2: Reserved 0x3: L2 Flat memory is ON	R	0x3
9:8	SHAREDL2CACHEFLATSTATEST	Shared L2 Cache and Flat memory state status 0x0: Shared L2 Cache and Flat memory is OFF 0x1: Shared L2 Cache and Flat memory is in RETENTION 0x2: Reserved 0x3: Shared L2 Cache and Flat memory is ON	R	0x3
7:6	L1FLATMEMSTATEST	L1 Flat memory state status 0x0: L1 Flat memory is OFF 0x1: L1 Flat memory is in RETENTION 0x2: Reserved 0x3: L1 Flat memory is ON	R	0x3
5:4	SHAREDL1CACHEFLATSTATEST	Shared L1 Cache and Flat memory state status 0x0: Shared L1 Cache and Flat memory is OFF 0x1: Shared L1 Cache and Flat memory is in RETENTION 0x2: Reserved 0x3: Shared L1 Cache and Flat memory is ON	R	0x3
3	RESERVED	Read returns 0.	R	0x0
2	LOGICSTATEST	Logic state status 0x0: IVA2 domain logic is OFF 0x1: IVA2 domain logic is ON	R	0x1

Bits	Field Name	Description	Type	Reset
1:0	POWERSTATEST	Current power state status 0x0: Power domain is OFF 0x1: Power domain is in RETENTION 0x2: Power domain is INACTIVE 0x3: Power domain is ON	R	0x3

Table 4-298. Register Call Summary for Register PM_PWSTST_IVA2

PRCM Basic Programming Model
- PM_PWSTST_ <domain_name> (Power State Status Register): [0]

PRCM Register Manual
- IVA2_PRM Registers: [1]

Table 4-299. PM_PREPWSTST_IVA2

Address Offset	0x0000 00E8		
Physical Address	0x4830 60E8	**Instance**	IVA2_PRM
Description	This register provides a status on the IVA2 domain previous power state. It indicates the state entered during the last sleep transition.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																				LASTL2FLATMEMSTATEENTERED		LASTSHAREDL2CACHEFLATSTATE		LASTL1FLATMEMSTATEENTERED		LASTSHAREDL1CACHEFLATSTATE		RESERVED		LASTLOGICSTATEENTERED		LASTPOWERSTATEENTERED	

Bits	Field Name	Description	Type	Reset
31:12	RESERVED	Read returns 0.	R	0x00000
11:10	LASTL2FLATMEMSTATEENTERED	Last L2 Flat memory state entered 0x0: L2 Flat memory was previously OFF 0x1: L2 Flat memory was previously in RETENTION 0x2: Reserved 0x3: L2 Flat memory was previously ON	RW	0x0
9:8	LASTSHAREDL2CACHEFLATSTATE ENTERED	Shared L2 Cache and Flat memory last state entered 0x0: Shared L2 Cache and Flat memory was previously OFF 0x1: Shared L2 Cache and Flat memory was previously in RETENTION 0x2: Reserved 0x3: Shared L2 Cache and Flat memory was previously ON	RW	0x0

Bits	Field Name	Description	Type	Reset
7:6	LASTL1FLATMEMSTATEENTERED	Last L1 Flat memory state entered 0x0: L1 Flat memory was previously OFF 0x1: L1 Flat memory was previously in RETENTION 0x2: Reserved 0x3: L1 Flat memory was previously ON	RW	0x0
5:4	LASTSHAREDL1CACHEFLATSTATE ENTERED	Shared L1 Cache and Flat memory last state entered 0x0: Shared L1 Cache and Flat memory was previously OFF 0x1: Shared L1 Cache and Flat memory was previously in RETENTION 0x2: Reserved 0x3: Shared L1 Cache and Flat memory was previously ON	RW	0x0
3	RESERVED	Read returns 0.	R	0x0
2	LASTLOGICSTATEENTERED	Last logic state entered 0x0: IVA2 domain logic was previously OFF 0x1: IVA2 domain logic was previously ON	RW	0x0
1:0	LASTPOWERSTATEENTERED	Last power state entered 0x0: IVA2 domain was previously OFF 0x1: IVA2 domain was previously in RETENTION 0x2: IVA2 domain was previously INACTIVE 0x3: IVA2 domain was previously ON	RW	0x0

Table 4-300. Register Call Summary for Register PM_PREPWSTST_IVA2

PRCM Register Manual
- IVA2_PRM Registers: [0]

Table 4-301. PRM_IRQSTATUS_IVA2

Address Offset	0x0000 00F8		
Physical Address	0x4830 60F8	**Instance**	IVA2_PRM
Description	This interrupt status register regroups all the status of the module internal events that can generate an interrupt. Write 1 to a given bit resets this bit. This register applies on the interrupt line 1 mapped to the IVA2 interrupt controller.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																													IVA2_DPLL_ST	FORCEWKUP_ST	WKUP_ST

Bits	Field Name	Description	Type	Reset
31:3	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x00000000
2	IVA2_DPLL_ST	DPLL2 recalibration event status Read 0x0: DPLL2 recalibration event is false Write 0x0: Status bit unchanged Read 0x1: DPLL2 recalibration event is true (pending) Write 0x1: Status bit is cleared to 0.	RW	0x0

Bits	Field Name	Description	Type	Reset
1	FORCEWKUP_ST	Force wake-up IVA2 domain transition completed event status Read 0x0: Wake-up event is false Write 0x0: Status bit unchanged Read 0x1: Wake-up event is true (pending) Write 0x1: Status bit is cleared to 0.	RW	0x0
0	WKUP_ST	IVA2 peripherals group wake-up event status Read 0x0: Wake-up event is false Write 0x0: Status bit unchanged Read 0x1: Wake-up event is true (pending) Write 0x1: Status bit is cleared to 0.	RW	0x0

Table 4-302. Register Call Summary for Register PRM_IRQSTATUS_IVA2

PRCM Clock Manager Functional Description • Recalibration: [0]
PRCM Interrupts • PRCM Interrupts: [1] [2] [3] [4]
PRCM Basic Programming Model • Interrupt Configuration Registers: [5]
PRCM Register Manual • IVA2_PRM Registers: [6]

Table 4-303. PRM_IRQENABLE_IVA2

Address Offset	0x0000 00FC		
Physical Address	0x4830 60FC	**Instance**	IVA2_PRM
Description	The interrupt enable register allows masking/unmasking the module internal sources of interrupt, on a event-by-event basis. This registers applies on the interrupt line 0 mapped to the IVA2 Wake-Up Generator.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																													IVA2_DPLL_RECAL_EN	FORCEWKUP_EN	WKUP_EN

Bits	Field Name	Description	Type	Reset
31:3	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x00000000
2	IVA2_DPLL_RECAL_EN	DPLL2 recalibration mask 0x0: DPLL2 recalibration event is masked 0x1: DPLL2 recalibration event generates an interrupt	RW	0x0
1	FORCEWKUP_EN	Force wake-up IVA2 domain transition completed event mask 0x0: Force wake-up IVA2 domain transition completed event is masked 0x1: Force wake-up IVA2 domain transition completed event generates an interrupt	RW	0x0

Bits	Field Name	Description	Type	Reset
0	WKUP_EN	IVA2 peripherals group wake-up event mask 0x0: IVA2 peripherals group wake-up event is masked 0x1: IVA2 peripherals group wake-up event generates an interrupt	RW	0x0

Table 4-304. Register Call Summary for Register PRM_IRQENABLE_IVA2

PRCM Clock Manager Functional Description • Recalibration: [0]
PRCM Idle and Wake-Up Management • Device Wake-Up Events: [1]
PRCM Interrupts • PRCM Interrupts: [2] [3] [4] [5]
PRCM Basic Programming Model • Interrupt Configuration Registers: [6]
PRCM Register Manual • IVA2_PRM Registers: [7]

4.14.2.3 OCP_System_Reg_PRM Registers

Table 4-305. OCP_System_Reg_PRM Register Summary

Register Name	Type	Register Width (Bits)	Address Offset	Physical Address	Reset Type
PRM_REVISION	R	32	0x0000 0004	0x4830 6804	C
PRM_SYSCONFIG	RW	32	0x0000 0014	0x4830 6814	W
PRM_IRQSTATUS_MPU	RW	32	0x0000 0018	0x4830 6818	W
PRM_IRQENABLE_MPU	RW	32	0x0000 001C	0x4830 681C	W

Table 4-306. PRM_REVISION

Address Offset	0x0000 0004		
Physical Address	0x4830 6804	**Instance**	OCP_System_Reg_PRM
Description	This register contains the IP revision code for the PRM part of the PRCM		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																								REV							

Bits	Field Name	Description	Type	Reset
31:8	RESERVED	Reads returns 0.	R	0x000000
7:0	REV	IP revision [7:4] Major revision [3:0] Minor revision Examples: 0x10 for 1.0, 0x21 for 2.1	R	0x10

Table 4-307. Register Call Summary for Register PRM_REVISION

PRCM Basic Programming Model • Revision Information Registers: [0]
PRCM Register Manual • OCP_System_Reg_PRM Registers: [1]

Table 4-308. PRM_SYSCONFIG

Address Offset	0x0000 0014		
Physical Address	0x4830 6814	**Instance**	OCP_System_Reg_PRM
Description	This register controls the various parameters of the interface		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																															AUTOIDLE

Bits	Field Name	Description	Type	Reset
31:1	RESERVED	Write 0s for future compatibility. Reads returns 0.	R	0x00000000
0	AUTOIDLE	Internal clock gating strategy (for the CM part of the PRCM) 0x0: Interface clock is free-running 0x1: Automatic clock gating strategy is enabled, based on the interface activity.	RW	0x1

Table 4-309. Register Call Summary for Register PRM_SYSCONFIG

PRCM Basic Programming Model
- PRCM Configuration Registers: [0]

PRCM Register Manual
- OCP_System_Reg_PRM Registers: [1]

Table 4-310. PRM_IRQSTATUS_MPU

Address Offset	0x0000 0018		
Physical Address	0x4830 6818	**Instance**	OCP_System_Reg_PRM
Description	This interrupt status register regroups all the status of the module internal events that can generate an interrupt. Write 1 to a given bit resets this bit. This registers applies on the interrupt line 0 mapped to the MPU interrupt controller.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED						SND_PERIPH_DPLL_ST	VC_TIMEOUTERR_ST	VC_RAERR_ST	VC_SAERR_ST	RESERVED												IO_ST	IVA2_DPLL_ST	MPU_DPLL_ST	PERIPH_DPLL_ST	CORE_DPLL_ST	TRANSITION_ST	EVGENOFF_ST	EVGENON_ST	RESERVED	WKUP_ST

Bits	Field Name	Description	Type	Reset
31:26	RESERVED	Write 0s for future compatibility. Reads returns 0.	R	0x00
25	SND_PERIPH_DPLL_ST	DPLL5 recalibration event status Read 0x0: DPLL5 recalibration event is false Write 0x0: Status bit unchanged Read 0x1: DPLL5 recalibration event is true (pending) Write 0x1: Status bit is cleared to 0.	RW	0x0

Bits	Field Name	Description	Type	Reset
24	VC_TIMEOUTERR_ST	Voltage Controller timeout error event status Read 0x0: Voltage Controller timeout error event is false Write 0x0: Status bit unchanged Read 0x1: Voltage Controller timeout error event is true (pending) Write 0x1: Status bit is cleared to 0.	RW	0x0
23	VC_RAERR_ST	Voltage Controller register address acknowledge error event status Read 0x0: Voltage Controller register address acknowledge error event is false Write 0x0: Status bit unchanged Read 0x1: Voltage Controller register address acknowledge error event is true (pending) Write 0x1: Status bit is cleared to 0.	RW	0x0
22	VC_SAERR_ST	Voltage Controller slave address acknowledge error event status Read 0x0: Voltage Controller slave address acknowledge error event is false Write 0x0: Status bit unchanged Read 0x1: Voltage Controller slave address acknowledge error event is true (pending) Write 0x1: Status bit is cleared to 0.	RW	0x0
21:10	RESERVED	This information is not available in public domain.	RW	0x000
9	IO_ST	IO pad event status Read 0x0: IO pad event is false Write 0x0: Status bit unchanged Read 0x1: IO pad event is true (pending) Write 0x1: Status bit is cleared to 0.	RW	0x0
8	IVA2_DPLL_ST	IVA2 DPLL recalibration event status Read 0x0: IVA2 DPLL recalibration event is false Write 0x0: Status bit unchanged Read 0x1: IVA2 DPLL recalibration event is true (pending) Write 0x1: Status bit is cleared to 0.	RW	0x0
7	MPU_DPLL_ST	DPLL1 recalibration event status Read 0x0: DPLL1 recalibration event is false Write 0x0: Status bit unchanged Read 0x1: DPLL1 recalibration event is true (pending) Write 0x1: Status bit is cleared to 0.	RW	0x0
6	PERIPH_DPLL_ST	DPLL4 recalibration event status Read 0x0: DPLL4 recalibration event is false Write 0x0: Status bit unchanged Read 0x1: DPLL4 recalibration event is true (pending) Write 0x1: Status bit is cleared to 0.	RW	0x0
5	CORE_DPLL_ST	DPLL3 recalibration event status Read 0x0: DPLL3 recalibration event is false Write 0x0: Status bit unchanged Read 0x1: DPLL3 recalibration event is true (pending) Write 0x1: Status bit is cleared to 0.	RW	0x0

Bits	Field Name	Description	Type	Reset
4	TRANSITION_ST	Software supervised transition completed event status Read 0x0: Software supervised transition completed event is false Write 0x0: Status bit unchanged Read 0x1: Software supervised transition completed event is true (pending) Write 0x1: Status bit is cleared to 0.	RW	0x0
3	EVGENOFF_ST	Event Generator endOFFtime status Read 0x0: End of OFF time event is false Write 0x0: Status bit unchanged Read 0x1: End of OFF time event is true (pending) Write 0x1: Status bit is cleared to 0.	RW	0x0
2	EVGENON_ST	Event Generator endONtime status Read 0x0: End of ON time event is false Write 0x0: Status bit unchanged Read 0x1: End of ON time event is true (pending) Write 0x1: Status bit is cleared to 0.	RW	0x0
1	RESERVED	Write 0s for future compatibility. Reads returns 0.	R	0x0
0	WKUP_ST	MPU peripherals group wake-up event status Read 0x0: Wake-up event is false Write 0x0: Status bit unchanged Read 0x1: Wake-up event is true (pending) Write 0x1: Status bit is cleared to 0.	RW	0x0

Table 4-311. Register Call Summary for Register PRM_IRQSTATUS_MPU

PRCM Clock Manager Functional Description • Recalibration: [0] [1] [2] [3] [4]
PRCM Interrupts • PRCM Interrupts: [5]
PRCM Voltage Management Functional Description • SmartReflex Voltage Control :
PRCM Basic Programming Model • Interrupt Configuration Registers: [18] • Changing OPP Using the SmartReflex Module :
PRCM Use Cases and Tips • DVFS Using SmartReflex :
PRCM Register Manual • OCP_System_Reg_PRM Registers: [29]

Table 4-312. PRM_IRQENABLE_MPU

Address Offset	0x0000 001C		
Physical Address	0x4830 681C	**Instance**	OCP_System_Reg_PRM
Description	The interrupt enable register allows masking/unmasking the module internal sources of interrupt, on a event-by-event basis. This registers applies on the interrupt line 0 mapped to the MPU interrupt controller.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED						SND_PERIPH_DPLL_RECAL_EN	VC_TIMEOUTERR_EN	VC_RAERR_EN	VC_SAERR_EN	RESERVED												IO_EN	IVA2_DPLL_RECAL_EN	MPU_DPLL_RECAL_EN	PERIPH_DPLL_RECAL_EN	CORE_DPLL_RECAL_EN	TRANSITION_EN	EVGENOFF_EN	EVGENON_EN	RESERVED	WKUP_EN

Bits	Field Name	Description	Type	Reset
31:26	RESERVED	Write 0s for future compatibility. Reads returns 0.	R	0x00
25	SND_PERIPH_DPLL_RECAL_EN	DPLL5 recalibration mask 0x0: DPLL5 recalibration event is masked 0x1: DPLL5 recalibration event generates an interrupt	RW	0x0
24	VC_TIMEOUTERR_EN	Voltage Controller timeout error mask. 0x0: Voltage Controller timeout error event is masked 0x1: Voltage Controller timeout error event generates an interrupt	RW	0x0
23	VC_RAERR_EN	Voltage Controller register address acknowledge error mask. 0x0: Voltage Controller register address acknowledge error event is masked 0x1: Voltage Controller register address acknowledge error event generates an interrupt	RW	0x0
22	VC_SAERR_EN	Voltage Controller slave address acknowledge error mask. 0x0: Voltage Controller slave address acknowledge error event is masked 0x1: Voltage Controller slave address acknowledge error event generates an interrupt	RW	0x0
21:10	RESERVED	This information is not available in public domain.	RW	0x000
9	IO_EN	IO pad event mask 0x0: IO pad event is masked 0x1: IO pad event generates an interrupt	RW	0x0
8	IVA2_DPLL_RECAL_EN	IVA2 DPLL recalibration mask 0x0: IVA2 DPLL recalibration event is masked 0x1: IVA2 DPLL recalibration event generates an interrupt	RW	0x0
7	MPU_DPLL_RECAL_EN	DPLL1 recalibration mask 0x0: DPLL1 recalibration event is masked 0x1: DPLL1 recalibration event generates an interrupt	RW	0x0

Bits	Field Name	Description	Type	Reset
6	PERIPH_DPLL_RECAL_EN	DPLL4 recalibration mask 0x0: DPLL4 recalibration event is masked 0x1: DPLL4 recalibration event generates an interrupt	RW	0x0
5	CORE_DPLL_RECAL_EN	DPLL3 recalibration mask 0x0: DPLL3 recalibration event is masked 0x1: DPLL3 recalibration event generates an interrupt	RW	0x0
4	TRANSITION_EN	Software supervised transition completed event mask 0x0: Software supervised transition completed event is masked. 0x1: Software supervised transition completed event generates an interrupt.	RW	0x0
3	EVGENOFF_EN	Event Generator endOFFtime mask 0x0: End of OFF time event is masked 0x1: End of OFF time event generates an interrupt	RW	0x0
2	EVGENON_EN	Event Generator endONtime mask 0x0: End of ON time event is masked 0x1: End of ON time event generates an interrupt	RW	0x0
1	RESERVED	Write 0s for future compatibility. Reads returns 0.	R	0x0
0	WKUP_EN	MPU peripherals group wake-up event mask 0x0: MPU peripherals group wake-up event is masked 0x1: MPU peripherals group wake-up event generates an interrupt	RW	0x0

Table 4-313. Register Call Summary for Register PRM_IRQENABLE_MPU

PRCM Clock Manager Functional Description • Recalibration: [0] [1] [2] [3] [4]
PRCM Idle and Wake-Up Management • Device Wake-Up Events: [5]
PRCM Interrupts • PRCM Interrupts: [6]
PRCM Voltage Management Functional Description • SmartReflex Voltage Control :
PRCM Basic Programming Model • Interrupt Configuration Registers: [19] [20] • Voltage Processor Initialization Basic Programming Model : • Changing OPP Using the SmartReflex Module : • Event Generator Programming Examples: [34]
PRCM Use Cases and Tips • DVFS Using SmartReflex :
PRCM Register Manual • OCP_System_Reg_PRM Registers: [39]

4.14.2.4 MPU_PRM Registers

Table 4-314. MPU_PRM Register Summary

Register Name	Type	Register Width (Bits)	Address Offset	Physical Address	Reset Type
RM_RSTST_MPU	RW	32	0x0000 0058	0x4830 6958	C
PM_WKDEP_MPU	RW	32	0x0000 00C8	0x4830 69C8	W

Table 4-314. MPU_PRM Register Summary (continued)

Register Name	Type	Register Width (Bits)	Address Offset	Physical Address	Reset Type
PM_EVGENCTRL_MPU	RW	32	0x0000 00D4	0x4830 69D4	W
PM_EVGENONTIM_MPU	RW	32	0x0000 00D8	0x4830 69D8	W
PM_EVGENOFFTIM_MPU	RW	32	0x0000 00DC	0x4830 69DC	W
PM_PWSTCTRL_MPU	RW	32	0x0000 00E0	0x4830 69E0	W
PM_PWSTST_MPU	R	32	0x0000 00E4	0x4830 69E4	C
PM_PREPWSTST_MPU	RW	32	0x0000 00E8	0x4830 69E8	C

Table 4-315. RM_RSTST_MPU

Address Offset	0x0000 0058		
Physical Address	0x4830 6958	**Instance**	MPU_PRM
Description	This register logs the different reset sources of the MPU domain. Each bit is set upon release of the domain reset signal. Must be cleared by software.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																				EMULATION_MPU_RST	RESERVED							COREDOMAINWKUP_RST	DOMAINWKUP_RST	GLOBALWARM_RST	GLOBALCOLD_RST				

Bits	Field Name	Description	Type	Reset
31:12	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x00000
11	EMULATION_MPU_RST	Emulation reset Read 0x0: No emulation reset. Write 0x0: Status bit unchanged Read 0x1: MPU domain has been reset upon an emulation reset Write 0x1: Status bit is cleared to 0.	RW	0x0
10:4	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x00
3	COREDOMAINWKUP_RST	CORE domain wake-up reset Read 0x0: No power domain wake-up reset. Write 0x0: Status bit unchanged Read 0x1: MPU domain has been reset following a CORE power domain wake-up from OFF to ON. Write 0x1: Status bit is cleared to 0.	RW	0x0
2	DOMAINWKUP_RST	Power domain wake-up reset Read 0x0: No power domain wake-up reset. Write 0x0: Status bit unchanged Read 0x1: MPU domain has been reset following a MPU power domain wake-up. Write 0x1: Status bit is cleared to 0.	RW	0x0

Bits	Field Name	Description	Type	Reset
1	GLOBALWARM_RST	Global warm reset Read 0x0: No global warm reset. Write 0x0: Status bit unchanged Read 0x1: MPU domain has been reset upon a global warm reset Write 0x1: Status bit is cleared to 0.	RW	0x0
0	GLOBALCOLD_RST	Global cold reset Read 0x0: No global cold reset. Write 0x0: Status bit unchanged Read 0x1: MPU domain has been reset upon a global cold reset Write 0x1: Status bit is cleared to 0.	RW	0x1

Table 4-316. Register Call Summary for Register RM_RSTST_MPU

PRCM Basic Programming Model • Reset Control: [0]
PRCM Register Manual • MPU_PRM Registers: [1]

Table 4-317. PM_WKDEP_MPU

Address Offset	0x0000 00C8		
Physical Address	0x4830 69C8	**Instance**	MPU_PRM
Description	This register allows enabling or disabling the wake-up of the MPU domain upon another domain wakeup events.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																								EN_PER	RESERVED	EN_DSS	RESERVED			EN_IVA2	RESERVED	EN_CORE

Bits	Field Name	Description	Type	Reset
31:8	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x000000
7	EN_PER	PER domain dependency 0x0: MPU domain is independent of PER domain wake-up event. 0x1: MPU domain is woken-up upon PER domain wake-up event.	RW	0x1
6	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x0
5	EN_DSS	DSS domain dependency 0x0: MPU domain is independent of DSS domain wake-up event. 0x1: MPU domain is woken-up upon DSS domain wake-up event.	RW	0x1
4:3	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x0
2	EN_IVA2	IVA2 domain dependency 0x0: MPU domain is independent of IVA2 domain wake-up event. 0x1: MPU domain is woken-up upon IVA2 domain wake-up event.	RW	0x1
1	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x0

Bits	Field Name	Description	Type	Reset
0	EN_CORE	CORE domain dependency 0x0: MPU domain is independent of CORE domain wake-up event. 0x1: MPU domain is is woken-up upon CORE domain wake-up event.	RW	0x1

Table 4-318. Register Call Summary for Register PM_WKDEP_MPU

PRCM Idle and Wake-Up Management • Device Wake-Up Events: [0] • Wake-Up Dependencies: [1]
PRCM Basic Programming Model • Power-Domain Clock Control Registers: [2] • Domain Wake-Up Control Registers: [3]
PRCM Register Manual • MPU_PRM Registers: [4]

Table 4-319. PM_EVGENCTRL_MPU

Address Offset	0x0000 00D4		
Physical Address	0x4830 69D4	**Instance**	MPU_PRM
Description	This register allows controlling the feature of the event generator.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																											OFFLOADMODE			ONLOADMODE		ENABLE

Bits	Field Name	Description	Type	Reset
31:5	RESERVED	Write 0s for future compatibility . Read returns 0	R	0x0000000
4:3	OFFLOADMODE	OFF load mode setting 0x0: Load on update of PM_EVGENOFFTIM_MPU 0x1: Reserved 0x2: Load on MPU standby signal assertion 0x3: Auto load	RW	0x2
2:1	ONLOADMODE	ON load mode setting 0x0: Load on update of PM_EVGENONTIM_MPU 0x1: Load on MPU standby signal de-assertion 0x2: Reserved 0x3: Auto load	RW	0x1
0	ENABLE	Event generator control 0x0: Disable event generator 0x1: Enable event generator	RW	0x0

Table 4-320. Register Call Summary for Register PM_EVGENCTRL_MPU

PRCM Power Manager Functional Description • Power Domain Controls: [0]

Table 4-320. Register Call Summary for Register PM_EVGENCTRL_MPU (continued)

PRCM Idle and Wake-Up Management • Device Wake-Up Events: [1]
PRCM Basic Programming Model • Event Generator Control Registers: [2] • Event Generator Programming Examples: [3]
PRCM Register Manual • MPU_PRM Registers: [4]

Table 4-321. PM_EVGENONTIM_MPU

Address Offset	0x0000 00D8		
Physical Address	0x4830 69D8	**Instance**	MPU_PRM
Description	This register sets the ON count duration of the event generator (number of system clock cycles).		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ONTIMEVAL																															

Bits	Field Name	Description	Type	Reset
31:0	ONTIMEVAL	Number of system clock cycles for the ON period.	RW	0x00000000

Table 4-322. Register Call Summary for Register PM_EVGENONTIM_MPU

PRCM Power Manager Functional Description • Power Domain Controls: [0] [1]
PRCM Basic Programming Model • Event Generator Control Registers: [2] • Event Generator Programming Examples: [3] [4]
PRCM Register Manual • MPU_PRM Registers: [5] [6]

Table 4-323. PM_EVGENOFFTIM_MPU

Address Offset	0x0000 00DC		
Physical Address	0x4830 69DC	**Instance**	MPU_PRM
Description	This register sets the OFF count duration of the event generator (number of system clock cycles).		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
OFFTIMEVAL																															

Bits	Field Name	Description	Type	Reset
31:0	OFFTIMEVAL	Number of system clock cycles for the OFF period.	RW	0x00000000

Table 4-324. Register Call Summary for Register PM_EVGENOFFTIM_MPU

PRCM Power Manager Functional Description • Power Domain Controls: [0] [1]
PRCM Basic Programming Model • Event Generator Control Registers: [2] • Event Generator Programming Examples: [3] [4]
PRCM Register Manual • MPU_PRM Registers: [5] [6]

Table 4-325. PM_PWSTCTRL_MPU

Address Offset	0x0000 00E0		
Physical Address	0x4830 69E0	**Instance**	MPU_PRM
Description	This register controls the MPU domain power state transition.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED														L2CACHEONSTATE		RESERVED							L2CACHERETSTATE	RESERVED				MEMORYCHANGE	LOGICL1CACHERETSTATE	POWERSTATE	

Bits	Field Name	Description	Type	Reset
31:18	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x0000
17:16	L2CACHEONSTATE	L2 Cache memory state when domain is ON; Other enums: Reserved 0x0: L2 Cache memory is OFF when domain is ON. 0x1: Reserved 0x2: Reserved 0x3: L2 Cache memory is ON when domain is ON.	RW	0x3
15:9	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x00
8	L2CACHERETSTATE	L2 Cache memory state when domain is RETENTION 0x0: L2 Cache memory is OFF when domain is in RETENTION state. 0x1: L2 Cache memory is retained when domain is in RETENTION state.	RW	0x1
7:4	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x0
3	MEMORYCHANGE	Memory change control in ON state 0x0: Disable memory change 0x1: Enable memory change state in ON state. This bit is automaticaly cleared when memory state is effectively changed.	RW	0x0
2	LOGICL1CACHERETSTATE	Logic and L1 Cache state when domain is RETENTION 0x0: Logic and L1 Cache are OFF when domain is in RETENTION state. 0x1: Logic and L1 Cache are retained when domain is in RETENTION state.	RW	0x1
1:0	POWERSTATE	Power state control 0x0: OFF state 0x1: RETENTION state 0x2: Reserved 0x3: ON state	RW	0x3

Table 4-326. Register Call Summary for Register PM_PWSTCTRL_MPU

PRCM Basic Programming Model

- PM_PWSTCTRL_ <domain_name> (Power State Control Register): [0]

Table 4-326. Register Call Summary for Register PM_PWSTCTRL_MPU (continued)

PRCM Register Manual • MPU_PRM Registers: [1]

Table 4-327. PM_PWSTST_MPU

Address Offset	0x0000 00E4		
Physical Address	0x4830 69E4	**Instance**	MPU_PRM
Description	This register provides a status on the MPU domain power state.		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED											INTRANSITION	RESERVED												L2CACHESTATEST		RESERVED			LOGICL1CACHESTATEST		POWERSTATEST	

Bits	Field Name	Description	Type	Reset
31:21	RESERVED	Read returns 0.	R	0x000
20	INTRANSITION	Domain transition status 0x0: No transition 0x1: MPU power domain transition is in progress.	R	0x0
19:8	RESERVED	Read returns 0.	R	0x000
7:6	L2CACHESTATEST	L2 Cache memory state status 0x0: L2 Cache memory is OFF 0x1: L2 Cache memory is in RETENTION 0x2: Reserved 0x3: L2 Cache memory is ON	R	0x3
5:3	RESERVED	Read returns 0.	R	0x0
2	LOGICL1CACHESTATEST	Logic and L1 Cache state status 0x0: MPU domain logic and L1 Cache is OFF 0x1: MPU domain logic and L1 Cache is ON	R	0x1
1:0	POWERSTATEST	Current power state status 0x0: Power domain is OFF 0x1: Power domain is in RETENTION 0x2: Power domain is INACTIVE 0x3: Power domain is ON	R	0x3

Table 4-328. Register Call Summary for Register PM_PWSTST_MPU

PRCM Basic Programming Model • PM_PWSTST_ <domain_name> (Power State Status Register): [0]
PRCM Register Manual • MPU_PRM Registers: [1]

Table 4-329. PM_PREPWSTST_MPU

Address Offset	0x0000 00E8		
Physical Address	0x4830 69E8	**Instance**	MPU_PRM
Description	This register provides a status on the MPU domain previous power state. It indicates the state entered during the last sleep transition.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																								LASTL2CACHESTATEENTERED		RESERVED			LASTLOGICL1CACHESTATE	LASTPOWERSTATEENTERED	

Bits	Field Name	Description	Type	Reset
31:8	RESERVED	Read returns 0.	R	0x000000
7:6	LASTL2CACHESTATEENTERED	Last L2 Cache memory state entered 0x0: L2 Cache memory was previously OFF 0x1: L2 Cache memory was previously in RETENTION 0x2: Reserved 0x3: L2 Cache memory was previously ON	RW	0x0
5:3	RESERVED	Read returns 0.	R	0x0
2	LASTLOGICL1CACHESTATEENTERED	Last logic and L1 Cache state entered 0x0: MPU domain logic and L1 Cache was previously OFF 0x1: MPU domain logic and L1 Cache was previously ON	RW	0x0
1:0	LASTPOWERSTATEENTERED	Last power state entered 0x0: MPU domain was previously OFF 0x1: MPU domain was previously in RETENTION 0x2: MPU domain was previously INACTIVE 0x3: MPU domain was previously ON	RW	0x0

Table 4-330. Register Call Summary for Register PM_PREPWSTST_MPU

PRCM Register Manual

- MPU_PRM Registers: [0]

4.14.2.5 CORE_PRM Registers

Table 4-331. CORE_PRM Register Summary

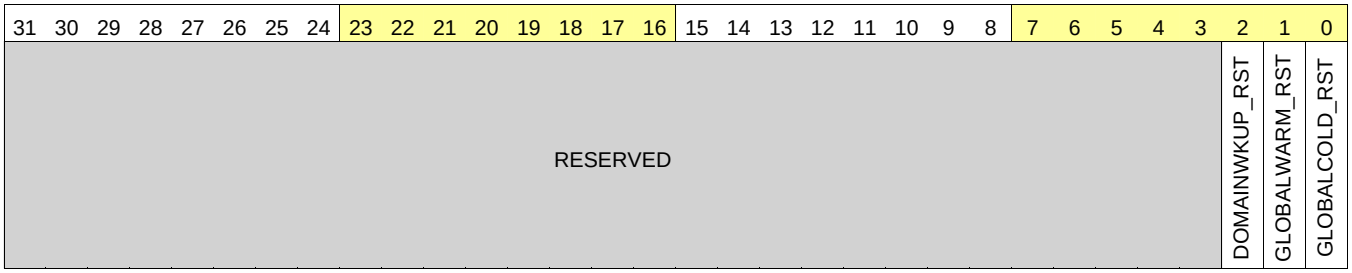
Register Name	Type	Register Width (Bits)	Address Offset	Physical Address	Reset Type
RM_RSTST_CORE	RW	32	0x0000 0058	0x4830 6A58	C
PM_WKEN1_CORE	RW	32	0x0000 00A0	0x4830 6AA0	W
PM_MPUGRPSEL1_CORE	RW	32	0x0000 00A4	0x4830 6AA4	W

Table 4-331. CORE_PRM Register Summary (continued)

Register Name	Type	Register Width (Bits)	Address Offset	Physical Address	Reset Type
PM_IVA2GRPSEL1_CORE	RW	32	0x0000 00A8	0x4830 6AA8	W
PM_WKST1_CORE	RW	32	0x0000 00B0	0x4830 6AB0	C
PM_WKST3_CORE	RW	32	0x0000 00B8	0x4830 6AB8	C
PM_PWSTCTRL_CORE	RW	32	0x0000 00E0	0x4830 6AE0	W
PM_PWSTST_CORE	R	32	0x0000 00E4	0x4830 6AE4	C
PM_PREPWSTST_CORE	RW	32	0x0000 00E8	0x4830 6AE8	C
PM_WKEN3_CORE	RW	32	0x0000 00F0	0x4830 6AF0	W
PM_IVA2GRPSEL3_CORE	RW	32	0x0000 00F4	0x4830 6AF4	W
PM_MPUGRPSEL3_CORE	RW	32	0x0000 00F8	0x4830 6AF8	W

Table 4-332. RM_RSTST_CORE

Address Offset	0x0000 0058		
Physical Address	0x4830 6A58	**Instance**	CORE_PRM
Description	This register logs the different reset sources of the CORE domain. Each bit is set upon release of the domain reset signal. Must be cleared by software.		
Type	RW		



Bits	Field Name	Description	Type	Reset
31:3	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x00000000
2	DOMAINWKUP_RST	Power domain wake-up reset Read 0x0: No power domain wake-up reset. Write 0x0: Status bit unchanged Read 0x1: CORE domain has been reset following a CORE power domain wake-up. Write 0x1: Status bit is cleared to 0.	RW	0x0
1	GLOBALWARM_RST	Global warm reset Read 0x0: No global warm reset. Write 0x0: Status bit unchanged Read 0x1: CORE domain has been reset upon a global warm reset Write 0x1: Status bit is cleared to 0.	RW	0x0
0	GLOBALCOLD_RST	Global cold reset Read 0x0: No global cold reset. Write 0x0: Status bit unchanged Read 0x1: CORE domain has been reset upon a global cold reset Write 0x1: Status bit is cleared to 0.	RW	0x1

Table 4-333. Register Call Summary for Register RM_RSTST_CORE

PRCM Basic Programming Model
- Reset Control: [0] [1]

PRCM Register Manual
- CORE_PRM Registers: [2]

Table 4-334. PM_WKEN1_CORE

Address Offset	0x0000 00A0		
Physical Address	0x4830 6AA0	**Instance**	CORE_PRM
Description	This register allows enabling/disabling modules wake-up events.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED	EN_MMC3	RESERVED				EN_MMC2	EN_MMC1	RESERVED		EN_MCSPI4	EN_MCSPI3	EN_MCSPI2	EN_MCSPI1	EN_I2C3	EN_I2C2	EN_I2C1	EN_UART2	EN_UART1	EN_GPT11	EN_GPT10	EN_MCBSP5	EN_MCBSP1	RESERVED				EN_HSOTGUSB	RESERVED			

Bits	Field Name	Description	Type	Reset
31	RESERVED	Write 1's for future compatibility. Read returns 1.	RW	0x1
30	EN_MMC3	MMC SDIO 3 wake-up control 0x0: MMC 3 wake-up is disabled 0x1: MMC 3 wake-up event is enabled	RW	0x1
29:26	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x0
25	EN_MMC2	MMC SDIO 2 wake-up control 0x0: MMC 2 wake-up is disabled 0x1: MMC 2 wake-up event is enabled	RW	0x1
24	EN_MMC1	MMC SDIO 1 wake-up control 0x0: MMC 1 wake-up is disabled 0x1: MMC 1 wake-up event is enabled	RW	0x1
23:22	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x0
21	EN_MCSPI4	McSPI 4 wake-up control 0x0: McSPI 4 wake-up is disabled 0x1: McSPI 4 wake-up event is enabled	RW	0x1
20	EN_MCSPI3	McSPI 3 wake-up control 0x0: McSPI 3 wake-up is disabled 0x1: McSPI 3 wake-up event is enabled	RW	0x1
19	EN_MCSPI2	McSPI 2 wake-up control 0x0: McSPI 2 wake-up is disabled 0x1: McSPI 2 wake-up event is enabled	RW	0x1
18	EN_MCSPI1	McSPI 1 wake-up control 0x0: McSPI 1 wake-up is disabled 0x1: McSPI 1 wake-up event is enabled	RW	0x1
17	EN_I2C3	I2C 3 wake-up control 0x0: I2C 3 wake-up is disabled 0x1: I2C 3 wake-up event is enabled	RW	0x1

Bits	Field Name	Description	Type	Reset
16	EN_I2C2	I2C 2 wake-up control 0x0: I2C 2 wake-up is disabled 0x1: I2C 2 wake-up event is enabled	RW	0x1
15	EN_I2C1	I2C 1 wake-up control 0x0: I2C 1 wake-up is disabled 0x1: I2C 1 wake-up event is enabled	RW	0x1
14	EN_UART2	UART 2 wake-up control 0x0: UART 2 wake-up is disabled 0x1: UART 2 wake-up event is enabled	RW	0x1
13	EN_UART1	UART 1 wake-up control 0x0: UART 1 wake-up is disabled 0x1: UART 1 wake-up event is enabled	RW	0x1
12	EN_GPT11	GPTIMER 11 wake-up control 0x0: GPTIMER 11 wake-up is disabled 0x1: GPTIMER 11 wake-up event is enabled	RW	0x1
11	EN_GPT10	GPTIMER 10 wake-up control 0x0: GPTIMER 10 wake-up is disabled 0x1: GPTIMER 10 wake-up event is enabled	RW	0x1
10	EN_MCBSP5	McBSP 5 wake-up control 0x0: McBSP 5 wake-up is disabled 0x1: McBSP 5 wake-up event is enabled	RW	0x1
9	EN_MCBSP1	McBSP 1 wake-up control 0x0: McBSP 1 wake-up is disabled 0x1: McBSP 1 wake-up event is enabled	RW	0x1
8:5	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x0
4	EN_HSOTGUSB	HS OTG USB wake-up control 0x0: HS OTG USB wake-up is disabled 0x1: HS OTG USB wake-up event is enabled	RW	0x1
3:0	RESERVED	Write 0x8 for future compatibility. Read returns 0x8.	R	0x8

Table 4-335. Register Call Summary for Register PM_WKEN1_CORE

PRCM Idle and Wake-Up Management • Device Wake-Up Events: [0]
PRCM Basic Programming Model • Domain Wake-Up Control Registers: [1]
PRCM Register Manual • CORE_PRM Registers: [2]

Table 4-336. PM_MPUGRPSEL1_CORE

Address Offset	0x0000 00A4		
Physical Address	0x4830 6AA4	**Instance**	CORE_PRM
Description	This register allows selecting the group of modules that wake-up the MPU.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED	GRPSEL_MMC3	RESERVED				GRPSEL_MMC2	GRPSEL_MMC1	RESERVED		GRPSEL_MCSPI4	GRPSEL_MCSPI3	GRPSEL_MCSPI2	GRPSEL_MCSPI1	GRPSEL_I2C3	GRPSEL_I2C2	GRPSEL_I2C1	GRPSEL_UART2	GRPSEL_UART1	GRPSEL_GPT11	GRPSEL_GPT10	GRPSEL_MCBSP5	GRPSEL_MCBSP1	RESERVED				GRPSEL_HSOTGUSB	RESERVED			

Bits	Field Name	Description	Type	Reset
31	RESERVED	Write 1's for future compatibility. Read returns 1.	RW	0x1
30	GRPSEL_MMC3	Select the MMC 3 in the MPU wake-up events group 0x0: MMC 3 is not attached to the MPU wake-up events group. 0x1: MMC 3 is attached to the MPU wake-up events group.	RW	0x1
29:26	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x0
25	GRPSEL_MMC2	Select the MMC 2 in the MPU wake-up events group 0x0: MMC 2 is not attached to the MPU wake-up events group. 0x1: MMC 2 is attached to the MPU wake-up events group.	RW	0x1
24	GRPSEL_MMC1	Select the MMC 1 in the MPU wake-up events group 0x0: MMC 1 is not attached to the MPU wake-up events group. 0x1: MMC 1 is attached to the MPU wake-up events group.	RW	0x1
23:22	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x0
21	GRPSEL_MCSPI4	Select the McSPI 4 in the MPU wake-up events group 0x0: McSPI 4 is not attached to the MPU wake-up events group. 0x1: McSPI 4 is attached to the MPU wake-up events group.	RW	0x1
20	GRPSEL_MCSPI3	Select the McSPI 3 in the MPU wake-up events group 0x0: McSPI 3 is not attached to the MPU wake-up events group. 0x1: McSPI 3 is attached to the MPU wake-up events group.	RW	0x1
19	GRPSEL_MCSPI2	Select the McSPI 2 in the MPU wake-up events group 0x0: McSPI 2 is not attached to the MPU wake-up events group. 0x1: McSPI 2 is attached to the MPU wake-up events group.	RW	0x1
18	GRPSEL_MCSPI1	Select the McSPI 1 in the MPU wake-up events group 0x0: McSPI 1 is not attached to the MPU wake-up events group. 0x1: McSPI 1 is attached to the MPU wake-up events group.	RW	0x1
17	GRPSEL_I2C3	Select the I2C 3 in the MPU wake-up events group 0x0: I2C 3 is not attached to the MPU wake-up events group. 0x1: I2C 3 is attached to the MPU wake-up events group.	RW	0x1
16	GRPSEL_I2C2	Select the I2C 2 in the MPU wake-up events group 0x0: I2C 2 is not attached to the MPU wake-up events group. 0x1: I2C 2 is attached to the MPU wake-up events group.	RW	0x1

Bits	Field Name	Description	Type	Reset
15	GRPSEL_I2C1	Select the I2C 1 in the MPU wake-up events group 0x0: I2C 1 is not attached to the MPU wake-up events group. 0x1: I2C 1 is attached to the MPU wake-up events group.	RW	0x1
14	GRPSEL_UART2	Select the UART 2 in the MPU wake-up events group 0x0: UART 2 is not attached to the MPU wake-up events group. 0x1: UART 2 is attached to the MPU wake-up events group.	RW	0x1
13	GRPSEL_UART1	Select the UART 1 in the MPU wake-up events group 0x0: UART 1 is not attached to the MPU wake-up events group. 0x1: UART 1 is attached to the MPU wake-up events group.	RW	0x1
12	GRPSEL_GPT11	Select the GPTIMER 11 in the MPU wake-up events group 0x0: GPTIMER 11 is not attached to the MPU wake-up events group. 0x1: GPTIMER 11 is attached to the MPU wake-up events group.	RW	0x1
11	GRPSEL_GPT10	Select the GPTIMER 10 in the MPU wake-up events group 0x0: GPTIMER 10 is not attached to the MPU wake-up events group. 0x1: GPTIMER 10 is attached to the MPU wake-up events group.	RW	0x1
10	GRPSEL_MCBSP5	Select the McBSP 5 in the MPU wake-up events group 0x0: McBSP 5 is not attached to the MPU wake-up events group. 0x1: McBSP 5 is attached to the MPU wake-up events group.	RW	0x1
9	GRPSEL_MCBSP1	Select the McBSP 1 in the MPU wake-up events group 0x0: McBSP 1 is not attached to the MPU wake-up events group. 0x1: McBSP 1 is attached to the MPU wake-up events group.	RW	0x1
8:5	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x0
4	GRPSEL_HSOTGUSB	Select the HS OTG USB in the MPU wake-up events group 0x0: HS OTG USB is not attached to the MPU wake-up events group. 0x1: HS OTG USB is attached to the MPU wake-up events group.	RW	0x1
3:0	RESERVED	Write 0x8 for future compatibility. Read returns 0x8.	R	0x8

Table 4-337. Register Call Summary for Register PM_MPUGRPSEL1_CORE

PRCM Idle and Wake-Up Management • Device Wake-Up Events: [0]
PRCM Basic Programming Model • Domain Wake-Up Control Registers: [1]
PRCM Register Manual • CORE_PRM Registers: [2]

Table 4-338. PM_IVA2GRPSEL1_CORE

Address Offset	0x0000 00A8		
Physical Address	0x4830 6AA8	**Instance**	CORE_PRM
Description	This register allows selecting the group of modules that wake-up the IVA2.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED	GRPSEL_MMC3	RESERVED				GRPSEL_MMC2	GRPSEL_MMC1	RESERVED		GRPSEL_MCSPI4	GRPSEL_MCSPI3	GRPSEL_MCSPI2	GRPSEL_MCSPI1	GRPSEL_I2C3	GRPSEL_I2C2	GRPSEL_I2C1	GRPSEL_UART2	GRPSEL_UART1	GRPSEL_GPT11	GRPSEL_GPT10	GRPSEL_MCBSP5	GRPSEL_MCBSP1	RESERVED				GRPSEL_HSOTGUSB	RESERVED			

Bits	Field Name	Description	Type	Reset
31	RESERVED	Write 1's for future compatibility. Read returns 1.	RW	0x1
30	GRPSEL_MMC3	Select the MMC 3 in the IVA2 wake-up events group 0x0: MMC 3 is not attached to the IVA2 wake-up events group. 0x1: MMC 3 is attached to the IVA2 wake-up events group.	RW	0x1
29:26	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x0
25	GRPSEL_MMC2	Select the MMC 2 in the IVA2 wake-up events group 0x0: MMC 2 is not attached to the IVA2 wake-up events group. 0x1: MMC 2 is attached to the IVA2 wake-up events group.	RW	0x1
24	GRPSEL_MMC1	Select the MMC 1 in the IVA2 wake-up events group 0x0: MMC 1 is not attached to the IVA2 wake-up events group. 0x1: MMC 1 is attached to the IVA2 wake-up events group.	RW	0x1
23:22	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x0
21	GRPSEL_MCSPI4	Select the McSPI 4 in the IVA2 wake-up events group 0x0: McSPI 4 is not attached to the IVA2 wake-up events group. 0x1: McSPI 4 is attached to the IVA2 wake-up events group.	RW	0x1
20	GRPSEL_MCSPI3	Select the McSPI 3 in the IVA2 wake-up events group 0x0: McSPI 3 is not attached to the IVA2 wake-up events group. 0x1: McSPI 3 is attached to the IVA2 wake-up events group.	RW	0x1
19	GRPSEL_MCSPI2	Select the McSPI 2 in the IVA2 wake-up events group 0x0: McSPI 2 is not attached to the IVA2 wake-up events group. 0x1: McSPI 2 is attached to the IVA2 wake-up events group.	RW	0x1
18	GRPSEL_MCSPI1	Select the McSPI 1 in the IVA2 wake-up events group 0x0: McSPI 1 is not attached to the IVA2 wake-up events group. 0x1: McSPI 1 is attached to the IVA2 wake-up events group.	RW	0x1

Bits	Field Name	Description	Type	Reset
17	GRPSEL_I2C3	Select the I2C 3 in the IVA2 wake-up events group 0x0: I2C 3 is not attached to the IVA2 wake-up events group. 0x1: I2C 3 is attached to the IVA2 wake-up events group.	RW	0x1
16	GRPSEL_I2C2	Select the I2C 2 in the IVA2 wake-up events group 0x0: I2C 2 is not attached to the IVA2 wake-up events group. 0x1: I2C 2 is attached to the IVA2 wake-up events group.	RW	0x1
15	GRPSEL_I2C1	Select the I2C 1 in the IVA2 wake-up events group 0x0: I2C 1 is not attached to the IVA2 wake-up events group. 0x1: I2C 1 is attached to the IVA2 wake-up events group.	RW	0x1
14	GRPSEL_UART2	Select the UART 2 in the IVA2 wake-up events group 0x0: UART 2 is not attached to the IVA2 wake-up events group. 0x1: UART 2 is attached to the IVA2 wake-up events group.	RW	0x1
13	GRPSEL_UART1	Select the UART 1 in the IVA2 wake-up events group 0x0: UART 1 is not attached to the IVA2 wake-up events group. 0x1: UART 1 is attached to the IVA2 wake-up events group.	RW	0x1
12	GRPSEL_GPT11	Select the GPTIMER 11 in the IVA2 wake-up events group 0x0: GPTIMER 11 is not attached to the IVA2 wake-up events group. 0x1: GPTIMER 11 is attached to the IVA2 wake-up events group.	RW	0x1
11	GRPSEL_GPT10	Select the GPTIMER 10 in the IVA2 wake-up events group 0x0: GPTIMER 10 is not attached to the IVA2 wake-up events group. 0x1: GPTIMER 10 is attached to the IVA2 wake-up events group.	RW	0x1
10	GRPSEL_MCBSP5	Select the McBSP 5 in the IVA2 wake-up events group 0x0: McBSP 5 is not attached to the IVA2 wake-up events group. 0x1: McBSP 5 is attached to the IVA2 wake-up events group.	RW	0x1
9	GRPSEL_MCBSP1	Select the McBSP 1 in the IVA2 wake-up events group 0x0: McBSP 1 is not attached to the IVA2 wake-up events group. 0x1: McBSP 1 is attached to the IVA2 wake-up events group.	RW	0x1
8:5	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x0
4	GRPSEL_HSOTGUSB	Select the HS OTG USB in the IVA2 wake-up events group 0x0: HS OTG USB is not attached to the IVA2 wake-up events group. 0x1: HS OTG USB is attached to the IVA2 wake-up events group.	RW	0x1
3:0	RESERVED	Write 0x8 for future compatibility. Read returns 0x8.	R	0x8

Table 4-339. Register Call Summary for Register PM_IVA2GRPSEL1_CORE

PRCM Idle and Wake-Up Management • Device Wake-Up Events: [0]
PRCM Basic Programming Model • Domain Wake-Up Control Registers: [1]
PRCM Register Manual • CORE_PRM Registers: [2]

Table 4-340. PM_WKST1_CORE

Address Offset	0x0000 00B0		
Physical Address	0x4830 6AB0	**Instance**	CORE_PRM
Description	This register logs module wake-up events. Must be cleared by software. If it is not cleared, it prevents further domain transition.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED	ST_MMC3	RESERVED				ST_MMC2	ST_MMC1	RESERVED		ST_MCSPI4	ST_MCSPI3	ST_MCSPI2	ST_MCSPI1	ST_I2C3	ST_I2C2	ST_I2C1	ST_UART2	ST_UART1	ST_GPT11	ST_GPT10	ST_MCBSP5	ST_MCBSP1	RESERVED				ST_HSOTGUSB	RESERVED			

Bits	Field Name	Description	Type	Reset
31	RESERVED	Write 0s for future compatibility. Read returns 0.	RW	0x0
30	ST_MMC3	MMC 3 Wake-up status Read 0x0: MMC 3 wake-up did not occur or was masked. Write 0x0: Status bit unchanged Read 0x1: MMC 3 wake-up occurred. Write 0x1: Status bit is cleared to 0.	RW	0x0
29:26	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x0
25	ST_MMC2	MMC 2 Wake-up status Read 0x0: MMC 2 wake-up did not occur or was masked. Write 0x0: Status bit unchanged Read 0x1: MMC 2 wake-up occurred. Write 0x1: Status bit is cleared to 0.	RW	0x0
24	ST_MMC1	MMC 1 Wake-up status Read 0x0: MMC 1 wake-up did not occur or was masked. Write 0x0: Status bit unchanged Read 0x1: MMC 1 wake-up occurred. Write 0x1: Status bit is cleared to 0.	RW	0x0
23:22	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x0
21	ST_MCSPI4	McSPI 4 Wake-up status Read 0x0: McSPI 4 wake-up did not occur or was masked. Write 0x0: Status bit unchanged Read 0x1: McSPI 4 wake-up occurred. Write 0x1: Status bit is cleared to 0.	RW	0x0

Bits	Field Name	Description	Type	Reset
20	ST_MCSPI3	McSPI 3 Wake-up status Read 0x0: McSPI 3 wake-up did not occur or was masked. Write 0x0: Status bit unchanged Read 0x1: McSPI 3 wake-up occurred. Write 0x1: Status bit is cleared to 0.	RW	0x0
19	ST_MCSPI2	McSPI 2 Wake-up status Read 0x0: McSPI 2 wake-up did not occur or was masked. Write 0x0: Status bit unchanged Read 0x1: McSPI 2 wake-up occurred. Write 0x1: Status bit is cleared to 0.	RW	0x0
18	ST_MCSPI1	McSPI 1 Wake-up status Read 0x0: McSPI 1 wake-up did not occur or was masked. Write 0x0: Status bit unchanged Read 0x1: McSPI 1 wake-up occurred. Write 0x1: Status bit is cleared to 0.	RW	0x0
17	ST_I2C3	I2C 3 Wake-up status Read 0x0: I2C 3 wake-up did not occur or was masked. Write 0x0: Status bit unchanged Read 0x1: I2C 3 wake-up occurred. Write 0x1: Status bit is cleared to 0.	RW	0x0
16	ST_I2C2	I2C 2 Wake-up status Read 0x0: I2C 2 wake-up did not occur or was masked. Write 0x0: Status bit unchanged Read 0x1: I2C 2 wake-up occurred. Write 0x1: Status bit is cleared to 0.	RW	0x0
15	ST_I2C1	I2C 1 Wake-up status Read 0x0: I2C 1 wake-up did not occur or was masked. Write 0x0: Status bit unchanged Read 0x1: I2C 1 wake-up occurred. Write 0x1: Status bit is cleared to 0.	RW	0x0
14	ST_UART2	UART 2 Wake-up status Read 0x0: UART 2 wake-up did not occur or was masked. Write 0x0: Status bit unchanged Read 0x1: UART 2 wake-up occurred. Write 0x1: Status bit is cleared to 0.	RW	0x0
13	ST_UART1	UART 1 Wake-up status Read 0x0: UART 1 wake-up did not occur or was masked. Write 0x0: Status bit unchanged Read 0x1: UART 1 wake-up occurred. Write 0x1: Status bit is cleared to 0.	RW	0x0
12	ST_GPT11	GPTIMER 11 Wake-up status Read 0x0: GPTIMER 11 wake-up did not occur or was masked. Write 0x0: Status bit unchanged Read 0x1: Status bit is cleared to 0. Write 0x1: GPTIMER 11 wake-up occurred.	RW	0x0

Bits	Field Name	Description	Type	Reset
11	ST_GPT10	GPTIMER 10 Wake-up status Read 0x0: GPTIMER 10 wake-up did not occur or was masked. Write 0x0: Status bit unchanged Read 0x1: GPTIMER 10 wake-up occurred. Write 0x1: Status bit is cleared to 0.	RW	0x0
10	ST_MCBSP5	McBSP 5 Wake-up status Read 0x0: McBSP 5 wake-up did not occur or was masked. Write 0x0: Status bit unchanged Read 0x1: McBSP 5 wake-up occurred. Write 0x1: Status bit is cleared to 0.	RW	0x0
9	ST_MCBSP1	McBSP 1 Wake-up status Read 0x0: McBSP 1 wake-up did not occur or was masked. Write 0x0: Status bit unchanged Read 0x1: McBSP 1 wake-up occurred. Write 0x1: Status bit is cleared to 0.	RW	0x0
8:5	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x0
4	ST_HSOTGUSB	HS OTG USB Wake-up status Read 0x0: HS OTG USB wake-up did not occur or was masked. Write 0x0: Status bit unchanged Read 0x1: HS OTG USB wake-up occurred. Write 0x1: Status bit is cleared to 0.	RW	0x0
3:0	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x0

Table 4-341. Register Call Summary for Register PM_WKST1_CORE

PRCM Basic Programming Model
- Domain Wake-Up Control Registers: [0]

PRCM Register Manual
- CORE_PRM Registers: [1]

Table 4-342. PM_WKST3_CORE

Address Offset	0x0000 00B8		
Physical Address	0x4830 6AB8	**Instance**	CORE_PRM
Description	This register logs module wake-up events. Must be cleared by software. If it is not cleared, it prevents further domain transition.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																													ST_USBTLL	RESERVED	

Bits	Field Name	Description	Type	Reset
31:3	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x00000000
2	ST_USBTLL	USB TLL wake-up status Read 0x0: USB TLL wake-up did not occur or was masked. Write 0x0: Status bit unchanged Read 0x1: USB TLL wake-up occurred. Write 0x1: Status bit is cleared to 0.	RW	0x0
1:0	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x0

Table 4-343. Register Call Summary for Register PM_WKST3_CORE

PRCM Basic Programming Model • Domain Wake-Up Control Registers: [0]
PRCM Register Manual • CORE_PRM Registers: [1]

Table 4-344. PM_PWSTCTRL_CORE

Address Offset	0x0000 00E0		
Physical Address	0x4830 6AE0	**Instance**	CORE_PRM
Description	This register controls the CORE domain power state transition.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED												MEM2ONSTATE		MEM1ONSTATE		RESERVED						MEM2RETSTATE	MEM1RETSTATE	RESERVED			SAVEANDRESTORE	MEMORYCHANGE	LOGICRETSTATE	POWERSTATE	

Bits	Field Name	Description	Type	Reset
31:20	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x000
19:18	MEM2ONSTATE	Memory block 2 state when domain is ON 0x0: Memory block 2 is OFF when domain is ON. 0x1: Memory block 2 is in RETENTION when domain is ON. 0x2: Reserved 0x3: Memory block 2 is ON when domain is ON.	RW	0x3
17:16	MEM1ONSTATE	Memory block 1 state when domain is ON 0x0: Memory block 1 is OFF when domain is ON. 0x1: Memory block 1 is in RETENTION when domain is ON. 0x2: Reserved 0x3: Memory block 1 is ON when domain is ON.	RW	0x3
15:10	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x00
9	MEM2RETSTATE	Memory block 2 state when domain is RETENTION 0x0: Memory block 2 is OFF when domain is in RETENTION state. 0x1: Memory block 2 is retained when domain is in RETENTION state.	RW	0x1

Bits	Field Name	Description	Type	Reset
8	MEM1RETSTATE	Memory block 1 state when domain is RETENTION 0x0: Memory block 1 is OFF when domain is in RETENTION state. 0x1: Memory block 1 is retained when domain is in RETENTION state.	RW	0x1
7:5	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x0
4	SAVEANDRESTORE	Save And Restore mechanism for the USB TLL module 0x0: Disable the save and restore mechanism for the USB TLL module 0x1: Enable the save and restore mechanism for the USB TLL module	RW	0x0
3	MEMORYCHANGE	Memory change control in ON state 0x0: Disable memory change. 0x1: Enable memory change state in ON state. This bit is automaticaly cleared when memory state is effectively changed.	RW	0x0
2	LOGICRETSTATE	Logic state when domain is RETENTION 0x0: Logic build with retention flip-flop (SMS, SDRC, control module, clock management) is retained and remaining logic is OFF when domain is in RETENTION state. 0x1: Logic is retained when domain is in RETENTION state.	RW	0x1
1:0	POWERSTATE	Power state control 0x0: OFF state 0x1: RETENTION state 0x2: Reserved 0x3: ON state	RW	0x3

Table 4-345. Register Call Summary for Register PM_PWSTCTRL_CORE

PRCM Power Manager Functional Description • Device Power Domains: [0] [1] [2]
PRCM Clock Manager Functional Description • CORE Power Domain Clock Controls: [3]
PRCM Idle and Wake-Up Management • USBHOST/USBTLL Save-and-Restore Management: [4] • USB TLL SAR Sequences: [5] [6] [7]
PRCM Basic Programming Model • PM_PWSTCTRL_ <domain_name> (Power State Control Register): [8]
PRCM Register Manual • CORE_PRM Registers: [9] [10] [11]

Note: Before initiating a domain sleep transition (ON to OFF) on the MPU and CORE power domains, the CORE memory banks must be configured to automatically switch to ON power state when the CORE power domain wakes up (OFF to ON). This is required for correct booting.

This is done by setting the PM_PWSTCTRL_CORE [19:18] MEM2ONSTATE bit field to 0x3 and the PM_PWSTCTRL_CORE [17:16] MEM1ONSTATE bit field to 0x3."

Table 4-346. PM_PWSTST_CORE

Address Offset	0x0000 00E4		
Physical Address	0x4830 6AE4	**Instance**	CORE_PRM
Description	This register provides a status on the power state transition of the CORE domain.		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED											INTRANSITION	RESERVED												MEM2STATEST		MEM1STATEST		RESERVED	LOGICSTATEST		POWERSTATEST	

Bits	Field Name	Description	Type	Reset
31:21	RESERVED	Read returns 0.	R	0x000
20	INTRANSITION	Domain transition status 0x0: No transition 0x1: CORE power domain transition is in progress.	R	0x0
19:8	RESERVED	Read returns 0.	R	0x000
7:6	MEM2STATEST	Memory block 2 state status 0x0: Memory is OFF 0x1: Memory is in RETENTION 0x2: Reserved 0x3: Memory is ON	R	0x3
5:4	MEM1STATEST	Memory block 1 state status 0x0: Memory is OFF 0x1: Memory is in RETENTION 0x2: Reserved 0x3: Memory is ON	R	0x3
3	RESERVED	Read returns 0.	R	0x0
2	LOGICSTATEST	Logic state status 0x0: CORE domain logic is OFF 0x1: CORE domain logic is ON	R	0x1
1:0	POWERSTATEST	Current power state status 0x0: Power domain is OFF 0x1: Power domain is in RETENTION 0x2: Power domain is INACTIVE 0x3: Power domain is ON	R	0x3

Table 4-347. Register Call Summary for Register PM_PWSTST_CORE

PRCM Basic Programming Model
- PM_PWSTST_ <domain_name> (Power State Status Register): [0]

PRCM Register Manual
- CORE_PRM Registers: [1]

Table 4-348. PM_PREPWSTST_CORE

Address Offset	0x0000 00E8		
Physical Address	0x4830 6AE8	**Instance**	CORE_PRM
Description	This register provides a status on the CORE domain previous power state. It indicates the state entered during the last sleep transition.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																								LASTMEM2STATEENTERED		LASTMEM1STATEENTERED		RESERVED	LASTLOGICSTATEENTERED	LASTPOWERSTATEENTERED	

Bits	Field Name	Description	Type	Reset
31:8	RESERVED	Read returns 0.	R	0x000000
7:6	LASTMEM2STATEENTERED	Last Memory block 2 state entered 0x0: Memory was previously OFF 0x1: Memory was previously in RETENTION 0x2: Reserved 0x3: Memory was previously ON	RW	0x0
5:4	LASTMEM1STATEENTERED	Last Memory block 1 state entered 0x0: Memory was previously OFF 0x1: Memory was previously in RETENTION 0x2: Reserved 0x3: Memory was previously ON	RW	0x0
3	RESERVED	Read returns 0.	R	0x0
2	LASTLOGICSTATEENTERED	Last logic state entered 0x0: CORE domain logic was previously OFF 0x1: CORE domain logic was previously ON	RW	0x0
1:0	LASTPOWERSTATEENTERED	Last power state entered 0x0: CORE domain was previously OFF 0x1: CORE domain was previously in RETENTION 0x2: CORE domain was previously INACTIVE 0x3: CORE domain was previously ON	RW	0x0

Table 4-349. Register Call Summary for Register PM_PREPWSTST_CORE

PRCM Register Manual

- CORE_PRM Registers: [0]

Table 4-350. PM_WKEN3_CORE

Address Offset	0x0000 00F0		
Physical Address	0x4830 6AF0	**Instance**	CORE_PRM
Description	This register allows enabling/disabling modules wake-up events.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																													EN_USBTLL	RESERVED	

Bits	Field Name	Description	Type	Reset
31:3	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x00000000
2	EN_USBTLL	USB TLL wake-up control 0x0: USB TLL wake-up is disabled 0x1: USB TLL wake-up event is enabled	RW	0x1
1:0	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x0

Table 4-351. Register Call Summary for Register PM_WKEN3_CORE

PRCM Basic Programming Model
- Domain Wake-Up Control Registers: [0]

PRCM Register Manual
- CORE_PRM Registers: [1]

Table 4-352. PM_IVA2GRPSEL3_CORE

Address Offset	0x0000 00F4		
Physical Address	0x4830 6AF4	**Instance**	CORE_PRM
Description	This register allows selecting the group of modules that wake-up the IVA2.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																													GRPSEL_USBTLL	RESERVED	

Bits	Field Name	Description	Type	Reset
31:3	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x00000000
2	GRPSEL_USBTLL	Select the USB TLL in the IVA2 wake-up events group 0x0: USB TLL is not attached to the IVA2 wake-up events group. 0x1: USB TLL is attached to the IVA2 wake-up events group.	RW	0x1
1:0	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x0

Table 4-353. Register Call Summary for Register PM_IVA2GRPSEL3_CORE

PRCM Idle and Wake-Up Management
- Device Wake-Up Events: [0]

Table 4-353. Register Call Summary for Register PM_IVA2GRPSEL3_CORE (continued)

PRCM Basic Programming Model • Domain Wake-Up Control Registers: [1]
PRCM Register Manual • CORE_PRM Registers: [2]

Table 4-354. PM_MPUGRPSEL3_CORE

Address Offset	0x0000 00F8		
Physical Address	0x4830 6AF8	**Instance**	CORE_PRM
Description	This register allows selecting the group of modules that wake-up the MPU.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																													GRPSEL_USBTLL	RESERVED	

Bits	Field Name	Description	Type	Reset
31:3	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x00000000
2	GRPSEL_USBTLL	Select the USB TLL in the MPU wake-up events group 0x0: USB TLL is not attached to the MPU wake-up events group. 0x1: USB TLL is attached to the MPU wake-up events group.	RW	0x1
1:0	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x0

Table 4-355. Register Call Summary for Register PM_MPUGRPSEL3_CORE

PRCM Idle and Wake-Up Management • Device Wake-Up Events: [0]
PRCM Basic Programming Model • Domain Wake-Up Control Registers: [1]
PRCM Register Manual • CORE_PRM Registers: [2]

4.14.2.6 SGX_PRM Registers

Table 4-356. SGX_PRM Register Summary

Register Name	Type	Register Width (Bits)	Address Offset	Physical Address	Reset Type
RM_RSTST_SGX	RW	32	0x0000 0058	0x4830 6B58	C
PM_WKDEP_SGX	RW	32	0x0000 00C8	0x4830 6BC8	W
PM_PWSTCTRL_SGX	RW	32	0x0000 00E0	0x4830 6BE0	W
PM_PWSTST_SGX	R	32	0x0000 00E4	0x4830 6BE4	C
PM_PREPWSTST_SGX	RW	32	0x0000 00E8	0x4830 6BE8	C

Table 4-357. RM_RSTST_SGX

Address Offset	0x0000 0058		
Physical Address	0x4830 6B58	**Instance**	SGX_PRM
Description	This register logs the different reset sources of the SGX domain. Each bit is set upon release of the domain reset signal. Must be cleared by software.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																												COREDOMAINWKUP_RST	DOMAINWKUP_RST	GLOBALWARM_RST	GLOBALCOLD_RST

Bits	Field Name	Description	Type	Reset
31:4	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x0000000
3	COREDOMAINWKUP_RST	CORE domain wake-up reset Read 0x0: No power domain wake-up reset. Write 0x0: Status bit unchanged Read 0x1: SGX domain has been reset following a CORE power domain wake-up from OFF to ON. Write 0x1: Status bit is cleared to 0.	RW	0x0
2	DOMAINWKUP_RST	Power domain wake-up reset Read 0x0: SGX domain is not reset Write 0x0: Status bit unchanged Read 0x1: SGX domain has been reset following a SGX domain wake-up. Write 0x1: Status bit is cleared to 0.	RW	0x0
1	GLOBALWARM_RST	Global warm reset Read 0x0: No global warm reset. Write 0x0: Status bit unchanged Read 0x1: SGX domain has been reset upon a global warm reset Write 0x1: Status bit is cleared to 0.	RW	0x0
0	GLOBALCOLD_RST	Global cold reset Read 0x0: No global cold reset. Write 0x0: Status bit unchanged Read 0x1: SGX domain has been reset upon a global cold reset Write 0x1: Status bit is cleared to 0.	RW	0x1

Table 4-358. Register Call Summary for Register RM_RSTST_SGX

PRCM Basic Programming Model
- Reset Control: [0]

PRCM Register Manual
- SGX_PRM Registers: [1]

Table 4-359. PM_WKDEP_SGX

Address Offset	0x0000 00C8		
Physical Address	0x4830 6BC8	**Instance**	SGX_PRM
Description	This register allows enabling or disabling the wake-up of the SGX domain upon another domain wakeup.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																											EN_WKUP	RESERVED	EN_IVA2	EN_MPU	RESERVED

Bits	Field Name	Description	Type	Reset
31:5	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x0000000
4	EN_WKUP	WAKEUP domain dependency 0x0: SGX domain is independent of WKUP domain wake-up event. 0x1: SGX domain is woken-up upon WKUP domain wake-up event.	RW	0x1
3	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x0
2	EN_IVA2	IVA2 domain dependency 0x0: SGX domain is independent of IVA2 domain wake-up event. 0x1: SGX domain is woken-up upon IVA2 domain wake-up event.	RW	0x1
1	EN_MPU	MPU domain dependency 0x0: SGX domain is independent of MPU domain wake-up. 0x1: SGX domain is woken-up upon MPU domain wake-up.	RW	0x1
0	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x0

Table 4-360. Register Call Summary for Register PM_WKDEP_SGX

PRCM Idle and Wake-Up Management
- Device Wake-Up Events: [0] [1] [2]

PRCM Register Manual
- SGX_PRM Registers: [3]

Table 4-361. PM_PWSTCTRL_SGX

Address Offset	0x0000 00E0		
Physical Address	0x4830 6BE0	**Instance**	SGX_PRM
Description	This register controls the SGX domain power state transition.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED														MEMONSTATE		RESERVED							MEMRETSTATE	RESERVED					LOGICRETSTATE	POWERSTATE	

Bits	Field Name	Description	Type	Reset
31:18	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x0000
17:16	MEMONSTATE	Memory state when ON 0x3: Memory is always ON when domain is ON.	R	0x3
15:9	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x00
8	MEMRETSTATE	Memory state when RETENTION 0x1: Memory is always retained when domain is in RETENTION state.	R	0x1
7:3	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x00
2	LOGICRETSTATE	Logic state when RETENTION 0x1: Logic is always retained when domain is in RETENTION state.	R	0x1
1:0	POWERSTATE	Power state control 0x0: OFF state 0x1: RETENTION state 0x2: Reserved 0x3: ON state	RW	0x3

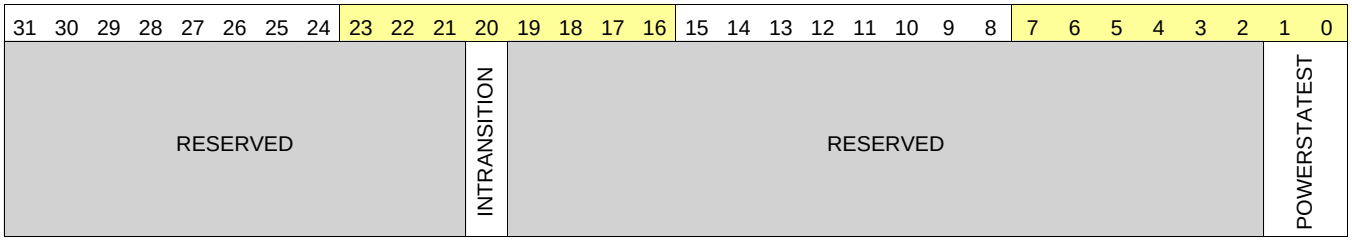
Table 4-362. Register Call Summary for Register PM_PWSTCTRL_SGX

PRCM Basic Programming Model
- PM_PWSTCTRL_ <domain_name> (Power State Control Register): [0]

PRCM Register Manual
- SGX_PRM Registers: [1]

Table 4-363. PM_PWSTST_SGX

Address Offset	0x0000 00E4		
Physical Address	0x4830 6BE4	**Instance**	SGX_PRM
Description	This register provides a status on the power state transition of the SGX domain.		
Type	R		



Bits	Field Name	Description	Type	Reset
31:21	RESERVED	Read returns 0.	R	0x000
20	INTRANSITION	Domain transition status 0x0: No transition 0x1: SGX power domain transition is in progress.	R	0x0
19:2	RESERVED	Read returns 0.	R	0x00000
1:0	POWERSTATEST	Current power state status 0x0: Power domain is OFF 0x1: Power domain is in RETENTION 0x2: Power domain is INACTIVE 0x3: Power domain is ON	R	0x3

Table 4-364. Register Call Summary for Register PM_PWSTST_SGX

PRCM Basic Programming Model • PM_PWSTST_ <domain_name> (Power State Status Register): [0]
PRCM Register Manual • SGX_PRM Registers: [1]

Table 4-365. PM_PREPWSTST_SGX

Address Offset	0x0000 00E8		
Physical Address	0x4830 6BE8	**Instance**	SGX_PRM
Description	This register provides a status on the SGX domain previous power state. It indicates the state entered during the last sleep transition.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																														LASTPOWERSTATEENTERED	

Bits	Field Name	Description	Type	Reset
31:2	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x00000000
1:0	LASTPOWERSTATEENTERED	Last power state entered	RW	0x0
		0x0: SGX domain was previously OFF		
		0x1: SGX domain was previously in RETENTION		
		0x2: SGX domain was previously INACTIVE		
		0x3: SGX domain was previously ON		

Table 4-366. Register Call Summary for Register PM_PREPWSTST_SGX

PRCM Register Manual • SGX_PRM Registers: [0]

4.14.2.7 WKUP_PRM Registers

Table 4-367. WKUP_PRM Register Summary

Register Name	Type	Register Width (Bits)	Address Offset	Physical Address	Reset Type
PM_WKEN_WKUP	RW	32	0x0000 00A0	0x4830 6CA0	W
PM_MPUGRPSEL_WKUP	RW	32	0x0000 00A4	0x4830 6CA4	W
PM_IVA2GRPSEL_WKUP	RW	32	0x0000 00A8	0x4830 6CA8	W
PM_WKST_WKUP	RW	32	0x0000 00B0	0x4830 6CB0	C

Table 4-368. PM_WKEN_WKUP

Address Offset	0x0000 00A0		
Physical Address	0x4830 6CA0	**Instance**	WKUP_PRM
Description	This register allows enabling/disabling modules wake-up events.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED															EN_IO_CHAIN	RESERVED						RESERVED	EN_IO	RESERVED		RESERVED		EN_GPIO1	RESERVED	RESERVED	EN_GPT1

Bits	Field Name	Description	Type	Reset
31:17	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x000000
16	EN_IO_CHAIN	I/O daisy chain wakeup is disabled. 0x0: I/O wake-up daisy chain is disabled. 0x1: I/O wake-up daisy chain event is enabled.	RW	1
15:10	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x000000
9	RESERVED	Reserved for non-GP devices	RW	0x1
8	EN_IO	IO pad wake-up control 0x0: IO pad wakeup is disabled 0x1: IO pad wake-up event is enabled	RW	0x1
7:6	RESERVED	This information is not available in the public domain.	RW	0x3
5:4	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x0
3	EN_GPIO1	GPIO 1 wake-up control 0x0: GPIO 1 wakeup is disabled 0x1: GPIO 1 wake-up event is enabled	RW	0x1
2	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x0
1	RESERVED	Reserved for non-GP devices	R	0x1
0	EN_GPT1	GPTIMER 1 wake-up control 0x0: GPTIMER 1 wakeup is disabled 0x1: GPTIMER 1 wake-up event is enabled	RW	0x1

Table 4-369. Register Call Summary for Register PM_WKEN_WKUP

PRCM Idle and Wake-Up Management
- Device Wake-Up Events: [0] [1]

PRCM Off-Mode Management
- Overview: [2] [3]
- I/O Wake-Up Mechanism: [4] [5] [6] [7]
- Sleep Sequences (Transition From On to Retention/Off): [8] [9]
- Wake-Up Sequences (Transition From Retention/Off to On): [10] [11]
- Sleep Sequences: [12] [13]
- Wake-Up Sequences: [14] [15]

PRCM Basic Programming Model
- Interrupt Configuration Registers: [16]
- Domain Wake-Up Control Registers: [17]

PRCM Register Manual
- WKUP_PRM Registers: [18]

Table 4-370. PM_MPUGRPSEL_WKUP

Address Offset	0x0000 00A4		
Physical Address	0x4830 6CA4	**Instance**	WKUP_PRM
Description	IO pad is always selected in the MPU wake-up events group		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																						RESERVED	GRPSEL_IO	RESERVED		RESERVED		GRPSEL_GPIO1	RESERVED	RESERVED	GRPSEL_GPT1

Bits	Field Name	Description	Type	Reset
31:10	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x000000
9	RESERVED	Reserved for non-GP devices.	RW	0x1
8	GRPSEL_IO	IO pad is always selected in the MPU wake-up events group	R	0x1
7:6	RESERVED	This information is not available in the public domain.	RW	0x3
5:4	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x0
3	GRPSEL_GPIO1	Select the GPIO 1 in the MPU wake-up events group 0x0: GPIO 1 is not attached to the MPU wake-up events group. 0x1: GPIO 1 is attached to the MPU wake-up events group.	RW	0x1
2	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x0
1	RESERVED	Reserved for non-GP devices.	R	0x1
0	GRPSEL_GPT1	Select the GPTIMER 1 in the MPU wake-up events group 0x0: GPTIMER 1 is not attached to the MPU wake-up events group. 0x1: GPTIMER 1 is attached to the MPU wake-up events group.	RW	0x1

Table 4-371. Register Call Summary for Register PM_MPUGRPSEL_WKUP

PRCM Idle and Wake-Up Management • Device Wake-Up Events: [0]
PRCM Basic Programming Model • Domain Wake-Up Control Registers: [1]
PRCM Register Manual • WKUP_PRM Registers: [2]

Table 4-372. PM_IVA2GRPSEL_WKUP

Address Offset	0x0000 00A8		
Physical Address	0x4830 6CA8	**Instance**	WKUP_PRM
Description	This register allows selecting the group of modules that wake-up the IVA2.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																						RESERVED	GRPSEL_IO	RESERVED		RESERVED		GRPSEL_GPIO1	RESERVED	RESERVED	GRPSEL_GPT1

Bits	Field Name	Description	Type	Reset
31:10	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x000000
9	RESERVED	Reserved for non-GP devices.	RW	0x0
8	GRPSEL_IO	Select the IO pad in the IVA2 wake-up events group 0x0: IO pad module is not attached to the IVA2 wake-up events group. 0x1: IO pad module is attached to the IVA2 wake-up events group.	RW	0x0
7:6	RESERVED	This information is not available in the public domain.	RW	0x0
5:4	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x0
3	GRPSEL_GPIO1	Select the GPIO 1 in the IVA2 wake-up events group 0x0: GPIO 1 is not attached to the IVA2 wake-up events group. 0x1: GPIO 1 is attached to the IVA2 wake-up events group.	RW	0x0
2	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x0
1	RESERVED	Reserved for non-GP devices.	RW	0x0
0	GRPSEL_GPT1	Select the GPTIMER 1 in the IVA2 wake-up events group 0x0: GPTIMER 1 is not attached to the IVA2 wake-up events group. 0x1: GPTIMER 1 is attached to the IVA2 wake-up events group.	RW	0x0

Table 4-373. Register Call Summary for Register PM_IVA2GRPSEL_WKUP

PRCM Basic Programming Model

- Interrupt Configuration Registers: [0]
- Domain Wake-Up Control Registers: [1]

PRCM Register Manual

- WKUP_PRM Registers: [2]

Table 4-374. PM_WKST_WKUP

Address Offset	0x0000 00B0		
Physical Address	0x4830 6CB0	**Instance**	WKUP_PRM
Description	This register logs module wake-up events. Must be cleared by software. If it is not cleared, it prevents further domain transition.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED															ST_IO_CHAIN	RESERVED						RESERVED	ST_IO	RESERVED		RESERVED		ST_GPIO1	RESERVED	RESERVED	ST_GPT1

Bits	Field Name	Description	Type	Reset
31:17	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x000000
16	ST_IO_CHAIN	I/O daisy chain wake-up status Read 0x0: I/O daisy chain wakeup did not occur or was masked. Write 0x0: Status bit is unchanged. Write 0x1: Status bit is cleared to 0. Read 0x1: I/O daisy chain wakeup occurred.	RW	0
15:10	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x000000
9	RESERVED	Reserved for non-GP devices	RW	0x0
8	ST_IO	IO pad wake-up status Read 0x0: IO pad wakeup did not occur or was masked. Write 0x0: Status bit unchanged Read 0x1: IO pad wakeup occurred. Write 0x1: Status bit is cleared to 0.	RW	0x0
7:6	RESERVED	This information is not available in the public domain.	RW	0x0
5:4	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x0
3	ST_GPIO1	GPIO 1 Wake-up status Read 0x0: GPIO 1 wakeup did not occur or was masked. Write 0x0: Status bit unchanged Read 0x1: GPIO 1 wakeup occurred. Write 0x1: Status bit is cleared to 0.	RW	0x0
2	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x0
1	RESERVED	Reserved for non-GP devices.	RW	0x0
0	ST_GPT1	GPTIMER 1 wake-up status Read 0x0: GPTIMER 1 wakeup did not occur or was masked. Write 0x0: Status bit unchanged Read 0x1: GPTIMER 1 wakeup occurred. Write 0x1: Status bit is cleared to 0.	RW	0x0

Table 4-375. Register Call Summary for Register PM_WKST_WKUP

PRCM Off-Mode Management
- I/O Wake-Up Mechanism: [0] [1]

PRCM Basic Programming Model
- Domain Wake-Up Control Registers: [2]

Table 4-375. Register Call Summary for Register PM_WKST_WKUP (continued)

PRCM Register Manual • WKUP_PRM Registers: [3]

4.14.2.8 Clock_Control_Reg_PRM Registers

Table 4-376. Clock_Control_Reg_PRM Register Summary

Register Name	Type	Register Width (Bits)	Address Offset	Physical Address	Reset Type
PRM_CLKSEL	RW	32	0x0000 0040	0x4830 6D40	C
PRM_CLKOUT_CTRL	RW	32	0x0000 0070	0x4830 6D70	C

Table 4-377. PRM_CLKSEL

Address Offset	0x0000 0040		
Physical Address	0x4830 6D40	**Instance**	Clock_Control_Reg_PRM
Description	This register controls the selection of the system clock frequency. This register is reset on power-up only.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																														SYS_CLKIN_SEL	

Bits	Field Name	Description	Type	Reset
31:3	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x00000000
2:0	SYS_CLKIN_SEL	System clock input selection; Other enums: Reserved 0x0: OSC_SYS_CLK is 12 MHz 0x1: OSC_SYS_CLK is 13 MHz 0x2: OSC_SYS_CLK is 19.2 MHz 0x3: OSC_SYS_CLK is 26 MHz 0x4: OSC_SYS_CLK is 38.4 MHz 0x5: OSC_SYS_CLK is 16.8 MHz	RW	0x4

Table 4-378. Register Call Summary for Register PRM_CLKSEL

PRCM Clock Manager Functional Description • External Clock Inputs: [0]
PRCM Register Manual • Clock_Control_Reg_PRM Registers: [1]

Table 4-379. PRM_CLKOUT_CTRL

Address Offset	0x0000 0070		
Physical Address	0x4830 6D70	**Instance**	Clock_Control_Reg_PRM
Description	This register provides control over the SYS_CLKOUT1 output clock.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																								CLKOUT_EN	RESERVED						

Bits	Field Name	Description	Type	Reset
31:8	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x000000
7	CLKOUT_EN	This bit controls the external output clock (sys_clkout1) activity 0x0: sys_clkout1 is disabled 0x1: sys_clkout1 is enabled	RW	0x1
6:0	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x00

Table 4-380. Register Call Summary for Register PRM_CLKOUT_CTRL

PRCM Clock Manager Functional Description
- External Output Clock1 (sys_clkout1) Control: [0] [1]
- PRM Source-Clock Controls: [2]

PRCM Register Manual
- Clock_Control_Reg_PRM Registers: [3]

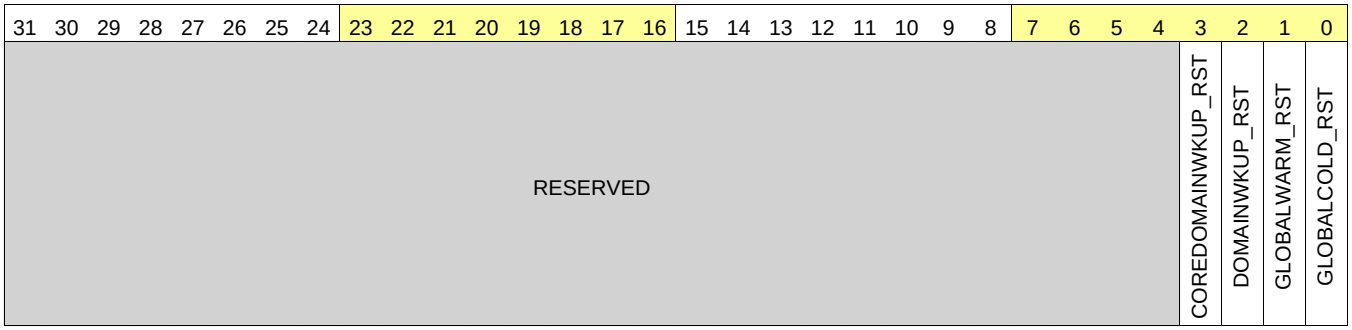
4.14.2.9 DSS_PRM Registers

Table 4-381. DSS_PRM Register Summary

Register Name	Type	Register Width (Bits)	Address Offset	Physical Address	Reset Type
RM_RSTST_DSS	RW	32	0x0000 0058	0x4830 6E58	C
PM_WKEN_DSS	RW	32	0x0000 00A0	0x4830 6EA0	W
PM_WKDEP_DSS	RW	32	0x0000 00C8	0x4830 6EC8	W
PM_PWSTCTRL_DSS	RW	32	0x0000 00E0	0x4830 6EE0	W
PM_PWSTST_DSS	R	32	0x0000 00E4	0x4830 6EE4	C
PM_PREPWSTST_DSS	RW	32	0x0000 00E8	0x4830 6EE8	C

Table 4-382. RM_RSTST_DSS

Address Offset	0x0000 0058		
Physical Address	0x4830 6E58	**Instance**	DSS_PRM
Description	This register logs the different reset sources of the DSS domain. Each bit is set upon release of the domain reset signal. Must be cleared by software.		
Type	RW		



Bits	Field Name	Description	Type	Reset
31:4	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x0000000
3	COREDOMAINWKUP_RST	CORE domain wake-up reset Read 0x0: No power domain wake-up reset. Write 0x0: Status bit unchanged Read 0x1: DSS domain has been reset following a CORE power domain wake-up from OFF to ON. Write 0x1: Status bit is cleared to 0.	RW	0x0
2	DOMAINWKUP_RST	Power domain wake-up reset Read 0x0: Status bit unchanged Write 0x0: No power domain wake-up reset. Read 0x1: DSS domain has been reset following a DSS power domain wake-up. Write 0x1: Status bit is cleared to 0.	RW	0x0
1	GLOBALWARM_RST	Global warm reset Read 0x0: No global warm reset. Write 0x0: Status bit unchanged Read 0x1: DISPLAY domain has been reset upon a global warm reset Write 0x1: Status bit is cleared to 0.	RW	0x0
0	GLOBALCOLD_RST	Global cold reset Read 0x0: No global cold reset. Write 0x0: Status bit unchanged Read 0x1: DISPLAY domain has been reset upon a global cold reset Write 0x1: Status bit is cleared to 0.	RW	0x1

Table 4-383. Register Call Summary for Register RM_RSTST_DSS

PRCM Basic Programming Model

- Reset Control: [0]

PRCM Register Manual

- DSS_PRM Registers: [1]

Table 4-384. PM_WKEN_DSS

Address Offset	0x0000 00A0		
Physical Address	0x4830 6EA0	**Instance**	DSS_PRM
Description	This register allows enabling/disabling modules wake-up events.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																															EN_DSS

Bits	Field Name	Description	Type	Reset
31:1	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x00000000
0	EN_DSS	DSS Wake-up enable 0x0: DSS wake-up is disabled 0x1: DSS wake-up event is enabled	RW	0x1

Table 4-385. Register Call Summary for Register PM_WKEN_DSS

PRCM Basic Programming Model
- Domain Wake-Up Control Registers: [0]

PRCM Register Manual
- DSS_PRM Registers: [1]

Table 4-386. PM_WKDEP_DSS

Address Offset	0x0000 00C8		
Physical Address	0x4830 6EC8	**Instance**	DSS_PRM
Description	This register allows enabling or disabling the wake-up of the DISPLAY domain upon another domain wakeup.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																											EN_WKUP	RESERVED	EN_IVA2	EN_MPU	RESERVED

Bits	Field Name	Description	Type	Reset
31:5	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x0000000
4	EN_WKUP	WAKEUP domain dependency 0x0: DSS domain is independent of WKUP domain wake-up event. 0x1: DSS domain is woken-up upon WKUP domain wake-up event.	RW	0x1
3	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x0
2	EN_IVA2	IVA2 domain dependency 0x0: DSS domain is independent of IVA2 domain wake-up event. 0x1: DSS domain is woken-up upon IVA2 domain wake-up event.	RW	0x1

Bits	Field Name	Description	Type	Reset
1	EN_MPU	MPU domain dependency 0x0: DSS domain is independent of MPU domain wake-up. 0x1: DSS domain is woken-up upon MPU domain wake-up.	RW	0x1
0	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x0

Table 4-387. Register Call Summary for Register PM_WKDEP_DSS

PRCM Idle and Wake-Up Management • Device Wake-Up Events: [0] [1] [2]
PRCM Register Manual • DSS_PRM Registers: [3]

Table 4-388. PM_PWSTCTRL_DSS

Address Offset	0x0000 00E0		
Physical Address	0x4830 6EE0	**Instance**	DSS_PRM
Description	This register controls the DISPLAY domain power state transition.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED														MEMONSTATE		RESERVED							MEMRETSTATE	RESERVED					LOGICRETSTATE	POWERSTATE	

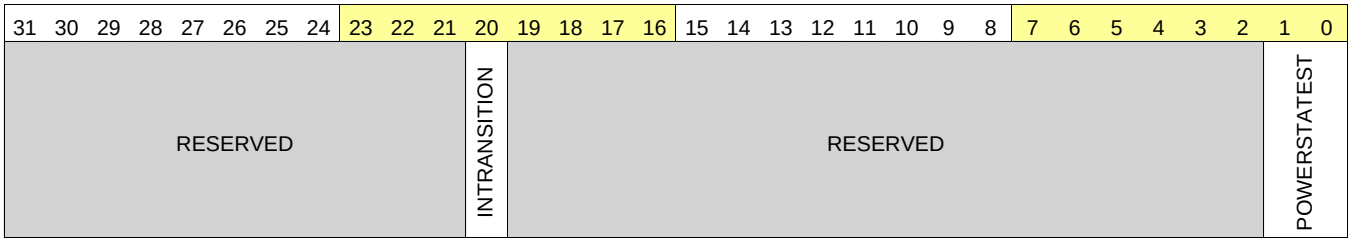
Bits	Field Name	Description	Type	Reset
31:18	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x0000
17:16	MEMONSTATE	Memory state when ON 0x3: Memory is always ON when domain is ON.	R	0x3
15:9	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x00
8	MEMRETSTATE	Memory state when RETENTION 0x1: Memory is always retained when domain is in RETENTION state.	R	0x1
7:3	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x00
2	LOGICRETSTATE	Logic state when RETENTION 0x1: Logic is always retained when domain is in RETENTION state.	R	0x1
1:0	POWERSTATE	Power state control 0x0: OFF state 0x1: RETENTION state 0x2: Reserved 0x3: ON state	RW	0x3

Table 4-389. Register Call Summary for Register PM_PWSTCTRL_DSS

PRCM Basic Programming Model • PM_PWSTCTRL_ <domain_name> (Power State Control Register): [0]
PRCM Register Manual • DSS_PRM Registers: [1]

Table 4-390. PM_PWSTST_DSS

Address Offset	0x0000 00E4		
Physical Address	0x4830 6EE4	**Instance**	DSS_PRM
Description	This register provides a status on the power state transition of the DSS domain.		
Type	R		



Bits	Field Name	Description	Type	Reset
31:21	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x000
20	INTRANSITION	Domain transition status 0x0: No transition 0x1: DISPLAY power domain transition is in progress.	R	0x0
19:2	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x00000
1:0	POWERSTATEST	Current power state status 0x0: Power domain is OFF 0x1: Power domain is in RETENTION 0x2: Power domain is INACTIVE 0x3: Power domain is ON	R	0x3

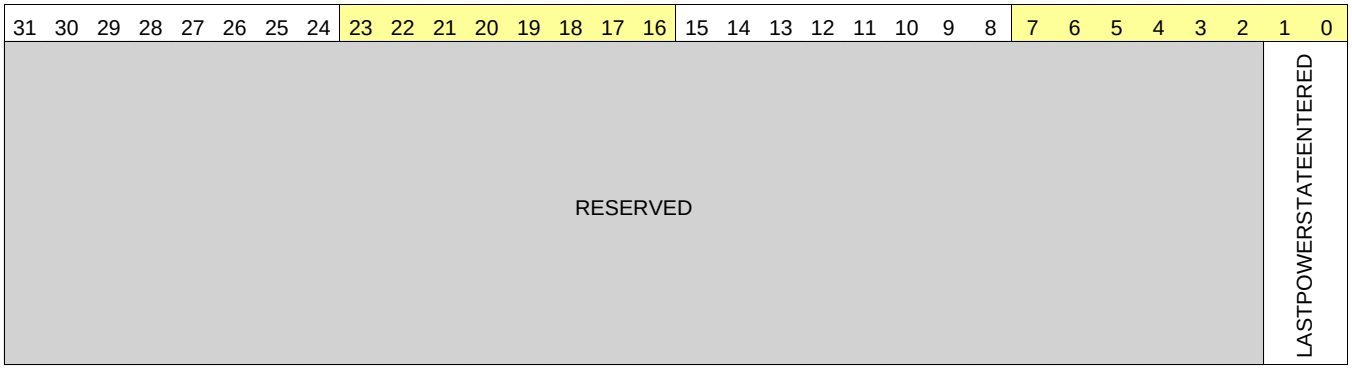
Table 4-391. Register Call Summary for Register PM_PWSTST_DSS

PRCM Basic Programming Model
- PM_PWSTST_ <domain_name> (Power State Status Register): [0]

PRCM Register Manual
- DSS_PRM Registers: [1]

Table 4-392. PM_PREPWSTST_DSS

Address Offset	0x0000 00E8		
Physical Address	0x4830 6EE8	**Instance**	DSS_PRM
Description	This register provides a status on the DSS domain previous power state. It indicates the state entered during the last sleep transition.		
Type	RW		



Bits	Field Name	Description	Type	Reset
31:2	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x00000000
1:0	LASTPOWERSTATEENTERED	Last power state entered	RW	0x0
		0x0: DSS domain was previously OFF		
		0x1: DSS domain was previously in RETENTION		
		0x2: DSS domain was previously INACTIVE		
		0x3: DSS domain was previously ON		

Table 4-393. Register Call Summary for Register PM_PREPWSTST_DSS

PRCM Register Manual
- DSS_PRM Registers: [0]

4.14.2.10 CAM_PRM Registers

Table 4-394. CAM_PRM Register Summary

Register Name	Type	Register Width (Bits)	Address Offset	Physical Address	Reset Type
RM_RSTST_CAM	RW	32	0x0000 0058	0x4830 6F58	C
PM_WKDEP_CAM	RW	32	0x0000 00C8	0x4830 6FC8	W
PM_PWSTCTRL_CAM	RW	32	0x0000 00E0	0x4830 6FE0	W
PM_PWSTST_CAM	R	32	0x0000 00E4	0x4830 6FE4	C
PM_PREPWSTST_CAM	RW	32	0x0000 00E8	0x4830 6FE8	C

Table 4-395. RM_RSTST_CAM

Address Offset	0x0000 0058		
Physical Address	0x4830 6F58	**Instance**	CAM_PRM
Description	This register logs the different reset sources of the CAMERA domain. Each bit is set upon release of the domain reset signal. Must be cleared by software.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																												COREDOMAINWKUP_RST	DOMAINWKUP_RST	GLOBALWARM_RST	GLOBALCOLD_RST

Bits	Field Name	Description	Type	Reset
31:4	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x0000000
3	COREDOMAINWKUP_RST	CORE domain wake-up reset	RW	0x0
		Read 0x0: No power domain wake-up reset.		
		Write 0x0: Status bit unchanged		
		Read 0x1: CAM domain has been reset following a CORE power domain wake-up from OFF to ON.		
		Write 0x1: Status bit is cleared to 0.		

Bits	Field Name	Description	Type	Reset
2	DOMAINWKUP_RST	Power domain wake-up reset Read 0x0: No power domain wake-up reset. Write 0x0: Status bit unchanged Read 0x1: CAM domain has been reset following a CAMERA power domain wake-up. Write 0x1: Status bit is cleared to 0.	RW	0x0
1	GLOBALWARM_RST	Global warm reset Read 0x0: No global warm reset. Write 0x0: Status bit unchanged Read 0x1: CAM domain has been reset upon a global warm reset Write 0x1: Status bit is cleared to 0.	RW	0x0
0	GLOBALCOLD_RST	Global cold reset Read 0x0: No global cold reset. Write 0x0: Status bit unchanged Read 0x1: CAM domain has been reset upon a global cold reset Write 0x1: Status bit is cleared to 0.	RW	0x1

Table 4-396. Register Call Summary for Register RM_RSTST_CAM

PRCM Basic Programming Model • Reset Control: [0]
PRCM Register Manual • CAM_PRM Registers: [1]

Table 4-397. PM_WKDEP_CAM

Address Offset	0x0000 00C8		
Physical Address	0x4830 6FC8	**Instance**	CAM_PRM
Description	This register allows enabling or disabling the wake-up of the CAM domain upon another domain wakeup.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																											EN_WKUP	RESERVED	EN_IVA2	EN_MPU	RESERVED

Bits	Field Name	Description	Type	Reset
31:5	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x0000000
4	EN_WKUP	WAKEUP domain dependency 0x0: CAM domain is independent of WKUP domain wake-up event. 0x1: CAM domain is woken-up upon WKUP domain wake-up event.	RW	0x1
3	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x0
2	EN_IVA2	IVA2 domain dependency 0x0: CAM domain is independent of IVA2 domain wake-up event. 0x1: CAM domain is woken-up upon IVA2 domain wake-up event.	RW	0x1

Bits	Field Name	Description	Type	Reset
1	EN_MPU	MPU domain dependency 0x0: CAM domain is independent of MPU domain wake-up. 0x1: CAM domain is woken-up upon MPU domain wake-up.	RW	0x1
0	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x0

Table 4-398. Register Call Summary for Register PM_WKDEP_CAM

PRCM Idle and Wake-Up Management • Device Wake-Up Events: [0] [1] [2]
PRCM Register Manual • CAM_PRM Registers: [3]

Table 4-399. PM_PWSTCTRL_CAM

Address Offset	0x0000 00E0		
Physical Address	0x4830 6FE0	**Instance**	CAM_PRM
Description	This register controls the CORE domain power state transition.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED														MEMONSTATE		RESERVED							MEMRETSTATE	RESERVED					LOGICRETSTATE	POWERSTATE	

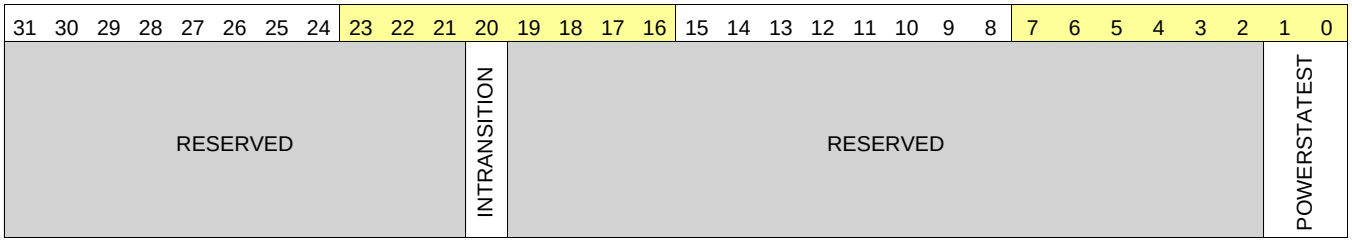
Bits	Field Name	Description	Type	Reset
31:18	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x0000
17:16	MEMONSTATE	Memory state when ON 0x3: Memory is always ON when domain is ON.	R	0x3
15:9	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x00
8	MEMRETSTATE	Memory state when RETENTION 0x1: Memory is always retained when domain is in RETENTION state.	R	0x1
7:3	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x00
2	LOGICRETSTATE	Logic state when RETENTION 0x1: Logic is always retained when domain is in RETENTION state.	R	0x1
1:0	POWERSTATE	Power state control 0x0: OFF state 0x1: RETENTION state 0x2: Reserved 0x3: ON state	RW	0x3

Table 4-400. Register Call Summary for Register PM_PWSTCTRL_CAM

PRCM Basic Programming Model • PM_PWSTCTRL_ <domain_name> (Power State Control Register): [0]
PRCM Register Manual • CAM_PRM Registers: [1]

Table 4-401. PM_PWSTST_CAM

Address Offset	0x0000 00E4		
Physical Address	0x4830 6FE4	**Instance**	CAM_PRM
Description	This register provides a status on the power state transition of the CAMERA domain.		
Type	R		



Bits	Field Name	Description	Type	Reset
31:21	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x000
20	INTRANSITION	Domain transition status 0x0: No transition 0x1: CAMERA power domain transition is in progress.	R	0x0
19:2	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x00000
1:0	POWERSTATEST	Current power state status 0x0: Power domain is OFF 0x1: Power domain is in RETENTION 0x2: Power domain is INACTIVE 0x3: Power domain is ON	R	0x3

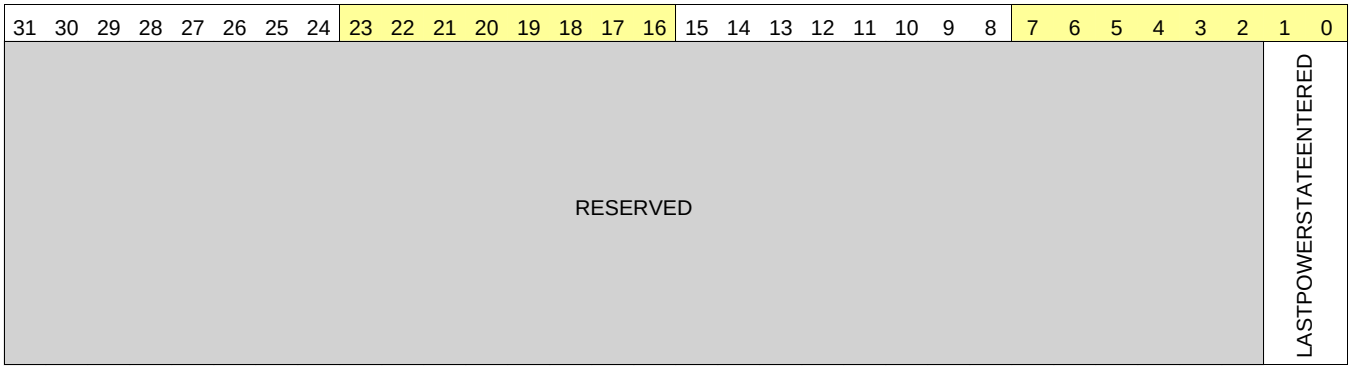
Table 4-402. Register Call Summary for Register PM_PWSTST_CAM

PRCM Basic Programming Model
- PM_PWSTST_ <domain_name> (Power State Status Register): [0]

PRCM Register Manual
- CAM_PRM Registers: [1]

Table 4-403. PM_PREPWSTST_CAM

Address Offset	0x0000 00E8		
Physical Address	0x4830 6FE8	**Instance**	CAM_PRM
Description	This register provides a status on the CAM domain previous power state. It indicates the state entered during the last sleep transition.		
Type	RW		



Bits	Field Name	Description	Type	Reset
31:2	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x00000000
1:0	LASTPOWERSTATEENTERED	Last power state entered 0x0: CAM domain was previously OFF 0x1: CAM domain was previously in RETENTION 0x2: CAM domain was previously INACTIVE 0x3: CAM domain was previously ON	RW	0x0

Table 4-404. Register Call Summary for Register PM_PREPWSTST_CAM

PRCM Register Manual
- CAM_PRM Registers: [0]

4.14.2.11 PER_PRM Registers

Table 4-405. PER_PRM Register Summary

Register Name	Type	Register Width (Bits)	Address Offset	Physical Address	Reset Type
RM_RSTST_PER	RW	32	0x0000 0058	0x4830 7058	C
PM_WKEN_PER	RW	32	0x0000 00A0	0x4830 70A0	W
PM_MPUGRPSEL_PER	RW	32	0x0000 00A4	0x4830 70A4	W
PM_IVA2GRPSEL_PER	RW	32	0x0000 00A8	0x4830 70A8	W
PM_WKST_PER	RW	32	0x0000 00B0	0x4830 70B0	C
PM_WKDEP_PER	RW	32	0x0000 00C8	0x4830 70C8	W
PM_PWSTCTRL_PER	RW	32	0x0000 00E0	0x4830 70E0	W
PM_PWSTST_PER	R	32	0x0000 00E4	0x4830 70E4	C
PM_PREPWSTST_PER	RW	32	0x0000 00E8	0x4830 70E8	C

Table 4-406. RM_RSTST_PER

Address Offset	0x0000 0058		
Physical Address	0x4830 7058	**Instance**	PER_PRM
Description	This register logs the different reset sources of the PERIPHERAL domain. Each bit is set upon release of the domain reset signal. Must be cleared by software.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																												COREDOMAINWKUP_RST	DOMAINWKUP_RST	GLOBALWARM_RST	GLOBALCOLD_RST

Bits	Field Name	Description	Type	Reset
31:4	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x0000000
3	COREDOMAINWKUP_RST	CORE domain wake-up reset Read 0x0: No power domain wake-up reset. Write 0x0: Status bit unchanged Read 0x1: PER domain has been reset following a CORE power domain wake-up from OFF to ON. Write 0x1: Status bit is cleared to 0.	RW	0x0
2	DOMAINWKUP_RST	Power domain wake-up reset Read 0x0: No power domain wake-up reset. Write 0x0: Status bit unchanged Read 0x1: PER domain has been reset following a PERIPHERAL power domain wake-up. Write 0x1: Status bit is cleared to 0.	RW	0x0
1	GLOBALWARM_RST	Global warm reset Read 0x0: No global warm reset. Write 0x0: Status bit unchanged Read 0x1: PER domain has been reset upon a global warm reset Write 0x1: Status bit is cleared to 0.	RW	0x0
0	GLOBALCOLD_RST	Global cold reset Read 0x0: No global cold reset. Write 0x0: Status bit unchanged Read 0x1: PER domain has been reset upon a global cold reset Write 0x1: Status bit is cleared to 0.	RW	0x1

Table 4-407. Register Call Summary for Register RM_RSTST_PER

PRCM Basic Programming Model • Reset Control: [0]
PRCM Register Manual • PER_PRM Registers: [1]

Table 4-408. PM_WKEN_PER

Address Offset	0x0000 00A0		
Physical Address	0x4830 70A0	**Instance**	PER_PRM
Description	This register allows enabling/disabling modules wake-up events.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED														EN_GPIO6	EN_GPIO5	EN_GPIO4	EN_GPIO3	EN_GPIO2	RESERVED	EN_UART3	EN_GPT9	EN_GPT8	EN_GPT7	EN_GPT6	EN_GPT5	EN_GPT4	EN_GPT3	EN_GPT2	EN_MCBSP4	EN_MCBSP3	EN_MCBSP2

Bits	Field Name	Description	Type	Reset
31:18	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x0000
17	EN_GPIO6	GPIO 6 wake-up control 0x0: GPIO 6 wake-up is disabled 0x1: GPIO 6 wake-up event is enabled	RW	0x1

Bits	Field Name	Description	Type	Reset
16	EN_GPIO5	GPIO 5 wake-up control 0x0: GPIO 5 wake-up is disabled 0x1: GPIO 5 wake-up event is enabled	RW	0x1
15	EN_GPIO4	GPIO 4 wake-up control 0x0: GPIO 4 wake-up is disabled 0x1: GPIO 4 wake-up event is enabled	RW	0x1
14	EN_GPIO3	GPIO 3 wake-up control 0x0: GPIO 3 wake-up is disabled 0x1: GPIO 3 wake-up event is enabled	RW	0x1
13	EN_GPIO2	GPIO 2 wake-up control 0x0: GPIO 2 wake-up is disabled 0x1: GPIO 2 wake-up event is enabled	RW	0x1
12	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x0
11	EN_UART3	UART 3 wake-up control 0x0: UART 3 wake-up is disabled 0x1: UART 3 wake-up event is enabled	RW	0x1
10	EN_GPT9	GPTIMER 9 wake-up control 0x0: GPTIMER 9 wake-up is disabled 0x1: GPTIMER 9 wake-up event is enabled	RW	0x1
9	EN_GPT8	GPTIMER 8 wake-up control 0x0: GPTIMER 8 wake-up is disabled 0x1: GPTIMER 8 wake-up event is enabled	RW	0x1
8	EN_GPT7	GPTIMER 7 wake-up control 0x0: GPTIMER 7 wake-up is disabled 0x1: GPTIMER 7 wake-up event is enabled	RW	0x1
7	EN_GPT6	GPTIMER 6 wake-up control 0x0: GPTIMER 6 wake-up is disabled 0x1: GPTIMER 6 wake-up event is enabled	RW	0x1
6	EN_GPT5	GPTIMER 5 wake-up control 0x0: GPTIMER 5 wake-up is disabled 0x1: GPTIMER 5 wake-up event is enabled	RW	0x1
5	EN_GPT4	GPTIMER 4 wake-up control 0x0: GPTIMER 4 wake-up is disabled 0x1: GPTIMER 4 wake-up event is enabled	RW	0x1
4	EN_GPT3	GPTIMER 3 wake-up control 0x0: GPTIMER 3 wake-up is disabled 0x1: GPTIMER 3 wake-up event is enabled	RW	0x1
3	EN_GPT2	GPTIMER 2 wake-up control 0x0: GPTIMER 2 wake-up is disabled 0x1: GPTIMER 2 wake-up event is enabled	RW	0x1
2	EN_MCBSP4	McBSP 4 wake-up control 0x0: McBSP 4 wake-up is disabled 0x1: McBSP 4 wake-up event is enabled	RW	0x1
1	EN_MCBSP3	McBSP3 wake-up control 0x0: McBSP 3 wake-up is disabled 0x1: McBSP 3 wake-up event is enabled	RW	0x1
0	EN_MCBSP2	McBSP 2 wake-up control 0x0: McBSP 2 wake-up is disabled 0x1: McBSP 2 wake-up event is enabled	RW	0x1

Table 4-409. Register Call Summary for Register PM_WKEN_PER

PRCM Idle and Wake-Up Management • Device Wake-Up Events: [0]
PRCM Basic Programming Model • Domain Wake-Up Control Registers: [1]
PRCM Register Manual • PER_PRM Registers: [2]

Table 4-410. PM_MPUGRPSEL_PER

Address Offset	0x0000 00A4		
Physical Address	0x4830 70A4	**Instance**	PER_PRM
Description	This register allows selecting the group of modules that wake-up the MPU.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED														GRPSEL_GPIO6	GRPSEL_GPIO5	GRPSEL_GPIO4	GRPSEL_GPIO3	GRPSEL_GPIO2	RESERVED	GRPSEL_UART3	GRPSEL_GPT9	GRPSEL_GPT8	GRPSEL_GPT7	GRPSEL_GPT6	GRPSEL_GPT5	GRPSEL_GPT4	GRPSEL_GPT3	GRPSEL_GPT2	GRPSEL_MCBSP4	GRPSEL_MCBSP3	GRPSEL_MCBSP2

Bits	Field Name	Description	Type	Reset
31:18	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x0000
17	GRPSEL_GPIO6	Select the GPIO 6 in the MPU wake-up events group 0x0: GPIO 6 is not attached to the MPU wake-up events group. 0x1: GPIO 6 is attached to the MPU wake-up events group.	RW	0x1
16	GRPSEL_GPIO5	Select the GPIO 5 in the MPU wake-up events group 0x0: GPIO 5 is not attached to the MPU wake-up events group. 0x1: GPIO 5 is attached to the MPU wake-up events group.	RW	0x1
15	GRPSEL_GPIO4	Select the GPIO 4 in the MPU wake-up events group 0x0: GPIO 4 is not attached to the MPU wake-up events group. 0x1: GPIO 4 is attached to the MPU wake-up events group.	RW	0x1
14	GRPSEL_GPIO3	Select the GPIO 3 in the MPU wake-up events group 0x0: GPIO 3 is not attached to the MPU wake-up events group. 0x1: GPIO 3 is attached to the MPU wake-up events group.	RW	0x1
13	GRPSEL_GPIO2	Select the GPIO 2 in the MPU wake-up events group 0x0: GPIO 2 is not attached to the MPU wake-up events group. 0x1: GPIO 2 is attached to the MPU wake-up events group.	RW	0x1
12	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x0

Bits	Field Name	Description	Type	Reset
11	GRPSEL_UART3	Select the UART 3 in the MPU wake-up events group 0x0: UART 3 is not attached to the MPU wake-up events group. 0x1: UART 3 is attached to the MPU wake-up events group.	RW	0x1
10	GRPSEL_GPT9	Select the GPTIMER 9 in the MPU wake-up events group 0x0: GPTIMER 9 is not attached to the MPU wake-up events group. 0x1: GPTIMER 9 is attached to the MPU wake-up events group.	RW	0x1
9	GRPSEL_GPT8	Select the GPTIMER 8 in the MPU wake-up events group 0x0: GPTIMER 8 is not attached to the MPU wake-up events group. 0x1: GPTIMER 8 is attached to the MPU wake-up events group.	RW	0x1
8	GRPSEL_GPT7	Select the GPTIMER 7 in the MPU wake-up events group 0x0: GPTIMER 7 is not attached to the MPU wake-up events group. 0x1: GPTIMER 7 is attached to the MPU wake-up events group.	RW	0x1
7	GRPSEL_GPT6	Select the GPTIMER 6 in the MPU wake-up events group 0x0: GPTIMER 6 is not attached to the MPU wake-up events group. 0x1: GPTIMER 6 is attached to the MPU wake-up events group.	RW	0x1
6	GRPSEL_GPT5	Select the GPTIMER 5 in the MPU wake-up events group 0x0: GPTIMER 5 is not attached to the MPU wake-up events group. 0x1: GPTIMER 5 is attached to the MPU wake-up events group.	RW	0x1
5	GRPSEL_GPT4	Select the GPTIMER 4 in the MPU wake-up events group 0x0: GPTIMER 4 is not attached to the MPU wake-up events group. 0x1: GPTIMER 4 is attached to the MPU wake-up events group.	RW	0x1
4	GRPSEL_GPT3	Select the GPTIMER 3 in the MPU wake-up events group 0x0: GPTIMER 3 is not attached to the MPU wake-up events group. 0x1: GPTIMER 3 is attached to the MPU wake-up events group.	RW	0x1
3	GRPSEL_GPT2	Select the GPTIMER 2 in the MPU wake-up events group 0x0: GPTIMER 2 is not attached to the MPU wake-up events group. 0x1: GPTIMER 2 is attached to the MPU wake-up events group.	RW	0x1
2	GRPSEL_MCBSP4	Select the McBSP 4 in the MPU wake-up events group 0x0: McBSP 4 is not attached to the MPU wake-up events group. 0x1: McBSP 4 is attached to the MPU wake-up events group.	RW	0x1
1	GRPSEL_MCBSP3	Select the McBSP 3 in the MPU wake-up events group 0x0: McBSP 3 is not attached to the MPU wake-up events group. 0x1: McBSP 3 is attached to the MPU wake-up events group.	RW	0x1

Bits	Field Name	Description	Type	Reset
0	GRPSEL_MCBSP2	Select the McBSP 2 in the MPU wake-up events group 0x0: McBSP 2 is not attached to the MPU wake-up events group. 0x1: McBSP 2 is attached to the MPU wake-up events group.	RW	0x1

Table 4-411. Register Call Summary for Register PM_MPUGRPSEL_PER

PRCM Idle and Wake-Up Management
- Device Wake-Up Events: [0]
- Wake-Up Dependencies: [1]

PRCM Basic Programming Model
- Domain Wake-Up Control Registers: [2]

PRCM Register Manual
- PER_PRM Registers: [3]

Table 4-412. PM_IVA2GRPSEL_PER

Address Offset	0x0000 00A8		
Physical Address	0x4830 70A8	**Instance**	PER_PRM
Description	This register allows selecting the group of modules that wake-up the IVA2.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED														GRPSEL_GPIO6	GRPSEL_GPIO5	GRPSEL_GPIO4	GRPSEL_GPIO3	GRPSEL_GPIO2	RESERVED	GRPSEL_UART3	GRPSEL_GPT9	GRPSEL_GPT8	GRPSEL_GPT7	GRPSEL_GPT6	GRPSEL_GPT5	GRPSEL_GPT4	GRPSEL_GPT3	GRPSEL_GPT2	GRPSEL_MCBSP4	GRPSEL_MCBSP3	GRPSEL_MCBSP2

Bits	Field Name	Description	Type	Reset
31:18	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x0000
17	GRPSEL_GPIO6	Select the GPIO 6 in the IVA2 wake-up events group 0x0: GPIO 6 is not attached to the IVA2 wake-up events group. 0x1: GPIO 6 is attached to the IVA2 wake-up events group.	RW	0x1
16	GRPSEL_GPIO5	Select the GPIO 5 in the IVA2 wake-up events group 0x0: GPIO 5 is not attached to the IVA2 wake-up events group. 0x1: GPIO 5 is attached to the IVA2 wake-up events group.	RW	0x1
15	GRPSEL_GPIO4	Select the GPIO 4 in the IVA2 wake-up events group 0x0: GPIO 4 is not attached to the IVA2 wake-up events group. 0x1: GPIO 4 is attached to the IVA2 wake-up events group.	RW	0x1
14	GRPSEL_GPIO3	Select the GPIO 3 in the IVA2 wake-up events group 0x0: GPIO 3 is not attached to the IVA2 wake-up events group. 0x1: GPIO 3 is attached to the IVA2 wake-up events group.	RW	0x1

Bits	Field Name	Description	Type	Reset
13	GRPSEL_GPIO2	Select the GPIO 2 in the IVA2 wake-up events group 0x0: GPIO 2 is not attached to the IVA2 wake-up events group. 0x1: GPIO 2 is attached to the IVA2 wake-up events group.	RW	0x1
12	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x0
11	GRPSEL_UART3	Select the UART 3 in the IVA2 wake-up events group 0x0: UART 3 is not attached to the IVA2 wake-up events group. 0x1: UART 3 is attached to the IVA2 wake-up events group.	RW	0x1
10	GRPSEL_GPT9	Select the GPTIMER 9 in the IVA2 wake-up events group 0x0: GPTIMER 9 is not attached to the IVA2 wake-up events group. 0x1: GPTIMER 9 is attached to the IVA2 wake-up events group.	RW	0x1
9	GRPSEL_GPT8	Select the GPTIMER 8 in the IVA2 wake-up events group 0x0: GPTIMER 8 is not attached to the IVA2 wake-up events group. 0x1: GPTIMER 8 is attached to the IVA2 wake-up events group.	RW	0x1
8	GRPSEL_GPT7	Select the GPTIMER 7 in the IVA2 wake-up events group 0x0: GPTIMER 7 is not attached to the IVA2 wake-up events group. 0x1: GPTIMER 7 is attached to the IVA2 wake-up events group.	RW	0x1
7	GRPSEL_GPT6	Select the GPTIMER 6 in the IVA2 wake-up events group 0x0: GPTIMER 6 is not attached to the IVA2 wake-up events group. 0x1: GPTIMER 6 is attached to the IVA2 wake-up events group.	RW	0x1
6	GRPSEL_GPT5	Select the GPTIMER 5 in the IVA2 wake-up events group 0x0: GPTIMER 5 is not attached to the IVA2 wake-up events group. 0x1: GPTIMER 5 is attached to the IVA2 wake-up events group.	RW	0x1
5	GRPSEL_GPT4	Select the GPTIMER 4 in the IVA2 wake-up events group 0x0: GPTIMER 4 is not attached to the IVA2 wake-up events group. 0x1: GPTIMER 4 is attached to the IVA2 wake-up events group.	RW	0x1
4	GRPSEL_GPT3	Select the GPTIMER 3 in the IVA2 wake-up events group 0x0: GPTIMER 3 is not attached to the IVA2 wake-up events group. 0x1: GPTIMER 3 is attached to the IVA2 wake-up events group.	RW	0x1
3	GRPSEL_GPT2	Select the GPTIMER 2 in the IVA2 wake-up events group 0x0: GPTIMER 2 is not attached to the IVA2 wake-up events group. 0x1: GPTIMER 2 is attached to the IVA2 wake-up events group.	RW	0x1
2	GRPSEL_MCBSP4	Select the McBSP 4 in the IVA2 wake-up events group 0x0: McBSP 4 is not attached to the IVA2 wake-up events group. 0x1: McBSP 4 is attached to the IVA2 wake-up events group.	RW	0x1

Bits	Field Name	Description	Type	Reset
1	GRPSEL_MCBSP3	Select the McBSP 3 in the IVA2 wake-up events group 0x0: McBSP 3 is not attached to the IVA2 wake-up events group. 0x1: McBSP 3 is attached to the IVA2 wake-up events group.	RW	0x1
0	GRPSEL_MCBSP2	Select the McBSP 2 in the IVA2 wake-up events group 0x0: McBSP 2 is not attached to the IVA2 wake-up events group. 0x1: McBSP 2 is attached to the IVA2 wake-up events group.	RW	0x1

Table 4-413. Register Call Summary for Register PM_IVA2GRPSEL_PER

PRCM Idle and Wake-Up Management • Device Wake-Up Events: [0]
PRCM Basic Programming Model • Domain Wake-Up Control Registers: [1]
PRCM Register Manual • PER_PRM Registers: [2]

Table 4-414. PM_WKST_PER

Address Offset	0x0000 00B0		
Physical Address	0x4830 70B0	**Instance**	PER_PRM
Description	This register logs module wake-up events. Must be cleared by software. If it is not cleared, it prevents further domain transition.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED														ST_GPIO6	ST_GPIO5	ST_GPIO4	ST_GPIO3	ST_GPIO2	RESERVED	ST_UART3	ST_GPT9	ST_GPT8	ST_GPT7	ST_GPT6	ST_GPT5	ST_GPT4	ST_GPT3	ST_GPT2	EN_MCBSP4	EN_MCBSP3	EN_MCBSP2

Bits	Field Name	Description	Type	Reset
31:18	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x0000
17	ST_GPIO6	GPIO 6 Wake-up status Read 0x0: GPIO 6 wake-up did not occur or was masked. Write 0x0: Status bit unchanged Read 0x1: GPIO 6 wake-up occurred. Write 0x1: Status bit is cleared to 0.	RW	0x0
16	ST_GPIO5	GPIO 5 Wake-up status Read 0x0: GPIO 5 wake-up did not occur or was masked. Write 0x0: Status bit unchanged Read 0x1: GPIO 5 wake-up occurred. Write 0x1: Status bit is cleared to 0.	RW	0x0

Bits	Field Name	Description	Type	Reset
15	ST_GPIO4	GPIO 4 Wake-up status Read 0x0: GPIO 4 wake-up did not occur or was masked. Write 0x0: Status bit unchanged Read 0x1: GPIO 4 wake-up occurred. Write 0x1: Status bit is cleared to 0.	RW	0x0
14	ST_GPIO3	GPIO 3 Wake-up status Read 0x0: GPIO 3 wake-up did not occur or was masked. Write 0x0: Status bit unchanged Read 0x1: GPIO 3 wake-up occurred. Write 0x1: Status bit is cleared to 0.	RW	0x0
13	ST_GPIO2	GPIO 2 Wake-up status Read 0x0: GPIO 2 wake-up did not occur or was masked. Write 0x0: Status bit unchanged Read 0x1: GPIO 2 wake-up occurred. Write 0x1: Status bit is cleared to 0.	RW	0x0
12	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x0
11	ST_UART3	UART 3 Wake-up status Read 0x0: UART 3 wake-up did not occur or was masked. Write 0x0: Status bit unchanged Read 0x1: UART 3 wake-up occurred. Write 0x1: Status bit is cleared to 0.	RW	0x0
10	ST_GPT9	GPTIMER 9 Wake-up status Read 0x0: GPTIMER 9 wake-up did not occur or was masked. Write 0x0: Status bit unchanged Read 0x1: GPTIMER 9 wake-up occurred. Write 0x1: Status bit is cleared to 0.	RW	0x0
9	ST_GPT8	GPTIMER 8 Wake-up status Read 0x0: GPTIMER 8 wake-up did not occur or was masked. Write 0x0: Status bit unchanged Read 0x1: GPTIMER 8 wake-up occurred. Write 0x1: Status bit is cleared to 0.	RW	0x0
8	ST_GPT7	GPTIMER 7 Wake-up status Read 0x0: GPTIMER 7 wake-up did not occur or was masked. Write 0x0: Status bit unchanged Read 0x1: GPTIMER 7 wake-up occurred. Write 0x1: Status bit is cleared to 0.	RW	0x0
7	ST_GPT6	GPTIMER 6 Wake-up status Read 0x0: GPTIMER 6 wake-up did not occur or was masked. Write 0x0: Status bit unchanged Read 0x1: GPTIMER 6 wake-up occurred. Write 0x1: Status bit is cleared to 0.	RW	0x0

Bits	Field Name	Description	Type	Reset
6	ST_GPT5	GPTIMER 5 Wake-up status Read 0x0: GPTIMER 5 wake-up did not occur or was masked. Write 0x0: Status bit unchanged Read 0x1: GPTIMER 5 wake-up occurred. Write 0x1: Status bit is cleared to 0.	RW	0x0
5	ST_GPT4	GPTIMER 4 Wake-up status Read 0x0: GPTIMER 4 wake-up did not occur or was masked. Write 0x0: Status bit unchanged Read 0x1: GPTIMER 4 wake-up occurred. Write 0x1: Status bit is cleared to 0.	RW	0x0
4	ST_GPT3	GPTIMER 3 Wake-up status Read 0x0: GPTIMER 3 wake-up did not occur or was masked. Write 0x0: Status bit unchanged Read 0x1: GPTIMER 3 wake-up occurred. Write 0x1: Status bit is cleared to 0.	RW	0x0
3	ST_GPT2	GPTIMER 2 Wake-up status Read 0x0: GPTIMER 2 wake-up did not occur or was masked. Write 0x0: Status bit unchanged Read 0x1: GPTIMER 2 wake-up occurred. Write 0x1: Status bit is cleared to 0.	RW	0x0
2	EN_MCBSP4	McBSP 4 Wake-up status Read 0x0: McBSP 4 wake-up did not occur or was masked. Write 0x0: Status bit unchanged Read 0x1: McBSP 4 wake-up occurred. Write 0x1: Status bit is cleared to 0.	RW	0x0
1	EN_MCBSP3	McBSP3 Wake-up status Read 0x0: McBSP 3 wake-up did not occur or was masked. Write 0x0: Status bit unchanged Read 0x1: McBSP 3 wake-up occurred. Write 0x1: Status bit is cleared to 0.	RW	0x0
0	EN_MCBSP2	McBSP 2 Wake-up status Read 0x0: McBSP 2 wake-up did not occur or was masked. Write 0x0: Status bit unchanged Read 0x1: McBSP 2 wake-up occurred. Write 0x1: Status bit is cleared to 0.	RW	0x0

Table 4-415. Register Call Summary for Register PM_WKST_PER

PRCM Basic Programming Model • Domain Wake-Up Control Registers: [0]
PRCM Register Manual • PER_PRM Registers: [1]

Table 4-416. PM_WKDEP_PER

Address Offset	0x0000 00C8		
Physical Address	0x4830 70C8	**Instance**	PER_PRM
Description	This register allows enabling or disabling the wake-up of the PER domain upon another domain wakeup events.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																											EN_WKUP	RESERVED	EN_IVA2	EN_MPU	EN_CORE

Bits	Field Name	Description	Type	Reset
31:5	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x0000000
4	EN_WKUP	WAKEUP domain dependency 0x0: PER domain is independent of WKUP domain wake-up event. 0x1: PER domain is woken-up upon WKUP domain wake-up event.	RW	0x1
3	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x0
2	EN_IVA2	IVA2 domain dependency 0x0: PER domain is independent of IVA2 domain wake-up event. 0x1: PER domain is woken-up upon IVA2 domain wake-up event.	RW	0x1
1	EN_MPU	MPU domain dependency 0x0: PER domain is independent of MPU domain wake-up event. 0x1: PER domain is is woken-up upon MPU domain wake-up event.	RW	0x1
0	EN_CORE	CORE domain dependency (CORE-L3 clock domain only, not CORE-L4) 0x0: PER domain is independent of CORE domain wake-up event (CORE-L3 clock domain only, not CORE-L4). 0x1: PER domain is not woken-up upon CORE domain wake-up event.	RW	0x1

Table 4-417. Register Call Summary for Register PM_WKDEP_PER

PRCM Idle and Wake-Up Management
- Device Wake-Up Events: [0] [1] [2] [3]

PRCM Register Manual
- PER_PRM Registers: [4]

Table 4-418. PM_PWSTCTRL_PER

Address Offset	0x0000 00E0		
Physical Address	0x4830 70E0	**Instance**	PER_PRM
Description	This register controls the PER domain power state transition.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED														MEMONSTATE		RESERVED							MEMRETSTATE	RESERVED					LOGICRETSTATE	POWERSTATE	

Bits	Field Name	Description	Type	Reset
31:18	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x0000
17:16	MEMONSTATE	Memory state when ON	R	0x3
		0x3: Memory is always ON when domain is ON.		
15:9	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x00
8	MEMRETSTATE	Memory state when RETENTION	R	0x1
		0x1: Memory is always retained when domain is in RETENTION state.		
7:3	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x00
2	LOGICRETSTATE	Logic state when domain is RETENTION	RW	0x1
		0x0: Logic build with retention flip-flop (GPIO) is retained and remaining logic is OFF when domain is in RETENTION state.		
		0x1: Logic is retained when domain is in RETENTION state.		
1:0	POWERSTATE	Power state control	RW	0x3
		0x0: OFF state		
		0x1: RETENTION state		
		0x2: Reserved		
		0x3: ON state		

Table 4-419. Register Call Summary for Register PM_PWSTCTRL_PER

PRCM Basic Programming Model

- PM_PWSTCTRL_ <domain_name> (Power State Control Register): [0]

PRCM Register Manual

- PER_PRM Registers: [1]

Table 4-420. PM_PWSTST_PER

Address Offset	0x0000 00E4		
Physical Address	0x4830 70E4	**Instance**	PER_PRM
Description	This register provides a status on the power state transition of the PERIPHERAL domain.		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED											INTRANSITION	RESERVED																		LOGICSTATEST	POWERSTATEST	

Bits	Field Name	Description	Type	Reset
31:21	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x000
20	INTRANSITION	Domain transition status 0x0: No transition 0x1: PERIPHERAL power domain transition is in progress.	R	0x0
19:3	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x00000
2	LOGICSTATEST	Logic state status 0x0: PER domain logic is OFF 0x1: PER domain logic is ON	R	0x1
1:0	POWERSTATEST	Current power state status 0x0: Power domain is OFF 0x1: Power domain is in RETENTION 0x2: Power domain is INACTIVE 0x3: Power domain is ON	R	0x3

Table 4-421. Register Call Summary for Register PM_PWSTST_PER

PRCM Basic Programming Model • PM_PWSTST_ <domain_name> (Power State Status Register): [0]
PRCM Register Manual • PER_PRM Registers: [1]

Table 4-422. PM_PREPWSTST_PER

Address Offset	0x0000 00E8		
Physical Address	0x4830 70E8	**Instance**	PER_PRM
Description	This register provides a status on the PER domain previous power state. It indicates the state entered during the last sleep transition.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																													LASTLOGICSTATEENTERED	LASTPOWERSTATEENTERED	

Bits	Field Name	Description	Type	Reset
31:3	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x00000000
2	LASTLOGICSTATEENTERED	Last logic state entered 0x0: PER domain logic was previously OFF 0x1: PER domain logic was previously ON	RW	0x0
1:0	LASTPOWERSTATEENTERED	Last power state entered 0x0: PER domain was previously OFF 0x1: PER domain was previously in RETENTION 0x2: PER domain was previously INACTIVE 0x3: PER domain was previously ON	RW	0x0

Table 4-423. Register Call Summary for Register PM_PREPWSTST_PER

PRCM Register Manual
- PER_PRM Registers: [0]

4.14.2.12 EMU_PRM Registers

Table 4-424. EMU_PRM Register Summary

Register Name	Type	Register Width (Bits)	Address Offset	Physical Address	Reset Type
RM_RSTST_EMU	RW	32	0x0000 0058	0x4830 7158	C
PM_PWSTST_EMU	RW	32	0x0000 00E4	0x4830 71E4	C

Table 4-425. RM_RSTST_EMU

Address Offset	0x0000 0058		
Physical Address	0x4830 7158	**Instance**	EMU_PRM
Description	This register logs the different reset sources of the EMU domain. Each bit is set upon release of the domain reset signal. Must be cleared by software.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																													DOMAINWKUP_RST	GLOBALWARM_RST	GLOBALCOLD_RST

Bits	Field Name	Description	Type	Reset
31:3	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x00000000
2	DOMAINWKUP_RST	Power domain wake-up reset Read 0x0: No power domain wake-up reset. Write 0x0: Status bit unchanged Read 0x1: EMULATION domain has been reset following an EMULATION power domain wake-up. Write 0x1: Status bit is cleared to 0.	RW	0x0

Bits	Field Name	Description	Type	Reset
1	GLOBALWARM_RST	Global warm reset Read 0x0: No global warm reset. Write 0x0: Status bit unchanged Read 0x1: MPU domain has been reset upon a global warm reset Write 0x1: Status bit is cleared to 0.	RW	0x0
0	GLOBALCOLD_RST	Global cold reset Read 0x0: No global cold reset. Write 0x0: Status bit unchanged Read 0x1: MPU domain has been reset upon a global cold reset Write 0x1: Status bit is cleared to 0.	RW	0x1

Table 4-426. Register Call Summary for Register RM_RSTST_EMU

PRCM Basic Programming Model • Reset Control: [0]
PRCM Register Manual • EMU_PRM Registers: [1]

Table 4-427. PM_PWSTST_EMU

Address Offset	0x0000 00E4		
Physical Address	0x4830 71E4	**Instance**	EMU_PRM
Description	This register provides a status on the power state transition of the EMULATION domain.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED											INTRANSITION	RESERVED																		POWERSTATEST	

Bits	Field Name	Description	Type	Reset
31:21	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x000
20	INTRANSITION	Domain transition status 0x0: No transition 0x1: EMULATION power domain transition is in progress.	R	0x0
19:2	RESERVED	Write 0x0040 for future compatibility. Read returns 0x0040.	R	0x00040
1:0	POWERSTATEST	Current power state status 0x0: Power domain is OFF 0x1: Reserved 0x2: Reserved 0x3: Power domain is ON	R	0x3

Table 4-428. Register Call Summary for Register PM_PWSTST_EMU

PRCM Basic Programming Model • PM_PWSTST_ <domain_name> (Power State Status Register): [0]

Table 4-428. Register Call Summary for Register PM_PWSTST_EMU (continued)

PRCM Register Manual • EMU_PRM Registers: [1]

4.14.2.13 Global_Reg_PRM Registers

Table 4-429. Global_Reg_PRM Register Summary

Register Name	Type	Register Width (Bits)	Address Offset	Physical Address	Reset Type
PRM_VC_SMPS_SA	RW	32	0x0000 0020	0x4830 7220	W
PRM_VC_SMPS_VOL_RA	RW	32	0x0000 0024	0x4830 7224	W
PRM_VC_SMPS_CMD_RA	RW	32	0x0000 0028	0x4830 7228	W
PRM_VC_CMD_VAL_0	RW	32	0x0000 002C	0x4830 722C	W
PRM_VC_CMD_VAL_1	RW	32	0x0000 0030	0x4830 7230	W
PRM_VC_CH_CONF	RW	32	0x0000 0034	0x4830 7234	W
PRM_VC_I2C_CFG	RW	32	0x0000 0038	0x4830 7238	W
PRM_VC_BYPASS_VAL	RW	32	0x0000 003C	0x4830 723C	W
PRM_RSTCTRL	RW	32	0x0000 0050	0x4830 7250	C
PRM_RSTTIME	RW	32	0x0000 0054	0x4830 7254	C
PRM_RSTST	RW	32	0x0000 0058	0x4830 7258	C
PRM_VOLTCTRL	RW	32	0x0000 0060	0x4830 7260	W
PRM_SRAM_PCHARGE	RW	32	0x0000 0064	0x4830 7264	C
PRM_CLKSRC_CTRL	RW	32	0x0000 0070	0x4830 7270	C
PRM_OBS	R	32	0x0000 0080	0x4830 7280	C
PRM_VOLTSETUP1	RW	32	0x0000 0090	0x4830 7290	C
PRM_VOLTOFFSET	RW	32	0x0000 0094	0x4830 7294	C
PRM_CLKSETUP	RW	32	0x0000 0098	0x4830 7298	C
PRM_POLCTRL	RW	32	0x0000 009C	0x4830 729C	C
PRM_VOLTSETUP2	RW	32	0x0000 00A0	0x4830 72A0	C
This information is not available in public domain			0x0000 00B0 - 0x0000 00E4	0x4830 72B0 - 0x4830 72E4	W

Table 4-430. PRM_VC_SMPS_SA

Address Offset	0x0000 0020		
Physical Address	0x4830 7220	**Instance**	Global_Reg_PRM
Description	This register allows the setting of the I^2C slave address of the Power IC device.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								SA1								RESERVED								SA0							

Bits	Field Name	Description	Type	Reset
31:23	RESERVED	Write 0s for future compatibility. Read is undefined.	R	0x000
22:16	SA1	Set the I^2C slave address value for the second (if any) Power IC device.	RW	0x00
15:7	RESERVED	Write 0s for future compatibility. Read is undefined.	R	0x000
6:0	SA0	Set the I^2C slave address value for the first Power IC device.	RW	0x00

Table 4-431. Register Call Summary for Register PRM_VC_SMPS_SA

PRCM Basic Programming Model
- Voltage Controller Registers: [0] [1] [2] [3]
- Voltage Controller Initialization Basic Programming Model: [4] [5]

PRCM Use Cases and Tips
- DVFS Using SmartReflex :

PRCM Register Manual
- Global_Reg_PRM Registers: [7]

Table 4-432. PRM_VC_SMPS_VOL_RA

Address Offset	0x0000 0024		
Physical Address	0x4830 7224	**Instance**	Global_Reg_PRM
Description	This register allows the setting of the voltage configuration register address for the VDD channels.		
Type	RW		

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10 9 8	7 6 5 4 3 2 1 0
RESERVED	VOLRA1	RESERVED	VOLRA0

Bits	Field Name	Description	Type	Reset
31:24	RESERVED	Write 0s for future compatibility. Read is undefined.	R	0x00
23:16	VOLRA1	Set the voltage configuration register address value for the second VDD channel (VDD2).	RW	0x00
15:8	RESERVED	Write 0s for future compatibility. Read is undefined.	R	0x00
7:0	VOLRA0	Set the voltage configuration register address value for the first VDD channel (VDD1).	RW	0x00

Table 4-433. Register Call Summary for Register PRM_VC_SMPS_VOL_RA

PRCM Basic Programming Model
- Voltage Controller Registers: [0] [1]
- Voltage Controller Initialization Basic Programming Model: [2] [3]

PRCM Use Cases and Tips
- DVFS Using SmartReflex :

PRCM Register Manual
- Global_Reg_PRM Registers: [6]

Table 4-434. PRM_VC_SMPS_CMD_RA

Address Offset	0x0000 0028		
Physical Address	0x4830 7228	**Instance**	Global_Reg_PRM
Description	This register allows the setting of the ON/Retention/OFF command configuration register address for the VDD channels. It is used if the Power IC device has different register addresses for voltage value and ON/Retention/OFF command.		
Type	RW		

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10 9 8	7 6 5 4 3 2 1 0
RESERVED	CMDRA1	RESERVED	CMDRA0

Bits	Field Name	Description	Type	Reset
31:24	RESERVED	Write 0s for future compatibility. Read is undefined.	R	0x00
23:16	CMDRA1	Set the ON/ON-LP/Retention/OFF command configuration register address value for the second VDD channel (VDD2).	RW	0x00
15:8	RESERVED	Write 0s for future compatibility. Read is undefined.	R	0x00

Bits	Field Name	Description	Type	Reset
7:0	CMDRA0	Set the ON/ON-LP/Retention/OFF command configuration register address value for the first VDD channel (VDD1).	RW	0x00

Table 4-435. Register Call Summary for Register PRM_VC_SMPS_CMD_RA

PRCM Basic Programming Model
- Voltage Controller Registers: [0] [1]
- Voltage Controller Initialization Basic Programming Model: [2] [3]

PRCM Register Manual
- Global_Reg_PRM Registers: [4]

Table 4-436. PRM_VC_CMD_VAL_0

Address Offset	0x0000 002C		
Physical Address	0x4830 722C	**Instance**	Global_Reg_PRM
Description	This register allows the setting of the ON/Retention/OFF voltage level values for the first VDD channel.		
Type	RW		

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10 9 8	7 6 5 4 3 2 1 0
ON	ONLP	RET	OFF

Bits	Field Name	Description	Type	Reset
31:24	ON	Set the ON voltage level value for the first VDD channel (VDD1).	RW	0x00
23:16	ONLP	Set the ON-LP voltage level value for the first VDD channel (VDD1).	RW	0x00
15:8	RET	Set the RET voltage level value for the first VDD channel (VDD1).	RW	0x00
7:0	OFF	Set the OFF voltage level value for the first VDD channel (VDD1).	RW	0x00

Table 4-437. Register Call Summary for Register PRM_VC_CMD_VAL_0

PRCM Off-Mode Management
- Sleep Sequences: [0]
- Wake-Up Sequences: [1]

PRCM Basic Programming Model
- Voltage Controller Registers: [2]

PRCM Register Manual
- Global_Reg_PRM Registers: [3]

Table 4-438. PRM_VC_CMD_VAL_1

Address Offset	0x0000 0030		
Physical Address	0x4830 7230	**Instance**	Global_Reg_PRM
Description	This register allows the setting of the ON/Retention/OFF voltage level values for the second VDD channel. It is used if the second channel has different values than the first channel.		
Type	RW		

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10 9 8	7 6 5 4 3 2 1 0
ON	ONLP	RET	OFF

Bits	Field Name	Description	Type	Reset
31:24	ON	Set the ON voltage level value for the second VDD channel (VDD2).	RW	0x00
23:16	ONLP	Set the ON-LP voltage level value for the second VDD channel (VDD2).	RW	0x00
15:8	RET	Set the RET voltage level value for the second VDD channel (VDD2).	RW	0x00
7:0	OFF	Set the OFF voltage level value for the second VDD channel (VDD2).	RW	0x00

Table 4-439. Register Call Summary for Register PRM_VC_CMD_VAL_1

PRCM Off-Mode Management
- Sleep Sequences: [0]
- Wake-Up Sequences: [1]

PRCM Basic Programming Model
- Voltage Controller Registers: [2]

PRCM Register Manual
- Global_Reg_PRM Registers: [3]

Table 4-440. PRM_VC_CH_CONF

Address Offset	0x0000 0034		
Physical Address	0x4830 7234	**Instance**	Global_Reg_PRM
Description	This register allows the configuration pointers for both VDD channels.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED											CMD1	RACEN1	RAC1	RAV1	SA1	RESERVED											CMD0	RACEN0	RAC0	RAV0	SA0

Bits	Field Name	Description	Type	Reset
31:21	RESERVED	Write 0s for future compatibility. Read is undefined.	R	0x000
20	CMD1	Selects the ON/ON-LP/Retention/OFF voltage values for the second VDD channel (VDD2).	RW	0x0
19	RACEN1	Enable bit for usage of RAC1.	RW	0x0
18	RAC1	Set the ON/ON-LP/Retention/OFF command configuration register address pointer for the second VDD channel (VDD2).	RW	0x0
17	RAV1	Set the voltage configuration register address pointer for the second VDD channel (VDD2).	RW	0x0
16	SA1	Set the slave address pointer for the second VDD channel (VDD2).	RW	0x0
15:5	RESERVED	Write 0s for future compatibility. Read is undefined.	R	0x000
4	CMD0	Selects the ON/ON-LP/Retention/OFF voltage values for the first VDD channel (VDD1).	RW	0x0
3	RACEN0	Enable bit for usage of RAC0.	RW	0x0
2	RAC0	Set the ON/ON-LP/Retention/OFF command configuration register address pointer for the first VDD channel (VDD1).	RW	0x0
1	RAV0	Set the voltage configuration register address pointer for the first VDD channel (VDD1).	RW	0x0
0	SA0	Set the slave address pointer for the first VDD channel (VDD1).	RW	0x0

Table 4-441. Register Call Summary for Register PRM_VC_CH_CONF

PRCM Basic Programming Model
- Voltage Controller Registers: [0] [1] [2] [3]
- Voltage Controller Initialization Basic Programming Model: [4] [5] [6] [7] [8] [9] [10] [11] [12] [13] [14]

PRCM Register Manual
- Global_Reg_PRM Registers: [15]

Table 4-442. PRM_VC_I2C_CFG

Address Offset	0x0000 0038		
Physical Address	0x4830 7238	**Instance**	Global_Reg_PRM
Description	This register allows the configuration pointers for both VDD channels.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																										HSMASTER	SREN	HSEN	MCODE		

Bits	Field Name	Description	Type	Reset
31:6	RESERVED	Write 0s for future compatibility. Read is undefined.	R	0x0000000
5	HSMASTER	Put the I2C pads in a low power mode in case of light load. 0x0: Disables the I2C pads low power mode 0x1: Enables the I2C pads low power mode	RW	0x0
4	SREN	Enables the I2C repeated start operation mode. 0x0: Disables the repeated start operation mode 0x1: Enables the repeated start operation mode	RW	0x1
3	HSEN	Enables I2C bus High Speed mode. 0x0: Disables the I2C high speed mode 0x1: Enables the I2C high speed mode	RW	0x1
2:0	MCODE	Master code value for I2C High Speed preamble transmission.	RW	0x0

Table 4-443. Register Call Summary for Register PRM_VC_I2C_CFG

PRCM Basic Programming Model
- Voltage Controller Registers: [0]
- Voltage Controller Initialization Basic Programming Model: [1] [2]

PRCM Use Cases and Tips
- DVFS Using SmartReflex :

PRCM Register Manual
- Global_Reg_PRM Registers: [5]

Table 4-444. PRM_VC_BYPASS_VAL

Address Offset	0x0000 003C		
Physical Address	0x4830 723C	**Instance**	Global_Reg_PRM
Description	This register allows the programming of the Power IC device using the bypass interface.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								VALID	DATA							REGADDR								RESERVED	SLAVEADDR						

Bits	Field Name	Description	Type	Reset
31:25	RESERVED	Write 0s for future compatibility. Read is undefined.	RW	0x00
24	VALID	This bit validates the bypass command. It is automatically cleared by HW either after getting the acknowledge back from the SMPS or if an error occured. 0x0: Reserved 0x0: The last command send has been acknowledged 0x1: The Voltage Controller send the command to the I2C interface 0x1: Pending command is being process	RW	0x0
23:16	DATA	Data to send to the Power IC device.	RW	0x00
15:8	REGADDR	Set the address of Power IC device register to configure.	RW	0x00
7	RESERVED	Write 0s for future compatibility. Read is undefined.	RW	0x0
6:0	SLAVEADDR	Set the I2C slave address value.	RW	0x00

Table 4-445. Register Call Summary for Register PRM_VC_BYPASS_VAL

PRCM Basic Programming Model
- Voltage Controller Registers: [0]

PRCM Register Manual
- Global_Reg_PRM Registers: [1]

Table 4-446. PRM_RSTCTRL

Address Offset	0x0000 0050		
Physical Address	0x4830 7250	**Instance**	Global_Reg_PRM
Description	Global software and DPLL3 reset control. This register is auto-cleared. Only write 1 is possible. A read returns 0 only.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																													RST_DPLL3	RST_GS	RESERVED

Bits	Field Name	Description	Type	Reset
31:3	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x00000000
2	RST_DPLL3	DPLL3 software reset control. This bit is reset only upon a global cold source of reset. 0x0: DPLL3 software reset is cleared. 0x1: Asserts the DPLL3 software reset and induces a global cold reset on the whole chip. The software must ensure the SDRAM is properly put in sef-refresh mode before applying this reset.	RW	0x0
1	RST_GS	Global software reset control. This bit is reset upon any global source of reset (warm and cold). 0x0: Global software reset is cleared. 0x1: Asserts a global software reset.	RW	0x0
0	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x0

Table 4-447. Register Call Summary for Register PRM_RSTCTRL

PRCM Reset Manager Functional Description • Global Reset Sources: [0] [1]
PRCM Basic Programming Model • Reset Control: [2] [3]
PRCM Register Manual • Global_Reg_PRM Registers: [4]

Table 4-448. PRM_RSTTIME

Address Offset	0x0000 0054		
Physical Address	0x4830 7254	**Instance**	Global_Reg_PRM
Description	Reset duration control.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																			RSTTIME2					RSTTIME1							

Bits	Field Name	Description	Type	Reset
31:13	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x00000
12:8	RSTTIME2	(Power domain) reset duration 2 (number of RM_ICLK clock cycles)	RW	0x10
7:0	RSTTIME1	(Global) reset duration 1 (number of Func_32k.clk clock cycles)	RW	0x06

Table 4-449. Register Call Summary for Register PRM_RSTTIME

PRCM Reset Manager Functional Description • Occurrence: [0] • External Warm Reset Assertion: [1] • Power-Up Sequence: [2] • Global Warm Reset Sequence: [3]
PRCM Basic Programming Model • Reset Management: [4]
PRCM Register Manual • Global_Reg_PRM Registers: [5]

Table 4-450. PRM_RSTST

Address Offset	0x0000 0058		
Physical Address	0x4830 7258	**Instance**	Global_Reg_PRM
Description	This register logs the global reset sources. Each bit is set upon release of the domain reset signal. Must be cleared by software.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																					ICECRUSHER_RST	ICEPICK_RST	VDD2_VOLTAGE_MANAGER_RST	VDD1_VOLTAGE_MANAGER_RST	EXTERNAL_WARM_RST	RESERVED	MPU_WD_RST	RESERVED	RESERVED	GLOBAL_SW_RST	GLOBAL_COLD_RST

Bits	Field Name	Description	Type	Reset
31:11	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x000000
10	ICECRUSHER_RST	IceCrusher reset event. This is a source of warm reset initiated by the emulation. Read 0x0: No IceCrusher reset. Write 0x0: Status bit unchanged Read 0x1: IceCrusher reset occurred. Write 0x1: Status bit is cleared to 0.	RW	0x0
9	ICEPICK_RST	IcePick reset event. This is a source of warm reset initiated by the emulation. Read 0x0: No IcePick reset. Write 0x0: Status bit unchanged Read 0x1: IcePick reset occurred. Write 0x1: Status bit is cleared to 0.	RW	0x0
8	VDD2_VOLTAGE_MANAGER_RST	VDD2 voltage manager reset event Read 0x0: No VDD2 voltage manager reset. Write 0x0: Status bit unchanged Read 0x1: VDD2 voltage manager reset occurred. Write 0x1: Status bit is cleared to 0.	RW	0x0
7	VDD1_VOLTAGE_MANAGER_RST	VDD1 voltage manager reset event Read 0x0: No VDD1 voltage manager reset. Write 0x0: Status bit unchanged Read 0x1: VDD1 voltage manager reset occurred. Write 0x1: Status bit is cleared to 0.	RW	0x0
6	EXTERNAL_WARM_RST	External warm reset event Read 0x0: No global warm reset. Write 0x0: Status bit unchanged Read 0x1: Global external warm reset occurred. Write 0x1: Status bit is cleared to 0.	RW	0x0
5	RESERVED	Reserved for non-GP devices.	RW	0x0

Bits	Field Name	Description	Type	Reset
4	MPU_WD_RST	MPU watchdog reset event Read 0x0: No MPU watchdog reset. Write 0x0: Status bit unchanged Read 0x1: MPU wachtdog reset occurred. Write 0x1: Status bit is cleared to 0.	RW	0x0
3	RESERVED	Reserved for non-GP devices.	RW	0x0
2	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x0
1	GLOBAL_SW_RST	Global software reset event Read 0x0: No global software reset. Write 0x0: Status bit unchanged Read 0x1: Global software reset occurred. Write 0x1: Status bit is cleared to 0.	RW	0x0
0	GLOBAL_COLD_RST	Power-up (cold) reset event Read 0x0: No power-on reset. Write 0x0: Status bit unchanged Read 0x1: Power-on reset occurred. Write 0x1: Status bit is cleared to 0.	RW	0x1

Table 4-451. Register Call Summary for Register PRM_RSTST

PRCM Reset Manager Functional Description
- Reset Logging: [0]
- PRCM Reset Logging Mechanism: [1]

PRCM Basic Programming Model
- Reset Control: [2] [3] [4]
- Reset Management: [5]

PRCM Register Manual
- Global_Reg_PRM Registers: [6]

Table 4-452. PRM_VOLTCTRL

Address Offset	0x0000 0060		
Physical Address	0x4830 7260	**Instance**	Global_Reg_PRM
Description	This register allows a direct control on the external power IC.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																											SEL_VMODE	SEL_OFF	AUTO_OFF	AUTO_RET	AUTO_SLEEP

Bits	Field Name	Description	Type	Reset
31:5	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x0000000
4	SEL_VMODE	This bit allows to select the mode used to control the Power IC (I2C or VMODE). 0x0: The Power IC is controlled through the I2C interface. 0x1: The Power IC is controlled through the VMODE interface.	RW	0x0

Bits	Field Name	Description	Type	Reset
3	SEL_OFF	This bit allows to select the mode used to send the OFF command. 0x0: The OFF command is send through the voltage controller I2C interface. 0x1: The signal SYS.OFF_MODE is asserted.	RW	0x0
2	AUTO_OFF(1)	This bit allows to send an OFF command to the Power IC. 0x0: The OFF command is not sent. 0x1: The OFF command is automatically sent when the voltage domain is in the appropriate standby mode.	RW	0x0
1	AUTO_RET(1)	This bit allows to send a RETENTION command to the Power IC through the voltage controller I2C interface. 0x0: The RETENTION command is not sent. 0x1: The RETENTION command is automatically sent when the voltage domain is in the appropriate standby mode.	RW	0x0
0	AUTO_SLEEP(1)	This bit allows to send a SLEEP command to the Power IC through the voltage controller I2C interface. 0x0: The SLEEP command is not sent. 0x1: The SLEEP command is automatically sent when the voltage domain is in the appropriate standby mode.	RW	0x0

(1) It is mandatory to program the AUTO_OFF, AUTO_RET, and AUTO_SLEEP bit fields in an exclusive manner.

Table 4-453. Register Call Summary for Register PRM_VOLTCTRL

PRCM Voltage Management Functional Description • Voltage Domain Dependencies: [0] [1] [2] • Direct Control With VMODE Signals: [3]
PRCM Off-Mode Management • Sleep Sequences: [4] • Wake-Up Sequences: [5]
PRCM Basic Programming Model • External Voltage Control Register Descriptions: [6] [7]
PRCM Use Cases and Tips • Programming Sequence: [8]
PRCM Register Manual • Global_Reg_PRM Registers: [9]

Table 4-454. PRM_SRAM_PCHARGE

Address Offset	0x0000 0064		
Physical Address	0x4830 7264	**Instance**	Global_Reg_PRM
Description	This register allows setting the pre-charge time of the SRAM.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																								PCHARGE_TIME							

Bits	Field Name	Description	Type	Reset
31:8	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x000000
7:0	PCHARGE_TIME	Number of system clock cycles for the SRAM pre-charge duration.	RW	0x50

Table 4-455. Register Call Summary for Register PRM_SRAM_PCHARGE

PRCM Register Manual

- Global_Reg_PRM Registers: [0]

Table 4-456. PRM_CLKSRC_CTRL

Address Offset	0x0000 0070		
Physical Address	0x4830 7270	**Instance**	Global_Reg_PRM
Description	This register provides control over the device source clock.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																								SYSCLKDIV		RESERVED	AUTOEXTCLKMODE		RESERVED	SYSCLKSEL	

Bits	Field Name	Description	Type	Reset
31:8	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x000000
7:6	SYSCLKDIV	This field controls the system clock input divider 0x0: Reserved 0x1: Syst_clk is external clock / 1 0x2: Syst_clk is external clock / 2 0x3: Reserved	RW	0x1
5	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x0
4:3	AUTOEXTCLKMODE	This field allows to control the external clock request (CLKREQ) and the oscillator 0x0: CLKREQ is kept asserted or the oscillator is always active (in master mode) 0x1: CLKREQ is de-asserted or the oscillator is put in power-down mode (in master mode) when all the voltages domains are SLEEP, RETENTION or OFF states. 0x2: CLKREQ is de-asserted or the oscillator is put in power-down mode (in master mode) when all the voltages domains are RETENTION or OFF states. 0x3: CLKREQ is de-asserted or the oscillator is put in power-down mode (in master mode) only when all the voltage domains are in OFF states.	RW	0x0
2	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x0
1:0	SYSCLKSEL	This field reflects the mode of the ocillator. It is automatically set accordingly to the external tied-off configuration and its value is unsignificant before the release of the power-on reset. 0x0: Bypass mode: the system clock is issued from an external square clock source 0x1: Oscillator mode: the system clock is issued from an external quartz 0x2: Reserved 0x3: Unknow state (not know before release of the power-on reset)	R	0x3

Table 4-457. Register Call Summary for Register PRM_CLKSRC_CTRL

PRCM Clock Manager Functional Description
- External Clock Inputs: [0]
- System Clock Oscillator Control: [1] [2]
- PRM Source-Clock Controls: [3] [4]

PRCM Basic Programming Model
- System Clock Control Registers: [5]

PRCM Register Manual
- Global_Reg_PRM Registers: [6]

Table 4-458. PRM_OBS

Address Offset	0x0000 0080		
Physical Address	0x4830 7280	**Instance**	Global_Reg_PRM
Description	This register logs the observable signals (18 bits). This register is intended to be read through the debugger interface when the device is in OFF mode.		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								OBS_BUS																							

Bits	Field Name	Description	Type	Reset
31:18	RESERVED	Read is undefined.	R	0x0000
17:0	OBS_BUS	Indicates the current value on the observable bus.	R	0x00000

Table 4-459. Register Call Summary for Register PRM_OBS

PRCM Register Manual
- Global_Reg_PRM Registers: [0]

Table 4-460. PRM_VOLTSETUP1

Address Offset	0x0000 0090		
Physical Address	0x4830 7290	**Instance**	Global_Reg_PRM
Description	This register allows setting the setup time of the VDD1 and VDD2 regulators. This register is used when exiting off/retention/sleep mode (or entering off/retention/sleep mode) and when OMAP manages the sequencing of the voltages regulation steps.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SETUP_TIME2																SETUP_TIME1															

Bits	Field Name	Description	Type	Reset
31:16	SETUP_TIME2	The value, in number of cycles of SYS_CLK, determines the setup duration of VDD2 regulator. The setup duration is computed as = 8 x number of cycles of SYS_CLK set in the register bit field.	RW	0x0000
15:0	SETUP_TIME1	The value, in number of cycles of SYS_CLK, determines the setup duration of VDD1 regulator. The setup duration is computed as = 8 x number of cycles of SYS_CLK set in the register bit field.	RW	0x0000

Table 4-461. Register Call Summary for Register PRM_VOLTSETUP1

PRCM Voltage Management Functional Description • Direct Control With VMODE Signals: [0]
PRCM Off-Mode Management • Device Off-Mode Sequences: [1] [2] • Sleep Sequences: [3] [4] • Wake-Up Sequences: [5] [6]
PRCM Basic Programming Model • External Voltage Control Register Descriptions: [7] [8] [9] [10] [11]
PRCM Use Cases and Tips • Programming Sequence: [12] [13]
PRCM Register Manual • Global_Reg_PRM Registers: [14]

Table 4-462. PRM_VOLTOFFSET

Address Offset	0x0000 0094		
Physical Address	0x4830 7294	**Instance**	Global_Reg_PRM
Description	This register allows controlling the sys_offmode signal upon wake-up from OFF mode when the OFF sequence is supervised by the Power IC. This register allows setting the offset-time to de-assert sys_offmode when exiting the OFF mode.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																OFFSET_TIME															

Bits	Field Name	Description	Type	Reset
31:16	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x0000
15:0	OFFSET_TIME	Number of 32kHz clock cycles for the OFF mode offset time	RW	0x0000

Table 4-463. Register Call Summary for Register PRM_VOLTOFFSET

PRCM Off-Mode Management • Device Off-Mode Sequences: [0]
PRCM Basic Programming Model • External Voltage Control Register Descriptions: [1]
PRCM Register Manual • Global_Reg_PRM Registers: [2]

Table 4-464. PRM_CLKSETUP

Address Offset	0x0000 0098		
Physical Address	0x4830 7298	**Instance**	Global_Reg_PRM
Description	This register allows setting the setup time of the oscillator system clock (sys_clk), based on number of 32 kHz clock cycles.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																SETUP_TIME															

Bits	Field Name	Description	Type	Reset
31:16	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x0000
15:0	SETUP_TIME	Number of 32kHz clock cycles for the SETUP duration	RW	0x0000

Table 4-465. Register Call Summary for Register PRM_CLKSETUP

PRCM Clock Manager Functional Description • System Clock Oscillator Control: [0] • PRM Source-Clock Controls: [1]
PRCM Off-Mode Management • Device Off-Mode Sequences: [2]
PRCM Basic Programming Model • System Clock Control Registers: [3] • External Voltage Control Register Descriptions: [4] [5]
PRCM Register Manual • Global_Reg_PRM Registers: [6]

Table 4-466. PRM_POLCTRL

Address Offset	0x0000 009C		
Physical Address	0x4830 729C	**Instance**	Global_Reg_PRM
Description	This register allows setting the polarity of device outputs control signals.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																												OFFMODE_POL	CLKOUT_POL	CLKREQ_POL	EXTVOL_POL

Bits	Field Name	Description	Type	Reset
31:4	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x0000000
3	OFFMODE_POL	Controls the polarity of the sys_offmode signal 0x0: sys_offmode is active low 0x1: sys_offmode is active high	RW	0x1
2	CLKOUT_POL	Controls the external output clock polarity when disabled 0x0: sys_clkout is gated low when inactive 0x1: sys_clkout is gated high when inactive	RW	0x0
1	CLKREQ_POL	Controls the polarity of the sys_clkreq signal 0x0: sys_clkreq is active low 0x1: sys_clkreq is active high	RW	0x1
0	EXTVOL_POL	Controls the polarity of sys_vmode signal 0x0: sys_vmode signal is active low 0x1: sys_vmode signal is active high	RW	0x0

Table 4-467. Register Call Summary for Register PRM_POLCTRL

PRCM Clock Manager Functional Description • Clock Request (sys_clkreq) Control: [0] [1] [2] [3] [4] • System Clock Oscillator Control: [5] [6] • External Output Clock1 (sys_clkout1) Control: [7] • PRM Source-Clock Controls: [8]
PRCM Voltage Management Functional Description • Direct Control With VMODE Signals: [9] [10]
PRCM Use Cases and Tips • Programming Sequence: [11]

Table 4-467. Register Call Summary for Register PRM_POLCTRL (continued)

PRCM Register Manual • Global_Reg_PRM Registers: [12]

Table 4-468. PRM_VOLTSETUP2

Address Offset	0x0000 00A0		
Physical Address	0x4830 72A0	**Instance**	Global_Reg_PRM
Description	This register allows setting the overall setup time of VDD1 and VDD2 regulators. This register is used when exiting OFF mode and when the Power IC manages the sequencing of the voltages regulation steps.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																OFFMODESETUPTIME															

Bits	Field Name	Description	Type	Reset
31:16	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x0000
15:0	OFFMODESETUPTIME	Number of 32kHz clock cycles for the overall setup time of VDD1 and VDD2 regulators.	RW	0x0000

Table 4-469. Register Call Summary for Register PRM_VOLTSETUP2

PRCM Basic Programming Model • External Voltage Control Register Descriptions: [0] [1] [2] [3]
PRCM Register Manual • Global_Reg_PRM Registers: [4]

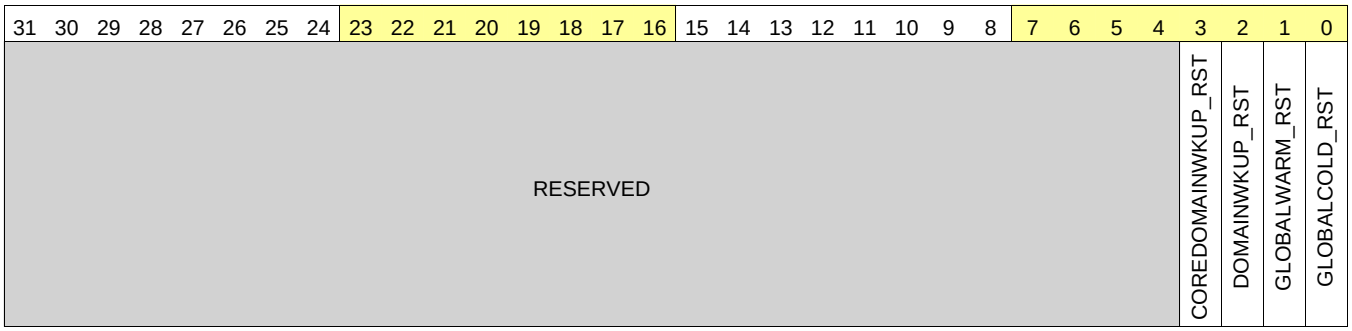
4.14.2.14 NEON_PRM Registers

Table 4-470. NEON_PRM Register Summary

Register Name	Type	Register Width (Bits)	Address Offset	Physical Address	Reset Type
RM_RSTST_NEON	RW	32	0x0000 0058	0x4830 7358	C
PM_WKDEP_NEON	RW	32	0x0000 00C8	0x4830 73C8	W
PM_PWSTCTRL_NEON	RW	32	0x0000 00E0	0x4830 73E0	W
PM_PWSTST_NEON	R	32	0x0000 00E4	0x4830 73E4	C
PM_PREPWSTST_NEON	RW	32	0x0000 00E8	0x4830 73E8	C

Table 4-471. RM_RSTST_NEON

Address Offset	0x0000 0058		
Physical Address	0x4830 7358	**Instance**	NEON_PRM
Description	This register logs the different reset sources of the NEON domain. Each bit is set upon release of the domain reset signal. Must be cleared by software.		
Type	RW		



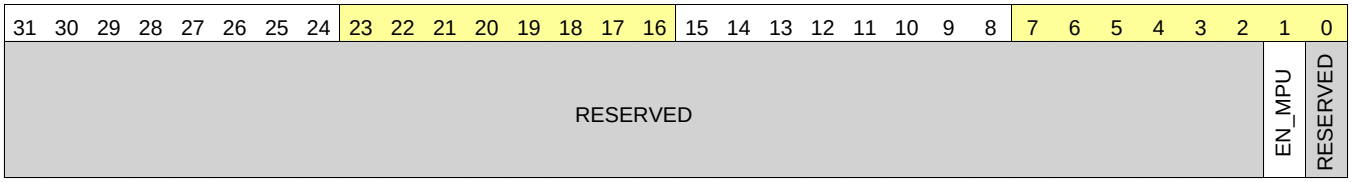
Bits	Field Name	Description	Type	Reset
31:4	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x0000000
3	COREDOMAINWKUP_RST	CORE domain wake-up reset Read 0x0: No power domain wake-up reset. Write 0x0: Status bit unchanged Read 0x1: NEON domain has been reset following a CORE power domain wake-up from OFF to ON. Write 0x1: Status bit is cleared to 0.	RW	0x0
2	DOMAINWKUP_RST	Power domain wake-up reset Read 0x0: No power domain wake-up reset. Write 0x0: Status bit unchanged Read 0x1: NEON domain has been reset following a NEON power domain wake-up. Write 0x1: Status bit is cleared to 0.	RW	0x0
1	GLOBALWARM_RST	Global warm reset Read 0x0: No global warm reset. Write 0x0: Status bit unchanged Read 0x1: NEON domain has been reset upon a global warm reset Write 0x1: Status bit is cleared to 0.	RW	0x0
0	GLOBALCOLD_RST	Global cold reset Read 0x0: No global cold reset. Write 0x0: Status bit unchanged Read 0x1: NEON domain has been reset upon a global cold reset Write 0x1: Status bit is cleared to 0.	RW	0x1

Table 4-472. Register Call Summary for Register RM_RSTST_NEON

PRCM Basic Programming Model • Reset Control: [0]
PRCM Register Manual • NEON_PRM Registers: [1]

Table 4-473. PM_WKDEP_NEON

Address Offset	0x0000 00C8		
Physical Address	0x4830 73C8	**Instance**	NEON_PRM
Description	This register allows enabling or disabling the wake-up of the NEON domain upon another domain wakeup.		
Type	RW		



Bits	Field Name	Description	Type	Reset
31:2	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x00000000
1	EN_MPU	MPU domain dependency 0x0: NEON domain is independent of MPU domain wake-up. 0x1: NEON domain is woken-up upon MPU domain wake-up.	RW	0x1
0	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x0

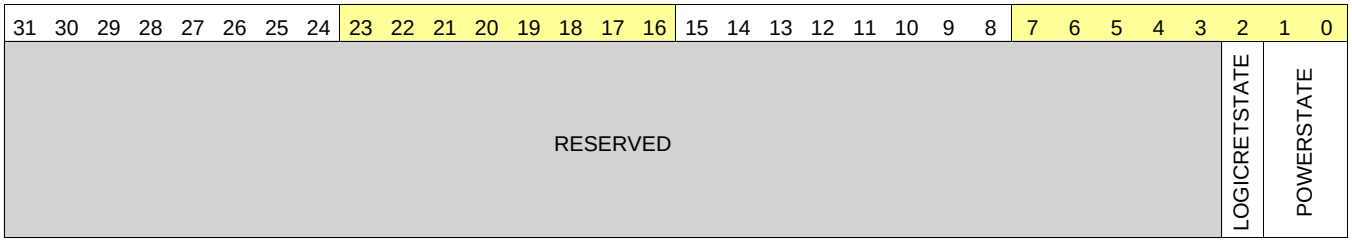
Table 4-474. Register Call Summary for Register PM_WKDEP_NEON

PRCM Idle and Wake-Up Management
- Device Wake-Up Events: [0]

PRCM Register Manual
- NEON_PRM Registers: [1]

Table 4-475. PM_PWSTCTRL_NEON

Address Offset	0x0000 00E0		
Physical Address	0x4830 73E0	**Instance**	NEON_PRM
Description	This register controls the NEON domain power state transition.		
Type	RW		



Bits	Field Name	Description	Type	Reset
31:3	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x00000000
2	LOGICRETSTATE	Logic state when RETENTION 0x1: Logic is always retained when domain is in RETENTION state.	R	0x1
1:0	POWERSTATE	Power state control 0x0: OFF state 0x1: RETENTION state 0x2: Reserved 0x3: ON state	RW	0x3

Table 4-476. Register Call Summary for Register PM_PWSTCTRL_NEON

PRCM Basic Programming Model
- PM_PWSTCTRL_ <domain_name> (Power State Control Register): [0]

PRCM Register Manual
- NEON_PRM Registers: [1]

Table 4-477. PM_PWSTST_NEON

Address Offset	0x0000 00E4		
Physical Address	0x4830 73E4	**Instance**	NEON_PRM
Description	This register provides a status on the power state transition of the NEON domain.		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED											INTRANSITION	RESERVED																		POWERSTATEST	

Bits	Field Name	Description	Type	Reset
31:21	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x000
20	INTRANSITION	Domain transition status 0x0: No transition 0x1: NEON power domain transition is in progress.	R	0x0
19:2	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x00000
1:0	POWERSTATEST	Current power state status 0x0: Power domain is OFF 0x1: Power domain is in RETENTION 0x2: Power domain is INACTIVE 0x3: Power domain is ON	R	0x3

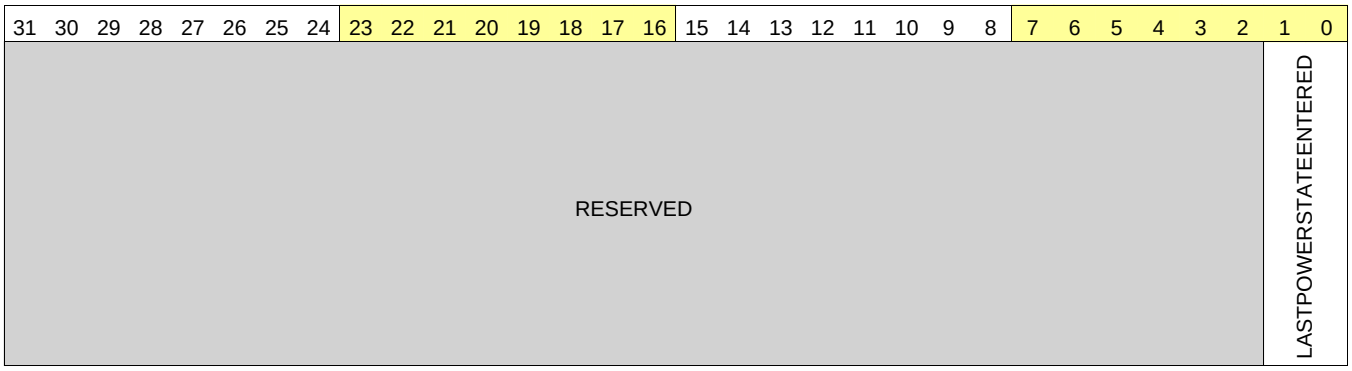
Table 4-478. Register Call Summary for Register PM_PWSTST_NEON

PRCM Basic Programming Model
- PM_PWSTST_ <domain_name> (Power State Status Register): [0]

PRCM Register Manual
- NEON_PRM Registers: [1]

Table 4-479. PM_PREPWSTST_NEON

Address Offset	0x0000 00E8		
Physical Address	0x4830 73E8	**Instance**	NEON_PRM
Description	This register provides a status on the NEON domain previous power state. It indicates the state entered during the last sleep transition.		
Type	RW		



Bits	Field Name	Description	Type	Reset
31:2	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x00000000
1:0	LASTPOWERSTATEENTERED	Last power state entered 0x0: NEON domain was previously OFF 0x1: NEON domain was previously in RETENTION 0x2: NEON domain was previously INACTIVE 0x3: NEON domain was previously ON	RW	0x0

Table 4-480. Register Call Summary for Register PM_PREPWSTST_NEON

PRCM Register Manual
- NEON_PRM Registers: [0]

4.14.2.15 USBHOST_PRM Registers

Table 4-481. USBHOST_PRM Register Summary

Register Name	Type	Register Width (Bits)	Address Offset	Physical Address	Reset Type
RM_RSTST_USBHOST	RW	32	0x0000 0058	0x4830 7458	C
PM_WKEN_USBHOST	RW	32	0x0000 00A0	0x4830 74A0	W
PM_MPUGRPSEL_USBHOST	RW	32	0x0000 00A4	0x4830 74A4	W
PM_IVA2GRPSEL_USBHOST	RW	32	0x0000 00A8	0x4830 74A8	W
PM_WKST_USBHOST	RW	32	0x0000 00B0	0x4830 74B0	W
PM_WKDEP_USBHOST	RW	32	0x0000 00C8	0x4830 74C8	W
PM_PWSTCTRL_USBHOST	RW	32	0x0000 00E0	0x4830 74E0	W
PM_PWSTST_USBHOST	R	32	0x0000 00E4	0x4830 74E4	C
PM_PREPWSTST_USBHOST	RW	32	0x0000 00E8	0x4830 74E8	C

Table 4-482. RM_RSTST_USBHOST

Address Offset	0x0000 0058		
Physical Address	0x4830 7458	**Instance**	USBHOST_PRM
Description	This register logs the different reset sources of the USB HOST domain. Each bit is set upon release of the domain reset signal. Must be cleared by software.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																												COREDOMAINWKUP_RST	DOMAINWKUP_RST	GLOBALWARM_RST	GLOBALCOLD_RST

Bits	Field Name	Description	Type	Reset
31:4	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x0000000
3	COREDOMAINWKUP_RST	CORE domain wake-up reset Read 0x0: No power domain wake-up reset. Write 0x0: Status bit unchanged Read 0x1: USB HOST domain has been reset following a CORE power domain wake-up from OFF to ON. Write 0x1: Status bit is cleared to 0.	RW	0x0
2	DOMAINWKUP_RST	Power domain wake-up reset Read 0x0: No power domain wake-up reset. Write 0x0: Status bit unchanged Read 0x1: USB HOST domain has been reset following an USB HOST power domain wake-up. Write 0x1: Status bit is cleared to 0.	RW	0x0
1	GLOBALWARM_RST	Global warm reset Read 0x0: No global warm reset. Write 0x0: Status bit unchanged Read 0x1: USB HOST domain has been reset upon a global warm reset Write 0x1: Status bit is cleared to 0.	RW	0x0
0	GLOBALCOLD_RST	Global cold reset Read 0x0: No global cold reset. Write 0x0: Status bit unchanged Read 0x1: USB HOST domain has been reset upon a global cold reset Write 0x1: Status bit is cleared to 0.	RW	0x1

Table 4-483. Register Call Summary for Register RM_RSTST_USBHOST

PRCM Basic Programming Model

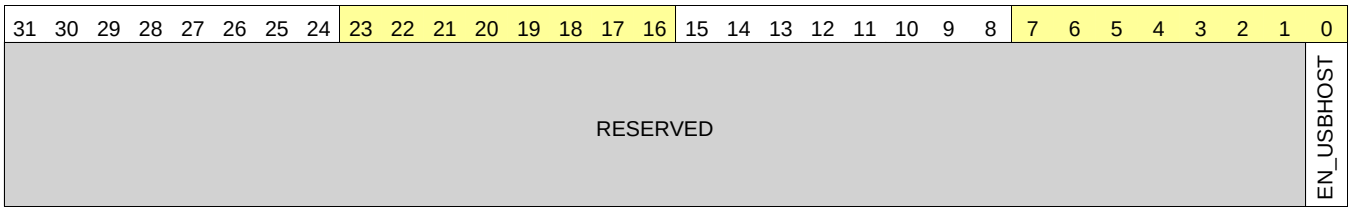
- Reset Control: [0]

PRCM Register Manual

- USBHOST_PRM Registers: [1]

Table 4-484. PM_WKEN_USBHOST

Address Offset	0x0000 00A0		
Physical Address	0x4830 74A0	**Instance**	USBHOST_PRM
Description	This register allows enabling/disabling modules wake-up events.		
Type	RW		



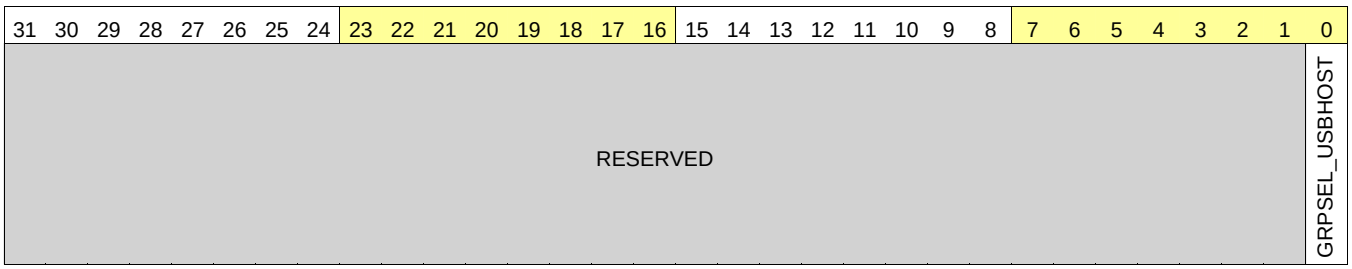
Bits	Field Name	Description	Type	Reset
31:1	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x00000000
0	EN_USBHOST	USB HOST Wake-up enable 0x0: USB HOST wake-up is disabled 0x1: USB HOST wake-up event is enabled	RW	0x1

Table 4-485. Register Call Summary for Register PM_WKEN_USBHOST

PRCM Idle and Wake-Up Management • Device Wake-Up Events: [0]
PRCM Basic Programming Model • Domain Wake-Up Control Registers: [1]
PRCM Register Manual • USBHOST_PRM Registers: [2]

Table 4-486. PM_MPUGRPSEL_USBHOST

Address Offset	0x0000 00A4		
Physical Address	0x4830 74A4	**Instance**	USBHOST_PRM
Description	This register allows selecting the group of modules that wake-up the MPU.		
Type	RW		



Bits	Field Name	Description	Type	Reset
31:1	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x00000000
0	GRPSEL_USBHOST	Select the USBHOST in the MPU wake-up events group 0x0: USBHOST is not attached to the MPU wake-up events group. 0x1: USBHOST is attached to the MPU wake-up events group.	RW	0x1

Table 4-487. Register Call Summary for Register PM_MPUGRPSEL_USBHOST

PRCM Basic Programming Model
- Domain Wake-Up Control Registers: [0]

PRCM Register Manual
- USBHOST_PRM Registers: [1]

Table 4-488. PM_IVA2GRPSEL_USBHOST

Address Offset	0x0000 00A8		
Physical Address	0x4830 74A8	**Instance**	USBHOST_PRM
Description	This register allows selecting the group of modules that wake-up the IVA2.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																															GRPSEL_USBHOST

Bits	Field Name	Description	Type	Reset
31:1	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x00000000
0	GRPSEL_USBHOST	Select the USBHOST in the IVA2 wake-up events group 0x0: USBHOST is not attached to the IVA2 wake-up events group. 0x1: USBHOST is attached to the IVA2 wake-up events group.	RW	0x1

Table 4-489. Register Call Summary for Register PM_IVA2GRPSEL_USBHOST

PRCM Basic Programming Model
- Domain Wake-Up Control Registers: [0]

PRCM Register Manual
- USBHOST_PRM Registers: [1]

Table 4-490. PM_WKST_USBHOST

Address Offset	0x0000 00B0		
Physical Address	0x4830 74B0	**Instance**	USBHOST_PRM
Description	This register logs module wake-up events. Must be cleared by software. If it is not cleared, it prevents further domain transition.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																															ST_USBHOST

Bits	Field Name	Description	Type	Reset
31:1	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x00000000
0	ST_USBHOST	USB HOST Wake-up status Read 0x0: USB HOST wake-up did not occur or was masked. Write 0x0: Status bit unchanged Read 0x1: USB HOST wake-up occurred. Write 0x1: Status bit is cleared to 0.	RW	0x0

Table 4-491. Register Call Summary for Register PM_WKST_USBHOST

PRCM Basic Programming Model
- Domain Wake-Up Control Registers: [0]

PRCM Register Manual
- USBHOST_PRM Registers: [1]

Table 4-492. PM_WKDEP_USBHOST

Address Offset	0x0000 00C8		
Physical Address	0x4830 74C8	**Instance**	USBHOST_PRM
Description	This register allows enabling or disabling the wake-up of the USB HOST domain upon another domain wakeup.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																											EN_WKUP	RESERVED	EN_IVA2	EN_MPU	EN_CORE

Bits	Field Name	Description	Type	Reset
31:5	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x0000000
4	EN_WKUP	WAKEUP domain dependency 0x0: USB HOST domain is independent of WKUP domain wake-up event. 0x1: USB HOST domain is woken-up upon WKUP domain wake-up event.	RW	0x1
3	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x0
2	EN_IVA2	IVA2 domain dependency 0x0: USB HOST domain is independent of IVA2 domain wake-up event. 0x1: USB HOST domain is woken-up upon IVA2 domain wake-up event.	RW	0x1
1	EN_MPU	MPU domain dependency 0x0: USB HOST domain is independent of MPU domain wake-up. 0x1: USB HOST domain is woken-up upon MPU domain wake-up.	RW	0x1
0	EN_CORE	CORE domain dependency 0x0: USB HOST domain is independent of CORE domain wake-up. 0x1: USB HOST domain is woken-up upon CORE domain wake-up.	RW	0x1

Table 4-493. Register Call Summary for Register PM_WKDEP_USBHOST

PRCM Idle and Wake-Up Management • Device Wake-Up Events: [0] [1] [2] [3]
PRCM Register Manual • USBHOST_PRM Registers: [4]

Table 4-494. PM_PWSTCTRL_USBHOST

Address Offset	0x0000 00E0		
Physical Address	0x4830 74E0	**Instance**	USBHOST_PRM
Description	This register controls the USB HOST domain power state transition.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED														MEMONSTATE		RESERVED							MEMRETSTATE	RESERVED			SAVEANDRESTORE	RESERVED	LOGICRETSTATE	POWERSTATE	

Bits	Field Name	Description	Type	Reset
31:18	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x0000
17:16	MEMONSTATE	Memory state when ON 0x3: Memory is always ON when domain is ON.	R	0x3
15:9	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x00
8	MEMRETSTATE	Memory state when RETENTION 0x1: Memory is always retained when domain is in RETENTION state.	R	0x1
7:5	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x0
4	SAVEANDRESTORE	Save And Restore mechanism for the USB HOST module 0x0: Disable the save and restore mechanism for the USB HOST module 0x1: Enable the save and restore mechanism for the USB HOST module	RW	0x0
3	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x0
2	LOGICRETSTATE	Logic state when RETENTION 0x1: Logic is always retained when domain is in RETENTION state.	R	0x1
1:0	POWERSTATE	Power state control 0x0: OFF state 0x1: RETENTION state 0x2: Reserved 0x3: ON state	RW	0x3

Table 4-495. Register Call Summary for Register PM_PWSTCTRL_USBHOST

PRCM Clock Manager Functional Description • Power Domain Clock Distribution: [0] • USBHOST Power Domain Clock Controls: [1]

Table 4-495. Register Call Summary for Register PM_PWSTCTRL_USBHOST (continued)

PRCM Idle and Wake-Up Management • USBHOST/USBTLL Save-and-Restore Management: [2] • USBHOST SAR Sequences: [3] [4] [5]
PRCM Basic Programming Model • PM_PWSTCTRL_ <domain_name> (Power State Control Register): [6]
PRCM Register Manual • USBHOST_PRM Registers: [7]

Table 4-496. PM_PWSTST_USBHOST

Address Offset	0x0000 00E4		
Physical Address	0x4830 74E4	**Instance**	USBHOST_PRM
Description	This register provides a status on the power state transition of the USB HOST domain.		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED											INTRANSITION	RESERVED																		POWERSTATEST	

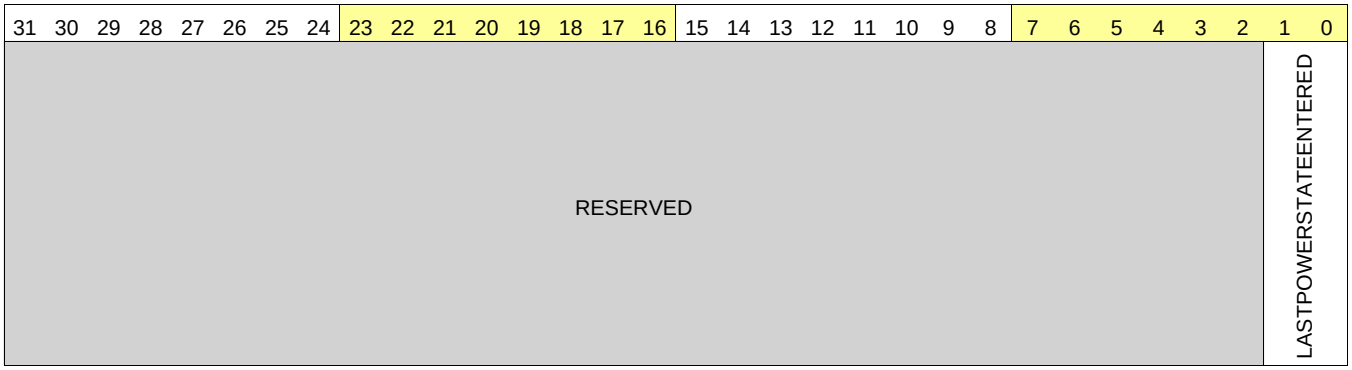
Bits	Field Name	Description	Type	Reset
31:21	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x000
20	INTRANSITION	Domain transition status 0x0: No transition 0x1: USB HOST power domain transition is in progress.	R	0x0
19:2	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x00000
1:0	POWERSTATEST	Current power state status 0x0: Power domain is OFF 0x1: Power domain is in RETENTION 0x2: Power domain is INACTIVE 0x3: Power domain is ON	R	0x3

Table 4-497. Register Call Summary for Register PM_PWSTST_USBHOST

PRCM Basic Programming Model • PM_PWSTST_ <domain_name> (Power State Status Register): [0]
PRCM Register Manual • USBHOST_PRM Registers: [1]

Table 4-498. PM_PREPWSTST_USBHOST

Address Offset	0x0000 00E8		
Physical Address	0x4830 74E8	**Instance**	USBHOST_PRM
Description	This register provides a status on the USBHOST domain previous power state. It indicates the state entered during the last sleep transition.		
Type	RW		



Bits	Field Name	Description	Type	Reset
31:2	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x00000000
1:0	LASTPOWERSTATEENTERED	Last power state entered	RW	0x0
		0x0: USB HOST domain was previously OFF		
		0x1: USB HOST domain was previously in RETENTION		
		0x2: USB HOST domain was previously INACTIVE		
		0x3: USB HOST domain was previously ON		

Table 4-499. Register Call Summary for Register PM_PREPWSTST_USBHOST

PRCM Register Manual

- USBHOST_PRM Registers: [0]

Interconnect

This chapter describes the OMAP34xx multimedia device interconnect.

Notes:

- The L3 interconnect is an instantiation of the **SonicsMX®** interconnect from Sonics, Inc.
- The L4 interconnects are instantiations of the **Sonics3220™** interconnect from Sonics, Inc.

This document contains materials that are ©2003-2007 Sonics, Inc., and that constitute proprietary information of Sonics, Inc.

SonicsMX and Sonics3220, are trademarks or registered trademarks of Sonics, Inc. All such materials and trademarks are used under license from Sonics, Inc. For additional information, see the SonicsMX or Sonics3220 Reference manuals, or contact Sonics, Inc.

SMX is an abbreviation for SonicsMX.

Note: This chapter gives information about all modules and features in the high-tier device. See Section 1.5, *OMAP34xx Family*, to check availability of modules and features. To flag interconnect response being blocked, the time-out of target agents attached to unavailable modules can be enabled with the lowest setting.

Topic **Page**

5.1 Interconnect Overview 614
5.2 L3 Interconnect 623
5.3 L4 Interconnects 677

5.1 Interconnect Overview

5.1.1 Terminology

The following terminology is critical to understanding the interconnect:

- Initiator: Module able to initiate read and write requests to the chip interconnect (typically: processors, DMA, etc.).
- Target: Unlike an initiator, a target module cannot generate read/write requests to the chip interconnect, but it can respond to these requests. However, it may generate interrupts or a DMA request to the system (typically: peripherals, memory controllers).

Note: A module can have several separate ports; therefore, a module can be both an initiator and a target.

- Agent: Each connection of one module to one interconnect is done using an agent, which is an adaptation (sometimes configurable) between the module and the interconnect. A target module is connected by a target agent (TA), and an initiator module is connected by an initiator agent (IA).
- OCP: Open-core protocol (www.ocpip.org) is point-to-point standard protocol between one master port and one slave port.
- OCP master port: Port that can generate OCP commands. An initiator includes at least one master port.
- OCP slave port: Port that responds to OCP commands. A target includes one slave port.
- Interconnect: The decoding, routing, and arbitration logic that enables the connection between multiple initiator modules and multiple target modules connected on it.
- Register target (RT): Special TA used to access the interconnect internal configuration registers
- Dataflow signal: Any OCP signal that is part of a clearly identified OCP transfer or dataflow (typically: command, address, byte enables, etc.). The signal behavior is defined by the OCP protocol semantics.
- Sideband signal: Any OCP signal whose behavior is not associated to a precise OCP transaction or dataflow. The OCP standard does not define specific semantics for these signals.
- Out-of-band error: Any OCP signal whose behavior is associated to an error-reporting scheme of the device, as opposed to in-band errors.

Note: Interrupt requests and DMA requests are not routed by the interconnect in the device.

- Firewall: A programmable feature integrated in a target agent or L4 interconnect to prevent unauthorised access to or from a module. A firewall can be configured using three criteria:
  - Initiator requesting access
  - Address space access
  - Type of access
- Thread: Logical entities that allow to have separate independent data flows on a single port.
- Multithreaded ports: A physical port able to simultaneously handle several outstanding transactions. On a multithreaded port, one physical channel (port) is used concurrently for several logical channels (threads). The transfer in each thread must remain in order with respect to each other, but the order between threads can change between requests and responses. Thread management is used for performance optimization purposes and is automatically handled by the system.
- ConnID: Any transaction in the system interconnect is tagged by an in-band qualifier ConnID, which uniquely identifies the initiator at a given interconnect point. A ConnID is transmitted inband with the request and is used for firewall and error-logging mechanism.
- Firewall comparison mechanism: A comparison made in the firewall between access in-band qualifiers and access permissions that are programmed in the firewall configuration registers. If the comparison is successful, access is allowed; otherwise, access is denied.
- MCmd qualifier: Command bus that indicates the type of transfer requested. Table 5-1 lists the

commands encoded.

Table 5-1. MCmd Qualifier Description

MCmd[2:0]	Transaction Type
0 0 0	Idle
0 0 1	Write
0 1 0	Read
0 1 1	ReadEx
1 0 0	Not used
1 0 1	Write nonposted
1 1 0	Not used
1 1 1	Not used

- MReqInfo qualifier: Four MReqInfo qualifiers describe the access during the use of the firewall comparison mechanism, as described in Table 5-2.

Table 5-2. MReqInfo Qualifier Description

Qualifiers	Description
MReqType	0: Data access
	1: Opcode fetch
MReqSupervisor	0: User mode
	1: Supervisor mode
MReqDebug	0: Functional access
	1: Debug access

- L3_PM_REQ_INFO_PERMISSION_i: Register that configures the combination of the MReqInfo, allowing access permission to the TM based on the MReqInfo in-band qualifier values.
- SError: Target that indicates an error condition to the initiator.
- SResp qualifier: Response from the target to the initiator concerning the transaction.

Table 5-3. SResp Qualifier Description

SResp[1:0]	Description
0 0	No response
0 1	Data valid/accept
1 0	Not used
1 1	Error

5.1.2 Architecture Overview

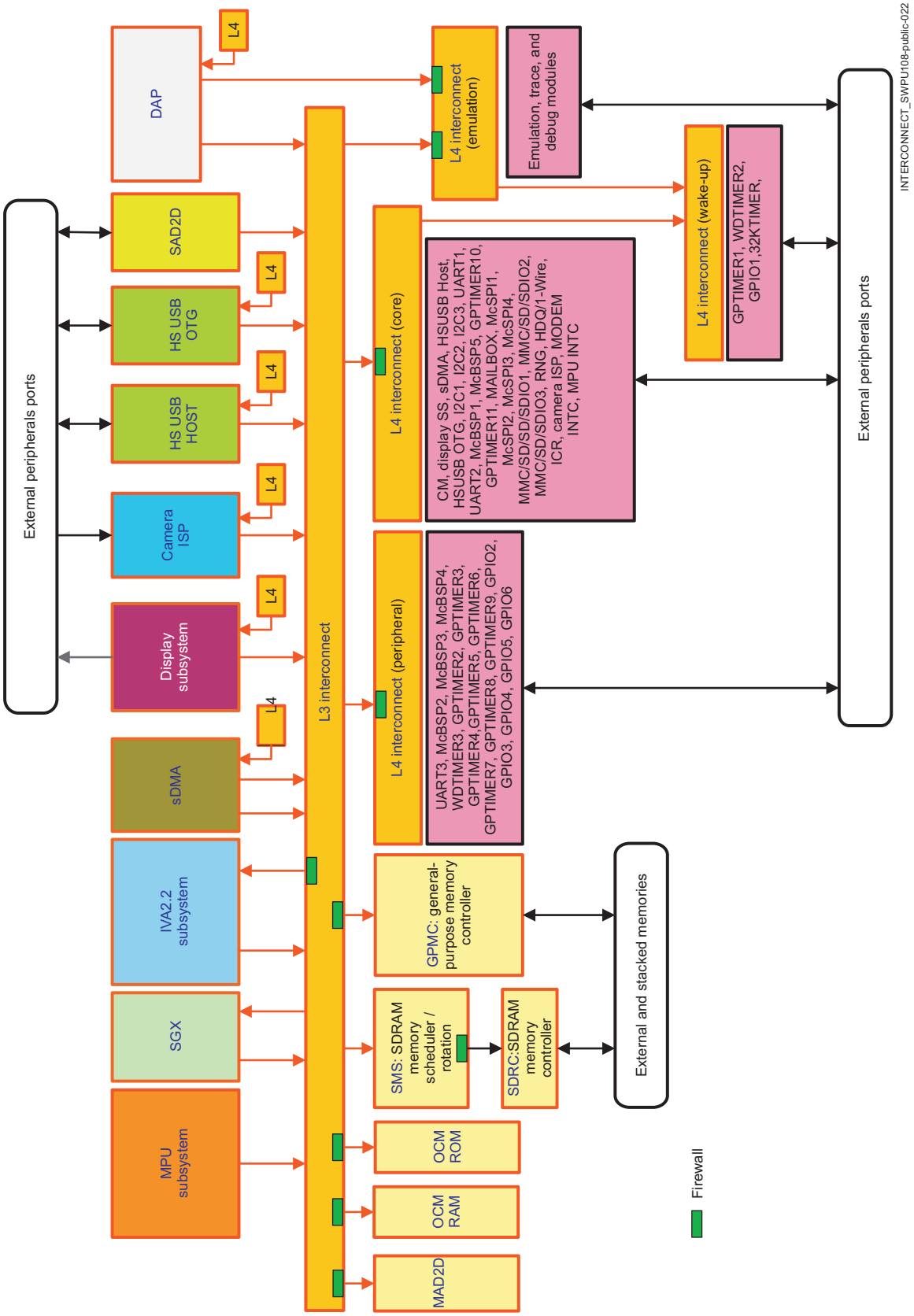
The device memory hierarchy includes four levels:

- L1 is internal to the CPUs. It concerns data exchange with the internal Level1 cache memory subsystem, and it is the closest memory to the microprocessor unit (MPU) core and the IVA2.2 core.
- L2 is included in the IVA2.2 subsystem and the MPU subsystem.
- The chip-level interconnect consists of one L3 interconnect and four L4 interconnects. It enables communication among the modules and subsystems in the device.

Figure 5-1 shows an overview of the L3 and L4 interconnect architecture.

- L3 handles many types of data transfers, especially exchanges with system-on-chip/external memories. L3 transfers data with a maximum width of 64 bits from the initiator to the target. The L3 interconnect is a little-endian platform
- L4 is composed of the L4-Core, L4-Per, L4-Wakeup, and L4-Emu interconnects and handles data transfers to peripherals. It supports 32-bit data width transfer and is optimized to support the interconnection of many peripheral targets. These backplanes assume little-endian transactions for narrower targets (8-bit, 16-bit) when doing data packing and unpacking.

Figure 5-1. Interconnect Architecture Overview



Modules are connected to the interconnect through an IA for the initiator module and a TA for target modules. Each module/subsystem connection is statically configured to tune the access depending on the characteristics of the module.

To unauthorised a module or L4 interconnect access, some TAs include configurable firewalls (FWs). A firewall restricts or filters the accesses allowed to an initiator according to different access criteria. The firewalls can usually be configured by software.

The L3 and L4 interconnect default setting is fully functional; it enables all possible functional data paths and a minimal default protection setting. However, it is possible to modify the interconnect parameters to fit user expectations.

5.1.3 Module Distribution

IAs and TAs provide the interface to connect the different modules and the interconnect.

Table 5-4 through Table 5-13 list the device modules, subsystems, and associated agents. The agents are listed for each interconnect domain:

- L3 initiator and target agents
- L4-Core initiator and target agents
- L4-Per initiator and target agents
- L4-Emu initiator and target agents
- L4-Wakeup initiator and target agents

5.1.3.1 L3 Interconnect Agents

Table 5-4 and Table 5-5 list the IAs and TAs, respectively, of the L3 interconnect.

Table 5-4. L3 Initiator Agents

Module Name	Description
MPU SS	MPU subsystem port
Display SS	Display subsystem port
IVA2.2 SS	IVA2.2 subsystem port
SGX SS	Graphics subsystem port
CAMERA SS	Camera subsystem port
SAD2D	Die-to-die port
sDMA read	System DMA read port
sDMA write	System DMA write port
High-Speed (HS) USB OTG	Universal Serial Bus High-Speed port OTG controller
High-Speed (HS) USB Host	Universal serial bus High-Speed port host controller
DAP	Debug access port (JTAG/Emulation access to system resources)

Table 5-5. L3 Target Agents

Module Name	Description
SMS	SDRAM memory scheduler port
GPMC	General-purpose memory controller (for flash memory, SRAM, SROM, etc.) port
OCM-ROM	On-chip memory ROM port
OCM-RAM	On-chip memory RAM port
SGX	Graphics subsystem port
IVA2.2	Image video and audio accelerator subsystem port
RT	Register target port to configure L3
L4-Core	Port for L4-Core interconnect

Table 5-5. L3 Target Agents (continued)

Module Name	Description
L4-Peripherals	Port for L4-Per interconnect
L4-Emu	Port for L4-Emu interconnect

For more details on the register target module, see Section 5.2.3.2, *Register Target*.

5.1.3.2 L4-Core Agents

Table 5-6. L4-Core Initiator Agent

Module Name	Description
L3 interconnect	L3 interconnect port

Note: A unique L3 port is used for communication with the L4-Core. For the list of initiators allowed to access the L4 Core peripherals, see Table 5-14.

Table 5-7. L4-Core Target Agents

Module Name	Description
Display subsystem	Display subsystem configuration port
Camera subsystem	Camera subsystem port
High-Speed (HS) USB OTG	Universal serial bus High-speed port OTG
High-Speed (FS) USB Host	Universal serial bus High-Speed port Host controller
UART1	Universal asynchronous receiver transmitter port 1
UART2	Universal asynchronous receiver transmitter port 2
I2C1	Multimaster interintegrated circuit 1
I2C2	Multimaster interintegrated circuit 2
I2C3	Multimaster interintegrated circuit 3
McBSP1	Multichannel buffered serial port 1
McBSP5	Multichannel buffered serial port 5
GPTIMER10	General-purpose timer 10
GPTIMER11	General-purpose timer 11
SPI1	Serial peripheral interface 1
SPI2	Serial peripheral interface 2
MMCHS1	Multimedia memory controller SDIO 1
MMCHS2	Multimedia memory controller SDIO 2
MMCHS3	Multimedia memory controller SDIO 3
HDQ/1-Wire	Single wire serial link low rate
MLB (Mailbox)	Mailbox
SPI1	Serial peripheral interface 1
SPI2	Serial peripheral interface 2
SPI3	Serial peripheral interface 3
SPI4	Serial peripheral interface 4
sDMA	System DMA controller
L4-Wakeup	L4-Wakeup interconnect
CM	Clock manager
SCM	System control module

5.1.3.3 L4-Per Agents

Table 5-8. L4-Per Initiator Agent

Module Name	Description
L3 interconnect	L3 interconnect port

Note: A unique L3 port is used for communication with L4-Per. For the list of initiators allowed to access the L4-Per peripherals, see Table 5-14.

Table 5-9. L4-Per Target Agents

Module Name	Description
UARTIrDA	Universal asynchronous receiver/transmitter and infrared data association port
McBSP2	Multichannel buffered serial port 2
McBSP3	Multichannel buffered serial port 3
McBSP4	Multichannel buffered serial port 4
GPTIMER2	General-purpose timer 2
GPTIMER3	General-purpose timer 3
GPTIMER4	General-purpose timer 4
GPTIMER5	General-purpose timer 5
GPTIMER6	General-purpose timer 6
GPTIMER7	General-purpose timer 7
GPTIMER8	General-purpose timer 8
GPTIMER9	General-purpose timer 9
GPIO2	General-purpose I/O 2
GPIO3	General-purpose I/O 3
GPIO4	General-purpose I/O 4
GPIO5	General-purpose I/O 5
GPIO6	General-purpose I/O 6

5.1.3.4 L4-Emu Agents

Table 5-10. L4-Emu Initiator Agents

Module Name	Description
L3 interconnect	L3 interconnect port
DAP	DAP port

Note: The L3 and DAP ports are used for communication with L4-Emu. For the list of initiators allowed to access the L4-Emu peripherals, see Table 5-14.

Table 5-11. L4-Emu Target Agents

Module Name	Description
L4-Wakeup	L4 wake-up interconnect
SDTI	System debug trace interface
ETB	Embedded trace buffer
TPIU	Trace port interface unit
MPU	ARM9
DAPCTL	Debug access port

5.1.3.5 L4-Wakeup Agents

Table 5-12. L4-Wakeup Initiator Agent

Module Name	Description
L4-Core interconnect	L4-Core interconnect port
L4-Emu interconnect	L4-Emulation interconnect port

Note: The L4-Emu and L4-Core ports are used to communicate with the L4-Wakeup. For the list of initiators allowed to access the L4-Wakeup peripherals, see Table 5-14.

Table 5-13. L4-Wakeup Target Agents

Module Name	Description
PRM	Power reset management
GPIO1	General-purpose I/O 1
GPTIMER1	General-purpose timer 1
WDTIMER2	MPU subsystem watchdog timer
32KTIMER	32-kHz timer

5.1.4 Connectivity Matrix

Table 5-14 lists the functional paths between the L3 interconnect initiator modules and the L3 and L4 TAs. The functional paths are indicated by the use of the following:

- Cell contains a + sign when a functional path exists.
- Cell is blank when no functional path exists.

Table 5-14. Connectivity Matrix

Initiator Ports	L4-Core Target (1)	L4-Per Target (1)	L4-Emu Target (1)	L4-Wakeup Target (1)	SMS Target	GPMC Target	OCM RAM Target	OCM ROM Target	RT Target	IVA2.2 Target	SGX Target	MAD2D
MPU IA	+	+	+	+	+	+	+	+	+	+	+	+
SGX IA					+	+	+					+
IVA2.2 IA	+	+	+	+	+	+	+		+	+		+
DSS IA					+		+					+
CAM IA					+		+					+
HS USB Host IA					+	+	+					+
HS USB OTG IA					+	+	+					+
sDMA RD IA	+	+	+	+	+	+	+			+	+	+
sDMA WR IA	+	+	+	+	+	+	+			+	+	+
DAP IA	+	+	+	+	+	+	+	+	+	+	+	+
SAD2D IA	+	+	+	+	+	+	+					

(1) A functional data path always exists from L4 IAs (Core, Per, Emu, and Wakeup) to any L4 target module (Core, Per, Emu, and Wakeup). As a consequence, all L3 initiator modules for which a data path exists to an L4 TA can access L4 peripherals. Restrictions on peripheral access depend on L4 protection mechanism. For more details, see Section 5.3.3.3.

5.2 L3 Interconnect

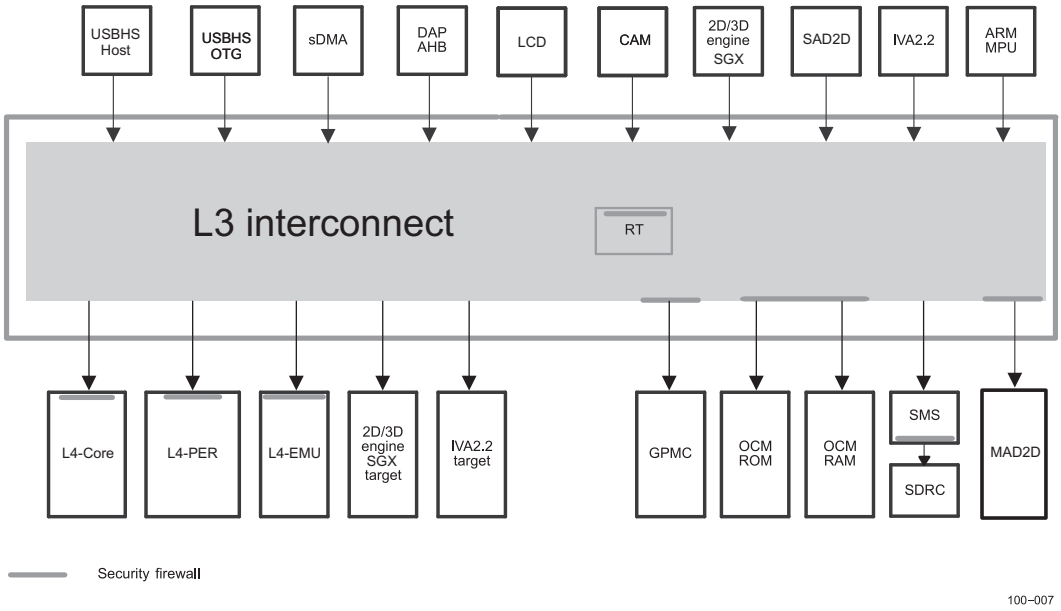
This section describes the L3 interconnect and its components. With the exception of register points, each component includes functionality for both the request and response network.

5.2.1 Overview

The L3 interconnect links cores in a flexible topology that couples low power with high performance. Innovative physical structures and advanced protocols ensure bandwidth and latency to individual IP cores, providing dedicated connections between IP cores and logical connections over a shared interconnect.

Figure 5-2 shows the L3 interconnect.

Figure 5-2. L3 Interconnect Overview



The following are the main features of the L3 interconnect:

- 64-bit multipath interconnect to eliminate on-chip bottlenecks
- Special internal target for access to L3 registers (RT)
- Guaranteed quality of service for real-time hardware operators, while maintaining optimal memory latency for MPU accesses to memory resources
- True little-endian platform
- Transaction error tracking and logging
- Built-in protection features:
  - Allow access only to authorized initiator
  - Distributed region-based firewalls for system resource sharing and protection management
- Signaling support for chip-level power management infrastructure
- Two interrupt line signaling transaction error

5.2.2 L3 Interconnect Integration

5.2.2.1 Clocking, Reset, and Power-Management Scheme

5.2.2.1.1 Clocks

The power, reset, and clock management (PRCM) module provides the L3_ICLK as the main clock to the L3 interconnect.

In addition to L3_ICLK, two additional sample signals are provided to module agents to allow internal synchronization. The modules are as follows:

- L4-Core
- L4-Per

For more details on the L3 clock and its setting, see Chapter 4, *Power, Reset, and Clock Management*.

Table 5-15. L3 Interconnect Clocks

Type	Name	Source	Description
Interface/functional	L3_ICLK	PRCM	Main clock for L3 interconnect

5.2.2.1.2 Resets

The L3 interconnect receives a single reset signal, CORE_RST, from the PRCM module. CORE_RST is the reset signal to the core power domain. (For more details, see Chapter 4, *Power, Reset, and Clock Management*.) When asserted, CORE_RST resets the L3 internal registers. There is no software reset for the L3 interconnect.

Table 5-16. L3 Interconnect Reset

Type	Reset Domain	Source	Description
Hardware	CORE_RST	PRCM	Asynchronous reset for the entire interconnect

5.2.2.1.3 Power Domain

The L3 interconnect connects into the CORE power domain, which can dynamically switch between supported OPPs. For more details on power voltage scaling, see Chapter 4, *Power, Reset, and Clock Management*.

Table 5-17. L3 Interconnect Power Domain

Interconnect	Power Domain
L3 interconnect	CORE

5.2.2.1.4 Power Management

As part of the system-wide power-management scheme, the L3 interconnect enters an idle state at the request of the PRCM module. (For more details, see Chapter 4, *Power, Reset, and Clock Management*.) The L3 interconnect is always in smart-idle mode; that is, it goes into idle state after receiving the request from the PRCM module once all transfer requests are serviced. This functionality is handled by hardware. The L3 interconnect sends an acknowledge signal back to the PRCM module when it enters the idle state.

5.2.2.2 Hardware Requests

5.2.2.2.1 Interrupt Requests

Three interrupt lines are present at the boundary of the L3 interconnect (see Table 5-18). They are used for hardware error management.

Table 5-18. L3 Interconnect Hardware Requests

Type	Name	Destination	Description
Interrupt	M_IRQ_9	MPU interrupt controller for debug errors	L3 interconnect provides a mechanism to group core-detected and internal interconnect errors. See Section 5.3.3.4, *Error Handling*.
Interrupt	M_IRQ_10	MPU interrupt controller for application errors	
Interrupt	IVA2_IRQ[39]	IVA2.2 interrupt controller for application errors	

5.2.3 L3 Interconnect Functional Description

5.2.3.1 Initiator Identification

An InitiatorID is assigned to every thread on every initiator socket. The ID uniquely identifies the initiator and thread for an interconnect transfer see Table 5-19. The interconnect uses InitiatorIDs for a number of purposes, including the following:

- Initiator source identification for the protection mechanism (see Section 5.2.3.3, *L3 Protection and Firewalls*)
- Response route generation (performed internally to the TAs)
- Firewall error logging
- L3 interconnect error logging

Table 5-19. InitiatorID Definition

Initiator	InitiatorID
HS USB Host	0
HS USB OTG	2
sDMA rd	3 , 4 , 5, 6
sDMA wr	7, 8
DAP	9
CAM	10, 11, 12
SGX	13
IVA2.2 SS DMA	14, 15, 16, 17, 18, 19
IVA2.2 SS	20, 21, 22
MPU SS	23, 24, 25, 26, 27
SAD2D	28
LCD	29

5.2.3.2 Register Target

An RT is a specialized TA used to access L3 interconnect internal configuration registers.

RT configuration options are a subset of those available for TAs. For more details, see Section 5.2.5.3.

5.2.3.3 L3 Protection and Firewalls

Protection in the device relies heavily on L3 firewalls and their configuration. Nine targets are protected through the use of firewalls. The number of protected regions varies on the target, with a maximum of eight regions. Table 5-20 lists the protection type and the number of protected regions for each target.

Table 5-20. Target Firewall and Region Configuration

Target	Firewall	Number of Regions
SMS	Included in the SMS module	See Chapter 11, *Memory Subsystem*.
GPMC	Yes	8
OCM-RAM	Yes	8
OCM-ROM	Yes	2
MAD2D	Yes	8
IVA2.2 target	Yes	4
L4-Core	Included in the L4 module	See Section 5.3, *L4 Interconnects*.
L4-Wakeup	Included in the L4 module	See Section 5.3, *L4 Interconnects*.
L4-Emu	Included in the L4 module	See Section 5.3, *L4 Interconnects*.
RT	Yes	2

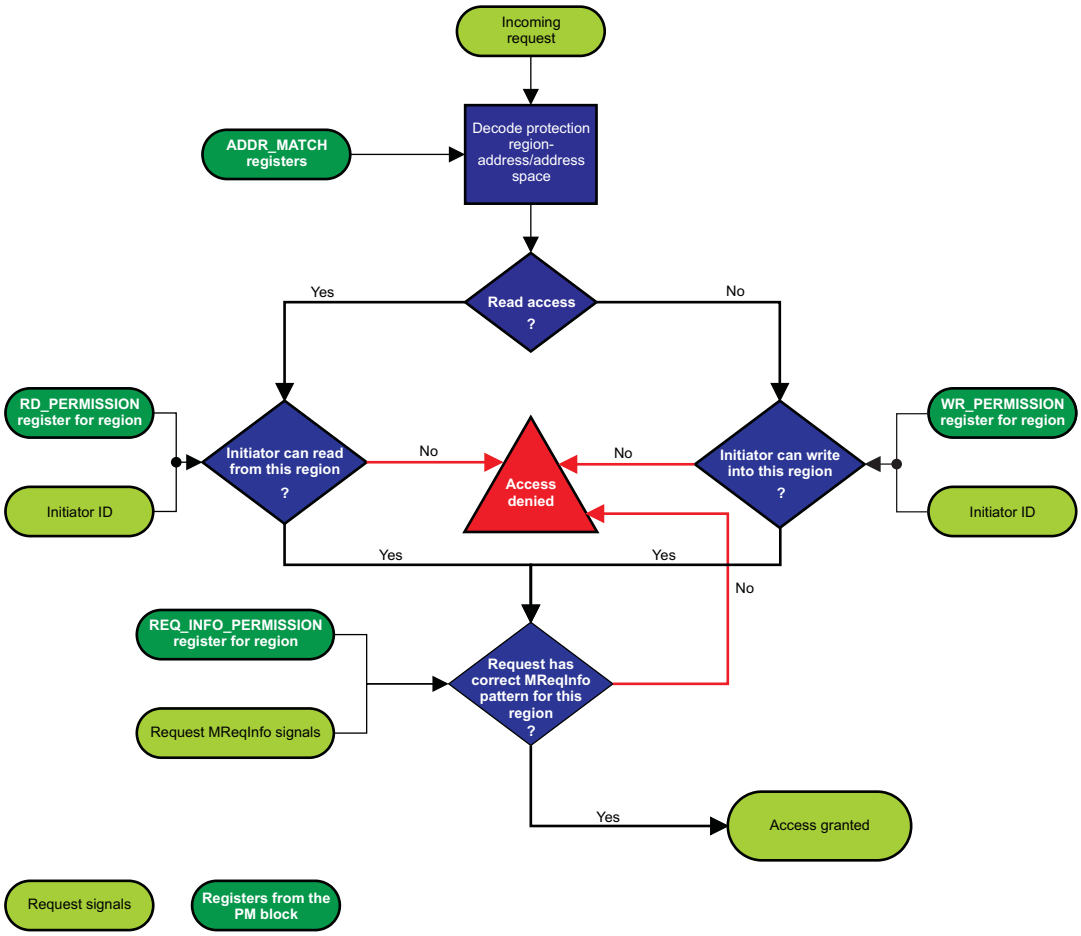
The protection mechanism designates protection regions within the address space of certain targets. Access to these regions is granted only for certain initiators, based on transaction attributes (transmitted through MReqInfo). Each protection region is characterized with several configurable attributes (base address, size, specific access rights, and priority setting).

The protection mechanism uses the following attributes of a request:

- The address field and the address space field are used to determine which region has been hit. The region ID selects one table entry of the firewall look-up table.
- The region ID points to a unique set of permission registers: Read_Permission, Write_Permission, ReqInfo_Permission.
- The Initiator ID is used to determine the permission of the initiator (read/write permission) with respect to the concerned region.
- The access types (read or write) and the transaction attributes (MReqInfo in-band qualifiers) are used to grant or reject the access.

The first check determines whether the type of incoming request (read or write) is allowed, according to the Initiator MConnID and its read and write permissions. The second check determines whether the MReqInfo bits of the incoming request are within the allowed pattern established by the MReqInfo permission bits. If both check results are positive, the request is allowed to access the target. Otherwise, the access is denied, the request is not forwarded to the target, an out-of-band error indication is reported, and an in-band error response is returned to the initiator (except for writes that were posted at the IA). Figure 5-3 shows the flow used to identify and accept a new request.

Figure 5-3. Flow Chart of the Protection Mechanism



To summarize, firewalls accept or reject a request depending on the following:

- Initiator originating the request
- Command (read or write) requested
- MReqInfo bus state
- Region access in the target memory space

Software must configure the L3 firewalls properly to allow the right initiators, with the right MReqInfo access, on the well-defined size region. All the registers relative to the L3 firewalls are grouped in the protection mechanism (PM) register block.

Note: The PM qualifies the protection mechanism register associated with a firewall target. The targets protected by a firewall are listed in Table 5-20. These PM registers do not exist if no firewall is associated with the target.

5.2.3.3.1 Protection Region

Two types of regions are distinguished in a target firewall (see Figure 5-4):

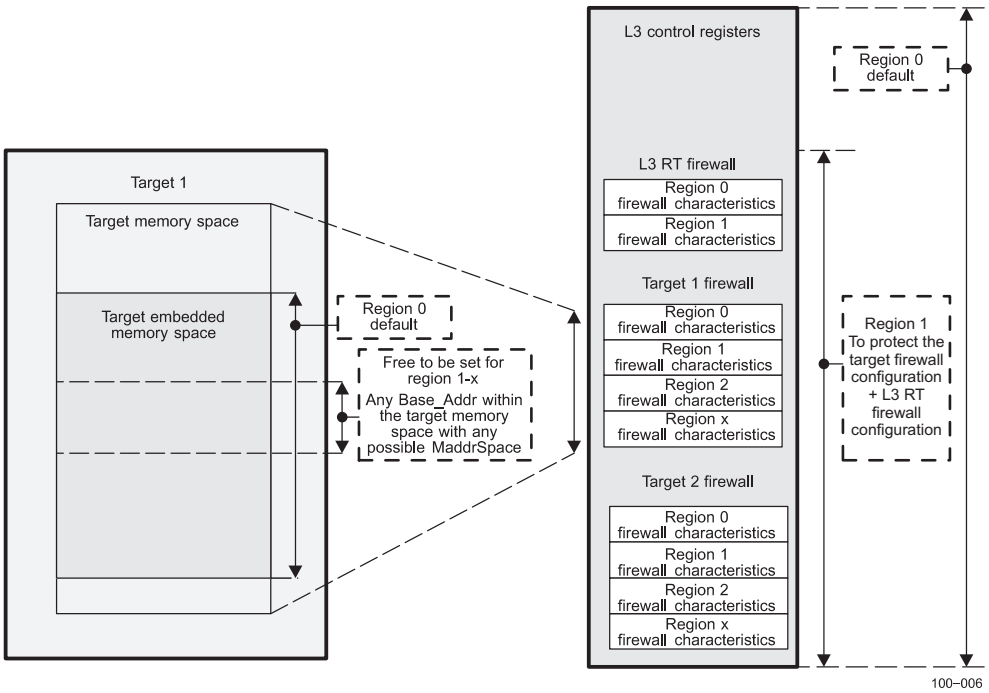
- Default region: Available in all targets; spans the entire target address range
- Normal region: Number varies in a target; they have identical capabilities

Each region has the following characteristics:

- A base address, relative to the target address itself, and an address space

- A size
- Specific access rights, as defined through the MReqInfo qualifiers
- A priority level, from 0 (lowest) to 3 (highest)

Figure 5-4. L3 Firewall Implementation



5.2.3.3.1.1 Default Region/Region0

Region 0 is the default region of a whole target; it spans the entire target address space.

The default region is always the lowest priority. This region is systematically overlapped when other regions are set.

The L3_PM_ADDR_MATCH_k register is not accessible. Its configuration corresponds to all the possible addresses for the target, including all of ADDR_SPACE. It is not possible to program multiple ADDR_SPACE addresses in a normal region.

5.2.3.3.1.2 Normal Regions

Normal regions have identical features.

A given request either maps into a specific protection region or is considered to have hit the default protection region if no normal region is hit.

The protection regions can only be configured for a memory space that is power-of-two in size and size-aligned using the SIZE bit field L3_PM_ADDR_MATCH_k [7:3] (as listed in Table 5-21). When the SIZE bit field L3_PM_ADDR_MATCH_k [7:3] is set to 0, the region is disabled.

Note: k denotes the region number. Depending on the target, n varies from 0 to 7.

Table 5-21 lists the size encoding for each value set in the SIZE bit field.

Table 5-21. L3 Firewall Size Parameter Definition

Possible Configuration		
Size[1]	**Region Size**	**Base_Addr**[2]
0x0	Region disabled	Any (nonsignificant)
0x1	1K-byte	0x0000000
		0x0000400
		0x0000800
		.
0x2	2K-byte	0x0000000
		0x0000800
		0x0001000
		...
...	...	...
0x17	2^(17-1)K-byte	
Others	Not allowed	-

(1) When the size parameter is set to 0, the region is disabled.
(2) The base address depends on user settings.

5.2.3.3.2 Priority Level Overview

Each L3 firewall region is prioritized. Depending on its priority level, a region can override the settings of another region.

- Region 0 is the only allowed priority 0 region (lowest priority).
- Region 1 is the only allowed priority 3 region (highest priority).
- Others regions are defined as priority 1 or 2.

CAUTION

LEVEL bitfield value of ADDR_MATCH_1 register (Region 1 firewall configuration) must be kept to his default reset value and must not be changed; otherwise, the result will be unpredictable and can create unexpected protection holes or denial of service.

Protection level is defined by the LEVEL bit L3_PM_ADDR_MATCH_k , where n is greater than 2. Figure 5-5 represents the priority level with associated regions.

When an address hits two or more regions with different priority levels, the highest priority region protections are applied. The overlay region can overlap all or a part of a nonoverlay region.

CAUTION

Configuring two overlapping protection regions with the same priority level leads to undefined behavior.

Hardware behavior in the case of overlapping protection regions is undefined. A region with higher priority must be used to mask a region that is being reprogrammed, and any protection holes must be avoided during this reconfiguration.

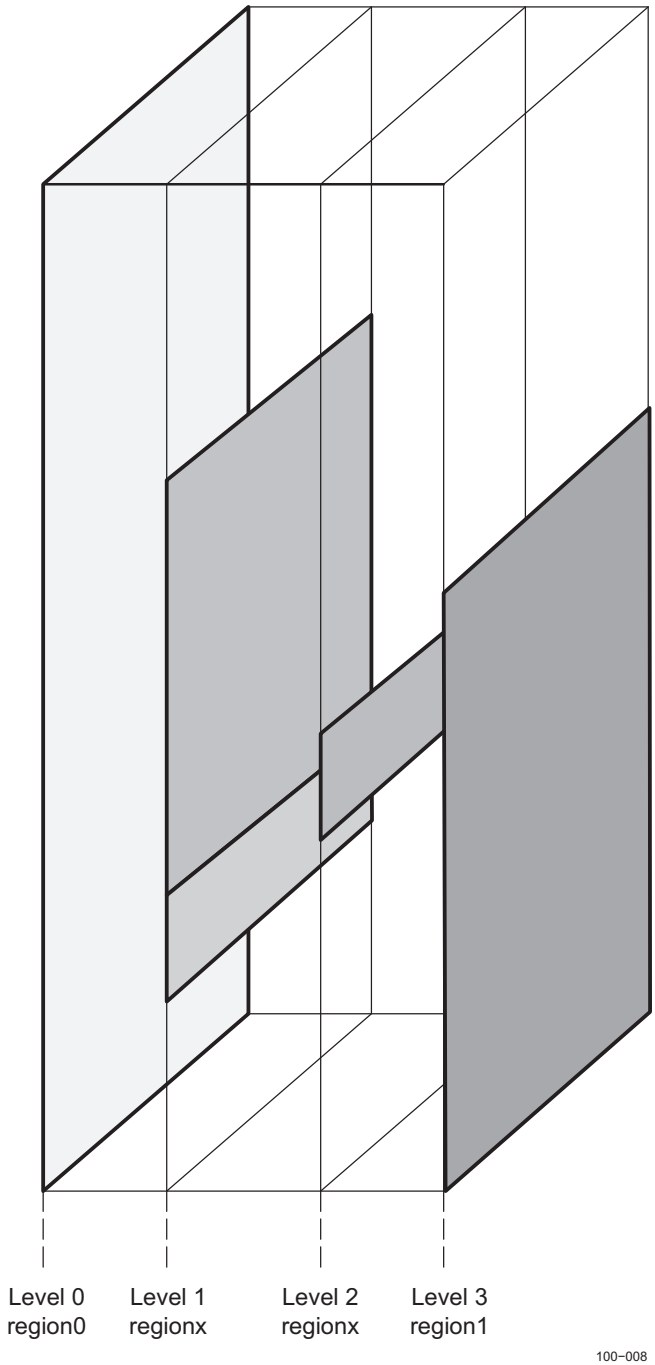
To change the protection settings of a region, follow this procedure:

1. Ensure that a free region is available to be used as a high-priority region.

2. Program this region as a high-priority region so its parameters match the region to be changed. The final programming must be the L3_PM_ADDR_MATCH_k register, which includes the priority attribute and the size parameter to enable the region.
3. Disable the region to be configured or reconfigure it by setting the SIZE bit field to 0.
4. Set up all region control registers with the new configuration. The final programming must be the L3_PM_ADDR_MATCH_k register, which includes the size parameter to enable the region.
5. Disable the high-priority region by setting its size to 0.

This procedure must be used each time there is an overlap between the originally defined region and the newly defined region. The use of a high-priority region is required to keep protection active during programming.

Figure 5-5. L3 Region Overlay and Priority Level Overview



5.2.3.3.3 Read and Write Permission

Read permission and write permission are configured using two registers:

- L3_PM_READ_PERMISSION_i
- L3_PM_WRITE_PERMISSION_i

The L3_PM_READ_PERMISSION_i and L3_PM_WRITE_PERMISSION_i registers allow the setting of read and write permission to one or more initiators. To grant read or write access, set the bit associated with the initiator to 1.

5.2.3.3.4 REQ_INFO_PERMISSION Configuration

The firewall comparison mechanism enables access to a protected target only when a correct combination of four MReqInfo in-band parameters is transmitted.

MReqInfo is a combination of a fixed 3-bit pattern that corresponds to a combination of the parameters MReqDebug, MReqType, and MReqSupervisor.

Different valid MReqInfo combinations can be defined for each L3 firewall region based on the L3_PM_REQ_INFO_PERMISSION_i value, which is programmed by software.

For each region, L3_PM_REQ_INFO_PERMISSION_i lists the possible MReqInfo combinations. Setting a Reqbit in this register determines the type of access authorised to the initiator.

Table 5-22 lists the MReqInfo combinations available and the Reqbit associated with it.

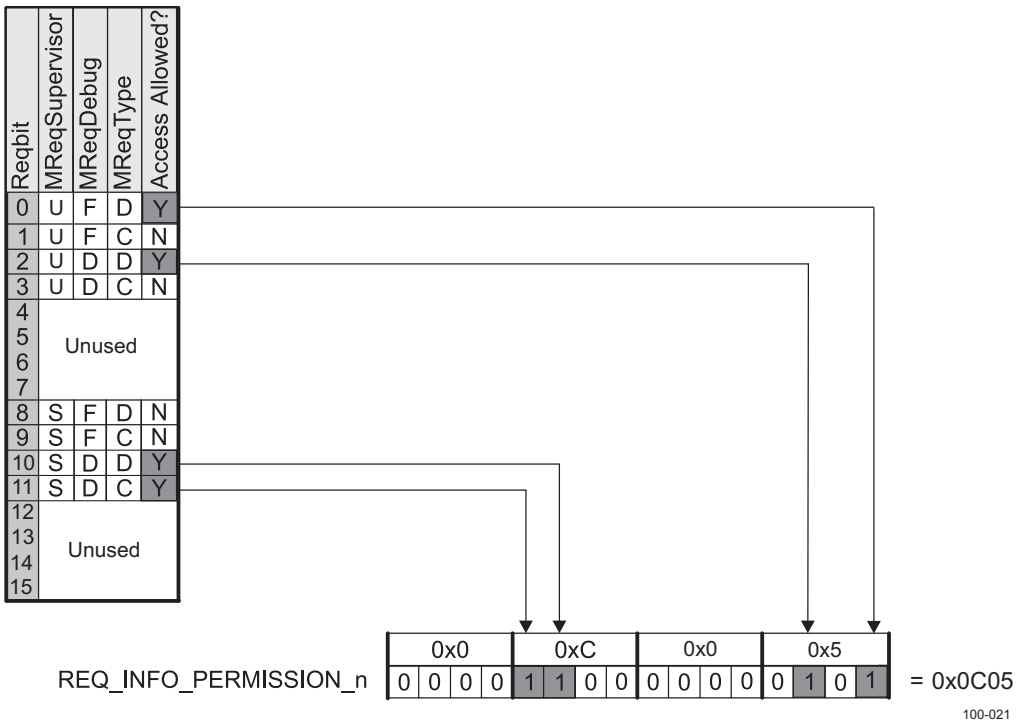
Table 5-22. MReqInfo Parameter Combinations

Reqbit	MReqInfo		
	MReqSupervisor	MReqDebug	MReqType
0	User	Functional	Data
1	User	Functional	Code
2	User	Debug	Data
3	User	Debug	Code
4		Reserved for non-GP devices	
5		Reserved for non-GP devices	
6		Reserved for non-GP devices	
7		Reserved for non-GP devices	
8	Supervisor	Functional	Data
9	Supervisor	Functional	Code
10	Supervisor	Debug	Data
11	Supervisor	Debug	Code
12		Reserved for non-GP devices	
13		Reserved for non-GP devices	
14		Reserved for non-GP devices	
15		Reserved for non-GP devices	

Figure 5-6 shows an example of the L3_PM_REQ_INFO_PERMISSION_i setting. In this example, L3_PM_REQ_INFO_PERMISSION_i is set to 0x0C05 (16'b0000_1100_0001_0101) for a specific region, which will grant access only if the request:

- User + Functional + Data
- User + Debug + Data
- Supervisor + Debug + Data
- Supervisor + Debug + Code

Figure 5-6. Example of REQ_INFO_PERMISSION Register



As another example, to configure a target accessible only for data and in a user mode, Reqbit 0, 1, 2 and 3 must be set. Therefore, the L3_PM_REQ_INFO_PERMISSION_i register must be set to 0x000F.

5.2.3.3.5 L3 Firewall Registers Overview

Table 5-23 lists the L3 firewall permission-setting registers. L3_PM_ADDR_MATCH_k, which is shown in this table, is not an accessible register.

Table 5-23. L3 Firewall Permission-Setting Registers

Register Name	Register Field Name	Field Modifiability	Parameter Comments	Region Comments
Region 0				This region is the default region. The default setting can be changed only with correct access according to L3 RT register.
L3_PM_ADDR_MATCH_k (k=0)	ADDR_SPACE[2:0]	Hard coded	Corresponds to all the target memory space	
	SIZE[7:3]	Hard coded	Corresponds to all the target memory space	
	Reserved		Default region: Level 0	
	BASE_ADDR[63:10]	Hard coded	Target-dependent	
L3_PM_REQ_INFO_PERMISSION_ i (i=0)	REQ_INFO[15:0]	Yes	Type of access permitted. See Table 5-22.	
L3_PM_READ_PERMISSION_i (i=0)	READ_PERMISSION[15:0]	Yes	Initiator read permission, depending on connections. See Table 5-14.	
L3_PM_WRITE_PERMISSION_i (i=0)	WRITE_PERMISSION[15:0]	Yes	Initiator write permission, depending on connections. See Table 5-14.	

Table 5-23. L3 Firewall Permission-Setting Registers (continued)

Register Name	Register Field Name	Field Modifiability	Parameter Comments	Region Comments
Region 1-7				The default settings can be changed only with correct access based on the L3 RT register.
L3_PM_ADDR_MATCH_k	ADDR_SPACE[2:0]	Yes		
	SIZE [7:3]	Yes	The regions are power-of-two in size and size-aligned with Base_Addr reference. When Size = 0x0, the firewall is deactivated.	
	LEVEL[9]	Yes	Protection region level 0x0: Level 1 0x1: Level 2 Region 1 is always Level 3.	
	BASE_ADDR[63:10]	Yes	Target-dependent	
L3_PM_REQ_INFO_PERMISSION_i	REQ_INFO[15:0]	Yes	Type of access permitted. See Table 5-22.	
L3_PM_READ_PERMISSION_i	READ_PERMISSION[15:0]	Yes	Initiators read permission ,depending on connections. See Table 5-14.	
L3_PM_WRITE_PERMISSION_i	WRITE_PERMISSION[15:0]	Yes	Initiators write permission, depending on connections. See Table 5-14.	

5.2.3.3.6 L3 Firewall Error-Logging Registers

Table 5-24 lists the L3 firewall error-logging registers.

Table 5-24. L3 Firewall Error Logging Registers

Register Name	Register Field Name	Field Modifiability	Parameter Comments
L3_PM_ERROR_LOG	CMD[2:0]	Read only	Log the OCP command of the request that caused a protection violation. See Table 5-1.
	REGION[6:4]	Read only	Log the region number targeted by the request that caused the protection violation.
	INITIATOR_ID[15:8]	Read only	Log the InitiatorID request that caused the protection violation. See Table 5-19
	REQ_INFO[20:16]	Read only	Log the MReqInfo bits of the request that caused the protection violation. See Table 5-22.
	CODE[27:24]	Read/write	Log the error that occurred. See Table 5-26.
	MULT[31]	Read/write	If a second error is detected before the first is cleared, the MULT bit is set. Once set by hardware, the CODE and MULT bits can be cleared only by software or a full hardware reset. Software clears the CODE and MULT bits by writing a non-zero value to the CODE field and writing 1 to the MULT bit.

5.2.3.3.7 L3 Firewall and System Control Module

When a protection violation occurs, an interrupt is sent to the MPU and IVA2.2 interrupt controller (if enabled). An in-band error is sent back, and an out-band error is logged in the CONTROL.CONTROL_PROT_ERR_STATUS register. Two logging registers are used, depending on the functional mode:

- In application mode:
  - CONTROL.CONTROL_PROT_ERR_STATUS [00]: OCM-ROM protection violation

  - CONTROL.CONTROL_PROT_ERR_STATUS [01]: OCM-RAM protection violation
  - CONTROL.CONTROL_PROT_ERR_STATUS [02]: GPMC protection violation
  - CONTROL.CONTROL_PROT_ERR_STATUS [04]: SMS protection violation
  - CONTROL.CONTROL_PROT_ERR_STATUS [05]: MAD2D protection violation
  - CONTROL.CONTROL_PROT_ERR_STATUS [06]: IVA2.2 protection violation
  - CONTROL.CONTROL_PROT_ERR_STATUS [07]: L4-Core protection violation
  - CONTROL.CONTROL_PROT_ERR_STATUS [12]: L3 RT protection violation
  - CONTROL.CONTROL_PROT_ERR_STATUS [15]: SAD2D protection violation
  - CONTROL.CONTROL_PROT_ERR_STATUS [16]: L4-Per protection violation
  - CONTROL.CONTROL_PROT_ERR_STATUS [17]: L4-Emu protection violation
- In debug mode:
  - CONTROL.CONTROL_PROT_ERR_STATUS_DEBUG [00]: OCM-ROM protection violation
  - CONTROL.CONTROL_PROT_ERR_STATUS_DEBUG [01]: OCM-RAM protection violation
  - CONTROL.CONTROL_PROT_ERR_STATUS_DEBUG [02]: GPMC protection violation
  - CONTROL.CONTROL_PROT_ERR_STATUS_DEBUG [03]: SMS protection violation
  - CONTROL.CONTROL_PROT_ERR_STATUS_DEBUG [05]: MAD2D protection violation
  - CONTROL.CONTROL_PROT_ERR_STATUS_DEBUG [06]: IVA2.2 protection violation
  - CONTROL.CONTROL_PROT_ERR_STATUS_DEBUG [12]: L3 RT protection violation
  - CONTROL.CONTROL_PROT_ERR_STATUS_DEBUG[16]: L4-Per protection violation
  - CONTROL.CONTROL_PROT_ERR_STATUS_DEBUG[17]: L4-Emu protection violation

When a violation occurs, these bits are cleared when the L3 or L4 firewall embedded error log registers are cleared.

5.2.3.4 Error Handling

5.2.3.4.1 Error Detection and Logging

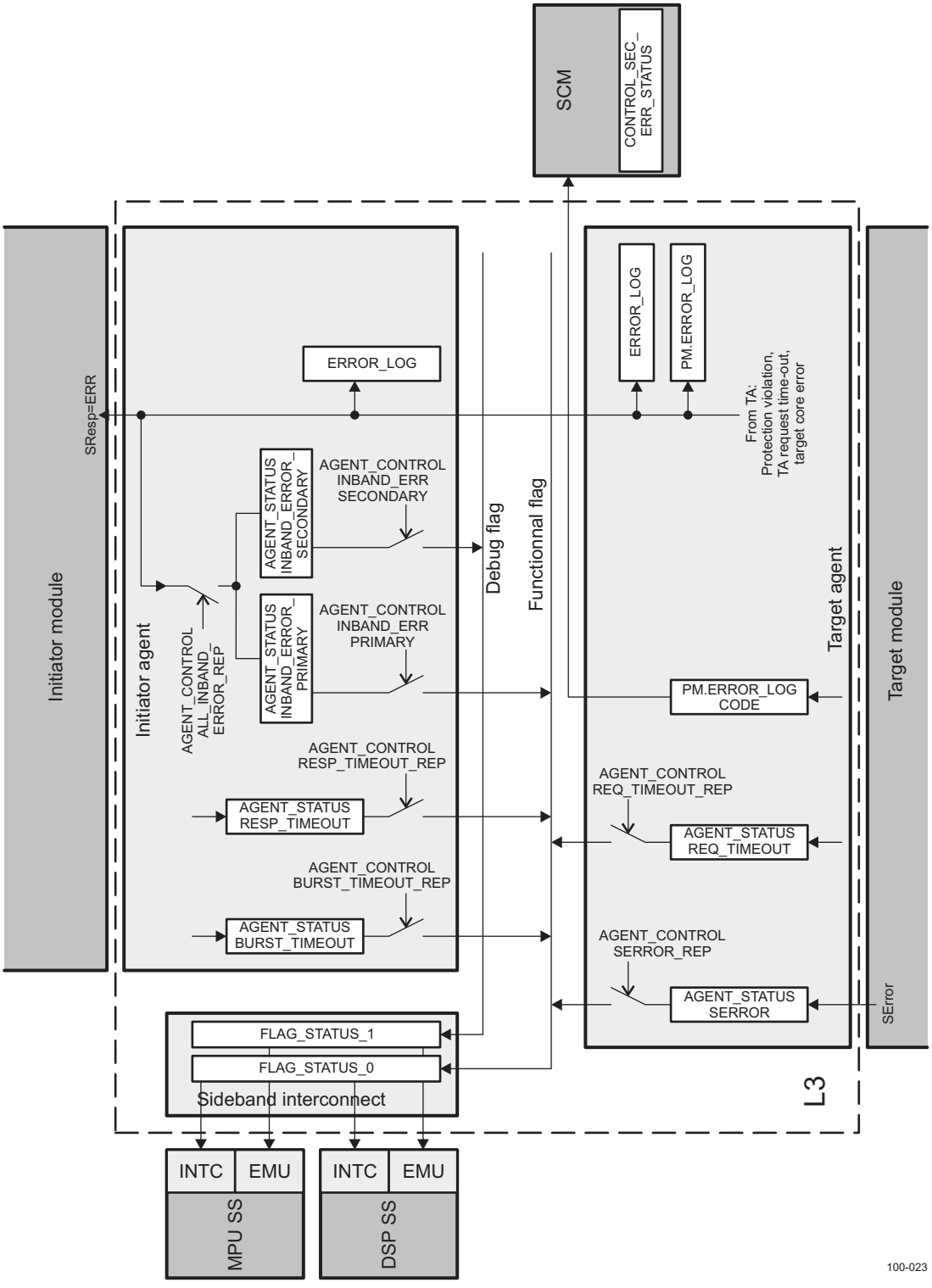
The L3 interconnect provides mechanisms for the detection, logging, and distribution of module-detected and internal interconnect errors. Hardware support is provide to assist in logging errors and cleaning up the state to allow error recovery software to run.

Two types of errors are identified by the L3 interconnect:

- Errors detected by modules and passed along by the interconnect:
  - SResp error: Using the SResp in-band qualifier from the target, the initiator is informed that its request was unsuccessful (see Table 5-3). This error is nonspecific and further analysis is required to find its cause.
  - SError: This out-of-band qualifier reports that the target is denied service due to an internal cause. No further analysis can be done in the L3 itself.
- Errors detected by the L3 interconnect:
  - Unsupported command: This error repports that the initiator sent a command that cannot be processed, because the target cannot accept it and no conversion to another command is possible. This error is detected only once per burst.
  - Address hole: This error reports an unknown address for a request. The address map is local to each IA; therefore, an address hole error is reported each time an initiator requests an access to a target it is not logically connected to, even if this address exists in the global L3 address map. This error is detected only once per burst.
  - Protection violation: This error indicates a request was rejected by a firewall.
  - Requests time-out: This error reports that the module did not end or respond to the request presented by the TA in the correct time interval. If the time interval for any request exceeds the time-out period, a time-out occurs. the target stops servicing requests.
  - Response time-out: This error reports that the module did not end the response presented by the IA in the correct time interval. If the time interval for any response exceeds the time-out period, a time-out occurs. The initiator stops servicing responses.
  - Burst open time-out: This error reports that a burst or ReadEx/Write pair is open and no command is presented to the module to end it. If the time interval for any command exceeds the time-out period, a time-out occurs. The initiator does not complete a burst or ReadEx/Write pair.

Figure 5-7 shows a global view of the register link to the error reporting structure in an initiator and the target agents.

Figure 5-7. L3 Error Reporting Structure



Most errors are reported to the IA that originated the request, except for the following:

- Initiator time-outs are reported only out-of-band. Any time-out in the IA results in flagging it as unavailable. A software reset of the agent is required to accept any new requests from the attached core.

- Posted write request, because the IA immediately generates a valid response to the initiator core. This error is only in out-of-band.

Table 5-25 lists where the errors are detected and logged.

Table 5-25. Error Types

Error	Detection	Logging	In-Band Report	Out-of-Band Report
SResp error	None needed	At IA	Yes(1)	Yes
SError assertion from target	None needed	At TA	No	Yes
Unsupported command	At IA	At IA	Yes	Yes
Address hole	At IA	At IA	Yes	Yes
Protection violation	At TA	At protection mechanism agent	Yes(1)	Yes
Request time-out	At TA	At TA	No	Yes
Response time out	At IA	At IA	No	Yes
Burst time-out	At IA	At IA	No	Yes

(1) In case of a posted write, errors cannot be reported in-band.

The time-out errors (request/response and burst) are persistent and require the software to reset both the module and the agent. No request can be processed in the agent until the reset is performed.

Other errors affect only the current request. Subsequent requests are treated normally. Errors are logged, however, and the information about the error used for debugging is kept until the software acknowledges the error.

Table 5-26 lists the information logged for each error code. Information logged in the two error-logging registers (L3_IA_ERROR_LOG, L3_IA_ERROR_LOG_ADDR, L3_TA_ERROR_LOG and L3_TA_ERROR_LOG_ADDR for each IA and TA) depends on the error itself. The CODE field ERROR_LOG[27:24] identifies the type of error occurring for the IA and TA.

Table 5-26. CODE Field Definition

		Type of Agent			Information Logged				
CODE[3:0]	Error Type	IA	TA	PM	REQ_INFO	Secondary	InitiatorID	CMD	Address
0	No error	x	x						
1	Unsupported command	x			x	x	x	x	x
2	Address hole	x			x	x	x	x	x
3	Protection violation			x	x		x	x	
4	In-band error	x				x	x		
5	Not used								
6	Not used								
7	Request time-out not accepted		x		x		x	x	x
8	Request time-out, no response		x				x		
9-15	Not used								

5.2.3.4.2 Time-Out

A time-out mechanism can be enabled in the target agent and initiator agent register. When the mechanism is enabled for a target agent or initiator agent and commands are not accepted or responses are not returned within the expected delay, the L3 interconnect generates an error event.

The error is logged in the target agent REQ_TIMEOUT bit L3_TA_AGENT_STATUS[8] for a target agent time-out, the BURST_TIMEOUT bit L3_IA_AGENT_STATUS[16] bit for an initiator burst time-out, and the RESP_TIMEOUT bit L3_IA_AGENT_STATUS[8] for an initiator response time-out. The affected agent enters an error state that causes it to send error responses to any new request. To recover from this state, the target agent must be reset by system software.

The time-out is counted starting from the moment a command is presented to the target, whatever the target response to this command is. The L3 interconnect implements a centralized time-base circuit that broadcasts a set of four periodic pulse signals to all connected target agents. These four signals are referred to as 1x time-base, 4x time-base, 16x time-base, and 64x time-base.

The time-base circuit offers four possible sets of four time-base signals selected by programming the TIMEOUT_BASE field L3_RT_NETWORK_CONTROL[10:8]. Table 5-27 lists all of the values in the number of L3 clock cycles.

Each target agent can be programmed to refer to one of the four time-base signals. A time-out condition is detected when either the command acceptance or the response is not received after a delay of between one and three time-base periods. After the time-out is detected and logged, the behavior of the attached module is ignored. A new request to the module arriving at the timed-out target agent receives an error response. If the request is addressed to the agent internal registers, it is processed normally.

To recover from a time-out error, the software is assumed to reset first the faulty module using its internal soft-reset bit, and then the agent using software reset.

Table 5-27. L3 Timeout Register Target and Agent Programming

REQ_TIMEOUT[2:0], BURST_TIMEOUT[2:0], and RESP_TIMEOUT[2:0]					
TIMEOUT_BASE[2:0]	**0**	**1**	**2**	**3**	**4**
0	All L3 time-out features are disabled.				
1	Locally disabled	64	256	1024	4096
2		256	1024	4096	16384
3		1024	4096	16384	65536
4		4096	16384	65536	262144

5.2.3.4.3 Error Steering

The error reporting structure consist of errors logged individually in the initiator or TAs. Some errors can be enabled and reported out-of-band by setting the following bits to 1:

- L3_IA_AGENT_CONTROL.BURST_TIMEOUT_REP bit for burst time-out
- L3_IA_AGENT_CONTROL.RESP_TIMEOUT_REP bit for response time-out
- L3_TA_AGENT_CONTROL.REQ_TIMEOUT_REP bit for request time-out
- L3_TA_AGENT_CONTROL.SERROR_REP bit for request time-out

Setting the L3_IA_AGENT_CONTROL.ALL_INBAND_ERROR_REP bit causes all in-band errors returned to the IA to be reported out-of-band.

IAs connected to processors capable of generating the debug-flagged requests (the MPU and IVA2.2 subsystems) use error steering. Any error linked to an application (nondebug) request is qualified as primary; any error linked to a debug request is qualified as secondary. Setting the IA.INBAND_ERROR_PRIMARY_REP or INBAND_ERROR_SECONDARY_REP bit to 1 allows the reporting of in-band to out-of-band primary and secondary errors.

The level of the current error is indicated in SECONDARY bit L3_IA_ERROR_LOG[30]. If an error occurs while another error is pending, the following occurs:

- If the pending error is primary, the new error is discarded at the IA level. MULTI bit L3_IA_ERROR_LOG[31] is set in the initiator error log register to indicate that another error has been detected; no further information can be stored.
- If both the pending error and the new error are secondary, the latest error is discarded and MULTI bit L3_IA_ERROR_LOG[31] is set.

- If the pending error is secondary and the incoming error is primary, MULTI bit L3_IA_ERROR_LOG[31] is set and all useful information about the new error (MCmd, MAddr, MReqInfo, etc.) is stored. All information relative to the secondary error is discarded.

For protection violation, errors are detected at the TA and steered to the control module (see Section 5.2.3.3.7, *L3 Firewall and System Control Module*).

5.2.3.4.4 Global Error Reporting

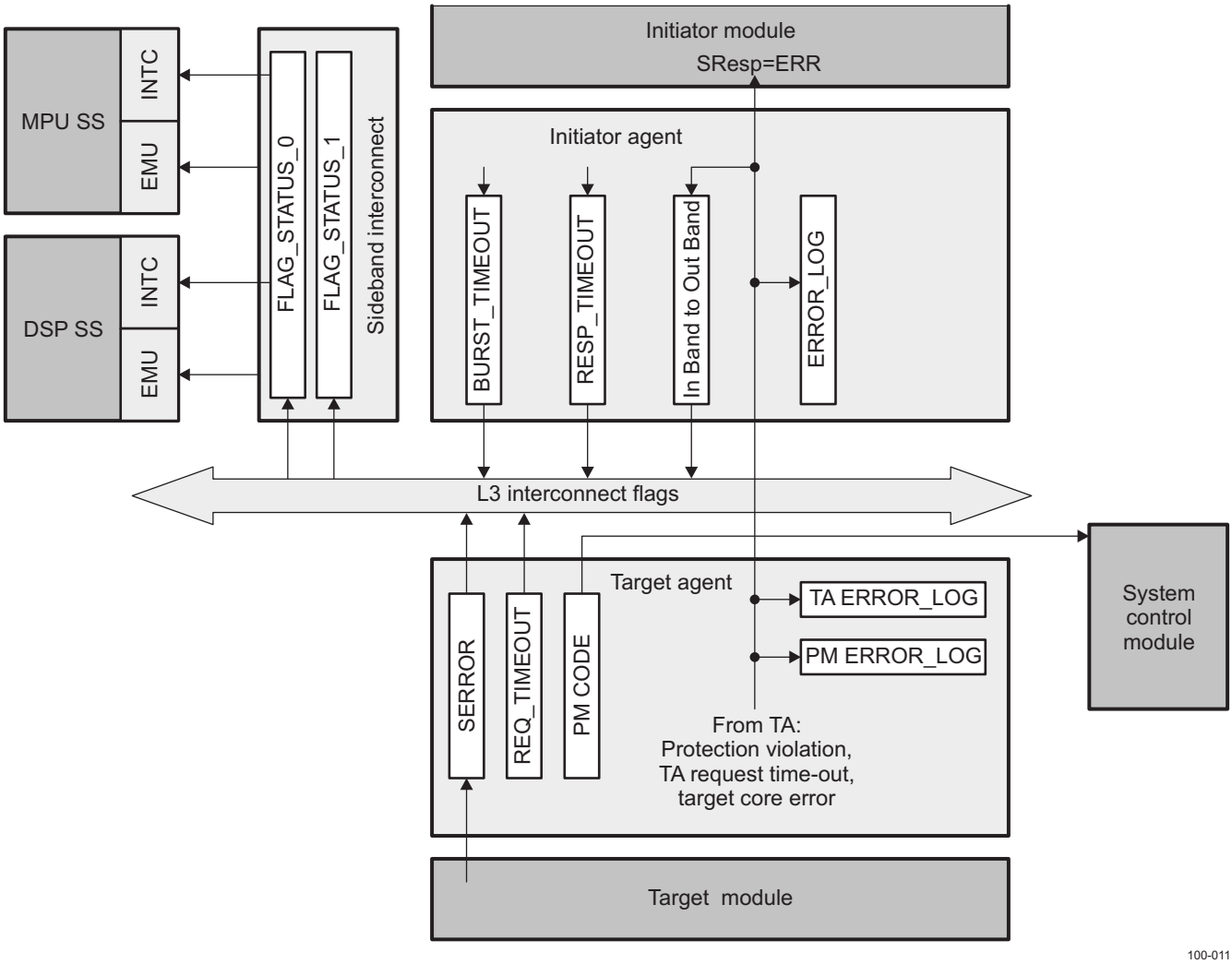
An out-of-band error refers to any flag output from the L3. It is not a formal error in the sense that it is not processed directly by the interconnect module; however, it is interpreted as such by one or several processors, typically by mapping it to an interrupt controller. All L3 out-of-band errors are ORed together and the result is transmitted to one or several interrupt controllers. All out-of-band errors are active high and must be acknowledged through a register access to be deasserted.

Out-of-band errors are reported through error or flag signals, asynchronously to any other data flow, but synchronously to the clock.

All errors are reported out-of-band to the IVA2.2 and MPU subsystems simultaneously. The processors must check whether an error is relevant to the subsystems. These errors are routed not only to the IVA2.2 and MPU subsystem interrupt controllers, but also to their emulation logic (see Figure 5-8). Two composite flags help in asserting the error origin and criticality:

- The L3 application error flag reports application or nonattributable (not related to a request such as a time-out, SError) errors.
- The L3 debug error flag reports debug errors.

Figure 5-8. Global Error Routing



In addition to the SResp qualifier error reporting, some external targets use an SError signal to indicate that an internal error has occurred. Some resetting action may be required before any new request can be accepted, depending on the module and on the error itself. These signals are routed as flags internally to the L3. Table 5-28 lists the possible error reported through an SError, propagated externally to the L3 interconnect, and aggregated to the L3 application error flag.

Table 5-28. L3 External Input Flags

Target	Flag	Description	Software Visible?
GPMC	SError	General-purpose error occurs in GPMC module	Yes, in FLAG_STATUS register and in L3 TA SERROR bit AGENT_STATUS[24]

Note: The user must ensure that the corresponding IRQ lines are set correctly in the interrupt controllers of the MPU and IVA2.2 subsystems.

Figure 5-9 shows the routing of errors to the application and debug error composite flags. Because both debug capable processors, the MPU and IVA2.2 subsystems have access to the full error report mechanism; they can be debugged independently of one another.

Figure 5-9. L3 Error Routing

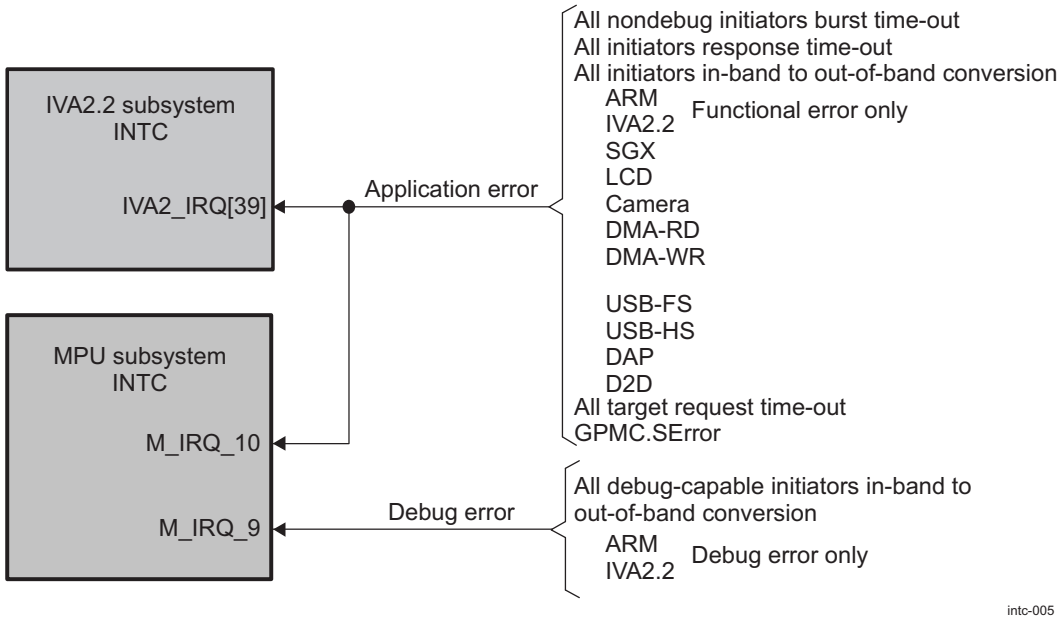


Table 5-29 lists the bit and source for the STATUS bit field L3_SI_FLAG_STATUS_0[63:0] for an application error, and Table 5-30 lists the bit and source for the STATUS bit field L3_SI_FLAG_STATUS_1[63:0] for an application error for a debug error.

Protection errors are reported in-band to the IA when possible. This may not be possible for posted writes, however. These errors are also reported asynchronously to the module.

Additionally, all protection errors (even posted writes) are seen by the in-band-to-out-of-band conversion logic in the IAs. Therefore, all protection errors feed into the L3 application error and L3 debug error composite flags, and hence back to the MPU and IVA2.2 subsystems.

The SMS, L4-Core, L-4 Per, and L4-Emu interconnects have internal firewalls that use the same reporting scheme. These errors are not routed directly to the L3 interconnect but are reported in-band to the initiator, which routes them to the composite flags. The SMS, L4-Core, L4-Per, and L4-Emu interconnects always provide an error on the in-band SResp qualifier for protection violations.

Table 5-29. L3_SI_FLAG_STATUS_0 for Application Error

Flag Bit Number	Source		Flag Bit Number	Source	
	Agent	Error		Agent	Error
0	MPU IA	Burst time-out	32	Reserved	
1	MPU IA	Response time-out	33	SAD2D IA	Burst time-out
2	MPU IA	Functional Inband error	34	SAD2D IA	Response time-out
3	Reserved		35	SAD2D IA	SError
4			36	Reserved	
5			37		
6	IVA2.2 IA	Burst time-out	38		
7	IVA2.2 IA	Response time-out	39		
8	IVA2.2 IA	Functional Inband error	40		
9	SGX IA	Burst time-out	41		
10	SGX IA	Functional Inband error	42		
11	Reserved		43	Reserved	

Table 5-29. L3_SI_FLAG_STATUS_0 for Application Error (continued)

Flag Bit Number	Source		Flag Bit Number	Source	
	Agent	Error		Agent	Error
12	CAMERA IA	Burst time-out	44		
13	CAMERA IA	Response time-out	45		
14	CAMERA IA	Functional Inband error	46		
15	Display SS IA	Burst time-out	47		
16	Display SS IA	Functional Inband error	48	SMS TA	Request time-out
17	Reserved		49	GPMC TA	Request time-out
18	sDMA Rd IA	Burst time-out	50	OCM RAM TA	Request time-out
19	sDMA Rd IA	Functional Inband error	51	OCM ROM TA	Request time-out
20	Reserved	Reserved	52	L4-Core TA	Request time-out
21	sDMA Wr IA	Burst time-out	53	L4-Per TA	Request time-out
22	sDMA Wr IA	Functional Inband error	54	IVA2.2 TA	Request time-out
23	Reserved		55	SGX TA	Request time-out
24	HS USB OTG IA	Burst time-out	56	L4-Emu TA	Request time-out
25	HS USB OTG IA	Response time-out	57	GPMC TA	SError assertion
26	HS USB OTG IA	Functional Inband error	58	L4-Core TA	Request time-out
27	HS USB Host IA	Burst time-out	59	L4-Per TA	Request time-out
28	HS USB Host IA	Functional Inband error	60	L4-Emu	Request time-out
29	Reserved		61	MAD2D TA	Request time-out
30			62	Reserved	
31			63		

Table 5-30. L3_SI_FLAG_STATUS_1 for Debug Error

Flag Bit Number	Source		Flag Bit Number	Source	
	Agent	Error		Agent	Error
0	MPU DATA IA	Debug error	32		
1			33		
2			34		
3	DAP IA	Debug error	35		
4	DAP IA	Debug error	36		
5			37		
6	IVA2.2 IA	Debug error	38		
7			39		
8			40		
9			41		
10			42		
11			43		
12			44		
13			45		
14			46		
15			47		
16			48		

Table 5-30. L3_SI_FLAG_STATUS_1 for Debug Error (continued)

Flag Bit Number	Source		Flag Bit Number	Source	
	Agent	Error		Agent	Error
17	Reserved		49	Reserved	
18			50		
19			51		
20			52		
21			53		
22			54		
23			55		
24			56		
25			57		
26			58		
27			59		
28			60		
29			61		
30			62		
31			63		

5.2.4 L3 Interconnect Basic Programming Model

5.2.4.1 General Recommendation

The L3 interconnect registers must be read or written with little-endian attributes; otherwise, the result is undefined.

CAUTION

Overlapping between protection regions with the same priority level leads to unpredictable behavior and must be avoided.

5.2.4.2 Initialization

At the release of power on reset, the L3 firewall default configuration enables all accesses to target modules, except for a section of the OCM ROM.

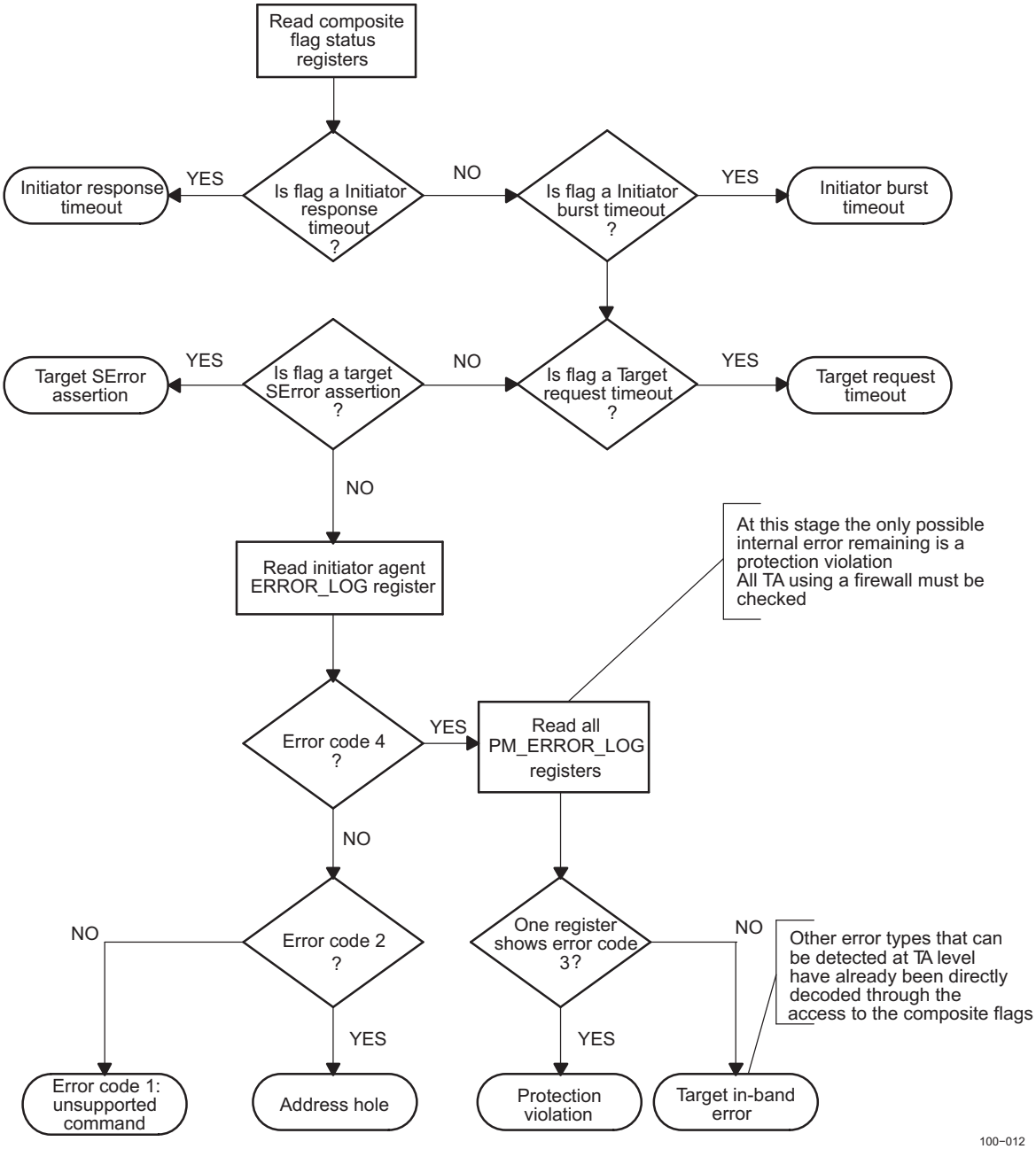
Generally, software must configure the firewall properly to avoid poor use of the hardware resources.

L3 time-out capabilities are also disabled at reset.

5.2.4.3 Error Analysis

The information required to analyze an error source is logged in several registers (see Table 5-26). The number of registers to access depends on the error source. When investigating the origin of an error, software reads a set of error log registers. At each stage, the register either states the current error or points to the next agent in which the error is logged. Figure 5-10 shows the software sequence required in most cases.

Figure 5-10. Typical Error Analysis Sequence



Errors that do not result from an in-band to out-of-band conversion can be extracted immediately from the application or debug error flag by reading the STATUS field L3_SI_FLAG_STATUS_0[63:0] and L3_SI_FLAG_STATUS_1[63:0] register; therefore, they do not require the whole analysis sequence shown in Table 5-26.

When analysis leads to a TA error, software must read the initiator agent ADDR field L3_IA_ERROR_LOG_ADDR[39:0] to extract the TA address causing the error.

5.2.4.3.1 Time-Out Handling

This section gives information about all modules and features in the high-tier device. See Section 1.5, *OMAP34xx Family*, to check availability of modules and features. To flag interconnect response being blocked, the time-out of target agents attached to unavailable modules can be enabled with the lowest setting.

Example 1

In this example, the MPU interrupt handler detects an error from the L3 interconnect. A read access from the L3_SI_FLAG_STATUS_0 register reports a value of 0x40000. As described in Table 5-29, the error detected is a burst time-out from the sDMA read port.

As with any time-out error, the affected module is now considered to be out-of-service. If necessary, the error can be cleared by sending a soft reset command to the agent (set CORE_RESET bit L3_IA_AGENT_CONTROL[0] to 1, and then to 0). During this time the initiator is off-line. Although the rest of the system still behaves normally, a time-out error is usually severe enough to require a complete reset of the chip.

Before resetting the agent or the system, the error log registers can learn the status of the interconnect and determine the type of failure. Reading the IA_SDMA_RD.L3_IA_AGENT_STATUS register shows whether the time-out was detected during a burst or during a read/write sequence.

Example 2

In this example, the MPU interrupt controller detects an error from the L3 interconnect. A read access from the L3_SI_FLAG_STATUS_0 register reports a value of 0x100. This is a functional error from the IVA2.2 subsystem initiator.

The IA_IVA2.2.L3_IA_ERROR_LOG register should be read next. The value stored in it is 0x12_0400_1302.

- CMD field IA_IVA2.2.L3_IA_ERROR_LOG[2:0]: Value 0x2. Not applicable for an in-band error.
- INITID field IA_IVA2.2.L3_IA_ERROR_LOG[15:8]: Value 0x13. The origin of the error is the IVA2.2 subsystem sDMA.
- CODE field IA_IVA2.2.L3_IA_ERROR_LOG[27:24]: Value 0x4. Error is an in-band error.
- SECONDARY bit IA_IVA2.2.L3_IA_ERROR_LOG[30]: Value 0x0. The error is functional.
- MULTI bit IA_IVA2.2.L3_IA_ERROR_LOG[31]: Value 0x0. No additional error has been detected.
- REQ_INFO field IA_IVA2.2.L3_IA_ERROR_LOG[43:32]: Value 0x12. Not applicable for an in-band error.

There is no simple way to determine which target originated the in-band error. An SError assertion or a request time-out would have been detected and logged in the L3_SI_FLAG_STATUS_0 register. Because no such error is asserted (bits 60:48 are still 0), the error can only be a firewall error or have originated in the target itself. The user must read all of the PM_xxx.L3_PM_ERROR_LOG registers.

CAUTION

The PM register blocks are sensitive registers and are usually protected. Ensure that the processor used to debug the error is allowed to access these registers. If not, the access will be rejected and another error will be generated.

All PM_xxx.L3_PM_ERROR_LOG registers are clear (bits 27:24 equal 0x0) except for PM_OCMRAM.L3_PM_ERROR_LOG, which reads 0x0302_1301. The error occurred while trying to access the OCM RAM.

Note: If all PM_xxx.L3_PM_ERROR_LOG registers are clear, the error is unrelated to the interconnect. Further analysis must be done in the targets.

- Bits 27:24: Value 0x3. There is a protection error.
- Bits 2:0: Value 0x1. The command was a posted write.

- Bits 6:4: Value 0x0. The address is protected by Region 0 (default region) of the firewall.
- Bits 15:8: Value 0x13. The origin of the error is the IVA2.2, thread 2. This is consistent with the error log on the initiator side, and confirms this error report is the correct one.
- Bits 20:16: Value 0x2. The protection parameters were data, debug, and user.
- Bit 31: Value 0x0. This is the only error detected.

A check on the protection configuration confirms that this access was not allowed, either because the initiator was not given write access or because the protection access parameters were not compatible with the PM_OCMRAM.REQINFO_PERMISSIONS_0 settings.

A protection violation is not terminal from the interconnect point of view. If the module responsible for the global chip protection do not reset the system (including the IVA2.2 subsystem and the OCM RAM), it continues to run normally. A simple clearing of the error on both the IA and PM sides is required to return to a clean status.

5.2.4.3.2 Acknowledging Errors

Time-out errors can never be acknowledged. To return to a normal operation after an error, the faulty agent must be reset . An agent can be reset by asserting CORE_RESET bit L3_IA_AGENT_CONTROL[0] or L3_TA_AGENT_CONTROL[0], or by resetting the L3 interconnect through the PRCM.

Functional errors, including an in-band signal reporting a protection error, must be inactived through software. Setting the INBAND_ERROR_PRIMARY and INBAND_ERROR_SECONDARY bits L3_IA_AGENT_STATUS[28-29] in an IA, or setting the SERROR bit L3_IA_AGENT_STATUS[24] in a TA clears the reported error. Table 5-31 lists the bit to clear and the associated type of error.

The L3_IA_ERROR_LOG or L3_TA_ERROR_LOG register must also be cleared by writing a nonzero value simultaneously to the CODE bit field and the values currently stored in the MULTI and SECONDARY fields.

Table 5-31. Error Clearing

Agent Type	Error	Register Field
Initiator	In-band primary (application) error	INBAND_ERROR_PRIMARY
	In-band secondary (debug) error	INBAND_ERROR_SECONDARY
Target	Target asserts SError	SERROR

The procedure to clear protection errors depends on the system protection configuration:

- Write a nonzero value simultaneously into CODE bit field L3_PM_ERROR_LOG[27:24] and the value currently stored in the MULTI bit L3_PM_ERROR_LOG[31] of the corresponding PM register block.
- Alternately, read either L3_PM_ERROR_CLEAR_SINGLE or L3_PM_ERROR_CLEAR_MULTI, depending on the current value of the MULTI field in the L3_PM_ERROR_LOG register. This solution, which allows the clearing of protection errors without having a write access on other protection registers, preserves protection.

L3_SI_FLAG_STATUS_0 or L3_SI_FLAG_STATUS_1 must be checked at the end of any error acknowledging sequence to confirm that the acknowledgement was successful and that no other error is pending.

5.2.4.4 Typical Example of Firewall Programming Example

All four regions in the IVA2.2 target firewall can be configured with no restrictions. However, protection is required for a 14K-byte region starting at address 0x0 in address space 2, and it must accept any access from any initiator for the rest of the target. The protection restricts allowed accesses as follows:

- Only the IVA2.2 and the MPU can read from this memory.
- Only the MPU can write to this memory.
- The protection parameters must include supervisor and functional.

Only accesses declared as supervisor and functional are allowed to pass through the protected region of the firewall. There is no restriction on other attributes.

Table 5-32 lists the possible parameter combinations, whether they are allowed to pass or not, and why they are rejected.

Table 5-32. MReqInfo Parameter Example

Reqbit	MReqInfo				
	MReqSupervisor	MReqDebug	MReqType	Access Allowed	Reason for Rejection
0	User	Functional	Data	No	Not supervisor
1	User	Functional	Code	No	Not supervisor
2	User	Debug	Data	No	Not supervisor
3	User	Debug	Code	No	Not supervisor
4	Reserved for non-GP devices			No	
5	Reserved for non-GP devices			No	
6	Reserved for non-GP devices			No	
7	Reserved for non-GP devices			No	
8	Supervisor	Functional	Data	Yes	
9	Supervisor	Functional	Code	Yes	
10	Supervisor	Debug	Data	No	Not functional
11	Supervisor	Debug	Code	No	Not functional
12	Reserved for non-GP devices			No	
13	Reserved for non-GP devices			No	
14	Reserved for non-GP devices			No	
15	Reserved for non-GP devices			No	

Only combinations with codes 8 and 9 are allowed to access a target. REQ_INFO field PM_IVA2.2.L3_PM_REQ_INFO_PERMISSION_i[15:0] for the protected region must be set to (0x8) | (0x9); that is, 0x0300.

Solution 1

Set Region 0 to accept all accesses:

- PM_IVA2.2.L3_PM_REQ_INFO_PERMISSION_i (i=0)= 0xFFFF
- PM_IVA2.2.L3_PM_READ_PERMISSION_i (i=0)= 0x140E
- PM_IVA2.2.L3_PM_WRITE_PERMISSION_i (i=0)= 0x140E

Set Region 1 to cover the first 8K bytes of the protected region:

- PM_IVA2.2.L3_PM_ADDR_MATCH_k (k=1) = 0x22 (start address 0x0, size 8K bytes, address space 2)
- PM_IVA2.2.L3_PM_REQ_INFO_PERMISSION_i (i=1) = 0x0300
- PM_IVA2.2.L3_PM_READ_PERMISSION_i (i=1) = 0x0406 (IVA2.2 DMA and MMU and MPU are allowed)
- PM_IVA2.2.L3_PM_WRITE_PERMISSION_i (i=1) = 0x0002 (only MPU is allowed to write)

Set Region 2 to cover the next 4K bytes:

- PM_IVA2.2.L3_PM_ADDR_MATCH_k (k=2) = 0x201A (start address 0x2000, level 1, size 4K bytes, address space 2)
- PM_IVA2.2.L3_PM_REQ_INFO_PERMISSION_i (i=2) = 0x0300
- PM_IVA2.2.L3_PM_READ_PERMISSION_i (i=2) = 0x0406
- PM_IVA2.2.L3_PM_WRITE_PERMISSION_i (i=2) = 0x0002

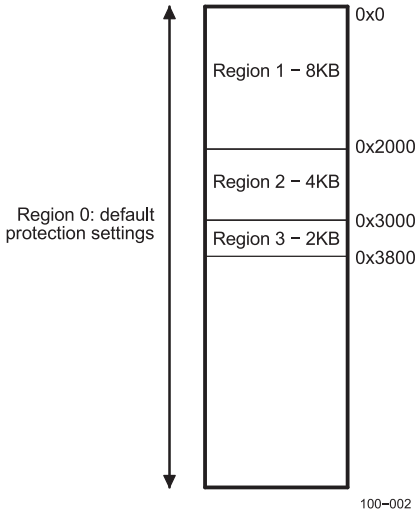
Set Region 3 to cover the last 2K bytes:

- PM_IVA2.2.L3_PM_ADDR_MATCH_k (k=3)= 0x3012 (start address 0x3000, level 1, size 2K bytes, address space 2)

- PM_IVA2.2.L3_PM_REQ_INFO_PERMISSION_i (i=3) = 0x0300
- PM_IVA2.2.L3_PM_READ_PERMISSION_i (i=3) = 0x0406
- PM_IVA2.2.L3_PM_WRITE_PERMISSION_i (i=3) = 0x0002

Figure 5-11 shows the firewall configuration for Solution 1.

Figure 5-11. Firewall Configuration Solution 1



Solution 2

Set Region 0 to accept all accesses:

- PM_IVA2.2.L3_PM_REQ_INFO_PERMISSION_i (i=0) = 0xFFFF
- PM_IVA2.2.L3_PM_READ_PERMISSION_i (i=0) = 0x140E
- PM_IVA2.2.L3_PM_WRITE_PERMISSION_i (i=0) = 0x140E

Set Region 2 to cover 16K bytes:

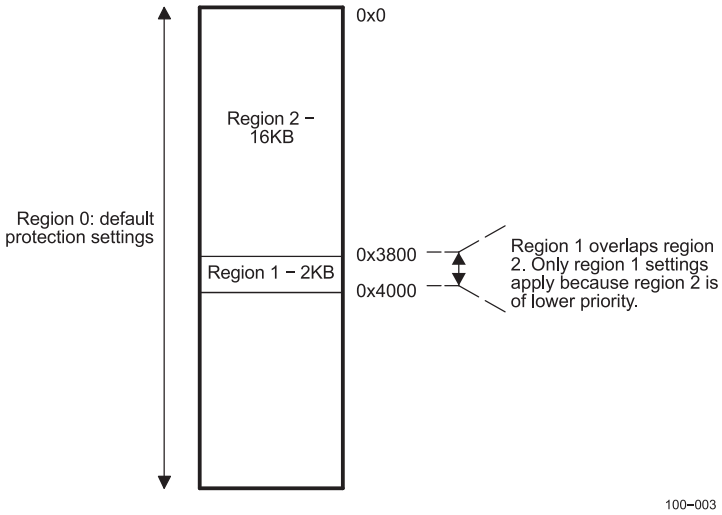
- PM_IVA2.2.L3_PM_ADDR_MATCH_k (k=2) = 0x2A (start address 0x0, level 1, size 16K bytes, address space 2)
- PM_IVA2.2.L3_PM_REQ_INFO_PERMISSION_i (i=2) = 0x0300
- PM_IVA2.2.L3_PM_READ_PERMISSION_i (i=2) = 0x0406 (IVA2.2 DMA and MMU and MPU are allowed)
- PM_IVA2.2.L3_PM_WRITE_PERMISSION_i (i=2) = 0x0002 (only MPU is allowed to write)

Set Region 1 to mask the last 2K bytes:

- PM_IVA2.2.L3_PM_ADDR_MATCH_k (k=2) = 0x3812 (start address 0x3800, size 2K bytes, address space 2)
- PM_IVA2.2.L3_PM_REQ_INFO_PERMISSION_i (i=1) = 0xFFFF
- PM_IVA2.2.L3_PM_READ_PERMISSION_i (i=1) = 0x042E
- PM_IVA2.2.L3_PM_WRITE_PERMISSION_i (i=1) = 0x042E

Figure 5-12 shows the firewall configuration for Solution 2.

Figure 5-12. Firewall Configuration Solution 2



5.2.5 L3 Interconnect Register Manual

Table 5-33 lists the base address and address space for all L3 register blocks.

Table 5-33. Instance Summary

Module Name	Base Address	Size
RT	0x6800 0000	1K byte
SI	0x6800 0400	1K byte
IA_MPUSS	0x6800 1400	1K byte
IA_IVA2.2	0x6800 1800	1K byte
IA_SGX	0x6800 1C00	1K byte
TA_SMS	0x6800 2000	1K byte
TA_GPMC	0x6800 2400	1K byte
TA_OCM_RAM	0x6800 2800	1K byte
TA_OCM_ROM	0x6800 2C00	1K byte
IA_SAD2D	0x6800 3000	1K byte
TA_MAD2D	0x6800 3400	1K byte
IA_USB_HS_Host	0x6800 4000	1K byte
IA_USB_HS_OTG	0x6800 4400	1K byte
IA_sDMA_RD	0x6800 4C00	1K byte
IA_sDMA_WR	0x6800 5000	1K byte
IA_DSS	0x6800 5400	1K byte
IA_CAM	0x6800 5800	1K byte
IA_DAP	0x6800 5C00	1K byte
TA_IVA2.2	0x6800 6000	1K byte
TA_SGX	0x6800 6400	1K byte
TA_L4_CORE	0x6800 6800	1K byte
TA_L4_PER	0x6800 6C00	1K byte
TA_L4_EMU	0x6800 7000	1K byte
PM_RT	0x6801 0000	1K byte
PM_GPMC	0x6801 2400	1K byte
PM_OCM_RAM	0x6801 2800	1K byte
PM_OCM_ROM	0x6801 2C00	1K byte
PM_IVA2.2	0x6801 4000	1K byte

5.2.5.1 L3 Initiator Agent (L3 IA)

This section describes the IA register block. Each IA in L3 interconnect has its own IA register block.

The following are the IA registers:

- MPU subsystem port
- IVA2.2 subsystem port
- SGX subsystem port
- High-Speed USB Host
- High-Speed USB OTG
- System DMA read port
- System DMA write port
- Display subsystem port
- Camera subsystem port
- DAP port
- Die-to-die

Table 5-34 through Table 5-37 list the IA registers and their physical addresses, depending on the module instance.

Table 5-34. Initiator Agent Common Register Summary

Register Name	Type	Register Width (Bits)	Address Offset	IA_MPUSS Physical Address	IA_IVA2.2 Physical Address	IA_SGX Physical Address
L3_IA_COMPONENT	R	64	0x000	0x6800 1400	0x6800 1800	0x6800 1C00
L3_IA_CORE	R	64	0x018	0x6800 1418	0x6800 1818	0x6800 1C18
L3_IA_AGENT_CONTROL	RW	64	0x020	0x6800 1420	0x6800 1820	0x6800 1C20
L3_IA_AGENT_STATUS	RW	64	0x028	0x6800 1428	0x6800 1828	0x6800 1C28
L3_IA_ERROR_LOG	RW	64	0x058	0x6800 1458	0x6800 1858	0x6800 1C58
L3_IA_ERROR_LOG_ADDR	R	64	0x060	0x6800 1460	0x6800 1860	0x6800 1C60

Table 5-35. Initiator Agent Common Register Summary

Register Name	Type	Register Width (Bits)	Address Offset	IA_USB_HS_Host Physical Address	IA_USB_HS_OTG Physical Address	IA_SAD2D Physical Address
L3_IA_COMPONENT	R	64	0x000	0x6800 4000	0x6800 4400	0x6800 3000
L3_IA_CORE	R	64	0x018	0x6800 4018	0x6800 4418	0x6800 3018
L3_IA_AGENT_CONTROL	RW	64	0x020	0x6800 4020	0x6800 4420	0x6800 3020
L3_IA_AGENT_STATUS	RW	64	0x028	0x6800 4028	0x6800 4428	0x6800 3028
L3_IA_ERROR_LOG	RW	64	0x058	0x6800 4058	0x6800 4458	0x6800 3058
L3_IA_ERROR_LOG_ADDR	R	64	0x060	0x6800 4060	0x6800 4460	0x6800 3060

Table 5-36. Initiator Agent Common Register Summary

Register Name	Type	Register Width (Bits)	Address Offset	IA_sDMA_RD Physica Address	IA_sDMA_WR Physical Address
L3_IA_COMPONENT	R	64	0x000	0x6800 4C00	0x6800 5000
L3_IA_CORE	R	64	0x018	0x6800 4C18	0x6800 5018
L3_IA_AGENT_CONTROL	RW	64	0x020	0x6800 4C20	0x6800 5020
L3_IA_AGENT_STATUS	RW	64	0x028	0x6800 4C28	0x6800 5028
L3_IA_ERROR_LOG	RW	64	0x058	0x6800 4C58	0x6800 5058
L3_IA_ERROR_LOG_ADDR	R	64	0x060	0x6800 4C60	0x6800 5060

Table 5-37. Initiator Agent Common Register Summary

Register Name	Type	Register Width (Bits)	Address Offset	IA_DSS Physical Address	IA_CAM Physical Address	IA_DAP Physical Address
L3_IA_COMPONENT	R	64	0x000	0x6800 5400	0x6800 5800	0x6800 5C00
L3_IA_CORE	R	64	0x018	0x6800 5418	0x6800 5818	0x6800 5C18
L3_IA_AGENT_CONTROL	RW	64	0x020	0x6800 5420	0x6800 5820	0x6800 5C20
L3_IA_AGENT_STATUS	RW	64	0x028	0x6800 5428	0x6800 5828	0x6800 5C28
L3_IA_ERROR_LOG	RW	64	0x058	0x6800 5458	0x6800 5858	0x6800 5C58
L3_IA_ERROR_LOG_ADDR	R	64	0x060	0x6800 5460	0x6800 5860	0x6800 5C60

5.2.5.1.1 L3 Initiator Agent (L3 IA) Registers Description

Table 5-38. L3_IA_COMPONENT

Address Offset	0x000
Physical Address	See Table 5-34 to Table 5-37
Description	Component register of IA
Type	RW

63	62	61	60	59	58	57	56	55	54	53	52	51	50	49	48	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32
Reserved																															

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CODE																REV															

Bits	Field Name	Description	Type	Reset
63:32	Reserved	Reserved	R	0x00000000
31:16	CODE	Component code	R	See (1).
15:0	REV	Revision of the component	R	See (1).

(1) TI internal data

Table 5-39. Register Call Summary for Register L3_IA_COMPONENT

L3 Interconnect

- L3 Initiator Agent (L3 IA) Registers Manual: [0] [1] [2] [3]

Table 5-40. L3_IA_CORE

Address Offset	0x018
Physical Address	See Table 5-34 to Table 5-37
Description	Core register of L3 IA block
Type	R

63	62	61	60	59	58	57	56	55	54	53	52	51	50	49	48	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32
Reserved																VENDOR_CODE															

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CORE_CODE																REV_CODE															

Bits	Field Name	Description	Type	Reset
63:48	Reserved	Reserved	R	0x0000
47:32	VENDOR_CODE	Vendor code	R	See (1).
31:16	CORE_CODE	Core code	R	See (1).
15:0	REV_CODE	Revision code	R	See (1).

(1) TI internal data

Table 5-41. Register Call Summary for Register L3_IA_CORE

L3 Interconnect

- L3 Initiator Agent (L3 IA) Registers Manual: [0] [1] [2] [3]

Table 5-42. L3_IA_AGENT_CONTROL

Address Offset	0x020
Physical Address	See Table 5-34 to Table 5-37
Description	Agent control register of IA block
Type	RW

63	62	61	60	59	58	57	56	55	54	53	52	51	50	49	48	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32
Reserved																															

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved		INBAND_ERROR_SECONDARY_REP	INBAND_ERROR_PRIMARY_REP	ALL_INBAND_ERROR_REP	BURST_TIMEOUT_REP	RESP_TIMEOUT_REP	Reserved							BURST_TIMEOUT		Reserved						RESP_TIMEOUT		Reserved		REJECT	Reserved			CORE_RESET	

Bits	Field Name	Description	Type	Reset
63:30	Reserved	Reserved	R	0x000000000
29	INBAND_ERROR_ SECONDARY_REP	Reporting of in-band errors indicating debug error. 0x0:No special reporting 0x1:Report error	RW	1
	Reserved for instances 3 to 12	Reserved	R	0x0
28	INBAND_ERROR_PRIMARY_REP	Reporting of in-band errors indicating application error. 0x0:No special reporting 0x1:Report error	RW	1
27	ALL_INBAND_ERROR_REP	Reporting of all in-band errors 0x0:Only report errors that cannnot be reported in-band 0x1:Report all in-band errors	RW	1
26	BURST_TIMEOUT_REP	Open burst and ReadEx/Write timeout reporting 0x0:No special reporting 0x1:Report out of band	RW	1
25	RESP_TIMEOUT_REP	Response timeout reporting 0x0:No special reporting 0x1:Report out-of-band	RW	1
	Reserved for instances 3, 6 to 10 and 12	Reserved	R	0x00
24:19	Reserved	Reserved	R	0x00
18:16	BURST_TIMEOUT	Response Timeout Bound: 0x0: No timeout 0x1: 1x base cycles 0x2: 4x base cycles 0x3: 16x base cycles 0x4: 64x base cycles	RW	0x00
15:11	Reserved	Reserved	R	0x00
10:8	RESP_TIMEOUT	Response Timeout Bound: 0x0: No timeout 0x1: 1x base cycles 0x2: 4x base cycles 0x3: 16x base cycles 0x4: 64x base cycles	RW	0x0

Bits	Field Name	Description	Type	Reset
	Reserved for instances 3, 6 to 10 and 12	Reserved	R	0x0
7:5	Reserved	Reserved	R	0x0
4	REJECT	Request rejection control 0x0:Normal operation 0x1:Block requests from the initiator.	RW	0
3:1	Reserved	Reserved	R	0x0
0	CORE_RESET	Reset control for agent and reset control on core 0x0:Core reset control inactive 0x1:Core reset control active	RW	0

Table 5-43. Register Call Summary for Register L3_IA_AGENT_CONTROL

L3 Interconnect

- Error Handling: [0] [1] [2]
- Error Analysis: [3] [4]
- L3 Initiator Agent (L3 IA) Registers Manual: [5] [6] [7] [8]

Table 5-44. L3_IA_AGENT_STATUS

Address Offset	0x028
Physical Address	See Table 5-34 to Table 5-37
Description	Agent Status Register
Type	RW

63	62	61	60	59	58	57	56	55	54	53	52	51	50	49	48	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32
Reserved																															

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved		INBAND_ERROR_SECONDARY	INBAND_ERROR_PRIMARY	Reserved											BURST_TIMEOUT	TIMEBASE				Reserved			RESP_TIMEOUT	READEX	BURST	RESP_WAITING	REQ_ACTIVE	Reserved			CORE_RESET

Bits	Field Name	Description	Type	Reset
63:30	Reserved	Reserved	R	0x000000000
29	INBAND_ERROR_ SECONDARY	Error Status for in-band errors indicating a debug error. Read 0x0:No in-band error received Write 0x0:Ignored Read 0x1:In-band error received Write 0x1:Clear in-band error	RW	0
	Reserved for instances 3 to 12	Reserved	R	0x0
28	INBAND_ERROR_ PRIMARY	Error Status for in-band errors indicating application Error Read 0x0:No in-band error received Write 0x0:Ignored	RW	0

Bits	Field Name	Description	Type	Reset
		Read 0x1:In-band error received Write 0x1:Clear in-band error		
27:17	Reserved	Reserved	R	0x0
16	BURST_TIMEOUT	Status of open burst and	R	0
15:12	TIMEBASE	Observation of timebase signals for internal verification	R	0x0
11:9	Reserved	Reserved	R	0x0
8	RESP_TIMEOUT	Response timeout status	R	0
	Reserved for instances 3, 6 to 10 and 12	Reserved	R	0
7	READEX	Status of ReadEx/Write	R	0
6	BURST	Status of open burst	R	0
5	RESP_WAITING	Response Waiting	R	0
	Reserved for instance 3,5,7 and 8	Reserved	R	0
4	REQ_ACTIVE	Requests outstanding	R	0
3:1	Reserved	Reserved	R	0x0
0	CORE_RESET	Reset input from core interface	R	0

Table 5-45. Register Call Summary for Register L3_IA_AGENT_STATUS

L3 Interconnect

- Error Handling: [0] [1]
- Error Analysis: [2] [3] [4]
- L3 Initiator Agent (L3 IA) Registers Manual: [5] [6] [7] [8]

Table 5-46. L3_IA_ERROR_LOG

Address Offset	0x058
Physical Address	See Table 5-34 to Table 5-37
Description	Error log register of IA block
Type	RW

63	62	61	60	59	58	57	56	55	54	53	52	51	50	49	48	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32
Reserved																REQ_INFO															

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MULTI	SECONDARY	Reserved		CODE				Reserved								INITID								Reserved					CMD		

Bits	Field Name	Description	Type	Reset
63:48	Reserved	Reserved	R	0x0000
47:32	REQ_INFO	MReqInfo bits of command that caused the error	R	0x0000
31	MULTI	Multiple Errors	RW	0
		Write 0x0:Ignored		
		Read 0x0:Multiple error not seen		
		Write 0x1:Clear MULTI flag		
		Read 0x1:Multiple error seen		
30	SECONDARY	Indicates whether error was primary or secondary	RW	0
		Write 0x0:Ignored		
		Read 0x0:Primary Error		

Bits	Field Name	Description	Type	Reset
		Write 0x1:Reset SECONDARY field Read 0x1:Secondary Error		
29:28	Reserved	Reserved	R	0x0
27:24	CODE	Error code	RW	0x0
23:16	Reserved	Reserved	R	0x00
15:8	INITID	Initiator ID from which the command was launched	R	0x00
7:3	Reserved	Reserved	R	0x00
2:0	CMD	Command that caused the error	R	0x0

Table 5-47. Register Call Summary for Register L3_IA_ERROR_LOG

L3 Interconnect
- Error Handling: [0] [1] [2] [3] [4]
- Error Analysis: [5] [6] [7] [8] [9] [10] [11] [12]
- L3 Initiator Agent (L3 IA) Registers Manual: [13] [14] [15] [16]

Table 5-48. L3_IA_ERROR_LOG_ADDR

Address Offset	0x060
Physical Address	See Table 5-34 to Table 5-37
Description	Error log address register of IA block
Type	R

63	62	61	60	59	58	57	56	55	54	53	52	51	50	49	48	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32
Reserved																															

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDR																															

Bits	Field Name	Description	Type	Reset
63:32	Reserved	Reserved	R	0x000000
31:0	ADDR	Address of the command that caused the error	R	0x0000000000

Table 5-49. Register Call Summary for Register L3_IA_ERROR_LOG_ADDR

L3 Interconnect
- Error Handling: [0]
- Error Analysis: [1]
- L3 Initiator Agent (L3 IA) Registers Manual: [2] [3] [4] [5]

5.2.5.2 L3 Target Agent (L3 TA)

This section describes the TA register block. Each TA in L3 interconnect has its own register block.

The following are the TA registers:
- SDRAM memory scheduler (TA_SMS module)
- General-purpose memory controller (TA_GPMC module)
- On-chip memory RAM (TA_OCM_RAM module)
- On-chip memory ROM (TA_OCM_ROM module)
- Master D2D (TA_MAD2D module)
- IVA2.2 subsystem (TA_IVA2.2 module)
- SGX subsystem (TA_SGX module)
- L4-Core interconnect (TA_L4_CORE module)

- L4-Per interconnect (TA_L4_PER module)
- L4-Emu interconnect (TA_L4_EMU module)

Table 5-50 through Table 5-53 lists all initiator target registers and their physical addresses depending on the module instance.

Table 5-54 through Table 5-64 describe the individual common registers in the module instance.

Table 5-50. Target Agent Common Register Summary

Register Name	Type	Register Width (Bits)	TA_SMS Physical Address	TA_GPMC Physical Address	TA_OCM_RAM Physical Address
L3_TA_COMPONENT	R	64	0x6800 2000	0x6800 2400	0x6800 2800
L3_TA_CORE	R	64	0x6800 2018	0x6800 2418	0x6800 2818
L3_TA_AGENT_CONTROL	RW	64	0x6800 2020	0x6800 2420	0x6800 2820
L3_TA_AGENT_STATUS	R	64	0x6800 2028	0x6800 2428	0x6800 2828
L3_TA_ERROR_LOG	RW	64	0x6800 2058	0x6800 2458	0x6800 2858
L3_TA_ERROR_LOG_ADDR	R	64	0x6800 2060	0x6800 2460	0x6800 2860

Table 5-51. Target Agent Common Register Summary

Register Name	Type	Register Width (Bits)	TA_OCM_ROM Physical Address	TA_MAD2D Physical Address
L3_TA_COMPONENT	R	64	0x6800 2C00	0x6800 3400
L3_TA_CORE	R	64	0x6800 2C18	0x6800 3418
L3_TA_AGENT_CONTROL	RW	64	0x6800 2C20	0x6800 3420
L3_TA_AGENT_STATUS	R	64	0x6800 2C28	0x6800 3428
L3_TA_ERROR_LOG	RW	64	0x6800 2C58	0x6800 3458
L3_TA_ERROR_LOG_ADDR	R	64	0x6800 2C60	0x6800 3460

Table 5-52. Target Agent Common Register Summary

Register Name	Type	Register Width (Bits)	TA_IVA2.2 Physical Address	TA_SGX Physical Address
L3_TA_COMPONENT	R	64	0x6800 6000	0x6800 6400
L3_TA_CORE	R	64	0x6800 6018	0x6800 6418
L3_TA_AGENT_CONTROL	RW	64	0x6800 6020	0x6800 6420
L3_TA_AGENT_STATUS	R	64	0x6800 6028	0x6800 6428
L3_TA_ERROR_LOG	RW	64	0x6800 6058	0x6800 6458
L3_TA_ERROR_LOG_ADDR	R	64	0x6800 6060	0x6800 6460

Table 5-53. Target Agent Common Register Summary

Register Name	Type	Register Width (Bits)	TA_L4_CORE Physical Address	TA_L4_PER Physical Address	TA_L4_EMU Physical Address
L3_TA_COMPONENT	R	64	0x6800 6800	0x6800 6C00	0x6800 7000
L3_TA_CORE	R	64	0x6800 6818	0x6800 6C18	0x6800 7018
L3_TA_AGENT_CONTROL	RW	64	0x6800 6820	0x6800 6C20	0x6800 7020
L3_TA_AGENT_STATUS	RW	64	0x6800 6828	0x6800 6C28	0x6800 7028
L3_TA_ERROR_LOG	RW	64	0x6800 6858	0x6800 6C58	0x6800 7058
L3_TA_ERROR_LOG_ADDR	R	64	0x6800 6860	0x6800 6C60	0x6800 7060

5.2.5.2.1 L3 Target Agent (L3 TA) Registers Description

Table 5-54. L3_TA_COMPONENT

Address Offset	0x000
Physical Address	See Table 5-50 to Table 5-53
Description	Component register of target agent
Type	R

63	62	61	60	59	58	57	56	55	54	53	52	51	50	49	48	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32
Reserved																															

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CODE																REV															

Bits	Field Name	Description	Type	Reset
63:32	Reserved	Reserved	R	0x00000000
31:16	CODE	Component Code	R	See (1).
15:0	REV	Revision of the component	R	See (1).

(1) TI Internal Data

Table 5-55. Register Call Summary for Register L3_TA_COMPONENT

L3 Interconnect

- L3 Target Agent (L3 TA) Registers Manual: [0] [1] [2] [3]

Table 5-56. L3_TA_CORE

Address Offset	0x018
Physical Address	See Table 5-50 to Table 5-53
Description	Core register of Target Agent
Type	RW

63	62	61	60	59	58	57	56	55	54	53	52	51	50	49	48	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32
Reserved																VEND_CODE															

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CORE_CODE																REV_CODE															

Bits	Field Name	Description	Type	Reset
63:48	Reserved	Reserved	R	0x0000
47:32	VEND_CODE	Vendor Code	R	See (1).
31:16	CORE_CODE	Core code	R	See (1).
15:0	REV_CODE	Revision Code	R	See (1).

(1) TI Internal Data

Table 5-57. Register Call Summary for Register L3_TA_CORE

L3 Interconnect

- L3 Target Agent (L3 TA) Registers Manual: [0] [1] [2] [3]

Table 5-58. L3_TA_AGENT_CONTROL

Address Offset	0x020
Physical Address	See Table 5-50 to Table 5-53
Description	Agent control register of TA block.
Type	RW

63	62	61	60	59	58	57	56	55	54	53	52	51	50	49	48	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32
Reserved																															

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved						REQ_TIMEOUT_REP	SERROR_REP	Reserved													REQ_TIMEOUT			Reserved			REJECT	Reserved			CORE_RESET

Bits	Field Name	Description	Type	Reset
63:26	Reserved	Reserved	R	0x0000000000
25	REQ_TIMEOUT_REP	Request Timeout Reporting	RW	1
		0x0: No special reporting		
		0x1: Report out of band		
24	SERROR_REP	SError reporting	RW	1
		0x0: Suppress Serror reporting		
		0x1: Report Serror		
	Reserved for instances 1, 3 to 6	Reserved	R	0x0000
23:11	Reserved	Reserved	R	0x0000
10:8	REQ_TIMEOUT	Request Timeout Bound:	RW	0x0
		0x0: No timeout		
		0x1: 1x base cycles		
		0x2: 4x base cycles		
		0x3: 16x base cycles		
		0x4: 64x base cycles		
7:5	Reserved	Reserved	R	0x0
4	REJECT	Request rejection control	RW	0
		0x0: Request rejection control		
		0x1: Block requests to this target		
3:1	Reserved	Reserved	R	0x0
0	CORE_RESET	Reset output on core	RW	0
		0x0: Inactive		
		0x1: Reset control active		

Table 5-59. Register Call Summary for Register L3_TA_AGENT_CONTROL

L3 Interconnect

- Error Handling: [0] [1]
- Error Analysis: [2]
- L3 Target Agent (L3 TA) Registers Manual: [3] [4] [5] [6]

Table 5-60. L3_TA_AGENT_STATUS

Address Offset	0x028
Physical Address	See Table 5-50 to Table 5-53
Description	Agent Status Register.
Type	RW

63	62	61	60	59	58	57	56	55	54	53	52	51	50	49	48	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32
Reserved																															

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							SERROR	Reserved							BURST_CLOSE	TIMEBASE				Reserved			REQ_TIMEOUT	READEX	BURST	RESP_ACTIVE	REQ_WAITING	Reserved			CORE_RESET

Bits	Field Name	Description	Type	Reset
63:25	Reserved	Reserved	R	0x0000000000
24	SERROR	Serror assertion detected	RW	0
	Reserved for instances 1, 3 to 6	Reserved	R	0
23:17	Reserved	Reserved	R	0x00
16	BURST_CLOSE	Forced burst close status Read 0x0: Normal operation Read 0x1: Burst close command	R	0
15:12	TIMEBASE	Observation of timebase signals.	R	0x0
11:9	Reserved	Reserved	R	0x0
8	REQ_TIMEOUT	Request timeout status Read 0x0: Normal operation Read 0x1: Request timed out, responding ERR to all the requests	R	0
7	READEX	Status of readEx/Write Read 0x0: No pending ReadEx Read 0x1: ReadEx pending on at lease one thread	R	0
6	BURST	Status of open burst Read 0x0: No open burst Read 0x1: Open burst on at least one thread	R	0
5	RESP_ACTIVE	Responses outstanding Read 0x0: No responses outstanding Read 0x1: Response outstanding in the target	R	0
4	REQ_WAITING	Requests waiting Read 0x0: No request waiting Read 0x1: Request waiting for acceptance by target	R	0
3:1	Reserved	Reserved	R	0x0
0	CORE_RESET	Reset input from core interface Read 0x0: Reset inactive Read 0x1: Reset active	R	0

Table 5-61. Register Call Summary for Register L3_TA_AGENT_STATUS

L3 Interconnect

- Error Handling: [0]
- L3 Target Agent (L3 TA) Registers Manual: [1] [2] [3] [4]

Table 5-62. L3_TA_ERROR_LOG

Address Offset	0x058
Physical Address	See Table 5-50 to Table 5-53
Description	Error log register of TA block - logs error detected by a target agent.
Type	RW

63	62	61	60	59	58	57	56	55	54	53	52	51	50	49	48	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32
Reserved																								REQ_INFO							

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MULTI	Reserved			CODE				Reserved								INITID								Reserved					CMD		

Bits	Field Name	Description	Type	Reset
63:42	Reserved	Reserved	R	0x0000
41:32	REQ_INFO	MReqInfo bits of command that caused the error	R	0x0000
31	MULTI	Multiple Errors	RW	0
		Write 0x0: Ignored		
		Read 0x0: Multiple error not seen		
		Write 0x1: Clear MULTI flag		
		Read 0x1: Multiple error seen		
30:28	Reserved	Reserved	R	0x0
27:24	CODE	Error code	RW	0x0
23:16	Reserved	Reserved	R	0x00
15:8	INITID	Initiator ID from which command was launched	R	0x00
7:3	Reserved	Reserved	R	0x00
2:0	CMD	Command that caused the error	R	0x0

Table 5-63. Register Call Summary for Register L3_TA_ERROR_LOG

L3 Interconnect

- Error Handling: [0]
- Error Analysis: [1]
- L3 Target Agent (L3 TA) Registers Manual: [2] [3] [4] [5]

Table 5-64. L3_TA_ERROR_LOG_ADDR

Address Offset	0x060
Physical Address	See Table 5-50 to Table 5-53
Description	Error log address register of TA block
Type	RW

63	62	61	60	59	58	57	56	55	54	53	52	51	50	49	48	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32
Reserved																															

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDR																															

Bits	Field Name	Description	Type	Reset
63:32	Reserved	Reserved	R	0x000000
31:0	ADDR	Address of the command that caused the error	R	0x0000000000

Table 5-65. Register Call Summary for Register L3_TA_ERROR_LOG_ADDR

L3 Interconnect

- Error Handling: [0]
- L3 Target Agent (L3 TA) Registers Manual: [1] [2] [3] [4]

5.2.5.3 Register Target (RT)

This section describes the RT module.

Table 5-66 lists the RT registers and their physical addresses.

Table 5-67 through Table 5-73 describe the individual registers in the module instance.

Table 5-66. RT Register Summary

Register Name	Type	Register Width (Bits)	Address Offset	Physical Address
L3_RT_COMPONENT	R	64	0x000	0x6800 0000
L3_RT_NETWORK	R	64	0x010	0x6800 0010
L3_RT_INITID_READBACK	R	64	0x070	0x6800 0070
L3_RT_NETWORK_CONTROL	RW	64	0x078	0x6800 0078

5.2.5.3.1 Register Target (RT) Registers Description

Table 5-67. L3_RT_COMPONENT

Address Offset	0x000		
Physical Address	0x6800 0000	**Instance**	RT
Description	This register identifies the component to which this register block belongs.		
Type	R		

63	62	61	60	59	58	57	56	55	54	53	52	51	50	49	48	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32
Reserved																															

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CODE																REV															

Bits	Field Name	Description	Type	Reset
63:32	Reserved	Reserved	R	0x00000000
31:16	CODE	Component Code	R	See (1).
15:0	REV	Revision of the component	R	See (1).

(1) TI Internal Data

Table 5-68. Register Call Summary for Register L3_RT_COMPONENT

L3 Interconnect

- Register Target (RT) Registers Manual: [0]

Table 5-69. L3_RT_NETWORK

Address Offset	0x010		
Physical Address	0x6800 0010	**Instance**	RT
Description	This register identifies the interconnect and is present only in the register target.		
Type	R		

63	62	61	60	59	58	57	56	55	54	53	52	51	50	49	48	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32
ID																															

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																															

Bits	Field Name	Description	Type	Reset
63:32	ID	Unique Interconnect ID	R	0x00000000
31:0	Reserved	Reserved	R	0x00000000

Table 5-70. Register Call Summary for Register L3_RT_NETWORK

L3 Interconnect

- Register Target (RT) Registers Manual: [0]

Table 5-71. L3_RT_INITID_READBACK

Address Offset	0x070		
Physical Address	0x6800 0070	**Instance**	RT
Description	This register is used by initiators to discover their own identity.		
Type	R		

63	62	61	60	59	58	57	56	55	54	53	52	51	50	49	48	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32
Reserved																															

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																								INITID							

Bits	Field Name	Description	Type	Reset
63:8	Reserved	Reserved	R	0x00000000000000
7:0	INITID	Returns initiator ID of core thread that initiated the read	R	0x18

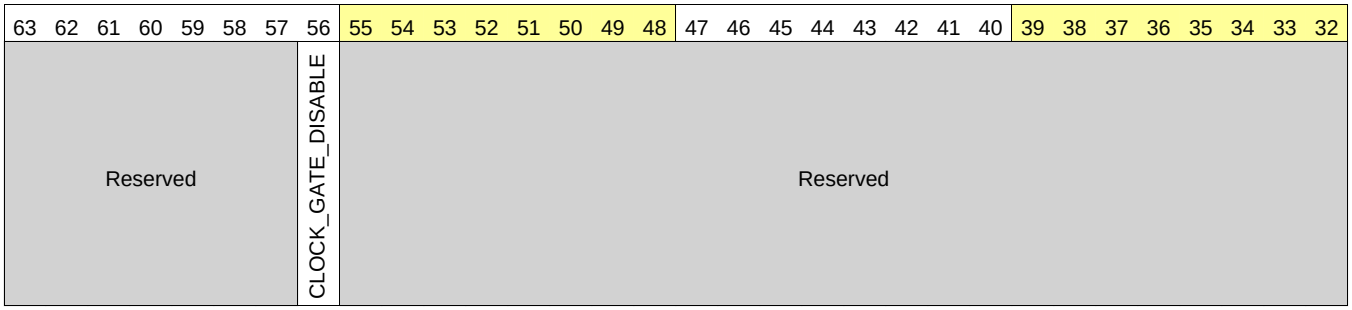
Table 5-72. Register Call Summary for Register L3_RT_INITID_READBACK

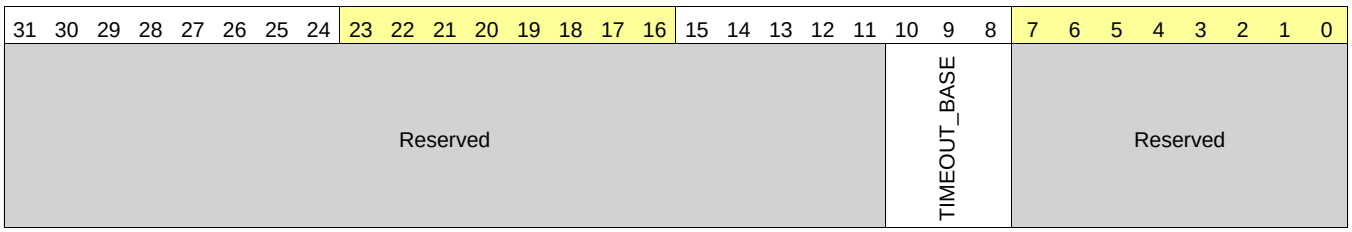
L3 Interconnect

- Register Target (RT) Registers Manual: [0]

Table 5-73. L3_RT_NETWORK_CONTROL

Address Offset	0x068		
Physical Address	0x6800 0078	**Instance**	RT
Description	It controls such interconnect wide functions as the timeout base scale and the disabling of fine grained hardware clock gating.		
Type	RW		





Bits	Field Name	Description	Type	Reset
63:57	Reserved	Reserved	R	0x00
56	CLOCK_GATE_ DISABLE	Overrides fine grained hardware clock gating	RW	0
55:11	Reserved	Reserved	R	0x000000000000
10:8	TIMEOUT_BASE	Timeout base period in register target clock cycles Program the timeout base period. Each of the agent timeout features is programmed as a multiple of the timeout base period. These timeout bases are: 0x0: Timeout disabled 0x1: L3 interconnect clock cycles divided by 64 0x2: L3 interconnect clock cycles divided by 256 0x3: L3 interconnect clock cycles divided by 1024 0x4: L3 interconnect clock cycles divided by 4096	RW	0x0
7:0	Reserved	Reserved	R	0x00

Table 5-74. Register Call Summary for Register L3_RT_NETWORK_CONTROL

L3 Interconnect
- Error Handling: [0]
- Register Target (RT) Registers Manual: [1]

5.2.5.4 Protection Mechanism (PM)

This section describes the protection mechanism register block.

The following are the protection mechanism registers:

- Register target (PM_RT module)
- General-purpose memory controller (PM_GPMC module)
- On-chip RAM (PM_OCM_RAM module)

- On-chip ROM (PM_OCM_ROM module)
- IVA2.2 subsystem (PM_IVA2.2 module)

Table 5-75 and Table 5-77 list the protection registers and their physical addresses, depending on the module instance.

Table 5-78 through Table 5-88 describe the individual common registers in the module instance.

Table 5-75. Protection Mechanism Common Register Summary

Register Name	Type	Register Width (Bits)	PM_RT Physical Address	PM_GPMC Physical Address
L3_PM_ERROR_LOG	RW	64	0x6801 0020	0x6801 2420
L3_PM_CONTROL	RW	64	0x6801 0028	0x6801 2428
L3_PM_ERROR_CLEAR_SINGLE	R	64	0x6801 0030	0x6801 2430
L3_PM_ERROR_CLEAR_MULTI	R	64	0x6801 0038	0x6801 2438
L3_PM_REQ_INFO_PERMISSION_i (1)	RW	64	0x6801 0048 + (0x20*i)	0x6801 2448 + (0x20*i)
L3_PM_READ_PERMISSION_i (1)	RW	64	0x6801 0050 + (0x20*i)	0x6801 2450 + (0x20*i)
L3_PM_WRITE_PERMISSION_i (1)	RW	64	0x6801 0058 + (0x20*i)	0x6801 2458 + (0x20*i)
L3_PM_ADDR_MATCH_k (2)	RW	64	0x6801 0060 + (0x20*k)	0x6801 2460 + (0x20*k)

(1) i = 0 to 1 for PM_RT
i = 0 to 7 for PM_GPMC
(2) k = 1 to 1 for PM_RT
k = 1 to 7 for PM_GPMC

Table 5-76. Protection Mechanism Common Register Summary

Register Name	Type	Register Width (Bits)	PM_OCM_RAM Physical Address	PM_OCM_ROM Physical Address
L3_PM_ERROR_LOG	RW	64	0x6801 2820	0x6801 2C20
L3_PM_CONTROL	RW	64	0x6801 2828	0x6801 2C28
L3_PM_ERROR_CLEAR_SINGLE	R	64	0x6801 2830	0x6801 2C30
L3_PM_ERROR_CLEAR_MULTI	R	64	0x6801 2838	0x6801 2C38
L3_PM_REQ_INFO_PERMISSION_i (1)	RW	64	0x6801 2848 + (0x20*i)	0x6801 2C48 + (0x20*i)
L3_PM_READ_PERMISSION_i (1)	RW	64	0x6801 2850 + (0x20*i)	0x6801 2C50 + (0x20*i)
L3_PM_WRITE_PERMISSION_i (1)	RW	64	0x6801 2858 + (0x20*i)	0x6801 2C58 + (0x20*i)
L3_PM_ADDR_MATCH_k (2)	RW	64	0x6801 2860 + (0x20*k)	0x6801 2C60 + (0x20*k)

(1) i = 0 to 1 for PM_OCM_ROM
i = 0 to 7 for PM_OCM_RAM
(2) k = 1 to 1 for PM_OCM_ROM
k = 1 to 7 for PM_OCM_RAM

Table 5-77. Protection Mechanism Common Register Summary

Register Name	Type	Register Width (Bits)	PM_MAD2D Physical Address	PM_IVA2.2 Physical Address
L3_PM_ERROR_LOG	RW	64	0x6801 3020	0x6801 4020
L3_PM_CONTROL	RW	64	0x6801 3028	0x6801 4028
L3_PM_ERROR_CLEAR_SINGLE	R	64	0x6801 3030	0x6801 4030
L3_PM_ERROR_CLEAR_MULTI	R	64	0x6801 3038	0x6801 4038
L3_PM_REQ_INFO_PERMISSION_i (1)	RW	64	0x6801 3048 + (0x20*i)	0x6801 4048 + (0x20*i)

(1) i = 0 to 7 for PM_MAD2D
i = 0 to 3 for PM_IVA2.2

Table 5-77. Protection Mechanism Common Register Summary (continued)

Register Name	Type	Register Width (Bits)	PM_MAD2D Physical Address	PM_IVA2.2 Physical Address
L3_PM_READ_PERMISSION_i (1)	RW	64	0x6801 3050 + (0x20*i)	0x6801 4050 + (0x20*i)
L3_PM_WRITE_PERMISSION_i (1)	RW	64	0x6801 3058 + (0x20*i)	0x6801 4058 + (0x20*i)
L3_PM_ADDR_MATCH_k (2)	RW	64	0x6801 3060 + (0x20*k)	0x6801 4060 + (0x20*k)

(2) k = 1 to 7 for PM_MAD2D
k = 1 to 3 for PM_IVA2.2

5.2.5.4.1 Protection Mechanism (PM) Registers Description

Table 5-78. L3_PM_ERROR_LOG

Address Offset	0x20
Physical Address	See Table 5-75 to Table 5-77
Description	This register logs errors detected by the protection mechanism.
Type	RW

63	62	61	60	59	58	57	56	55	54	53	52	51	50	49	48	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32
Reserved																															

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MULTI	SECONDARY	Reserved		CODE				Reserved			REQ_INFO					INITID								Reserved	REGION			Reserved	CMD		

Bits	Field Name	Description	Type	Reset
63:32	Reserved	Reserved	R	0x00000000
31	MULTI	Multiple errors 0x0:Multiple errors not seen 0x1:Multiple errors seen	RW1toClr	0
30	SECONDARY	Secondary error present	RW1toClr	0
29:28	Reserved	Reserved	R	0x0
27:24	CODE	Error Code see Table 5-26	RW1toClr	0x0
23:21	Reserved	Reserved	R	0x0
20:16	REQ_INFO	MReqInfo bits of command selected for protection checking see Table 5-22	R	0x00
15:8	INITID	Initiator ID from which the command was launched see Table 5-19	R	0x00
7	Reserved	Reserved	R	0
6:4	REGION	Protection region number that command mapped to	R	0x0
3	Reserved	Reserved	R	0
2:0	CMD	Command that caused the error see Table 5-1	R	0x0

Table 5-79. Register Call Summary for Register L3_PM_ERROR_LOG

L3 Interconnect

- L3 Protection and Firewalls: [0]
- Error Analysis: [1] [2] [3] [4] [5] [6] [7]
- Protection Mechanism (PM) Registers Manual: [8] [9] [10]

Table 5-80. L3_PM_CONTROL

Address Offset	0x28
Physical Address	See Table 5-75 to Table 5-77
Description	This register controls protection mechanism functions such as error reporting.
Type	RW

63	62	61	60	59	58	57	56	55	54	53	52	51	50	49	48	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32
Reserved																															

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved						ERROR_SECONDARY_REP	ERROR_REP	Reserved																							

Bits	Field Name	Description	Type	Reset
63:26	Reserved	Reserved	R	0x0000000000
25	ERROR_SECONDARY_REP	Out of band error reporting 0x0: Out of band error reporting suppress 0x1: Out of band error report	RW	1
24	ERROR_REP	Out of band error reporting 0x0: Out of band error reporting suppress 0x1: Out of band error report	RW	1
23:0	Reserved	Reserved	R	0x000000

Table 5-81. Register Call Summary for Register L3_PM_CONTROL

L3 Interconnect

- Protection Mechanism (PM) Registers Manual: [0] [1] [2]

Table 5-82. L3_PM_ERROR_CLEAR_SINGLE

Address Offset	0x30
Physical Address	See Table 5-75 to Table 5-77
Description	Read to clear single errors from error log
Type	R

63	62	61	60	59	58	57	56	55	54	53	52	51	50	49	48	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32
Reserved																															

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																															CLEAR

Bits	Field Name	Description	Type	Reset
63:1	Reserved	Reserved	R	0x0000000000000000
0	CLEAR	Clear single error from log	R	0

Table 5-83. Register Call Summary for Register L3_PM_ERROR_CLEAR_SINGLE

L3 Interconnect

- Error Analysis: [0]
- Protection Mechanism (PM) Registers Manual: [1] [2] [3]

Table 5-84. L3_PM_ERROR_CLEAR_MULTI

Address Offset	0x38
Physical Address	See Table 5-75 to Table 5-77
Description	Read to clear multiple errors from error log
Type	R

63	62	61	60	59	58	57	56	55	54	53	52	51	50	49	48	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32
Reserved																															

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																															CLEAR

Bits	Field Name	Description	Type	Reset
63:1	Reserved	Reserved	R	0x0000000000000000
0	CLEAR	Clear multiple error from log	R	0

Table 5-85. Register Call Summary for Register L3_PM_ERROR_CLEAR_MULTI

L3 Interconnect

- Error Analysis: [0]
- Protection Mechanism (PM) Registers Manual: [1] [2] [3]

Table 5-86. L3_PM_REQ_INFO_PERMISSION_i

Address Offset	0x38
Physical Address	See Table 5-75 to Table 5-77
Description	It configures a protection region's permissions using the MReqInfo bits selected for the PM by the structural configuration.
Type	R/W

63	62	61	60	59	58	57	56	55	54	53	52	51	50	49	48	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32
Reserved																															

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																REQ_INFO															

Bits	Field Name	Description	Type	Reset
63:16	Reserved	Reserved	R	0x000000000000
15:0	REQ_INFO	Request info permission bits for region i, see Table 5-22 for bitfield description.	RW	See Table 5-88.

Table 5-87. Register Call Summary for Register L3_PM_REQ_INFO_PERMISSION_i

Interconnect Overview

- Terminology: [0]

L3 Interconnect

- L3 Protection and Firewalls: [1] [2] [3] [4] [5]
- Typical Example of Firewall Programming Example: [6] [7] [8] [9] [10] [11] [12] [13]
- Protection Mechanism (PM) Registers Manual: [14] [15] [16]

Table 5-88. Reset Value for REQ_INFO_PERMISSION

	Regions							
	0	**1**	**2**	**3**	**4**	**5**	**6**	**7**
PM_RT	0xFFFF	0x000						
PM_GPMC	0x0000	0x----	0x----	0xFFFF	0xFFFF	0xFFFF	0xFFFF	0xFFFF
PM_OCM_RAM	0x0000	0xFFFF	0xFFFF	0xFFFF	0xFFFF	0xFFFF	0xFFFF	0xFFFF
PM_OCM_ROM	0x----	0xFFFF						
PM_MAD2D	0x0000	0xFFFF	0xFFFF	0xFFFF	0xFFFF	0xFFFF	0xFFFF	0xFFFF
PM_IVA2.2	0x0000	0xFFFF	0xFFFF	0xFFFF				

Table 5-89. L3_PM_READ_PERMISSION_i

Address Offset	0x050 + (0x20+i)	**Index**	i = 0 to 1 for PM_RT i = 0 to 7 for PM_GPMC i = 0 to 1 for PM_OCM_ROM i = 0 to 3 for PM_IVA2 i = 0 to 7 for PM_OCM_RAM
Physical Address	See Table 5-75 to Table 5-77		
Description	It configures protection region permissions for read incoming commands.		
Type	RW		

63	62	61	60	59	58	57	56	55	54	53	52	51	50	49	48	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32
Reserved																															

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																SGX	Reserved	DAP	CAM	IVA2_MMU	USB_HS_Host	DISPSS	Reserved	SAD2D	USB_HS_OTG	SDMA	IVA2_DMA	MPU	Reserved		

Bits	Field Name	Description	Type	Reset
63:15	Reserved	Reserved	R	0x0
14	SGX	Read permission for the SGX	RW	See Table 5-93
13	Reserved	Reserved	RW	0x0
12	DAP	Read permission for the DAP	RW	See Table 5-93
11	CAM	Read permission for the CAMERA SS	RW	See Table 5-93
10	IVA2_MMU	Read permission for the IVA2 MMU	RW	See Table 5-93
9	USB_HS_Host	Read permission for the USB_HS_Host	RW	See Table 5-93
8	DISPSS	Write permission for the DISPLAY SS	RW	See Table 5-93
7:6	Reserved	Reserved	RW	0x0
5	SAD2D	Read permission for the SAD2D	RW	See Table 5-93
4	USB_HS_OTG	Read permission for the USB_HS_OTG	RW	See Table 5-93
3	SDMA	Read permission for the system DMA	RW	See Table 5-93

Bits	Field Name	Description	Type	Reset
2	IVA2_DMA	Read permission for the IVA2	RW	See Table 5-93
1	MPU	Read permission for the MPU	RW	See Table 5-93
0	Reserved	Reserved	RW	0x0

Table 5-90. Register Call Summary for Register L3_PM_READ_PERMISSION_i

L3 Interconnect

- L3 Protection and Firewalls: [0] [1]
- Typical Example of Firewall Programming Example: [2] [3] [4] [5] [6] [7] [8]
- Protection Mechanism (PM) Registers Manual: [9] [10] [11] [12]

Table 5-91. L3_PM_WRITE_PERMISSION_i

Address Offset	0x058 + (0x20+i)	**Index**	i = 0 to 1 for PM_RT i = 0 to 7 for PM_GPMC i = 0 to 1 for PM_OCM_ROM i = 0 to 3 for PM_IVA2 i = 0 to 7 for PM_OCM_RAM
Physical Address	See Table 5-75 to Table 5-77		
Description	It configures protection region permissions for write incoming commands.		
Type	RW		

63	62	61	60	59	58	57	56	55	54	53	52	51	50	49	48	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32
Reserved																															

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																	SGX	Reserved	DAP	CAM	IVA2_MMU	USB_HS_Host	DISPSS	Reserved		SAD2D	USB_HS_OTG	SDMA	IVA2_DMA	MPU	Reserved

Bits	Field Name	Description	Type	Reset
63:15	Reserved	Reserved	R	0x0
14	SGX	Write permission for the SGX	RW	See Table 5-93
13	Reserved	Reserved	RW	0x0
12	DAP	Write permission for the DAP	RW	See Table 5-93
11	CAM	Write permission for the CAMERA SS	RW	See Table 5-93
10	IVA2_MMU	Write permission for the IVA2 MMU	RW	See Table 5-93
9	USB_HS_Host	Write permission for the USB_HS_Host	RW	See Table 5-93
8	DISPSS	Write permission for the DISPLAY SS	RW	See Table 5-93
7:6	Reserved	Reserved	RW	0x0
5	SAD2D	Write permission for the SAD2D	RW	See Table 5-93
4	USB_HS_OTG	Write permission for the USB_HS_OTG	RW	See Table 5-93
3	SDMA	Write permission for the system DMA	RW	See Table 5-93
2	IVA2_DMA	Write permission for the IVA2	RW	See Table 5-93
1	MPU	Write permission for the MPU	RW	See Table 5-93
0	Reserved	Reserved	RW	0x0

Table 5-92. Register Call Summary for Register L3_PM_WRITE_PERMISSION_i

L3 Interconnect

- L3 Protection and Firewalls: [0] [1]
- Typical Example of Firewall Programming Example: [2] [3] [4] [5] [6] [7] [8]
- Protection Mechanism (PM) Registers Manual: [9] [10] [11] [12]

Table 5-93 shows bit available in L3_PM_READ_PERMISSION_i and L3_PM_WRITE_PERMISSION_i registers. All N/A are considered as reserved bits with a Read access.

Table 5-93. Bit Availability and Initialization Values for L3_PM_READ_PERMISSION_i and L3_PM_WRITE_PERMISSION_i

PM	BITS											
	1: MPU	2: IVA2_ DMA	3: SDMA	4: USB_HS_ OTG	5: SAD2D	8: DISP SS	9: USB_HS_ Host	10: IVA2_ MMU	11: CAM	12: DAP	14: GFX	63:15 Reserved
PM_RT region 0 to 1	1	1	N/A	N/A	N/A	N/A	N/A	1	N/A	1	N/A	0x0000000000000
PM_GPMC region 0 to 7	1	1	1	1	1	N/A	1	1	N/A	1	1	0x0000000000001
PM_OCM_RAM region 0 to 7	1	1	1	1	1	1	1	1	1	1	1	0x0000000000001
PM_OCM_ROM region 0 to 1 (ro)[(1)]	1	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	1	N/A	0x0000000000000
PM_MAD2D region 0 to 7	1	1	1	1	N/A	1	1	1	1	1	1	0x0000000000001
PM_IVA2.2 region 0 to 3	1	1	1	N/A	N/A	N/A	N/A	1	N/A	1	N/A	0x0000000000000

(1) ROM is a read only memory; therefore, the write permission is set to 0.

Table 5-94. L3_PM_ADDR_MATCH_k

Address Offset	0x060 + (0x20+k)	**Index**	k = 1 to 1 for PM_RT k = 1 to 7 for PM_GPMC k = 1 to 1 for PM_OCM_ROM k = 1 to 3 for PM_IVA2 k = 1 to 7 for PM_OCM_RAM
Physical Address	See Table 5-75 to Table 5-77		
Description			
Type	R		

63	62	61	60	59	58	57	56	55	54	53	52	51	50	49	48	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32
Reserved																															

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								BASE_ADDR														LEVEL	Reserved	SIZE						ADDR_SPACE	

Bits	Field Name	Description	Type	Reset
63:20	Reserved	Reserved	R	0x00000000000
19:10	BASE_ADDR	Protection region base address	R	see Table 5-96
9	LEVEL	Protection region level.	R	see Table 5-96
8	Reserved	Reserved	R	0
7:3	SIZE	Protection region size	R	see Table 5-96
2:0	ADDR_SPACE	Protection region address space	R	see Table 5-96

Table 5-95. Register Call Summary for Register L3_PM_ADDR_MATCH_k

L3 Interconnect

- L3 Protection and Firewalls: [0] [1] [2] [3] [4] [5] [6]
- Typical Example of Firewall Programming Example: [7] [8] [9] [10] [11]
- Protection Mechanism (PM) Registers Manual: [12] [13] [14]

Table 5-96. Reset Value for L3_PM_ADDR_MATCH_k

PM	Region	BASE ADDRESS	LEVEL	SIZE	ADDR_SPACE
PM_RT	1	0x040	0x1	0x06	0x0
PM_GPMC	1	0x000	0x0	0x23	0x0
	2	0x000	0x0	0x00	0x0
	3	0x000	0x0	0x00	0x0
	4	0x000	0x0	0x00	0x0
	5	0x000	0x0	0x00	0x0
	6	0x000	0x0	0x00	0x0
	7	0x000	0x0	0x00	0x0
PM_OCM_RAM	1	0x000	0x0	0x00	0x0
	2	0x03E	0x0	0x02	0x0
	3	0x000	0x0	0x00	0x0
	4	0x000	0x0	0x00	0x0
	5	0x000	0x0	0x00	0x0
	6	0x000	0x0	0x00	0x0
	7	0x000	0x0	0x00	0x0

Table 5-96. Reset Value for L3_PM_ADDR_MATCH_k (continued)

PM	Region	BASE ADDRESS	LEVEL	SIZE	ADDR_SPACE
PM_OCM_ROM	1	0x050	0x0	0x05	0x0
	2	0x000	0x0	0x00	0x0
	3	0x000	0x0	0x00	0x0
PM_MAD2D	1	0x000	0x0	0x00	0x0
	2	0x000	0x0	0x00	0x0
	3	0x000	0x0	0x00	0x0
	4	0x000	0x0	0x00	0x0
	5	0x000	0x0	0x00	0x0
	6	0x000	0x0	0x00	0x0
	7	0x000	0x0	0x00	0x0
PM_IVA2	1	0x000	0x0	0x00	0x0
	2	0x000	0x0	0x00	0x0
	3	0x000	0x0	0x00	0x0

5.2.5.5 Sideband Interconnect (SI)

This section describes the sideband interconnect register block.

Table 5-97 lists the SI registers and their physical addresses.

Table 5-98 through Table 5-102 describe the individual registers in the module instance.

Table 5-97. SI Register Summary

Register Name	Type	Register Width (Bits)	Address Offset	Physical Address
L3_SI_CONTROL	RW	64	0x020	0x6800 0420
L3_SI_FLAG_STATUS_0	R	64	0x110	0x6800 0510
L3_SI_FLAG_STATUS_1	R	64	0x130	0x6800 0530

5.2.5.5.1 Sideband Interconnect (SI) Registers Description

Table 5-98. L3_SI_CONTROL

Address Offset	0x020		
Physical Address	0x6800 0420	**Instance**	SI
Description	Control of register and sideband interconnect		
Type	RW		

63	62	61	60	59	58	57	56	55	54	53	52	51	50	49	48	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32
Reserved								CLOCK_GATE_DISABLE								Reserved															

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																															

Bits	Field Name	Description	Type	Reset
63:57	Reserved	Reserved for future use	R	0x00
56	CLOCK_GATE_DISABLE	Overrides fine grained hardware clock gating in register and sideband interconnect 0x0: Normal clock gating 0x1: Clock gating disabled	RW	0
55:0	Reserved	Reserved for future use	R	0x00000000000000

Table 5-99. Register Call Summary for Register L3_SI_CONTROL

L3 Interconnect

- Sideband Interconnect (SI) Registers Manual: [0]

Table 5-100. L3_SI_FLAG_STATUS_0

Address Offset	0x110		
Physical Address	0x6800 0510	**Instance**	SI
Description	They are used to observe the individual bits that make up a composite interconnect flag.		
Type	R		

63	62	61	60	59	58	57	56	55	54	53	52	51	50	49	48	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32
STATUS																															

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
STATUS																															

Bits	Field Name	Description	Type	Reset
63:0	STATUS	Status of sideband signals making up composite interconnect flag for application. See Table 5-29	R	0x0000000000000000

Table 5-101. Register Call Summary for Register L3_SI_FLAG_STATUS_0

L3 Interconnect

- Error Handling: [0]
- Error Analysis: [1] [2] [3] [4] [5]
- Sideband Interconnect (SI) Registers Manual: [6]

Table 5-102. L3_SI_FLAG_STATUS_1

Address Offset	0x130		
Physical Address	0x6800 0530	**Instance**	SI
Description	They are used to observe the individual bits that make up a composite interconnect flag.		
Type	R		

63	62	61	60	59	58	57	56	55	54	53	52	51	50	49	48	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32
STATUS																															

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
STATUS																															

Bits	Field Name	Description	Type	Reset
63:0	STATUS	Status of sideband signals making up composite interconnect flag for debug. See Table 5-30.	R	0x0000000000000000

Table 5-103. Register Call Summary for Register L3_SI_FLAG_STATUS_1

L3 Interconnect

- Error Handling: [0]
- Error Analysis: [1] [2]
- Sideband Interconnect (SI) Registers Manual: [3]

5.3 L4 Interconnects

5.3.1 Overview

To connect peripheral modules, the device uses four separate L4 interconnect structures. Although all L4 interconnects handle transfers with peripherals, the interconnects are in different power domains. The L4 interconnect is composed of the following:

- L4-Core: Includes the majority of the peripherals and the configuration interface for L3 interconnect system modules
- L4-Per: Includes peripherals that do not need to be mapped in the CORE power domain
- L4-Wakeup: Includes the peripherals attached to the WKUP power domain
- L4-Emu: Includes emulation peripherals attached to the EMU power domain

The following are the main features of the L4 interconnects:

- Single port to connect to L3 interconnect
  - L4-Core
  - L4-Per
- Dual ports for the following:
  - L4-Emu to connect to the L3 interconnect and DAP
  - L4-Wakeup to connect to the L4-Core and L4-Emu
- Single 32-bit initiator for the L3 port
- Multitarget ports (one per target interface on the L4 interconnect)
- 8-, 16-, or 32-bit data, single, or burst transactions
- Little-endian
- Nonblocking with fair arbitration between threads
- Target interfaces: Fully synchronous or divided synchronous
- Peak bandwidth of 2Mb/s of L4 frequency (in MHz)
- Latency: Three cycles on request, one cycle on response
- Protection logic provides user-configurable access control to targets by each initiator:
  - Firewall in L4-Core protects the core and wake-up peripherals.
  - Firewall in L4-Per protects per peripherals.
  - Firewall in L4-Emu protects emulation and wake-up peripherals

Note: L4-Wakeup has two input ports from L4-Core and L4-Emu. Therefore, wake-up peripherals appear in two locations in the memory mapping. Normally, L4-Emu limits its access except when debugging.

Figure 5-13 shows an overview of the L4 interconnects and the periphreals attached to them.

Figure 5-13. L4 Interconnect Overview

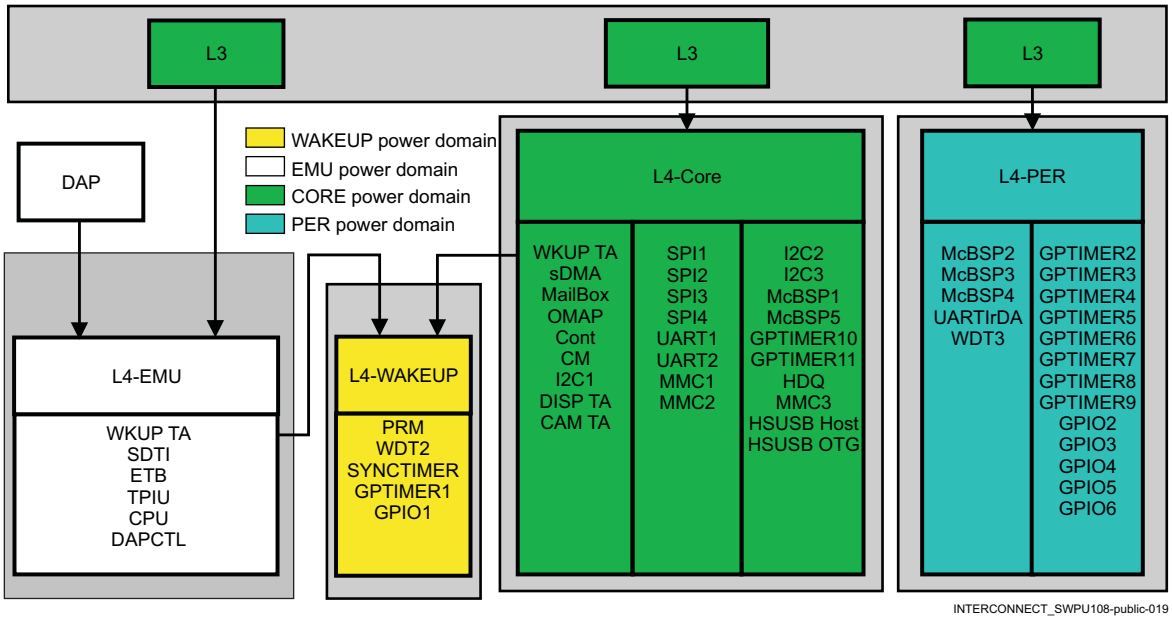
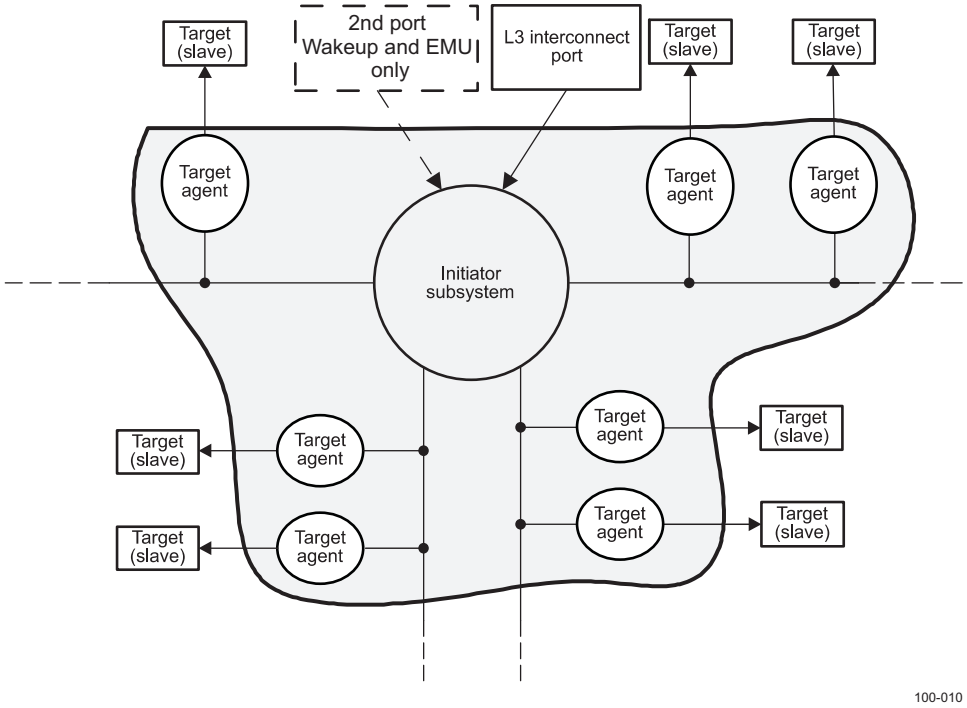


Figure 5-14 shows an internal view of the L4 interconnects in the overall interconnect. This architecture, with only one initiator module (the initiator subsystem) distributing transactions to all target modules (peripherals), enables the firewall functions of the L4 interconnects to be centralized at the L4 initiator level. The L4 firewall filters the accesses according to the configurable protection groups defined in the L4 address protection (AP) registers. Each module or agent is assigned to a protection group. Configuration is also defined in the L4 AP and is programmable on a module-per-module basis.

Figure 5-14. L4 Initiator-Target Connectivity for L4-Core and L4-Per



Note: As Figure 5-14 shows, targets are attached to branches. These branches have no functional effect and are present for timing closure reasons only.

5.3.1.1 L4-Core Interconnect

The L4-core interconnect handles only transfers to peripherals in the CORE power domain. Table 5-104 lists the TAs.

Table 5-104. L4-Core Target Agents

Module Name	Description
Display subsystem	Display subsystem configuration port
Camera subsystem	Camera subsystem port
USBHS OTG	Universal serial bus High-Speed port OTG
USBHS Host	Universal serial bus High-Speed port Host
USBTLL	USB Transceiver Less LInk
UART1	Universal asynchronous receiver transmitter port 1
UART2	Universal asynchronous receiver transmitter port 2
I2C1	Multimaster inter-integrated circuit 1
I2C2	Multimaster inter-integrated circuit 2
I2C3	Multimaster inter-integrated circuit 3
McBSP1	Multichannel buffered serial port 1
McBSP5	Multichannel buffered serial port 5
GPTIMER10	General-purpose timer 10
GPTIMER11	General-purpose timer 11
SPI1	Serial peripheral interface 1
SPI2	Serial peripheral interface 2
MMCHS1	Multimedia memory controller SDIO 1
MMCHS2	Multimedia memory controller SDIO 2
MMCHS3	Multimedia memory controller SDIO 3
HDQ/1-Wire	Single wire serial link low rate
MLB (mailbox)	Mailbox
SPI1	Serial peripheral interface 1
SPI2	Serial peripheral interface 2
SPI3	Serial peripheral interface 3
SPI4	Serial peripheral interface 4
sDMA	System DMA controller
L4-Wakeup	L4-Wakeup interconnect
CM	Clock manager
SCM	System control module

Note: A unique port is used for communication between the L3 interconnect and the L4-Core interconnect to allow the L3 initiators to access the L4-Core targets.

For the list of initiators authorized to access the L4-Core peripherals, see Table 5-14. For details on restricted access, see Section 5.3.3.3.1, *Protection Mechanism*.

5.3.1.2 L4-Per Interconnect

The L4-Per interconnect handles only transfers to peripherals in the PER power domain. Table 5-105 lists the TAs.

Table 5-105. L4-Per Target Agents

Module Name	Description
UARTIrDA	Universal asynchronous receiver/transmitter and infrared data association port
McBSP2	Multichannel buffered serial port 2
McBSP3	Multichannel buffered serial port 3
GPTIMER2	General-purpose timer 2
GPTIMER3	General-purpose timer 3
GPTIMER4	General-purpose timer 4
GPTIMER5	General-purpose timer 5
GPTIMER6	General-purpose timer 6
GPTIMER7	General-purpose timer 7
GPTIMER8	General-purpose timer 8
GPTIMER9	General-purpose timer 9
GPIO2	General-purpose I/O 2
GPIO3	General-purpose I/O 3
GPIO4	General-purpose I/O 4
GPIO5	General-purpose I/O 5
GPIO6	General-purpose I/O 6

Note: A unique port is used for communication between the L3 interconnect and the L4-Core interconnect to allow the L3 initiators to access the L4-Per targets.

For the list of initiators authorized to access the L4-Per peripherals, see Table 5-14. For details on restricted access, see Section 5.3.3.3.1, *Protection Mechanism*.

5.3.1.3 L4-Emu Interconnect

The L4-Emu interconnect handles only transfers to peripherals in the EMU power domain. Table 5-106 lists the TAs.

Table 5-106. L4-Emu Target Agents

Module Name	Description
L4-Wakeup	L4-Wakeup interconnect
SDTI	System debug trace interface
ETB	Embedded trace buffer
TPIU	Trace port interface unit
MPU	ARM9
DAPCTL	Debug access port

Not all initiators can access all the targets in the L4-Emu interconnect. Additional restrictions affect the ability of these initiators to access the L4-Emu peripherals. Table 5-107 lists which initiators can access the L4-Emu interconnect.

Note: For the list of initiators authorized to access the L4-Emu peripherals, see Table 5-14. For details on restricted access, see Section 5.3.3.3.1, *Protection Mechanism*.

Table 5-107. L4-Emu Initiator Agents

Module Name	Description
L3 interconnect	L3 interconnect port
DAP	DAP port

5.3.1.4 L4-Wakeup Interconnect

The L4-Wakeup interconnect handles only transfers to peripherals in the WKUP power domain. Table 5-108 lists the TAs.

Table 5-108. L4-Wakeup Target Agents

Module Name	Description
PRM	Power reset manager
GPIO1	General-purpose I/O 1
GPTIMER1	General-purpose timer 1
WDTIMER2	MPU subsystem watchdog timer
32KTIMER	32-kHz timer

Initiators that can access the L4-Core or L4-Emu can access all the targets in the L4-Wakeup interconnect. Table 5-109 lists the initiators that can access the L4-Wakeup interconnect. For details on restricted access, see Section 5.3.3.3.1, *Protection Mechanism*.

Table 5-109. L4-Wakeup Initiator Agents

Module Name	Description
L4-Core interconnect	L4-Core interconnect port
L4-Emu interconnect	L4-Emu interconnect port

5.3.2 L4 Interconnects Integration

5.3.2.1 Clocking, Reset, and Power-Management Scheme

5.3.2.1.1 Clocks

Four functional clocks are used in each L4 interconnect (see Table 5-110).

Table 5-110. L4 Interconnect Clocks

Clock	Frequency	Name	Comments
L4-Core interconnect clock	Up to Core_L3_ICLK/2	CORE_L4_GICLK	Source, control, and gating handled by PRCM module
L4-Per interconnect clock	Up to Core_L3_ICLK/2	PER_L4_GICLK	Source, control, and gating handled by PRCM module
L4-Emu clock		L4_EMU	Clock for emulation
L4-Wakeup interconnect clock		WKUP_L4_GICLK	Source, control, and gating handled by PRCM module

5.3.2.1.2 Resets

5.3.2.1.2.1 Hardware Reset

L4 interconnects receive a reset signal from the PRCM module, which is the reset signal to the CORE power domain. For more details, see Chapter 4, *Power, Reset, and Clock Management*.

Table 5-111 lists the hardware reset for the L4-Core interconnect.

Table 5-111. L4 Interconnect Hardware Reset

Interconnect	Reset Domain
L4-Core interconnect	CORE_RST
L4-Per interconnect	PER_RST
L4-Wakeup interconnect	WKUP_RST
L4-Emu interconnect	EMU_RST

5.3.2.1.2.2 Software Reset

The L4 interconnects have hardware reset capabilities, but do not have software reset capabilities. The hardware reset capabilities are controlled by the PRCM module and are applied to the L4 interconnects and the connected L4 TAs and L4 target modules.

5.3.2.1.3 Power Domain

For more details on power voltage scaling, see Chapter 4, *Power, Reset, and Clock Management*.

Table 5-112 lists the power domains for the L4-Core and L4-Wakeup interconnects.

Table 5-112. L4 Interconnect Power Domains

Interconnect	Power Domain
L4-Core interconnect	CORE
L4-Per interconnect	PER
L4-Emu interconnect	EMU
L4-Wakeup interconnect	WKUP

5.3.2.1.4 Power Management

5.3.2.1.4.1 Module Power-Saving

The L4 interconnect automatically performs internal clock autogating to reduce power consumption. Though not recommended, it is possible to deactivate clock autogating by writing 1 to the CLOCK_GATE_DISABLE bit L4_LA_NETWORK_CONTROL_H[24] of each L4 interconnect. Clock autogating is enabled by default.

5.3.2.1.4.2 System Power Management and Wakeup

As part of the system-wide power-management scheme, the L4 interconnect enters an idle state at the request of the PRCM module (for more information, see Chapter 4, *Power, Reset, and Clock Management*). The L4 interconnect is always in smart-idle mode; that is, it goes into idle state after receiving the request from the PRCM module once all the transfer requests are complete. This functionality is handled by hardware. The L4 interconnect sends an acknowledge signal back to the PRCM module when it enters idle state.

5.3.3 L4 Interconnects Functional Description

5.3.3.1 L4-Interconnects Initiator Identification

In the device interconnect, a ConnID is an initiator module identifier. The L4 interconnect uses the same ConnID as L3.

5.3.3.2 Endianness Management

Both L4 interconnects are little-endian only. Any initiator accessing the L4 interconnect module must consider byte ordering and perform a conversion, if necessary.

5.3.3.3 L4 Protection and Firewalls

5.3.3.3.1 Protection Mechanism

The following two parameters are used to set up access permission because of the large address spaces and the number of peripherals connected to the L4 interconnects:

- Programmable groups for initiators:
  - 8 protection groups for the L4-Core interconnect
  - 8 protection groups for the L4-Per interconnect
  - 6 protection groups for the L4-Emu interconnect
- Each segment is divided into regions of 2K bytes:
  - 100 regions for the L4-Core interconnect
  - 43 regions for the L4-Per interconnect
  - 26 regions for the L4-Emu interconnect

Note: Regions and segments are present for the L4-Wakeup interconnect but cannot be programmed. The L4-Wakeup protection is done through the L4-Core and L4-Emu interconnects.

A protection group is a group of targets that have the same protection settings. Initiator access is defined by CONNID_BIT_VECTOR field L4_AP_PROT_GROUP_k_L[15:0]. The initiators have the same access permission to all regions in this group.

A region is programmed to allow access to a unique selectable protection group by using PROT_GROUP_ID field L4_AP_REGION_I_H[22:20].

Note: k denotes the protection group number.

l denotes the region number.

5.3.3.3.2 Protection Group

A protection group defines which initiators with a given MReqInfo can access the targets agent protected by this group.

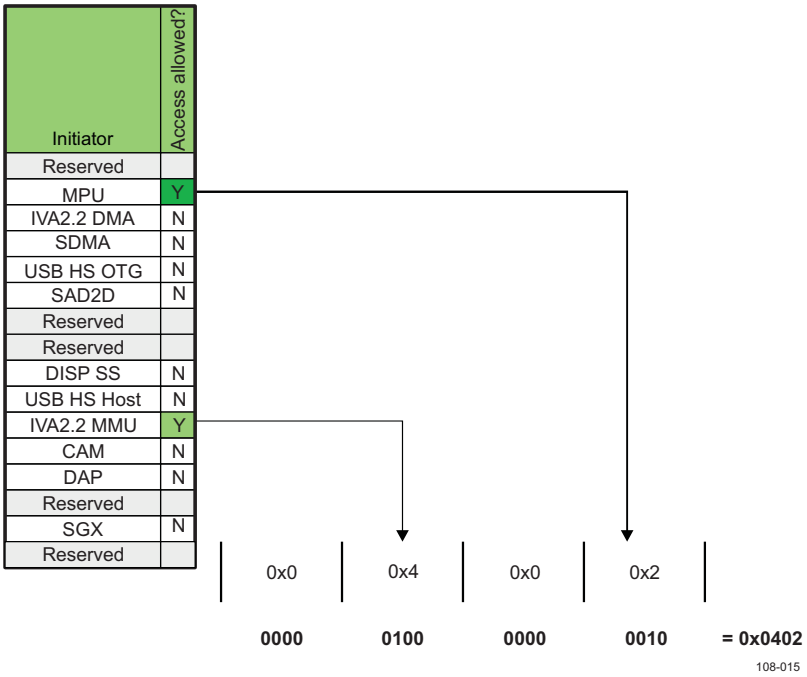
The CONNID_BIT_VECTOR field L4_AP_PROT_GROUP_MEMBERS_k[15:0] (see Figure 5-15) is a 1-bit vector that sets up initiator permission access regions. A protection group is accessible by an initiator if the bit position corresponding to its ConnID is set to one in the CONNID_BIT_VECTOR field.

The ENABLE field L4_AP_PROT_GROUP_ROLES_k[31:0] lists all possible MReqInfo combinations. Setting a Req bit in this register determines the type of access allowed to the initiator. See Section 5.2.3.3.4, *REQ_INFO_PERMISSION Configuration*, for more information. MReqInfo is used in L4 the same way it is used in the L3 firewall configuration.

Note: Permissions are identical for read and write accesses in L4 interconnect targets.

Figure 5-15 shows an example of CONNID_BIT_VECTOR.

Figure 5-15. Example of CONNID_BIT_VECTOR



Setting bits 1 and 4 in the PROT_GROUP_ID_1 defines a group initiator able to access targets in protection group 1, and includes:

- MPU SS
- IVA2.2 MMU

Protection group 1 (PG1) can be applied to multiple protection regions without limitation. Each protection region l that is configured with PG1 enables permission access to these two initiators only.

The firewall default configuration for the L4-Core interconnect contains eight protection groups:

- Most regions are, by default, set with protection group 7 (PG7), which is configured for all access (see Table 5-113).

- Protection group 0 (PG0) is restricted to the MPU subsystem.
- The LCD is attached to protection group 2 (PG2).
- The Camera is attached to protection group 3 (PG3).
- By default, PG5 to PG7 are configured for all access.

The L4-Per interconnect contains eight protection groups:

- By default, PG0 to PG7 are configured for all access.
- By default, most regions are set with PG7, which is configured for all access (see Table 5-114).

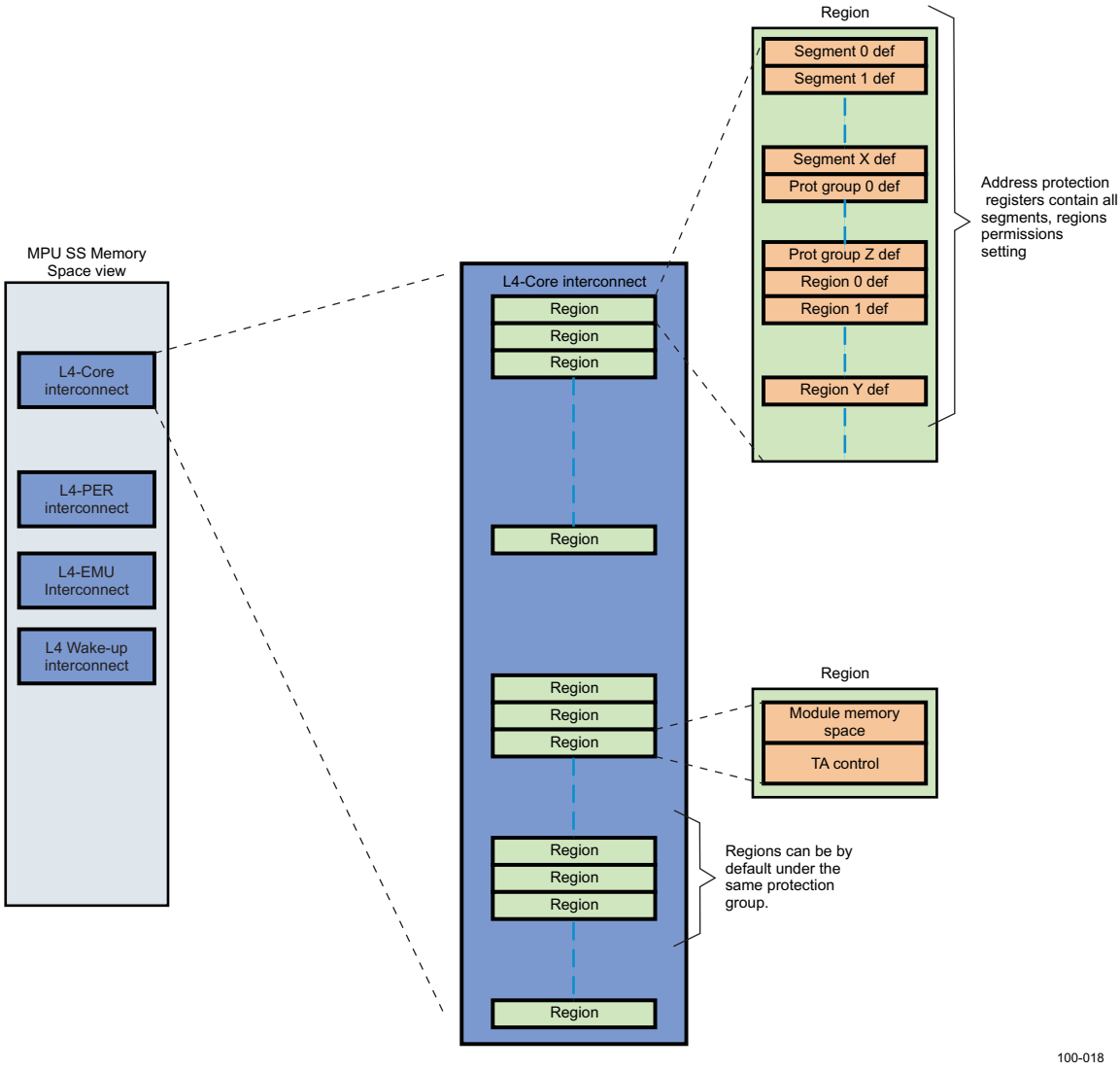
The L4-Emu interconnect contains six protection groups:

- By default, most regions are set with PG5, which is configured for all access (see Table 5-115).
- The AP is attached to PG0.
- The emulation organs are attached to PG3.

5.3.3.3.3 Segments and Regions

The protection mechanism for L4 interconnects is based on a hierarchical segmentation (see Figure 5-16). By default, some regions are attached to a specific protection group. This specificity allows the user to set up the permission access for certain types of modules that require the same access protection without managing the region allocation.

Figure 5-16. L4 Firewall Overview



All interconnect address spaces are covered by regions. Table 5-113to Table 5-115 list the module mapping, including the address, region number, and default protection group allocated to it.

By setting ENABLE bit L4_AP_REGION_y_H[0] to 0, the region becomes inactive and therefore inaccessible. Setting the bit to 1 activates the region.

Table 5-113. Region Allocation for L4-Core Interconnect

Device Name	Start Address (Hex)	Description	Region Number	Default Protection Group
Reserved	0x4800 0000	Reserved		
System control module	0x4800 2000	Module	73	7
	0x4800 3000	L4 interconnect	75	7
Clock manager	0x4800 4000	Module region A	66	7
	0x4800 6000	Module region B	72	7
	0x4800 6800	Reserved		
	0x4800 7000	L4 interconnect	67	7

Table 5-113. Region Allocation for L4-Core Interconnect (continued)

Device Name	Start Address (Hex)	Description	Region Number	Default Protection Group
Reserved	0x4800 8000 0x4802 4000 0x4802 5000 0x4802 6000	Reserved		
L4-Core configuration	0x4804 0000	Address protection (AP)	0	7
	0x4804 0800	Initiator port (IP)	1	7
	0x4804 1000	Link agent (LA)	2	0
Reserved	0x4804 2000	Reserved		
Display subsystem	0x4804 FC00	DSI	104	2
	0x4805 0400	Display subsystem top	4	2
	0x4805 0400	Display controller	4	2
	0x4805 0800	RFBI	5	2
	0x4805 0C00	Video encoder	6	2
	0x4805 1000	L4 interconnect	7	2
Reserved	0x4805 2000	Reserved		
sDMA	0x4805 6000	Module	9	7
	0x4805 7000	L4 interconnect	10	7
Reserved	0x4805 8000 0x4805 9000 0x4805 A000 0x4805 B000 0x4805 C000	Reserved		
Reserved	0x4805 D000	Reserved		
I2C3	0x4806 0000	Module	73	7
	0x4806 1000	L4 interconnect	74	7
USBTLL	0x4806 2000	Module	100	7
	0x4806 3000	L4 interconnect	101	7
USBHS Host	0x4806 4000	Module	15	7
	0x4806 5000	L4 interconnect	16	7
UART1	0x4806 A000	Module	17	7
	0x4806 B000	L4 interconnect	18	7
UART2	0x4806 C000	Module	19	7
	0x4806 D000	L4 interconnect	20	7
Reserved	0x4806 E000	Reserved		
I2C1	0x4807 0000	Module	21	7
	0x4807 1000	L4 interconnect	22	7
I2C2	0x4807 2000	Module	23	7
	0x4807 3000	L4 interconnect	24	7
McBSP1	0x4807 4000	Module	25	7
(digital base band data)	0x4807 5000	L4 interconnect	26	7
Reserved	0x4807 6000	Reserved		
GPTIMER10	0x4808 6000	Module	27	7
	0x4808 7000	L4 interconnect	28	7
GPTIMER11	0x4808 8000	Module	29	7
	0x4808 9000	L4 interconnect	30	7

Table 5-113. Region Allocation for L4-Core Interconnect (continued)

Device Name	Start Address (Hex)	Description	Region Number	Default Protection Group
Reserved	0x4808 A000	Reserved		
	0x4808 B000			
	0x4808 C000			
MAILBOX	0x4809 4000	Module	33	7
	0x4809 5000	L4 interconnect	34	7
McBSP5	0x4809 6000	Module	59	7
(MIDI data)	0x4809 7000	L4 interconnect	60	7
SPI1	0x4809 8000	Module	35	7
	0x4809 9000	L4 interconnect	36	7
SPI2	0x4809 A000	Module	37	7
	0x4809 B000	L4 interconnect	38	7
MMC/SD/SDIO1	0x4809 C000	Module	39	7
	0x4809 D000	L4 interconnect	40	7
Reserved	0x4809 E000	Reserved		
	0x4809 F000			
USBHS OTG	0x480A B000	Module	13	7
	0x480A C000	L4 interconnect	14	7
MMC/SD/SDIO3	0x480A D000	Module	98	7
	0x480A E000	L4 interconnect	99	7
Reserved	0x480B 0000	Reserved		
	0x480B 1000			
HDQ/1-Wire	0x480B 2000	Module	57	7
	0x480B 3000	L4 interconnect	58	7
MMC/SD/SDIO2	0x480B 4000	Module	41	7
	0x480B 5000	L4 interconnect	42	7
ICR	0x480B 6000	Module	88	7
	0x480B 7000	L4 interconnect	89	7
SPI3	0x480B 8000	Module	66	7
	0x480B 9000	L4 interconnect	67	7
SPI4	0x480B A000	Module	77	7
	0x480B B000	L4 interconnect	78	7
Camera ISP	0x480B C000	Camera ISP	8	3
	0x480C 0000	L4 interconnect	75	3
MODEM INTC	0x480C 7000	Module	90	7
	0x480C 8000	L4 interconnect	91	7
ICR	0x480C D000	Module	86	7
	0x480C E000	L4 interconnect	87	7
Reserved	0x480C F000	Reserved		
MPU INTC	0x4820 0000	Nonshared device mapping		
Reserved	0x4820 1000	Reserved		
Reserved	0x4828 1000	Reserved		

Table 5-113. Region Allocation for L4-Core Interconnect (continued)

Device Name	Start Address (Hex)	Description	Region Number	Default Protection Group
L4-Wakeup interconnect	0x4830 0000	L4 wakeup	76	7
	0x4830 6000	L4 wakeup	92	7
	0x4830 8000	L4 wakeup	93	7
	0x4830 9000	L4 wakeup	94	7
	0x4830 C000	L4 wakeup	95	7
	0x4831 0000	L4 wakeup	96	7
	0x4832 0000	L4 wakeup	97	7
	0x4834 0000	L4 wakeup	102	7
Reserved	0x4834 1000	Reserved		

Table 5-114. Region Allocation for L4-Per Interconnect

Device Name	Start Address (hex)	Description	Region Number	Default Protection Group
L4-Per configuration	0x4900 0000	Address protection (AP)	0	0
	0x4900 0800	Initiator port (IP)	1	7
	0x4900 1000	Link agent (LA)	2	7
Reserved	0x4900 2000	Reserved		
UART3	0x4902 0000	Module	3	7
(Infrared)	0x4902 1000	L4 interconnect	4	7
McBSP2	0x4902 2000	Module	5	7
(Audio for codec)	0x4902 3000	L4 interconnect	6	7
McBSP3	0x4902 4000	Module	7	7
(Bluetooth voice data)	0x4902 5000	L4 interconnect	8	7
McBSP4 (digital base band voice data)	0x4902 6000	Module	9	7
	0x4902 7000	L4 interconnect	10	7
McBSP2 (Sidetone)	0x4902 8000	Module	39	7
	0x4902 9000	L4 interconnect	40	7
McBSP3 (Sidetone)	0x4902 A000	Module	41	7
	0x4902 B000	L4 interconnect	42	7
Reserved	0x4902 C000	Reserved		
WDTIMER3	0x4903 0000	Module	11	7
	0x4903 1000	L4 interconnect	12	7
GPTIMER2	0x4903 2000	Module	13	7
	0x4903 3000	L4 interconnect	14	7
GPTIMER3	0x4903 4000	Module	15	7
	0x4903 5000	L4 interconnect	16	7
GPTIMER4	0x4903 6000	Module	17	7
	0x4903 7000	L4 interconnect	18	7
GPTIMER5	0x4903 8000	Module	19	7
	0x4903 9000	L4 interconnect	20	7
GPTIMER6	0x4903 A000	Module	21	7
	0x4903 B000	L4 interconnect	22	7
GPTIMER7	0x4903 C000	Module	23	7
	0x4903 D000	L4 interconnect	24	7

Table 5-114. Region Allocation for L4-Per Interconnect (continued)

Device Name	Start Address (hex)	Description	Region Number	Default Protection Group
GPTIMER8	0x4903 E000	Module	25	7
	0x4903 F000	L4 interconnect	26	7
GPTIMER9	0x4904 0000	Module	27	7
	0x4904 1000	L4 interconnect	28	7
Reserved	0x4904 2000	Reserved		
GPIO2	0x4905 0000	Module	29	7
	0x4905 1000	L4 interconnect	30	7
GPIO3	0x4905 2000	Module	31	7
	0x4905 3000	L4 interconnect	32	7
GPIO4	0x4905 4000	Module	33	7
	0x4905 5000	L4 interconnect	34	7
GPIO5	0x4905 6000	Module	35	7
	0x4905 7000	L4 interconnect	36	7
GPIO6	0x4905 8000	Module	37	7
	0x4905 9000	L4 interconnect	38	7
Reserved	0x4905 A000	Reserved		

Table 5-115. Region Allocation for L4-Emu Interconnect

Device Name	Start Address (hex)	Description	Region Number	Default Protection Group
Reserved	0x5400 0000	Reserved		
TEST–Chip-level TAP	0x5400 4000	Module	1	5
	0x5400 5000	L4 interconnect	2	5
L4-Emu configuration	0x5400 6000	Address protection (AP)	3	0
	0x5400 6800	Initiator port (IP) L4-Core	4	5
	0x5400 7000	Link agent (LA)	5	5
	0x5400 8000	Initiator port (IP) DAP	6	5
Reserved	0x5400 8800	Reserved		
MPU SS (emulation, trace, and debug)	0x5401 0000	Module	16	3
	0x5401 8400	L4 interconnect	7	3
TPIU	0x5401 9000	Module	8	3
	0x5401 A000	L4 interconnect	9	3
ETB	0x5401 B000	Module	10	3
	0x5401 C000	L4 interconnect	11	3
DAPCTL	0x5401 D000	Module	12	3
	0x5401 E000	L4 interconnect	13	3
SDTI	0x5401 F000	L4 interconnect	15	5
	0x5402 0000	Reserved		
	0x5450 0000	SDTI module (configuration)	14	5
	0x5451 0000	Reserved		
	0x5460 0000	SDTI module (window)	0	5
Reserved	0x5470 0000	Reserved		
Power and reset manager • Power manager • Reset manager	0x5470 6000	Module region A	17	5
	0x5470 8000	Module region B	18	5
	0x5470 8800	Reserved	20	5

Table 5-115. Region Allocation for L4-Emu Interconnect (continued)

Device Name	Start Address (hex)	Description	Region Number	Default Protection Group
(WKUP power domain)	0x5470 9000	L4 interconnect	21	5
Reserved	0x5470 A000		Reserved	
Reserved	0x5470 E000		Reserved	
GPIO1 (WKUP power domain)	0x5471 0000	Module	23	5
	0x5471 1000	L4 interconnect		
Reserved	0x5471 2000	Reserved		
WDTIMER2 (WKUP power domain)	0x5471 4000	Module		
	0x5471 5000	L4 interconnect		
Reserved	0x5471 6000	Reserved		
GPTIMER1 (WKUP power domain)	0x5471 8000	Module		
	0x5471 9000	L4 interconnect		
Reserved	0x5471 A000		Reserved	
32KTIMER (WKUP power domain)	0x5472 0000	Module	24	5
	0x5472 1000	L4 interconnect		
Reserved	0x5472 2000	Reserved		
L4-Wakeup configuration (WKUP power domain)	0x5472 8000	Address protection (AP)		
	0x5472 8800	Initiator port (IP) L4-Core		
	0x5472 9000	Link agent (LA)		
	0x5472 A000	Initiator port (IP) L4-Emu		
Reserved	0x5472 A800	Reserved		
L4-Wakeup	0x5473 0000	L4 interconnect	25	5

5.3.3.3.4 L4 Firewall Address and Protection Registers Setting

Table 5-116 lists the settings of the AP registers for an L4 interconnect firewall. These values are computed based on the physical implementation of each L4 interconnect.

Table 5-116. L4 Firewall Register Description Overview

Register Type	Register Name	Bits	Field	Description
Segment	L4_AP_SEGMENT_i_L	23:0	Base	Segment base address
	L4_AP_SEGMENT_i_H	4:0	SIZE	Segment size equals to 2 power of SIZE
Protection groups	L4_AP_PROT_GROUP_MEMBERS_k_L	15:0	CONNID_BIT_VECTOR	See Section 5.3.3.1 for L4ConnID).
	L4_AP_PROT_GROUP_ROLES_k_L	15:0	ENABLE	Refer to Table 5-22 for MReq description
Region setting	L4_AP_REGION_I_L	23:0	BASE	Define the base address of region in respect to its respective segment base address
	L4_AP_REGION_I_H	27:24	SEGMENT_ID	Segment ID number of the region
		22:20	PROT_GROUP_ID	The protection group attached to the region
		19:17	BYTE_DATA_WIDTH_EXP	Determine the number of bytes in an access
		5:1	SIZE	Size of the region equals to 2 power of SIZE
		0	ENABLE	Enable the region protection

5.3.3.4 Error Handling

5.3.3.4.1 Overview

The L4 interconnect provides mechanisms for handling either internally detected errors or errors reported by modules attached to the L4 target ports. Hardware support facilitates logging errors and cleaning up the state to allow error recovery software to treat the error.

As an L3 target, the L4 interconnect reports errors to the L3 interconnect in-band whenever possible. In-band error reporting is the default and recommended configuration. It is assumed that INBAND_ERROR_REP bit L4_IA_AGENT_CONTROL_L[27] is set to 1.

Note: *L4_IA* denotes which interconnect is considered: L4-Core, L4-Per, L4-Emu, or L4-Wakeup.

L4_TA denotes the module name, such as UART1, McBSP1, etc.

The L4 interconnects handle three types of errors:

- No target core found or address hole
- Request protection violation
- Failure of the target to service a request before a time-out expires

5.3.3.4.2 Error Logging

5.3.3.4.2.1 No Target Core Found/Address Hole

This error indicates that a request was addressed to a hole in the L4 address map.

When this occurs, an in-band error response is returned to the L3 level.

The error is also logged into the INBAND_ERROR bit L4_IA_AGENT_STATUS_L[27].

Additionally, an address hole error code is logged into the CODE field L4_IA_ERROR_LOG_L[25:24].

The L4_IA_ERROR_LOG register also includes MULTI bit L4_IA_ERROR_LOG_L[31], which is asserted when multiple errors are detected. In this case, the error code corresponds to the first error that occurs.

5.3.3.4.2.2 Protection Violation

This error indicates that an initiator has accessed a restricted region. This error is reported using an in-band error. It is written to the INBAND_ERROR field. A protection violation error code is also saved in the CODE field L4_IA_ERROR_LOG_L[25:24].

The protection violation is also logged in the CONTROL.CONTROL_PROT_ERR_STATUS [7] register of the system control module.

Note: There is no specific error signal for the protection violation that goes out from the L4 interconnects. The out-band error that goes to the system control module indicates a protection violation, but it also indicates normal interconnect errors.

5.3.3.4.2.3 Time-Out

This section gives information about all modules and features in the high-tier device. See Section 1.5, *OMAP34xx Family*, to check availability of modules and features. To flag interconnect response being blocked, the time-out of target agents attached to unavailable modules can be enabled with the lowest setting.

A time-out mechanism can be enabled on a per-target basis in the L4_TA_AGENT_CONTROL_L register. If the mechanism is enabled for a TA, and commands are not accepted or responses are not returned within the expected delay, the L4 interconnect generates an error event.

The error is logged in the target agent REQ_TIMEOUT bit L4_TA_AGENT_STATUS_L[8]. The affected TA enters an error state that causes it to send error responses to any new request targeted at it. To recover from this state, the TA must be reset by system software. The time-out is counted starting from the moment a command is presented to the target, regardless of how the target responds to the command.

The L4 interconnect implements a centralized time-base circuit that broadcasts a set of four periodic pulse signals to all connected TAs. These four signals are referred to as 1X-time base, 4X-time base, 16X-time base, and 64X-time base.

The time-base circuit offers four possible sets of time-base signals. Selection is done by programming TIMEOUT_BASE field L4_LA_NETWORK_CONTROL_L[10:8]. Table 5-117 lists the values in the number of L4 clock cycles.

Table 5-117. L4 Time-Out Link and TA Programming

	REQ_TIMEOUT[2:0]				
TIMEOUT_BASE[2:0]	**0**	**1**	**2**	**3**	**4**
0	All L4 time-out features are disabled.				
1	Locally disabled	64	256	1024	4096
2		256	1024	4096	16384
3		1024	4096	16384	65536
4		4096	16384	65536	262144

The reset value is 0x4, resulting in the longest possible time-out.

The selected time-base signals are available at any TA. Each TA can be programmed to refer to one of these four time-base signals, by using REQ_TIMEOUT field L4_TA_AGENT_CONTROL_L[10:8] (see Table 5-118).

Table 5-118. L4 Time-Out TA Programming

REQ_TIMEOUT	Target Agent Time-Out Reference
0	Time-out locally disabled
1	1X-base cycle
2	4X-base cycle
3	16X-base cycle
4	64X-base cycle

A time-out condition is detected when the command acceptance or the response is not received after a delay of between one and three time-base periods.

Example:

- L4 frequency = 100 MHz
- TIMEOUT_BASE = 4 in the L4_LA_NETWORK_CONTROL_L register
- REQ_TIMEOUT = 2 in the L4_TA_AGENT_CONTROL_L for target agent A
- REQ_TIMEOUT = 4 in the L4_TA_AGENT_CONTROL_L for target agent B

At agent A, the time-base unit is 16384 cycles. A time-out is issued when a request to the attached module is not accepted or no response is sent after a delay of 164 µs to 492 µs.

At agent B, the time-base unit is 262,144 cycles. A time-out is issued when a request to the attached module is not accepted or no response is sent after a delay of 2.6 ms to 7.8 ms.

On detection of a time-out condition, the agent automatically generates an error response to the inter_IA, which is forwarded to the L3. The agent also logs the time-out to REQ_TIMEOUT bit L4_TA_AGENT_STATUS_L[8].

After the time-out has been detected and logged, the behavior of the attached module is ignored. A new request targeting the module arriving at the timed out TA receives an error response. If the request is addressed to the agent internal registers, it is processed normally.

To recover from a time-out error, the software is assumed to reset first the faulty module by using its internal soft reset bit, and then the TA by using OCP_RESET bit L4_TA_AGENT_CONTROL_L[0].

5.3.3.4.3 TA Software Reset

Writing 1 to OCP_RESET bit L4_TA_AGENT_CONTROL_L[0] initiates the software reset period. The software reset must be asserted for at least 16 cycles of the target module OCP clock, which can be a divided clock with respect to the L4 clock.

During the software reset period the following occur:

- Requests sent to the target module receive error responses. Therefore, if the faulty request is part of a DMA transfer, it is necessary to stop the DMA to avoid getting unwanted errors.
- Requests sent to the TA register block are processed as usual.
- The L4_TA_AGENT_STATUS_L[8] REQ_TIMEOUT status bit is cleared.

Writing 0 to the L4_TA_AGENT_CONTROL_L[0] OCP_RESET bit terminates the software reset period.

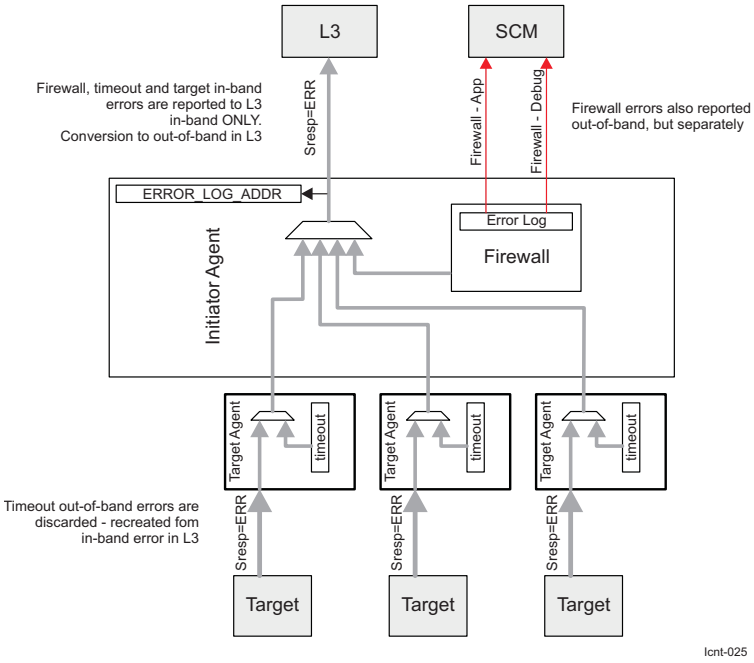
The attached module must then be reset to complete the recovery.

5.3.3.4.4 Error Reporting

Figure 5-17 illustrates the error-reporting scheme used in the L4_Core, L4_Per, and L4_EMU interconnects.

- Timeout error generation is enabled at each TA.
- Timeout error and target in-band errors are only reported in-band to the L3 initiator.
- Protection violations are reported out-of-band. Error steering is used to distinguish between application or debug errors.

Figure 5-17. L4 Error Reporting



5.3.4 L4 Interconnects Register Manual

A summary of the hardware interface for the L4-Core, L4-Per, L4-Emu and L4-Wkup interconnects is given in Table 5-119 to Table 5-122. Each module instance in the design is shown with the module register map and bit definitions for each bit field.

Table 5-119. L4- Core Instance Summary

Module Name	Base Address	Size
CORE_TA_CONTROL	0x4800 3000	4K bytes
CORE_TA_CM	0x4800 7000	4K bytes
CORE_AP	0x4804 0000	2K bytes
CORE_IA	0x4804 0800	2K bytes
CORE_LA	0x4804 1000	4K bytes
CORE_TA_DISPLAY_SS	0x4805 1000	4K bytes
CORE_TA_SDMA	0x4805 7000	4K bytes
CORE_TA_I2C3	0x4806 1000	4K bytes
CORE_TA_USB_HS_TLL	0x4806 3000	4K bytes
CORE_TA_USB_HS_Host	0x4806 5000	4K bytes
CORE_TA_UART1	0x4806 B000	4K bytes
CORE_TA_UART2	0x4806 D000	4K bytes
CORE_TA_I2C1	0x4807 1000	4K bytes
CORE_TA_I2C2	0x4807 3000	4K bytes
CORE_TA_MCBSP1	0x4807 5000	4K bytes
CORE_TA_GPTIMER10	0x4808 7000	4K bytes
CORE_TA_GPTIMER11	0x4808 9000	4K bytes
CORE_TA_MAILBOX	0x4809 5000	4K bytes
CORE_TA_MCBSP5	0x4809 7000	4K bytes
CORE_TA_MCSPI1	0x4809 9000	4K bytes
CORE_TA_MCSPI2	0x4809 B000	4K bytes
CORE_TA_MMCHS1	0x4809 D000	4K bytes
CORE_TA_USB_HS_OTG	0x480A C000	4K bytes
CORE_TA_MMCHS3	0x480A E000	4K bytes
CORE_TA_HDQ1W	0x480B 3000	4K bytes
CORE_TA_MMCHS2	0x480B 5000	4K bytes
CORE_TA_ICR	0x480B 7000	4K bytes
CORE_TA_MCSPI3	0x480B 9000	4K bytes
CORE_TA_MCSPI4	0x480B B000	4K bytes
CORE_TA_CAMERA	0x480C 0000	4K bytes
CORE_TA_INTH	0x480C 8000	4K bytes
CORE_TA_ICR_MODEM	0x480C E000	4K bytes
CORE_TA_WKUP	0x4834 0000	4K bytes

Table 5-120. L4-Per Instance Summary

Module Name	Base Address	Size
PER_AP	0x4900 0000	2K bytes
PER_IA	0x4900 0800	2K bytes
PER_LA	0x4900 1000	4K bytes
PER_TA_UART3	0x4902 1000	512 bytes
PER_TA_MCBSP2	0x4902 3000	1K byte
PER_TA_MCBSP3	0x4902 5000	1K byte
PER_TA_MCBSP4	0x4902 7000	1K byte

Table 5-120. L4-Per Instance Summary (continued)

Module Name	Base Address	Size
PER_TA_MCBSP2_SIDETONE	0x4902 9000	4K bytes
PER_TA_MCBSP3_SIDETONE	0x4902 B000	4K bytes
PER_TA_WDTIMER3	0x4903 1000	2K bytes
PER_TA_GPTIMER2	0x4903 3000	1K byte
PER_TA_GPTIMER3	0x4903 5000	1K byte
PER_TA_GPTIMER4	0x4903 7000	1K byte
PER_TA_GPTIMER5	0x4903 9000	1K byte
PER_TA_GPTIMER6	0x4903 B000	1K byte
PER_TA_GPTIMER7	0x4903 D000	1K byte
PER_TA_GPTIMER8	0x4903 F000	1K byte
PER_TA_GPTIMER9	0x4904 1000	1K byte
PER_TA_GPIO2	0x4905 1000	1K byte
PER_TA_GPIO3	0x4905 3000	1K byte
PER_TA_GPIO4	0x4905 5000	1K byte
PER_TA_GPIO5	0x4905 7000	1K byte
PER_TA_GPIO6	0x4905 9000	1K byte

Table 5-121. L4-Emu Instance Summary

Module Name	Base Address	Size
EMU_TA_TEST_TAP	0x5400 5000	4K bytes
EMU_AP	0x5400 6000	2K bytes
EMU_IA_L3	0x5400 6800	2K bytes
EMU_LA	0x5400 7000	4K bytes
EMU_IA_DAP	0x5400 8000	2K bytes
EMU_TA_MPU	0x5401 8000	4K bytes
EMU_TA_TPIU	0x5401 A000	4K bytes
EMU_TA_ETB	0x5401 C000	4K bytes
EMU_TA_DAPCTL	0x5401 E000	4K bytes
EMU_TA_SDTI	0x5401 F000	4K bytes
EMU_TA_L4WKUP	0x5473 0000	4K bytes

Table 5-122. L4-WKUP Instance Summary

Module Name	Base Address	Size
WKUP_TA_PRM	0x4830 9000	4K bytes
WKUP_TA_GPIO1	0x4831 1000	4K bytes
WKUP_TA_WDTIMER2	0x4831 5000	4K bytes
WKUP_TA_GPTIMER1	0x4831 9000	4K bytes
WKUP_TA_SYNCTIMER32K	0x4832 1000	4K bytes
WKUP_AP	0x4832 8000	2K bytes
WKUP_IA_CORE	0x4832 8800	2K bytes
WKUP_LA	0x4832 9000	4K bytes
WKUP_IA_EMU	0x4832 A000	2K bytes

5.3.4.1 L4 Iniator Agent (L4 IA)

This section provides information on the L4 IA module. Each of the registers within the module instance is described seperatly in Table 5-123 to Table 5-124.

The initiator OCP interface register block (IA) consists of the status, control, and error log registers that can be used to configure the interface of an initiator OCP. There can be only one register block for each initiator OCP interface.

Table 5-123. L4 IA Register Summary (1)

Register Name	Type	Register Width (Bits)	CORE_IA Physical Address	PER_IA Physical Address	EMU_IA_L3 Physical Address
L4_IA_COMPONENT_L	R	32	0x4804 0800	0x4900 0800	0x5400 6800
L4_IA_COMPONENT_H	R	32	0x4804 0804	0x4900 0804	0x5400 6804
L4_IA_CORE_L	RW	32	0x4804 0818	0x4900 0818	0x5400 6818
L4_IA_CORE_H	RW	32	0x4804 081C	0x4900 081C	0x5400 681C
L4_IA_AGENT_CONTROL_L	RW	32	0x4804 0820	0x4900 0820	0x5400 6820
L4_IA_AGENT_CONTROL_H	RW	32	0x4804 0824	0x4900 0824	0x5400 6824
L4_IA_AGENT_STATUS_L	RW	32	0x4804 0828	0x4900 0828	0x5400 6828
L4_IA_AGENT_STATUS_H	RW	32	0x4804 082C	0x4900 082C	0x5400 682C
L4_IA_ERROR_LOG_L	RW	32	0x4804 0858	0x4900 0858	0x5400 6858
L4_IA_ERROR_LOG_H	RW	32	0x4804 085C	0x4900 085C	0x5400 685C

Table 5-124. L4 IA Register Summary (2)

Register Name	Type	Register Width (Bits)	EMU_IA_DAP Physical Address	WKUP_IA_CORE Physical Address	WKUP_IA_EMU Physical Address
L4_IA_COMPONENT_L	R	32	0x5400 8000	0x4832 8800	0x4832 A000
L4_IA_COMPONENT_H	R	32	0x5400 8004	0x4832 8804	0x4832 A004
L4_IA_CORE_L	RW	32	0x5400 8018	0x4832 8818	0x4832 A018
L4_IA_CORE_H	RW	32	0x5400 801C	0x4832 881C	0x4832 A01C
L4_IA_AGENT_CONTROL_L	RW	32	0x5400 8020	0x4832 8820	0x4832 A020
L4_IA_AGENT_CONTROL_H	RW	32	0x5400 8024	0x4832 8824	0x4832 A024
L4_IA_AGENT_STATUS_L	RW	32	0x5400 8028	0x4832 8828	0x4832 A028
L4_IA_AGENT_STATUS_H	RW	32	0x5400 802C	0x4832 882C	0x4832 A02C
L4_IA_ERROR_LOG_L	RW	32	0x5400 8058	0x4832 8858	0x4832 A058
L4_IA_ERROR_LOG_H	RW	32	0x5400 805C	0x4832 885C	0x4832 A05C

5.3.4.1.1 L4 Iniator Agent (L4 IA) Registers Description

Table 5-125. L4_IA_COMPONENT_L

Address Offset	0x000
Physical Address	SeeTable 5-123 to Table 5-124
Description	COMPONENT register identifies the component to which this register block belongs. The register contains a component code and revision, which are used to identify the hardware of the component. The COMPONENT register is read-only.
Type	R

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CODE																REV															

Bits	Field Name	Description	Type	Reset
31:16	CODE	Interconnect code	R	See (1).
15:0	REV	Component revision code	R	See (1).

(1) TI Internal data

Table 5-126. Register Call Summary for Register L4_IA_COMPONENT_L

L4 Interconnects

- L4 Iniator Agent (L4 IA) Registers Manual: [0] [1]

Table 5-127. L4_IA_COMPONENT_H

Address Offset	0x004
Physical Address	SeeTable 5-123 to Table 5-124
Description	COMPONENT register identifies the component to which this register block belongs. The register contains a component code and revision, which are used to identify the hardware of the component. The COMPONENT register is read-only.
Type	R

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																VENDOR_CODE															

Bits	Field Name	Description	Type	Reset
31:16	Reserved	Reserved	R	0x0000
15:0	VENDOR_CODE	Vendor revision code	R	See (1).

(1) TI Internal data

Table 5-128. Register Call Summary for Register L4_IA_COMPONENT_H

L4 Interconnects

- L4 Iniator Agent (L4 IA) Registers Manual: [0] [1]

Table 5-129. L4_IA_CORE_L

Address Offset	0x018
Physical Address	SeeTable 5-123 to Table 5-124
Description	Provide information about the core initiator
Type	RW

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CORE_CODE																CORE_REV															

Bits	Field Name	Description	Type	Reset
31:16	CORE_CODE	Interconnect core code	R	See (1).
15:0	CORE_REV	Component revision code code	R	See (1).

(1) TI Internal data

Table 5-130. Register Call Summary for Register L4_IA_CORE_L

L4 Interconnects

- L4 Iniator Agent (L4 IA) Registers Manual: [0] [1]

Table 5-131. L4_IA_CORE_H

Address Offset	0x01C
Physical Address	SeeTable 5-123 to Table 5-124
Description	Provide information about the core initiator
Type	RW

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																VENDOR_CODE															

Bits	Field Name	Description	Type	Reset
31:16	Reserved	Reserved	R	0x0000
15:0	VENDOR_CODE	Vendor revision core code	R	See (1).

(1) TI Internal data

Table 5-132. Register Call Summary for Register L4_IA_CORE_H

L4 Interconnects

- L4 Iniator Agent (L4 IA) Registers Manual: [0] [1]

Table 5-133. L4_IA_AGENT_CONTROL_L

Address Offset	0x020
Physical Address	SeeTable 5-123 to Table 5-124
Description	Enable error reporting on an initiator interface.The error reporting mechanism is enabled when the INBAND_ERROR_REP bit field is set to 1. The out-of-band OCP MError reporting mechanism is enabled when the MERROR_REP bit field is set to 1.
Type	RW

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved				INBAND_ERROR_REP	Reserved		MERROR_REP	Reserved																							

Bits	Field Name	Description	Type	Reset
31:28	Reserved	Read returns 0.	R	0x0
27	INBAND_ERROR_REP	Setting this field to 1 reports on in-band errors using the INBAND_ERROR log bit of IA.AGENT_STATUS register.	R	1
26:25	Reserved	Read returns 0.	R	0x0
24	MERROR_REP	Enable MError reporting	R	0x0
23:0	Reserved	Read returns 0.	R	0x0000000

Table 5-134. Register Call Summary for Register L4_IA_AGENT_CONTROL_L

L4 Interconnects

- Error Handling: [0]
- L4 Iniator Agent (L4 IA) Registers Manual: [1] [2]

Table 5-135. L4_IA_AGENT_CONTROL_H

Address Offset	0x024
Physical Address	SeeTable 5-123 to Table 5-124
Description	Enable error reporting on an initiator interface.
Type	RW

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																															

Bits	Field Name	Description	Type	Reset
31:0	Reserved	Read returns 0.	R	0x0000 0000

Table 5-136. Register Call Summary for Register L4_IA_AGENT_CONTROL_H

L4 Interconnects

- L4 Iniator Agent (L4 IA) Registers Manual: [0] [1]

Table 5-137. L4_IA_AGENT_STATUS_L

Address Offset	0x028
Physical Address	SeeTable 5-123 to Table 5-124
Description	Stores status information for an initiator. The INBAND_ERROR and MERROR fields are read/write and are implemented as log bits.
Type	RW

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved				INBAND_ERROR_REP	Reserved		MERROR_REP	Reserved																							

Bits	Field Name	Description	Type	Reset
31:28	Reserved	Read returns 0.	R	0x0
27	INBAND_ERROR_REP	0x0 No In-Band error present. 0x1 In-Band error present.	R 1toClr	0x0
26:25	Reserved	Read returns 0.	R	0x0
24	MERROR_REP	0x0 No MError error present. 0x1 MErorr error present.	R	0x0
23:0	Reserved	Read returns 0.	R	0x0000000

Table 5-138. Register Call Summary for Register L4_IA_AGENT_STATUS_L

L4 Interconnects

- Error Handling: [0]
- L4 Iniator Agent (L4 IA) Registers Manual: [1] [2]

Table 5-139. L4_IA_AGENT_STATUS_H

Address Offset	0x02C
Physical Address	SeeTable 5-123 to Table 5-124
Description	Stores status information for an initiator.
Type	RW

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																															

Bits	Field Name	Description	Type	Reset
31:0	Reserved	Read returns 0.	R	0x0000 0000

Table 5-140. Register Call Summary for Register L4_IA_AGENT_STATUS_H

L4 Interconnects

- L4 Iniator Agent (L4 IA) Registers Manual: [0] [1]

Table 5-141. L4_IA_ERROR_LOG_L

Address Offset	0x058
Physical Address	SeeTable 5-123 to Table 5-124
Description	Log information about error conditions. The CODE field logs any protection violation or address hole errors detected by the initiator subsystem while decoding a request.
Type	RW

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MULTI	Reserved					CODE		Reserved																							

Bits	Field Name	Description	Type	Reset
31	MULTI	While the CODE field is not zero, the MULTI bit is asserted whenever an additional error is detected. Once set by hardware, the MULTI bit can only be cleared by writing a 1 to it while writing a value other than zero to the CODE field.	RW	0
30:26	Reserved	Read returns 0.	R	0x00
25:24	CODE	The error code of an initiator request. 0x00: No errors 0x01: Reserved 0x10: Address hole 0x11: Protection violation The CODE field, once set by hardware, can only be cleared by writing a non-zero value to it, in conjunction with writing a 1 to the MULTI bit field.	RW	0x0
23:0	Reserved	Read returns 0.	R	0x000000

Table 5-142. Register Call Summary for Register L4_IA_ERROR_LOG_L

L4 Interconnects

- Error Handling: [0] [1] [2]
- L4 Iniator Agent (L4 IA) Registers Manual: [3] [4]

Table 5-143. L4_IA_ERROR_LOG_H

Address Offset	0x05C
Physical Address	SeeTable 5-123 to Table 5-124
Description	Log information about error conditions.
Type	RW

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																															

Bits	Field Name	Description	Type	Reset
31:0	Reserved	Read returns 0.	R	0x0000 0000

Table 5-144. Register Call Summary for Register L4_IA_ERROR_LOG_H

L4 Interconnects

- L4 Iniator Agent (L4 IA) Registers Manual: [0] [1]

5.3.4.2 L4 Target Agent (L4 TA)

This section provides information on the L4 Target agent (TA) register module. Each of the registers within the module instance is described seperatly in Table 5-145 to Table 5-168.

Table 5-145 to Table 5-169 provide information on the L4 Target Agent having two supplementary registers OCP_CONTROL and OCP_STATUS

The TA register block consists of the status and control registers that can be used to configure the interfaces of the targets.

Table 5-145. CORE_TA Common Register Summary

Register Name	Type	Register Width (Bits)	CORE_TA_CONTROL Physical Address
L4_TA_COMPONENT_L	R	32	0x4800 3000
L4_TA_COMPONENT_H	R	32	0x4800 3004
L4_TA_CORE_L	R	32	0x4800 3018
L4_TA_CORE_H	R	32	0x4800 301C
L4_TA_AGENT_CONTROL_L	RW	32	0x4800 3020
L4_TA_AGENT_CONTROL_H	RW	32	0x4800 3024
L4_TA_AGENT_STATUS_L	RW	32	0x4800 3028
L4_TA_AGENT_STATUS_H	RW	32	0x4800 302C

Table 5-146. CORE_TA Common Register Summary

Register Name	Type	Register Width (Bits)	CORE_TA_CM Physical Address	CORE_TA_DISPLAY_SS Physical Address	CORE_TA_SDMA Physical Address
L4_TA_COMPONENT_L	R	32	0x4802 7000	0x4805 1000	0x4805 7000
L4_TA_COMPONENT_H	R	32	0x4802 7004	0x4805 1004	0x4805 7004
L4_TA_CORE_L	R	32	0x4802 7018	0x4805 1018	0x4805 7018
L4_TA_CORE_H	R	32	0x4802 701C	0x4805 101C	0x4805 701C
L4_TA_AGENT_CONTROL_L	RW	32	0x4802 7020	0x4805 1020	0x4805 7020
L4_TA_AGENT_CONTROL_H	RW	32	0x4802 7024	0x4805 1024	0x4805 7024
L4_TA_AGENT_STATUS_L	RW	32	0x4802 7028	0x4805 1028	0x4805 7028
L4_TA_AGENT_STATUS_H	RW	32	0x4802 702C	0x4805 102C	0x4805 702C

Table 5-147. CORE_TA Common Register Summary

Register Name	Type	Register Width (Bits)	CORE_TA_I2C3 Physical Address
L4_TA_COMPONENT_L	R	32	0x4806 1000
L4_TA_COMPONENT_H	R	32	0x4806 1004
L4_TA_CORE_L	R	32	0x4806 1018
L4_TA_CORE_H	R	32	0x4806 101C
L4_TA_AGENT_CONTROL_L	RW	32	0x4806 1020
L4_TA_AGENT_CONTROL_H	RW	32	0x4806 1024
L4_TA_AGENT_STATUS_L	RW	32	0x4806 1028
L4_TA_AGENT_STATUS_H	RW	32	0x4806 102C

Table 5-148. CORE_TA Common Register Summary

Register Name	Type	Register Width (Bits)	CORE_TA_USB_TLL Physical Address	CORE_TA_USB_HS_Host Physical Address	CORE_TA_UART1 Physical Address
L4_TA_COMPONENT_L	R	32	0x4806 3000	0x4806 5000	0x4806 B000
L4_TA_COMPONENT_H	R	32	0x4806 3004	0x4806 5004	0x4806 B004
L4_TA_CORE_L	R	32	0x4806 3018	0x4806 5018	0x4806 B018
L4_TA_CORE_H	R	32	0x4806 301C	0x4806 501C	0x4806 B01C
L4_TA_AGENT_CONTROL_L	RW	32	0x4806 3020	0x4806 5020	0x4806 B020
L4_TA_AGENT_CONTROL_H	RW	32	0x4806 3024	0x4806 5024	0x4806 B024
L4_TA_AGENT_STATUS_L	RW	32	0x4806 3028	0x4806 5028	0x4806 B028
L4_TA_AGENT_STATUS_H	RW	32	0x4806 302C	0x4806 502C	0x4806 B02C

Table 5-149. CORE_TA Common Register Summary

Register Name	Type	Register Width (Bits)	CORE_TA_USB_HS_TLL Physical Address	CORE_TA_USB_HS_HOST Physical Address
L4_TA_COMPONENT_L	R	32	0x4806 3000	0x4806 5000
L4_TA_COMPONENT_H	R	32	0x4806 3004	0x4806 5004
L4_TA_CORE_L	R	32	0x4806 3018	0x4806 5018
L4_TA_CORE_H	R	32	0x4806 301C	0x4806 501C
L4_TA_AGENT_CONTROL_L	RW	32	0x4806 3020	0x4806 5020
L4_TA_AGENT_CONTROL_H	RW	32	0x4806 3024	0x4806 5024
L4_TA_AGENT_STATUS_L	RW	32	0x4806 3028	0x4806 5028
L4_TA_AGENT_STATUS_H	RW	32	0x4806 302C	0x4806 502C

Table 5-150. CORE_TA Common Register Summary

Register Name	Type	Register Width (Bits)	CORE_TA_UART2 Physical Address	CORE_TA_I2C1 Physical Address	CORE_TA_I2C2 Physical Address
L4_TA_COMPONENT_L	R	32	0x4806 D000	0x4807 1000	0x4807 3000
L4_TA_COMPONENT_H	R	32	0x4806 D004	0x4807 1004	0x4807 3004
L4_TA_CORE_L	R	32	0x4806 D018	0x4807 1018	0x4807 3018
L4_TA_CORE_H	R	32	0x4806 D01C	0x4807 101C	0x4807 301C
L4_TA_AGENT_CONTROL_L	RW	32	0x4806 D020	0x4807 1020	0x4807 3020
L4_TA_AGENT_CONTROL_H	RW	32	0x4806 D024	0x4807 1024	0x4807 3024
L4_TA_AGENT_STATUS_L	RW	32	0x4806 D028	0x4807 1028	0x4807 3028
L4_TA_AGENT_STATUS_H	RW	32	0x4806 D02C	0x4807 102C	0x4807 302C

Table 5-151. CORE_TA Common Register Summary

Register Name	Type	Register Width (Bits)	CORE_TA_MCBSP1 Physical Address	CORE_TA_ GPTIMER10 Physical Address	CORE_TA_ GPTIMER11 Physical Address
L4_TA_COMPONENT_L	R	32	0x4807 5000	0x4808 7000	0x4808 9000
L4_TA_COMPONENT_H	R	32	0x4807 5004	0x4808 7004	0x4808 9004
L4_TA_CORE_L	R	32	0x4807 5018	0x4808 7018	0x4808 9018
L4_TA_CORE_H	R	32	0x4807 501C	0x4808 701C	0x4808 901C
L4_TA_AGENT_CONTROL_L	RW	32	0x4807 5020	0x4808 7020	0x4809 9020
L4_TA_AGENT_CONTROL_H	RW	32	0x4807 5024	0x4808 7024	0x4809 9024
L4_TA_AGENT_STATUS_L	RW	32	0x4807 5028	0x4808 7028	0x4809 9028
L4_TA_AGENT_STATUS_H	RW	32	0x4807 502C	0x4808 702C	0x4809 902C

Table 5-152. CORE_TA Common Register Summary

Register Name	Type	Register Width (Bits)	CORE_TA_MAILBOX Physical Address	CORE_TA_MCBSP5 Physical Address
L4_TA_COMPONENT_L	R	32	0x4809 5000	0x4809 9000
L4_TA_COMPONENT_H	R	32	0x4809 5004	0x4809 9004
L4_TA_CORE_L	R	32	0x4809 5018	0x4809 9018
L4_TA_CORE_H	R	32	0x4809 501C	0x4809 901C
L4_TA_AGENT_CONTROL_L	RW	32	0x4809 5020	0x4809 9020
L4_TA_AGENT_CONTROL_H	RW	32	0x4809 5024	0x4809 9024
L4_TA_AGENT_STATUS_L	RW	32	0x4809 5028	0x4809 9028
L4_TA_AGENT_STATUS_H	RW	32	0x4809 502C	0x4809 902C

Table 5-153. CORE_TA Common Register Summary

Register Name	Type	Register Width (Bits)	CORE_TA_MCSPI1 Physical Address	CORE_TA_MCSPI2 Physical Address	CORE_TA_MMCHS 1 Physical Address
L4_TA_COMPONENT_L	R	32	0x4809 9000	0x4809 B000	0x4809 D000
L4_TA_COMPONENT_H	R	32	0x4809 9004	0x4809 B004	0x4809 D004
L4_TA_CORE_L	R	32	0x4809 9018	0x4809 B018	0x4809 D018
L4_TA_CORE_H	R	32	0x4809 901C	0x4809 B01C	0x4809 D01C
L4_TA_AGENT_CONTROL_L	RW	32	0x4809 9020	0x4809 B020	0x4809 D020
L4_TA_AGENT_CONTROL_H	RW	32	0x4809 9024	0x4809 B024	0x4809 D024
L4_TA_AGENT_STATUS_L	RW	32	0x4809 9028	0x4809 B028	0x4809 D028
L4_TA_AGENT_STATUS_H	RW	32	0x4809 902C	0x4809 B02C	0x4809 D02C

Table 5-154. CORE_TA Common Register Summary

Register Name	Type	Register Width (Bits)	CORE_TA_USB_HS_OTG Physical Address
L4_TA_COMPONENT_L	R	32	0x480A C000
L4_TA_COMPONENT_H	R	32	0x480A C004
L4_TA_CORE_L	R	32	0x480A C018
L4_TA_CORE_H	R	32	0x480A C01C
L4_TA_AGENT_CONTROL_L	RW	32	0x480A C020
L4_TA_AGENT_CONTROL_H	RW	32	0x480A C024
L4_TA_AGENT_STATUS_L	RW	32	0x480A C028

Table 5-154. CORE_TA Common Register Summary (continued)

Register Name	Type	Register Width (Bits)	CORE_TA_USB_HS_OTG Physical Address
L4_TA_AGENT_STATUS_H	RW	32	0x480A C02C

Table 5-155. CORE_TA Common Register Summary

Register Name	Type	Register Width (Bits)	CORE_TA_MMCHS3 Physical Address	CORE_TA_HDQ1W Physical Address
L4_TA_COMPONENT_L	R	32	0x480A E000	0x480B 3000
L4_TA_COMPONENT_H	R	32	0x480A E004	0x480B 3004
L4_TA_CORE_L	R	32	0x480A E018	0x480B 3018
L4_TA_CORE_H	R	32	0x480A E01C	0x480B 301C
L4_TA_AGENT_CONTROL_L	RW	32	0x480A E020	0x480B 3020
L4_TA_AGENT_CONTROL_H	RW	32	0x480A E024	0x480B 3024
L4_TA_AGENT_STATUS_L	RW	32	0x480A E028	0x480B 3028
L4_TA_AGENT_STATUS_H	RW	32	0x480A E02C	0x480B 302C

Table 5-156. CORE_TA Common Register Summary

Register Name	Type	Register Width (Bits)	CORE_TA_MMCHS2 Physical Address	CORE_TA_ICR Physical Address	CORE_TA_MCSPI3 Physical Address
L4_TA_COMPONENT_L	R	32	0x480B 5000	0x480B 7000	0x480B 9000
L4_TA_COMPONENT_H	R	32	0x480B 5004	0x480B 7004	0x480B 9004
L4_TA_CORE_L	R	32	0x480B 5018	0x480B 7018	0x480B 9018
L4_TA_CORE_H	R	32	0x480B 501C	0x480B 701C	0x480B 901C
L4_TA_AGENT_CONTROL_L	RW	32	0x480B 5020	0x480B 7020	0x480B 9020
L4_TA_AGENT_CONTROL_H	RW	32	0x480B 5024	0x480B 7024	0x480B 9024
L4_TA_AGENT_STATUS_L	RW	32	0x480B 5028	0x480B 7028	0x480B 9028
L4_TA_AGENT_STATUS_H	RW	32	0x480B 502C	0x480B 702C	0x480B 902C

Table 5-157. CORE_TA Common Register Summary

Register Name	Type	Register Width (Bits)	CORE_TA_MCSPI4 Physical Address	CORE_TA_CAMERA Physical Address	CORE_TA_INTH Physical Address
L4_TA_COMPONENT_L	R	32	0x480B B000	0x480C 0000	0x480C 8000
L4_TA_COMPONENT_H	R	32	0x480B B004	0x480C 0004	0x480C 8004
L4_TA_CORE_L	R	32	0x480B B018	0x480C 0018	0x480C 8018
L4_TA_CORE_H	R	32	0x480B B01C	0x480C 001C	0x480C 801C
L4_TA_AGENT_CONTROL_L	RW	32	0x480B B020	0x480C 0020	0x480C 8020
L4_TA_AGENT_CONTROL_H	RW	32	0x480B B024	0x480C 0024	0x480C 8024
L4_TA_AGENT_STATUS_L	RW	32	0x480B B028	0x480C 0028	0x480C 8028
L4_TA_AGENT_STATUS_H	RW	32	0x480B B02C	0x480C 002C	0x480C 802C

Table 5-158. CORE_TA Common Register Summary

Register Name	Type	Register Width (Bits)	CORE_TA_ICR_ MODEM Physical Address	CORE_TA_WKUP Physical Address
L4_TA_COMPONENT_L	R	32	0x480C E000	0x4834 0000
L4_TA_COMPONENT_H	R	32	0x480C E004	0x4834 0004
L4_TA_CORE_L	R	32	0x480C E018	0x4834 0018

Table 5-158. CORE_TA Common Register Summary (continued)

Register Name	Type	Register Width (Bits)	CORE_TA_ICR_ MODEM Physical Address	CORE_TA_WKUP Physical Address
L4_TA_CORE_H	R	32	0x480C E01C	0x4834 001C
L4_TA_AGENT_CONTROL_L	RW	32	0x480C E020	0x4834 0020
L4_TA_AGENT_CONTROL_H	RW	32	0x480C E024	0x4834 0024
L4_TA_AGENT_STATUS_L	RW	32	0x480C E028	0x4834 0028
L4_TA_AGENT_STATUS_H	RW	32	0x480C E02C	0x4834 002C

Table 5-159. PER_TA Common Register Summary

Register Name	Type	Register Width (Bits)	PER_TA_UART3 Physical Address	PER_TA_MCBSP2 Physical Address	PER_TA_MCBSP3 Physical Address
L4_TA_COMPONENT_L	R	32	0x4902 1000	0x4902 3000	0x4902 5000
L4_TA_COMPONENT_H	R	32	0x4902 1004	0x4902 3004	0x4902 5004
L4_TA_CORE_L	R	32	0x4902 1018	0x4902 3018	0x4902 5018
L4_TA_CORE_H	R	32	0x4902 101C	0x4902 301C	0x4902 501C
L4_TA_AGENT_CONTROL_L	RW	32	0x4902 1020	0x4902 3020	0x4902 5020
L4_TA_AGENT_CONTROL_H	RW	32	0x4902 1024	0x4902 3024	0x4902 5024
L4_TA_AGENT_STATUS_L	RW	32	0x4902 1028	0x4902 3028	0x4902 5028
L4_TA_AGENT_STATUS_H	RW	32	0x4902 102C	0x4902 302C	0x4902 502C

Table 5-160. PER_TA Common Register Summary

Register Name	Type	Register Width (Bits)	PER_TA_MCBSP4 Physical Address	PER_TA_MCBSP2_ SIDETONE Physical Address	PER_TA_MCBSP3_ SIDETONE Physical Address
L4_TA_COMPONENT_L	R	32	0x4902 7000	0x4902 9000	0x4902 B000
L4_TA_COMPONENT_H	R	32	0x4902 7004	0x4902 9004	0x4902 B004
L4_TA_CORE_L	R	32	0x4902 7018	0x4902 9018	0x4902 B018
L4_TA_CORE_H	R	32	0x4902 701C	0x4902 901C	0x4902 B01C
L4_TA_AGENT_CONTROL_L	RW	32	0x4902 7020	0x4902 9020	0x4902 B020
L4_TA_AGENT_CONTROL_H	RW	32	0x4902 7024	0x4902 9024	0x4902 B024
L4_TA_AGENT_STATUS_L	RW	32	0x4902 7028	0x4902 9028	0x4902 B028
L4_TA_AGENT_STATUS_H	RW	32	0x4902 702C	0x4902 902C	0x4902 B02C

Table 5-161. PER_TA Common Register Summary

Register Name	Type	Register Width (Bits)	PER_TA_WDTIMER3 Physical Address	PER_TA_GPTIMER2 Physical Address
L4_TA_COMPONENT_L	R	32	0x4903 1000	0x4903 3000
L4_TA_COMPONENT_H	R	32	0x4903 1004	0x4903 3004
L4_TA_CORE_L	R	32	0x4903 1018	0x4903 3018
L4_TA_CORE_H	R	32	0x4903 101C	0x4903 301C
L4_TA_AGENT_CONTROL_L	RW	32	0x4903 1020	0x4903 3020
L4_TA_AGENT_CONTROL_H	RW	32	0x4903 1024	0x4903 3024
L4_TA_AGENT_STATUS_L	RW	32	0x4903 1028	0x4903 3028
L4_TA_AGENT_STATUS_H	RW	32	0x4903 102C	0x4903 302C

Table 5-162. PER_TA Common Register Summary

Register Name	Type	Register Width (Bits)	PER_TA_GPTIMER3 Physical Address	PER_TA_GPTIMER4 Physical Address	PER_TA_GPTIMER5 Physical Address
L4_TA_COMPONENT_L	R	32	0x4903 5000	0x4903 7000	0x4903 9000
L4_TA_COMPONENT_H	R	32	0x4903 5004	0x4903 7004	0x4903 9004
L4_TA_CORE_L	R	32	0x4903 5018	0x4903 7018	0x4903 9018
L4_TA_CORE_H	R	32	0x4903 501C	0x4903 701C	0x4903 901C
L4_TA_AGENT_CONTROL_L	RW	32	0x4903 5020	0x4903 7020	0x4903 9020
L4_TA_AGENT_CONTROL_H	RW	32	0x4903 5024	0x4903 7024	0x4903 9024
L4_TA_AGENT_STATUS_L	RW	32	0x4903 5028	0x4903 7028	0x4903 9028
L4_TA_AGENT_STATUS_H	RW	32	0x4903 502C	0x4903 702C	0x4903 902C

Table 5-163. PER_TA Common Register Summary

Register Name	Type	Register Width (Bits)	PER_TA_GPTIMER6 Physical Address	PER_TA_GPTIMER7 Physical Address	PER_TA_GPTIMER8 Physical Address
L4_TA_COMPONENT_L	R	32	0x4903 B000	0x4903 D000	0x4903 F000
L4_TA_COMPONENT_H	R	32	0x4903 B004	0x4903 D004	0x4903 F004
L4_TA_CORE_L	R	32	0x4903 B018	0x4903 D018	0x4903 F018
L4_TA_CORE_H	R	32	0x4903 B01C	0x4903 D01C	0x4903 F01C
L4_TA_AGENT_CONTROL_L	RW	32	0x4903 B020	0x4903 D020	0x4903 F020
L4_TA_AGENT_CONTROL_H	RW	32	0x4903 B024	0x4903 D024	0x4903 F024
L4_TA_AGENT_STATUS_L	RW	32	0x4903 B028	0x4903 D028	0x4903 F028
L4_TA_AGENT_STATUS_H	RW	32	0x4903 B02C	0x4903 D02C	0x4903 F02C

Table 5-164. PER_TA Common Register Summary

Register Name	Type	Register Width (Bits)	PER_TA_GPTIMER9 Physical Address	PER_TA_GPIO2 Physical Address	PER_TA_GPIO3 Physical Address
L4_TA_COMPONENT_L	R	32	0x4904 1000	0x4905 1000	0x4905 3000
L4_TA_COMPONENT_H	R	32	0x4904 1004	0x4905 1004	0x4905 3004
L4_TA_CORE_L	R	32	0x4904 1018	0x4905 1018	0x4905 3018
L4_TA_CORE_H	R	32	0x4904 101C	0x4905 101C	0x4905 301C
L4_TA_AGENT_CONTROL_L	RW	32	0x4904 1020	0x4905 1020	0x4905 3020
L4_TA_AGENT_CONTROL_H	RW	32	0x4904 1024	0x4905 1024	0x4905 3024
L4_TA_AGENT_STATUS_L	RW	32	0x4904 1028	0x4905 1028	0x4905 3028
L4_TA_AGENT_STATUS_H	RW	32	0x4904 102C	0x4905 102C	0x4905 302C

Table 5-165. PER_TA Common Register Summary

Register Name	Type	Register Width (Bits)	PER_TA_GPIO4 Physical Address	PER_TA_GPIO5 Physical Address	PER_TA_GPIO6 Physical Address
L4_TA_COMPONENT_L	R	32	0x4905 5000	0x4905 7000	0x4905 9000
L4_TA_COMPONENT_H	R	32	0x4905 5004	0x4905 7004	0x4905 9004
L4_TA_CORE_L	R	32	0x4905 5018	0x4905 7018	0x4905 9018
L4_TA_CORE_H	R	32	0x4905 501C	0x4905 701C	0x4905 901C
L4_TA_AGENT_CONTROL_L	RW	32	0x4905 5020	0x4905 7020	0x4905 9020
L4_TA_AGENT_CONTROL_H	RW	32	0x4905 5024	0x4905 7024	0x4905 9024
L4_TA_AGENT_STATUS_L	RW	32	0x4905 5028	0x4905 7028	0x4905 9028
L4_TA_AGENT_STATUS_H	RW	32	0x4905 502C	0x4905 702C	0x4905 902C

Table 5-166. EMU_TA Common Register Summary

Register Name	Type	Register Width (Bits)	EMU_TA_ TEST_TAP Physical Address	EMU_TA_MPU Physical Address	EMU_TA_TPIU Physical Address
L4_TA_COMPONENT_L	R	32	0x5400 5000	0x5401 8000	0x5401 A000
L4_TA_COMPONENT_H	R	32	0x5400 5004	0x5401 8004	0x5401 A004
L4_TA_CORE_L	R	32	0x5400 5018	0x5401 8018	0x5401 A018
L4_TA_CORE_H	R	32	0x5400 501C	0x5401801C	0x5401 A01C
L4_TA_AGENT_CONTROL_L	RW	32	0x5400 5020	0x5401 8020	0x5401 A020
L4_TA_AGENT_CONTROL_H	RW	32	0x5400 5024	0x5401 8024	0x5401 A024
L4_TA_AGENT_STATUS_L	RW	32	0x5400 5028	0x5401 8028	0x5401 A028
L4_TA_AGENT_STATUS_H	RW	32	0x5400 502C	0x5401 802C	0x5401 A02C

Table 5-167. EMU_TA Common Register Summary

Register Name	Type	Register Width (Bits)	EMU_TA_ETB Physical Address	EMU_TA_DAPCTL Physical Address	EMU_TA_SDTI Physical Address	EMU_TA_L4WKUP Physical Address
L4_TA_COMPONENT_L	R	32	0x5401 C000	0x5401 E000	0x5401 F000	0x5473 0000
L4_TA_COMPONENT_H	R	32	0x5401 C004	0x5401 E004	0x5401 F004	0x5473 0004
L4_TA_CORE_L	R	32	0x5401 C018	0x5401 E018	0x5401 F018	0x5473 0018
L4_TA_CORE_H	R	32	0x5401 C01C	0x5401 E01C	0x5401 F01C	0x5473 001C
L4_TA_AGENT_CONTROL_L	RW	32	0x5401 C020	0x5401 E020	0x5401 F020	0x5473 0020
L4_TA_AGENT_CONTROL_H	RW	32	0x5401 C024	0x5401 E024	0x5401 F024	0x5473 0024
L4_TA_AGENT_STATUS_L	RW	32	0x5401 C028	0x5401 E028	0x5401 F028	0x5473 0028
L4_TA_AGENT_STATUS_H	RW	32	0x5401 C02C	0x5401 E02C	0x5401 F02C	0x5473 002C

Table 5-168. WKUP_TA Common Register Summary

Register Name	Type	Register Width (Bits)	WKUP_TA_PRM Physical Address	WKUP_TA_GPIO1 Physical Address	WKUP_TA_WDTIMER2 Physical Address
L4_TA_COMPONENT_L	R	32	0x4830 9000	0x4831 1000	0x4831 5000
L4_TA_COMPONENT_H	R	32	0x4830 9004	0x4831 1004	0x4831 5004
L4_TA_CORE_L	R	32	0x4830 9008	0x4831 1018	0x4831 5018
L4_TA_CORE_H	R	32	0x4830 901C	0x4831 101C	0x4831 501C
L4_TA_AGENT_CONTROL_L	RW	32	0x4830 9020	0x4831 1020	0x4831 5020
L4_TA_AGENT_CONTROL_H	RW	32	0x4830 9024	0x4831 1024	0x4831 5024
L4_TA_AGENT_STATUS_L	RW	32	0x4830 9028	0x4831 1028	0x4831 5028
L4_TA_AGENT_STATUS_H	RW	32	0x4830 902C	0x4831 102C	0x4831 502C

Table 5-169. WKUP_TA Common Register Summary

Register Name	Type	Register Width (Bits)	WKUP_TA_GPTIMER1 Physical Address	WKUP_TA_ SYNCTIMER32K Physical Address
L4_TA_COMPONENT_L	R	32	0x4831 9000	0x4832 1000
L4_TA_COMPONENT_H	R	32	0x4831 9004	0x4832 1004
L4_TA_CORE_L	R	32	0x4831 9008	0x4832 1018
L4_TA_CORE_H	R	32	0x4831 901C	0x4832 101C
L4_TA_AGENT_CONTROL_L	RW	32	0x4831 9020	0x4832 1020
L4_TA_AGENT_CONTROL_H	RW	32	0x4831 9024	0x4832 1024
L4_TA_AGENT_STATUS_L	RW	32	0x4831 9028	0x4832 1028

Table 5-169. WKUP_TA Common Register Summary (continued)

Register Name	Type	Register Width (Bits)	WKUP_TA_GPTIMER1 Physical Address	WKUP_TA_ SYNCTIMER32K Physical Address
L4_TA_AGENT_STATUS_H	RW	32	0x4831 902C	0x4832 102C

5.3.4.2.1 L4 Target Agent (L4 TA) Registers Description

Table 5-170. L4_TA_COMPONENT_L

Address Offset	0x000
Physical Address	SeeTable 5-145 to Table 5-168
Description	Contains a component code and revision.
Type	R

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CODE																REV															

Bits	Field Name	Description	Type	Reset
31:16	CODE	Interconnect code.	R	See (1).
15:0	REV	Component revision code.	R	See (1).

(1) TI internal Data

Table 5-171. Register Call Summary for Register L4_TA_COMPONENT_L

L4 Interconnects

- L4 Target Agent (L4 TA) Registers Manual: [0] [1] [2] [3] [4] [5] [6] [7] [8] [9] [10] [11] [12] [13] [14] [15] [16] [17] [18] [19] [20] [21] [22] [23] [24]

Table 5-172. L4_TA_COMPONENT_H

Address Offset	0x004
Physical Address	SeeTable 5-145 to Table 5-169
Description	Contains a component code and revision.
Type	R

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																															

Bits	Field Name	Description	Type	Reset
31:0	Reserved	Read returns 0	R	0x0000 000

Table 5-173. Register Call Summary for Register L4_TA_COMPONENT_H

L4 Interconnects

- L4 Target Agent (L4 TA) Registers Manual: [0] [1] [2] [3] [4] [5] [6] [7] [8] [9] [10] [11] [12] [13] [14] [15] [16] [17] [18] [19] [20] [21] [22] [23] [24]

Table 5-174. L4_TA_CORE_L

Address Offset	0x018
Physical Address	SeeTable 5-145 to Table 5-169
Description	Contains a component code and revision.
Type	R

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CORE_CODE																CORE_REV															

Bits	Field Name	Description	Type	Reset
31:16	CORE_CODE	Interconnect core code	R	See (1).
15:0	CORE_REV	Component revision code code	R	See (1).

(1) TI Internal data

Table 5-175. Register Call Summary for Register L4_TA_CORE_L

L4 Interconnects

- L4 Target Agent (L4 TA) Registers Manual: [0] [1] [2] [3] [4] [5] [6] [7] [8] [9] [10] [11] [12] [13] [14] [15] [16] [17] [18] [19] [20] [21] [22] [23] [24]

Table 5-176. L4_TA_CORE_H

Address Offset	0x01C
Physical Address	SeeTable 5-145 to Table 5-169
Description	Contains a component code and revision.
Type	R

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																VENDOR_CODE															

Bits	Field Name	Description	Type	Reset
31:16	Reserved	Reserved	R	0x0000
15:0	VENDOR_CODE	Vendor revision core code	R	See (1).

(1) TI Internal data

Table 5-177. Register Call Summary for Register L4_TA_CORE_H

L4 Interconnects

- L4 Target Agent (L4 TA) Registers Manual: [0] [1] [2] [3] [4] [5] [6] [7] [8] [9] [10] [11] [12] [13] [14] [15] [16] [17] [18] [19] [20] [21] [22] [23] [24]

Table 5-178. L4_TA_AGENT_CONTROL_L

Address Offset	0x020
Physical Address	SeeTable 5-145 to Table 5-169
Description	Enable error reporting
Type	RW

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							SERROR_REP	Reserved													REQ_TIMEOUT			Reserved							OCP_RESET

Bits	Field Name	Description	Type	Reset
31:25	Reserved	Read returns 0.	R	0x00
24	SERROR_REP	Enable logging of error	R	0x0
23:11	Reserved	Read returns 0.		
10:8	REQ_TIMEOUT	Timeout Bound. Values are:0 - No timeout 1 - 1x base cycles 2 - 4x base cycles 3 - 16x base cycles 4 - 64x base cycles	RW	0x2
7:1	Reserved	Read returns 0.	R	0x00
0	OCP_RESET	The OCP_RESET field controls the OCP reset signal to the attached core. Setting this bit clears any pending transfers and resets the OCP interface. The bit must be cleared to de-assert the OCP reset signal. When the software reset feature is available on a target agent, the target agent OCP must also have a reset signal directed to the target core.	RW	0

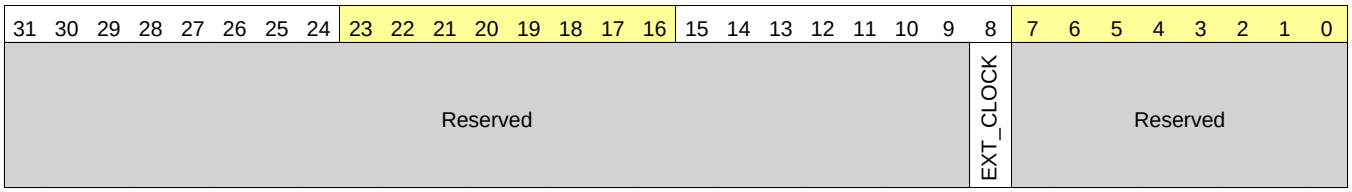
Table 5-179. Register Call Summary for Register L4_TA_AGENT_CONTROL_L

L4 Interconnects

- Error Handling: [0] [1] [2] [3] [4] [5] [6]
- L4 Target Agent (L4 TA) Registers Manual: [7] [8] [9] [10] [11] [12] [13] [14] [15] [16] [17] [18] [19] [20] [21] [22] [23] [24] [25] [26] [27] [28] [29] [30] [31]

Table 5-180. L4_TA_AGENT_CONTROL_H

Address Offset	0x024
Physical Address	SeeTable 5-145 to Table 5-169
Description	Enable clock power management
Type	RW



Bits	Field Name	Description	Type	Reset
31:9	Reserved	Read returns 0.	R	0x000000
8	EXT_CLOCK	When set to 1, the ext_clk_off_i signal on a target agent indicates when the target agent should shut off.	R	0
7:0	Reserved	Read returns 0.	R	0x00

Table 5-181. Register Call Summary for Register L4_TA_AGENT_CONTROL_H

L4 Interconnects

- L4 Target Agent (L4 TA) Registers Manual: [0] [1] [2] [3] [4] [5] [6] [7] [8] [9] [10] [11] [12] [13] [14] [15] [16] [17] [18] [19] [20] [21] [22] [23] [24]

Table 5-182. L4_TA_AGENT_STATUS_L

Address Offset	0x028
Physical Address	See Table 5-145 to Table 5-169
Description	Error reporting
Type	RW

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								Reserved															REQ_TIMEOUT	Reserved							

Bits	Field Name	Description	Type	Reset
31:24	Reserved	Read returns 0.	R	0x00
23:9	Reserved	Read returns 0.	R	0x0001
8	REQ_TIMEOUT	0x0: No request timeout 0x1: A request timeout has occurred	R 1toCLR	0
7:0	Reserved	Read returns 0.	R	0x00

Table 5-183. Register Call Summary for Register L4_TA_AGENT_STATUS_L

L4 Interconnects

- Error Handling: [0] [1] [2]
- L4 Target Agent (L4 TA) Registers Manual: [3] [4] [5] [6] [7] [8] [9] [10] [11] [12] [13] [14] [15] [16] [17] [18] [19] [20] [21] [22] [23] [24] [25] [26] [27]

Table 5-184. L4_TA_AGENT_STATUS_H

Address Offset	0x02C
Physical Address	SeeTable 5-145 to Table 5-169
Description	Error reporting
Type	RW

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																															

Bits	Field Name	Description	Type	Reset
31:0	Reserved	Read returns 0	R	0x0000 000

Table 5-185. Register Call Summary for Register L4_TA_AGENT_STATUS_H

L4 Interconnects

- L4 Target Agent (L4 TA) Registers Manual: [0] [1] [2] [3] [4] [5] [6] [7] [8] [9] [10] [11] [12] [13] [14] [15] [16] [17] [18] [19] [20] [21] [22] [23] [24]

5.3.4.3 L4 Link Register Agent (LA)

This section provides information on the L4-Core link agent (LA) register module. Each of the registers within the module instance is described seperatly in Table 5-186.

The LA register blockcontains the initiator subsystem information register and the composite sideband signal mask and status registers.

Table 5-186. L4 LA Register Summary

Register Name	Type	Register Width (Bits)	CORE_LA Physical Address	PER_LA Physical Address	EMU_LA Physical Address	WKUP_LA Physical Address
L4_LA_COMPONENT_L	R	32	0x4804 1000	4900 1000	0x5400 7000	0x4832 9000
L4_LA_COMPONENT_H	R	32	0x4804 1004	0x4900 1004	0x5400 7004	0x4832 9004
L4_LA_NETWORK_L	R	32	0x4804 1010	0x4900 1010	0x5400 7010	0x4832 9010
L4_LA_NETWORK_H	R	32	0x4804 1014	0x4900 1014	0x5400 7014	0x4832 9014
L4_LA_INITIATOR_INFO_L	R	32	0x4804 1018	0x4900 1018	0x5400 7018	0x4832 9018
L4_LA_INITIATOR_INFO_H	R	32	0x4804 101C	0x4900 101C	0x5400 701C	0x4832 901C
L4_LA_NETWORK_CONTROL_L	RW	32	0x4804 1020	0x4900 1020	0x5400 7020	0x4832 9020
L4_LA_NETWORK_CONTROL_H	RW	32	0x4804 1024	0x4900 1024	0x5400 7024	0x4832 9024

5.3.4.3.1 L4 Link Register Agent (LA) Registers Description

Table 5-187. L4_LA_COMPONENT_L

Address Offset	0x000
Physical Address	Please refer to Table 5-186
Description	Contain a component code and revision, which are used to identify the hardware of the component.
Type	R

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CODE																REV															

Bits	Field Name	Description	Type	Reset
31:16	CODE	Interconnect code.	R	See (1).
15:0	REV	Component revision code.	R	See (1).

(1) TI internal data

Table 5-188. Register Call Summary for Register L4_LA_COMPONENT_L

L4 Interconnects

- L4 Link Register Agent (LA) Registers Manual: [0]

Table 5-189. L4_LA_COMPONENT_H

Address Offset	0x004
Physical Address	Please refer to Table 5-186
Description	Contain a component code and revision, which are used to identify the hardware of the component.
Type	R

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																															

Bits	Field Name	Description	Type	Reset
31:0	Reserved	Read returns 0	R	0x0000 0000

Table 5-190. Register Call Summary for Register L4_LA_COMPONENT_H

L4 Interconnects

- L4 Link Register Agent (LA) Registers Manual: [0]

Table 5-191. L4_LA_NETWORK_L

Address Offset	0x010
Physical Address	Please refer to Table 5-186
Description	Identify the interconnect
Type	R

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																															

Bits	Field Name	Description	Type	Reset
31:0	Reserved	Read returns 0	R	0x0000 0000

Table 5-192. Register Call Summary for Register L4_LA_NETWORK_L

L4 Interconnects

- L4 Link Register Agent (LA) Registers Manual: [0]

Table 5-193. L4_LA_NETWORK_H

Address Offset	0x014
Physical Address	Please refer to Table 5-186
Description	Identify the interconnect
Type	R

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ID																															

Bits	Field Name	Description	Type	Reset
31:0	ID	The ID field uniquely identifies this interconnect, and can serve as a chip ID.	R	0x00010000

Table 5-194. Register Call Summary for Register L4_LA_NETWORK_H

L4 Interconnects

- L4 Link Register Agent (LA) Registers Manual: [0]

Table 5-195. L4_LA_INITIATOR_INFO_L

Address Offset	0x018
Physical Address	Please refer to Table 5-186
Description	Contain initiator subsystem information.
Type	R

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved				PROT_GROUPS				NUMBER_REGIONS								Reserved												SEGMENTS			

Bits	Field Name	Description	Type	Reset
31:28	Reserved	Read returns 0.	R	0x0
27:24	PROT_GROUPS	The number of protection groups. The PROT_GROUPS field contains read-only configuration information for the address mapping and protection structure of the initiator subsystem. If the PROT_GROUPS field is set to 0, there are no protection group registers.	R	see Table 5-197
23:16	NUMBER_REGIONS	The number of regions. The NUMBER_REGIONS field contains read-only configuration information for the region register of the initiator subsystem.	R	see Table 5-197
15:4	Reserved	Read returns 0.	R	0x000
3:0	SEGMENTS	The number of segments. The SEGMENT fields contains read-only configuration information for the segment register of the initiator subsystem.	R	see Table 5-197

Table 5-196. Register Call Summary for Register L4_LA_INITIATOR_INFO_L

L4 Interconnects

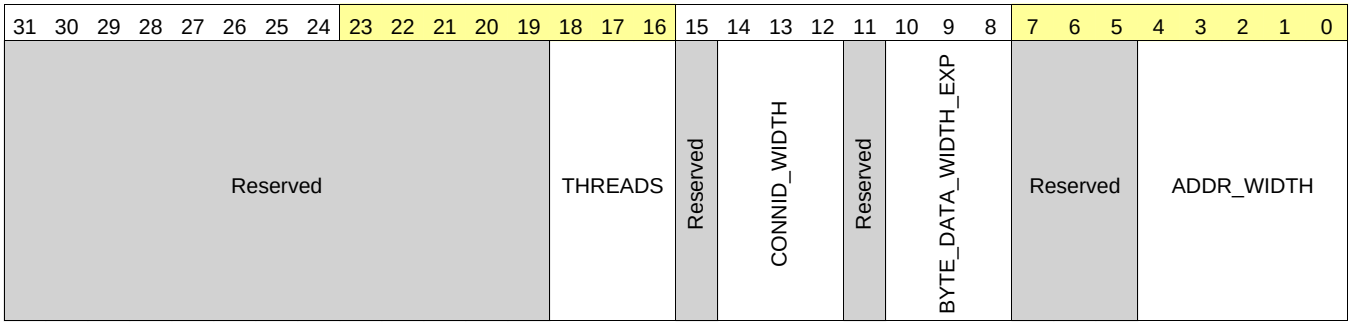
- L4 Link Register Agent (LA) Registers Manual: [0]

Table 5-197. Reset value for L4_LA_INITIATOR_INFO_L

Field Name	CORE_LA	PER_LA	EMU_LA	WKUP_LA
PROT_GROUPS	0x8	0x8	0x8	0x8
NUMBER_REGIONS	0x64	0x2B	0x1A	0x13
SEGMENTS	0x6	0x5	0x3	0x2

Table 5-198. L4_LA_INITIATOR_INFO_H

Address Offset	0x01C
Physical Address	Please refer to Table 5-186
Description	Contain initiator subsystem information.
Type	R



Bits	Field Name	Description	Type	Reset
31:19	Reserved	Read returns 0.	R	0x0000
18:16	THREADS	The THREADS field specifies the number of initiator threads connected to the interconnect. The field contains read-only configuration information for the initiator subsystem.	R	see Table 5-200
15	Reserved	Read returns 0.	R	0
14:12	CONNID_WIDTH	The initiator subsystem connID width. The CONNID_WIDTH field contains read-only configuration information for the initiator subsystem.	R	see Table 5-200
11	Reserved	Read returns 0.	R	0

Bits	Field Name	Description	Type	Reset
10:8	BYTE_DATA_ WIDTH_EXP	This field specifies the initiator subsystem data width. 1:2^1 bytes specifies a 16-bit data width and 2:2^2 bytes specifies a 32-bit data width. The BYTE_DATA_WIDTH_EXP field contains read-only configuration information for the initiator subsystem.	R	see Table 5-200
7:5	Reserved	Read returns 0.	R	0x0
4:0	ADDR_WIDTH	This field specifies the initiator subsystem address width. The ADDR_WIDTH field contains read-only configuration information for the initiator subsystem.	R	see Table 5-200

Table 5-199. Register Call Summary for Register L4_LA_INITIATOR_INFO_H

L4 Interconnects

- L4 Link Register Agent (LA) Registers Manual: [0]

Table 5-200. Reset value for L4_LA_INITIATOR_INFO_H

Field Name	CORE_LA	PER_LA	EMU_LA	WKUP_LA
THREADS	0x4	0x4	0x2	0x2
CONNID_WIDTH	0x4	0x4	0x4	0x0
BYTE_DATA_WIDTH_EXP	0x2	0x2	0x2	0x2
ADDR_WIDTH	0x18	0x14	0x18	0x14

Table 5-201. L4_LA_NETWORK_CONTROL_L

Address Offset	0x020
Physical Address	Please refer to Table 5-186
Description	Control interconnect minimum timeout values.
Type	RW

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																					TIMEOUT_BASE			Reserved							

Bits	Field Name	Description	Type	Reset
31:11	Reserved	Read returns 0.	R	0x000000
10:8	TIMEOUT_BASE	The TIMEOUT_BASE field indicates the timeout period (that is, base cycles) for the highest frequency time-base signal sent from the L4 initiator subsystem to all target agents that have timeout enabled. Values for the field are: 0 - Timeout disabled 1 - L4 interconnect clock cycles divided by 64 2 - L4 interconnect clock cycles divided by 256 3 - L4 interconnect clock cycles divided by 1024 4 - L4 interconnect clock cycles divided by 4096	RW	0x4
7:0	Reserved	Read returns 0.	R	0x00

Table 5-202. Register Call Summary for Register L4_LA_NETWORK_CONTROL_L

L4 Interconnects

- Error Handling: [0] [1]
- L4 Link Register Agent (LA) Registers Manual: [2]

Table 5-203. L4_LA_NETWORK_CONTROL_H

Address Offset	0x024
Physical Address	Please refer to Table 5-186
Description	Control interconnect global power control
Type	RW

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							CLOCK_GATE_DISABLE	Reserved			THREAD0_PRI	Reserved											EXT_CLOCK	Reserved							

Bits	Field Name	Description	Type	Reset
31:25	Reserved	Read returns 0.	R	0x00
24	CLOCK_GATE_DISABLE	When set to 1 this field disables all clock gating.	RW	0
23:21	Reserved	Read returns 0.	R	0x0
20	THREAD0_PRI	Sets thread priority. If the field is set to 0, the default, all initiator threads are treated the same. Setting the THREAD0_PRI field to 1 assigns a higher arbitration priority to thread 0 of the first initiator OCP interface. To avoid starvation, arbitration is imposed by the initiator subsystem. When multiple requests from different initiator threads are dispatched to targets simultaneously, the oldest request is dispatched first. If thread 0 is assigned a higher priority, a request on thread 0 always wins arbitration. Assigning thread 0 of the first initiator OCP the highest priority on a request or response can result in the starvation of other threads.	R	1
19:9	Reserved	Read returns 0.	R	0x000
8	EXT_CLOCK	When set to 1, the ext_clk_off_i signal on the initiator subsystem instructs the entire L4 to shut off.	R	1
7:0	Reserved	Read returns 0.	R	0x00

Table 5-204. Register Call Summary for Register L4_LA_NETWORK_CONTROL_H

L4 Interconnects

- Clocking, Reset, and Power-Management Scheme: [0]
- L4 Link Register Agent (LA) Registers Manual: [1]

5.3.4.4 L4 Address Protection (AP)

This section provides information on the L4-Core link agent (LA) register module. Each of the registers within the module instance is described seperatly in Table 5-205.

The AP register block contains the segment, address region, and protection group registers that can be used to specify the addressing scheme or restrict the access of target address regions.

Table 5-205. L4 AP Register Summary

Register Name	Type	Register Width (Bits)	CORE_AP Physical Address	PER_AP Physical Address
L4_AP_COMPONENT_L	R	32	0x4804 0000	0x4900 0 000
L4_AP_COMPONENT_H	R	32	0x4804 0004	0x4900 0000

Table 5-205. L4 AP Register Summary (continued)

Register Name	Type	Register Width (Bits)	CORE_AP Physical Address	PER_AP Physical Address
L4_AP_SEGMENT_i_L (1)	RW	32	0x4804 0100 + (0x08*i)	0x4900 0100 + (0x08*i)
L4_AP_SEGMENT_i_H (1)	RW	32	0x4804 0104 + (0x08*i)	0x4900 0104 + (0x08*i)
L4_AP_PROT_GROUP_MEMBERS_k_L (2)	R	32	0x4804 0200 + (0x08*k)	0x4900 0200 + (0x08*k)
L4_AP_PROT_GROUP_MEMBERS_k_H (2)	R	32	0x4804 0204 + (0x08*k)	0x4900 0204 + (0x08*k)
L4_AP_PROT_GROUP_ROLES_k_L (2)	R	32	0x4804 0280 + (0x08*k)	0x4900 0280 + (0x08*k)
L4_AP_PROT_GROUP_ROLES_k_H (2)	R	32	0x4804 0284 + (0x08*k)	0x4900 0284 + (0x08*k)
L4_AP_REGION_l_L (3)	RW	32	0x4804 0300 + (0x08*l)	0x4900 0300 + (0x08*l)
L4_AP_REGION_l_H (3)	RW	32	0x4804 0304 + (0x08*l)	0x4900 0304 + (0x08*l)

(1) i = 0 to 5 for CORE_AP
i = 0 to 4 for PER_AP
i = 0 to 2 for EMU_AP
i = 0 to 1 for WKUP_AP
(2) k = 0 to 7 for CORE_AP and PER_AP
k = 0 to 5 for EMU_AP
(3) l = 0 to 99 for CORE_AP
l = 0 to 42 for PER_AP
l = 0 to 25 for EMU_AP
l = 0 to 18 for WKUP_AP

Table 5-206. L4 AP Register Summary

Register Name	Type	Register Width (Bits)	EMU_AP Physical Address	WKUP_AP Physical Address
L4_AP_COMPONENT_L	R	32	0x5400 6000	0x4832 8000
L4_AP_COMPONENT_H	R	32	0x5400 6004	0x4832 8000
L4_AP_SEGMENT_i_L (1)	RW	32	0x5400 6100 + (0x08*i)	0x4832 8100 + (0x08*i)
L4_AP_SEGMENT_i_H (1)	RW	32	0x5400 6104 + (0x08*i)	0x4832 8104 + (0x08*i)
L4_AP_PROT_GROUP_MEMBERS_k_L (2)	R	32	0x5400 6200 + (0x08*k)	N/A
L4_AP_PROT_GROUP_MEMBERS_k_H (2)	R	32	0x5400 6204 + (0x08*k)	N/A
L4_AP_PROT_GROUP_ROLES_k_L (2)	R	32	0x5400 6280 + (0x08*k)	N/A
L4_AP_PROT_GROUP_ROLES_k_H (3)	R	32	0x5400 6284 + (0x08*k)	N/A
L4_AP_REGION_l_L (4)	RW	32	0x5400 6300 + (0x08*l)	0x4832 8300 + (0x08*l)
L4_AP_REGION_l_H (4)	RW	32	0x5400 6304 + (0x08*l)	0x4832 8304 + (0x08*l)

(1) i = 0 to 5 for CORE_AP
i = 0 to 4 for PER_AP
i = 0 to 2 for EMU_AP
i = 0 to 1 for WKUP_AP
(2) k = 0 to 7 for CORE_AP and PER_AP
k = 0 to 5 for EMU_AP
(3) k = 0 to 7 for CORE_AP and PER_AP
k = 0 to 5 for EMU_AP
(4) l = 0 to 99 for CORE_AP
l = 0 to 42 for PER_AP
l = 0 to 25 for EMU_AP
l = 0 to 18 for WKUP_AP

5.3.4.4.1 L4 Address Protection (AP) Registers Description

Table 5-207. L4_AP_COMPONENT_L

Address Offset	0x000
Physical Address	Please refer to Table 5-205
Description	Contains a component code and revision, which are used to identify the hardware of the component.
Type	R

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CODE																REV															

Bits	Field Name	Description	Type	Reset
31:16	CODE	Interconnect code.	R	See (1).
15:0	REV	Component revision code.	R	See (1).

(1) TI Internal Data

Table 5-208. Register Call Summary for Register L4_AP_COMPONENT_L

L4 Interconnects

- L4 Address Protection (AP) Registers Manual: [0] [1]

Table 5-209. L4_AP_COMPONENT_H

Address Offset	0x004
Physical Address	Please refer to Table 5-205
Description	Contains a component code and revision, which are used to identify the hardware of the component.
Type	R

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																															

Bits	Field Name	Description	Type	Reset
31:0	Reserved	Read returns 0	R	0x0000 0000

Table 5-210. Register Call Summary for Register L4_AP_COMPONENT_H

L4 Interconnects

- L4 Address Protection (AP) Registers Manual: [0] [1]

Table 5-211. L4_AP_SEGMENT_i_L

Address Offset	0x100 + (0x08*i)	**Index**	i = 0 to 5 for CORE_AP, i = 0 to 4 for PER_AP, i = 0 to 2 for EMU_AP, i = 0 to 1 for WKUP_AP,
Physical Address	Please refer to Table 5-205		
Description	Define the base address of each segments		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								BASE																							

Bits	Field Name	Description	Type	Reset
31:24	Reserved	Read returns 0.	R	0x00
23:0	BASE	The base address of the segment (with 0s from bit 0 to bit SIZE-1).	R	see Table 5-213

Table 5-212. Register Call Summary for Register L4_AP_SEGMENT_i_L

L4 Interconnects

- L4 Protection and Firewalls: [0]
- L4 Address Protection (AP) Registers Manual: [1] [2]

Table 5-213. L4_AP_SEGMENT_i_L Reset Values

BASE	CORE_AP	PER_AP	EMU_AP	WKUP_AP
i = 0	0x00 0000	0x00 0000	0x00 0000	0x00 0000
i = 1	0x04 0000	0x02 0000	0x40 0000	0x02 0000
i = 2	0x08 0000	0x03 0000	0x60 0000	N/A
i = 3	0x0C 0000	0x04 0000	N/A	N/A
i = 4	0x30 0000	0x05 0000	N/A	N/A
i = 5	0x32 0000	N/A	N/A	N/A

Table 5-214. L4_AP_SEGMENT_i_H

Address Offset	0x104 + (0x08*i)	**Index**	i = 0 to 5 for CORE_AP, i = 0 to 4 for PER_AP, i = 0 to 2 for EMU_AP, i = 0 to 1 for WKUP_AP,
Physical Address	Please refer to Table 5-205		
Description	Define the size of each segments		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																								SIZE							

Bits	Field Name	Description	Type	Reset
31:5	Reserved	Read returns 0.	R	0x0000000
4:0	SIZE	Segment size is a power of 2, where 2^SIZE is the byte size of a segment (all segment registers use the same size).	R	see Table 5-216

Table 5-215. Register Call Summary for Register L4_AP_SEGMENT_i_H

L4 Interconnects

- L4 Protection and Firewalls: [0]
- L4 Address Protection (AP) Registers Manual: [1] [2]

Table 5-216. L4_AP_SEGMENT_i_H reset values

SIZE	CORE_AP	PER_AP	EMU_AP	WKUP_AP
For all i	0x12	0x10	0x15	0x11

Table 5-217. L4_AP_PROT_GROUP_MEMBERS_k_L

Address Offset	0x200 + (0x08*k)	**Index**	k = 0 to 7 for CORE_ AP and PER_AP. k = 0 to 5 for EMU_AP
Physical Address	Please refer to Table 5-205		
Description	Define connID bit vectors for a protection group.		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																CONNID_BIT_VECTOR															

Bits	Field Name	Description	Type	Reset
31:16	Reserved	Read returns 0	R	0x0000
15:0	CONNID_BIT_VECTOR	A bit of 1 in position n means that connID n is allowed in this protection group. Illegal connIDs have their bits set to 0s.	R	0xFFFF

Table 5-218. Register Call Summary for Register L4_AP_PROT_GROUP_MEMBERS_k_L

L4 Interconnects

- L4 Protection and Firewalls: [0]
- L4 Address Protection (AP) Registers Manual: [1] [2]

Table 5-219. L4_AP_PROT_GROUP_MEMBERS_k_H

Address Offset	0x204 + (0x08*k)	**Index**	k = 0 to 7 for CORE_ AP and PER_AP. k = 0 to 5 for EMU_AP
Physical Address	Please refer to Table 5-205		
Description	Define connID bit vectors for a protection group.		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																															

Bits	Field Name	Description	Type	Reset
31:0	Reserved	Read returns 0's	R	0x0000 0000

Table 5-220. Register Call Summary for Register L4_AP_PROT_GROUP_MEMBERS_k_H

L4 Interconnects

- L4 Address Protection (AP) Registers Manual: [0] [1]

Table 5-221. L4_AP_PROT_GROUP_ROLES_k_L

Address Offset	0x200 + (0x08*k)	**Index**	k = 0 to 7 for CORE_ AP and PER_AP. k = 0 to 5 for EMU_AP
Physical Address	Please refer to Table 5-205		
Description	Define MReqInfo bit vectors for a protection group.		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																ENABLE															

Bits	Field Name	Description	Type	Reset
31:16	Reserved	Read returns 0	R	0x0000
15:0	ENABLE	Setting of type acces allowed for the group of initiators see Table 5-22.	R	0xFFFF

Table 5-222. Register Call Summary for Register L4_AP_PROT_GROUP_ROLES_k_L

L4 Interconnects

- L4 Protection and Firewalls: [0]
- L4 Address Protection (AP) Registers Manual: [1] [2]

Table 5-223. L4_AP_PROT_GROUP_ROLES_k_H

Address Offset	0x204 + (0x08*k)	**Index**	k = 0 to 7 for CORE_ AP and PER_AP. k = 0 to 5 for EMU_AP
Physical Address	Please refer to Table 5-205		
Description	Define connID bit vectors for a protection group.		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																															

Bits	Field Name	Description	Type	Reset
31:0	Reserved	Read returns 0's	R	0x0000 0000

Table 5-224. Register Call Summary for Register L4_AP_PROT_GROUP_ROLES_k_H

L4 Interconnects

- L4 Address Protection (AP) Registers Manual: [0] [1]

Table 5-225. L4_AP_REGION_l_L

Address Offset	0x300 + (0x08*l)	**Index**	l = 0 to 99 for CORE_AP, l = 0 to 42 for PER_AP, l = 0 to 25 for EMU_AP, l = 0 to 18 for WKUP_AP,
Physical Address	Please refer to Table 5-205		
Description	Define the base address of the region in respect to the segment it belongs to.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								BASE																							

Bits	Field Name	Description	Type	Reset
31:24	Reserved	Read returns 0.	R	0x00
23:0	BASE	Sets the base address of this region relative to its segment base.	R	See Table 5-229 to Table 5-232

Table 5-226. Register Call Summary for Register L4_AP_REGION_l_L

L4 Interconnects

- L4 Protection and Firewalls: [0]
- L4 Address Protection (AP) Registers Manual: [1] [2]

Table 5-227. L4_AP_REGION_l_H

Address Offset	0x304 + (0x08*l)	**Index**	l = 0 to 99 for CORE_AP, l = 0 to 42 for PER_AP, l = 0 to 25 for EMU_AP, l = 0 to 18 for WKUP_AP,
Physical Address	Please refer to Table 5-205		
Description	Define the size, protection group^and segment ID of the region		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved				SEGMENT_ID				Reserved	PROT_GROUP_ID			BYTE_DATA_WIDTH_EXP				Reserved									SIZE						ENABLE

Bits	Field Name	Description	Type	Reset
31:28	Reserved	Read returns 0	R	0x0
27:24	SEGMENT_ID	Identifies the segment to which the region is part of.	R	See Table 5-229 to Table 5-232
23	Reserved	Read returns 0.	R	0
22:20	PROT_GROUP_ID	The protection group containing this region.	RW	See Table 5-229 to Table 5-232
19:17	BYTE_DATA_ WIDTH_EXP	The target OCP data byte width is 2^(BYTE_DATA_WIDTH_EXP) bytes. The value of this field is derived from the target OCP data_wdth parameter.	R	See Table 5-229 to Table 5-232
16:6	Reserved	Read returns 0.	R	0x0
5:1	SIZE	Define the size of the region in bytes. 2^SIZE equals the region.	R	See Table 5-229 to Table 5-232
0	ENABLE	0x0: Disable the region, no access allows 0x1: Enable the region, with access as define in registers	R	See Table 5-229 to Table 5-232

Table 5-228. Register Call Summary for Register L4_AP_REGION_I_H

L4 Interconnects

- L4 Protection and Firewalls: [0] [1]
- L4 Address Protection (AP) Registers Manual: [2] [3]

Table 5-229. Reset Values for CORE_AP L4_AP_REGION_I_L and L4_AP_REGION_I_H

y	BASE	SEGMENT_ID	PROT_GROUP_ID	BYTE_DATA_WIDTH_EXP	SIZE
0	0x00 0000	1	0	2	0x0B
1	0x00 0800	1	7	2	0x0B
2	0x00 1000	1	7	2	0x0C
3	0x01 0000	1	2	2	0x0A
4	0x01 0400	1	2	2	0x0A
5	0x01 0800	1	2	2	0x0A
6	0x01 0C00	1	2	2	0x0A
7	0x01 1000	1	2	2	0X0C
8	0x00 0000	3	3	2	0x0C

Table 5-229. Reset Values for CORE_AP L4_AP_REGION_I_L and L4_AP_REGION_I_H (continued)

y	BASE	SEGMENT_ID	PROT_GROUP_ID	BYTE_DATA_WIDTH_EXP	SIZE
9	0x01 6000	1	7	2	0x0C
10	0x01 7000	1	7	2	0x0C
11	0x01 8000	1	7	2	0x0C
12	0x01 C000	1	7	2	0x0C
13	0x02 B000	2	7	2	0x0C
14	0x02 C000	2	7	2	0x0C
15	0x01 E000	1	7	2	0x0C
16	0x01 F000	1	7	2	0x0C
17	0x02 A000	1	7	2	0x0C
18	0x02 B000	1	7	2	0x0C
19	0x02 C000	1	7	2	0X0C
20	0x02 D000	1	7	2	0x0C
21	0x03 0000	1	7	1	0x0C
22	0x03 1000	1	7	1	0x0C
23	0x03 2000	1	7	1	0x0C
24	0x03 3000	1	7	1	0x0C
25	0x03 4000	1	7	2	0x0C
26	0x03 5000	1	7	2	0x0C
27	0x00 6000	2	7	2	0x0C
28	0x00 7000	2	7	2	0x0C
29	0x00 8000	2	7	2	0x0C
30	0x00 9000	2	7	2	0x0C
31	0x00 9000	3	7	2	0x0C
32	0x00 A000	3	7	2	0x0C
33	0x01 2000	2	7	2	0x0C
34	0x01 3000	2	7	2	0x0C
35	0x01 4000	2	7	2	0x0C
36	0x01 5000	2	7	2	0x0C
37	0x01 8000	2	7	2	0x0C
38	0x01 9000	2	7	2	0x0C
39	0x01 A000	2	7	2	0x0C
40	0x01 B000	2	7	2	0x0C
41	0x01 C000	2	7	2	0x0C
42	0x01 D000	2	7	2	0x0C
43	0x03 4000	2	7	2	0x0C
44	0x03 5000	2	7	2	0x0C
45	0x01 E000	2	7	2	0x0C
46	0x01 F000	2	7	2	0x0C
47	0x02 0000	2	1	2	0x0C
48	0x02 1000	2	1	2	0x0C
49	0x02 2000	2	1	2	0x0C
50	0x02 3000	2	1	2	0x0C
51	0x02 4000	2	1	2	0x0C
52	0x02 5000	2	1	2	0x0C
53	0x02 6000	2	1	2	0x0C
54	0x02 7000	2	1	2	0x0C
55	0x02 8000	2	1	2	0x0D

Table 5-229. Reset Values for CORE_AP L4_AP_REGION_I_L and L4_AP_REGION_I_H (continued)

y	BASE	SEGMENT_ID	PROT_GROUP_ID	BYTE_DATA_WIDTH_EXP	SIZE
56	0x02 A000	2	1	2	0x0C
57	0x03 0000	2	7	2	0x0C
58	0x03 1000	2	7	2	0x0C
59	0x03 2000	2	7	2	0x0C
60	0x03 3000	2	7	2	0x0C
61	0x01 6000	2	7	2	0x0C
62	0x01 7000	2	7	2	0x0C
63	0x01 9000	1	7	2	0x0C
64	0x01 A000	1	7	2	0x0C
65	0x01 B000	1	7	2	0x0C
66	0x03 8000	2	7	2	0x0C
67	0x03 9000	2	7	2	0x0C
68	0x00 4000	0	7	2	0x0C
69	0x00 7000	0	7	2	0x0C
70	0x00 B000	3	7	2	0x0C
71	0x00 C000	3	7	2	0x0C
72	0x00 6000	0	7	2	0x0B
73	0x02 0000	1	7	1	0x0C
74	0x02 1000	1	7	1	0x0C
75	0x00 2000	0	7	2	0x0C
76	0x00 3000	0	7	2	0x0C
77	0x03 C000	2	3	3	0x0C
78	0x00 4000	4	4	2	0x0D
79	0x03 A000	2	7	2	0x0C
80	0x03 B000	2	7	2	0x0C
81	0x00 0000	5	7	2	0x0C
82	0x00 1000	3	1	2	0x0C
83	0x00 2000	3	1	2	0x0C
84	0x00 3000	3	1	2	0x0C
85	0x00 4000	3	1	2	0x0C
86	0x00 5000	3	1	2	0x0C
87	0x00 6000	3	1	2	0x0C
88	0x03 6000	2	7	2	0x0C
89	0x03 7000	2	7	2	0x0C
90	0x00 7000	3	7	2	0x0C
91	0x00 8000	3	7	2	0x0C
92	0x00 D000	3	7	2	0x0C
93	0x00 E000	3	7	2	0x0C
94	0x00 6000	4	7	2	0x0D
95	0x00 8800	4	7	2	0x0B
96	0x00 9000	4	7	2	0x0C
97	0x00 C000	4	1	2	0x0D
98	0x01 0000	4	7	2	0x10
99	0x02 0000	4	7	2	0x10

Table 5-230. Reset Values for PER_AP L4_AP_REGION_l_L and L4_AP_REGION_l_H

y	BASE	SEGMENT_ID	PROT_GROUP_ID	BYTE_DATA_WIDTH_EXP	SIZE
0	0x00 0000	0	0	2	0x0B
1	0x00 0800	0	7	2	0x0B
2	0x00 1000	0	7	2	0x0C
3	0x00 0000	1	7	2	0x0C
4	0x00 1000	1	7	2	0X0C
5	0x00 2000	1	7	2	0X0C
6	0x00 3000	1	7	2	0X0C
7	0x00 4000	1	7	2	0X0C
8	0x00 5000	1	7	2	0x0C
9	0x00 6000	1	7	2	0x0C
10	0x00 7000	1	7	2	0x0C
11	0x00 0000	1	7	2	0x0C
12	0x00 1000	2	7	2	0x0C
13	0x00 2000	2	7	2	0x0C
14	0x00 3000	2	7	2	0x0C
15	0x00 4000	2	7	2	0x0C
16	0x00 5000	2	7	2	0x0C
17	0x00 6000	2	7	2	0x0C
18	0x00 7000	2	7	2	0x0C
19	0x00 8000	2	7	2	0X0C
20	0x00 9000	2	7	2	0x0C
21	0x00 A000	2	7	1	0x0C
22	0x00 B000	2	7	2	0x0C
23	0x00 C000	2	7	2	0x0C
24	0x00 D000	2	7	2	0x0C
25	0x00 E000	2	7	2	0x0C
26	0x00 F000	2	7	2	0x0C
27	0x00 0000	3	7	2	0x0C
28	0x00 1000	3	7	2	0x0C
29	0x00 0000	4	7	2	0x0C
30	0x00 1000	4	7	2	0x0C
31	0x00 2000	4	7	2	0x0C
32	0x00 3000	4	7	2	0x0C
33	0x00 4000	4	7	2	0x0C
34	0x00 5000	4	7	2	0x0C
35	0x00 6000	4	7	2	0x0C
36	0x00 7000	4	7	2	0x0C
37	0x00 8000	4	7	2	0x0C
38	0x00 9000	4	7	2	0x0C
39	0x00 8000	4	7	2	0x0C
40	0x00 9000	1	7	2	0x0C
41	0x00 A000	1	7	2	0x0C
42	0x00 B000	1	7	2	0x0C

Table 5-231. Reset Values for EMU_AP L4_AP_REGION_l_L and L4_AP_REGION_l_H

y	BASE	SEGMENT_ID	PROT_GROUP_ID	BYTE_DATA_WIDTH_EXP	SIZE
0	0x00 0000	2	5	2	0x14
1	0x00 4000	0	5	2	0x0C
2	0x00 5000	0	5	2	0x0C
3	0x00 6000	0	5	0	0x0B
4	0x00 6800	0	5	2	0x0B
5	0x00 7000	0	5	2	0x0B
6	0x00 8000	0	5	2	0x0C
7	0x01 8000	0	3	2	0x0C
8	0x01 9000	0	3	2	0x0C
9	0x01 A000	0	3	2	0x0C
10	0x01 B000	0	3	2	0x0C
11	0x01 C000	0	3	2	0x0C
12	0x01 D000	0	3	2	0x0C
13	0x01 E000	0	3	2	0x0C
14	0x10 0000	1	5	2	0x0C
15	0x01 F000	0	5	2	0x0C
16	0x01 0000	0	3	2	0x0E
17	0x10 6000	2	5	2	0x0C
18	0x10 8000	2	5	2	0x0C
19	0x10 4000	2	2	2	0X0D
20	0x10 8800	2	5	2	0x0B
21	0x10 9000	2	5	2	0x0C
22	0x10C000	2	1	2	0x0D
23	0x11 0000	2	5	2	0x10
24	0x12 0000	2	5	2	0x10
25	0x13 0000	2	5	2	0x0C

Table 5-232. Reset Values for WKPUP_AP L4_AP_REGION_I_L and L4_AP_REGION_I_H

y	BASE	SEGMENT_ID	PROT_GROUP_ID	BYTE_DATA_WIDTH_EXP	SIZE
0	0x00 0000	1	0	2	0x0B
1	0x00 6000	0	0	2	0x0D
2	0x00 9000	0	0	2	0x0C
3	0x00 C000	0	0	2	0x0C
4	0x00 D000	0	0	2	0x0C
5	0x01 8000	0	0	2	0x0C
6	0x01 9000	0	0	2	0x0C
7	0x01 4000	0	0	2	0x0C
8	0x01 5000	0	0	2	0x0C
9	0x00 9000	1	0	2	0x0C
10	0x00 4000	0	0	2	0x0C
11	0x00 5000	0	0	2	0x0C
12	0x00 8800	1	0	2	0x0B
13	0x00 8000	0	0	2	0x0B
14	0x01 0000	0	0	2	0x0C
15	0x01 1000	0	0	2	0x0C
16	0x00 0000	1	0	2	0x0C
17	0x00 1000	1	0	2	0x0C
18	0x00 A000	1	0	2	0x0C

Interprocessor Communication

This chapter describes the interprocessor communication (IPC) module.

Topic **Page**

6.1	IPC Overview	730
6.2	IPC Integration	731
6.3	IPC Mailbox Functional Description	734
6.4	IPC Mailbox Basic Programming Model	737
6.5	IPC Mailbox Use Cases and Tips	742
6.6	IPC Mailbox Register Manual	748

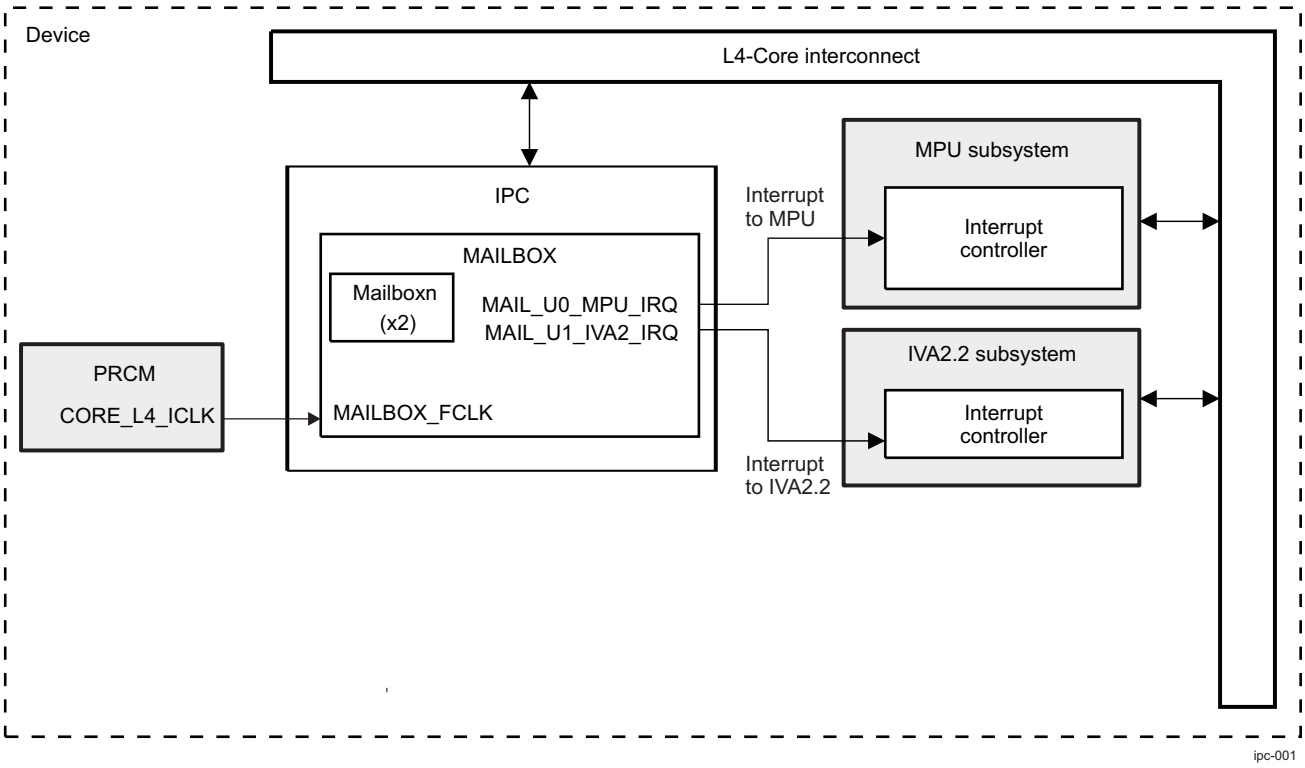
6.1 IPC Overview

Communication between the on-chip processors of the device uses a queued mailbox-interrupt mechanism.

The queued mailbox-interrupt mechanism allows the software to establish a communication channel between two processors through a set of registers and associated interrupt signals by sending and receiving messages (mailboxes).

Figure 6-1 shows a block diagram of the interprocessor communication (IPC) module.

Figure 6-1. Simplified Block Diagram of the IPC



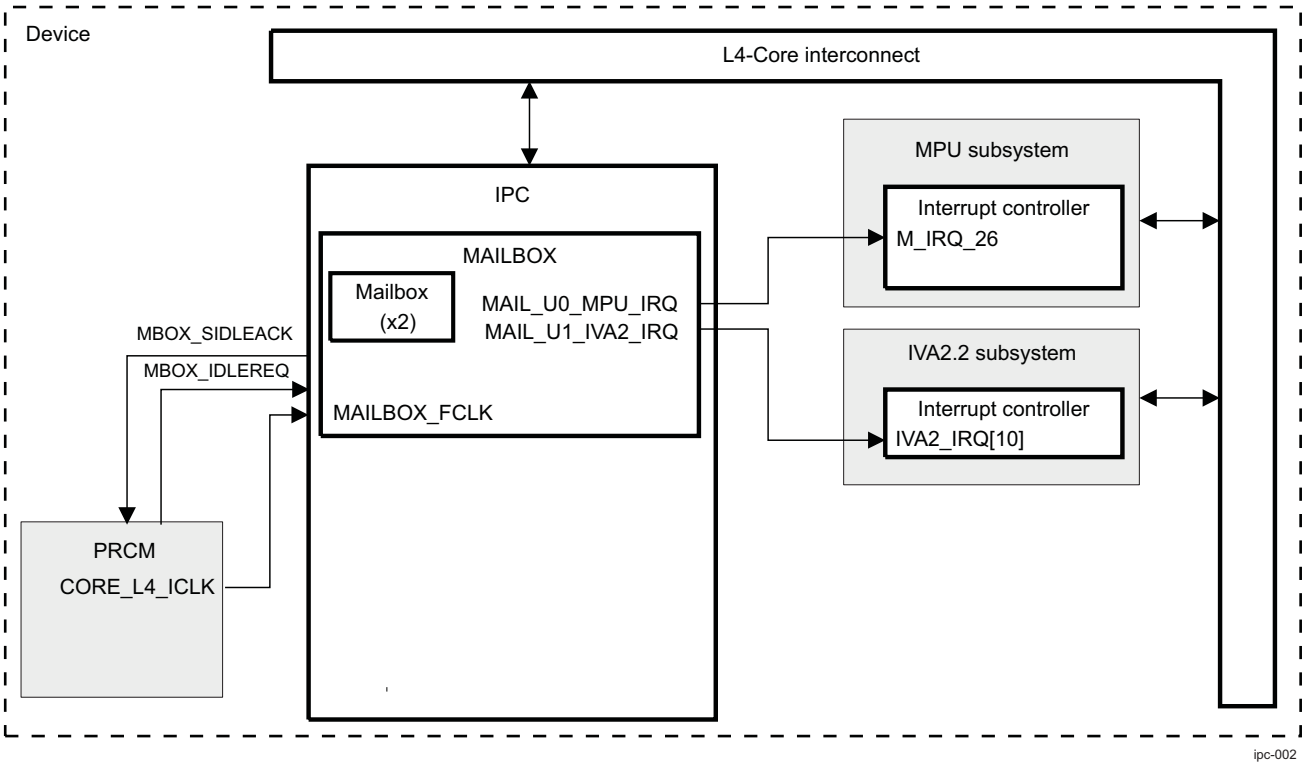
The mailbox module includes these features:

- Two mailbox message queues for microprocessor unit (MPU) and imaging video and audio accelerator (IVA2.2) communications.
- Flexible assignment of receiver and sender for each mailbox through interrupt configuration
- 32-bit message width
- Four-message FIFO depth for each message queue
- Message reception and queue-not-full notification using interrupts
- Support of 16-/32-bit addressing scheme
- Power management support
- Automatic idle mode for power savings

6.2 IPC Integration

Figure 6-2 highlights the IPC integration in the device.

Figure 6-2. IPC Integration



6.2.1 Clocking, Reset, and Power-Management Scheme

6.2.1.1 Clocks

6.2.1.1.1 Module Clocks

The mailbox module receives one input clock, CORE_L4_ICLK, from the power, reset, and clock management (PRCM) module. CORE_L4_ICLK is gated internally and can be turned off to lower operating power when a module is not active. The exact frequency of this clock depends on PRCM programming.

6.2.1.2 Resets

The IPC supports both a hardware reset and a software reset.

6.2.1.2.1 Hardware Reset

The mailbox module receives its reset signal, CORE_RST (the reset signal of the CORE power domain), from the PRCM module.

6.2.1.2.2 Software Reset

The mailbox module supports a software reset by accessing the MAILBOX.MAILBOX_SYSCONFIG[1] SOFTRESET bit (0: normal mode, 1: module is reset).

6.2.1.3 Power Domains

The mailbox module connects to the CORE power domain.

6.2.1.4 Power Management

6.2.1.4.1 System Power Management

This section describes system power management for the mailbox module.

As part of the system-wide power-management scheme, the mailbox module supports a communication protocol with the PRCM that allows the PRCM to request the mailbox to enter a low-power state. When the mailbox module acknowledges a low-power-mode request from the PRCM, the clock to the module is gated off at the PRCM clock generator. Because the clock is disabled at the source, the low-power mode offers lower power consumption than the internal clock-gating method in the local power management.

The PRCM.CM_ICLKEN1_CORE[7] EN_MAILBOXES bit in the PRCM module controls the mailbox clock. When this bit is 1, the clock to the mailbox module is enabled; otherwise, the clock is disabled (see Chapter 4, *Power, Reset, and Clock Management*, for more information).

The mailbox module can be configured using the MAILBOX.MAILBOX_SYSCONFIG[4:3] SIDLEMODE field to one of the following acknowledgment modes:

- No-idle mode: The mailbox module never enters the idle state.
- Force-idle mode: The mailbox module immediately enters the idle state on receiving a low-power-mode request from the PRCM module. In this mode, the software must ensure that there are no asserted output interrupts before requesting this mode to go into the idle state.
- Smart-idle mode: After receiving a low-power-mode request from the PRCM module, the mailbox module enters the idle state only after all asserted output interrupts are acknowledged.

Table 6-1 describes the mailbox power-management modes.

Table 6-1. Mailbox Power Management Modes

Power-Management Mode Requested by the PRCM	MAILBOX.MAILBOX_SYSCONFIG[4:3] SIDLEMODE Bit Field (Offset: 0x010)
Force-idle	00
No-idle	01
Smart-idle	10
Reserved (not used)	11

Note: The mailbox idle status can be read from the PRCM.CM_IDLEST1_CORE[7] ST_MAILBOXES bit. When this bit is 0, the mailbox module cannot be accessed; otherwise, the mailbox module can be accessed (see Chapter 4, *Power, Reset, and Clock Management*, for more information).

6.2.1.4.2 Module Power Management

This section describes local power management for the mailbox module.

To conserve power, the mailbox module supports an automatic idle mode whenever no activity is detected on the mailbox L4-Core interconnect interface. The automatic idle mode is enabled or disabled through the MAILBOX.MAILBOX_SYSCONFIG[0] AUTOIDLE bit.

When the MAILBOX.MAILBOX_SYSCONFIG[0] AUTOIDLE bit is asserted, the automatic idle mode is enabled in cases in which no activity is detected on the L4-Core interconnect interface, and the mailbox clock is disabled internally to the module, thus reducing power consumption.

When new activity is detected on the L4-Core interconnect interface, the clock is restarted with no latency penalty. After reset, the automatic idle mode is disabled; therefore, it is recommended that software enable the automatic idle mode for reduced power consumption.

Note: The PRCM.CM_AUTOIDLE1_CORE[7] AUTO_MAILBOXES bit controls whether the mailbox interface clock is enabled or disabled in synchronization with the CORE power domain state transition (see Chapter 4, *Power, Reset, and Clock Management*, for more information).

6.2.2 Hardware Requests

6.2.2.1 Interrupt Requests

The mailbox module can generate two interrupts:

- MAIL_U0_MPU_IRQ, mapped on M_IRQ_26 of the MPU subsystem interrupt controller
- MAIL_U1_IVA2_IRQ, mapped on IVA2_IRQ[10] of the IVA2.2 subsystem interrupt controller

Each interrupt allows the user (MPU subsystem or IVA2.2 subsystem) of the mailbox to be notified when a message is received or when the message queue is not full. There is one interrupt per user.

6.2.2.2 Idle Handshake Protocol

The PRCM module handles an idle handshake protocol for the mailbox module. The PRCM requires the mailbox module to enter idle mode. The mailbox module acknowledges when it is ready.

6.3 IPC Mailbox Functional Description

Note: In the mailbox functional description, *u* is the user number from 0 to 1 and *m* is the mailbox number from 0 to 1.

The mailbox module provides a means of communication through message queues among the MPU and the IVA2.2. The two individual mailbox modules, or FIFOs, can associate with any of the processors using the MAILBOX.MAILBOX_IRQENABLE_u registers.

The mailbox module includes the following two user subsystems:

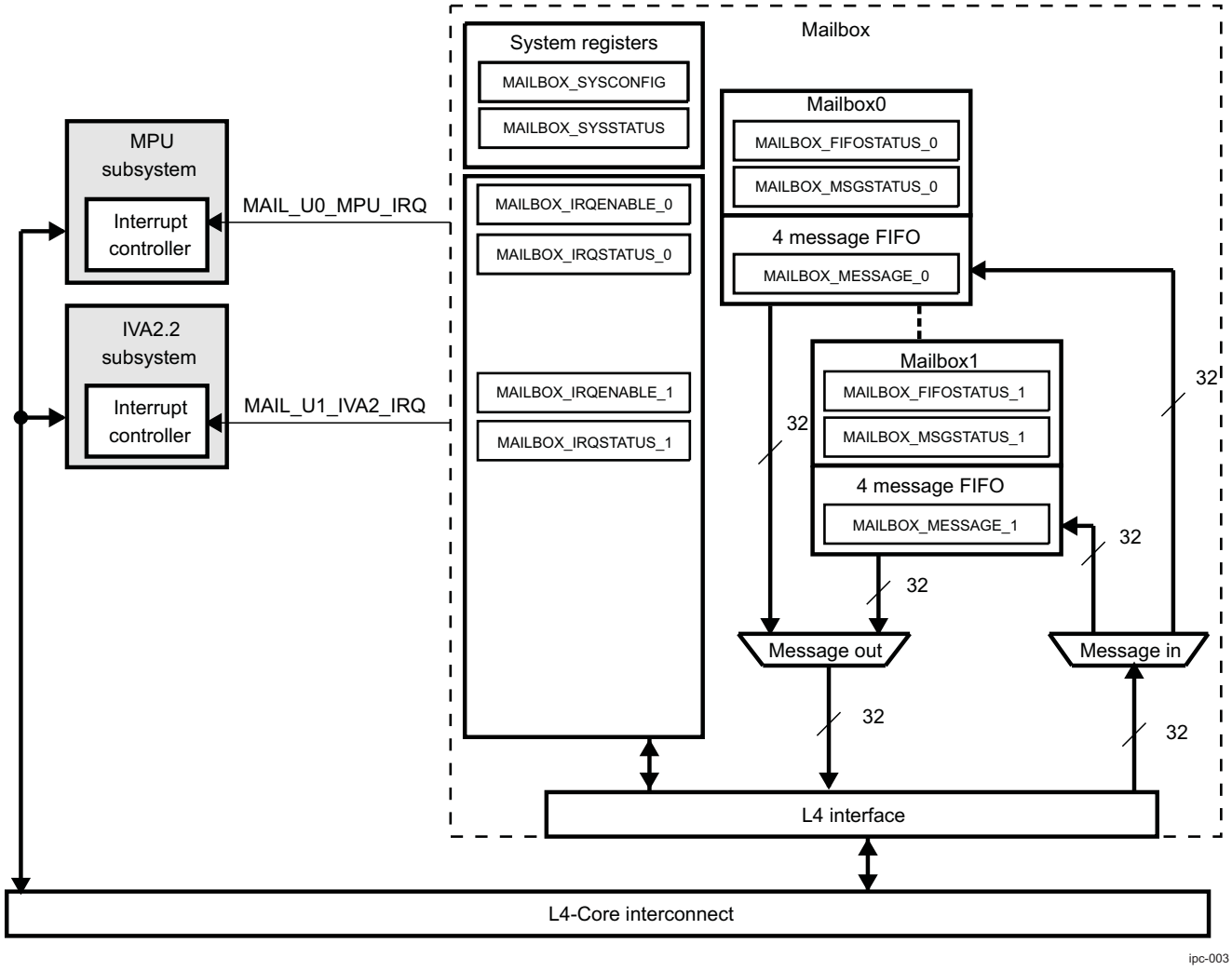
- User 0: MPU subsystem (u = 0)
- User 1: IVA2.2 subsystem (u = 1)

Each user has a dedicated interrupt signal from the mailbox module and a dedicated pair of interrupt enabling and status registers. Each MAILBOX.MAILBOX_IRQSTATUS_u interrupt status register corresponds to a particular user. A user can query its interrupt status register through the L4-Core interconnect.

6.3.1 Block Diagram

Figure 6-3 shows the mailbox block diagram.

Figure 6-3. Mailbox Block Diagram



6.3.2 Mailbox Assignment

6.3.2.1 Description

To assign a receiver to a mailbox, set the new message interrupt enable bit corresponding to the desired mailbox in the MAILBOX.MAILBOX_IRQENABLE_u register. The receiver reads the MAILBOX.MAILBOX_MESSAGE_m register to retrieve a message from the mailbox.

An alternate method for the receiver that does not use the interrupts is to poll the MAILBOX.MAILBOX_FIFOSTATUS_m and/or MAILBOX.MAILBOX_MSGSTATUS_m registers to know when to send or retrieve a message to or from the mailbox. This method does not require assigning a receiver to a mailbox. Because this method does not include the explicit assignment of the mailbox, the software must avoid having multiple receivers use the same mailbox, which can result in incoherency.

To assign a sender to a mailbox, set the queue-not-full interrupt enable bit of the desired mailbox in the MAILBOX.MAILBOX_IRQENABLE_u register, where *u* is the number of the receiving user. However, direct allocation of a mailbox to a sender is not recommended because it can cause the sending processor to be constantly interrupted.

It is recommended that register polling be used to:

- Check the status of either the MAILBOX.MAILBOX_FIFOSTATUS_m or MAILBOX.MAILBOX_MSGSTATUS_m registers
- Write the message to the corresponding MAILBOX.MAILBOX_MESSAGE_m register, if space is available.

The sender should use the queue-not-full interrupt when the initial mailbox status check indicates the mailbox is full. In this case, the sender can enable the queue-not-full interrupt for its mailbox in the appropriate MAILBOX.MAILBOX_IRQENABLE_u register. This allows the sender to be notified by interrupt only when a FIFO queue has at least one available entry.

Reading the MAILBOX.MAILBOX_IRQSTATUS_u register determines the status of the new message and the queue-not-full interrupts for a particular user. Writing 1 to the corresponding bit in the same register location acknowledges, and subsequently clears, an interrupt.

CAUTION

Assigning multiple senders or multiple receivers to the same mailbox is not recommended.

6.3.3 Sending and Receiving Messages

6.3.3.1 Description

When a 32-bit message is written to the MAILBOX.MAILBOX_MESSAGE_m register, the message is appended into the FIFO queue. This queue holds four messages. If the queue is full, the message is discarded.

Queue overflow can be avoided by first reading the MAILBOX.MAILBOX_FIFOSTATUS_m register to check that the mailbox message queue is not full before writing a new message to it.

Reading the MAILBOX.MAILBOX_MESSAGE_m register returns the message at the beginning of the FIFO queue and removes it from the queue. If the FIFO queue is empty when the MAILBOX.MAILBOX_MESSAGE_m register is read, the value 0 is returned.

The new message interrupt is asserted when at least one message is in the mailbox message FIFO queue. To determine the number of messages in the mailbox message FIFO queue, read the MAILBOX.MAILBOX_MSGSTATUS_m register.

6.3.4 16-Bit Register Access

6.3.4.1 Description

So that 16-bit processors can access the mailbox module, the device allows 16-bit register read and write access, with restrictions for the MAILBOX.MAILBOX_MESSAGE_m registers. The 16-bit half-words are organized in little endian fashion; that is, the least-significant 16 bits are at the low address and the most-significant 16 bits are at the high address (low address + 0x02).

All mailbox module registers can be read or written to directly using individual 16-bit accesses with no restriction on interleaving, except the MAILBOX.MAILBOX_MESSAGE_m registers, which must always be accessed by either single 32-bit accesses or two consecutive 16-bit accesses.

CAUTION

When using 16-bit accesses, it is critical to ensure that the mailbox used has only one assigned receiver and only one assigned sender.

When using 16-bit accesses to the MAILBOX.MAILBOX_MESSAGE_m registers, the order of access must be the least-significant half-word first (low address) and the most-significant half-word last (high address). This requirement is due to the update operation by the message FIFO of the MAILBOX.MAILBOX_MSGSTATUS_m registers. The update of the FIFO queue contents and the associated status registers and possible interrupt generation occurs only when the most-significant 16 bits of a MAILBOX.MAILBOX_MESSAGE_m are accessed.

6.4 IPC Mailbox Basic Programming Model

6.4.1 Initialization Flow for the Mailbox Module

The initialization flow for the mailbox module consists of the following steps:

1. Perform a software reset of the mailbox module (see Section 6.4.1.1, *Software Reset*).
2. Set the idle mode and clock configuration of the mailbox module (see Section 6.4.1.2, *Idle Mode and Clock Configuration*).

6.4.1.1 Software Reset

To perform a software reset, write 1 in the MAILBOX.MAILBOX_SYSCONFIG[1] SOFTRESET bit. The MAILBOX.MAILBOX_SYSSTATUS[0] RESETDONE bit indicates that the software reset is complete when its value is 1.

When the software reset completes, the MAILBOX.MAILBOX_SYSCONFIG[1] SOFTRESET bit is automatically reset. The software must ensure that the software reset completes before doing mailbox operations.

CAUTION

When performing a software reset by writing 1 in the MAILBOX.MAILBOX_SYSCONFIG[1] SOFTRESET bit, 0 must be written in the other bits of the MAILBOX.MAILBOX_SYSCONFIG register.

6.4.1.2 Idle Mode and Clock Configuration

The idle mode and clock configuration is done by setting the MAILBOX.MAILBOX_SYSCONFIG[4:3] SIDLEMODE field and the MAILBOX.MAILBOX_SYSCONFIG[0] AUTOIDLE bit (see Section 6.6, *Mailbox Register Manual*, for more information).

6.4.2 Mailbox Assignment

Before communicating, mailboxes can be explicitly assigned to a user using the appropriate MAILBOX.MAILBOX_IRQENABLE_u register. The software must ensure that only one sender and one receiver are assigned per mailbox.

For example, to assign mailbox 1 (m = 1) to the MPU (u = 0, see Section 6.3, *Mailbox Functional Description*, for the user number) as a receiver, set the MAILBOX.MAILBOX_IRQENABLE_0[2] NEWMSGENABLEUUMB1 bit to generate an interrupt to the MPU when a new message is received in mailbox 1.

To assign mailbox 0 (m = 0) to the MPU (u = 0) as a sender, set the MAILBOX.MAILBOX_IRQENABLE_0[1] NOTFULLENABLEUUMB0 bit to generate an interrupt to the MPU when the message queue of mailbox 0 is not full.

6.4.3 Mailbox Communication Preparation

Before communicating with another user, the sender must first use one of the following methods to determine that the mailbox message FIFO queue is not full:

- Poll the MAILBOX.MAILBOX_FIFOSTATUS_m[0] FIFOFULLMB bit or the MAILBOX.MAILBOX_MSGSTATUS_m[2:0] NBOFMSGMB field to determine if there is an open slot available to write a message.

- If the queue-not-full interrupt is enabled by setting the corresponding bit in the MAILBOX.MAILBOX_IRQENABLE_u register, an interrupt to the sender indicates that the mailbox has an available slot. To avoid continuous interrupt to the sender, it is recommended that the software waits until the message queue is full by reading the MAILBOX.MAILBOX_MSGSTATUS_m[2:0] NBOFMSGMB field before enabling the interrupt in the MAILBOX.MAILBOX_IRQENABLE_u register. When a queue-not-full interrupt is generated to the sender, the interrupt should be disabled until the message queue is full for the same reason.

The receiver can also detect new messages from another user using two methods:

- Poll the MAILBOX.MAILBOX_FIFOSTATUS_m[0] FIFOFULLMB bit or the MAILBOX.MAILBOX_MSGSTATUS_m[2:0] NBOFMSGMB field.
- Use a new message ISR (interrupt service routine). In this case, the receiver must enable the appropriate interrupt in the MAILBOX.MAILBOX_IRQENABLE_u register.

Note: After an interrupt is generated, and before exiting the ISR, write 1 in each bit responsible for this generation in the MAILBOX.MAILBOX_IRQSTATUS_u register, thereby clearing these bits.

6.4.4 Mailbox Communication Sequence

When a message slot is available in the mailbox, a message can be transmitted by a sender to a receiver using the following steps:

1. The sender writes a message in the MAILBOX.MAILBOX_MESSAGE_m register. This results in the following actions:
   - The message is stored at the tail of the FIFO queue of mailbox *m*, and the MAILBOX.MAILBOX_FIFOSTATUS_m and MAILBOX.MAILBOX_MSGSTATUS_m registers are updated.
   - If the FIFO queue was previously empty, a new message interrupt can be generated to the receiver to which the mailbox is allocated; otherwise, the interrupt is already asserted and remains so.
2. The receiver can either use an ISR or poll the MAILBOX.MAILBOX_FIFOSTATUS_m or MAILBOX.MAILBOX_MSGSTATUS_m registers to detect new messages and read them by accessing the MAILBOX.MAILBOX_MESSAGE_m register.
   - If using interrupts, the receiver enters the ISR when it detects the new message interrupt. The receiver checks both the MAILBOX.MAILBOX_IRQSTATUS_u register to determine the source of the interrupt and the MAILBOX.MAILBOX_MSGSTATUS_m register(s) to determine the number of messages in the FIFO queue.
   - The receiver can poll the appropriate MAILBOX.MAILBOX_MSGSTATUS_m register(s) to check the status and determine if there are any pending messages to read. The receiver can read the MAILBOX.MAILBOX_MSGSTATUS_m[2:0] NBOFMSGMB field to determine how many messages are available.
3. Using either ISR or polling method, when the receiver determines that it has a message pending in a mailbox, it repeatedly reads the MAILBOX.MAILBOX_MESSAGE_m register to remove all messages from the FIFO queue until a read in the MAILBOX.MAILBOX_MSGSTATUS_m register indicates no more messages are available (MAILBOX.MAILBOX_MSGSTATUS_m[2:0] NBOFMSGMB field = 0x00).
4. After reading all of the messages, the receiver can acknowledge the new message interrupt by writing 1 in the appropriate bit of the MAILBOX.MAILBOX_IRQSTATUS_u register to clear the interrupt flag before exiting the ISR.

6.4.5 Example of Communication

This example shows how communication is established between the MPU and IVA2.2 subsystems in the device. The MPU subsystem sends messages to the IVA2.2 subsystem through mailbox 0, and the IVA2.2 subsystem sends messages to the MPU subsystem through mailbox 1.

To establish communication, the software follows these steps:

1. Turn on the automatic idle feature by writing 1 in the MAILBOX.MAILBOX_SYSCONFIG[0] AUTOIDLE bit.
2. Configure the mailbox in smart-idle mode by setting the MAILBOX.MAILBOX_SYSCONFIG[4:3] SIDLEMODE field in smart-idle mode (smart-idle is the recommended mode; see Section 6.6, *Mailbox Register Manual*, for more information). Smart-idle mode allows the PRCM low-power-mode requests to be acknowledged only after clearing any pending interrupts.
3. Write 1 in the MAILBOX.MAILBOX_IRQENABLE_1[0] NEWMSGENABLEUUMB0 bit to enable interrupts to the IVA2.2 subsystem when a new message is received in mailbox 0.
4. Write 1 in the MAILBOX.MAILBOX_IRQENABLE_0[2] NEWMSGENABLEUUMB1 bit to enable interrupts to the MPU subsystem when a new message is received in mailbox 1.
5. Enable interrupts in the corresponding subsystems.

6.4.5.1 Sending a Message (Polling Method)

To send a message using the polling method, the MPU or IVA2.2 subsystem follows these steps:

1. The MPU subsystem (or IVA2.2 subsystem) determines if the message queue of mailbox 0 (or mailbox 1 for the IVA2.2 subsystem) is full by reading the MAILBOX.MAILBOX_FIFOSTATUS_0[0] FIFOFULLMB bit (or the MAILBOX.MAILBOX_FIFOSTATUS_1[0] FIFOFULLMB bit for the IVA2.2 subsystem).
2. If the MAILBOX.MAILBOX_FIFOSTATUS_0[0] FIFOFULLMB bit (or the MAILBOX.MAILBOX_FIFOSTATUS_1[0] FIFOFULLMB bit for the IVA2.2 subsystem) is 0, mailbox 0 (or mailbox 1 for the IVA2.2 subsystem) has a message slot available to store a new message; go to step 4.
3. If the MAILBOX.MAILBOX_FIFOSTATUS_0[0] FIFOFULLMB bit (or the MAILBOX.MAILBOX_FIFOSTATUS_1[0] FIFOFULLMB bit for the IVA2.2 subsystem) is 1, mailbox 0 (or mailbox 1 for the IVA2.2 subsystem) is full; go back to step 1.
4. The MPU subsystem (or IVA2.2 subsystem) can send a message by writing it into the MAILBOX.MAILBOX_MESSAGE_0 register (or the MAILBOX.MAILBOX_MESSAGE_1 register for the IVA2.2 subsystem).

6.4.5.2 Sending a Message (Interrupt Method)

To send a message using the interrupt method, the MPU or IVA2.2 subsystem follows these steps:

1. To avoid continuous interruption, the MPU subsystem (or the IVA2.2 subsystem) must determine if mailbox 0 (or mailbox 1 for the IVA2.2 subsystem) is full by reading the MAILBOX.MAILBOX_FIFOSTATUS_0[0] FIFOFULLMB bit (or the MAILBOX.MAILBOX_FIFOSTATUS_1[0] FIFOFULLMB bit for the IVA2.2 subsystem).
2. If mailbox 0 (or mailbox 1 for the IVA2.2 subsystem) is full, the MPU subsystem (or the IVA2.2 subsystem) can enable the queue-not-full interrupt by setting the MAILBOX.MAILBOX_IRQENABLE_0[1] NOTFULLENABLEUUMB0 bit (or the MAILBOX.MAILBOX_IRQENABLE_1[3] NOTFULLENABLEUUMB1 bit for the IVA2.2 subsystem), and perform another task before an interrupt occurs; go to step 4).
3. If mailbox 0 (or mailbox 1 for the IVA2.2 subsystem) is not full, the MPU can go back to step 1 and wait for mailbox 0 (or mailbox 1 for the IVA2.2 subsystem) to fill, or the MPU can send a message, if necessary, by writing in the MAILBOX.MAILBOX_MESSAGE_0 register (or the MAILBOX.MAILBOX_MESSAGE_1 register for the IVA2.2 subsystem).
4. After receiving an interrupt, the MPU subsystem (or the IVA2.2 subsystem) enters the ISR and reads the MAILBOX.MAILBOX_IRQSTATUS_0[1] NOTFULLSTATUSUUMB0 bit (or the MAILBOX.MAILBOX_IRQSTATUS_1[3] NOTFULLSTATUSUUMB1 bit for the IVA2.2 subsystem) to determine if mailbox 0 (or mailbox 1 for the IVA2.2 subsystem) is not full and thus send its message.

   The MPU subsystem writes the message in the MAILBOX.MAILBOX_MESSAGE_0 register (or the MAILBOX.MAILBOX_MESSAGE_1 register for the IVA2.2 subsystem).

   The MPU subsystem (or the IVA2.2 subsystem) then acknowledges the interrupt by writing 1 in the MAILBOX.MAILBOX_IRQSTATUS_0[1] NOTFULLSTATUSUUMB0 bit (or the MAILBOX.MAILBOX_IRQSTATUS_1[3] NOTFULLSTATUSUUMB1 bit for the IVA2.2 subsystem).

Note: To send several messages, a subsystem or processor must determine if the message queue of mailbox m has enough available slots by checking the MAILBOX.MAILBOX_MSGSTATUS_m[2:0] NBOFMSGMB bit field before writing all of the messages in this mailbox (see Section 6.6, *Mailbox Register Manual*, for more information).

6.4.5.3 Receiving Messages (Interrupt Method)

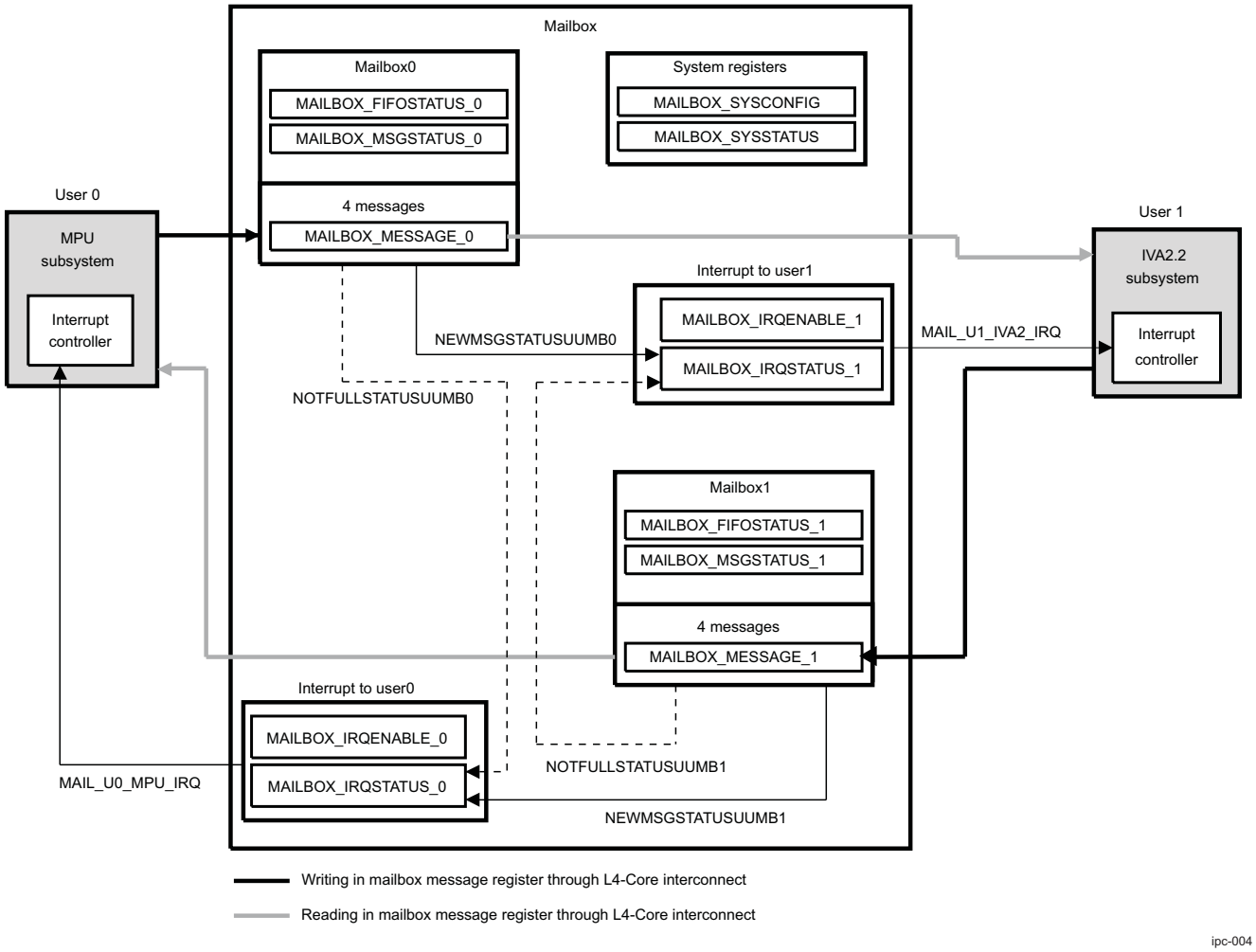
After receiving an interrupt indicating that a new message is received, the MPU or IVA2.2 subsystem follows these steps:

1. The MPU subsystem (or the IVA2.2 subsystem) determines how many messages are stored in the message queue of mailbox 1 by reading the MAILBOX.MAILBOX_MSGSTATUS_1[2:0] NBOFMSGMB field (or the MAILBOX.MAILBOX_MSGSTATUS_0[2:0] NBOFMSGMB bit field for the IVA2.2 subsystem).
2. The MPU subsystem (or the IVA2.2 subsystem) reads the MAILBOX.MAILBOX_MESSAGE_1 register (or the MAILBOX.MAILBOX_MESSAGE_0 register for the IVA2.2 subsystem) as many times as there are messages in mailbox 1 (or mailbox 0 for the IVA2.2 subsystem).
3. Finally, the MPU subsystem (or the IVA2.2 subsystem) acknowledges the interrupt by writing 1 in the MAILBOX.MAILBOX_IRQSTATUS_0[2] NEWMSGSTATUSUUMB1 bit (or the MAILBOX.MAILBOX_IRQSTATUS_1[0] NEWMSGSTATUSUUMB0 bit for the IVA2.2 subsystem).

Note: After the interrupt is acknowledged, if the mailbox message queue is not empty, the interrupt is reasserted.

Figure 6-4 shows an example of communication:

Figure 6-4. Example of Communication



6.5 IPC Mailbox Use Cases and Tips

6.5.1 Camcorder Use Case: How to Configure the Mailbox Module for Communication Between the MPU and the IVA2.2 Subsystems

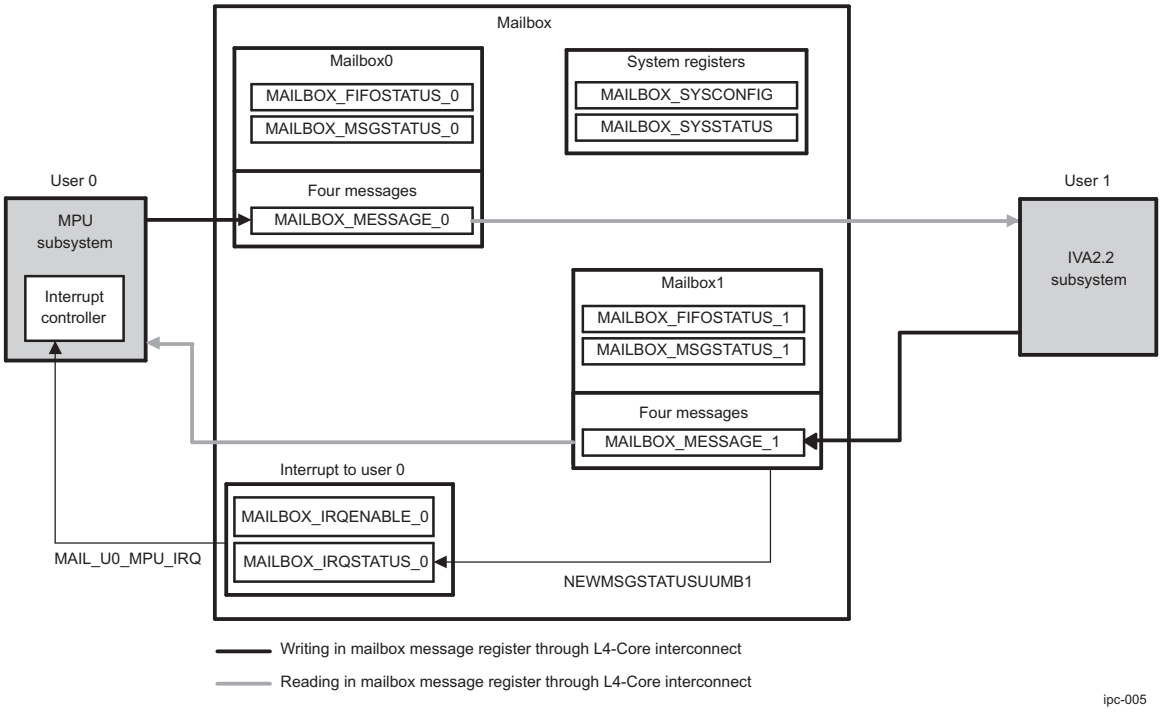
6.5.1.1 Overview

In a whole Camcorder use case, the mailbox module is in charge of the communication between the MPU and the IVA2.2 subsystems in the device.

The MPU subsystem send messages to the IVA2.2 subsystem through mailbox 0: typically, an audio or video compression request (a command and a memory address). And the IVA2.2 subsystem sends messages to the MPU subsystem through mailbox 1: typically, a compression completion message with error management.

Figure 6-5 shows an overview of the use of the mailbox module in the case of the Camcorder use case.

Figure 6-5. Overview



For the Camcorder use case, the configuration is the following:

- 32-bit message width
- Use of polling method for MPU subsystem message sending (mailbox 0), IVA2.2 subsystem message receiving (mailbox 0), and IVA2.2 subsystem message sending (mailbox 1)
- Use of interrupt method for MPU subsystem message receiving (mailbox 1)
- NewMessage interrupt to the MPU subsystem indicating a new message is received on mailbox 1

6.5.1.2 Programming Flow

6.5.1.2.1 Initial Configuration

The initialization of the mailbox module is done in two parts:

1. Mailbox module initialization
2. Interrupt enabling

The following two subsections describe these initialization parts.

See Figure 6-6 for the corresponding programming flow chart.

6.5.1.2.1.1 Mailbox Module Initialization

To initialize the mailbox module, the following steps must be done:

1. Enable the clock for the mailbox module (set the PRCM.CM_ICLKEN1_CORE[7] EN_MAILBOXES bit to 1)
2. Set the MAILBOX.MAILBOX_SYSCONFIG[1] SOFTRESET bit to 1. This bit is automatically reset by the hardware
3. Poll the MAILBOX.MAILBOX_SYSSTATUS[0] RESETDONE bit until it is set to 1 by the mailbox module to indicate that the mailbox module is complete

Table 6-2 shows all the registers to be configured for the mailbox module initialization.

Table 6-2. Register Print After the Mailbox Module Initialization

Register Name	Physical Address	Value	Value Description
PRCM.CM_ICLKEN1_CORE	0x4800 4A10	0x0000 0080	EN_MAILBOXES bit set to 1 to enable the mailbox clock
MAILBOX.MAILBOX_SYSCONFIG	0x4809 4010	0x0000 0002	SOFTRESET bit set to 1 to reset the mailbox module
MAILBOX.MAILBOX_SYSSTATUS	0x4809 4014	0x0000 0001	The RESETDONE bit is set to 1 by the mailbox module when the software reset is complete

6.5.1.2.1.2 Interrupt Enabling

The NewMessage interrupt is asserted when at least one message is in the mailbox message FIFO queue.

To generate an interrupt to the MPU subsystem when a new message is received on mailbox 1, set the MAILBOX.MAILBOX_IRQENABLE_0[2] NEWMSGENABLEUUMB1 bit to 1.

Table 6-3 shows all the registers to be configured for the interrupt enabling.

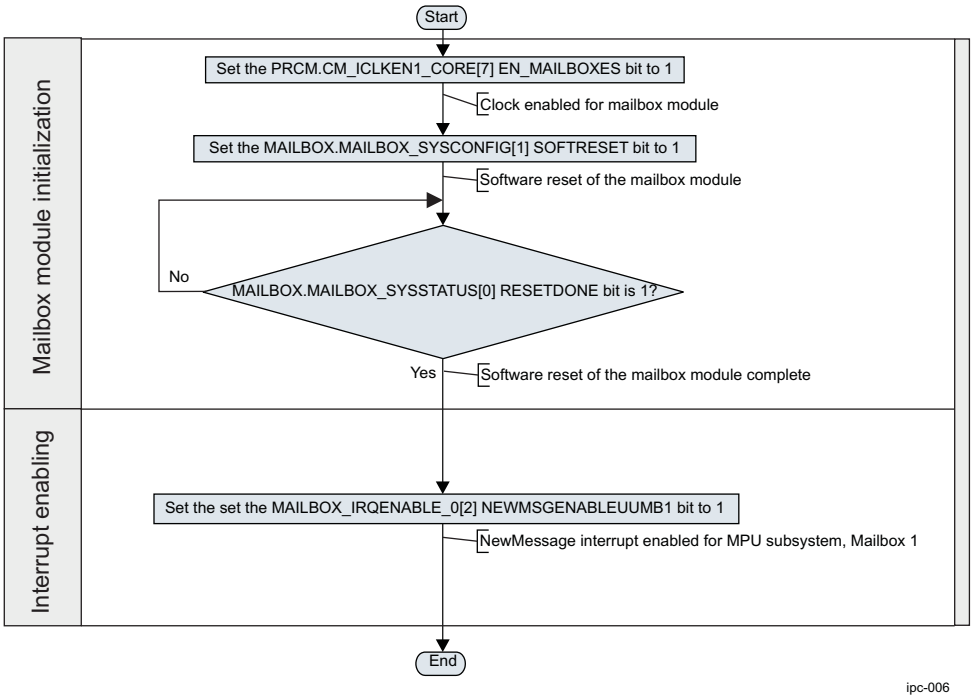
Table 6-3. Register Print After the Interrupt Enabling

Register Name	Physical Address	Value	Value Description
MAILBOX.MAILBOX_ IRQENABLE_0	0x4809 4104	0x0000 0004	NEWMSGENABLEUUMB1 bit is set to 1 to enable the NewMessage interrupt to the MPU subsystem on mailbox 1

6.5.1.2.1.3 Initial Configuration Flow Chart

Figure 6-6 describes the programming flow chart corresponding to the initial configuration.

Figure 6-6. Initial Configuration Flow Chart



6.5.1.2.2 Operational Mode

For the Camcorder use case, the communication between the MPU and the IVA2.2 subsystems is done in four parts:

1. MPU subsystem message sending (polling method)
2. IVA2.2 subystem message receiving (polling method)
3. IVA2.2 subsystem message sending (polling method)
4. MPU subsystem message receiving (interrupt method)

6.5.1.2.2.1 MPU Subsystem Message Sending (Polling Method)

To send a message with mailbox 0 using the polling method, the MPU subsystem follows these steps:

1. Poll the MAILBOX.MAILBOX_FIFOSTATUS_0[0] FIFOFULLMB bit until it is cleared to 0 by the mailbox module to indicate that the mailbox 0 message queue is not full
2. Write the message into the MAILBOX.MAILBOX_MESSAGE_0 register

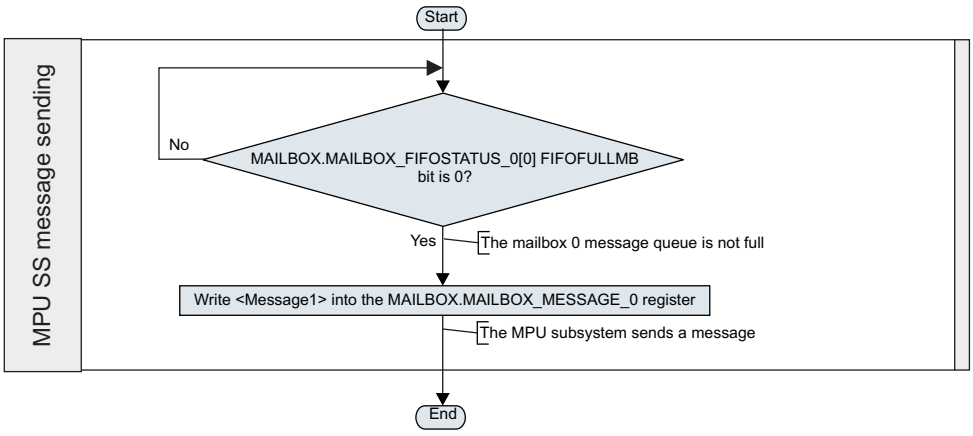
Table 6-4 shows all the registers to be configured for the MPU subsystem message sending.

Table 6-4. Register Print After the MPU Subsystem Message Sending

Register Name	Physical Address	Value	Value Description
MAILBOX.MAILBOX_ FIFOSTATUS_0	0x4809 4080	0x0000 0000	FIFOFULLMB bit is cleared to 0 when the mailbox 0 message queue is not full
MAILBOX.MAILBOX_ MESSAGE_0	0x4809 4040	<Message1>	<Message1> is written into the MAILBOX.MAILBOX_MESSAGE_0 register

Figure 6-7 describes the programming flow chart corresponding to the MPU subsystem message sending.

Figure 6-7. MPU Subsystem Message Sending Flow Chart



6.5.1.2.2.2 IVA2.2 Subsystem Message Receiving (Polling Method)

To receive a message with mailbox 0 using the polling method, the IVA2.2 subsystem follows these steps:

1. Poll the MAILBOX.MAILBOX_MSGSTATUS_0[2:0] NBOFMSGMB field until it is set to 0x1 to detect a new message in mailbox 0
2. Read the message into the MAILBOX.MAILBOX_MESSAGE_0 register

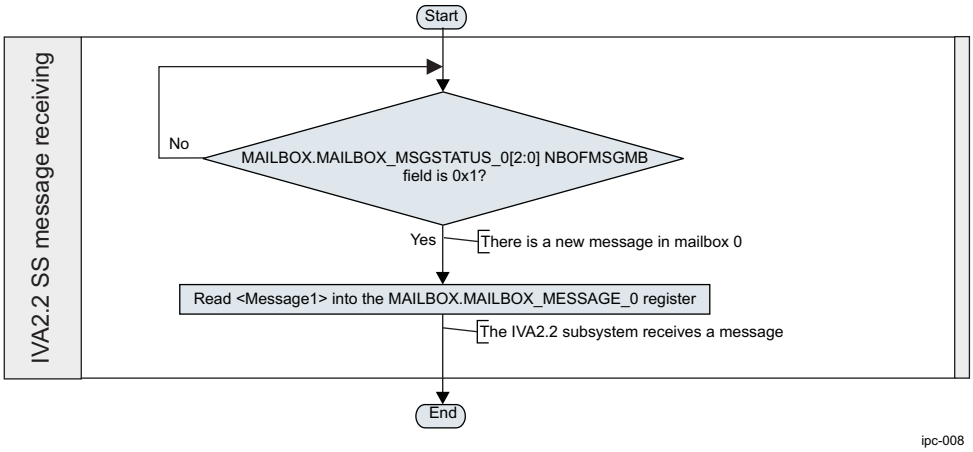
Table 6-5 shows all the registers to be configured for the IVA2.2 subsystem message receiving.

Table 6-5. Register Print After the IVA2.2 Subsystem Message Receiving

Register Name	Physical Address	Value	Value Description
MAILBOX.MAILBOX_ MSGSTATUS_0	0x4809 40C0	0x0000 0001	NBOFMSGMB field is set to 0x1. <Message1> is received.
MAILBOX.MAILBOX_ MESSAGE_0	0x4809 4040	<Message1>	<Message1> is read into the MAILBOX.MAILBOX_MESSAGE_0 register

Figure 6-8 describes the programming flow chart corresponding to the IVA2.2 subsystem message receiving. See Section 6.5.1.2.2.2, *IVA2.2 Subsystem Message Receiving (Polling Method)*, for more information.

Figure 6-8. IVA2.2 Subsystem Message Receiving Flow Chart



6.5.1.2.2.3 IVA2.2 Subsystem Message Sending (Polling Method)

When the IVA2.2 subsystem data processing is done, mailbox 1 is used to send a message. After sending the message, a NewMessage interrupt is generated and received by the MPU subsystem. To send a message with mailbox 1 using the polling method, the IVA2.2 subsystem follows these steps:

1. Poll the MAILBOX.MAILBOX_FIFOSTATUS_1[0] FIFOFULLMB bit until it is cleared to 0 by the mailbox module to indicate that the mailbox 0 message queue is not full.
2. Write the message into the MAILBOX.MAILBOX_MESSAGE_1 register.

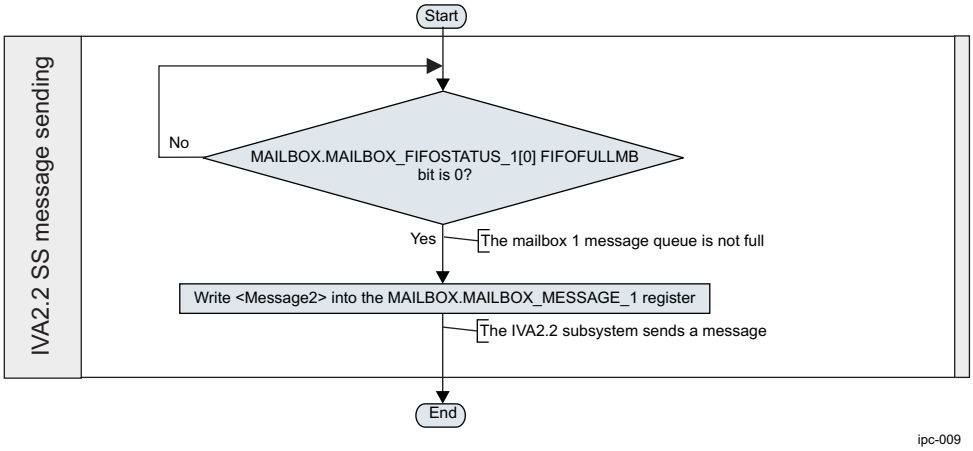
Table 6-6 shows all the registers to be configured for the IVA2.2 subsystem message sending.

Table 6-6. Register Print After the IVA2.2 Subsystem Message Sending

Register Name	Physical Address	Value	Value Description
MAILBOX.MAILBOX_ FIFOSTATUS_1	0x4809 4084	0x0000 0000	FIFOFULLMB bit is cleared to 0 when the FIFO is not full
MAILBOX.MAILBOX_ MESSAGE_1	0x4809 4044	<Message2>	<Message2> is written into the MAILBOX.MAILBOX_MESSAGE_1 register

Figure 6-9 describes the programming flow chart corresponding to the IVA2.2 subsystem message sending. See Section 6.5.1.2.2.3 for more information.

Figure 6-9. IVA2.2 Subsystem Message Sending Flow Chart



6.5.1.2.2.4 MPU Subsystem Message Receiving (Interrupt Method)

After receiving a NewMessage interrupt (indicating a new message is received), the MPU subsystem reads the message from the IVA2.2 subsystem. To receive a message with mailbox 0 using the polling method, the MPU subsystem follows these steps:

1. Read the MAILBOX_IRQSTATUS_0[2] NEWMSGSTATUSUMB1 bit
2. Read the message into the MAILBOX_MESSAGE_1 register
3. Set the MAILBOX_IRQSTATUS_0[2] NEWMSGSTATUSUMB1 bit to 1 to clear the interrupt flag

Table 6-7 shows all the registers to be configured for the MPU subsystem message receiving.

Table 6-7. Register Print After the MPU Subsystetm Message Receiving

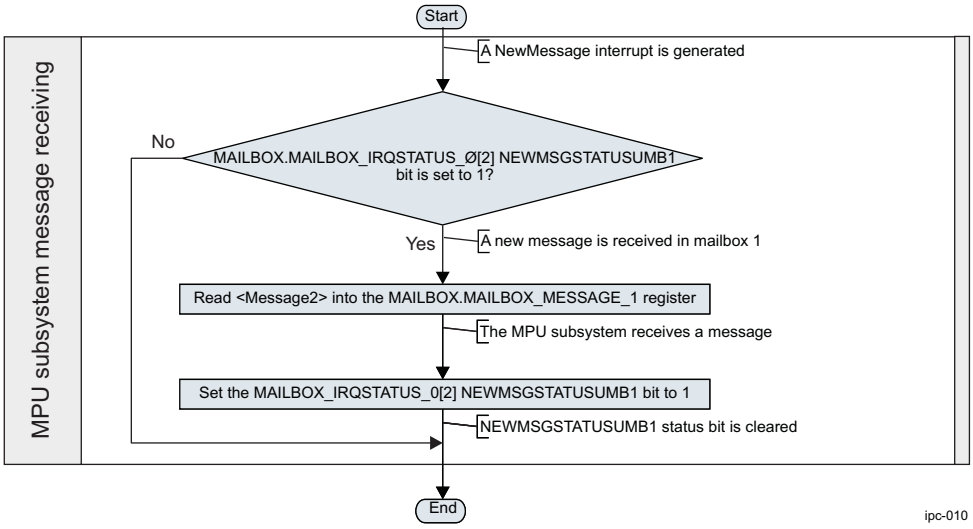
Register Name	Physical Address	Value	Value Description
MAILBOX.MAILBOX_ IRQSTATUS_0	0x4809 4100	0x0000 0004	A NewMessage interrupt is generated. NEWMSGSTATUSUMB1 status bit is set to 1.
MAILBOX.MAILBOX_ MESSAGE_1	0x4809 4044	<Message2>	<Message2> is read into the MAILBOX.MAILBOX_MESSAGE_1 register

Table 6-7. Register Print After the MPU Subsystetm Message Receiving (continued)

Register Name	Physical Address	Value	Value Description
MAILBOX.MAILBOX_ IRQSTATUS_0	0x4809 4100	0x0000 0000	NEWMSGSTATUSUMB1 status bit is cleared

Figure 6-10 describes the programming flow chart corresponding to the MPU subsystem messagr receiving. See Section 6.5.1.2.2.4 for more information.

Figure 6-10. MPU Subsystem Message Receiving Flow Chart



6.6 IPC Mailbox Register Manual

Table 6-8 summarizes the mailbox instance.

Table 6-8. Mailbox Instance Summary

Module Name	Base Address	Size
MLB	0x4809 4000	4K bytes

6.6.1 Mailbox Register Mapping Summary

Table 6-9 summarizes the MLB registers.

Table 6-9. MLB Register Summary

Register Name	Type	Register Width (Bits)	Address Offset	Physical Address
MAILBOX_REVISION	R	32	0x000	0x4809 4000
MAILBOX_SYSCONFIG	RW	32	0x010	0x4809 4010
MAILBOX_SYSSTATUS	R	32	0x014	0x4809 4014
MAILBOX_MESSAGE_m (1)	RW	32	0x040 + (0x04 * m)	0x4809 4040 + (0x04 * m)
MAILBOX_FIFOSTATUS_m (1)	R	32	0x080 + (0x04 * m)	0x4809 4080 + (0x04 * m)
MAILBOX_MSGSTATUS_m (1)	R	32	0x0C0 + (0x04 * m)	0x4809 40C0 + (0x04 * m)
MAILBOX_IRQSTATUS_u (2)	RW	32	0x100 + (0x08 * u)	0x4809 4100 + (0x08 * u)
MAILBOX_IRQENABLE_u (2)	RW	32	0x104 + (0x08 * u)	0x4809 4104 + (0x08 * u)

(1) m = 0 to 1
(2) u = 0 to 1

Note: In MAILBOX_MESSAGE_0, MAILBOX_MESSAGE_1, MAILBOX_FIFOSTATUS_0, MAILBOX_FIFOSTATUS_1, MAILBOX_MSGSTATUS_0, and MAILBOX_MSGSTATUS_1 register names, *0* or *1* is the mailbox number.

In MAILBOX_IRQSTATUS_0, MAILBOX_IRQSTATUS_1, MAILBOX_IRQENABLE_0, and MAILBOX_IRQENABLE_1 register names, *0* or *1* is the user number:

- User 0: MPU subsystem
- User 1: IVA2.2 subsystem

6.6.2 Register Description

Table 6-10 through Table 6-24 describe the register bits.

Table 6-10. MAILBOX_REVISION

Address Offset	0x000		
Physical Address	0x4809 4000	Instance	MLB
Description	This register contains the IP revision code		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																								REV							

Bits	Field Name	Description	Type	Reset
31:8	Reserved	Reads returns 0	R	0x000000
7:0	REV	IP revision [7:4] Major revision [3:0] Minor revision Examples: 0x10 for 1.0, 0x21 for 2.1	R	See(1)

(1) TI internal data

Table 6-11. Register Call Summary for Register MAILBOX_REVISION

Mailbox Register Manual

- Mailbox Register Mapping Summary: [0]

Table 6-12. MAILBOX_SYSCONFIG

Address Offset	0x010		
Physical Address	0x4809 4010	Instance	MLB
Description	This register controls the various parameters of the L4-Core interface		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																							CLOCKACTIVITY	Reserved			SIDLEMODE		Reserved	SOFTRESET	AUTOIDLE

Bits	Field Name	Description	Type	Reset
31:9	Reserved	Write 0's for future compatibility Read returns 0	RW	0x000000
8	CLOCKACTIVITY	Clock activity during wake up mode period Clock can always be switched off and read returns 0	R	0
7:5	Reserved	Write 0's for future compatibility Read returns 0	RW	0x0
4:3	SIDLEMODE		RW	0x0
		0x0: Force-idle. An idle request is acknowledged unconditionally		
		0x1: No-idle. An idle request is never acknowledged		
		0x2: Smart-idle. Acknowledgement to an idle request is given based on the internal activity of the module based on the internal activity of the module		
		0x3: Reserved. Do not use.		
2	Reserved	Write 0's for future compatibility Read returns 0	RW	0
1	SOFTRESET	Software reset. This bit is automatically reset by the hardware. During reads, it always return 0	RW	0
		0x0: Normal mode		
		0x1: The module is reset		
0	AUTOIDLE	Internal interface clock gating strategy	RW	0
		0x0: Interface clock is free-running		
		0x1: Automatic interface clock gating strategy is applied, based on the L4-Core interface activity		

Table 6-13. Register Call Summary for Register MAILBOX_SYSCONFIG

IPC Integration • Software Reset: [0] • System Power Management: [1] [2] • Module Power Management: [3] [4]
Mailbox Basic Programming Model • Software Reset: [5] [6] [7] [8] • Idle Mode and Clock Configuration: [9] [10] • Example of Communication: [11] [12]
Mailbox Use Cases and Tips • Initial Configuration: [13] [14]
Mailbox Register Manual • Mailbox Register Mapping Summary: [15]

Table 6-14. MAILBOX_SYSSTATUS

Address Offset	0x014		
Physical Address	0x4809 4014	Instance	MLB
Description	This register provides status information about the module, excluding the interrupt status information		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																								Reserved							RESETDONE

Bits	Field Name	Description	Type	Reset
31:8	Reserved	Read returns 0	R	0x000000
7:1	Reserved	Read returns 0	R	0x00
0	RESETDONE	Internal reset monitoring	R	1
		Read 0x0: Internal module reset in on-going		
		Read 0x1: Reset completed		

Table 6-15. Register Call Summary for Register MAILBOX_SYSSTATUS

Mailbox Basic Programming Model • Software Reset: [0]
Mailbox Use Cases and Tips • Initial Configuration: [1] [2]
Mailbox Register Manual • Mailbox Register Mapping Summary: [3]

Table 6-16. MAILBOX_MESSAGE_m

Address Offset	0x040 = MAILBOX_MESSAGE_0 for mailbox 0 0x044 = MAILBOX_MESSAGE_1 for mailbox 1
Physical Address	0x4809 4040 = MAILBOX_MESSAGE_0 for mailbox 0 **Instance** MLB 0x4809 4044 = MAILBOX_MESSAGE_1 for mailbox 1
Description	The message register stores the next to be read message of the mailbox X
Type	RW

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MESSAGEVALUEMB																															

Bits	Field Name	Description	Type	Reset
31:0	MESSAGEVALUEMB	Message in Mailbox	RW	0x00000000

Table 6-17. Register Call Summary for Register MAILBOX_MESSAGE_m

Mailbox Functional Description
- Description: [0] [1]
- Description: [2] [3] [4]
- Description: [5] [6] [7] [8]

Mailbox Basic Programming Model
- Mailbox Communication Sequence: [9] [10] [11]

Mailbox Register Manual
- Mailbox Register Mapping Summary: [12]

Table 6-18. MAILBOX_FIFOSTATUS_m

Address Offset	0x080 = MAILBOX_FIFOSTATUS_0 for mailbox 0 0x084 = MAILBOX_FIFOSTATUS_1 for mailbox 1
Physical Address	0x4809 4080 = MAILBOX_FIFOSTATUS_0 for mailbox 0 **Instance** MLB 0x4809 4084 = MAILBOX_FIFOSTATUS_1 for mailbox 1
Description	The FIFO status register has the status related to the mailbox internal FIFO
Type	R

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																															FIFOFULLMB

Bits	Field Name	Description	Type	Reset
31:1	Reserved	Read returns 0	R	0x00000000
0	FIFOFULLMB	Full flag for Mailbox	R	0

Table 6-19. Register Call Summary for Register MAILBOX_FIFOSTATUS_m

Mailbox Functional Description
- Description: [0] [1]
- Description: [2]

Mailbox Basic Programming Model
- Mailbox Communication Preparation: [3] [4]
- Mailbox Communication Sequence: [5] [6]

Mailbox Register Manual
- Mailbox Register Mapping Summary: [7]

Table 6-20. MAILBOX_MSGSTATUS_m

Address Offset	0x0C0 = MAILBOX_MSGSTATUS_0 for mailbox 0 0x0C4 = MAILBOX_MSGSTATUS_1 for mailbox 1		
Physical Address	0x4809 40C0 = MAILBOX_MSGSTATUS_0 for mailbox 0 0x4809 40C4 = MAILBOX_MSGSTATUS_1 for mailbox 1	**Instance**	MLB
Description	The message status register has the status of the messages in the mailbox		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																														NBOFMSGMB	

Bits	Field Name	Description	Type	Reset
31:3	Reserved	Read returns 0	R	0x0000000
2:0	NBOFMSGMB	Number of Messages in Mailbox Note: Limited to four messages per mailbox.	R	0x00

Table 6-21. Register Call Summary for Register MAILBOX_MSGSTATUS_m

Mailbox Functional Description

- Description: [0] [1]
- Description: [2]
- Description: [3]

Mailbox Basic Programming Model

- Mailbox Communication Preparation: [4] [5] [6]
- Mailbox Communication Sequence: [7] [8] [9] [10] [11] [12] [13]
- Sending a Message (Interrupt Method): [14]

Mailbox Register Manual

- Mailbox Register Mapping Summary: [15]

Table 6-22. MAILBOX_IRQSTATUS_u

Address Offset	0x100 = MAILBOX_IRQSTATUS_0 for user 0 0x108 = MAILBOX_IRQSTATUS_1 for user 1		
Physical Address	0x4809 4100 = MAILBOX_IRQSTATUS_0 for user 0 0x4809 4108 = MAILBOX_IRQSTATUS_1 for user 1	**Instance**	MLB
Description	The interrupt status register has the status for each event that may be responsible for the generation of an interrupt to the corresponding user - write 1 to a given bit resets this bit		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																												NOTFULLSTATUSUUMB1	NEWMSGSTATUSUUMB1	NOTFULLSTATUSUUMB0	NEWMSGSTATUSUUMB0

Bits	Field Name	Description	Type	Reset
31:4	Reserved	Write 0s for future compatibility Read returns 0	RW	0x0000000
3	NOTFULLSTATUSUUMB1	NotFull Status bit for User u, Mailbox 1	RW	0
2	NEWMSGSTATUSUUMB1	NewMessage Status bit for User u, Mailbox 1	RW	0
1	NOTFULLSTATUSUUMB0	NotFull Status bit for User u, Mailbox 0	RW	0
0	NEWMSGSTATUSUUMB0	NewMessage Status bit for User u, Mailbox 0	RW	0

Table 6-23. Register Call Summary for Register MAILBOX_IRQSTATUS_u

Mailbox Functional Description • Mailbox Functional Description: [0] • Description: [1]
Mailbox Basic Programming Model • Mailbox Communication Preparation: [2] • Mailbox Communication Sequence: [3] [4]
Mailbox Register Manual • Mailbox Register Mapping Summary: [5]

Table 6-24. MAILBOX_IRQENABLE_u

Address Offset	0x104 = MAILBOX_IRQENABLE_0 for user 0 0x10C = MAILBOX_IRQENABLE_1 for user 1		
Physical Address	0x4809 4104 = MAILBOX_IRQENABLE_0 for user 0 0x4809 410C = MAILBOX_IRQENABLE_1 for user 1	**Instance**	MLB
Description	The interrupt enable register enables to mask/unmask the module internal source of interrupt to the corresponding user		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																												NOTFULLENABLEUUMB1	NEWMSGENABLEUUMB1	NOTFULLENABLEUUMB0	NEWMSGENABLEUUMB0

Bits	Field Name	Description	Type	Reset
31:4	Reserved	Write 0s for future compatibility. Read returns 0	RW	0x0000000
3	NOTFULLENABLEUUMB1	NotFull Enable bit for User u, Mailbox 1	RW	0
2	NEWMSGENABLEUUMB1	NewMessage Enable bit for User u, Mailbox 1	RW	0
1	NOTFULLENABLEUUMB0	NotFull Enable bit for User u, Mailbox 0	RW	0
0	NEWMSGENABLEUUMB0	NewMessage Enable bit for User u, Mailbox 0	RW	0

Table 6-25. Register Call Summary for Register MAILBOX_IRQENABLE_u

Mailbox Functional Description • Mailbox Functional Description: [0] • Description: [1] [2] [3]
Mailbox Basic Programming Model • Mailbox Assignment: [4] • Mailbox Communication Preparation: [5] [6] [7]

Table 6-25. Register Call Summary for Register MAILBOX_IRQENABLE_u (continued)

Mailbox Register Manual • Mailbox Register Mapping Summary: [8]

System Control Module

This chapter describes the system control module (SCM) for the OMAP34xx multimedia device.

Note: This chapter gives information about all modules and features in the high-tier device. See Section 1.5, *OMAP34xx Family*, to check availability of modules and features. Unavailable module and feature pins are not functional.

Topic **Page**

7.1 SCM Overview 756
7.2 SCM Environment 756
7.3 SCM Integration 758
7.4 SCM Functional Description 762
7.5 SCM Programming Model 838
7.6 SCM Register Manual 851

7.1 SCM Overview

The system control module (SCM) allows software control of the various static modes supported by the device. The SCM is on the L4-Core interconnect, but it is sensitive only to the device internal power-on reset. It is not affected by the L4-Core reset.

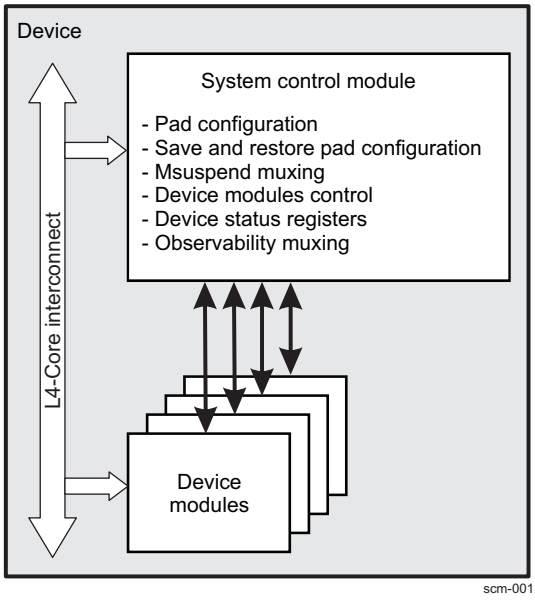
For emulator devices, the power-on reset can also be controlled by the debugger through the JTAG interface.

The device uses the SCM as the primary point of control for these areas:

- Functional I/O multiplexing
- Pad configuration (pull-up or pull-down enable)
- Device status
- Peripheral sensitivity to the MPU and/or the DSP (IVA2.2) MSuspend signals
- Static device configuration
- Debug and observability I/O multiplexing
- Save-and-restore pad configuration

Figure 7-1 provides an overview of the SCM.

Figure 7-1. SCM Overview



The SCM primarily implements a bank of registers accessible by the software. Some are read-only registers that carry status information, and others are fully accessible (read/write).

The read/write registers are divided into the following classes:

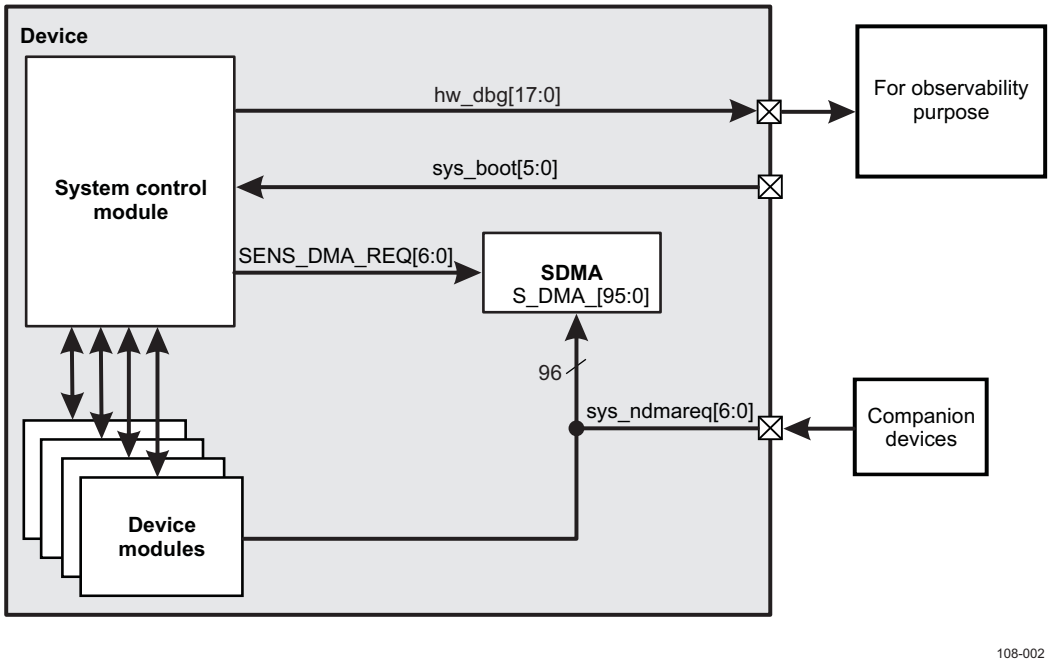
- Pad functional multiplexing and configuration registers (32-bit registers, one register per two pins)
- Debug and observability I/O multiplexing register (32-bit read/write register)
- Static device configuration registers (32-bit read/write registers for module specific configuration)
- Scratchpad memory bank (256* 32-bit)

7.2 SCM Environment

The SCM allows the setting of external system DMA (sDMA) request pin sensitivity and controls the multiplexing of device internal modules signals routed to external pins for hardware debug purposes. The SCM also integrates the decoding logic of sys_boot[5:0].

Figure 7-2 shows an overview of the SCM environment.

Figure 7-2. SCM Environment Overview



The seven sys_ndamreq[6:0] pins are optional external direct memory access (DMA) requests that can be managed directly by the sDMA controller. The SCM can configure these requests to either level sensitive (active low) or edge sensitive (falling edge) through the correct setting of the SENSDMAREQN bit (where N is between 0 and 6) in the CONTROL.CONTROL_DEVCONF0 and CONTROL.CONTROL_DEVCONF1 registers.

The sys_boot[5:0] pins are read-accessible in a status register (SYSBOOT field CONTROL.CONTROL_STATUS[5:0]) following a power-on reset. The SCM does not use sys_boot[6].

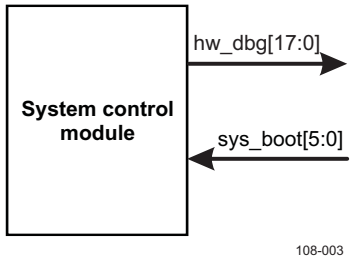
With the correct pad configuration, the SCM maps the hw_dbg[17:0] pins at the device boundary to observe hardware debug signals from device modules. The internal observable signals are PRCM signals, DMA requests, and interrupts.

7.2.1 Functional Interfaces

7.2.1.1 Basic SCM Pins

Figure 7-3 shows the SCM functional interface configured to observe device module debug signals.

Figure 7-3. SCM Interface Signals



7.2.1.2 SCM Interface Description

Table 7-1 lists the SCM input and output configured to observe device module debug signals.

Table 7-1. SCM I/O Description

Signal Name	I/O[1]	Description	Reset Value
hw_dbg[17:0]	O	Debug signals 0 to 17	N/A
sys_boot[5:0]	I	Boot configuration mode bits 0 to 5	Unknown

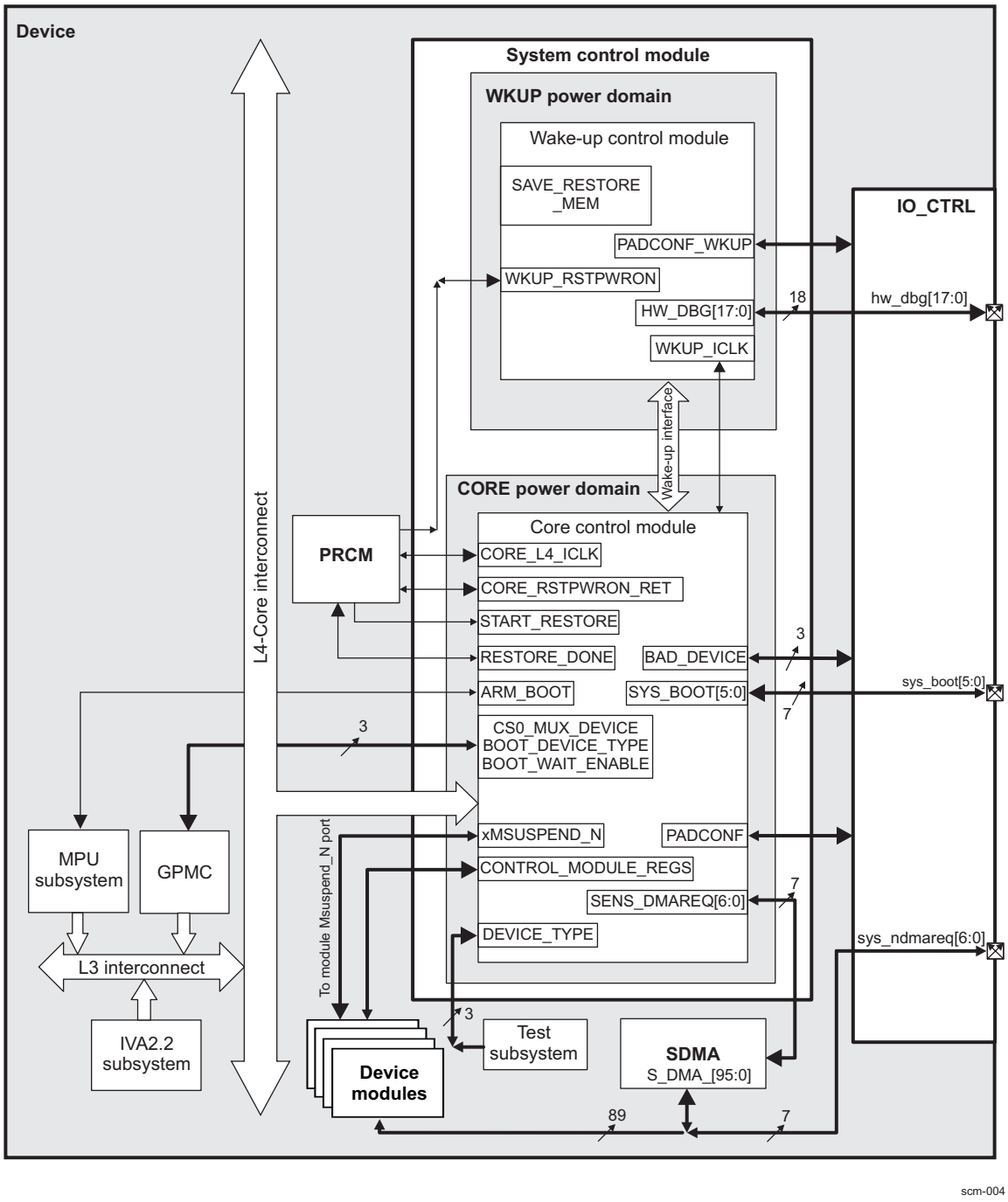
(1) I = Input, O = Output

7.3 SCM Integration

This section describes the integration of the SCM within the device.

Figure 7-4 shows the SCM integration in the device.

Figure 7-4. SCM Integration



scm-004

The SCM is split into two blocks: the core control module in the CORE power domain, and the wake-up control module in the WKUP power domain. The wake-up control module contains the save-and-restore memory, some observability multiplexing and pad configurations. For more information about the wake-up control module, see Section 7.4.3, *Wake-up Control Module*.

The software sets the configuration registers to the desired values depending on the configuration of the requested device. Static device configuration registers can be set by the software at any time and are effective immediately.

The SCM is connected on the L4-Core interconnect. The power, reset, and clock management (PRCM) module provides the module interface clock (CORE_L4_ICLK) and the power-on reset signal. The PRCM generates one global reset per power domain (CORE_RSTPWRON_RET for the CORE power domain and WKUP_RSTPWRON for the WKUP power domain). The SCM does not respond to a warm reset or to an L4 reset.

7.3.1 Clocking, Reset, and Power-Management Scheme

7.3.1.1 Clock

The main sequential logic within the SCM is accessible in a register file through the L4-Core. The only clock provided to the SCM is the interface clock, CORE_L4_ICLK. This clock comes from the PRCM module and is controlled by the EN_OMAPCTRL bit PRCM.CM_ICLKEN1_CORE[6] (0 = disables the clock, 1 = enables the clock) and the AUTO_OMAPCTRL bit PRCM.CM_AUTOIDLE1_CORE[6] (enables/disables automatic control of the interface clock).

The wake-up control module is configured through the L4-Core interface of the core control module and is accessed from the core control module through a dedicated interface. This interface uses the L4-Core interface clock (CORE_L4_ICLK) divided by 4 or 2 according to the WKUPCTRLCLOCKDIV bit CONTROL.CONTROL_PADCONF_OFF[2]. Only this wake-up interface clock (WKUP_ICLK) is propagated to the wake-up control module.

For further information, see Chapter 4, *Power, Reset, and Clock Management.*

7.3.1.2 Resets

The SCM responds only to the internal power-on reset and to the device type. The SOFTRESET bit CONTROL.CONTROL_SYSCONFIG[1] has no effect; the SCM is not affected by a warm reset.

The internal power-on reset is not a direct image of the power-on reset input pin (SYS_NRESPWRON). The PRCM module generates an internal power-on reset signal per power domain and activates the internal power-on reset when the eFuse-related settings (such as the device type) are initialized. The core control module of the CORE power domain responds to the CORE power-on reset (CORE_RSTPWRON_RET). The wake-up control module of the WKUP power domain responds to the power-on reset (WKUP_RSTPWRON).

Note: References in the TRM to the power-on reset refer to the internal power-on reset as seen by the SCM.

On emulator devices, the debugger can also control the internal power-on reset signal though the JTAG interface.

For further information, see Chapter 4, *Power, Reset, and Clock Management*.

7.3.1.3 Power Domain

The SCM is split into two blocks: the core control module, which is attached to the CORE power domain and can be in either active, retention, or off state, and the wake-up control module, which belongs to the WKUP power domain.

Note: The SCM is fully built with retention Flip-Flop.

For further information, see Chapter 4, *Power, Reset, and Clock Management.*

7.3.1.4 Power Management

7.3.1.4.1 System Power Management

The PRCM module can require the SCM to be idled to save power. The SCM enters idle mode through the IDLEMODE field CONTROL.CONTROL_SYSCONFIG[4:3].

The PRCM module requests idle mode, and the SCM always accepts an idle command. Idle mode can be configured to either force-idle or smart-idle mode.

When the SCM is in force-idle mode (IDLEMODE bits CONTROL.CONTROL_SYSCONFIG[4:3] = 0b00) and it receives an idle request from the PRCM module (PRCM.CM_ICLKEN1_CORE[6] set to 0 or PRCM.CM_AUTOIDLE1_CORE[6] set to 1 and the L4-Core interface clock idle transitions), the SCM waits unconditionally for active system clock gating by the PRCM module. Active system clock gating occurs only when all peripherals supplied by the same L4-Core interface clock domain are also ready for idle.

In smart-idle mode (IDLEMODE bits CONTROL.CONTROL_SYSCONFIG[4:3] = 0b10), an idle command is accepted only when the save mechanism completes.

In idle mode (when the PRCM module has gated the interface clock), the SCM is not active and the interface clock paths are gated.

Note: The ST_OMAPCTRL bit PRCM.CM_IDLEST1_CORE[6] (0b0 = active, 0b1 = idle) can check the SCM idle state.

The SCM idle mode is a function of the EN_OMAPCTRL bit PRCM.CM_ICLKEN1_CORE[6] configuration and can be controlled automatically with hardware, depending on the AUTO_OMAPCTRL bit PRCM.CM_AUTOIDLE1_CORE[6] configuration.

7.3.1.4.2 Module Power Saving

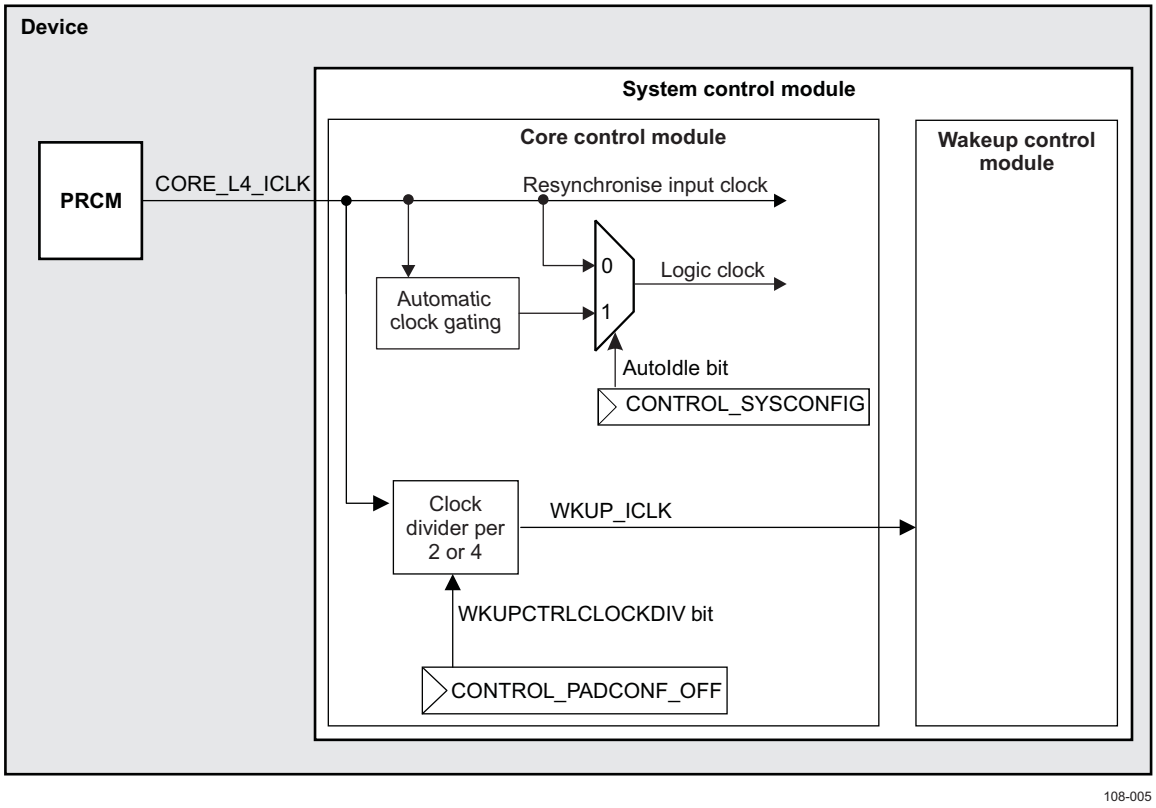
An internal interface clock-gating feature provides SCM local power management. The clock for the logic within the module can be gated when there is no access to the module according to the AUTOIDLE bit CONTROL.CONTROL_SYSCONFIG[0]. Otherwise, this logic is free-running on the interface clock CORE_L4_ICLK.

The L4-Core interface clock (CORE_L4_ICLK) is also used to synchronize and resample module inputs. The clock for those functions must always be free-running. Therefore, it is not gated.

The wake-up control module is clocked only by a local wake-up interface clock from the core control module. The wake-up interface clock (WKUP_ICLK) uses the L4-Core interface clock (CORE_L4_ICLK) divided by 4 or 2 to reduce power consumption according to the WKUPCTRLCLOCKDIV bit CONTROL.CONTROL_PADCONF_OFF[2] (0 = divided by 4, 1 = divided by 2).

Figure 7-5 shows the SCM internal clock implementation.

Figure 7-5. Internal Clock Implementation



7.3.2 Hardware Requests

The SCM does not generate interrupt or wake-up requests.

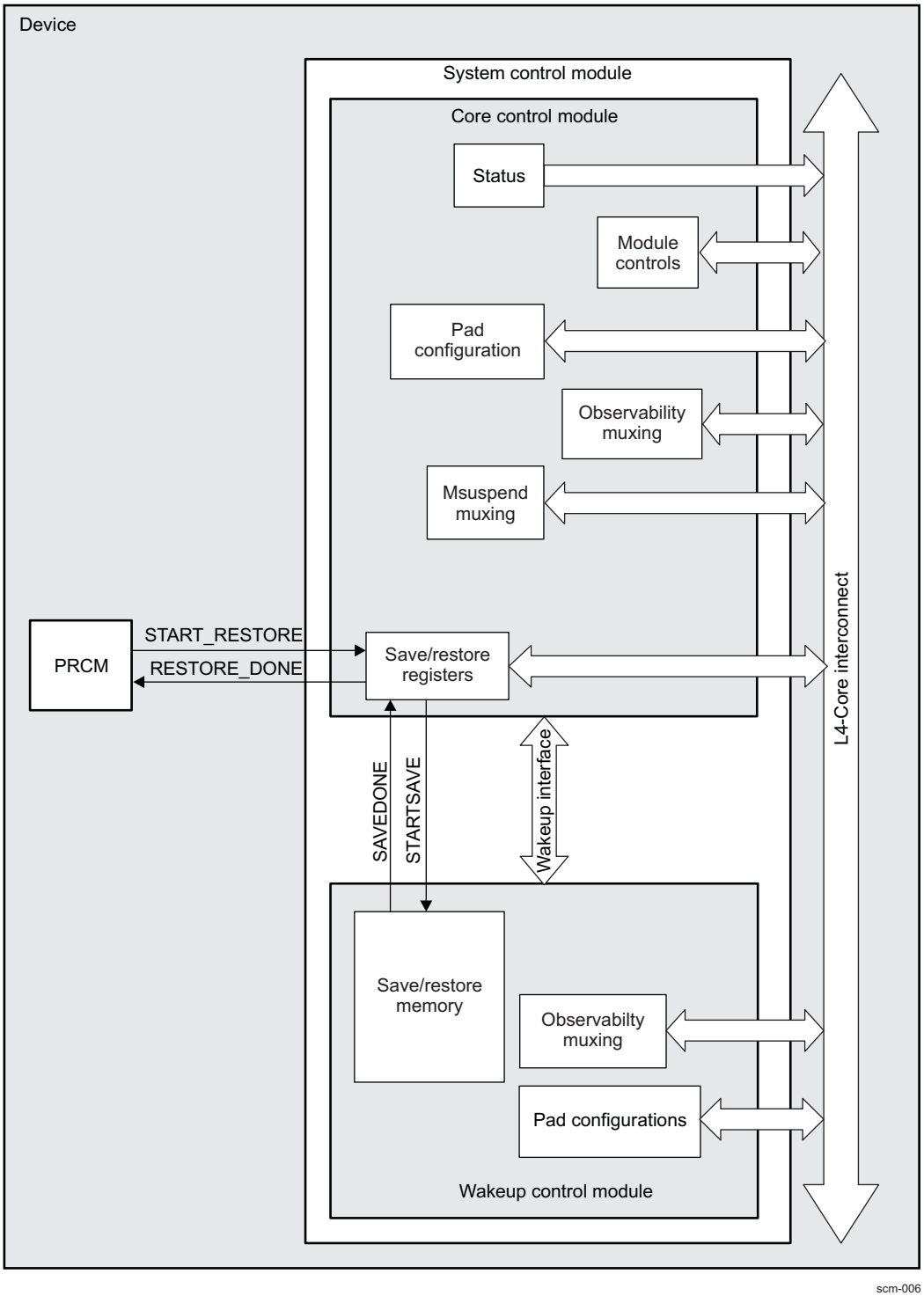
7.4 SCM Functional Description

7.4.1 Block Diagram

The SCM controls various device modules settings through register configuration and internal signals. It also controls the pad configuration and multiplexing and the routing of internal signals (such as PRCM signals or DMA requests) to observable pins for debug.

Figure 7-6 shows the SCM block diagram.

Figure 7-6. SCM Block Diagram



The following sections describe the functionality of the SCM registers.

7.4.2 SCM Initialization

The SCM responds only to the internal power-on reset and to the device type. At power-on, reset values for the registers define the safe state for the device. In the initialization mode, only modules used at boot time are associated with the pads. Other module inputs are internally tied, and outputs pads are turned off each time the feature is available.

For the pad configuration, pull-up/pull-down fields are set according to the device pin list.

General-purpose devices include features that are inaccessible or unavailable. These inaccessible registers define the default or fixed device configuration or behavior.

7.4.3 Wake-up Control Module

The wake-up control module in the SCM belongs to the WKUP power domain. It contains a 1K-byte memory in which to save the pad configuration registers in the core control module before going to off mode. Pad configuration registers driving the I/O pad control in the WKUP power domain are also instantiated in the wake-up control module.

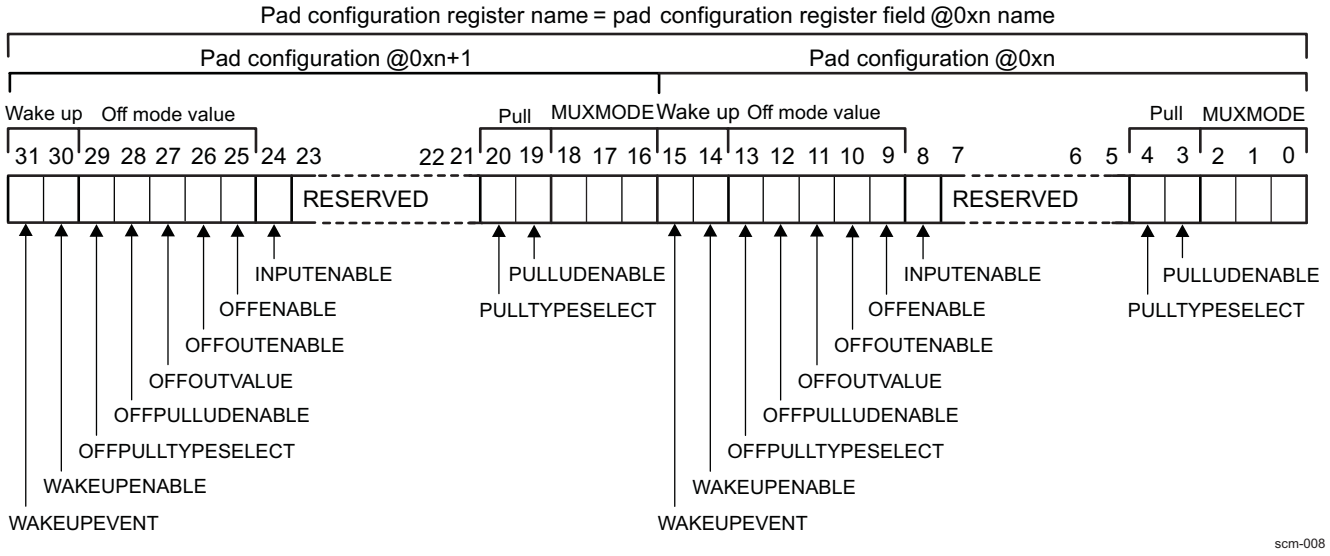
The wake-up control module is configured through the L4-Core interface in the core control module and is accessed from the core control module through a dedicated interface. This interface between the core control module and the wake-up control module uses the CORE_L4_ICLK clock divided by 4 or 2 to reduce power consumption in the WKUP power domain.

7.4.4 Pad Functional Multiplexing and Configuration

After power-on reset, the software sets the pad functional multiplexing and configuration registers to the requested device pad configurations. Data written in these registers command directly the multiplexing of the pad configuration logic.

Each pin is configurable by software using its associated pad configuration register field, which is 16 bits wide (see Figure 7-7).

Figure 7-7. Pad Configuration Register Functionality



One pad configuration register field is available for each pin. Each 32-bit pad configuration register is grouped into two 16-bit pad configuration register fields. One pad configuration register provides control for two different pins.

Some pad configuration registers control the configuration of pads in the CORE power domain. These registers are instantiated in the CORE power domain of the SCM (core control module, physical addresses 0x4800 2030 to 0x4800 2260). Pad configuration registers also control the configuration of pads in the WKUP power domain. These registers are instantiated in the WKUP power domain of the SCM (wake-up control module, physical addresses 0x4800 2A00 to 0x4800 2A4C).

Note: These registers can be accessed using 8-, 16-, and 32-bit operations.

The functional bits of a pad configuration register field are divided into the following five fields:

- MUXMODE (3 bits) defines the multiplexing mode applied to the pin. A mode corresponds to the selection of the functionality mapped on the pin with six (0 to 5) possible functional modes for each pin.
- PULL (2 bits) for combinational pullup/pulldown configuration:
  - PULLTYPESELECT: Pullup/pulldown selection for the pin.
  - PULLUDENABLE: Pullup/pulldown enable for the pin.
- INPUTENABLE (1 bit) drives an input enable signal to the I/O CTRL.
  - INPUTENABLE = 0: Input Disable. Pin is configured in output only mode.
  - INPUTENABLE = 1: Input Enable. Pin is configured in bidirectional mode.
- Off mode values (5 bits) override the pin state when the OFFENABLE bit CONTROL.CONTROL_PADCONF_X is set and off mode is active. This feature allows having separate configurations for the pins when in off mode:
  - OFFENABLE: Off mode pin state override control. Set to 1 to enable the feature and to 0 to disable it.
  - OFFOUTENABLE: Off mode output enable value. Set to 0 to enable the feature and to 1 to disable it.
  - OFFOUTVALUE: Off mode output value.
  - OFFPULLUDENABLE: Off mode pullup/pulldown enable
  - OFFPULLTYPESELECT: Off mode pullup/pulldown selection

CAUTION

The OFFOUTENABLE and OFFOUTVALUE bits are functional only if the pad configuration supports output mode on at least one MUXMODE. For a pad that supports only the input feature, the OFFOUTENABLE and OFFOUTVALUE bits cannot be configured (they are don't care and read always returns 0).

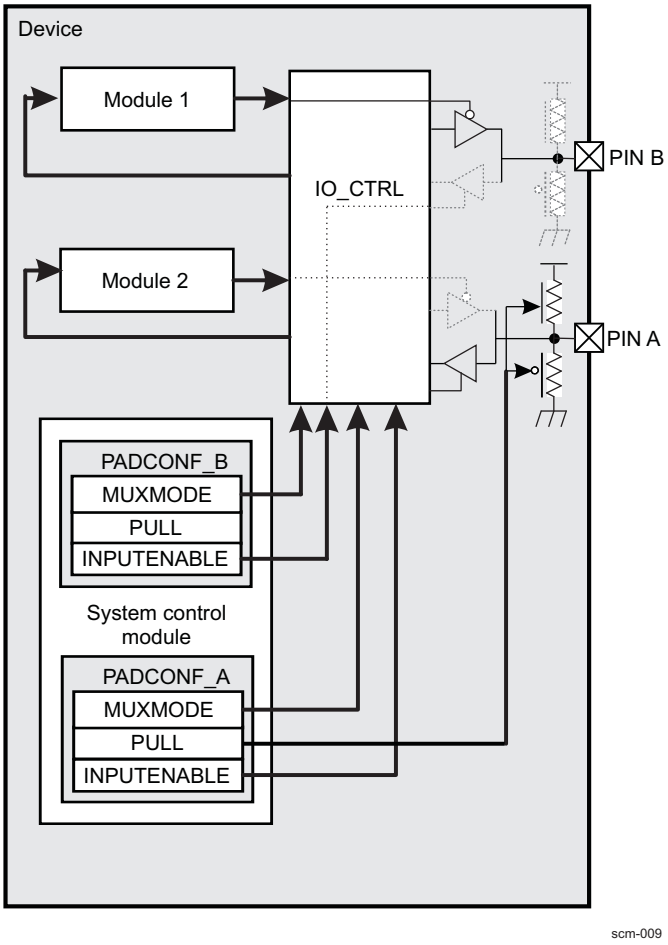
- Wake-up bits (2 bits):
  - WAKEUPENABLE: Enable wake-up detection on input. It is also the off mode input enable value.
  - WAKEUPEVENT: Wake-up event status for the pin.

CAUTION

The software must configure the OFF mode pads. It must ensure that the input/output capability is enabled for each pin.

Figure 7-8 shows the pad configuration functionality when off mode is inactive. For more information about off mode, see Section 7.4.4.4, *Off Mode*.

Figure 7-8. Pad Configuration Diagram



7.4.4.1 Mode Selection

Table 7-2 lists the multiplexing modes and settings.

Table 7-2. Mode Selection

MUXMODE	Selected Mode
0b000	Primary mode = Mode 0
0b001	Mode 1
0b010	Mode 2
0b011	Mode 3
0b100	Mode 4
0b101	Mode 5
0b110	Mode 6
0b111	Safe mode = Mode 7

The MUXMODE field CONTROL.CONTROL_PADCONF_X defines the multiplexing mode applied to the pad. Modes are referred to by their decimal (from 0 to 7) or binary (from 0b000 to 0b111) representation. Functional modes are defined from 0b000 to 0b101; mode 0b111 is referred to as the safe mode.

For most pads, the reset value for the MUXMODE field CONTROL.CONTROL_PADCONF_X is 0b111. The exceptions are pads to be used at boot time to transfer data from selected peripherals to the external flash memory.

Mode 0 is the primary mode. When mode 0 is set, the function mapped to the pin corresponds to the name of the pin.

Mode 1 to mode 6 are possible modes for alternate functions. On each pin, some modes are used effectively for alternate functions, while other modes are unused and correspond to no functional configuration.

The safe mode avoids any risk of electrical contention by configuring the pin as an input with no functional interface mapped to it. The safe mode is used mainly as the default mode for all pins containing no mandatory interface at the release of power-on reset.

For more information about the configurable mode on each pin, see Table 7-4 through Table 7-6.

7.4.4.2 Pull Selection

Whichever pull value is configured, pulls are automatically disabled when a pin is configured as an output (see Table 7-3).

Table 7-3. Pull Selection

PULL		Pin Behavior
PULLTYPESELECT	PULLUDENABLE	
0b0	0b0	Pull-down selected but not activated
0b0	0b1	Pull-down selected and activated if pin is NOT configured as OUTPUT
0b1	0b0	Pull-up selected but not activated
0b1	0b1	Pull-up selected and activated if pin is NOT configured as OUTPUT

For more information on the pull available on each pin, see Table 7-4 through Table 7-6.

7.4.4.3 Pad Multiplexing Register Fields

Table 7-4 through Table 7-6 provide for each pad configuration register field the address offset , reset values, and associated signal name for each multiplexing mode (as set by bit field MUXMODE). Mode 0 is always defined. Modes with no signal name are undefined for the given pad.

Notes:

- Pad configuration registers are split into three types, which correspond to the following three tables:
  - Table 7-4 lists the pad configuration registers instantiated in the CORE power domain that drive the pads in the CORE power domain.
  - Table 7-5 lists the pad configuration registers instantiated in the CORE power domain that drive the D2D pads. These pads are available in stacked mode only.
  - Table 7-6 lists the pad configuration registers instantiated in the WKUP power domain that drive the pads in the WKUP power domain.
- In Table 7-4 through Table 7-6, an empty cell indicates that the mode or pull is not available for this pin.

Table 7-4. Core Control Module Pad Configuration Register Fields

Register Name	Physical Address	Mode 0	Mode 1	Mode 2	Mode 3	Mode 4	Mode 5	Mode 6	Mode 7
CONTROL_PADCONF_SDRC_D0[15:0]	0x4800 2030	sdrc_d0							
CONTROL_PADCONF_SDRC_D0[31:16]	0x4800 2030	sdrc_d1							
CONTROL_PADCONF_SDRC_D2[15:0]	0x4800 2034	sdrc_d2							
CONTROL_PADCONF_SDRC_D2[31:16]	0x4800 2034	sdrc_d3							
CONTROL_PADCONF_SDRC_D4[15:0]	0x4800 2038	sdrc_d4							
CONTROL_PADCONF_SDRC_D4[31:16]	0x4800 2038	sdrc_d5							
CONTROL_PADCONF_SDRC_D6[15:0]	0x4800 203C	sdrc_d6							
CONTROL_PADCONF_SDRC_D6[31:16]	0x4800 203C	sdrc_d7							
CONTROL_PADCONF_SDRC_D8[15:0]	0x4800 2040	sdrc_d8							
CONTROL_PADCONF_SDRC_D8[31:16]	0x4800 2040	sdrc_d9							
CONTROL_PADCONF_SDRC_D10[15:0]	0x4800 2044	sdrc_d10							
CONTROL_PADCONF_SDRC_D10[31:16]	0x4800 2044	sdrc_d11							
CONTROL_PADCONF_SDRC_D12[15:0]	0x4800 2048	sdrc_d12							
CONTROL_PADCONF_SDRC_D12[31:16]	0x4800 2048	sdrc_d13							
CONTROL_PADCONF_SDRC_D14[15:0]	0x4800 204C	sdrc_d14							
CONTROL_PADCONF_SDRC_D14[31:16]	0x4800 204C	sdrc_d15							
CONTROL_PADCONF_SDRC_D16[15:0]	0x4800 2050	sdrc_d16							
CONTROL_PADCONF_SDRC_D16[31:16]	0x4800 2050	sdrc_d17							
CONTROL_PADCONF_SDRC_D18[15:0]	0x4800 2054	sdrc_d18							
CONTROL_PADCONF_SDRC_D18[31:16]	0x4800 2054	sdrc_d19							
CONTROL_PADCONF_SDRC_D20[15:0]	0x4800 2058	sdrc_d20							

Table 7-4. Core Control Module Pad Configuration Register Fields (continued)

Register Name	Physical Address	Mode 0	Mode 1	Mode 2	Mode 3	Mode 4	Mode 5	Mode 6	Mode 7
CONTROL_PADCONF_SDRC_D20[31:16]	0x4800 2058	sdrc_d21							
CONTROL_PADCONF_SDRC_D22[15:0]	0x4800 205C	sdrc_d22							
CONTROL_PADCONF_SDRC_D22[31:16]	0x4800 205C	sdrc_d23							
CONTROL_PADCONF_SDRC_D24[15:0]	0x4800 2060	sdrc_d24							
CONTROL_PADCONF_SDRC_D24[31:16]	0x4800 2060	sdrc_d25							
CONTROL_PADCONF_SDRC_D26[15:0]	0x4800 2064	sdrc_d26							
CONTROL_PADCONF_SDRC_D26[31:16]	0x4800 2064	sdrc_d27							
CONTROL_PADCONF_SDRC_D28[15:0]	0x4800 2068	sdrc_d28							
CONTROL_PADCONF_SDRC_D28[31:16]	0x4800 2068	sdrc_d29							
CONTROL_PADCONF_SDRC_D30[15:0]	0x4800 206C	sdrc_d30							
CONTROL_PADCONF_SDRC_D30[31:16]	0x4800 206C	sdrc_d31							
CONTROL_PADCONF_SDRC_CLK[15:0]	0x4800 2070	sdrc_clk							
CONTROL_PADCONF_SDRC_CLK[31:16]	0x4800 2070	sdrc_dqs0							
CONTROL_PADCONF_SAD2D_SBUSFLAG[31:16]	0x4800 2260	sdrc_cke0							safe_mode
CONTROL_PADCONF_SDRC_CKE1[15:0]	0x4800 2264	sdrc_cke1							safe_mode
CONTROL_PADCONF_SDRC_DQS1[15:0]	0x4800 2074	sdrc_dqs1							
CONTROL_PADCONF_SDRC_DQS1[31:16]	0x4800 2074	sdrc_dqs2							
CONTROL_PADCONF_SDRC_DQS3[15:0]	0x4800 2078	sdrc_dqs3							
CONTROL_PADCONF_SDRC_DQS3[31:16]	0x4800 2078	gpmc_a1				gpio_34			safe_mode
CONTROL_PADCONF_GPMC_A2[15:0]	0x4800 207C	gpmc_a2				gpio_35			safe_mode
CONTROL_PADCONF_GPMC_A2[31:16]	0x4800 207C	gpmc_a3				gpio_36			safe_mode
CONTROL_PADCONF_GPMC_A4[15:0]	0x4800 2080	gpmc_a4				gpio_37			safe_mode
CONTROL_PADCONF_GPMC_A4[31:16]	0x4800 2080	gpmc_a5				gpio_38			safe_mode
CONTROL_PADCONF_GPMC_A6[15:0]	0x4800 2084	gpmc_a6				gpio_39			safe_mode
CONTROL_PADCONF_GPMC_A6[31:16]	0x4800 2084	gpmc_a7				gpio_40			safe_mode
CONTROL_PADCONF_GPMC_A8[15:0]	0x4800 2088	gpmc_a8				gpio_41			safe_mode
CONTROL_PADCONF_GPMC_A8[31:16]	0x4800 2088	gpmc_a9	sys_ndmareq2			gpio_42			safe_mode
CONTROL_PADCONF_GPMC_A10[15:0]	0x4800 208C	gpmc_a10	sys_ndmareq3			gpio_43			safe_mode
CONTROL_PADCONF_GPMC_A10[31:16]	0x4800 208C	gpmc_d0							
CONTROL_PADCONF_GPMC_D1[15:0]	0x4800 2090	gpmc_d1							
CONTROL_PADCONF_GPMC_D1[31:16]	0x4800 2090	gpmc_d2							
CONTROL_PADCONF_GPMC_D3[15:0]	0x4800 2094	gpmc_d3							
CONTROL_PADCONF_GPMC_D3[31:16]	0x4800 2094	gpmc_d4							
CONTROL_PADCONF_GPMC_D5[15:0]	0x4800 2098	gpmc_d5							
CONTROL_PADCONF_GPMC_D5[31:16]	0x4800 2098	gpmc_d6							
CONTROL_PADCONF_GPMC_D7[15:0]	0x4800 209C	gpmc_d7							

Table 7-4. Core Control Module Pad Configuration Register Fields (continued)

Register Name	Physical Address	Mode 0	Mode 1	Mode 2	Mode 3	Mode 4	Mode 5	Mode 6	Mode 7
CONTROL_PADCONF_GPMC_D7[31:16]	0x4800 209C	gpmc_d8				gpio_44			safe_mode
CONTROL_PADCONF_GPMC_D9[15:0]	0x4800 20A0	gpmc_d9				gpio_45			safe_mode
CONTROL_PADCONF_GPMC_D9[31:16]	0x4800 20A0	gpmc_d10				gpio_46			safe_mode
CONTROL_PADCONF_GPMC_D11[15:0]	0x4800 20A4	gpmc_d11				gpio_47			safe_mode
CONTROL_PADCONF_GPMC_D11[31:16]	0x4800 20A4	gpmc_d12				gpio_48			safe_mode
CONTROL_PADCONF_GPMC_D13[15:0]	0x4800 20A8	gpmc_d13				gpio_49			safe_mode
CONTROL_PADCONF_GPMC_D13[31:16]	0x4800 20A8	gpmc_d14				gpio_50			safe_mode
CONTROL_PADCONF_GPMC_D15[15:0]	0x4800 20AC	gpmc_d15				gpio_51			safe_mode
CONTROL_PADCONF_GPMC_D15[31:16]	0x4800 20AC	gpmc_ncs0							
CONTROL_PADCONF_GPMC_NCS1[15:0]	0x4800 20B0	gpmc_ncs1				gpio_52			safe_mode
CONTROL_PADCONF_GPMC_NCS1[31:16]	0x4800 20B0	gpmc_ncs2				gpio_53			safe_mode
CONTROL_PADCONF_GPMC_NCS3[15:0]	0x4800 20B4	gpmc_ncs3	sys_ndmareq0			gpio_54			safe_mode
CONTROL_PADCONF_GPMC_NCS3[31:16]	0x4800 20B4	gpmc_ncs4	sys_ndmareq1	mcbsp4_clkx	gpt9_pwm_evt	gpio_55			safe_mode
CONTROL_PADCONF_GPMC_NCS5[15:0]	0x4800 20B8	gpmc_ncs5	sys_ndmareq2	mcbsp4_dr	gpt10_pwm_evt	gpio_56			safe_mode
CONTROL_PADCONF_GPMC_NCS5[31:16]	0x4800 20B8	gpmc_ncs6	sys_ndmareq3	mcbsp4_dx	gpt11_pwm_evt	gpio_57			safe_mode
CONTROL_PADCONF_GPMC_NCS7[15:0]	0x4800 20BC	gpmc_ncs7	gpmc_io_dir	mcbsp4_fsx	gpt8_pwm_evt	gpio_58			safe_mode
CONTROL_PADCONF_GPMC_NCS7[31:16]	0x4800 20BC	gpmc_clk				gpio_59			safe_mode
CONTROL_PADCONF_GPMC_NADV_ALE[15:0]	0x4800 20C0	gpmc_nadv_ale							
CONTROL_PADCONF_GPMC_NADV_ALE[31:16]	0x4800 20C0	gpmc_noe							
CONTROL_PADCONF_GPMC_NWE[15:0]	0x4800 20C4	gpmc_nwe							
CONTROL_PADCONF_GPMC_NWE[31:16]	0x4800 20C4	gpmc_nbe0_cle				gpio_60			safe_mode
CONTROL_PADCONF_GPMC_NBE1[15:0]	0x4800 20C8	gpmc_nbe1				gpio_61			safe_mode
CONTROL_PADCONF_GPMC_NBE1[31:16]	0x4800 20C8	gpmc_nwp				gpio_62			safe_mode
CONTROL_PADCONF_GPMC_WAIT0[15:0]	0x4800 20CC	gpmc_wait0							
CONTROL_PADCONF_GPMC_WAIT0[31:16]	0x4800 20CC	gpmc_wait1				gpio_63			safe_mode
CONTROL_PADCONF_GPMC_WAIT2[15:0]	0x4800 20D0	gpmc_wait2				gpio_64			safe_mode
CONTROL_PADCONF_GPMC_WAIT2[31:16]	0x4800 20D0	gpmc_wait3	sys_ndmareq1			gpio_65			safe_mode
CONTROL_PADCONF_DSS_PCLK[15:0]	0x4800 20D4	dss_pclk				gpio_66	hw_dbg12		safe_mode
CONTROL_PADCONF_DSS_PCLK[31:16]	0x4800 20D4	dss_hsync				gpio_67	hw_dbg13		safe_mode
CONTROL_PADCONF_DSS_VSYNC[15:0]	0x4800 20D8	dss_vsync				gpio_68			safe_mode
CONTROL_PADCONF_DSS_VSYNC[31:16]	0x4800 20D8	dss_acbias				gpio_69			safe_mode
CONTROL_PADCONF_DSS_DATA0[15:0]	0x4800 20DC	dss_data0	dsi_dx0	uart1_cts	dssvenc656_data0	gpio_70			safe_mode
CONTROL_PADCONF_DSS_DATA0[31:16]	0x4800 20DC	dss_data1	dsi_dy0	uart1_rts	dssvenc656_data1	gpio_71			safe_mode
CONTROL_PADCONF_DSS_DATA2[15:0]	0x4800 20E0	dss_data2	dsi_dx1		dssvenc656_data2	gpio_72			safe_mode

Table 7-4. Core Control Module Pad Configuration Register Fields (continued)

Register Name	Physical Address	Mode 0	Mode 1	Mode 2	Mode 3	Mode 4	Mode 5	Mode 6	Mode 7
CONTROL_PADCONF_DSS_DATA2[31:16]	0x4800 20E0	dss_data3	dsi_dy1		dssvenc656_data3	gpio_73			safe_mode
CONTROL_PADCONF_DSS_DATA4[15:0]	0x4800 20E4	dss_data4	dsi_dx2	uart3_rx_irrx	dssvenc656_data4	gpio_74			safe_mode
CONTROL_PADCONF_DSS_DATA4[31:16]	0x4800 20E4	dss_data5	dsi_dy2	uart3_rx_irtx	dssvenc656_data5	gpio_75			safe_mode
CONTROL_PADCONF_DSS_DATA6[15:0]	0x4800 20E8	dss_data6		uart1_tx	dssvenc656_data6	gpio_76	hw_dbg14		safe_mode
CONTROL_PADCONF_DSS_DATA6[31:16]	0x4800 20E8	dss_data7		uart1_rx	dssvenc656_data7	gpio_77	hw_dbg15		safe_mode
CONTROL_PADCONF_DSS_DATA8[15:0]	0x4800 20EC	dss_data8				gpio_78	hw_dbg16		safe_mode
CONTROL_PADCONF_DSS_DATA8[31:16]	0x4800 20EC	dss_data9				gpio_79	hw_dbg17		safe_mode
CONTROL_PADCONF_DSS_DATA10[15:0]	0x4800 20F0	dss_data10	sdi_dat1n			gpio_80			safe_mode
CONTROL_PADCONF_DSS_DATA10[31:16]	0x4800 20F0	dss_data11	sdi_dat1p			gpio_81			safe_mode
CONTROL_PADCONF_DSS_DATA12[15:0]	0x4800 20F4	dss_data12	sdi_dat2n			gpio_82			safe_mode
CONTROL_PADCONF_DSS_DATA12[31:16]	0x4800 20F4	dss_data13	sdi_dat2p			gpio_83			safe_mode
CONTROL_PADCONF_DSS_DATA14[15:0]	0x4800 20F8	dss_data14	sdi_dat3n			gpio_84			safe_mode
CONTROL_PADCONF_DSS_DATA14[31:16]	0x4800 20F8	dss_data15	sdi_dat3p			gpio_85			safe_mode
CONTROL_PADCONF_DSS_DATA16[15:0]	0x4800 20FC	dss_data16				gpio_86			safe_mode
CONTROL_PADCONF_DSS_DATA16[31:16]	0x4800 20FC	dss_data17				gpio_87			safe_mode
CONTROL_PADCONF_DSS_DATA18[15:0]	0x4800 2100	dss_data18	sdi_vsync	mcspi3_clk	dss_data0	gpio_88			safe_mode
CONTROL_PADCONF_DSS_DATA18[31:16]	0x4800 2100	dss_data19	sdi_hsync	mcspi3_simo	dss_data1	gpio_89			safe_mode
CONTROL_PADCONF_DSS_DATA20[15:0]	0x4800 2104	dss_data20	sdi_den	mcspi3_somi	dss_data2	gpio_90			safe_mode
CONTROL_PADCONF_DSS_DATA20[31:16]	0x4800 2104	dss_data21	sdi_stp	mcspi3_cs0	dss_data3	gpio_91			safe_mode
CONTROL_PADCONF_DSS_DATA22[15:0]	0x4800 2108	dss_data22	sdi_clkp	mcspi3_cs1	dss_data4	gpio_92			safe_mode
CONTROL_PADCONF_DSS_DATA22[31:16]	0x4800 2108	dss_data23	sdi_clkn		dss_data5	gpio_93			safe_mode
CONTROL_PADCONF_CAM_HS[15:0]	0x4800 210C	cam_hs				gpio_94	hw_dbg0		safe_mode
CONTROL_PADCONF_CAM_HS[31:16]	0x4800 210C	cam_vs				gpio_95	hw_dbg1		safe_mode
CONTROL_PADCONF_CAM_XCLKA[15:0]	0x4800 2110	cam_xclka				gpio_96			safe_mode
CONTROL_PADCONF_CAM_XCLKA[31:16]	0x4800 2110	cam_pclk				gpio_97	hw_dbg2		safe_mode
CONTROL_PADCONF_CAM_FLD[15:0]	0x4800 2114	cam_fld		cam_global_reset		gpio_98	hw_dbg3		safe_mode
CONTROL_PADCONF_CAM_FLD[31:16]	0x4800 2114	cam_d0		csi2_dx2		gpio_99			safe_mode
CONTROL_PADCONF_CAM_D1[15:0]	0x4800 2118	cam_d1		csi2_dy2		gpio_100			safe_mode
CONTROL_PADCONF_CAM_D1[31:16]	0x4800 2118	cam_d2				gpio_101	hw_dbg4		safe_mode
CONTROL_PADCONF_CAM_D3[15:0]	0x4800 211C	cam_d3				gpio_102	hw_dbg5		safe_mode
CONTROL_PADCONF_CAM_D3[31:16]	0x4800 211C	cam_d4				gpio_103	hw_dbg6		safe_mode
CONTROL_PADCONF_CAM_D5[15:0]	0x4800 2120	cam_d5				gpio_104	hw_dbg7		safe_mode

Table 7-4. Core Control Module Pad Configuration Register Fields (continued)

Register Name	Physical Address	Mode 0	Mode 1	Mode 2	Mode 3	Mode 4	Mode 5	Mode 6	Mode 7
CONTROL_PADCONF_CAM_D5[31:16]	0x4800 2120	cam_d6				gpio_105			safe_mode
CONTROL_PADCONF_CAM_D7[15:0]	0x4800 2124	cam_d7				gpio_106			safe_mode
CONTROL_PADCONF_CAM_D7[31:16]	0x4800 2124	cam_d8				gpio_107			safe_mode
CONTROL_PADCONF_CAM_D9[15:0]	0x4800 2128	cam_d9				gpio_108			safe_mode
CONTROL_PADCONF_CAM_D9[31:16]	0x4800 2128	cam_d10				gpio_109	hw_dbg8		safe_mode
CONTROL_PADCONF_CAM_D11[15:0]	0x4800 212C	cam_d11				gpio_110	hw_dbg9		safe_mode
CONTROL_PADCONF_CAM_D11[31:16]	0x4800 212C	cam_xclkb				gpio_111			safe_mode
CONTROL_PADCONF_CAM_WEN[15:0]	0x4800 2130	cam_wen		cam_shutter		gpio_167	hw_dbg10		safe_mode
CONTROL_PADCONF_CAM_WEN[31:16]	0x4800 2130	cam_strobe				gpio_126	hw_dbg11		safe_mode
CONTROL_PADCONF_CSI2_DX0[15:0]	0x4800 2134	csi2_dx0				gpio_112			safe_mode
CONTROL_PADCONF_CSI2_DX0[31:16]	0x4800 2134	csi2_dy0				gpio_113			safe_mode
CONTROL_PADCONF_CSI2_DX1[15:0]	0x4800 2138	csi2_dx1				gpio_114			safe_mode
CONTROL_PADCONF_CSI2_DX1[31:16]	0x4800 2138	csi2_dy1				gpio_115			safe_mode
CONTROL_PADCONF_MCBSP2_FSX[15:0]	0x4800 213C	mcbsp2_fsx				gpio_116			safe_mode
CONTROL_PADCONF_MCBSP2_FSX[31:16]	0x4800 213C	mcbsp2_clkx				gpio_117			safe_mode
CONTROL_PADCONF_MCBSP2_DR[15:0]	0x4800 2140	mcbsp2_dr				gpio_118			safe_mode
CONTROL_PADCONF_MCBSP2_DR[31:16]	0x4800 2140	mcbsp2_dx				gpio_119			safe_mode
CONTROL_PADCONF_MMC1_CLK[15:0]	0x4800 2144	mmc1_clk	ms_clk			gpio_120			safe_mode
CONTROL_PADCONF_MMC1_CLK[31:16]	0x4800 2144	mmc1_cmd	ms_bs			gpio_121			safe_mode
CONTROL_PADCONF_MMC1_DAT0[15:0]	0x4800 2148	mmc1_dat0	ms_dat0			gpio_122			safe_mode
CONTROL_PADCONF_MMC1_DAT0[31:16]	0x4800 2148	mmc1_dat1	ms_dat1			gpio_123			safe_mode
CONTROL_PADCONF_MMC1_DAT2[15:0]	0x4800 214C	mmc1_dat2	ms_dat2			gpio_124			safe_mode
CONTROL_PADCONF_MMC1_DAT2[31:16]	0x4800 214C	mmc1_dat3	ms_dat3			gpio_125			safe_mode
CONTROL_PADCONF_MMC1_DAT4[15:0]	0x4800 2150	mmc1_dat4		sim_io		gpio_126			safe_mode
CONTROL_PADCONF_MMC1_DAT4[31:16]	0x4800 2150	mmc1_dat5		sim_clk		gpio_127			safe_mode
CONTROL_PADCONF_MMC1_DAT6[15:0]	0x4800 2154	mmc1_dat6		sim_pwrctrl		gpio_128			safe_mode
CONTROL_PADCONF_MMC1_DAT6[31:16]	0x4800 2154	mmc1_dat7		sim_rst		gpio_129			safe_mode
CONTROL_PADCONF_MMC2_CLK[15:0]	0x4800 2158	mmc2_clk	mcspi3_clk			gpio_130			safe_mode
CONTROL_PADCONF_MMC2_CLK[31:16]	0x4800 2158	mmc2_cmd	mcspi3_simo			gpio_131			safe_mode
CONTROL_PADCONF_MMC2_DAT0[15:0]	0x4800 215C	mmc2_dat0	mcspi3_somi			gpio_132			safe_mode
CONTROL_PADCONF_MMC2_DAT0[31:16]	0x4800 215C	mmc2_dat1				gpio_133			safe_mode
CONTROL_PADCONF_MMC2_DAT2[15:0]	0x4800 2160	mmc2_dat2	mcspi3_cs1			gpio_134			safe_mode
CONTROL_PADCONF_MMC2_DAT2[31:16]	0x4800 2160	mmc2_dat3	mcspi3_cs0			gpio_135			safe_mode
CONTROL_PADCONF_MMC2_DAT4[15:0]	0x4800 2164	mmc2_dat4	mmc2_dir_dat0		mmc3_dat0	gpio_136			safe_mode
CONTROL_PADCONF_MMC2_DAT4[31:16]	0x4800 2164	mmc2_dat5	mmc2_dir_dat1	cam_global_reset	mmc3_dat1	gpio_137	hsusb3_tll_stp	mm3_rxdp	safe_mode

Table 7-4. Core Control Module Pad Configuration Register Fields (continued)

Register Name	Physical Address	Mode 0	Mode 1	Mode 2	Mode 3	Mode 4	Mode 5	Mode 6	Mode 7
CONTROL_PADCONF_MMC2_DAT6[15:0]	0x4800 2168	mmc2_dat6	mmc2_dir_cmd	cam_shutter	mmc3_dat2	gpio_138	hsusb3_tll_dir		safe_mode
CONTROL_PADCONF_MMC2_DAT6[31:16]	0x4800 2168	mmc2_dat7	mmc2_clkin		mmc3_dat3	gpio_139	hsusb3_tll_nxt	mm3_rxdm	safe_mode
CONTROL_PADCONF_MCBSP3_DX[15:0]	0x4800 216C	mcbsp3_dx	uart2_cts			gpio_140	hsusb3_tll_data4		safe_mode
CONTROL_PADCONF_MCBSP3_DX[31:16]	0x4800 216C	mcbsp3_dr	uart2_rts			gpio_141	hsusb3_tll_data5		safe_mode
CONTROL_PADCONF_MCBSP3_CLKX[15:0]	0x4800 2170	mcbsp3_clkx	uart2_tx			gpio_142	hsusb3_tll_data6		safe_mode
CONTROL_PADCONF_MCBSP3_CLKX[31:16]	0x4800 2170	mcbsp3_fsx	uart2_rx			gpio_143	hsusb3_tll_data7		safe_mode
CONTROL_PADCONF_UART2_CTS[15:0]	0x4800 2174	uart2_cts	mcbsp3_dx	gpt9_pwm_evt		gpio_144			safe_mode
CONTROL_PADCONF_UART2_CTS[31:16]	0x4800 2174	uart2_rts	mcbsp3_dr	gpt10_pwm_evt		gpio_145			safe_mode
CONTROL_PADCONF_UART2_TX[15:0]	0x4800 2178	uart2_tx	mcbsp3_clkx	gpt11_pwm_evt		gpio_146			safe_mode
CONTROL_PADCONF_UART2_TX[31:16]	0x4800 2178	uart2_rx	mcbsp3_fsx	gpt8_pwm_evt		gpio_147			safe_mode
CONTROL_PADCONF_UART1_TX[15:0]	0x4800 217C	uart1_tx				gpio_148			safe_mode
CONTROL_PADCONF_UART1_TX[31:16]	0x4800 217C	uart1_rts				gpio_149			safe_mode
CONTROL_PADCONF_UART1_CTS[15:0]	0x4800 2180	uart1_cts				gpio_150	hsusb3_tll_clk		safe_mode
CONTROL_PADCONF_UART1_CTS[31:16]	0x4800 2180	uart1_rx		mcbsp1_clkr	mcspi4_clk	gpio_151			safe_mode
CONTROL_PADCONF_MCBSP4_CLKX[15:0]	0x4800 2184	mcbsp4_clkx				gpio_152	hsusb3_tll_data1	mm3_txse0	safe_mode
CONTROL_PADCONF_MCBSP4_CLKX[31:16]	0x4800 2184	mcbsp4_dr				gpio_153	hsusb3_tll_data0	mm3_rxrcv	safe_mode
CONTROL_PADCONF_MCBSP4_DX[15:0]	0x4800 2188	mcbsp4_dx				gpio_154	hsusb3_tll_data2	mm3_txdat	safe_mode
CONTROL_PADCONF_MCBSP4_DX[31:16]	0x4800 2188	mcbsp4_fsx				gpio_155	hsusb3_tll_data3	mm3_txen_n	safe_mode
CONTROL_PADCONF_MCBSP1_CLKR[15:0]	0x4800 218C	mcbsp1_clkr	mcspi4_clk	sim_cd		gpio_156			safe_mode
CONTROL_PADCONF_MCBSP1_CLKR[31:16]	0x4800 218C	mcbsp1_fsr	adpllv2d_dithering_en1	cam_global_reset		gpio_157			safe_mode
CONTROL_PADCONF_MCBSP1_DX[15:0]	0x4800 2190	mcbsp1_dx	mcspi4_simo	mcbsp3_dx		gpio_158			safe_mode
CONTROL_PADCONF_MCBSP1_DX[31:16]	0x4800 2190	mcbsp1_dr	mcspi4_somi	mcbsp3_dr		gpio_159			safe_mode
CONTROL_PADCONF_MCBSP_CLKS[15:0]	0x4800 2194	mcbsp_clks		cam_shutter		gpio_160	uart1_cts		safe_mode
CONTROL_PADCONF_MCBSP_CLKS[31:16]	0x4800 2194	mcbsp1_fsx	mcspi4_cs0	mcbsp3_fsx		gpio_161			safe_mode
CONTROL_PADCONF_MCBSP1_CLKX[15:0]	0x4800 2198	mcbsp1_clkx		mcbsp3_clkx		gpio_162			safe_mode
CONTROL_PADCONF_MCBSP1_CLKX[31:16]	0x4800 2198	uart3_cts_rctx				gpio_163			safe_mode
CONTROL_PADCONF_UART3_RTS_SD[15:0]	0x4800 219C	uart3_rts_sd				gpio_164			safe_mode
CONTROL_PADCONF_UART3_RTS_SD[31:16]	0x4800 219C	uart3_rx_irrx				gpio_165			safe_mode
CONTROL_PADCONF_UART3_TX_IRTX[15:0]	0x4800 21A0	uart3_tx_irtx				gpio_166			safe_mode
CONTROL_PADCONF_UART3_TX_IRTX[31:16]	0x4800 21A0	hsusb0_clk				gpio_120			safe_mode
CONTROL_PADCONF_HSUSB0_STP[15:0]	0x4800 21A4	hsusb0_stp				gpio_121			safe_mode
CONTROL_PADCONF_HSUSB0_STP[31:16]	0x4800 21A4	hsusb0_dir				gpio_122			safe_mode
CONTROL_PADCONF_HSUSB0_NXT[15:0]	0x4800 21A8	hsusb0_nxt				gpio_124			safe_mode
CONTROL_PADCONF_HSUSB0_NXT[31:16]	0x4800 21A8	hsusb0_data0		uart3_tx_irtx		gpio_125			safe_mode
CONTROL_PADCONF_HSUSB0_DATA1[15:0]	0x4800 21AC	hsusb0_data1		uart3_rx_irrx		gpio_130			safe_mode

Table 7-4. Core Control Module Pad Configuration Register Fields (continued)

Register Name	Physical Address	Mode 0	Mode 1	Mode 2	Mode 3	Mode 4	Mode 5	Mode 6	Mode 7
CONTROL_PADCONF_HSUSB0_DATA1[31:16]	0x4800 21AC	hsusb0_data2		uart3_rts_sd		gpio_131			safe_mode
CONTROL_PADCONF_HSUSB0_DATA3[15:0]	0x4800 21B0	hsusb0_data3		uart3_cts_rctx		gpio_169			safe_mode
CONTROL_PADCONF_HSUSB0_DATA3[31:16]	0x4800 21B0	hsusb0_data4				gpio_188			safe_mode
CONTROL_PADCONF_HSUSB0_DATA5[15:0]	0x4800 21B4	hsusb0_data5				gpio_189			safe_mode
CONTROL_PADCONF_HSUSB0_DATA5[31:16]	0x4800 21B4	hsusb0_data6				gpio_190			safe_mode
CONTROL_PADCONF_HSUSB0_DATA7[15:0]	0x4800 21B8	hsusb0_data7				gpio_191			safe_mode
CONTROL_PADCONF_HSUSB0_DATA7[31:16]	0x4800 21B8	i2c1_scl							
CONTROL_PADCONF_I2C1_SDA[15:0]	0x4800 21BC	i2c1_sda							
CONTROL_PADCONF_I2C1_SDA[31:16]	0x4800 21BC	i2c2_scl				gpio_168			safe_mode
CONTROL_PADCONF_I2C2_SDA[15:0]	0x4800 21C0	i2c2_sda				gpio_183			safe_mode
CONTROL_PADCONF_I2C2_SDA[31:16]	0x4800 21C0	i2c3_scl				gpio_184			safe_mode
CONTROL_PADCONF_I2C3_SDA[15:0]	0x4800 21C4	i2c3_sda				gpio_185			safe_mode
CONTROL_PADCONF_I2C3_SDA[31:16]	0x4800 21C4	hdq_sio	sys_altclk	i2c2_sccbe	i2c3_sccbe	gpio_170			safe_mode
CONTROL_PADCONF_MCSPI1_CLK[15:0]	0x4800 21C8	mcspi1_clk	mmc2_dat4			gpio_171			safe_mode
CONTROL_PADCONF_MCSPI1_CLK[31:16	0x4800 21C8	mcspi1_simo	mmc2_dat5			gpio_172			safe_mode
CONTROL_PADCONF_MCSPI1_SOMI[15:0]	0x4800 21CC	mcspi1_somi	mmc2_dat6			gpio_173			safe_mode
CONTROL_PADCONF_MCSPI1_SOMI[31:16]	0x4800 21CC	mcspi1_cs0	mmc2_dat7			gpio_174			safe_mode
CONTROL_PADCONF_MCSPI1_CS1[15:0]	0x4800 21D0	mcspi1_cs1	adpllv2d_dithering_en2		mmc3_cmd	gpio_175			safe_mode
CONTROL_PADCONF_MCSPI1_CS1[31:16]	0x4800 21D0	mcspi1_cs2			mmc3_clk	gpio_176			safe_mode
CONTROL_PADCONF_MCSPI1_CS3[15:0]	0x4800 21D4	mcspi1_cs3		hsusb2_tll_data2	hsusb2_data2	gpio_177	mm2_txdat		safe_mode
CONTROL_PADCONF_MCSPI1_CS3[31:16]	0x4800 21D4	mcspi2_clk		hsusb2_tll_data7	hsusb2_data7	gpio_178			safe_mode
CONTROL_PADCONF_MCSPI2_SIMO[15:0]	0x4800 21D8	mcspi2_simo	gpt9_pwm_evt	hsusb2_tll_data4	hsusb2_data4	gpio_179			safe_mode
CONTROL_PADCONF_MCSPI2_SIMO[31:16]	0x4800 21D8	mcspi2_somi	gpt10_pwm_evt	hsusb2_tll_data5	hsusb2_data5	gpio_180			safe_mode
CONTROL_PADCONF_MCSPI2_CS0[15:0]	0x4800 21DC	mcspi2_cs0	gpt11_pwm_evt	hsusb2_tll_data6	hsusb2_data6	gpio_181			safe_mode
CONTROL_PADCONF_MCSPI2_CS0[31:16]	0x4800 21DC	mcspi2_cs1	gpt8_pwm_evt	hsusb2_tll_data3	hsusb2_data3	gpio_182	mm2_txen_n		safe_mode
CONTROL_PADCONF_SYS_NIRQ[15:0]	0x4800 21E0	sys_nirq				gpio_0			safe_mode
CONTROL_PADCONF_SYS_NIRQ[31:16]	0x4800 21E0	sys_clkout2				gpio_186			safe_mode
CONTROL_PADCONF_ETK_CLK[15:0]	0x4800 25D8	etk_clk	mcbsp5_clkx	mmc3_clk	hsusb1_stp	gpio_12	mm1_rxdp	hsusb1_tll_stp	hw_dbg0
CONTROL_PADCONF_ETK_CLK[31:16]	0x4800 25D8	etk_ctl		mmc3_cmd	hsusb1_clk	gpio_13		hsusb1_tll_clk	hw_dbg1
CONTROL_PADCONF_ETK_D0[15:0]	0x4800 25DC	etk_d0	mcspi3_simo	mmc3_dat4	hsusb1_data0	gpio_14	mm1_rxrcv	hsusb1_tll_data0	hw_dbg2
CONTROL_PADCONF_ETK_D0[31:16]	0x4800 25DC	etk_d1	mcspi3_somi		hsusb1_data1	gpio_15	mm1_txse0	hsusb1_tll_data1	hw_dbg3
CONTROL_PADCONF_ETK_D2[15:0]	0x4800 25E0	etk_d2	mcspi3_cs0		hsusb1_data2	gpio_16	mm1_txdat	hsusb1_tll_data2	hw_dbg4
CONTROL_PADCONF_ETK_D2[31:16]	0x4800 25E0	etk_d3	mcspi3_clk	mmc3_dat3	hsusb1_data7	gpio_17		hsusb1_tll_data7	hw_dbg5
CONTROL_PADCONF_ETK_D4[15:0]	0x4800 25E4	etk_d4	mcbsp5_dr	mmc3_dat0	hsusb1_data4	gpio_18		hsusb1_tll_data4	hw_dbg6
CONTROL_PADCONF_ETK_D4[31:16]	0x4800 25E4	etk_d5	mcbsp5_fsx	mmc3_dat1	hsusb1_data5	gpio_19		hsusb1_tll_data5	hw_dbg7

Table 7-4. Core Control Module Pad Configuration Register Fields (continued)

Register Name	Physical Address	Mode 0	Mode 1	Mode 2	Mode 3	Mode 4	Mode 5	Mode 6	Mode 7
CONTROL_PADCONF_ETK_D6[15:0]	0x4800 25E8	etk_d6	mcbsp5_dx	mmc3_dat2	hsusb1_data6	gpio_20		hsusb1_tll_data6	hw_dbg8
CONTROL_PADCONF_ETK_D6[31:16]	0x4800 25E8	etk_d7	mcspi3_cs1	mmc3_dat7	hsusb1_data3	gpio_21	mm1_txen_n	hsusb1_tll_data3	hw_dbg9
CONTROL_PADCONF_ETK_D8[15:0]	0x4800 25EC	etk_d8	Reserved for non-GP devices	mmc3_dat6	hsusb1_dir	gpio_22		hsusb1_tll_dir	hw_dbg10
CONTROL_PADCONF_ETK_D8[31:16]	0x4800 25EC	etk_d9	Reserved for non-GP devices	mmc3_dat5	hsusb1_nxt	gpio_23	mm1_rxdm	hsusb1_tll_nxt	hw_dbg11
CONTROL_PADCONF_ETK_D10[15:0]	0x4800 25F0	etk_d10		uart1_rx	hsusb2_clk	gpio_24		hsusb2_tll_clk	hw_dbg12
CONTROL_PADCONF_ETK_D10[31:16]	0x4800 25F0	etk_d11			hsusb2_stp	gpio_25	mm2_rxdp	hsusb2_tll_stp	hw_dbg13
CONTROL_PADCONF_ETK_D12[15:0]	0x4800 25F4	etk_d12			hsusb2_dir	gpio_26		hsusb2_tll_dir	hw_dbg14
CONTROL_PADCONF_ETK_D12[31:16]	0x4800 25F4	etk_d13			hsusb2_nxt	gpio_27	mm2_rxdm	hsusb2_tll_nxt	hw_dbg15
CONTROL_PADCONF_ETK_D14[15:0]	0x4800 25F8	etk_d14			hsusb2_data0	gpio_28	mm2_rxrcv	hsusb2_tll_data0	hw_dbg16
CONTROL_PADCONF_ETK_D14[31:16]	0x4800 25F8	etk_d15			hsusb2_data1	gpio_29	mm2_txse0	hsusb2_tll_data1	hw_dbg17

Table 7-5. Core Control Module D2D Pad Configuration Register Fields

Register Name	Physical Address	Mode 0	Mode 1	Mode 2	Mode 3	Mode 4	Mode 5	Mode 6	Mode 7
CONTROL_PADCONF_SAD2D_MCAD0[15:0]	0x4800 21E4	sad2d_mcad0	mad2d_mcad0						
CONTROL_PADCONF_SAD2D_MCAD0[31:16]	0x4800 21E4	sad2d_mcad1	mad2d_mcad1						
CONTROL_PADCONF_SAD2D_MCAD2[15:0]	0x4800 21E8	sad2d_mcad2	mad2d_mcad2						
CONTROL_PADCONF_SAD2D_MCAD2[31:16]	0x4800 21E8	sad2d_mcad3	mad2d_mcad3						
CONTROL_PADCONF_SAD2D_MCAD4[15:0]	0x4800 21EC	sad2d_mcad4	mad2d_mcad4						
CONTROL_PADCONF_SAD2D_MCAD4[31:16]	0x4800 21EC	sad2d_mcad5	mad2d_mcad5						
CONTROL_PADCONF_SAD2D_MCAD6[15:0]	0x4800 21F0	sad2d_mcad6	mad2d_mcad6						
CONTROL_PADCONF_SAD2D_MCAD6[31:16]	0x4800 21F0	sad2d_mcad7	mad2d_mcad7						
CONTROL_PADCONF_SAD2D_MCAD8[15:0]	0x4800 21F4	sad2d_mcad8	mad2d_mcad8						
CONTROL_PADCONF_SAD2D_MCAD8[31:16]	0x4800 21F4	sad2d_mcad9	mad2d_mcad9						
CONTROL_PADCONF_SAD2D_MCAD10[15:0]	0x4800 21F8	sad2d_mcad10	mad2d_mcad10						
CONTROL_PADCONF_SAD2D_MCAD10[31:16]	0x4800 21F8	sad2d_mcad11	mad2d_mcad11						
CONTROL_PADCONF_SAD2D_MCAD12[15:0]	0x4800 21FC	sad2d_mcad12	mad2d_mcad12						
CONTROL_PADCONF_SAD2D_MCAD12[31:16]	0x4800 21FC	sad2d_mcad13	mad2d_mcad13						
CONTROL_PADCONF_SAD2D_MCAD14[15:0]	0x4800 2200	sad2d_mcad14	mad2d_mcad14						
CONTROL_PADCONF_SAD2D_MCAD14[31:16]	0x4800 2200	sad2d_mcad15	mad2d_mcad15						
CONTROL_PADCONF_SAD2D_MCAD16[15:0]	0x4800 2204	sad2d_mcad16	mad2d_mcad16						
CONTROL_PADCONF_SAD2D_MCAD16[31:16]	0x4800 2204	sad2d_mcad17	mad2d_mcad17						
CONTROL_PADCONF_SAD2D_MCAD18[15:0]	0x4800 2208	sad2d_mcad18	mad2d_mcad18						
CONTROL_PADCONF_SAD2D_MCAD18[31:16]	0x4800 2208	sad2d_mcad19	mad2d_mcad19						
CONTROL_PADCONF_SAD2D_MCAD20[15:0]	0x4800 220C	sad2d_mcad20	mad2d_mcad20						

Table 7-5. Core Control Module D2D Pad Configuration Register Fields (continued)

Register Name	Physical Address	Mode 0	Mode 1	Mode 2	Mode 3	Mode 4	Mode 5	Mode 6	Mode 7
CONTROL_PADCONF_SAD2D_MCAD20[31:16]	0x4800 220C	sad2d_mcad21	mad2d_mcad21						
CONTROL_PADCONF_SAD2D_MCAD22[15:0]	0x4800 2210	sad2d_mcad22	mad2d_mcad22						
CONTROL_PADCONF_SAD2D_MCAD22[31:16]	0x4800 2210	sad2d_mcad23	mad2d_mcad23						
CONTROL_PADCONF_SAD2D_MCAD24[15:0]	0x4800 2214	sad2d_mcad24	mad2d_mcad24						
CONTROL_PADCONF_SAD2D_MCAD24[31:16]	0x4800 2214	sad2d_mcad25	mad2d_mcad25						
CONTROL_PADCONF_SAD2D_MCAD26[15:0]	0x4800 2218	sad2d_mcad26	mad2d_mcad26						
CONTROL_PADCONF_SAD2D_MCAD26[31:16]	0x4800 2218	sad2d_mcad27	mad2d_mcad27						
CONTROL_PADCONF_SAD2D_MCAD28[15:0]	0x4800 221C	sad2d_mcad28	mad2d_mcad28						
CONTROL_PADCONF_SAD2D_MCAD28[31:16]	0x4800 221C	sad2d_mcad29	mad2d_mcad29						
CONTROL_PADCONF_SAD2D_MCAD30[15:0]	0x4800 2220	sad2d_mcad30	mad2d_mcad30						
CONTROL_PADCONF_SAD2D_MCAD30[31:16]	0x4800 2220	sad2d_mcad31	mad2d_mcad31						
CONTROL_PADCONF_SAD2D_MCAD32[15:0]	0x4800 2224	sad2d_mcad32	mad2d_mcad32						
CONTROL_PADCONF_SAD2D_MCAD32[31:16]	0x4800 2224	sad2d_mcad33	mad2d_mcad33						
CONTROL_PADCONF_SAD2D_MCAD34[15:0]	0x4800 2228	sad2d_mcad34	mad2d_mcad34						
CONTROL_PADCONF_SAD2D_MCAD34[31:16]	0x4800 2228	sad2d_mcad35	mad2d_mcad35						
CONTROL_PADCONF_SAD2D_MCAD36[15:0]	0x4800 222C	sad2d_mcad36	mad2d_mcad36						
CONTROL_PADCONF_SAD2D_MCAD36[31:16]	0x4800 222C	sad2d_clk26mi							safe_mode
CONTROL_PADCONF_SAD2D_NRESPWRON[15:0]	0x4800 2230	sad2d_nrespwron							
CONTROL_PADCONF_SAD2D_NRESPWRON[31:16]	0x4800 2230	sad2d_nreswarm							
CONTROL_PADCONF_SAD2D_ARMNIRQ[15:0]	0x4800 2234	sad2d_armnirq							
CONTROL_PADCONF_SAD2D_ARMNIRQ[31:16]	0x4800 2234	sad2d_umafiq							
CONTROL_PADCONF_SAD2D_SPINT[15:0]	0x4800 2238	sad2d_spint				gpio_187			
CONTROL_PADCONF_SAD2D_SPINT[31:16]	0x4800 2238	sad2d_frint				gpio_32			
CONTROL_PADCONF_SAD2D_DMAREQ0[15:0]	0x4800 223C	sad2d_dmareq0		uart2_dma_tx	mmc1_dma_tx				
CONTROL_PADCONF_SAD2D_DMAREQ0[31:16]	0x4800 223C	sad2d_dmareq1		uart2_dma_rx	mmc1_dma_rx				
CONTROL_PADCONF_SAD2D_DMAREQ2[15:0]	0x4800 2240	sad2d_dmareq2	uart1_dma_tx		uart3_dma_tx				
CONTROL_PADCONF_SAD2D_DMAREQ2[31:16]	0x4800 2240	sad2d_dmareq3	uart1_dma_rx		uart3_dma_rx				
CONTROL_PADCONF_SAD2D_NTRST[15:0]	0x4800 2244	sad2d_ntrst							
CONTROL_PADCONF_SAD2D_NTRST[31:16]	0x4800 2244	sad2d_tdi							
CONTROL_PADCONF_SAD2D_TDO[15:0]	0x4800 2248	sad2d_tdo							
CONTROL_PADCONF_SAD2D_TDO[31:16]	0x4800 2248	sad2d_tms							
CONTROL_PADCONF_SAD2D_TCK[15:0]	0x4800 224C	sad2d_tck							
CONTROL_PADCONF_SAD2D_TCK[31:16]	0x4800 224C	sad2d_rtck							
CONTROL_PADCONF_SAD2D_MSTDBY[15:0]	0x4800 2250	sad2d_mstdby							
CONTROL_PADCONF_SAD2D_MSTDBY[31:16]	0x4800 2250	sad2d_idlereq							
CONTROL_PADCONF_SAD2D_IDLEACK[15:0]	0x4800 2254	sad2d_idleack							

Table 7-5. Core Control Module D2D Pad Configuration Register Fields (continued)

Register Name	Physical Address	Mode 0	Mode 1	Mode 2	Mode 3	Mode 4	Mode 5	Mode 6	Mode 7
CONTROL_PADCONF_SAD2D_IDLEACK[31:16]	0x4800 2254	sad2d_mwrite	mad2d_swrite						
CONTROL_PADCONF_SAD2D_SWRITE[15:0]	0x4800 2258	sad2d_swrite	mad2d_mwrite						
CONTROL_PADCONF_SAD2D_SWRITE[31:16]	0x4800 2258	sad2d_mread	mad2d_sread						
CONTROL_PADCONF_SAD2D_SREAD[15:0]	0x4800 225C	sad2d_sread	mad2d_mread						
CONTROL_PADCONF_SAD2D_SREAD[31:16]	0x4800 225C	sad2d_mbusflag	mad2d_sbusflag						
CONTROL_PADCONF_SAD2D_SBUSFLAG15:0]	0x4800 2260	sad2d_sbusflag	mad2d_mbusflag						

CAUTION

The D2D pads are available in stacked mode only.

Table 7-6. Wake-Up Control Module Pad Configuration Register Fields

Register Name	Physical Address	Mode 0	Mode 1	Mode 2	Mode 3	Mode 4	Mode 5	Mode 6	Mode 7
CONTROL_PADCONF_I2C4_SCL[15:0]	0x4800 2A00	i2c4_scl	sys_nvmode1						safe_mode
CONTROL_PADCONF_I2C4_SCL[31:16]	0x4800 2A00	i2c4_sda	sys_nvmode2						safe_mode
CONTROL_PADCONF_SYS_32K[15:0]	0x4800 2A04	sys_32k							
CONTROL_PADCONF_SYS_32K[31:16]	0x4800 2A04	sys_clkreq				gpio_1			safe_mode
CONTROL_PADCONF_SYS_NRESWARM[15:0]	0x4800 2A08	sys_nreswarm				gpio_30			safe_mode
CONTROL_PADCONF_SYS_NRESWARM[31:16]	0x4800 2A08	sys_boot0				gpio_2			safe_mode
CONTROL_PADCONF_SYS_BOOT1[15:0]	0x4800 2A0C	sys_boot1				gpio_3			safe_mode
CONTROL_PADCONF_SYS_BOOT1[31:16]	0x4800 2A0C	sys_boot2				gpio_4			safe_mode
CONTROL_PADCONF_SYS_BOOT3[15:0]	0x4800 2A10	sys_boot3				gpio_5			safe_mode
CONTROL_PADCONF_SYS_BOOT3[31:16]	0x4800 2A10	sys_boot4	mmc2_dir_dat2			gpio_6			safe_mode
CONTROL_PADCONF_SYS_BOOT5[15:0]	0x4800 2A14	sys_boot5	mmc2_dir_dat3			gpio_7			safe_mode
CONTROL_PADCONF_SYS_BOOT5[31:16]	0x4800 2A14	sys_boot6				gpio_8			safe_mode
CONTROL_PADCONF_SYS_OFF_MODE[15:0]	0x4800 2A18	sys_off_mode				gpio_9			safe_mode
CONTROL_PADCONF_SYS_OFF_MODE[31:16]	0x4800 2A18	sys_clkout1				gpio_10			safe_mode
CONTROL_PADCONF_JTAG_NTRST[15:0]	0x4800 2A1C	jtag_ntrst							
CONTROL_PADCONF_JTAG_NTRST[31:16]	0x4800 2A1C	jtag_tck							
CONTROL_PADCONF_JTAG_TMS_TMSC[15:0]	0x4800 2A20	jtag_tms_tmsc							
CONTROL_PADCONF_JTAG_TMS_TMSC[31:16]	0x4800 2A20	jtag_tdi							
CONTROL_PADCONF_JTAG_EMU0[15:0]	0x4800 2A24	jtag_emu0				gpio_11			safe_mode
CONTROL_PADCONF_JTAG_EMU0[31:16]	0x4800 2A24	jtag_emu1				gpio_31			safe_mode

Table 7-6. Wake-Up Control Module Pad Configuration Register Fields (continued)

Register Name	Physical Address	Mode 0	Mode 1	Mode 2	Mode 3	Mode 4	Mode 5	Mode 6	Mode 7
CONTROL_PADCONF_SAD2D_SWAKEUP[15:0](1)	0x4800 2A4C	sad2d_swakeup							
CONTROL_PADCONF_SAD2D_SWAKEUP[31:16]	0x4800 2A4C	jtag_rtck							
CONTROL_PADCONF_JTAG_TDO[15:0]	0x4800 2A50	jtag_tdo							

(1) The D2D_swakeup pad is available in stacked mode only.

Note: Pad names are signal names avalaible in mode 0.

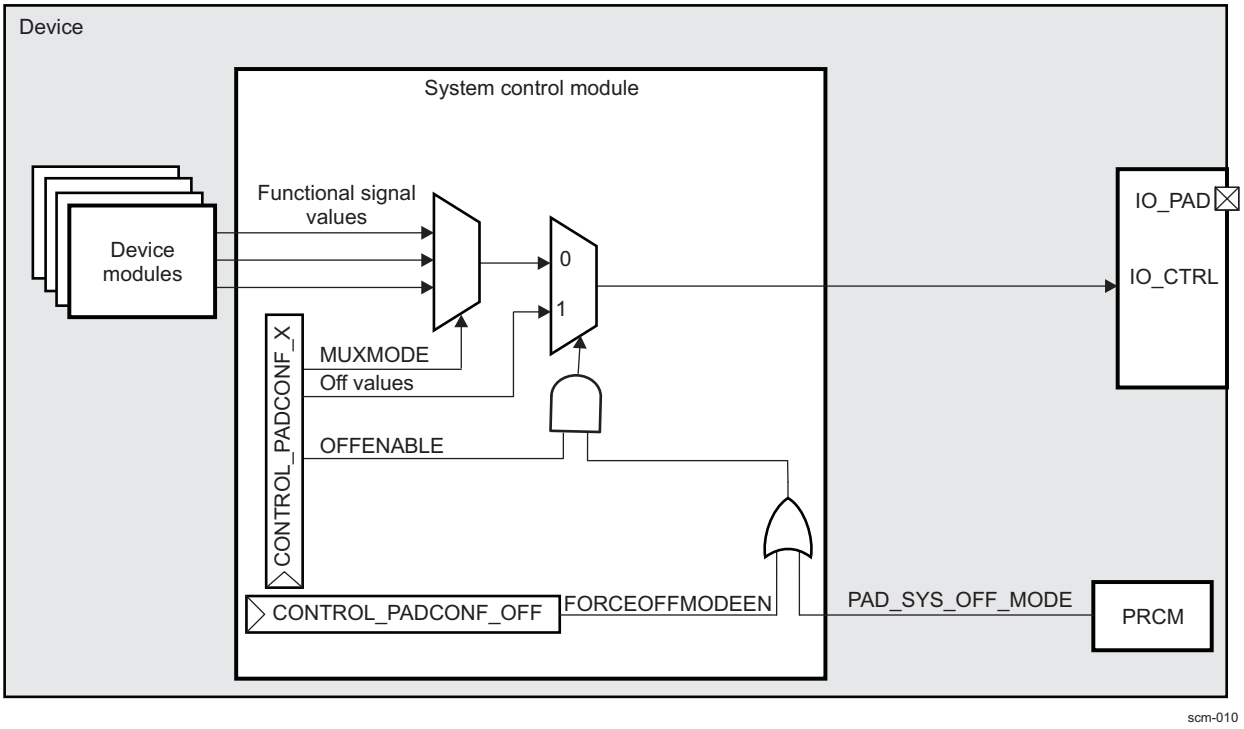
For more information on the pad default states, see the *OMAP34xx Data Manual*.

7.4.4.4 System Off Mode

When off mode is active (PAD_SYS_OFF_MODE = 0b1 from PRCM or FORCEOFFMODEENABLE bit CONTROL.CONTROL_PADCONF_OFF[0] = 0b1), the off mode values field CONTROL.CONTROL_PADCONF_X overrides the pad state when OFFENABLE bit CONTROL.CONTROL_PADCONF_X is set.

Figure 7-9 shows the off mode pad control.

Figure 7-9. System Off Mode Pad Control Overview



If off mode is active and the OFFENABLE bit is set to disable, the pad keeps the AND value of the configuration (input/output, PU/PD) it had before going into off mode:

- For an input, the pad is isolated and the pull remains active.
- For an output, the value is latched before going into off mode, to drive the same value in off mode

For more information about off mode, see Chapter 4, *Power, Reset, and Clock Management.*

For more information about the preliminary settings that must be done before performing OFF <-> ON transitions, see Section 7.5.3, *Off Mode Preliminary Settings*.

7.4.4.4.1 Save-and-Restore Mechanism

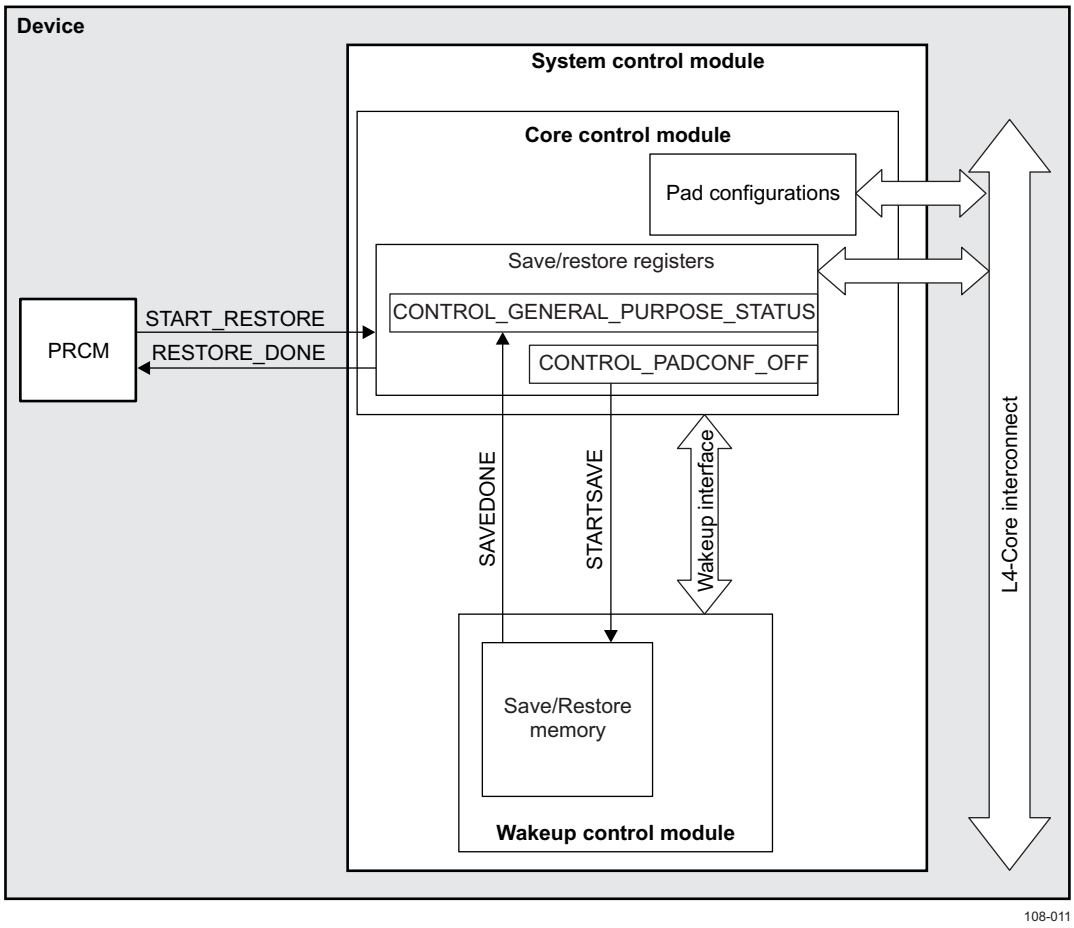
Before going to off mode there is a context saving of the device. The save-and-restore mechanism saves the pad configuration registers (in the CORE power domain) in a WKUP power domain memory (physical addresses 0x4800 2600 to 0x4800 29FC) before going to off mode, and restores those registers when returning from off mode. This mechanism uses the dedicated wake-up interface between the core control module and the wake-up control module.

The save mechanism in the pad configuration registers is activated by setting the STARTSAVE bit CONTROL.CONTROL_PADCONF_OFF[1]. When all pad configuration registers have been saved to the wake-up memory, the status SAVEDONE bit CONTROL.CONTROL_GENERAL_PURPOSE_STATUS[0] is set. In smart-idle mode, the idleAck is returned when the save process is complete.

When returning from off mode, the registers are restored after the PRCM module asserts the START_RESTORE signal. The SCM returns a RESTORE_DONE signal to the PRCM module when the restore process completes.

Figure 7-10 shows an overview of the save-and-restore mechanism in off mode.

Figure 7-10. Save-and-Restore Mechanism Overview

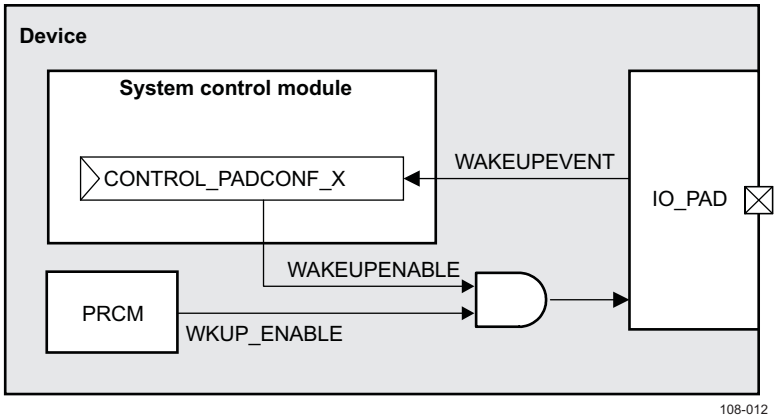


7.4.4.4.2 Wake-Up Event Detection

In off mode, wake-up event detection can also be enabled on an input pad. The pad wake-up event is latched in the WAKEUPEVENT bit CONTROL.CONTROL_PADCONF_X.

The off mode I/O pads wake-up scheme is enabled by setting the EN_I/O bit PRCM.PM _WKEN_WKUP[8]. The wake-up scheme status is transmitted by the WKUP_ENABLE signal. The wake-up event detection capability of each I/O pad of the device is individually enabled/disabled by writing WAKEUPENABLE bit CONTROL.CONTROL_PADCONF_X.

Figure 7-11. Wake-Up Event Detection Overview



For more information about the wake-up sequences, see Chapter 4, *Power, Reset, and Clock Management*.

Note: When wake-up detection is enabled for a pad, the pad must be configured as input to avoid contention between the OMAP output buffer and an external driver. If this pin is already configured as an input in active mode, hardware automatically retains the same input state; no software action is required.

If this pin is configured as an output in active mode, the OFF override function must be enabled to set the pin as an input during off mode: in the CONTROL.CONTROL_PADCONF register, set OFFENABLE to 0x1 to enable the OFF override function, and set OFFOUTENABLE to 0x1 to switch this pin to input mode.

Table 7-7 lists the bit directions of the CONTROL_PADCONF_x registers.

Table 7-7. Bit Directions for CONTROL_PADCONF_x Registers

CONTROL_PADCONF_x Bit	Bit Direction	
	0	1
PULLUDENABLE	Not activated	Activated
PULLTYPESELECT	Pulldown	Pullup
INPUTENABLE	Input enable signal inactive	Input enable signal active
OFFENABLE	Off mode values are invalid.	Off mode values are valid.
OFFOUTENABLE	Output	Input
OFFOUTVALUE	Low	High
OFFPULLUDENABLE	Not activated	Activated
OFFPULLTYPESELECT	Pulldown	Pullup
WAKEUPENABLE	Disable I/O wake-up function.	Enable I/O wake-up function.
WAKEUPEVENT	Wake-up event not detected	Detect wake-up event.

7.4.5 Extended-Drain I/O Pin and PBIAS Cell

The device mainly supports 1.8-V I/O voltage on its interfaces, with the exception of MMC/SD/SDIO1 interface, which support both 1.8-V and 3.0-V voltages via internal PBIAS generation and extended-drain I/Os.

The need for embedded extended-drain I/Os on MMC/SD/SDIO1 interface imposes the use of embedded PBIAS cells to provide 1.8-V or 3.0-V reference voltage. The PBIAS cells and the extended-drain I/Os are software-controlled by bits located in the CONTROL.CONTROL_PBIAS_LITE register of the SCM. See Section 7.6.3, *Register Descriptions*, for the description of this register.

Figure 7-12 shows the functional block diagram between the PBIAS cells and the extended I/O cells.

Figure 7-12. Functional Block Diagram

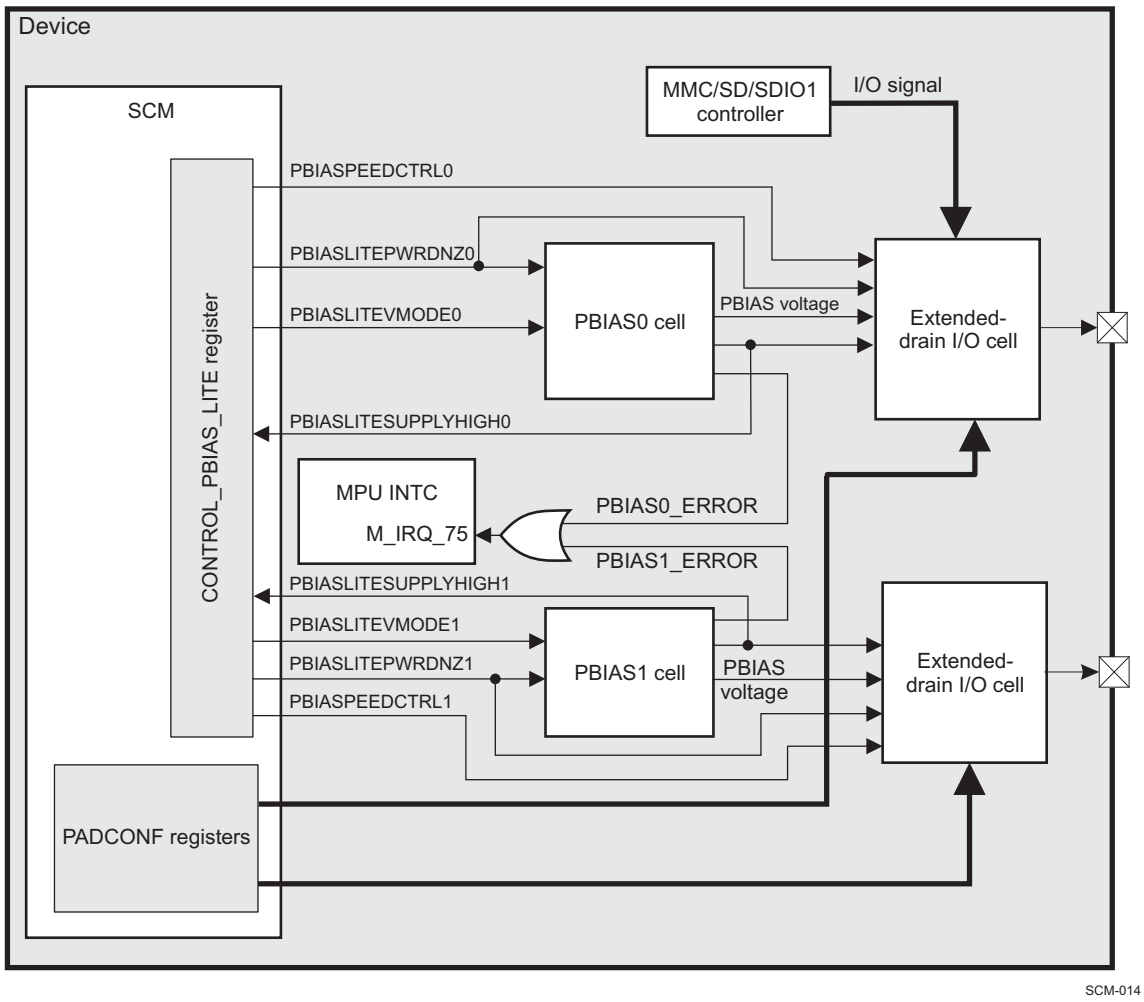


Table 7-8 describes the CONTROL.CONTROL_PBIAS_LITE bit controls for the PBIAS and the extended-drain I/O cells.

Table 7-8. PBIAS Cell and Extended-Drain I/O Pin CONTROL_PBIAS_LITE Bit Controls

Control Signals for PBIAS0 and/or Associated Extended-Drain I/O Cell	Control Signals for PBIAS1 and/or Associated Extended-Drain I/O Cell	Description
PBIASLITEPWRDNZ0	PBIASLITEPWRDNZ1	The PBIASLITEPWRDNZ0 bit is used to simultaneously protect the PBIAS0 cell and its associated (MMC/SD/SDIO1) extended-drain I/O cell when the MMC1_VDDS power supply voltage is not stable. The PBIASLITEPWRDNZ1 bit is used to simultaneously protect the PBIAS1 cell and its associated extended-drain I/O cell when the SIM_VDDS power supply voltage is not stable.

Table 7-8. PBIAS Cell and Extended-Drain I/O Pin CONTROL_PBIAS_LITE Bit Controls (continued)

Control Signals for PBIAS0 and/or Associated Extended-Drain I/O Cell	Control Signals for PBIAS1 and/or Associated Extended-Drain I/O Cell	Description
		Software must keep the PBIASLITEPWRDNZ0/PBIASLITEPWRDNZ1 signal to 0b0 whenever the MMC1_VDDS/SIM_VDDS signal is ramping. When this bit is at 0, the PAD is floating.
PBIASLITEVMODE0	PBIASLITEVMODE1	Controls the MMC1_VDDS (for PBIAS0) or SIM_VDDS (for PBIAS1) voltage level. The default state of this bit is HIGH, indicating that the voltage level is 3.0 V.
PBIASLITESUPPLYHIGH0	PBIASLITESUPPLYHIGH1	Describes whether the MMC1_VDDS (for PBIAS0) or SIM_VDDS (for PBIAS1) supply is 3.0 V or 1.8 V
PBIASSPEEDCTRL0	PBIASSPEEDCTRL1	Controls I/O cell speed
PBIASLITEVMODEERROR	PBIASLITEVMODEERROR1	Indicates whether the software-programmed VMODE level matches the SUPPLY_HI output signal
PBIAS0_ERROR	PBIAS1_ERROR	Determines whether the software-programmed PBIASLITEVMODEx level matches the PBIASLITESUPPLYHIGHx. This signal is generated only when PBIASLITEVMODEENx is HIGH.

Table 7-9 lists the power supplies for the PBIAS and the extended-drain I/O cells.

Table 7-9. Power Supplies

Name	Description
VDD2	Core voltage supply
MMC1_VDDS or SIM_VDDS (vdds_mmc1 or vdds_sim power pins of the device)	I/O supply voltage nominal 1.8 V/ 3.0 V
VDDS	1.8-V supply for the input buffer

7.4.5.1 PBIAS Cells

The PBIAS0 and PBIAS1 cells provide a bias for the extended-drain I/O cells used with high voltage for the MMC/SD/SDIO1 interface. The PBIAS cells provide a voltage reference (PBIAS voltage) for biasing extended drain in the MMC/SD/SDIO1 I/O cells. The I/O cells must be configured for low-voltage operation when the GPIO interface (gpio_120 through gpio_129 signals) is selected.

Note: The gpio_120 through gpio_129 signals are available at the mmc1 I/O pads of the device when muxmode is set to 0x4 in the CONTROL_PADCONF_MMC1_DAT0 through CONTROL_PADCONF_MMC1_DAT6 registers. In this case, PBIAS cells must be configured to provide low (1.8 V) voltage for the GPIO I/Os.

In addition to generating the bias voltage, the PBIAS cells can detect the value (1.8 V or 3.0 V) of the supply voltage (MMC1_VDDS or SIM_VDDS) and update, with its status, the PBIASLITESUPPLYHIGH0 or PBIASLITESUPPLYHIGH1 bit.

CAUTION

The PBIAS cells let the MMC/SD/SDIO1 peripheral support 1.8-V and 3.0-V voltages. A PBIAS cell is not a part of the MMC/SD/SDIO1 peripheral, but a part of the device I/O to which the MMC/SD/SDIO1 peripheral is internally connected. These device I/Os are not exclusive to the MMC/SD/SDIO1 peripheral; through I/O multiplexing they can be connected to other internal signals. It is necessary to configure the PBIAS to enable the I/Os, regardless of how I/O multiplexing is configured for these device I/Os. In other words, independently of which signal is internally connected to a device I/O powered by MMC1_VDDS or SIM_VDDS, the PBIAS must be configured.

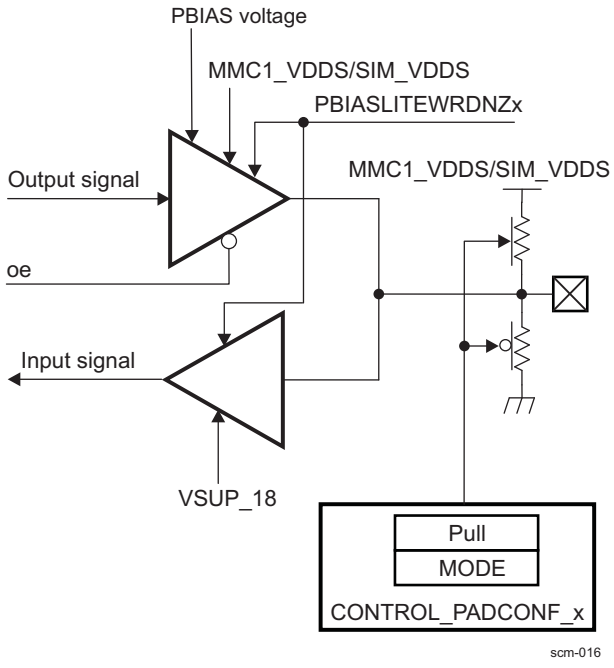
7.4.5.2 Extended-Drain I/Os

On the device, the following extended-drain I/Os are used by I/Os from the MMC/SD/SDIO1:

- mmc1_clk
- mmc1_cmd
- mmc1_dati (where i = 0 to 7)

Figure 7-13 describes the extended-drain I/O cell.

Figure 7-13. Extended-Drain I/O



The extended-drain I/O cells have the following I/O signals:

- Output signal, input signal, and oe, which comes from the selected I/O module with the correct MODE field CONTROL.CONTROL_PADCONF_X configuration.

Note: MMC/SD/SDIO1 extended-drain I/Os are muxed I/Os with mode and pull-up/pull-down configurations programmable in the SCM.

- The PBIAS voltage is a voltage reference for biasing the extended-drain in the MMC/SD/SDIO1I/O cells
- The PBIASLITEPWRDNZ0 or PBIASLITEPWRDNZ1 bits are used to protect the two I/O cells when the voltage of MMC1_VDDS or SIM_VDDS, respectively, is not stable.

CAUTION

Software must keep the PWRDNZ-related signals to 0b0 when the MMC1_VDDS or SIM_VDDS signal is ramping up/down or changing. When PBIASLITEPWRDNZ0 or PBIASLITEPWRDNZ1 is 0, the PAD is floating (the PAD may not reflect the state of the output signal, and the input signal may not reflect the state of the PAD).

- The PBIASSPEEDCTRL0 or PBIASSPEEDCTRL1 bits control the speed of I/O cells and can be used to reduce dynamic current if fast rise/fall times are not required.
- The PBIASLITESUPPLYHIGH0 bit is a status bit on the MMC1_VDDS value and is used to inform the PBIAS0 cell about the value of the MMC1_VDDS signal.
- The PBIASLITESUPPLYHIGH1 bit is a status bit on the SIM_VDDS value and is used to inform the PBIAS1 cell about the value of the SIM_VDDS signal.

For more information about the software configurations of the extended-drain I/Os and PBIAS cells, see Section 7.5.2, *Extended-Drain I/Os and PBIAS Cells Programming Guide*.

7.4.6 Band Gap Voltage and Temperature Sensor

The device supplies a voltage reference and a temperature sensor feature which are gathered in the band gap voltage and temperature sensor (BGAPTS) module. The band gap provides current and voltage reference for its internal circuits and also to other analog IP blocks. The A/D converter (ADC) produces an output value that is proportional to the silicon temperature.

Main features of the BGAPTS module are:

- A constant voltage reference output: 0.5 V
- Four constant current reference outputs of 1µA
- Analog supply is a nominal 1.8 V.
- Small A/D converter with 7-bit digital output
- OFF mode compatible
- Thermal shutdown comparator output

The band gap and the temperature sensor are software-controlled by bits located in the CONTROL.CONTROL_TEMP_SENSOR register of the SCM. See Section 7.4.7.6, *Register Descriptions*, for the description of this register.

Figure 7-14 shows the functional block diagram of the band gap and the temperature sensor module.

Figure 7-14. Functional Block Diagram

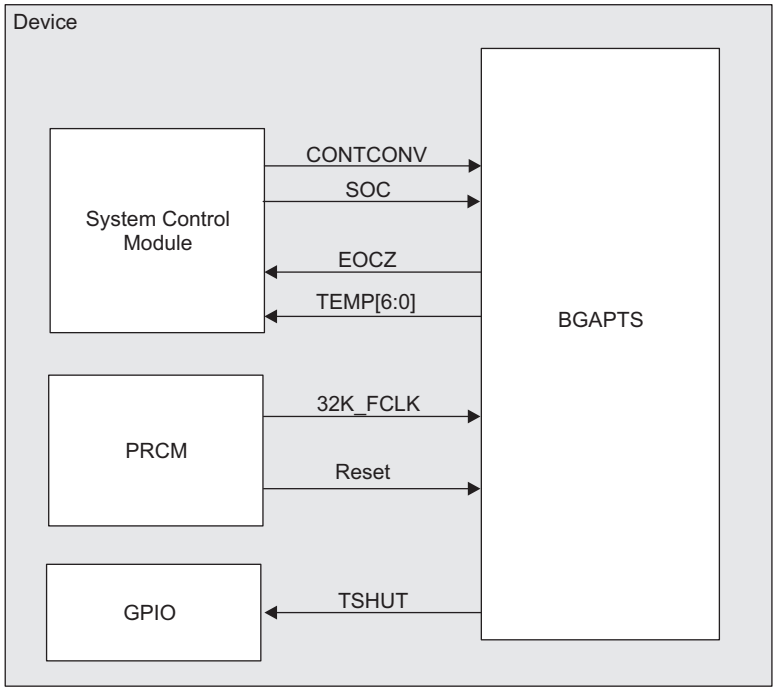


Table 7-10 describes the input and output signals for the band gap and the temperature sensor.

Table 7-10. Band Gap Voltage and Temperature Sensor Signals Description

Pin	I/O[1]	Description
SOC	I	Start Of Conversion signal. A transition to high starts a new ADC conversion cycle.
EOCZ	I	End of Conversion signal. When low, the signal indicates that the value of TEMP[6:0] is valid.
TEMP[6:0]	O	Temperature data from the temperature sensor. This value is valid when EOCZ is low.
CONTCONV	I	Configures the temperature sensor in continuous conversion mode. 0 – ADC single conversion mode 1– ADC continuous conversion mode
TSHUT	O	Thermal shutdown comparator output. The signal is low during normal operation and goes high during a thermal shutdown event.
RESET	I	When low the temperature sensor is in RESET mode. This signal needs to be LOW when OFFMODE is HIGH.
32K_FCLK	I	32-kHz functional clock from the CORE domain used by the temperature sensor during temperature conversion.

(1) I=Input, O=Output

All the configuration signals are available through the CONTROL.CONTROL_TEMP_SENSOR register.

Note: BGAPTS 32K_FCLK is used only for the temperature sensor. When ADC is not converting, this clock can be switched OFF (32K_FCLK = 0). In this case, TEMP[6:0] retains the temperature code obtained at the previous conversion.

7.4.6.1 Band Gap Voltage Reference

The band gap voltage reference feature provides several constant voltage and current references for its internal circuits, but also for other analog modules.

The voltage reference signal is a 0.5-V output which is internally used by the module and that is exported for external modules or for tests.

The current reference signals consist in four lines that supply each a 1-µA current with a 1.8-V voltage. These lines can be used to bias other modules.

7.4.6.2 Temperature Sensor

The temperature sensor feature is used to convert the temperature of the device into a decimal value coded on 7 bits. An internal ADC (realized with a resistor array) is used for conversion. The recommended operating temperatures have to be in the -40 to 125°C for standard conversion, and can go up to 160°C when using the thermal shutdown comparator output (TSHUT signal). This signal indicates a too high temperature of the device. This signal is internally connected to a GPI/O pin. See Chapter 25, *General-Purpose Interface*, for more information.

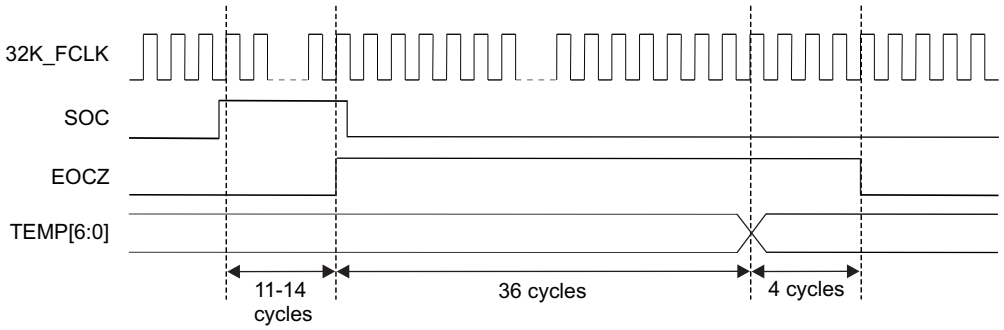
The temperature sensor offers two operating modes: a single conversion mode and a continuous conversion mode.

7.4.6.2.1 Single Conversion Mode (CONTCONV = 0)

When the ADC is idle (EOCZ = 0), it is possible to ask for a single temperature conversion. To initiate a new conversion, the start of conversion (SOC) signal should be asserted and maintained high until the end of conversion (EOCZ) signal goes high, after which the SOC should be made low by writing 0 to the CONTROL.CONTROL_TEMP_SENSOR[8] SOC bit. Conversion completion is indicated by a negative edge on EOCZ (CONTROL.CONTROL_TEMP_SENSOR[7] EOCZ bit).

The timing sequence for a single temperature conversion is shown in Figure 7-15.

Figure 7-15. Single Conversion Mode (CONTCONV = 0)

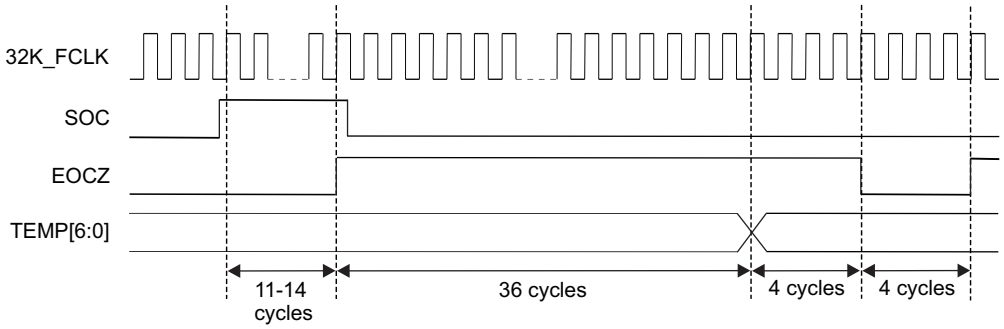


7.4.6.2.2 Continuous Conversion Mode (CONTCONV = 1)

When the ADC is idle (EOCZ = 0), it is possible to ask for a single temperature conversion. To initiate a new conversion, the SOC signal should be asserted and maintained high until the EOCZ signal goes high, after which the SOC should be made low by writing 0 to the CONTROL.CONTROL_TEMP_SENSOR[8] SOC bit. The ADC starts converting continuously, and when a conversion is complete, data is written to the CONTROL.CONTROL_TEMP_SENSOR[6:0] TEMP bits and EOCZ goes low after four clock cycles.

The timing sequence for a continuous temperature conversion is shown in Figure 7-16.

Figure 7-16. Continuous Conversion Mode (CONTCONV = 1)



7.4.6.2.3 ADC Codes Versus Temperature

Table 7-11 gives the temperature range corresponding to each value of the CONTROL.CONTROL_SENSOR[6:0] TEMP bits.

Table 7-11. ADC Codes Versus Temperature

ADC Code	Temperature °C		ADC Code	Temperature °C		ADC Code	Temperature °C		ADC Code	Temperature °C	
	From	To		From	To		From	To		From	To
0	-40	-40	32	-0.7	0.8	64	44.9	46.3	96	89.3	90.6
1	-40	-40	33	0.8	2.3	65	46.3	47.7	97	90.6	92
2	-40	-40	34	2.3	3.7	66	47.7	49.1	98	92	93.4
3	-40	-40	35	3.7	5.1	67	49.1	50.5	99	93.4	94.7
4	-40	-40	36	5.1	6.6	68	50.5	51.9	100	94.7	96.1
5	-40	-38.9	37	6.6	8	69	51.9	53.3	101	96.1	97.5
6	-38.9	-37.5	38	8	9.4	70	53.3	54.6	102	97.5	98.9
7	-37.5	-36.1	39	9.4	10.8	71	54.6	56	103	98.9	100.2
8	-36.1	-33.3	40	10.8	12.3	72	56	57.4	104	100.2	101.6
9	-33.3	-31.8	41	12.3	13.7	73	57.4	58.8	105	101.6	103
10	-31.8	-30.4	42	13.7	15.1	74	58.8	60.2	106	103	104.3
11	-30.4	-29	43	15.1	16.5	75	60.2	61.6	107	104.3	105.7
12	-29	-27.5	44	16.5	17.9	76	61.6	63	108	105.7	107.1
13	-27.5	-26.1	45	17.9	19.4	77	63	64.4	109	107.1	108.5
14	-26.1	-24.7	46	19.4	20.8	78	64.4	65.7	110	108.5	109.8
15	-24.7	-23.3	47	20.8	22.2	79	65.7	67.1	111	109.8	111.2
16	-23.3	-21.9	48	22.2	23.6	80	67.1	68.5	112	111.2	112.6
17	-21.9	-20.5	49	23.6	25.1	81	68.5	69.9	113	112.6	114
18	-20.5	-19.1	50	25.1	26.5	82	69.9	71.3	114	114	115.3
19	-19.1	-17.7	51	26.5	27.9	83	71.3	72.7	115	115.3	116.7
20	-17.7	-16.3	52	27.9	29.3	84	72.7	74.1	116	116.7	118.1
21	-16.3	-14.9	53	29.3	30.7	85	74.1	75.5	117	118.1	119.4
22	-14.9	-13.4	54	30.7	32.1	86	75.5	76.9	118	119.4	120.8
23	-13.4	-12	55	32.1	33.5	87	76.9	78.3	119	120.8	122.2
24	-12	-10.6	56	33.5	34.9	88	78.3	79.7	120	122.2	123.5
25	-10.6	-9.2	57	34.9	36.4	89	79.7	81.1	121	123.5	124.9
26	-9.2	-7.8	58	36.4	37.8	90	81.1	82.3	122	124.9	125
27	-7.8	-6.4	59	37.8	39.2	91	82.3	83.8	123	125	125
28	-6.4	-5	60	39.2	40.6	92	83.8	85.2	124	125	125

Table 7-11. ADC Codes Versus Temperature (continued)

29	-5	-3.5	61	40.6	42	93	85.2	86.5	125	125	125
30	-3.5	-2.1	62	42	43.4	94	86.5	87.9	126	125	125
31	-2.1	-0.7	63	43.4	44.9	95	87.9	89.3	127	125	125

7.4.7 Functional Register Description

7.4.7.1 Static Device Configuration Registers

Table 7-12 describes the static device configuration registers.

Table 7-12. Static Device Configuration Registers

Physical Address	Register Name	Description	Access
0x4800 2274	CONTROL_DEVCONF0	Module dedicated configurations	R/W
0x4800 22D8	CONTROL_DEVCONF1	Module dedicated configurations	R/W

These registers allow the static configuration of device modules such as USB, McBSP, and I^2C. For example, they allow selecting external or internal clocks for McBSP modules.

7.4.7.2 Control CSIRXFE Register

Table 7-13 describes the CONTROL.CONTROL_CSIRXFE register, which controls some CSIRXFE analog cell settings. The CSIRXFE analog cell is used in the interface between a digital camera module and a mobile engine phone.

Table 7-13. Control CSIRXFE Register

Physical Address	Register Name	Description	Access
0x4800 22DC	CONTROL_CSIRXFE	Controls settings for CSIb interface pin	R/W

7.4.7.3 MPU and/or DSP (IVA2.2) MSuspend Configuration Registers

Table 7-14 describes the MPU and/or DSP (IVA2.2) MSuspend configuration registers.

Table 7-14. MSuspendMux Control Registers

Physical Address	Register Name	Description	Access
0x4800 2290	CONTROL_MSUSPENDMUX_0	Control the use of MSuspend signals at module level	R/W
0x4800 2294	CONTROL_MSUSPENDMUX_1		R/W
0x4800 2298	CONTROL_MSUSPENDMUX_2		R/W
0x4800 229C	CONTROL_MSUSPENDMUX_3		R/W
0x4800 22A0	CONTROL_MSUSPENDMUX_4		R/W
0x4800 22A4	CONTROL_MSUSPENDMUX_5		R/W

These registers provide an entry for each module that must consider the MSuspend signals from the processors (MPU and/or DSP). For each module, the sensitivity to the MSuspend signals is defined within five possibilities (coded using 3 bits):

- 0b000: No sensitivity; no MSuspend signal reaches the module.
- 0b001: Sensitivity to the MPU MSuspend signal (the DSP signal is ignored)
- 0b010: Sensitivity to the DSP MSuspend signal (the MPU signal is ignored)
- 0b011: Sensitivity to the logical ORed MPU and DSP MSuspend signals
- 0b100: Sensitivity to the logical ANDed MPU and DSP MSuspend signals

- Other values: No sensitivity; no MSuspend signal reaches the module.

The logic used to combine the MSuspend signals from the processors is implemented within the SCM. MSuspend signals are active low.

CAUTION

Use care when using combined sensitivity settings (ANDing or ORing DSP and MPU MSuspend signals).

ORing the DSP and MPU MSuspend signals creates a situation where the module is suspended when at least one processor is under debug; therefore, when one processor is halted, stepping within the code of the other one does not change the module suspended state.

Not all modules use the MSUSPEND signal. See the TRM chapter for each module to determine whether the module supports the MSUSPEND signal.

All MSUSPEND signals coming out of the MPU and DSP are resynchronized within the SCM by using the control module interface clock.

7.4.7.4 IVA2.2 Boot Registers

Table 7-15 describes the IVA2.2 boot registers.

Table 7-15. IVA2.2 Boot Registers

Physical Address	Register Name	Description	Access
0x4800 2400	CONTROL_IVA2_BOOTADDR	IVA2.2 boot loader address register	R/W
0x4800 2404	CONTROL_IVA2_BOOTMOD	IVA2.2 boot mode register	R/W

The CONTROL.CONTROL_IVA2_BOOTADDR register defines the physical address for the IVA2 boot loader and drives the IVA2_BOOTADDR[21:0] signals out from the control block to the IVA2 subsystem.

The CONTROL.CONTROL_IVA2_BOOTMOD register defines the IVA2 boot mode and drives the IVA2_BOOTMOD[3:0] signals out from the control block to the IVA2.2 subsystem. Based on the value of IVA2_BOOTMOD[3:0], the ROM boot loader executes different boot modes of IVA2.2.

Table 7-16 lists the IVA2.2 boot modes.

Table 7-16. IVA2.2 Boot Modes

IVA2_BOOTMOD[3:0] Value	Meaning
0x0	Direct boot: The ROM loader is not executed. Instead, IVA2.2 directly starts executing the bootstrap at the address contained in the CONTROL_IVA2_BOOTADDR register.
0x1	Idle boot: The boot loader executes the IDLE instruction.
0x2	Wait in self-loop boot: The boot loader puts IVA2.2 in a self-loop.
0x3	User-defined bootstrap mode: The boot loader copies the boot strap into internal memory and branches to it.
0x4	The boot loader executes the default context restore code, which is part of the ROM boot loader.

For further information, see Chapter 14, *IVA2.2 Subsystem*.

7.4.7.5 PBIAS LITE Control Register

Table 7-17 describes the CONTROL.CONTROL_PBIAS_LITE register, which controls some settings of PBIAS LITE cells for MMC/SD/SDI/O interface.

Table 7-17. PBIAS Control Register

Physical Address	Register Name	Description	Access
0x4800 2520	CONTROL_PBIAS_LITE	Control settings for PBIAS and extended-drain I/O cells	R/W

For more information about the PBIAS cells, see Section 7.4.5, *Extended-Drain I/O Pin and PBIAS Cell*.

7.4.7.6 Temperature Sensor Control Register

Table 7-18 describes the CONTROL.CONTROL_TEMP_SENSOR register, which controls the temperature sensor.

Table 7-18. Temperature Sensor Register

Physical Address	Register Name	Description	Access
0x4800 2524	CONTROL_TEMP_SENSOR	Temperature sensor control register	R/W

7.4.7.7 CSI Reciever Control Register

Table 7-19 describes the CONTROL.CONTROL_CSI register, which controls the CSIb receiver trimming override.

Table 7-19. CSI Receiver Control Register

Physical Address	Register Name	Description	Access
0x4800 2530	CONTROL_CSI	CSIb receiver trimming override	R/W

7.4.8 Protection Status Registers

Table 7-20 lists the status registers.

Table 7-20. Protection Status Registers

Physical Address	Register Name	Description	Access
0x4800 22E4	CONTROL_PROT_ERR_STATUS	Protection error status register	Public (R)
0x4800 22E8	CONTROL_PROT_ERR_STATUS_DEBUG	Protection error status register debug	Public (R)

These registers do not depend on the device type.

The CONTROL.CONTROL_PROT_ERR_STATUS, and CONTROL.CONTROL_PROT_ERR_STATUS_DEBUG registers can be read in public mode, but can not be written.

These bits are cleared when the L3 and L4 firewall embedded error log registers are cleared. All bits in these registers reflect device internal events related to the device protection.

On a specific event (signal rising edge), the corresponding bit is set. On a rising edge, the input signal must stay high for at least two interface clocks periods to be recognized. The software must clear each bit after reviewing the events.

When a protection violation occurs, the following bits are set :

- In application mode
  - CONTROL.CONTROL_PROT_ERR_STATUS [00] = OCM-ROM protection violation
  - CONTROL.CONTROL_PROT_ERR_STATUS [01] = OCM-RAM protection violation
  - CONTROL.CONTROL_PROT_ERR_STATUS [02] = GPMC protection violation
  - CONTROL.CONTROL_PROT_ERR_STATUS [04] = SMS protection violation
  - CONTROL.CONTROL_PROT_ERR_STATUS [06] = IVA2.2 protection violation
  - CONTROL.CONTROL_PROT_ERR_STATUS [07] = L4-Core protection violation

  - CONTROL.CONTROL_PROT_ERR_STATUS [12] = L3 RT protection violation
  - CONTROL.CONTROL_PROT_ERR_STATUS [15] = D2D protection violation
  - CONTROL.CONTROL_PROT_ERR_STATUS [16] = L4-Peri protection violation
  - CONTROL.CONTROL_PROT_ERR_STATUS [17] = L4-Emu protection violation
- In debug mode
  - CONTROL.CONTROL_PROT_ERR_STATUS_DEBUG [00] = OCM-ROM protection violation
  - CONTROL.CONTROL_PROT_ERR_STATUS_DEBUG [01] = OCM-RAM protection violation
  - CONTROL.CONTROL_PROT_ERR_STATUS_DEBUG [02] = GPMC protection violation
  - CONTROL.CONTROL_PROT_ERR_STATUS_DEBUG [03] = SMS protection violation
  - CONTROL.CONTROL_PROT_ERR_STATUS_DEBUG [06] = IVA2.2 protection violation
  - CONTROL.CONTROL_PROT_ERR_STATUS_DEBUG [12] = L3 RT protection violation

For more information, see Chapter 5, *Interconnect*.

7.4.9 SDRC Registers

Table 7-21 lists the SDRAM controller (SDRC) registers, which export reset values to the SDRC registers.

Table 7-21. SDRC Registers

Physical Address	Register Name	Description	Access
0x4800 2460	CONTROL_SDRC_SHARING	SDRC sharing configuration	R/W
0x4800 2464	CONTROL_SDRC_MCFG0	SDRC configuration register 0	R/W
0x4800 2468	CONTROL_SDRC_MCFG1	SDRC configuration register 1	R/W

At reset, the following occur:

- CONTROL.CONTROL_SDRC_SHARING[30:0] copies into SDRC.SDRC_SHARING[30:0].
- CONTROL.CONTROL_SDRC_MCFG0[30:0] copies into SDRC.SDRC_MCFG_0[30:0].
- CONTROL.CONTROL_SDRC_MCFG1[30:0] copies into SDRC.SDRC_MCFG_1[30:0].

When LOCK bit SDRC.SDRC_SHARING[30] is set, a copy of SDRC.SDRC_SHARING[30:0] is made into CONTROL.CONTROL_SDRC_SHARING[30:0].

When LOCK bit SDRC.SDRC_MCFG_0[30] is set, a copy of SDRC.SDRC_MCFG_0[30:0] is made into CONTROL.CONTROL_SDRC_MCFG0[30:0].

When LOCK bit SDRC.SDRC_MCFG_1[30] is set, a copy of SDRC.SDRC_MCFG_1[30:0] is made into CONTROL.CONTROL_SDRC_MCFG1[30:0].

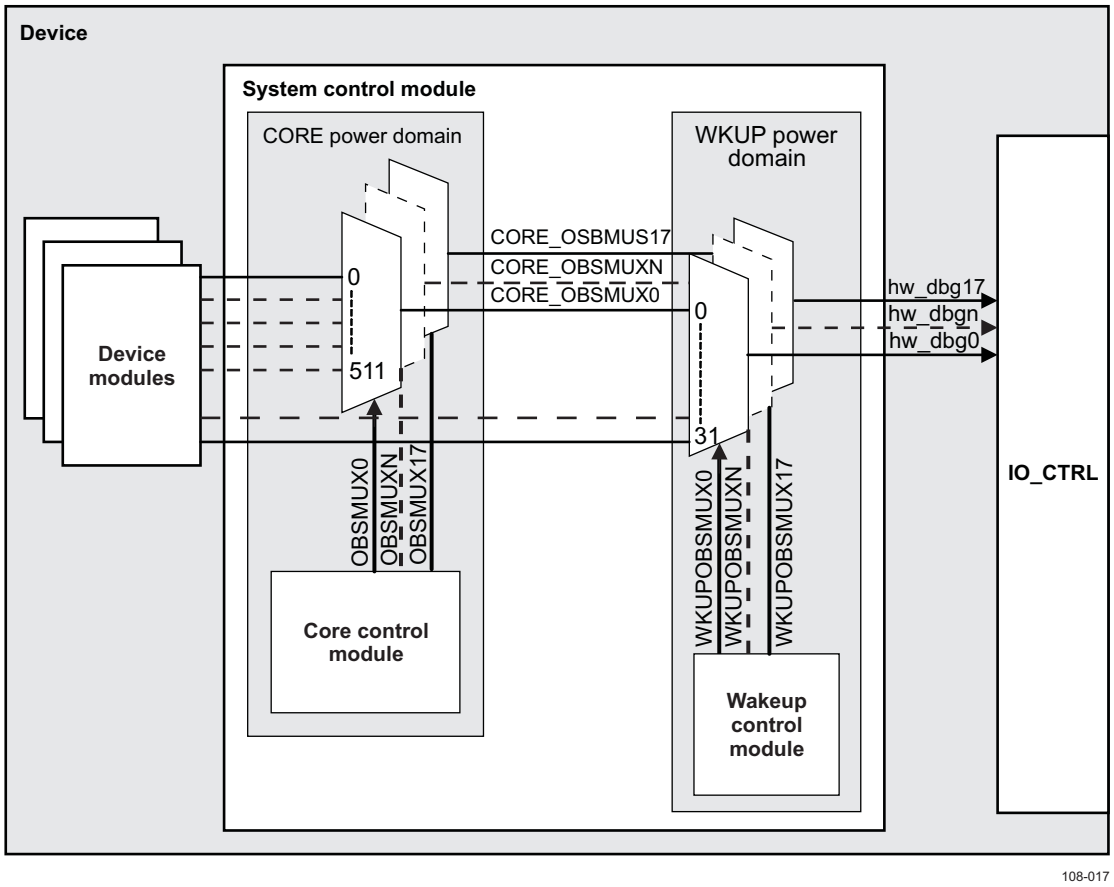
7.4.10 Debug and Observability

7.4.10.1 Description

Note: This feature is restricted to emulator and GP devices.

Figure 7-17 shows an overview of observability multiplexing, which minimizes the number of signals exchanged at the power domain boundary.

Figure 7-17. Overview of the Debug and Observability Register Functionality



Two layers of multiplexer are used to select the set of internal observable signals (PRCM signals, DMA requests, and interrupts) to be routed to the pins dedicated to the hardware debug.

The first layer is in the CORE power domain. It is controlled by the core control module registers and selects the set of internal signals from the CORE power domain to be routed. The second layer is in the WKUP power domain. It is controlled by the wake-up control module registers and selects the set of internal signals from the WKUP power domain.

The pads used for the hardware debug must be properly configured by selecting the hardware debug function (hw_dbgn) of the pad. To configure the pads, select mode 5 (0b101) in the MUXMODE bit field of the CONTROL.CONTROL_PADCONF_CAM_x register (only for hw_dbg0 to hw_dbg11), or select mode 7 (0b111) in the MUXMODE bit field of the CONTROL.CONTROL_PADCONF_ETK_x register (for all hw_dbgn). Before selecting the CORE signals, the WKUPOBSMUX bit field of the CONTROL.CONTROL_WKUP_DEBOBS_n registers must be set to 0.

Table 7-22. Observability Registers

Physical Address	Register Name	Description	Access	
			Device Type	
			E/T/G	S/B
0x4800 2420	CONTROL_DEBOBS_0	Set and configure CORE observable signals 1 and 0	R/W	R
0x4800 2424	CONTROL_DEBOBS_1	Set and configure CORE observable signals 3 and 2	R/W	R
0x4800 2428	CONTROL_DEBOBS_2	Set and configure CORE observable signals 5 and 4	R/W	R

Table 7-22. Observability Registers (continued)

Physical Address	Register Name	Description	Access	
			Device Type	
			E/T/G	S/B
0x4800 242C	CONTROL_DEBOBS_3	Set and configure CORE observable signals 7 and 6	R/W	R
0x4800 2430	CONTROL_DEBOBS_4	Set and configure CORE observable signals 9 and 8	R/W	R
0x4800 2434	CONTROL_DEBOBS_5	Set and configure CORE observable signals 11 and 10	R/W	R
0x4800 2438	CONTROL_DEBOBS_6	Set and configure CORE observable signals 13 and 12	R/W	R
0x4800 243C	CONTROL_DEBOBS_7	Set and configure CORE observable signals 15 and 14	R/W	R
0x4800 2440	CONTROL_DEBOBS_8	Set and configure CORE observable signals 17 and 16	R/W	R
0x4800 2A68	CONTROL_WKUP_DEBOBS_0	Set and configure WKUP observable pins 3, 2, 1, 0	R/W	R
0x4800 2A6C	CONTROL_WKUP_DEBOBS_1	Set and configure WKUP observable pins 7, 6, 5, 4	R/W	R
0x4800 2A70	CONTROL_WKUP_DEBOBS_2	Set and configure WKUP observable pins 11, 10, 9, 8	R/W	R
0x4800 2A74	CONTROL_WKUP_DEBOBS_3	Set and configure WKUP observable pins 15, 14, 13, 12	R/W	R
0x4800 2A78	CONTROL_WKUP_DEBOBS_4	Set and configure WKUP observable pins 17, 16	R/W	R

The write capabilities of these registers differ according to the device type.

Perform the following steps to configure observability:

1. To configure the pads properly for hardware debug and observability, select the hardware debug (hw_dbg) function mode 5 in the MUXMODE bit field of the CONTROL.CONTROL_PADCONF_CAM_x or mode 7 in the MUXMODE bit field of the CONTROL.CONTROL_PADCONF_ETK_X registers.
2. For the observability pads, set the proper values of the WKUPOBSMUX field CONTROL.CONTROL_WKUP_DEBOBS_n. Up to 5 bits are used to select the signal set to be observed (0x00 selection sets the output to CORE_OSMUXn signal). For more information, see the description of each register in Section 7.4.10.2, *Observability Tables*.
3. To observe the CORE_OSMUXn signals from the first layer of the multiplexer, set the WKUPOBSMUX field CONTROL.CONTROL_WKUP_DEBOBS_n to 0x00, and then set the proper values of the OBSMUX field CONTROL.CONTROL_DEBOBS_n. A maximum of 7 bits is used to select the signal set to be observed (0b0000000 selection sets the output to 0).

   For more information, see the description of each register in Section 7.4.10.2, *Observability Tables*.

   The observability feature of the CONTROL.CONTROL_WKUP_DEBOBS_n registers can be gated using the OBSERVABILITYDISABLE bit CONTROL.CONTROL_WKUP_DEBOBS_4[31]. When set, if the pads are configured for the hardware debug, outputs are set to 0.

7.4.10.2 Observability Tables

This section gives information about all modules and features in the high-tier device. See Section 1.5, *OMAP34xx Family*, to check availability of modules and features. Unavailable module and feature pins are not functional.

Table 7-23 through Table 7-58 define the mapped internal signals for each OBSMUX and WKUPOBSMUX value.

Table 7-23. Internal Signals Multiplexed on OBSMUX0

Out Signal Name	Muxed Signal Name	OBSMUX0 Field CONTROL.CONTROL_DEBOBS_0 [22:16] (dec)	Description	High State	Low State
CORE_OBSMUX0 (1)	tie_low	0	-	-	-
	CM_96_FCLK	1	96-MHz functional clock of the CM module	-	-
	CM_32K_CLK	2	32-kHz functional clock of the CM module	-	-
	PRCM_DPLL3_enable	3	Signal used to enable DPLL3.	DPLL is enabled.	DPLL is disabled.
	PRCM_CAM_domainFreeze	4	Indicates whether the CAM domain is frozen	Domain is frozen.	Domain is not frozen.
	PRCM_NEON_forceWakeup	5	Indicates whether a wakeup of the NEON domain is forced	Wakeup is forced.	Wakeup is not forced.
	PRCM_COREL4_domainNready	6	Indicates whether the CORE_L4 domain is ready. In other words, is domain transition ongoing?	Domain is not ready.	Domain is ready.
	PRCM_WKUP_domainNready	7	Indicates whether the WKUP domain is ready. In other words, is domain transition ongoing?	Domain is not ready.	Domain is ready.
	PRCM_STATE_IS_OFF_IVA2	8	Indicates to the global power manager FSM that the IVA2 domain power state is ON	FSM state is OFF.	FSM state is not OFF.
	Reserved	(10:9)	-	-	-
	SAD2D_MSTANDBY	11	SAD2D Mstandby assertion	-	-
	Reserved	12	-	-	-
	SAD2D_GICLK	13	Interface clock of the L4 interconnect in the SAD2D domain	-	-
	Reserved	(16:14)	-	-	-
	sdma_PI_DMAREQ	(87:17)	DMA requests lines mapped to the system DMA module. See Chapter 9, *DMA*, for more information about the system DMA request mapping.	-	-
	sgx_SINTERRUPTN	(104:88)	Interrupt lines from the SGX. See Chapter 13, *SGX*, for more information about the interrupt requests mapping.	-	-
	Reserved	(127:105)	-	-	-

(1) 0x00 in WKUPOBSMUX0 field CONTROL.CONTROL_WKUP_DEBOBS_0[4:0]

Table 7-24. Internal Signals Multiplexed on OBSMUX1

Out Signal Name	Muxed Signal Name	OBSMUX1 Field CONTROL.CONTROL_DEBOBS_0[6:0] (dec)	Description	High State	Low State
CORE_OBSMUX1 (1)	tie_low	0	-	-	-
	PRCM_DPLL3_M2_CLK	1	M2 clock generated by DPLL3	-	-
	CM_SYS_CLK	2	System clock	-	-
	PRCM_DPLL3_enablediv	3	Signal used to enable the clock divisor of DPLL3	DPLL clock divisor is enabled.	DPLL clock divisor is disabled.
	PRCM_DPLL2_ClkIsNot Running	4	Indicates whether the clock of DPLL2 is running or not	Clock is not running.	Clock is running.

(1) 0x00 in WKUPOBSMUX1 field CONTROL.CONTROL_WKUP_DEBOBS_0[12:8]

Table 7-24. Internal Signals Multiplexed on OBSMUX1 (continued)

Out Signal Name	Muxed Signal Name	OBSMUX1 Field CONTROL.CONTROL_DEBOBS_0[6:0] (dec)	Description	High State	Low State
	PRCM_NEON_domain Nready	5	Indicates whether the NEON domain is ready. In other words, is domain transition ongoing?	Domain is not ready.	Domain is ready.
	PRCM_CORED2D_domain Nready	6	Indicates if the CORE_D2D domain is ready. In other words, is domain transition ongoing?	Domain is not ready.	Domain is ready.
	PRCM_WKUP_domain Nready	7	Indicates whether the WKUP domain is ready. In other words, is domain transition ongoing?	Domain is not ready.	Domain is ready.
	PRCM_STATE_IS_ON_ CORE	8	Indicates to the global power manager FSM that the MPU domain power state is ON	FSM state is ON.	FSM state is not ON.
	Reserved	(10:9)	-	-	-
	SAD2D.WAIT	11	SAD2D wait signal	-	-
	Reserved	12	-	-	-
	PRCM_CORE_96M_GFCLK	13	96-MHz functional clock of the CORE domain	-	-
	Reserved	(16:14)	-	-	-
	sdma_PI_DMAREQ	(87:17)	DMA requests lines mapped to the system DMA module. SeeChapter 9, *DMA*, for more information about the system DMA request mapping.	-	-
	Reserved	(127:88)	-	-	-

Table 7-25. Internal Signals Multiplexed on OBSMUX2

Out Signal Name	Muxed Signal Name	OBSMUX2 Field CONTROL.CONTROL_DEBOBS_1[22:16] (dec)	Description	High State	Low State
CORE_OBSMUX2(1)	tie_low	0	-	-	-
	PRCM_DPLL3_M2X2_CLK	1	M2X2 clock generated by DPLL3	-	-
	PRCM_DPLL1_freqlock	2	Indicates whether the frequency of DPLL1 is locked	DPLL frequency is locked.	DPLL frequency is not locked.
	PRCM_DPLL4_freqlock	3	Indicates whether the frequency of DPLL4 is locked	DPLL frequency is locked.	DPLL frequency is not locked.
	PRCM_COREL3_IClkIsNot Running	4	Indicates whether the interface clock of the COREL3 domain is running or not.	Clock is not running.	Clock is running.
	PRCM_NEON_forceSleep	5	Indicates whether the NEON domain is forced to sleep mode	Sleep mode is forced.	Sleep mode is not forced.
	PRCM_CAM_domainIsIdle	6	Indicates whether CAM domain is idle	Domain is in Idle mode.	Domain is not in Idle mode.
	PRCM_EMU_domainIsIdle	7	Indicates whether EMU domain is idle	Domain is in Idle mode.	Domain is not in Idle mode.
	PRCM_STATE_IS_OFF_ CORE	8	Indicates to the global power manager FSM that the MPU domain power state is OFF	FSM state is OFF.	FSM state is not OFF.

(1) 0x00 in WKUPOBSMUX2 field CONTROL.CONTROL_WKUP_DEBOBS_0[20:16]

Table 7-25. Internal Signals Multiplexed on OBSMUX2 (continued)

Out Signal Name	Muxed Signal Name	OBSMUX2 Field CONTROL.CONTROL_DEBOBS_1[22:16] (dec)	Description	High State	Low State
	Reserved	(12:9)	-	-	-
	PRCM_CORE_48M_GFCLK	13	96-MHz functional clock of the CORE domain	-	-
	Reserved	(16:14)	-	-	-
	sdma_PI_DMAREQ	(87:17)	DMA requests lines mapped to the system DMA module. See Chapter 9, *DMA*, for more information about the system DMA request mapping.	-	-
	Reserved	(127:88)	-	-	-

Table 7-26. Internal Signals Multiplexed on OBSMUX3

Out Signal Name	Muxed Signal Name	OBSMUX3 Field CONTROL.CONTROL_DEBOBS_1[6:0] (dec)	Description	High State	Low State
CORE_OBSMUX3(1)	tie_low	0	-	-	-
	PRCM_L3_ICLK	1	Interface clock of the L3 interconnect	-	-
	PRCM_DPLL1_bypass	2	Indicates whether the DPLL1 is bypassed, in other words if the clock divisor is 1	DPLL is bypassed.	DPLL is not bypassed.
	PRCM_DPLL4_bypass	3	Indicates whether the DPLL4 is bypassed, in other words if the clock divisor is 1	DPLL is bypassed.	DPLL is not bypassed.
	PRCM_CORED2D__IclkIs NotRunning	4	Indicates whether the interface clock of the CORELD2D domain is running	The clock is not running.	The clock is running.
	PRCM_IVA2_domainIsIdle	5	Indicates whether the IVA2 domain is in idle	Domain is in idle mode.	Domain is not in Idle mode.
	PRCM_CAM_forceWakeup	6	Indicates whether a wakeup of the CAM domain is forced	Wakeup is forced.	Wakeup is not forced.
	PRCM_EMU_domainMute	7	Genarated by EMU domain to indicate that it does not require services from domain A (Domain A can go in INACTIVE state)	Domain is in standby mode.	Domain is not in standby mode.
	PRCM_SEQ_FORCE CLKON	8	Indicates whether the EMU is forcing the IVA2 clocks to ON	Clocks are forced.	Clocks are not forced.
	Reserved	(12:9)	-	-	-
	PRCM_CORE_12M_ GFCLK	13	12-MHz functional clock of the CORE domain	-	-
	Reserved	(16:14)	-	-	-
	sdma_PI_DMAREQ	(87:17)	DMA requests lines mapped to the system DMA module. See Chapter 9, *DMA*, for more information about the system DMA request mapping.	-	-

(1) 0x00 in WKUPOBSMUX3 field CONTROL.CONTROL_WKUP_DEBOBS_0[28:24]

Table 7-26. Internal Signals Multiplexed on OBSMUX3 (continued)

Out Signal Name	Muxed Signal Name	OBSMUX3 Field CONTROL.CONTROL_DEBOBS_1[6:0] (dec)	Description	High State	Low State
	iva_gl_dmarq_na	(107:88)	DMA requests lines used by the IVA2 subsystem. See Chapter 14, *IVA2 Subsystem*, for more information about these DMA request lines.	-	-
	Reserved	(127:108)	-	-	-

Table 7-27. Internal Signals Multiplexed on OBSMUX4

Out Signal Name	Muxed Signal Name	OBSMUX4 Field CONTROL.CONTROL_DEBOBS_2[22:16] (dec)	Description	High State	Low State
CORE_OBSMUX4 (1)	tie_low	0	-	-	-
	PRCM_L4_ICLK	1	Interface clock of the L4 interconnect	-	-
	PRCM_DPLL1_idle	2	Indicates whether DPLL1 is idle	Domain is in idle mode.	Domain is not in Idle mode.
	PRCM_DPLL4_idle	3	Indicates whether DPLL4 is idle	Domain is in idle mode.	Domain is not in Idle mode.
	Reserved	4	-	-	-
	PRCM_IVA2_force Wakeup	5	Indicates whether a wakeup of the IVA2 domain is forced	Wakeup is forced.	Wakeup is not forced.
	PRCM_CAM_domain Nready	6	Indicates whether the CAM domain is ready. In other words, is domain transition ongoing?	Domain is not ready.	Domain is ready.
	PRCM_EMU_force Wakeup	7	Indicates whether a wakeup of the EMU domain is forced	Wakeup is forced.	Wakeup is not forced.
	SEQ_FORCECLKON ACK	8	Indicates whether the forcing to ON of the clocks of the IVA2 domain is acknowledged	Command acknowledged	Command not acknowledged
	Reserved	(12:9)	-	-	-
	PRCM_CORE_L3_ GICLK	13	Interface clock of the L3 interconnect	-	-
	Reserved	(16:14)	-	-	-
	mpu_PIIRQ	(112:17)	Interrupt request lines mapped to the interrupt controller. See Chapter 10, *Interrupt Controller*, for more information about these interrupt lines.	-	-
	Reserved	(127:113)	-	-	-

(1) 0x00 in WKUPOBSMUX4 field CONTROL.CONTROL_WKUP_DEBOBS_1[4:0]

Table 7-28. Internal Signals Multiplexed on OBSMUX5

Out Signal Name	Muxed Signal Name	OBSMUX5 Field CONTROL.CONTROL_DEBOBS_2[6:0] (dec)	Description	High State	Low State
CORE_OBSMUX5 (1)	tie_low	0	-	-	-
	PRCM_RM_ICLK	1	Interface clock of the RM block	-	-

(1) 0x00 in WKUPOBSMUX5 field CONTROL.CONTROL_WKUP_DEBOBS_1[12:8]

Table 7-28. Internal Signals Multiplexed on OBSMUX5 (continued)

Out Signal Name	Muxed Signal Name	OBSMUX5 Field CONTROL.CONTROL_DEBOBS_2[6:0] (dec)	Description	High State	Low State
	PRCM_DPLL1_initz	2	Lock sequence initialization (HLH) of DPLL1	-	-
	PRCM_DPLL4_initz	3	Lock sequence initialization (HLH) of DPLL4	-	-
	CM_SysClkIsRunning	4	Indicates whether the system clock is running or not	The clock is running.	The clock is not running.
	PRCM_IVA2_domain Nready	5	Indicates whether the IVA2 domain is ready. In other words, is domain transition ongoing?	Domain is not ready.	Domain is ready.
	PRCM_CAM_forceSleep	6	Indicates whether the CAM domain is forced to sleep mode	Sleep mode is forced.	Sleep mode is not forced.
	PRCM_EMU_domain Nready	7	Indicates whether the EMU domain is ready. In other words, is domain transition ongoing?	Domain is not ready.	Domain is ready.
	Reserved	(12:8)	-	-	-
	PRCM_CORE_L4_ GICLK	13	Interface clock of the L4 interconnect	-	-
	Reserved	(16:14)	-	-	-
	mpu_PIIRQ	(112:17)	Interrupt request lines mapped to the interrupt controller. See Chapter 10, *Interrupt Controller*, for more information about these interrupt lines.	-	-
	Reserved	(127:113)	-	-	-

Table 7-29. Internal Signals Multiplexed on OBSMUX6

Out Signal Name	Muxed Signal Name	OBSMUX6 Field CONTROL.CONTROL_DEBOBS_3[22:16] (dec)	Description	High State	Low State
CORE_OBSMUX6(1)	tie_low	0	-	-	-
	PRCM_FUNC_96M_FCLK	1	96 MHz functional clock	-	-
	PRCM_DPLL1_enable	2	Signal used to enable DPLL1	DPLL is enabled.	DPLL is disabled.
	PRCM_DPLL4_enable	3	Signal used to enable DPLL4	DPLL is enabled.	DPLL is disabled.
	PRCM_WKUP_IClkIs Running	4	Indicates whether the interface clock of the WKUP domain is running or not	The clock is running.	The clock is not running.
	PRCM_IVA2_forceSleep	5	Indicates whether the IVA2 domain is forced to sleep mode	Sleep mode is forced.	Sleep mode is not forced.
	PRCM_CAM_domainFreeze	6	Indicates whether the CAM domain is frozen	Domain is frozen.	Domain is not frozen.
	PRCM_USBHOST_domain IsIdle	7	Indicates whether the USBHOST domain is idle	Domain is in idle mode.	Domain is not in idle mode.
	Reserved	(11:8)	-	-	-
	PRCM_DSS_GICLK	12	Interface clock of the DSS module	-	-
	Reserved for non-GP devices	13	Reserved for non-GP devices	-	-

(1) 0x00 in WKUPOBSMUX6 field CONTROL.CONTROL_WKUP_DEBOBS_1[20:16]

Table 7-29. Internal Signals Multiplexed on OBSMUX6 (continued)

Out Signal Name	Muxed Signal Name	OBSMUX6 Field CONTROL.CONTROL_DEBOBS_3[22:16] (dec)	Description	High State	Low State
	Reserved	(16:14)	-	-	-
	mpu_PIIRQ	(112:17)	Interrupt request lines mapped to the interrupt controller. See Chapter 10, *Interrupt Controller*, for more information about these interrupt lines.	-	-
	Reserved	(114:113)	-	-	-
	PRCM_DPLL1_LOSSREF	115	Reference input loss acknowledge of DPLL1	Signal acknowledged	Signal not acknowledged
	PRCM_DPLL2_LOSSREF	116	Reference input loss acknowledge of DPLL2	Signal acknowledged	Signal not acknowledged
	Reserved	(123:117)	-	-	-
	PRCM_DPLL3_LOSSREF	124	Reference input loss acknowledge of DPLL3	Signal acknowledged	Signal not acknowledged
	PRCM_DPLL4_LOSSREF	125	Reference input loss acknowledge of DPLL4	Signal acknowledged	Signal not acknowledged
	Reserved	(127:126)	-	-	-

Table 7-30. Internal Signals Multiplexed on OBSMUX7

Out Signal Name	Muxed Signal Name	OBSMUX7 Field CONTROL.CONTROL_DEBOBS_3[6:0] (dec)	Description	High State	Low State
CORE_OBSMUX7 (1)	tie_low	0	-	-	-
	PRCM_FUNC_48M_FCLK	1	48 MHz functional clock	-	-
	PRCM_DPLL1_enablediv	2	Signal used to enable the clock divisor of DPLL1	DPLL frequency divisor is enabled.	DPLL frequency divisor is disabled.
	PRCM_DPLL4_enablediv	3	Signal used to enable the clock divisor of DPLL4	DPLL frequency divisor is enabled.	DPLL frequency divisor is disabled.
	This information is not available in public domain.	4	This information is not available in public domain.		
	PRCM_IVA2_domainFreeze	5	Indicates whether the IVA2 domain is frozen	Domain is frozen.	Domain is not frozen.
	PRCM_DSS_domainIsIdle	6	Indicates whether the DSS domain is in idle	Domain is in idle mode.	Domain is not in idle mode.
	PRCM_USBHOST_forceWakeup	7	Indicates whether a wakeup of the USBHOST domain is forced	Wakeup is forced.	Wakeup is not forced.
	Reserved	(11:8)	-	-	-
	PRCM_CAM_GICLK	12	Interface clock of the CAM module	-	-
	Reserved for non-GP devices	13	Reserved for non-GP devices	-	-
	Reserved	(16:14)	-	-	-
	mpu_PIIRQ	(112:17)	Interrupt request lines mapped to the interrupt controller. See Chapter 10, *Interrupt Controller*, for more information about these interrupt lines.	-	-
	Reserved	(114:113)	-	-	-
	PRCM_DPLL1_BREAKLOCKZ	115	Indicates whether DPLL1 is in frequency lock condition	Lock conditions not reached	Lock conditions reached
	Reserved	(121:116)	-	-	-
	PRCM_DPLL2_BREAKLOCKZ	122	Indicates whether DPLL2 is in frequency lock condition	Lock conditions not reached	Lock conditions reached
	PRCM_DPLL3_BREAKLOCKZ	123	Indicates whether DPLL3 is in frequency lock condition	Lock conditions not reached	Lock conditions reached
	PRCM_DPLL4_BREAKLOCKZ	124	Indicates whether DPLL4 is in frequency lock condition	Lock conditions not reached	Lock conditions reached
	Reserved	(127:125)	-	-	-

(1) 0x00 in WKUPOBSMUX7 field CONTROL.CONTROL_WKUP_DEBOBS_1[28:24]

Table 7-31. Internal Signals Multiplexed on OBSMUX8

Out Signal Name	Muxed Signal Name	OBSMUX8 Field CONTROL.CONTROL_DEBOBS_4[22:16] (dec)	Description	High State	Low State
CORE_OBSMUX8 (1)	tie_low	0	-	-	-
	PRCM_DSS_TV_FCLK	1	Functional clock of the DSS module for TV output	-	-
	PRCM_DPLL2_freqlock	2	Indicates whether the frequency of DPLL2 is locked	DPLL frequency is locked.	DPLL frequency is not locked.
	PRCM_DPLL5_freqlock	3	Indicates whether the frequency of DPLL5 is locked	DPLL frequency is locked.	DPLL frequency is not locked.
	PRCM_GFX_IClkIsNotRunning	4	Indicates whether the interface clock of the 2D/3D graphics accelerator is running	Clock is not running.	Clock is running.
	PRCM_SGX_domainIsIdle	5	Indicates whether SGX domain is idle	Domain is in idle mode.	Domain is not in idle mode.
	PRCM_DSS_forceWakeup	6	Indicates whether a wakeup of the DSS domain is forced	Wakeup is forced.	Wakeup is not forced.
	PRCM_USBHOST_domainNready	7	Indicates whether the DSS domain is ready. In other words, is domain transition ongoing?	Domain is not ready.	Domain is ready
	Reserved	(11:8)	-	-	-
	PRCM_CSIb_96M_GFCLK	12	96-MHz functional clock of the CSIb module	-	-
	Reserved	(18:13)	-	-	-
	PRCM_DPLL1_PHASELOCK	19	Indicates whether DPLL1 is in phase lock condition	Lock conditions reached	Lock conditions not reached
	Reserved	(25:20)	-	-	-
	PRCM_DPLL2_PHASELOCK	26	Indicates whether DPLL2 is in phase lock condition	Lock conditions reached	Lock conditions not reached
	PRCM_DPLL3_PHASELOCK	27	Indicates whether DPLL3 is in phase lock condition	Lock conditions reached	Lock conditions not reached
	PRCM_DPLL4_PHASELOCK	28	Indicates whether DPLL4 is in phase lock condition	Lock conditions reached	Lock conditions not reached
	Reserved	(127:29)	-	-	-

(1) 0x00 in WKUPOBSMUX8 field CONTROL.CONTROL_WKUP_DEBOBS_2[4:0]

Table 7-32. Internal Signals Multiplexed on OBSMUX9

Out Signal Name	Muxed Signal Name	OBSMUX9 Field CONTROL.CONTROL_DEBOBS_4[8:0] (dec)	Description	High State	Low State
CORE_OBSMUX9 (1)	tie_low	0	-	-	-
	Reserved	1	-	-	-
	PRCM_DPLL2_bypass	2	Indicates whether the DPLL2 is bypassed, in other words if the clock divisor is 1	DPLL is bypassed.	DPLL is not bypassed.
	PRCM_DPLL5_bypass	3	Indicates whether the DPLL5 is bypassed, in other words if the clock divisor is 1	DPLL is bypassed.	DPLL is not bypassed.
	PRCM_DSS_IClkIsNotRunning	4	Indicates whether the DSS interface clock is running or not	Clock is not running.	Clock is running.
	PRCM_SGX_forceWakeup	5	Indicates whether a wakeup of the SGX domain is forced	Wakeup is forced.	Wakeup is not forced.
	PRCM_DSS_domainNready	6	Indicates whether the DSS domain is ready. In other words, is domain transition ongoing ?	Domain is not ready.	Domain is ready.
	PRCM_USBHOST_forceSleep	7	Indicates whether the USBHOST domain forced to sleep mode	Sleep mode is forced.	Sleep mode is not forced.
	Reserved	(11:8)	-	-	-
	PRCM_PER_48M_GFCLK	12	48-MHz functional clock of the PER domain	-	-
	PRCM_DSS_96M_GFCLK	13	96-MHz functional clock of the DSS domain	-	-
	Reserved	(22:14)	-	-	-
	iva_gl_dmarq_na	(42:23)	DMA requests lines used by the IVA2 subsystem. See Chapter 14, *IVA2 Subsystem*, for more information about these DMA request lines.	-	-
	PRCM_DPLL1_RECAL	43	Indicates that DPLL1 needs to perform internal recalibration	Recalibration required	Recalibration not required
	PRCM_DPLL2_RECAL	44	Indicates that DPLL2 needs to perform internal recalibration	Recalibration required	Recalibration not required
	PRCM_DPLL3_RECAL	45	Indicates that DPLL3 needs to perform internal recalibration	Recalibration required	Recalibration not required
	PRCM_DPLL4_RECAL	46	Indicates that DPLL4 needs to perform internal recalibration	Recalibration required	Recalibration not required
	Reserved	(127:47)	-	-	-

(1) 0x00 in WKUPOBSMUX9 field CONTROL.CONTROL_WKUP_DEBOBS_2[12:8]

Table 7-33. Internal Signals Multiplexed on OBSMUX10

Out Signal Name	Muxed Signal Name	OBSMUX10 Field CONTROL.CONTROL_DEBOBS_5[22:16] (dec)	Description	High State	Low State
CORE_OBSMUX10 (1)	tie_low	0	-	-	-
	PRCM_CPEFUSE_FCLK	1	Functional clock of the EFUSE module	-	-
	PRCM_DPLL2_idle	2	Indicates whether the DPLL2 is idle	Domain is in idle mode.	Domain is not in idle mode.
	PRCM_DPLL5_idle	3	Indicates whether the DPLL5 is idle	Domain is in idle mode.	Domain is not in idle mode.
	PRCM_CAM_IClkIsNotRunning	4	Indicates is the CAM interface clock is running or not	Clock is not running.	Clock is running.
	PRCM_SGX_domainNready	5	Indicates whether the MPU domain is ready. In other words, is domain transition ongoing ?	Domain is not ready.	Domain is ready.
	PRCM_DSS_forceSleep	6	Indicates whether the DSS domain is forced to sleep mode	Sleep mode is forced.	Sleep mode is not forced.
	PRCM_USBHOST_domainFreeze	7	Indicates whether the USBHOST domain is frozen	Domain is frozen.	Domain is not frozen.
	Reserved	(10:8)	-	-	-
	This information is not available in public domain.	11	This information is not available in public domain.		
	PRCM_PER_96M_GFCLK	12	96-MHz functional clock of the PER domain	-	-
	PRCM_MPU_EMU_CLK	13	Functional clock of the MPU module in the EMU domain	-	-
	Reserved	(25:14)	-	-	-
	PRCM_IVA_FCLK	26	Functional clock of the IVA2 module.	-	-
	Reserved	(30:27)	-	-	-
	PRCM_DPLL1_BYPASS	31	Indicates whether DPLL1 is bypassed, in other words if the clock divisor is 1	DPLL is bypassed.	DPLL is not bypassed.
	PRCM_DPLL2_BYPASS	32	Indicates whether DPLL2 is bypassed, in other words if the clock divisor is 1	DPLL is bypassed.	DPLL is not bypassed.
	PRCM_DPLL3_BYPASS	33	Indicates whether DPLL3 is bypassed, in other words if the clock divisor is 1	DPLL is bypassed.	DPLL is not bypassed.
	PRCM_DPLL4_BYPASS	34	Indicates whether DPLL4 is bypassed, in other words if the clock divisor is 1	DPLL is bypassed.	DPLL is not bypassed.
	Reserved	(127:35)	-	-	-

(1) 0x00 in WKUPOBSMUX10 field CONTROL.CONTROL_WKUP_DEBOBS_2[20:16]

Table 7-34. Internal Signals Multiplexed on OBSMUX11

Out Signal Name	Muxed Signal Name	OBSMUX11 Field CONTROL.CONTROL_DEBOBS_5[6:0] (dec)	Description	High State	Low State
CORE_OBSMUX11(1)	tie_low	0	-	-	-
	PRCM_DPLL5_M2_CLK	1	M2 clock generated by DPLL5	-	-
	PRCM_DPLL2_initz	2	Lock sequence initialization (HLH) of DPLL2	-	-
	PRCM_DPLL5_initz	3	Lock sequence initialization (HLH) of DPLL5	-	-
	PRCM_PERL4_IClkIsNotRunning	4	Indicates whether the interface clock of the L4 interconnect in the PER domain is running or not	Clock is not running.	Clock is running.
	PRCM_SGX_forceSleep	5	Indicates whether the SGX domain is forced to sleep mode	Sleep mode is forced.	Sleep mode is not forced.
	PRCM_DSS_domainFreeze	6	Indicates whether the DSS domain is frozen	Domain is frozen.	Domain is not frozen.
	PRCM_emudpll3srcclkactivitystatus	7	Indicates (when enabled and required) if DPLL3 is forced in lock mode in order to ensure high speed emulation or trace clocks	Lock mode forced	Lock mode not forced
	Reserved	(10:8)	-	-	-
	This information is not available in public domain.	11	This information is not available in public domain.		
	PRCM_PER_L4_GICLK	12	Interface clock of the L4 interconnect in the PER clock domain	-	-
	Reserved	13	-	-	-
	Reserved	(29:14)	-	-	-
	PRCM_DPLL1_TICOPWDN	30	Indicates a Core DCO power down condition for DPLL1	Conditions reached	Conditions not reached
	PRCM_DPLL2_TICOPWDN	31	Indicates a Core DCO power down condition for DPLL2	Conditions reached	Conditions not reached
	PRCM_DPLL3_TICOPWDN	32	Indicates a Core DCO power down condition for DPLL3	Conditions reached	Conditions not reached
	PRCM_DPLL4_TICOPWDN	33	Indicates a Core DCO power down condition for DPLL4	Conditions reached	Conditions not reached
	Reserved	(127:34)	-	-	-

(1) 0x00 in WKUPOBSMUX11 field CONTROL.CONTROL_WKUP_DEBOBS_2[28:24]

Table 7-35. Internal Signals Multiplexed on OBSMUX12

Out Signal Name	Muxed Signal Name	OBSMUX12 Field CONTROL.CONTROL_DEBOBS_6[22:16] (dec)	Description	High State	Low State
CORE_OBSMUX12(1)	tie_low	0	-	-	-
	PRCM_DPLL1_FCLK	1	Functional clock of DPLL1	-	-
	PRCM_DPLL2_enable	2	Signal used to enable DPLL2	DPLL is enabled.	DPLL is disabled.
	PRCM_DPLL5_enable	3	Signal used to enable DPLL5	DPLL is enabled.	DPLL is disabled.
	PRCM_EMU_ClkIsRunning	4	Indicates whether the EMU clock is running or not	The clock is running.	The clock is not running.
	PRCM_COREL3_domainIsIdle	5	Indicates whether the COREL3 domain is in idle	Domain is in idle mode.	Domain is not in Idle mode.
	PRCM_PER_domainIsIdle	6	Indicates whether the PER domain is in idle	Domain is in idle mode.	Domain is not in Idle mode.
	PRCM_IVA2_DPLLCHANGE	7	Indicates whether the IVA2 DPLL needs to be recalibrated	Recalibration required	Recalibration not required
	Reserved	(11:8)	-	-	-
	This information is not available in public domain.	12	This information is not available in public domain.		
	PRCM_DPLL2_M2X2_CLK	13	M2X2 clock generated by DPLL2	-	-
	Reserved	(127:14)	-	-	-

(1) 0x00 in WKUPOBSMUX12 field CONTROL.CONTROL_WKUP_DEBOBS_3[4:0]

Table 7-36. Internal Signals Multiplexed on OBSMUX13

Out Signal Name	Muxed Signal Name	OBSMUX13 Field CONTROL.CONTROL_DEBOBS_6 [6:0] (dec)	Description	High State	Low State
CORE_OBSMUX13(1)	tie_low	0	-	-	-
	PRCM_DPLL2_FCLK	1	Functional clock of DPLL2.	-	-
	PRCM_DPLL2_enablediv	2	Signal used to enable the clock divisor of DPLL2.	DPLL colck divisor is enabled	DPLL clock divisor is disabled
	PRCM_DPLL5_enablediv	3	Signal used to enable the clock divisor of DPLL5.	DPLL colck divisor is enabled	DPLL clock divisor is disabled
	PRCM_CPEFUSE_FClkIsNotRunning	4	Indicates whether the CPEFUSE functional clock is running or not.	Clock is not running	Clock is running
	PRCM_COREL4_domainIsIdle	5	Indicates whether the COREL4 domain is in Idle.	Domain is in Idle mode	Domain is not in Idle mode
	PRCM_PER_forceWakeup	6	Indicates whether a wakeup of the PER domain is forced.	Wakeup is forced	Wakeup is not forced
	PRCM_IVA2_DPLLCHANGEACK	7	Indicates whether the IVA2 DPLL re-calibration is acknowledged.	re-calibration required	re-calibration not required
	Reserved	(11:8)	-	-	-
	PRCM_USBHOST_GICLK	12	Interface clock of the L4 interconnect in the USBHOST module.	-	-
	PRCM_EMU_CORE_ALWON_CLK	13	Emulation clock used by PRCM in order to run the emulation trace. Supplied by DPLL3.	-	-
	Reserved	14	-	-	-
	Reserved	15	-	-	-

(1) 0x00 in WKUPOBSMUX13 field CONTROL.CONTROL_WKUP_DEBOBS_3[12:8]

Table 7-36. Internal Signals Multiplexed on OBSMUX13 (continued)

Out Signal Name	Muxed Signal Name	OBSMUX13 Field CONTROL.CONTROL_DEBOBS_6 [6:0] (dec)	Description	High State	Low State
	Reserved	(127:16)	-	-	-

Table 7-37. Internal Signals Multiplexed on OBSMUX14

Out Signal Name	Muxed Signal Name	OBSMUX14 Field CONTROL.CONTROL_DEBOBS_7 [22:16] (dec)	Description	High State	Low State
CORE_OBSMUX14(1)	tie_low	0	-	-	-
	Reserved	1	-	-	-
	PRCM_DPLL3_freqlock	2	Indicates whether the frequency of DPLL3 is locked	DPLL frequency is locked.	DPLL frequency is not locked.
	PRCM_MPU_domainFreeze	3	Indicates whether the MPU domain is frozen	Domain is frozen.	Domain is not frozen.
	PRCM_MPU_domainIsIdle	4	Indicates whether MPU domain is idle	Domain is in idle mode.	Domain is not in idle mode.
	PRCM_CORED2D_domainIsIdle	5	Indicates if the CORED2D domain is in idle	Domain is in idle mode.	Domain is not in idle mode.
	PRCM_PER_domainNready	6	Indicates whether the PER domain is ready. In other words, is domain transition ongoing?	Domain is not ready.	Domain is ready.
	PRCM_EMU_MStandby	7	EMU asserts this signal to initiate a transition to standby mode	Transition initiated	Transition not initiated
	Reserved	(11:8)	-	-	-
	PRCM_USBHOST_48M_GFCLK	12	48-MHz functional clock of the USBHOST module	-	-
	PRCM_EMU_PER_ALWON_CLK	13	High-frequency always-on clock in the PER domain used to generate for emulation clocks	-	-
	Reserved	14	-	-	-
	PRCM_IVA_CLK	15	Functional clock of the IVA2	-	-
	Reserved	(127:16)	-	-	-

(1) 0x00 in WKUPOBSMUX14 field CONTROL.CONTROL_WKUP_DEBOBS_3[20:16]

Table 7-38. Internal Signals Multiplexed on OBSMUX15

Out Signal Name	Muxed Signal Name	OBSMUX15 Field CONTROL.CONTROL_DEBOBS_7 [6:0] (dec)	Description	High State	Low State
CORE_OBSMUX15(1)	tie_low	0	-	-	-
	PRCM_GPT10_FCLK	1	Functional clock of GPTIMER10.	-	-
	PRCM_DPLL3_bypass	2	Indicates whether DPLL3 is bypassed, in other words if the clock divisor is 1.	DPLL is bypassed	DPLL is not bypassed
	PRCM_IVA2_domainFreeze	3	Indicates whether the IVA2 domain is frozen.	Domain is frozen	Domain is not frozen
	PRCM_MPU_domainNready	4	Indicates whether the MPU domain is ready. In other words, is domain transition ongoing ?	Domain is not ready	Domain is ready
	PRCM_CORECM_domainIsIdle	5	Indicates whether CORECM domain is in idle.	Domain is in Idle mode	Domain is not in Idle mode
	PRCM_PER_forceSleep	6	Indicates whether the PER domain is forced to sleep mode.	Sleep mode is forced	Sleep mode is not forced
	PRCM_STATE_IS_ON_MPU	7	Indicates to the global Power Manager FSM that the MPU domain power state is ON.	FSM state is ON	FSM state is not ON
	Reserved	(11:8)	-	-	-
	PRCM_USBHOST_120M_GFCLK	12	96 MHz functional clock of the USBHOST module.	-	-
	PRCM_DPLL4_M2X2_CLK	13	M2X2 clock generated by DPLL4.	-	-
	Reserved	(16:14)	-	-	-
	This information is not available in public domain.	17	This information is not available in public domain.		
	Reserved	18	-	-	-
	iva_gl_irq_na	(66:19)	External interrupts requests generated by peripherals external to the IVA2 subsystem (such as SPI, display subsystem, or camera subsystem). See Chapter 14, *IVA2 Subsystem*, for more information about these interrupt request lines.	-	-
	Reserved	(127:67)	-	-	-

(1) 0x00 in WKUPOBSMUX15 field CONTROL.CONTROL_WKUP_DEBOBS_3[28:24]

Table 7-39. Internal Signals Multiplexed on OBSMUX16

Out Signal Name	Muxed Signal Name	OBSMUX16 Field CONTROL.CONTROL_DEBOBS_8[22:16] (dec)	Description	High State	Low State
CORE_OBSMUX16 (1)	tie_low	0	-	-	-
	PRCM_GPT11_FCLK	1	Functional clock of GPTIMER11.	-	-
	PRCM_DPLL3_idle	2	Indicates whether the DPLL3 is idle.	Domain is in Idle mode	Domain is not in Idle mode
	PRCM_CORED2D_domainFreeze	3	Indicates whether the CORELD2D domain is ready (is domain transition ongoing?)	Domain is ready.	Domain is not ready.
	PRCM_MPU_domainFreeze	4	Indicates whether the MPU domain is frozen.	Domain is frozen	Domain is not frozen
	PRCM_CORE_domainNready	5	Indicates whether the CORE domain is ready. In other words, is domain transition ongoing ?	Domain is not ready	Domain is ready
	PRCM_WKUP_domainIsIdle	6	Indicates whether WKUP domain is Idle.	Domain is in Idle mode	Domain is not in Idle mode
	PRCM_STATE_IS_OFF_MPU	7	Indicates to the global Power Manager FSM that the MPU domain power state is OFF.	FSM state is OFF	FSM state is not OFF
	Reserved	(11:8)	-	-	-
	PRCM_SGX_GICLK	12	Interface clock of the L4 interconnect in the SGX clock module.	-	-
	PRCM_DPLL4_M4X2_CLK	13	M4X2 clock generated by DPLL4.	-	-
	Reserved	(16:14)	-	-	-
	This information is not available in public domain.	17	This information is not available in public domain.		
	Reserved	18	-	-	-
	iva_gl_irq_na	(66:19)	External interrupts requests generated by peripherals external to the IVA2 subsystem (such as SPI, display subsystem, or camera subsystem). See Chapter 14, *IVA2 Subsystem*, for more information about these interrupt request lines.	-	-
	Reserved	(127:67)	-	-	-

(1) 0x00 in WKUPOBSMUX16 field CONTROL.CONTROL_WKUP_DEBOBS_4[4:0]

Table 7-40. Internal Signals Multiplexed on OBSMUX17

Out Signal Name	Muxed Signal Name	OBSMUX17 Field CONTROL.CONTROL_DEBOBS_8 [6:0] (dec)	Description	High State	Low State
CORE_OBSMUX17(1)	tie_low	0	-	-	-
	PRCM_SGX_GFCLK	1	Functional clock of the L4 interconnect in the SGX clock module.	-	-
	PRCM_DPLL3_initz	2	Lock sequence initialization (HLH) of DPLL3	-	-
	PRCM_DSS_domainFreeze	3	Indicates whether the DSS domain is frozen.	Domain is frozen	Domain is not frozen
	PRCM_NEON_domainIsIdle	4	Indicates whether the NEON domain is Idle.	Domain is in Idle mode	Domain is not in Idle mode
	PRCM_COREL3_domainNready	5	Indicates whether the COREL3 domain is ready. In other words, is domain transition ongoing?	Domain is not ready	Domain is ready
	PRCM_WKUP_domainWakeupAck	6	Indicates whether a wakeup of the WKUP domain is acknowledged.	Command acknowledged	Command not acknowledged
	PRCM_STATE_IS_ON_IVA2	7	Indicates to the global Power Manager FSM that the IVA2 domain power state is ON.	FSM state is ON	FSM state is not ON
	Reserved	(12:8)	-	-	-
	PRCM_DPLL4_M5X2_CLK	13	M5X2 clock generated by DPLL4.	-	-
	Reserved	(127:14)	-	-	-

(1) 0x00 in WKUPOBSMUX17 field CONTROL.CONTROL_WKUP_DEBOBS_4[12:8]

Table 7-41. Internal Signals Multiplexed on WKUPOBSMUX0

Pin Name	Observed Signal Name	WKUPOBSMUX0 Field CONTROL.CONTROL_WKUP_DEBOBS_0[4:0] (dec)	Description	High State	Low State
hw_dbg0(1)	CORE_OBSMUX0	0	Signal multiplexed by OBSMUX0.	-	-
	PRCM_SYS_CLK	1	System clock running at the system clock frequency.	-	-
	PRCM_GPT5_ALWON_FCLK	2	Always-on functional clock of GP-Timer #5.	-	-
	PRCM_SGX_RST	3	Reset signal for SGX.	Reset not active	Reset active
	PRCM_USBHOST_RST	4	Reset signal for USBHOST.	Reset not active	Reset active
	Reserved	5	-	-	-
	PRCM_CORECM_domainIsOn	6	Indicates to the global Power Manager FSM that the CORECM domain power state is ON.	State is ON	State is not ON
	PRCM_WKUP_domainWakeup	7	Indicates that a wake-up condition is detected for the WKUP domain.	Conditions reached	Conditions not reached
	PRCM_vdd2_domain_is_ret	8	Indicates to the global Power Manager FSM that the VDD2 domain power state is RETENTION.	State is RETENTION	State is not RETENTION
	PRCM_vdd5_domain_is_on	9	Indicates to the global Power Manager FSM that the VDD5 domain power state is ON.	State is ON	State is not ON
	PRCM_device_is_on	10	Indicates whether the device is ON.	State is ON	State is not ON
	PRCM_MPU_SRAMAONIN[1]	11	SRAM array power control input (bit 1) for MPU power domain.	Array is powered	Array is not powered
	PRCM_IVA2_SRAMAONIN[3]	12	SRAM array power control input (bit 3) for IVA2 power domain.	Array is powered	Array is not powered
	PRCM_CORE_POWER_ISO	13	PM command for CORE domain isolation.	Isolation is ON	Isolation is OFF
	PRCM_CORE_SRAMRETONIN[0]	14	SRAM retention on signal to CORE power domain (bit 0).	Retention is ON	Retention is OFF
	PRCM_SGX_POWER_GOOD[0]	15	Indicates whether the SGX power switches status (bit 0).	Power is stable	Power is not stable
	PRCM_CAM_POWER_ON	16	CAM switch command for power-on.	Power is ON	Power is OFF
	This information is not available in public domain.	17	This information is not available in public domain.		
	PRCM_PER_POWER_GOOD[0]	18	Indicates whether the PER power switches status (bit 0).	Power is stable	Power is not stable
	PRCM_forceemucm	19	Indicates whether the EMULATION mode of the clock manager is forced.	Forced	Not forced
	PRCM_MPU_INTC_SWAKEUP	20	MPU Interrupt Controller asserts this signal to initiate a transition to WAKEUP mode.	Transition initiated	Wakeup conditions not reached
	Reserved	(31:21)	-	-	-

(1) Mode 5 in MUXMODE0 field CONTROL.CONTROL_PADCONF_CAM_HS[2:0]

Table 7-42. Internal Signals Multiplexed on WKUPOBSMUX1

Pin Name	Observed Signal Name	WKUPOBSMUX1 Field CONTROL.CONTROL_WKUP_DEBOBS_0[12:8] (dec)	Description	High State	Low State
hw_dbg1[(1)]	CORE_OBSMUX1	0	Signal multiplexed by OBSMUX1.	-	-
	PRCM_DPLL1_ALWON_FCLK	1	Always-on functional clock of DPLL1.	-	-
	PRCM_GPT6_ALWON_FCLK	2	Always-on functional clock of GP-Timer #6.	-	-
	PRCM_DSS_RST	3	Reset signal for DSS.	Reset not active	Reset active
	PRCM_ICEPICK_RSTPWRON	4	Cold reset signal for ICEPICK.	Reset not active	Reset active
	PRCM_eFuseReady_n	5	Indicates whether the device eFuse scan is completed.	Scan complete	Scan not complete
	PRCM_CAM_domainWakeup	6	Indicates that a wake-up condition is detected for the CAM domain.	Conditions reached	Conditions not reached
	PRCM_cam_domain_is_inactive	7	Indicates whether the CAM domain is active or not.	Domain is inactive	Domain is active
	PRCM_cam_domain_is_retention	8	Indicates to the global Power Manager FSM that the CAM domain power state is RETENTION.	State is RETENTION	State is not RETENTION
	PRCM_Control_start_restore	9	When asserted by the PRM, the Control module starts restoring pad configuration which has been saved before going to OFF mode. See Chapter 7, *SCM*, for more information about the save and restore mechanism.	-	-
	PRCM_device_is_lp	10	Indicates whether the voltage supply of the device is in a low power mode: OFF, RETENTION or SLEEP.	Low power acticated	Low power not activated
	PRCM_MPU_SRAMAGOODOUT[0]	11	MPU array Powergood indication from SRAM to PSCON (bit 0).	Power is stable	Power is not stable
	PRCM_IVA2_SRAMAONIN[4]	12	SRAM array power control input (bit 4) for IVA2 power domain.	Array is powered	Array is not powered
	PRCM_CORE_POWER_GOOD[0]	13	Indicates whether the CORE power switches status (bit 0).	Power is stable	Power is not stable
	PRCM_CORE_SRAMRETONIN[1]	14	SRAM retention on signal to CORE power domain (bit 1).	Retention is ON	Retention is OFF
	PRCM_SGX_POWER_GOOD[1]	15	Indicates whether the SGX power switches status (bit 1).	Power is stable	Power is not stable
	PRCM_CAM_POWER_ISO	16	PM command for CAM domain isolation.	Isolation is ON	Isolation is OFF
	This information is not available in public domain.	17	This information is not available in public domain.		
	PRCM_PER_POWER_GOOD[1]	18	Indicates whether the PER power switches status (bit 1).	Power is stable	Power is not stable
	PRCM_emudpll3srcclkactivityctrl	19	Indicates (when enabled and required) if DPLL3 can be forced in lock mode in order to ensure high speed emulation or trace clocks.	DPLL lock can be forced	DPLL lock can not be forced
	PRCM_IVA2_WUGEN_SWAKEUP	20	IVA2 asserts this signal to initiate a transition to WAKEUP mode.	Transition initiated	Wakeup conditions not reached
	Reserved	(31:21)	-	-	-

(1) Mode 5 in MUXMODE1 field CONTROL.CONTROL_PADCONF_CAM_HS[18:16]

Table 7-43. Internal Signals Multiplexed on WKUPOBSMUX2

Pin Name	Observed Signal Name	WKUPOBSMUX2 Field CONTROL.CONTROL_WKUP_DEBOBS_0[20:16] (dec)	Description	High State	Low State
hw_dbg2[1]	CORE_OBSMUX2	0	Signal multiplexed by OBSMUX2.	-	-
	PRCM_DPLL2_ALWON_FCLK	1	Always-on functional clock of DPLL2.	-	-
	PRCM_GPT7_ALWON_FCLK	2	Always-on functional clock of GP-Timer #7.	-	-
	Reserved	3	-	-	-
	PRCM_BANDGAP_RSTPWRON	4	Reset signal for BANDGAP.	Reset not active	Reset active
	PRCM_cam_domain_is_off	5	Indicates to the global Power Manager FSM that the CAM domain power state is OFF.	State is OFF	State is not OFF
	PRCM_CAM_domainIsOn	6	Indicates to the global Power Manager FSM that the CAM domain power state is ON.	State is ON	State is not ON
	PRCM_EMU_domainWakeup	7	Indicates that a wake-up condition is detected for the EMU domain.	Conditions reached	Conditions not reached
	PRCM_vdd2_domain_is_sleep	8	Indicates to the global Power Manager FSM that the VDD2 domain power state is SLEEP.	State is SLEEP	State is not SLEEP
	PRCM_Control_restore_done	9	Signal asserted by the Control module when pad configuration has been restored. See Chapter 7, *SCM*, for more information about the save and restore mechanism.	Context restored	Context not restored yet
	Reserved	10	-	-	-
	PRCM_MPU_SRAMAGOODOUT[1]	11	MPU array Powergood indication from SRAM to PSCON (bit 1).	Power is stable	Power is not stable
	PRCM_IVA2_SRAMAGOODOUT[0]	12	IVA2 array Powergood indication from SRAM to PSCON (bit 0).	Power is stable	Power is not stable
	PRCM_CORE_POWER_GOOD[1]	13	Indicates whether the CORE power switches status (bit 1).	Power is stable	Power is not stable
	PRCM_CORE_SRAMRETONIN[2]	14	SRAM retention on signal to CORE power domain (bit 2).	Retention is ON	Retention is OFF
	PRCM_SGX_POWER_GOOD[2]	15	Indicates whether the SGX power switches status (bit 2).	Power is stable	Power is not stable
	PRCM_CAM_POWER_GOOD[0]	16	Indicates whether the CAM power switches status (bit 0).	Power is stable	Power is not stable
	PRCM_DPLL1_POWER_ON	17	DPLL1 switch command for power-on.	Power is ON	Power is OFF
	PRCM_PER_POWER_GOOD[2]	18	Indicates whether the PER power switches status (bit 2).	Power is stable	Power is not stable
	PRCM_EMU_block_reset_cm	19	Reset signal of the block that controls the EMU domain.	Reset not active	Reset active
	PRCM_DSS_SWAKEUP	20	DSS asserts this signal to initiate a transition to WAKEUP mode.	Transition initiated	Wakeup conditions not reached
	Reserved	(31:21)	-	-	-

(1) Mode 5 in MUXMODE1 field CONTROL.CONTROL_PADCONF_CAM_XCLKA[18:16]

Table 7-44. Internal Signals Multiplexed on WKUPOBSMUX3

Pin Name	Observed Signal Name	WKUPOBSMUX3 Field CONTROL.CONTROL_WKUP_DEBOBS_0[28:24] (dec)	Description	High State	Low State
hw_dbg3(1)	CORE_OBSMUX3	0	Signal multiplexed by OBSMUX3.	-	-
	PRCM_DPLL3_ALWON_FCLK	1	Always-on functional clock of DPLL3.	-	-
	PRCM_GPT8_ALWON_FCLK	2	Always-on functional clock of GP-Timer #8.	-	-
	Reserved	3	-	-	-
	PRCM_MPU_WD_RST	4	Reset signal for MPU from MPU-Watchdog.	Reset not active	Reset active
	PRCM_MPU_domainWakeup	5	Indicates that a wake-up condition is detected for the MPU domain.	Conditions reached	Conditions not reached
	PRCM_efuseClk_is_not_running	6	Indicates whether the interface clock of eFuse is running or not.	Clock is not running	Clock is running
	PRCM_EMU_domainIsOn	7	Indicates to the global Power Manager FSM that the EMU domain power state is ON.	State is ON	State is not ON
	PRCM_vdd2_domain_is_lp	8	Indicates whether the voltage supply of the VDD2 domain is in a low power mode: OFF, RETENTION or SLEEP.	Low power acticated	Low power not activated
	PRCM_SYS_CLKREQ_out	9	Value of the SYS_CLKREQ pad when used as an output.	-	-
	Reserved	10	-	-	-
	PRCM_MPU_SRAMRETONIN[0]	11	SRAM retention on signal to MPU power domain (bit 0).	Retention is ON	Retention is OFF
	PRCM_IVA2_SRAMAGOODOUT[1]	12	IVA2 array Powergood indication from SRAM to PSCON (bit 1).	Power is stable	Power is not stable
	PRCM_CORE_POWER_GOOD[2]	13	Indicates whether the CORE power switches status (bit 2).	Power is stable	Power is not stable
	PRCM_CORE_SRAMRETONIN[3]	14	SRAM retention on signal to CORE power domain (bit 3).	Retention is ON	Retention is OFF
	PRCM_SGX_POWER_GOOD[3]	15	Indicates whether the SGX power switches status (bit 3).	Power is stable	Power is not stable
	PRCM_CAM_POWER_GOOD[1]	16	Indicates whether the CAM power switches status (bit 1).	Power is stable	Power is not stable
	PRCM_DPLL1_POWER_ISO	17	PM command for DPLL1 domain isolation.	Isolation is ON	Isolation is OFF
	PRCM_PER_POWER_GOOD[3]	18	Indicates whether the PER power switches status (bit 3).	Power is stable	Power is not stable
	PRCM_FORCEACTIVEWKUP	19	If asserted, the WKUP domain is unable to start a sleep transition.	Sleep transition impossible	Sleep transition possible
	PRCM_USBHOST_SWAKEUP	20	USBHOST asserts this signal to initiate a transition to WAKEUP mode.	Transition initiated	Wakeup conditions not reached
	Reserved	(31:21)	-	-	-

(1) Mode 5 in MUXMODE0 field CONTROL.CONTROL_PADCONF_CAM_FLD[2:0]

Table 7-45. Internal Signals Multiplexed on WKUPOBSMUX4

Pin Name	Observed Signal Name	WKUPOBSMUX4 Field CONTROL.CONTROL_WKUP_DEBOBS_1[4:0] (dec)	Description	High State	Low State
hw_dbg4(1)	CORE_OBSMUX4	0	Signal multiplexed by OBSMUX4.	-	-
	PRCM_DPLL4_ALWON_FCLK	1	Always-on functional clock of DPLL4.	-	-
	PRCM_GPT9_ALWON_FCLK	2	Always-on functional clock of GP-Timer #9.	-	-
	PRCM_CAM_RST	3	Reset signal for CAM module.	Reset not active	Reset active
	Reserved for non-GP devices	4	Reserved for non-GP devices	Reset not active	Reset active
	PRCM_MPU_domainIsOn	5	Indicates to the global Power Manager FSM that the MPU domain power state is ON.	State is ON	State is not ON
	PRCM_mpu_domain_is_inactive	6	Indicates whether the MPU domain is active or not.	Domain is inactive	Domain is active
	PRCM_mpu_domain_is_retention	7	Indicates to the global Power Manager FSM that the MPU domain power state is RETENTION.	State is RETENTION	State is not RETENTION
	PRCM_mpu_domain_is_off	8	Indicates to the global Power Manager FSM that the MPU domain power state is OFF.	State is OFF	State is not OFF
	PRCM_SYS_CLKREQ_in	9	When SYS_CLKREQ is used as an input, it is used to control the SYS.CLKOUT1 (and the internal oscillator).	-	-
	PRCM_vdd1_domain_go_off_mode	10	Indicates a transition of the VDD1 domain to OFF_MODE.	Transition initiated	No transition
	PRCM_MPU_SRAMRETONIN[1]	11	SRAM retention on signal to MPU power domain (bit 1).	Retention is ON	Retention is OFF
	PRCM_IVA2_SRAMAGOODOUT[2]	12	IVA2 array Powergood indication from SRAM to PSCON (bit 2).	Power is stable	Power is not stable
	PRCM_CORE_POWER_GOOD[3]	13	Indicates whether the CORE power switches status (bit 3).	Power is stable	Power is not stable
	PRCM_CORE_SRAMRETONIN[4]	14	SRAM retention on signal to CORE power domain (bit 4).	Retention is ON	Retention is OFF
	PRCM_SGX_SRAMAONIN[0]	15	SRAM array power control input (bit 2) for SGX power domain.	Array is powered	Array is not powered
	PRCM_CAM_POWER_GOOD[2]	16	Indicates whether the CAM power switches status (bit 2).	Power is stable	Power is not stable
	PRCM_DPLL1_POWER_GOOD[0]	17	Indicates whether the DPLL1 power switches status (bit 0).	Power is stable	Power is not stable
	PRCM_PER_POWER_RET	18	PER domain switch command for power-retention.	Retention is ON	Retention is OFF
	PRCM_FORCEACTIVECORE	19	If asserted, the CORE domain is unable to start a sleep transition.	Sleep transition impossible	Sleep transition possible
	PRCM_GPIO2_SWAKEUP	20	GPIO2 asserts this signal to initiate a transition to WAKEUP mode.	Transition initiated	Wakeup conditions not reached
	Reserved	(31:21)	-	-	-

(1) Mode 5 in MUXMODE1 field CONTROL.CONTROL_PADCONF_CAM_D1[18:16]

Table 7-46. Internal Signals Multiplexed on WKUPOBSMUX5

Pin Name	Observed Signal Name	WKUPOBSMUX5 Field CONTROL.CONTROL_WKUP_DEBOBS_1[12:8] (dec)	Description	High State	Low State
hw_dbg5[(1)]	CORE_OBSMUX5	0	Signal multiplexed by OBSMUX5.	-	-
	PRCM_DPLL5_ALWON_FCLK	1	Always-on functional clock of DPLL5.	-	-
	PRCM_MPU_RST	2	Reset signal for MPU.	Reset not active	Reset active
	PRCM_PER_RST	3	Reset signal for PER domain.	Reset not active	Reset active
	PRCM_SYNCT_RST	4	Reset signal for SYNCT domain.	Reset not active	Reset active
	PRCM_NEON_domainWakeup	5	Indicates that a wake-up condition is detected for the NEON domain.	Conditions reached	Conditions not reached
	PRCM_vdd1_domain_is_on	6	Indicates to the global Power Manager FSM that the VDD1 domain power state is ON.	State is ON	State is not ON
	PRCM_emu_domain_is_off	7	Indicates to the global Power Manager FSM that the EMU domain power state is OFF.	State is OFF	State is not OFF
	PRCM_vdd4_domain_is_on	8	Indicates to the global Power Manager FSM that the VDD4 domain power state is ON.	State is ON	State is not ON
	PRCM_SYS_CLKREQ_oen	9	Indicates whether the SYS_CLKREQ pin is used as an output (Output ENable).	Output enabled	Output disabled
	PRCM_SRAMALLRET[0]	10	SRAM retention on signal to all power domain (bit 0).	Retention is ON	Retention is OFF
	PRCM_MPU_SRAMRETGOODOUT[0]	11	Indicates whether the SRAM retention banks in the MPU domain are stable (bit 0).	Power is stable	Power is not stable
	PRCM_IVA2_SRAMAGOODOUT[3]	12	IVA2 array Powergood indication from SRAM to PSCON (bit 3).	Power is stable	Power is not stable
	PRCM_CORE_POWER_RET	13	CORE domain switch command for power-retention.	Retention is ON	Retention is OFF
	PRCM_CORE_SRAMRETONIN[5]	14	SRAM retention on signal to CORE power domain (bit 5).	Retention is ON	Retention is OFF
	PRCM_SGX_SRAMAGOODOUT[0]	15	SGX array Powergood indication from SRAM to PSCON (bit 0).	Power is stable	Power is not stable
	PRCM_CAM_POWER_GOOD[3]	16	Indicates whether the CAM power switches status (bit 3).	Power is stable	Power is not stable
	PRCM_DPLL2_POWER_ON	17	DPLL2 switch command for power-on.	Power is ON	Power is OFF
	PRCM_PER_POWER_ISO	18	PM command for PER domain isolation.	Isolation is ON	Isolation is OFF
	PRCM_WAKEEMU	19	This event is used to power-up the EMULATION power domain if it has been previously turned-off; or prevent any further transition to OFF triggered by the applicative software if it was already ON.	EMU domain is forced ON	-
	PRCM_GPT2_SWAKEUP	20	GP-Timer2 asserts this signal to initiate a transition to WAKEUP mode.	Transition initiated	Wakeup conditions not reached
	Reserved	(31:21)	-	-	-

(1) Mode 5 in MUXMODE0 field CONTROL.CONTROL_PADCONF_CAM_D3[2:0]

Table 7-47. Internal Signals Multiplexed on WKUPOBSMUX6

Pin Name	Observed Signal Name	WKUPOBSMUX6 Field CONTROL.CONTROL_WKUP_DEBOBS_1[20:16] (dec)	Description	High State	Low State
hw_dbg6[(1)]	CORE_OBSMUX6	0	Signal multiplexed by OBSMUX6.	-	-
	This information is not available in public domain.	1	This information is not available in public domain.	-	-
	PRCM_MPU_RSTPWRON	2	Cold reset signal for MPU.	Reset not active	Reset active
	PRCM_PER_RSTRET	3	Retention reset signal for PER domain.	Reset not active	Reset active
	Reserved for non-GP devices	4	Reserved for non-GP devices	Reset not active	Reset active
	PRCM_NEON_domainIsOn	5	Indicates to the global Power Manager FSM that the NEON domain power state is ON.	State is ON	State is not ON
	PRCM_neon_domain_is_inactive	6	Indicates whether the NEON domain is active or not.	Domain is inactive	Domain is active
	PRCM_neon_domain_is_retention	7	Indicates to the global Power Manager FSM that the NEON domain power state is RETENTION.	State is RETENTION	State is not RETENTION
	PRCM_neon_domain_is_off	8	Indicates to the global Power Manager FSM that the NEON domain power state is OFF.	State is OFF	State is not OFF
	Reserved	9	-	-	-
	PRCM_SRAMALLRET[1]	10	SRAM retention on signal to all power domain (bit 1).	Retention is ON	Retention is OFF
	PRCM_MPU_SRAMRETGOODOUT[1]	11	Indicates whether the SRAM retention banks in the MPU domain are stable (bit 1).	Power is stable	Power is not stable
	PRCM_IVA2_SRAMAGOODOUT[4]	12	IVA2 array Powergood indication from SRAM to PSCON (bit 4).	Power is stable	Power is not stable
	PRCM_CORE_SRAMAONIN[0]	13	SRAM array power control input (bit 0) for CORE power domain.	Array is powered	Array is not powered
	PRCM_CORE_SRAMRETGOODOUT[0]	14	Indicates whether the SRAM retention banks in the CORE domain are stable (bit 0).	Power is stable	Power is not stable
	PRCM_SGX_SRAMRETONIN[0]	15	SRAM retention on signal to SGX power domain (bit 0).	Retention is ON	Retention is OFF
	PRCM_CAM_SRAMAONIN[0]	16	SRAM array power control input (bit 0) for CAM power domain.	Array is powered	Array is not powered
	PRCM_DPLL2_POWER_ISO	17	PM command for DPLL2 domain isolation.	Isolation is ON	Isolation is OFF
	PRCM_PER_SRAMAONIN[0]	18	SRAM array power control input (bit 0) for PER power domain.	Array is powered	Array is not powered
	PRCM_IPPWR	19	Indicates whether the emulation power domain is fully operational.	EMU power operational	EMU power not operational
	PRCM_MCBSP2_SWAKEUP	20	MCBSP2 asserts this signal to initiate a transition to WAKEUP mode.	Transition initiated	Wakeup conditions not reached
	Reserved	(31:21)	-	-	-

(1) Mode 5 in MUXMODE1 field CONTROL.CONTROL_PADCONF_CAM_D3[18:16] 894

Table 7-48. Internal Signals Multiplexed on WKUPOBSMUX7

Pin Name	Observed Signal Name	WKUPOBSMUX7 Field CONTROL.CONTROL_WKUP_DEBOBS_1[28:24] (dec)	Description	High State	Low State
hw_dbg7[1]	CORE_OBSMUX7	0	Signal multiplexed by OBSMUX7.	-	-
	Reserved	1	-	-	-
	PRCM_IVA2_RST1	2	Reset signal 1 for IVA2.	Reset not active	Reset active
	PRCM_WKUP_RST	3	Reset signal for WKUP domain.	Reset not active	Reset active
	PRCM_ICEPICK_RST	4	Reset signal for ICEPICK module.	Reset not active	Reset active
	PRCM_IVA2_domainWakeup	5	Indicates that a wake-up condition is detected for the IVA2 domain.	Conditions reached	Conditions not reached
	PRCM_usbhost_domain_is_inactive	6	Indicates whether the USBHOST domain is active or not.	Domain is inactive	Domain is active
	PRCM_usbhost_domain_is_retention	7	Indicates to the global Power Manager FSM that the USBHOST domain power state is RETENTION.	State is RETENTION	State is not RETENTION
	PRCM_usbhost_domain_is_off	8	Indicates to the global Power Manager FSM that the USBHOST domain power state is OFF.	State is OFF	State is not OFF
	Reserved	9	-	-	-
	PRCM_SRAMALLOFF[0]	10	Signal is asserted by the PRM when the device enters in OFF mode(bit 0).	State is OFF	State is not OFF
	PRCM_IVA2_POWER_ON	11	IVA2 switch command for power-on.	Power is ON	Power is OFF
	PRCM_IVA2_SRAMRETONIN[0]	12	SRAM retention on signal to IVA2 power domain (bit 0).	Retention is ON	Retention is OFF
	PRCM_CORE_SRAMAONIN[1]	13	SRAM array power control input (bit 1) for CORE power domain.	Array is powered	Array is not powered
	PRCM_CORE_SRAMRETGOODOUT[1]	14	Indicates whether the SRAM retention banks in the CORE domain are stable (bit 1).	Power is stable	Power is not stable
	PRCM_SGX_SRAMRETGOODOUT[0]	15	Indicates whether the SRAM retention banks in the SGX domain are stable (bit 0).	Power is stable	Power is not stable
	PRCM_CAM_SRAMAGOODOUT[0]	16	CAM array Powergood indication from SRAM to PSCON (bit 0).	Power is stable	Power is not stable
	PRCM_DPLL2_POWER_GOOD[0]	17	Indicates whether the DPLL2 power switches status (bit 0).	Power is stable	Power is not stable
	PRCM_PER_SRAMAGOODOUT[0]	18	PER array Powergood indication from SRAM to PSCON (bit 0).	Power is stable	Power is not stable
	Reserved for non-GP devices	19	Reserved for non-GP devices .	Dectection possible	Detection impossible
	PRCM_USBHS_SWAKEUP	20	USBOTGHS asserts this signal to initiate a transition to WAKEUP mode.	Transition initiated	Wakeup conditions not reached
	Reserved	(31:21)	-	-	-

(1) Mode 5 in MUXMODE0 field CONTROL.CONTROL_PADCONF_CAM_D5[2:0]

Table 7-49. Internal Signals Multiplexed on WKUPOBSMUX8

Pin Name	Observed Signal Name	WKUPOBSMUX8 Field CONTROL.CONTROL_WKUP_DEBOBS_2[4:0] (dec)	Description	High State	Low State
hw_dbg8(1)	CORE_OBSMUX8	0	Signal multiplexed by OBSMUX8.	-	-
	PRCM_USBTHOST_SAR_FCLK	1	Functional clock of USBHOST.	-	-
	PRCM_NEON_RST	2	Reset signal for NEON.	Reset not active	Reset active
	PRCM_WKUP_RSTPWRON	3	Cold reset signal for WKUP domain.	Reset not active	Reset active
	PRCM_VDD1_VOLCON_RST	4	Reset signal for VDD1_VOLCON.	Reset not active	Reset active
	PRCM_IVA2_domainIsOn	5	Indicates to the global Power Manager FSM that the IVA2 domain power state is ON.	State is ON	State is not ON
	PRCM_iva2_domain_is_inactive	6	Indicates whether the IVA2 domain is active or not.	Domain is inactive	Domain is active
	PRCM_iva2_domain_is_retention	7	Indicates to the global Power Manager FSM that the IVA2 domain power state is RETENTION.	State is RETENTION	State is not RETENTION
	PRCM_iva2_domain_is_off	8	Indicates to the global Power Manager FSM that the IVA2 domain power state is OFF.	State is OFF	State is not OFF
	Reserved	9	-	-	-
	PRCM_SRAMALLOFF[1]	10	Signal is asserted by the PRM when the device enters in OFF mode (bit 1).	State is OFF	State is not OFF
	PRCM_IVA2_POWER_ISO	11	PM command for IVA2 domain isolation.	Isolation is ON	Isolation is OFF
	PRCM_IVA2_SRAMRETONIN[1]	12	SRAM retention on signal to IVA2 power domain (bit 1).	Retention is ON	Retention is OFF
	PRCM_CORE_SRAMAONIN[2]	13	SRAM array power control input (bit 2) for CORE power domain.	Array is powered	Array is not powered
	PRCM_CORE_SRAMRETGOODOUT[2]	14	Indicates whether the SRAM retention banks in the CORE domain are stable (bit 2).	Power is stable	Power is not stable
	PRCM_DSS_POWER_ON	15	DSS switch command for power-on.	Power is ON	Power is OFF
	PRCM_CAM_SRAMRETONIN[0]	16	SRAM retention on signal to CAM power domain (bit 0).	Retention is ON	Retention is OFF
	PRCM_DPLL3_POWER_ON	17	DPLL3 switch command for power-on.	Power is ON	Power is OFF
	PRCM_PER_SRAMRETONIN[0]	18	SRAM retention on signal to PER power domain (bit 0).	Retention is ON	Retention is OFF
	PRCM_USBHOST_ISO_SAR	19	PM command for USBHOST domain isolation.	Isolation is ON	Isolation is OFF
	PRCM_GPT10_SWAKEUP	20	GP-Timer10 asserts this signal to initiate a transition to WAKEUP mode.	Transition initiated	Wakeup conditions not reached
	Reserved	(31:21)	-	-	-

(1) Mode 5 in MUXMODE1 field CONTROL.CONTROL_PADCONF_CAM_D9[18:16]

Table 7-50. Internal Signals Multiplexed on WKUPOBSMUX9

Pin Name	Observed Signal Name	WKUPOBSMUX9 Field CONTROL.CONTROL_WKUP_DEBOBS_2[12:8] (dec)	Description	High State	Low State
hw_dbg9[1]	CORE_OBSMUX9	0	Signal multiplexed by OBSMUX9.	-	-
	PRCM_USBTLL_SAR_FCLK	1	Functional clock of USBTLL.	-	-
	PRCM_IVA2_RST2	2	Reset signal 2 for IVA2.	Reset not active	Reset active
	PRCM_EMU_RST	3	Reset signal for EMU domain.	Reset not active	Reset active
	PRCM_VDD2_VOLCON_RST	4	Reset signal for VDD2_VOLCON.	Reset not active	Reset active
	PRCM_COREL3_domainWakeup	5	Indicates that a wake-up condition is detected for the COREL3 domain.	Conditions reached	Conditions not reached
	PRCM_vdd1_domain_is_good	6	Indicates whether the VDD1 power domain is stable.	Power is stable	Power is not stable
	PRCM_vdd1_domain_is_sleep	7	Indicates to the global Power Manager FSM that the VDD1 domain power state is SLEEP.	State is SLEEP	State is not SLEEP
	PRCM_vdd4_domain_is_off	8	Indicates to the global Power Manager FSM that the VDD4 domain power state is OFF.	State is OFF	State is not OFF
	PRCM_Device_wakeup	9	When a wakeup transition occurs, this signal (sent by the last pad) indicates that all pads have been waken-up.	All pads have been waken-up	Some pads are in SLEEP mode
	PRCM_MPU_POWER_ON	10	MPU switch command for power-on.	Power is ON	Power is OFF
	PRCM_IVA2_POWER_GOOD[0]	11	Indicates whether the IVA2 power switches status (bit 0).	Power is stable	Power is not stable
	PRCM_IVA2_SRAMRETONIN[2]	12	SRAM retention on signal to IVA2 power domain (bit 2).	Retention is ON	Retention is OFF
	PRCM_CORE_SRAMAONIN[3]	13	SRAM array power control input (bit 3) for CORE power domain.	Array is powered	Array is not powered
	PRCM_CORE_SRAMRETGOODOUT[3]	14	Indicates whether the SRAM retention banks in the CORE domain are stable (bit 3).	Power is stable	Power is not stable
	PRCM_DSS_POWER_ISO	15	PM command for DSS domain isolation.	Isolation is ON	Isolation is OFF
	PRCM_CAM_SRAMRETGOODOUT[0]	16	Indicates whether the SRAM retention banks in the CAM domain are stable (bit 0).	Power is stable	Power is not stable
	PRCM_DPLL3_POWER_ISO	17	PM command for DPLL3 domain isolation.	Isolation is ON	Isolation is OFF
	PRCM_PER_SRAMRETGOODOUT[0]	18	Indicates whether the SRAM retention banks in the PER domain are stable (bit 0).	Power is stable	Power is not stable
	PRCM_USBHOST_STARTSAVE	19	Context Save Start Control.	Started	Not started
	PRCM_USBTLL_SWAKEUP	20	USBTLL asserts this signal to initiate a transition to WAKEUP mode.	Transition initiated	Wakeup conditions not reached
	Reserved	(31:21)	-	-	-

(1) Mode 5 in MUXMODE0 field CONTROL.CONTROL_PADCONF_CAM_D11[2:0]

Table 7-51. Internal Signals Multiplexed on WKUPOBSMUX10

Pin Name	Observed Signal Name	WKUPOBSMUX10 Field CONTROL.CONTROL_WKUP_DEBOBS_2[20:16] (dec)	Description	High State	Low State
hw_dbg10(1)	CORE_OBSMUX10	0	Signal multiplexed by OBSMUX 10.	-	-
	PRCM_FUNC_96M_ALWON_CLK	1	96 MHz Alway-on functional clock.	-	-
	PRCM_IVA2_RST3	2	Reset signal 3 for IVA2 domain.	Reset not active	Reset active
	PRCM_EMU_RSTPWRON	3	Cold Reset signal for EMU domain.	Reset not active	Reset active
	PRCM_CM_RSTPWRONRET	4	Cold retention reset signal for CM.	Reset not active	Reset active
	PRCM_COREL3_domainIsOn	5	Indicates to the global Power Manager FSM that the COREL3 domain power state is ON.	State is ON	State is not ON
	PRCM_sgx_domain_is_inactive	6	Indicates whether the SGX domain is active or not.	Domain is inactive	Domain is active
	PRCM_sgx_domain_is_retention	7	Indicates to the global Power Manager FSM that the SGX domain power state is RETENTION.	State is RETENTION	State is not RETENTION
	PRCM_sgx_domain_is_off	8	Indicates to the global Power Manager FSM that the SGX domain power state is OFF.	State is OFF	State is not OFF
	PRCM_PADS_OFF_mode	9	Indicates whether the I/O pads are in OFF-mode.	State is OFF	State is not OFF
	PRCM_MPU_POWER_ISO	10	PM command for MPU domain isolation.	Isolation is ON	Isolation is OFF
	PRCM_IVA2_POWER_GOOD[1]	11	Indicates whether the IVA2 power switches status (bit 1).	Power is stable	Power is not stable
	PRCM_IVA2_SRAMRETONIN[3]	12	SRAM retention on signal to IVA2 power domain (bit 3).	Retention is ON	Retention is OFF
	PRCM_CORE_SRAMAONIN[4]	13	SRAM array power control input (bit 4) for CORE power domain.	Array is powered	Array is not powered
	PRCM_CORE_SRAMRETGOODOUT[4]	14	Indicates whether the SRAM retention banks in the CORE domain are stable (bit 4).	Power is stable	Power is not stable
	PRCM_DSS_POWER_GOOD[0]	15	Indicates whether the DSS power switches status (bit 0).	Power is stable	Power is not stable
	PRCM_EMU_POWER_ON	16	EMU switch command for power-on.	Power is ON	Power is OFF
	PRCM_DPLL3_POWER_GOOD[0]	17	Indicates whether the DPLL3 power switches status (bit 0).	Power is stable	Power is not stable
	PRCM_USBHOST_POWER_ON	18	USBHOST switch command for power-on.	Power is ON	Power is OFF
	PRCM_USBHOST_SAVEDONE	19	Indicates that the Context Save is done.	Context saved	Context not saved
	PRCM_GPIO1_SWAKEUP	20	GPIO1 asserts this signal to initiate a transition to WAKEUP mode.	Transition initiated	Wakeup conditions not reached
	Reserved	(31:21)	-	-	-

(1) Mode 5 in MUXMODE0 field CONTROL.CONTROL_PADCONF_CAM_WEN[2:0]

Table 7-52. Internal Signals Multiplexed on WKUPOBSMUX11

Pin Name	Observed Signal Name	WKUPOBSMUX11 Field CONTROL.CONTROL_WKUP_DEBOBS_2[28:24] (dec)	Description	High State	Low State
hw_dbg11(1)	CORE_OBSMUX11	0	Signal multiplexed by OBSMUX11.	-	-
	PRCM_DSS2_ALWON_FCLK	1	Always-on functional clock 2 of DSS module.	-	-
	PRCM_IVA2_RSTPWRON	2	Cold reset signal for IVA2 domain.	Reset not active	Reset active
	PRCM_EFUSE_RSTPWRON	3	Cold reset signal for EFUSE.	Reset not active	Reset active
	PRCM_DPLL3_rstdata	4	Reset signal for DPLL3.	Reset active	Reset not active
	PRCM_SGX_domainWakeup	5	Indicates that a wake-up condition is detected for the SGX domain.	Conditions reached	Conditions not reached
	PRCM_vdd1_domain_is_off	6	Indicates to the global Power Manager FSM that the VDD1 domain power state is OFF.	State is OFF	State is not OFF
	PRCM_vdd1_domain_is_lp	7	Indicates whether the voltage supply of the VDD1 domain is in a low power mode: OFF, RETENTION or SLEEP.	Low power acticated	Low power not activated
	PRCM_PRM2MPU_IRQ	8	Interrupt line from PRM to the MPU.	-	-
	PRCM_GLOBAL_WKUP_en	9	Signal that enables a Global Wakeup.	Wakeup enabled	Wakeup disabled
	PRCM_MPU_POWER_GOOD[0]	10	Indicates whether the MPU power switches status (bit 0).	Power is stable	Power is not stable
	PRCM_IVA2_POWER_GOOD[2]	11	Indicates whether the IVA2 power switches status (bit 2).	Power is stable	Power is not stable
	PRCM_IVA2_SRAMRETONIN[4]	12	SRAM retention on signal to IVA2 power domain (bit 4).	Retention is ON	Retention is OFF
	PRCM_CORE_SRAMAONIN[5]	13	SRAM array power control input (bit 5) for CORE power domain.	Array is powered	Array is not powered
	PRCM_CORE_SRAMRETGOODOUT[5]	14	Indicates whether the SRAM retention banks in the CORE domain are stable (bit 5).	Power is stable	Power is not stable
	PRCM_DSS_POWER_GOOD[1]	15	Indicates whether the DSS power switches status (bit 1).	Power is stable	Power is not stable
	PRCM_EMU_POWER_ISO	16	PM command for EMU domain isolation.	Isolation is ON	Isolation is OFF
	PRCM_DPLL4_POWER_ON	17	DPLL4 switch command for power-on.	Power is ON	Power is OFF
	PRCM_USBHOST_POWER_ISO	18	PM command for USBHOST domain isolation.	Isolation is ON	Isolation is OFF
	PRCM_USBHOST_STARTRESTORE	19	Context Restore Start Control.	Context restoration started	Context restoration not started
	Reserved	20	-	-	-
	PRCM_WAITINRESET_WK	21	Indicates whether the device is booting in Wait-In-Reset mode.	Wait-In-Reset mode	Standard boot
	Reserved	(31:22)	-	-	-

(1) Mode 5 in MUXMODE1 field CONTROL.CONTROL_PADCONF_CAM_WEN[18:16]

Table 7-53. Internal Signals Multiplexed on WKUPOBSMUX12

Pin Name	Observed Signal Name	WKUPOBSMUX12 Field CONTROL.CONTROL_WKUP_DEBOBS_3[4:0] (dec)	Description	High State	Low State
hw_dbg12(1)	CORE_OBSMUX12	0	Signal multiplexed by OBSMUX12.	-	-
	PRCM_EFUSE_ALWON_FCLK	1	Always-on functional clock of Efuse.	-	-
	PRCM_IVA2_RSTDONE	2	Status of the IVA2.2 module after a reset.	Reset not active	Reset active
	This information is not available in public domain.	3	This information is not available in public domain.		
	PRCM_GlbRstData	4	Global reset signal.	Reset not active	Reset active
	PRCM_SGX_domainIsOn	5	Indicates to the global Power Manager FSM that the SGX domain power state is ON.	State is ON	State is not ON
	PRCM_core_domain_is_inactive	6	Indicates whether the CORE domain is active or not.	Domain is inactive	Domain is active
	PRCM_core_domain_is_retention	7	Indicates to the global Power Manager FSM that the CORE domain power state is RETENTION.	State is RETENTION	State is not RETENTION
	PRCM_core_domain_is_off	8	Indicates to the global Power Manager FSM that the CORE domain power state is OFF.	State is OFF	State is not OFF
	PRCM_IO_WUCLK	9	I/O Wakeup clock.	-	-
	PRCM_MPU_POWER_GOOD[1]	10	Indicates whether the MPU power switches status (bit 1).	Power is stable	Power is not stable
	PRCM_IVA2_POWER_GOOD[3]	11	Indicates whether the IVA2 power switches status (bit 3).	Power is stable	Power is not stable
	PRCM_IVA2_SRAMRETGOODOUT[0]	12	Indicates whether the SRAM retention banks in the IVA2 domain are stable (bit 0).	Power is stable	Power is not stable
	PRCM_CORE_SRAMAGOODOUT[0]	13	CORE array Powergood indication from SRAM to PSCON (bit 0).	Power is stable	Power is not stable
	PRCM_NEON_POWER_ON	14	NEON switch command for power-on.	Power is ON	Power is OFF
	PRCM_DSS_POWER_GOOD[2]	15	Indicates whether the DSS power switches status (bit 2).	Power is stable	Power is not stable
	PRCM_EMU_POWER_GOOD[0]	16	Indicates whether the EMU power switches status (bit 0).	Power is stable	Power is not stable
	PRCM_DPLL4_POWER_ISO	17	PM command for DPLL4 domain isolation.	Isolation is ON	Isolation is OFF
	PRCM_USBHOST_POWER_GOOD[0]	18	Indicates whether the USBHOST power switches status (bit 0).	Power is stable	Power is not stable
	PRCM_USBHOST_RESTOREDONE	19	Indicates that the Context Restore is done.	Restore done	Restore not finished
	Reserved	(31:20)	-	-	-

(1) Mode 5 in MUXMODE0 field CONTROL.CONTROL_PADCONF_DSS_PCLK[2:0]

Table 7-54. Internal Signals Multiplexed on WKUPOBSMUX13

Pin Name	Observed Signal Name	WKUPOBSMUX13 Field CONTROL.CONTROL_WKUP_DEBOBS_3[12:8] (dec)	Description	High State	Low State
hw_dbg13(1)	CORE_OBSMUX13	0	Signal multiplexed by OBSMUX13.	-	-
	PRCM_PER_32K_ALWON_FCLK	1	Always-on 23KHz functional clock of PER domain.	-	-
	PRCM_CORE_RST	2	Reset signal for CORE domain.	Reset not active	Reset active
	PRCM_DPLL1_RSTPWRON	3	Cold reset signal for DPLL1.	Reset active	Reset not active
	PRCM_GlblWarmRst_n	4	Global warm reset signal.	Reset active	Reset not active
	PRCM_COREL4_domainWakeup	5	Indicates that a wake-up condition is detected for the COREL4 domain.	Conditions reached	Conditions not reached
	PRCM_vdd1_domain_is_ret	6	Indicates to the global Power Manager FSM that the VDD1 domain power state is RETENTION.	State is RETENTION	State is not RETENTION
	PRCM_vdd2_domain_is_on	7	Indicates to the global Power Manager FSM that the VDD2 domain power state is ON.	State is ON	State is not ON
	PRCM_USBHOST_domainWakeup	8	Indicates that a wake-up condition is detected for the USBHOST domain.	Conditions reached	Conditions not reached
	PRCM_IO_ISOCLK	9	PM Iinput/Output isolation clock.	-	-
	PRCM_MPU_POWER_GOOD[2]	10	Indicates whether the MPU power switches status (bit 2).	Power is stable	Power is not stable
	PRCM_IVA2_POWER_GOOD[4]	11	Indicates whether the IVA2 power switches status (bit 4).	Power is stable	Power is not stable
	PRCM_IVA2_SRAMRETGOODOUT[1]	12	Indicates whether the SRAM retention banks in the IVA2 domain are stable (bit 1).	Power is stable	Power is not stable
	PRCM_CORE_SRAMAGOODOUT[1]	13	CORE array Powergood indication from SRAM to PSCON (bit 1).	Power is stable	Power is not stable
	PRCM_NEON_POWER_ISO	14	PM command for NEON domain isolation.	Isolation is ON	Isolation is OFF
	PRCM_DSS_POWER_GOOD[3]	15	Indicates whether the DSS power switches status (bit 3).	Power is stable	Power is not stable
	PRCM_EMU_SRAMAONIN[0]	16	SRAM array power control input (bit 0) for EMU power domain.	Array is powered	Array is not powered
	PRCM_DPLL4_POWER_GOOD[0]	17	Indicates whether the DPLL4 power switches status (bit 0).	Power is stable	Power is not stable
	PRCM_USBHOST_SRAMAONIN[0]	18	SRAM array power control input (bit 0) for USBHOST power domain.	Array is powered	Array is not powered
	PRCM_USBTLL_ISO_SAR	19	PM command for USBTLL domain isolation.	Isolation is ON	Isolation is OFF
	PRCM_GPT1_SWAKEUP	20	FP-Timer1 asserts this signal to initiate a transition to WAKEUP mode.	Transition initiated	Wakeup conditions not reached
	Reserved	(31:21)	-	-	-

(1) Mode 5 in MUXMODE1 field CONTROL.CONTROL_PADCONF_DSS_PCLK[18:16]

Table 7-55. Internal Signals Multiplexed on WKUPOBSMUX14

Pin Name	Observed Signal Name	WKUPOBSMUX14 Field CONTROL.CONTROL_WKUP_DEBOBS_3[20:16] (dec)	Description	High State	Low State
hw_dbg14(1)	CORE_OBSMUX14	0	Signal multiplexed by OBSMUX14.	-	-
	PRCM_GPT1_FCLK	1	Functional clock of GP-Timer1.	-	-
	PRCM_CORE_RSTPWRONRET	2	Cold retention reset signal for CORE domain.	Reset not active	Reset active
	PRCM_DPLL2_RSTPWRON	3	Cold reset signal for DPLL2.	Reset active	Reset not active
	PRCM_GlblPwrOnRst_n	4	Global cold reset signal.	Reset active	Reset not active
	PRCM_COREL4_domainIsOn	5	Indicates to the global Power Manager FSM that the COREL4 domain power state is ON.	State is ON	State is not ON
	PRCM_DSS_domainWakeup	6	Indicates that a wake-up condition is detected for the DSS domain.	Conditions reached	Conditions not reached
	PRCM_vdd2_domain_is_good	7	Indicates whether the VDD2 power domain is stable.	Power is stable	Power is not stable
	PRCM_USBHOST_domainIsOn	8	Indicates to the global Power Manager FSM that the USBHOST domain power state is ON.	State is ON	State is not ON
	PRCM_IO_ISO	9	PM command for I/O isolation.	Isolation is ON	Isolation is OFF
	PRCM_MPU_POWER_GOOD[3]	10	Indicates whether the MPU power switches status (bit 3).	Power is stable	Power is not stable
	PRCM_IVA2_POWER_GOOD[5]	11	Indicates whether the IVA2 power switches status (bit 5).	Power is stable	Power is not stable
	PRCM_IVA2_SRAMRETGOODOUT[2]	12	Indicates whether the SRAM retention banks in the IVA2 domain are stable (bit 2).	Power is stable	Power is not stable
	PRCM_CORE_SRAMAGOODOUT[2]	13	CORE array Powergood indication from SRAM to PSCON (bit 2).	Power is stable	Power is not stable
	PRCM_NEON_POWER_GOOD[0]	14	Indicates whether the NEON power switches status (bit 0).	Power is stable	Power is not stable
	PRCM_DSS_SRAMAONIN[0]	15	SRAM array power control input (bit 0) for DSS power domain.	Array is powered	Array is not powered
	PRCM_EMU_SRAMAGOODOUT[0]	16	EMU array Powergood indication from SRAM to PSCON (bit 0).	Power is stable	Power is not stable
	PRCM_DPLL5_POWER_ON	17	DPLL5 switch command for power-on.	Power is ON	Power is OFF
	PRCM_USBHOST_SRAMAGOODOUT[0]	18	USBHOST array Powergood indication from SRAM to PSCON (bit 0).	Power is stable	Power is not stable
	PRCM_USBTLL_STARTSAVE	19	Context Save Start Control.	Context save started	Context save not started
	PRCM_EMU_SYS_GCLK	20	System clock of the EMU block of the PRCM module.	-	-
	Reserved	(31:21)	-	-	-

(1) Mode 5 in MUXMODE0 field CONTROL.CONTROL_PADCONF_DSS_DATA6[2:0]

Table 7-56. Internal Signals Multiplexed on WKUPOBSMUX15

Pin Name	Observed Signal Name	WKUPOBSMUX15 Field CONTROL.CONTROL_WKUP_DEBOBS_3[28:24] (dec)	Description	High State	Low State
hw_dbg15(1)	CORE_OBSMUX15	0	Signal multiplexed by OBSMUX15.	-	-
	PRCM_GPT2_ALWON_FCLK	1	Always-on functional clock of GP-Timer #2.	-	-
	PRCM_CORE_RSTRET	2	Retention reset signal for CORE domain.	Reset not active	Reset active
	PRCM_DPLL3_RSTPWRON	3	Cold reset signal for DPLL3.	Reset active	Reset not active
	Reserved for non-GP devices	4	Reserved for non-GP devices	Reset active	Reset not active
	Reserved	5	-	-	-
	PRCM_DSS_domainIsOn	6	Indicates to the global Power Manager FSM that the DSS domain power state is ON.	State is ON	State is not ON
	PRCM_dss_domain_is_inactive	7	Indicates whether the DSS domain is active or not.	Domain is inactive	Domain is active
	PRCM_dss_domain_is_retention	8	Indicates to the global Power Manager FSM that the DSS domain power state is RETENTION.	State is RETENTION	State is not RETENTION
	PRCM_dss_domain_is_off	9	Indicates to the global Power Manager FSM that the DSS domain power state is OFF.	State is OFF	State is not OFF
	PRCM_MPU_POWER_GOOD[4]	10	Indicates whether the MPU power switches status (bit 4).	Power is stable	Power is not stable
	PRCM_IVA2_SRAMAONIN[0]	11	SRAM array power control input (bit 0) for IVA2 power domain.	Array is powered	Array is not powered
	PRCM_IVA2_SRAMRETGOODOUT[3]	12	Indicates whether the SRAM retention banks in the IVA2 domain are stable (bit 3).	Power is stable	Power is not stable
	PRCM_CORE_SRAMAGOODOUT[3]	13	CORE array Powergood indication from SRAM to PSCON (bit 3).	Power is stable	Power is not stable
	PRCM_NEON_POWER_GOOD[1]	14	Indicates whether the NEON power switches status (bit 1).	Power is stable	Power is not stable
	PRCM_DSS_SRAMAGOODOUT[0]	15	DSS array Powergood indication from SRAM to PSCON (bit 0).	Power is stable	Power is not stable
	PRCM_EMU_SRAMRETONIN[0]	16	SRAM retention on signal to EMU power domain (bit 0).	Retention is ON	Retention is OFF
	PRCM_DPLL5_POWER_ISO	17	PM command for DPLL5 domain isolation.	Isolation is ON	Isolation is OFF
	PRCM_USBHOST_SRAMRETONIN[0]	18	SRAM retention on signal to USBHOST power domain (bit 0).	Retention is ON	Retention is OFF
	PRCM_USBTLL_SAVEDONE	19	Indicates that the Context Save is done.	Context saved	Context not saved yet
	Reserved	(31:20)	-	-	-

(1) Mode 5 in MUXMODE1 field CONTROL.CONTROL_PADCONF_DSS_DATA6[18:16]

Table 7-57. Internal Signals Multiplexed on WKUPOBSMUX16

Pin Name	Observed Signal Name	WKUPOBSMUX16 Field CONTROL.CONTROL_WKUP_DEBOBS_4[4:0] (dec)	Description	High State	Low State
hw_dbg16(1)	CORE_OBSMUX16	0	Signal multiplexed by OBSMUX16.	-	-
	PRCM_GPT3_ALWON_FCLK	1	Always-on functional clock of GP-Timer #3.	-	-
	PRCM_CPEFUSE_RST	2	Reset signal for Efuse.	Reset not active	Reset active
	PRCM_DPLL4_RSTPWRON	3	Cold reset signal for DPLL4.	Reset active	Reset not active
	Reserved for non-GP devices	4	Reserved for non-GP devices	Reset not active	Reset active
	Reserved	5	-	-	-
	PRCM_PER_domainWakeup	6	Indicates that a wake-up condition is detected for the PER domain.	Conditions reached	Conditions not reached
	PRCM_vdd2_domain_is_off	7	Indicates to the global Power Manager FSM that the VDD2 domain power state is OFF.	State is OFF	State is not OFF
	PRCM_vdd5_domain_is_off	8	Indicates to the global Power Manager FSM that the VDD5 domain power state is OFF.	State is OFF	State is not OFF
	PRCM_PRM2IVA2_IRQ	9	Interrupt line from PRM to the IVA2.	-	-
	PRCM_MPU_POWER_GOOD[5]	10	Indicates whether the MPU power switches status (bit 5).	Power is stable	Power is not stable
	PRCM_IVA2_SRAMAONIN[1]	11	SRAM array power control input (bit 4) for IVA2 power domain.	Array is powered	Array is not powered
	PRCM_IVA2_SRAMRETGOODOUT[4]	12	Indicates whether the SRAM retention banks in the IVA2 domain are stable (bit 4).	Power is stable	Power is not stable
	PRCM_CORE_SRAMAGOODOUT[4]	13	CORE array Powergood indication from SRAM to PSCON (bit 4).	Power is stable	Power is not stable
	PRCM_SGX_POWER_ON	14	SGX switch command for power-on.	Power is ON	Power is OFF
	PRCM_DSS_SRAMRETONIN[0]	15	SRAM retention on signal to DSS power domain (bit 0).	Retention is ON	Retention is OFF
	PRCM_EMU_SRAMRETGOODOUT[0]	16	Indicates whether the SRAM retention banks in the EMU domain are stable (bit 0).	Power is stable	Power is not stable
	PRCM_DPLL5_POWER_GOOD[0]	17	Indicates whether the DPLL5 power switches status (bit 0).	Power is stable	Power is not stable
	PRCM_USBHOST_SRAMRETGOODOUT[0]	18	Indicates whether the SRAM retention banks in the USBHOST domain are stable (bit 0).	Power is stable	Power is not stable
	PRCM_USBTLL_STARTRESTORE	19	Context Restore Start Control.	Context restore saved	Saving context
	PRCM_WKUP_32K_GFCLK	20	32KHz functional clock of the WKUP domain.	-	-
	Reserved	(31:21)	-	-	-

(1) Mode 5 in MUXMODE0 field CONTROL.CONTROL_PADCONF_DSS_DATA8[2:0]

Table 7-58. Internal Signals Multiplexed on WKUPOBSMUX17

Pin Name	Observed Signal Name	WKUPOBSMUX17 Field CONTROL.CONTROL_WKUP_DEBOBS_4[12:8] (dec)	Description	High State	Low State
hw_dbg17[1]	CORE_OBSMUX17	0	Signal multiplexed by OBSMUX17.	-	-
	PRCM_GPT4_ALWON_FCLK	1	Always-on functional clock of GP-Timer #4.	-	-
	PRCM_USBTLL_RST	2	Reset signal for USBTLL.	Reset not active	Reset active
	PRCM_DPLL5_RSTPWRON	3	Cold reset signal for DPLL5.	Reset active	Reset not active
	PRCM_ICECRUSHER_RST	4	Reset signal for ICECRUSHER.	Reset not active	Reset active
	PRCM_CORECM_domainWakeup	5	Indicates that a wake-up condition is detected for the CORECM domain.	Conditions reached	Conditions not reached
	PRCM_PER_domainIsOn	6	Indicates to the global Power Manager FSM that the PER domain power state is ON.	State is ON	State is not ON
	PRCM_PER_domain_is_inactive	7	Indicates whether the PER domain is active or not.	Domain is inactive	Domain is active
	PRCM_PER_domain_is_retention	8	Indicates to the global Power Manager FSM that the PER domain power state is RETENTION.	State is RETENTION	State is not RETENTION
	PRCM_PER_domain_is_off	9	Indicates to the global Power Manager FSM that the PER domain power state is OFF.	State is OFF	State is not OFF
	PRCM_MPU_SRAMAONIN[0]	10	SRAM array power control input (bit 0) for MPU power domain.	Array is powered	Array is not powered
	PRCM_IVA2_SRAMAONIN[2]	11	SRAM array power control input (bit 2) for IVA2 power domain.	Array is powered	Array is not powered
	PRCM_CORE_POWER_ON	12	CORE switch command for power-on.	Power is ON	Power is OFF
	PRCM_CORE_SRAMAGOODOUT[5]	13	CORE array Powergood indication from SRAM to PSCON (bit 5).	Power is stable	Power is not stable
	PRCM_SGX_POWER_ISO	14	PM command for SGX domain isolation.	Isolation is ON	Isolation is OFF
	PRCM_DSS_SRAMRETGOODOUT[0]	15	Indicates whether the SRAM retention banks in the DSS domain are stable (bit 0).	Power is stable	Power is not stable
	This information is not available in public domain.	16	This information is not available in public domain.	Power is ON	Power is OFF
	PRCM_PER_POWER_ON	17	PER switch command for power-on.	Power is ON	Power is OFF
	PRCM_ldoemuonz	18	WAKE-UP LDO ON signal for EMU domain.	Power is ON	Power is OFF
	PRCM_USBTLL_RESTOREDONE	19	Indicates that the Context Restore is done.	Context restored	Context restoration
	PRCM_WKUP_L4_GICLK	20	Interface clock of WKUPL4 domain.	-	-
	Reserved	(31:21)	-	-	-

(1) Mode 5 in MUXMODE1 field CONTROL.CONTROL_PADCONF_DSS_DATA8[18:16]

7.4.11 Electomagnetic Interference Reduction for Clocking Generation (Spreading)

7.4.11.1 Overview

There are two major forms of digital signals: digital-data and digital-clock (CLK). Digital signals are the principal signal form in today's digital electronic products.

- A digital-data signal can be viewed as a sequence of pulses with different pulse widths
- A clock (CLK) signal is usually a string of rectangular pulses with the same pulse width

In the case of a clock, the generated signal has a predetermined frequency. Electromagnetic radiations such as radio frequency emissions are also generated from the constant frequency clock signals and their harmonics. Unfortunately, a problem exists with some devices such as communication devices for example in that the system clock generates unwanted spurious signals that interfere with the decoding of information from received signals by a receiver of the communication device. Note that data periodicity also causes interference.

This periodicity generates a significant power peak at the selected frequency, which in turn causes an EMI disturbance to the environment.

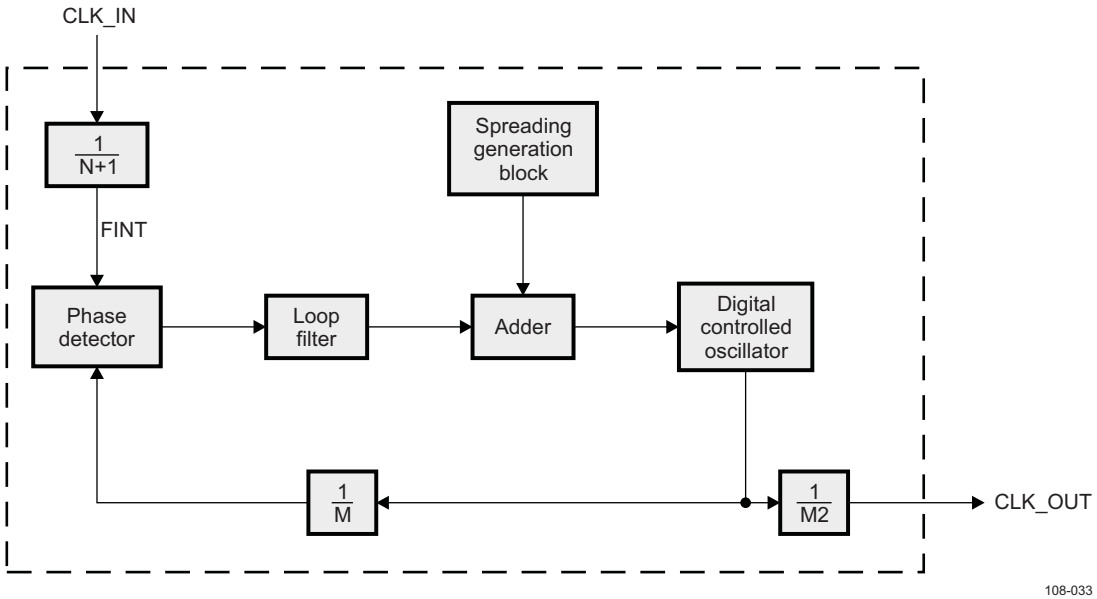
The harmonics of the generated signal system clock radiates in the other modules of the device, and the spurious energy can be enough to cause enough interference, which results in performance degradation (in the form of high bit error rates in case of a wireless communication interface for example).

In order to reduce the power peaks (and thus the energy of the electromagnetic noise generated for a specific frequency), an internal frequency modulation of the DPLL is used to distribute the energy to many different frequencies, thus reducing the power peaks.

This frequency modulation feature is directly integrated in some DPLLs of the device. The DPLLs that support this additional feature are called DPLL-D in the following.

Figure 7-18 shows a diagram of a DPLL that supports the EMI reduction feature.

Figure 7-18. DPLL With EMI Reduction Feature



The additional features of the DPLL compliant with EMI reduction are:

- EMI reduction using triangular frequency modulation, also called spread sprectrum clocking (SSC).

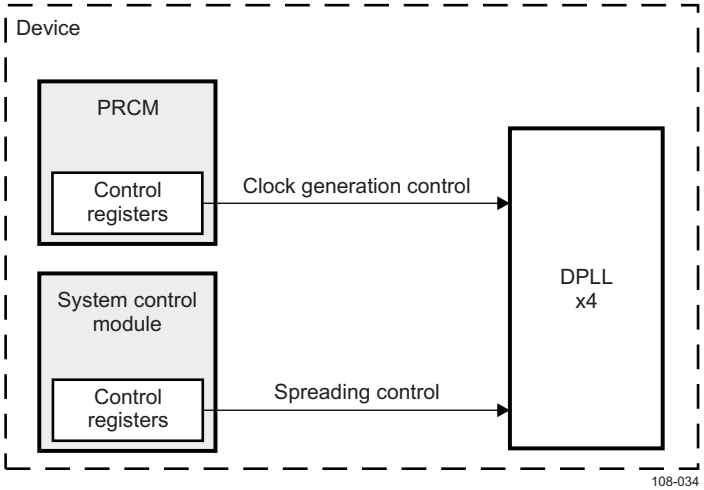
7.4.11.2 Integration

The DPLLs/domains of the devices that support the EMI reduction feature are:

- DSS/DSI DPLL
- CORE domain DPLL
- PER domain DPLL
- USBHOST DPLL

Figure 7-19 shows the four DPLL-D integration in the device.

Figure 7-19. DPLL-D Integration



The control of the DPLL is done by the PRCM and the SCM:

- The PRCM controls the clocking generation of the DPLL. For more information, see Chapter 4, *Power, Reset, and Clock Management*.
- The SCM contains all necessary bits that controls the EMI reduction feature (SSC).

7.4.11.2.1 Clocking, Reset, and Power-Management Scheme

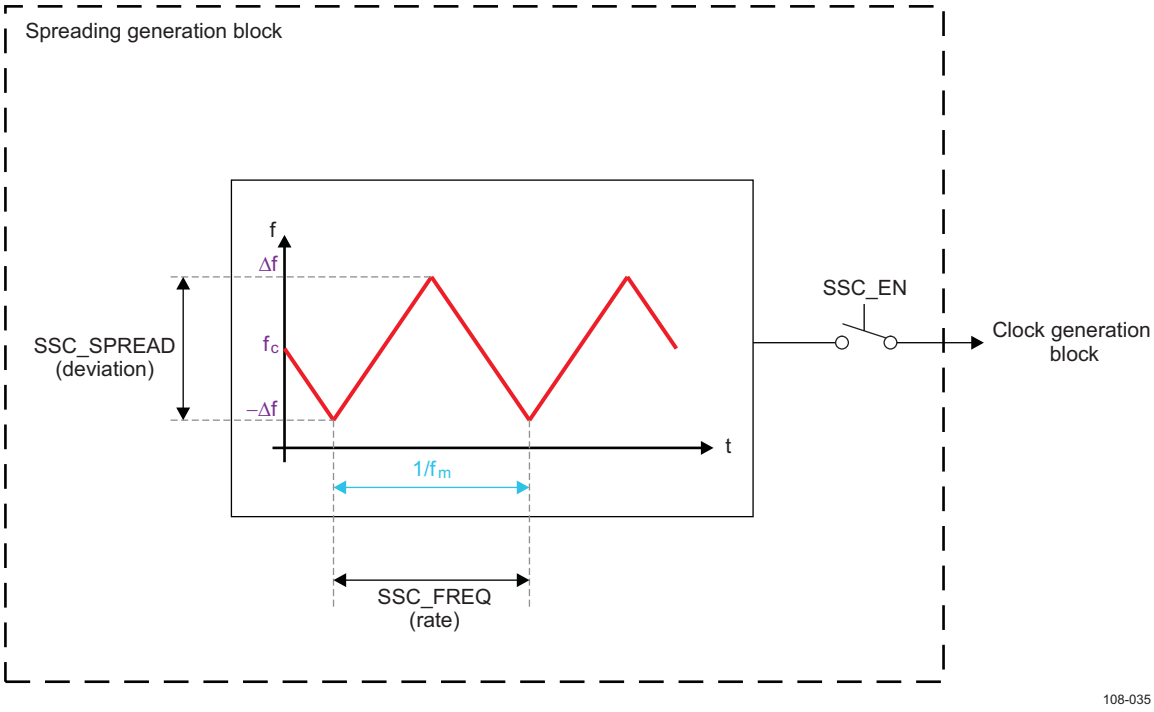
See Chapter 4, *Power, Reset, and Clock Management*.

7.4.11.3 Functional Description

7.4.11.3.1 Spreading Generation Block

Figure 7-20 is a diagram of the spreading generation block.

Figure 7-20. Spreading Generation Block Diagram



Note: Δf is the deviation from the center frequency. The total spreading deviation is equal to twice Δf.

f_c is the original clock frequency.

f_m is the spreading frequency.

This additional block generates the required waveform used to reduce EMI. This waveform is then modulated with the initial signal ito add some controlled deviation in the clock signal frequency, which spreads the energy of the clock and its harmonics into a band of frequencies, and then reduces the electomagnetic interferences.

The SSC_SPREAD value handles the deviation (amplitude) of the generated signal from the original signal. It is controlled by the CONTROL.CONTROL_XXX_DPLL_SPREADING[3:2] XXX_SPREADING_AMPLITUDE bit field of the corresponding registers (where XXX is the name of the concerned DPLLs). This register contains only a spreading ratio K, which is equal to Δf / f_m.

The SSC_FREQ value controls the rate of the generated signal. It is controlled by the CONTROL.CONTROL_XXX_DPLL_SPREADING[1:0] XXX_SPREADING_RATE bit field of the corresponding registers (where XXX is the name of the concerned DPLLs).

The SSC_EN signal enables/disables the frequency modulation feature of the DPLL. It is controlled by the CONTROL.CONTROL_XXX_DPLL_SPREADING[4] XXX_SPREADING_ENABLE bit field of the corresponding registers (where XXX is the name of the concerned DPLLs).

In any case, the frequency modulation is programmed in the SCM and does not generally need to be changed in real time.

7.4.11.3.2 Spread Spectrum Clocking

7.4.11.3.2.1 Definition

The aim of SSC is to add a variation in the frequency of an original clock, which spreads the generated interferences over a larger band of frequency.

In theory, SSC means that the clock signal is varied around the desired frequency. For example, for a 1-GHz clock, the frequency might be 999.5 Mhz at one moment and 1.0005 GHz at another. Doing this constantly causes the power of the tone to be spread out more over a broader band of tight frequencies (centered at the desired tone). To realize this constant variation on the original signal, a modulation with an additional signal (called spreading waveform) is realized.

Creating an SSC by spreading the initial clock frequency is done by defining the following parameters:

- The spreading frequency (deviation), which is the ratio of the range of spreading frequency over the original clock frequency
- The modulation rate (f_m), which is used to determine the clock-frequency spreading-cycling rate and is the time during which the generated clock frequency varies through Δf and returns to the original frequency
- The modulation waveform, which describes the variation curve in terms of time

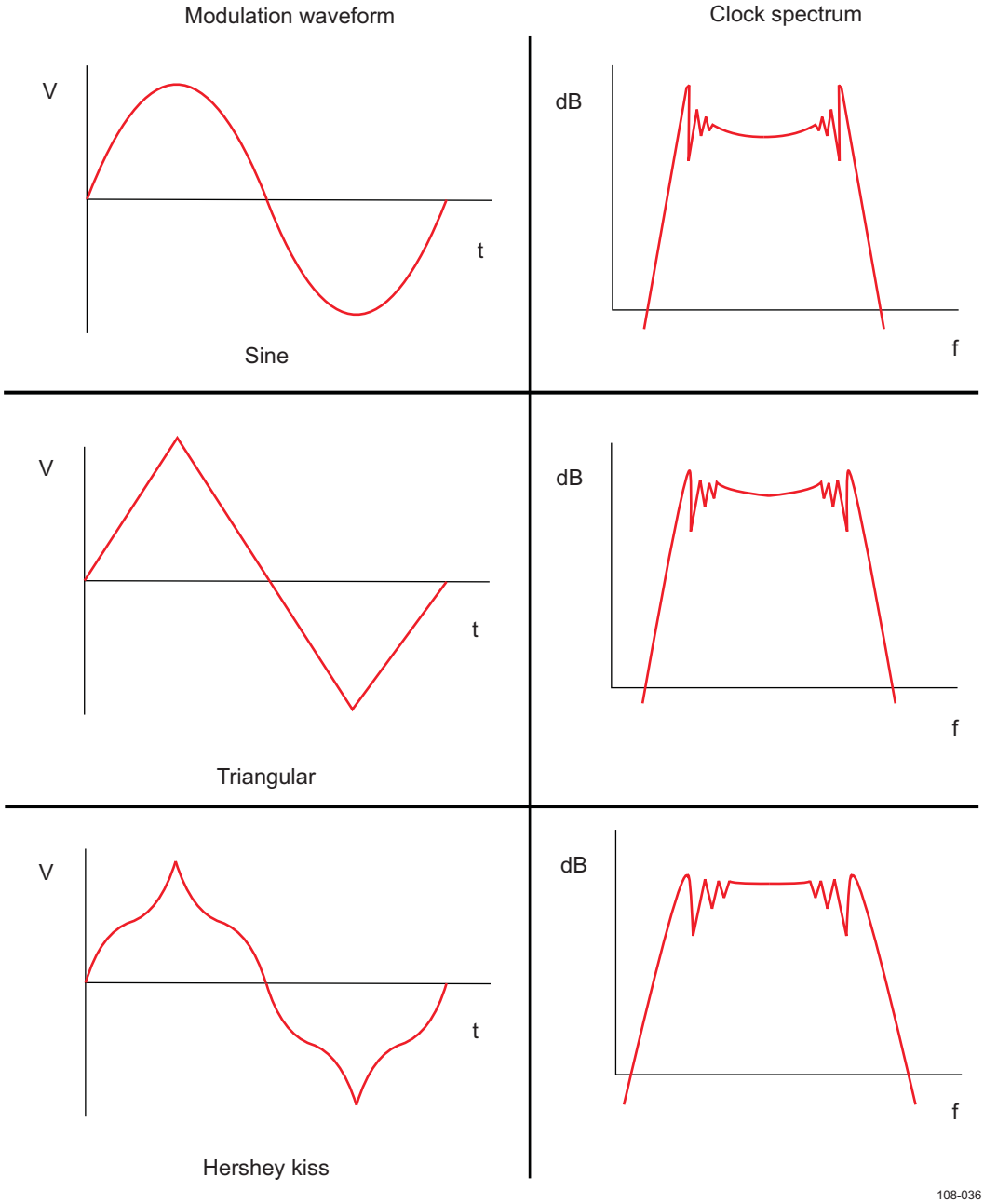
The spectral power reduction in the DPLL clocks is dependent on the modulation index (K), which is a ratio of spreading frequency calculated from the frequency deviation (Δf) and the modulation rate (f_m) .

7.4.11.3.2.2 The Modulation Waveforms

The shape (profile) of the generated clock signal depends on the modulation waveform that is used during the frequency modulation. Several profiles can be used, according to the desired shaping for the energy spreading.

Figure 7-21 shows three exemples of modulation waveforms and the spectrum of the corresponding modulated clock signal.

Figure 7-21. Modulation Profiles



The triangular wave gives a relatively flat spectrum and is also easy to generate. The triangular waveform is used by the DPLL-D.

7.4.11.3.2.3 Effect on the Clock Signal

Figure 7-22 gives an example of the effect of a triangular spreading on a clock signal.

Figure 7-22. Effect of the SSC in Frequency

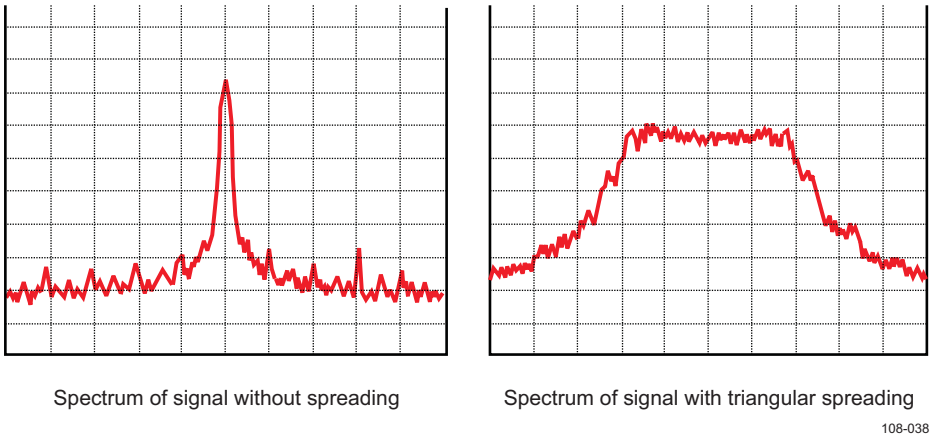
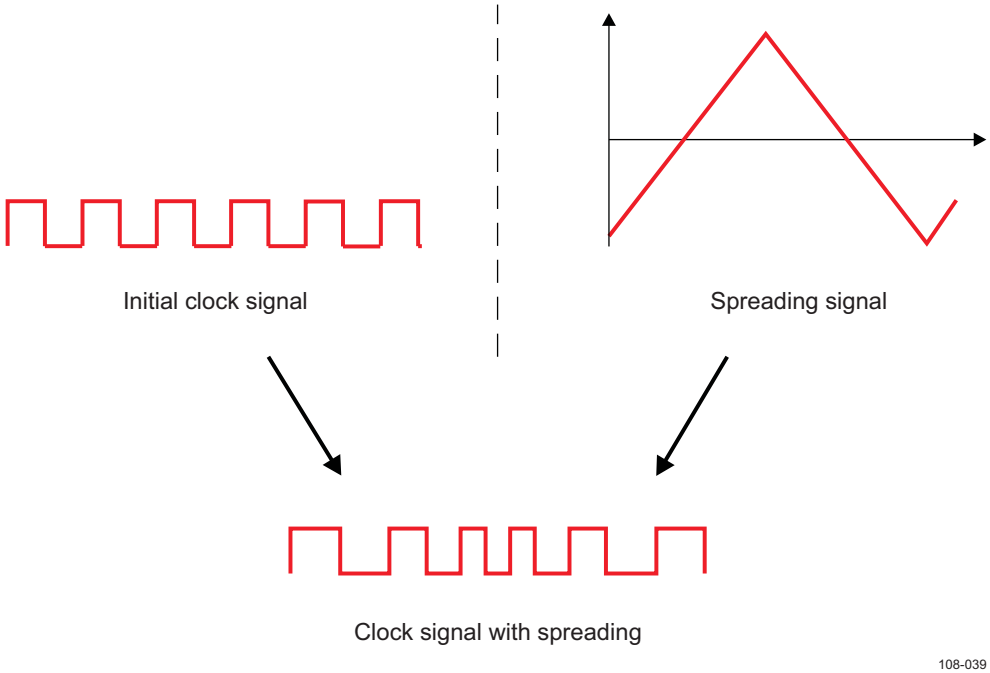


Figure 7-22 shows not only the power reduction of the main peak, but also the flatter aspect of the modulated signal. The minimum level of the second signal is higher than the minimum level of the first signal. This effect is normal and is due to the "noise" added for the modulation.

Note: The spreading technique "scatters" the energy of the peaks on the other frequencies, which reduces the power of the peaks but increases the global "noise" of the signal.

Figure 7-23 shows the effect of triangular spreading on a clock signal in the time domain.

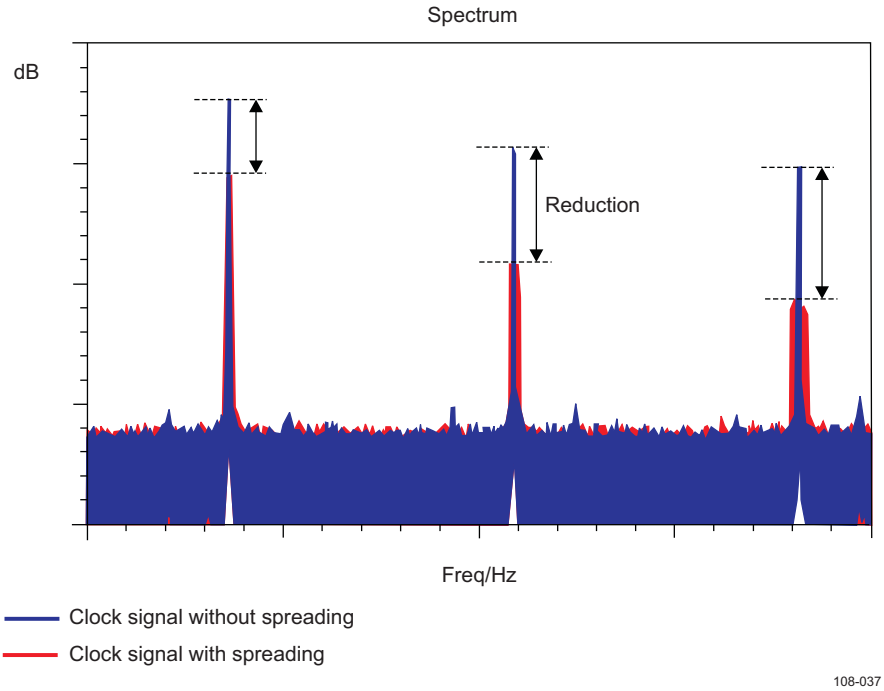
Figure 7-23. Effect of the SSC in the Time Domain



7.4.11.3.2.4 Estimation of the EMI Reduction Level

Figure 7-24 shows the effect of spreading on a clock and its harmonics.

Figure 7-24. Peak Reduction Caused by Spreading



The electromagnetic interference reduction can be estimated with the following equation:

(1) Peak_power_reduction = 10 * log ((Deviation * f_c) / f_m)

With:

- Peak_power_reduction in dB
- Deviation in % of the initial clock frequecy (f_c), equals $\Delta f / f_c$
- f_c is the original clock frequency, in MHz
- f_m is the spreading frequency, in MHz

According to equation (1), it is also possible to compute the deviation, and then Δf, for a required peak power reduction:

(2) Deviation = (f_m / f_c) * $10^{(Peak_power_reduction\ /\ 10)}$

Example:

For f_c=400 MHz, deviation =1% peak from f_c(Δf = 4MHz) and f_m=400kHz; the estimated peak power reduction is 10dB.

7.4.11.3.2.5 Bandwidth Calculation (Carson Bandwidth Rule)

The Carson bandwidth rule defines the approximate bandwidth requirements of communications system components for a carrier signal that is frequency-modulated by a continuous or broad spectrum of frequencies rather than a single frequency.

The Carson bandwidth rule is expressed by the relation CBR = 2 * (Δf + f_m), where CBR is the bandwidth requirement, Δf is the peak frequency deviation, and f_m is the highest frequency in the modulating signal.

For example, an FM signal with a 5-kHz peak deviation and a maximum audio frequency of 3 kHz, would require an approximate bandwidth 2*(5+3) = 16 kHz.

Theoretically, any FM signal has an infinite number of sidebands and hence an infinite bandwidth, but in practice all significant sideband energy (98% or more) is concentrated within the bandwidth defined by the Carson bandwidth rule.

7.4.11.3.3 Frequency Limitations

There are some limitations on the use of the spreading feature. Figure 7-25 shows the supported spreading frequency and deviation.

Figure 7-25. Supported Spreading Frequency and Deviation

		SPREADING_AMPLITUDE (bits [3:2])			
		00	01	10	11
		K			
		4	6	8	10
SPREADING_RATE (bits [1:0])	Min to Max rate (kHz)	Frequency deviation +/- kHz			
00	62.5 to 125	250 to 500	375 to 750	500 to 1000	625 to 1250
01	125 to 250	500 to 1000	750 to 1500	1000 to 2000	1250 to 2500
10	250 to 500	1000 to 2000	1500 to 3000	2000 to 4000	2500 to 5000
11	500 to 1000	2000 to 4000	3000 to 6000	4000 to 8000	5000 to 10000

Not recommended. PLL jitter degradation and modulation amplitude not a linear function of FM.

108-040

Even if all the previous frequencies and deviations are supported by the DPLL-D, a range of "safe operating regions" is defined according to its impact on jitter.

Figure 7-25 shows the supported safe operating regions supported by the DPLL-D.

Figure 7-26. Supported Safe Operating Regions and Jitter Impact

CLKOUT FREQUENCY (25–900 MHz)	Recommended safe operating region	Max jitter impact in % due to spreading
400–900		+/– 1.25%
200–399		+/– 1.25%
100–199		+/– 1.25%
50–99		+/– 1.00%
25–49		+/– 2.00%

108-041

(1) See Figure 7-25 for the color coding.

7.4.11.4 Basic Programming Model

7.4.11.4.1 SSC Configuration

The configuration of the spreading feature is not mandatory when programming the DPLL. This feature is usually enabled when the DPLL clocks generate harmonics that can potentially interfere with the GSM carrier frequencies.

Once the "clock generation control" registers are configured, it is possible to configure the spreading on the clock signal.

The first thing to do is to calculate the deviation and/or the spreading rate according to the desired peak power reduction. In addition to this, it is necessary to have K (modulation index), which value is equal to $\Delta f / f_m$.

Then, the XXX_SPREADING_AMPLITUDE and XXX_SPREADING_RATE bit fields can be configured with the calculated values.

Finally, the spreading must be enabled using the XXX_SPREADING_ENABLE bit.

Note: It is necessary to configure the spreading on a clock carefully to avoid adding some noise on frequencies that are used by another module. For example, adding spreading on a clock to reduce noise on GSM frequencies can "move" the generated noise to the frequency of the memory controller and degrade its performance.

Example:

For f_c=400 MHz, deviation =1% peak from f_c (Δf = 4 MHz) and f_m=400 kHz; the estimated peak power reduction is 10dB. K is then equal to 10. The DPLL can be configured as follows:

- XXX_SPREADING_RATE = 10 (250 to 500 KHz)
- XXX_SPREADING_AMPLITUDE = 11 (K = 10)
- XXX_SPREADING_ENABLE = 1

The state of the modulation feature can be monitored with the XXX_SPREADING_ENABLE_STATUS bit of the corresponding register.

Notes:

- To support the spreading feature, the DPLL bandwidth is restricted to a maximum of 70 kHz. This implies the reference clock frequency to be in range 0.75 MHz to 2.1 MHz, and the XXX_DPLL_FREQSEL[3:0] bit field to be in range of 0011 to 0111.
- The required Δf (frequency deviation) is targeted on the CLKOUT frequency and is achieved when M2 is programmed to 1. Otherwise, the frequency deviation is scaled down by the factor of the M2 value. However, this does not affect the deviation; therefore, the frequency change of harmonics at GSM frequencies is the same.
- The spread of modulation frequency within each range is due to an internal auto-ranging function and specific frequencies cannot be selected.
- The lowest range (62.5 to 125 kHz) must only be selected when the reference clock (PLL internal reference frequency) < 1.1 MHz is being used, to prevent the PLL feedback loop from canceling the modulation.

When deactivating the spreading (XXX_SPREADING_ENABLE = 0), the end-of-spreading is synchronous to the internal spreading cycle. Therefore, there is no residual average frequency error.

7.5 SCM Programming Model

7.5.1 Feature Settings

7.5.1.1 Force Pad Configuration MuxMode by High-Speed USB

The two pads hsusb0_data0 and hsusb0_data1 can force pad configuration MuxMode when the CARKITEN signal is generated:

- CARKITHSUSB0DATA0AUTOEN bit CONTROL.CONTROL_DEVCONF1[19]:
  0: The hsusb0_data0 MuxMode signal is driven by MUXMODE1 bit CONTROL.CONTROL_PADCONF_HSUSB0_NXT[18:16].
  1: The hsusb0_data0 MuxMode signal is forced to 0x2 when the CARKITEN signal is active.
- CARKITHSUSB0DATA1AUTOEN bit CONTROL.CONTROL_DEVCONF1[20]:
  0: The hsusb0_data1 MuxMode signal is driven by MUXMODE0 bit CONTROL.CONTROL_PADCONF_HSUSB0_DATA[2:0].
  1: The hsusb0_data1 MuxMode signal is forced to 0x2 when the CARKITEN signal is active.

Note: Associated pad configuration registers remain unchanged.

For more details on High-Speed USB, see Chapter 24, *High-Speed USB Controllers*.

7.5.1.2 Video Driver

This section gives information about all modules and features in the high-tier device. See Section 1.5, *OMAP34xx Family*, to check availability of modules and features. Unavailable module and feature pins are not functional.

- TVOUTBYPASS bit CONTROL.CONTROL_DEVCONF1[18]:
  0b0: Dual 10-bit video DAC TV out bypass disable
  0b1: Dual 10-bit video DAC TV out bypass enable
- TVACEN bit CONTROL.CONTROL_DEVCONF1[11]:
  0b0: Enables dc coupling for TV output
  0b1: Enables ac coupling for TV output

For more details on video DACs, see Chapter 15, *Display Subsystem*.

7.5.1.3 McBSP1 Internal Clock

The McBSP1 internal clock gates the internal interconnect clock and selects the FSR, CLKR, and CLKS input for the McBSP1.

- MCBSP1_FSR bit CONTROL.CONTROL_DEVCONF0[4]:
  0: FSR is from the pin mcbsp1_fsr.
  1: FSR is from the pin mcbsp1_fsx.
- MCBSP1_CLKR bit CONTROL.CONTROL_DEVCONF0[3]:
  0: CLKR is from the pin mcbsp1_clkr.
  1: CLKR is from the pin mcbsp1_clkx.
- MCBSP1_CLKS bit CONTROL.CONTROL_DEVCONF0[2]:
  0: CLKS is from the PRCM functional clock.
  1: CLKS is from the pin mcbsp_clks.

For more details on McBSP, see Chapter 21, *McBSP*.

7.5.1.4 McBSP2 Internal Clock

The McBSP2 internal clock gates the internal interconnect clock and selects the FSR, CLKR, and CLKS input for the McBSP2. The McBSP2 does not have mcbsp2_clkr and mcbsp2_fsr external pins. Clock input is from the mcbsp2_clkx pin; FSR input is from the mcbsp2_fsx pin.

- MCBSP2_CLKS register bit CONTROL.CONTROL_DEVCONF0[6]:
  0: Clock is from the PRCM functional clock.
  1: Clock is from the external pin mcbsp_clks.

For more details on McBSP, see Chapter 21, *McBSP*.

7.5.1.5 McBSP3 Internal Clock

The McBSP3 internal clock gates the internal interconnect clock and selects the FSR, CLKR, and CLKS input for the McBSP3. The McBSP3 does not have mcbsp3_clkr and mcbsp3_fsr external pins. Clock input is from the mcbsp3_clkx pin; FSR input is from the mcbsp3_fsx pin.

- MCBSP3_CLKS register bit CONTROL.CONTROL_DEVCONF1[0]:
  0: Clock is from the PRCM functional clock.
  1: Clock is from the external pin mcbsp_clks.

For more details on McBSP, see Chapter 21, *McBSP*.

7.5.1.6 McBSP4 Internal Clock

The McBSP4 internal clock gates the internal interconnect clock and selects the FSR, CLKR, and CLKS input for the McBSP4. The McBSP4 does not have mcbsp4_clkr and mcbsp4_fsr external pins. Clock input is from the mcbsp4_clkx pin; FSR input is from the mcbsp4_fsx pin.

- MCBSP4_CLKS register bit CONTROL.CONTROL_DEVCONF1[2]:
  0: Clock is from the PRCM functional clock.
  1: Clock is from the external pin mcbsp_clks.

For more details on McBSP, see Chapter 21, *McBSP*.

7.5.1.7 McBSP5 Internal Clock

The McBSP5 internal clock gates the internal interconnect clock and selects the FSR, CLKR, and CLKS input for the McBSP5. The McBSP5 does not have mcbsp5_clkr and mcbsp5_fsr external pins. Clock input is from the mcbsp5_clkx pin; FSR input is from the mcbsp5_fsx pin.

- MCBSP5_CLKS register bit CONTROL.CONTROL_DEVCONF1[4]:
  0: Clock is from the PRCM functional clock.
  1: Clock is from the external pin mcbsp_clks.

For more details on McBSP, see Chapter 21, *McBSP*.

7.5.1.8 MMC/SD/SDIO1 Module Input Clock Selection

- MMCSDIO1ADPCLKISEL bit CONTROL.CONTROL_DEVCONF0[24]:
  0: Input clock is from the external pin.
  1: Internal loopback, module input clock is copied from the module output clock.

7.5.1.9 MMC/SD/SDIO2 Module Input Clock Selection

- MMCSDIO2ADPCLKISEL bit CONTROL.CONTROL_DEVCONF1[6]:
  0: Input clock is from the external pin.
  1: Internal loopback, module input clock is copied from the module output clock.

7.5.1.10 Setting Sensitivity on sys_ndmareq[6:0] Input Pins

The seven sys_ndmareq0 to sys_ndmareq6 input pins can be either level or edge sensitive.

- SENSDMAREQ0 bit CONTROL.CONTROL_DEVCONF0[0]:
  0: Level sensitivity
  1: Edge sensitivity
- SENSDMAREQ1 bit CONTROL.CONTROL_DEVCONF0[1]:
  0: Level sensitivity
  1: Edge sensitivity
- SENSDMAREQ2 bit CONTROL.CONTROL_DEVCONF1[7]:
  0: Level sensitivity
  1: Edge sensitivity
- SENSDMAREQ3 bit CONTROL.CONTROL_DEVCONF1[8]:
  0: Level sensitivity
  1: Edge sensitivity
- SENSDMAREQ4 bit CONTROL.CONTROL_DEVCONF1[21]:
  0: Level sensitivity
  1: Edge sensitivity
- SENSDMAREQ5 bit CONTROL.CONTROL_DEVCONF1[22]:
  0: Level sensitivity
  1: Edge sensitivity
- SENSDMAREQ6 bit CONTROL.CONTROL_DEVCONF1[23]:
  0: Level sensitivity
  1: Edge sensitivity

For more details on DMA, see Chapter 9, *DMA*.

7.5.1.11 I^2C I/O Internal Pull-up Enable

The I/O internal pull-up of I2C1, I2C2, and I2C3 can be enabled:

- I2C1HSMASTER bit CONTROL.CONTROL_DEVCONF1[12]:
  0: I2C1 I/O internal pull-up disable
  1: I2C1 I/O internal pull-up enable
- I2C2HSMASTER bit CONTROL.CONTROL_DEVCONF1[13]:
  0: I2C2 I/O internal pull-up disable
  1: I2C2 I/O internal pull-up enable
- I2C3HSMASTER bit CONTROL.CONTROL_DEVCONF1[14]:
  0: I2C3 I/O internal pull-up disable
  1: I2C3 I/O internal pull-up enable

Note: This feature is used for the master component.

For more details on I^2C, see Chapter 18, *I^2C*.

7.5.1.12 SDRC I/O Drive Strength Selection

- SDRC_LOWDATA bit CONTROL.CONTROL_PROG_IO0[31] selects data[15:0] DQS0, DQS1, DM0, and DM1 I/O drive strength:
  0: Load range = [2 pF - 6 pF]
  1: Load range = [6 pF - 12 pF]
- SDRC_HIGHDATA bit CONTROL.CONTROL_PROG_IO0[30] selects data[31:16] DQS2, DQS3, DM2, and DM3 I/O drive strength:

0: Load range = [2 pF - 6 pF]
1: Load range = [6 pF - 12 pF]

- SDRC_ADDRCRTL bit CONTROL.CONTROL_PROG_IO0[29] selects address NRAS, NCAS, NWE, NCLK, CLK, BA0, and BA1 I/O drive strength:
  0: Load range = [2 pF - 6 pF]
  1: Load range = [6 pF - 12 pF]
- SDRC_NCS0 bit CONTROL_PROG_IO0[28] selects NCS0 and CKE0 I/O drive strength
- SDRC_NCS1 bit CONTROL_PROG_IO0[27] selects NCS1 and CKE1 I/O drive strength

7.5.1.13 GPMC I/O Drive Strength Selection

- GPMC_A1 bit CONTROL.CONTROL_PROG_IO0[26] selects GPMC_A1 I/O drive strength.
  0: Load range = [2 pF - 6 pF]
  1: Load range = [6 pF - 12 pF]
- GPMC_A2 bit CONTROL.CONTROL_PROG_IO0[25] selects GPMC_A2 I/O drive strength.
  0: Load range = [2 pF - 6 pF]
  1: Load range = [6 pF - 12 pF]
- GPMC_A3 bit CONTROL.CONTROL_PROG_IO0[24] selects GPMC_A3 I/O drive strength.
  0: Load range = [2 pF - 6 pF]
  1: Load range = [6 pF - 12 pF]
- GPMC_A4 bit CONTROL.CONTROL_PROG_IO0[23] selects GPMC_A4 I/O drive strength.
  0: Load range = [2 pF - 6 pF]
  1: Load range = [6 pF - 12 pF]
- GPMC_A5 bit CONTROL.CONTROL_PROG_IO0[22] selects GPMC_A5 I/O drive strength.
  0: Load range = [2 pF - 6 pF]
  1: Load range = [6 pF - 12 pF]
- GPMC_A6 bit CONTROL.CONTROL_PROG_IO0[21] selects GPMC_A6 I/O drive strength.
  0: Load range = [2 pF - 6 pF]
  1: Load range = [6 pF - 12 pF]
- GPMC_A7 bit CONTROL.CONTROL_PROG_IO0[20] selects GPMC_A7 I/O drive strength.
  0: Load range = [2 pF - 6 pF]
  1: Load range = [6 pF - 12 pF]
- GPMC_A8 bit CONTROL.CONTROL_PROG_IO0[19] selects GPMC_A8 I/O drive strength.
  0: Load range = [2 pF - 6 pF]
  1: Load range = [6 pF - 12 pF]
- GPMC_A9 bit CONTROL.CONTROL_PROG_IO0[18] selects GPMC_A9 I/O drive strength.
  0: Load range = [2 pF - 6 pF]
  1: Load range = [6 pF - 12 pF]
- GPMC_A10 bit CONTROL.CONTROL_PROG_IO0[17] selects GPMC_A10 I/O drive strength.
  0: Load range = [2 pF - 6 pF]
  1: Load range = [6 pF - 12 pF]
- GPMC_MIN_CFG bit CONTROL.CONTROL_PROG_IO0[16] selects data[7:0] I/O drive strength.
  0: Load range = [2 pF - 6 pF]
  1: Load range = [6 pF - 12 pF]
- GPMC_D8_D15 bit CONTROL.CONTROL_PROG_IO0[15] selects data[15:8], NOE, NWE, NADV_ALE and WAIT0 I/O drive strength.
  0: Load range = [2 pF - 6 pF]
  1: Load range = [6 pF - 12 pF]
- GPMC_NCS0 bit CONTROL.CONTROL_PROG_IO0[14] selects GPMC_NCS0 I/O drive strength.

0: Load range = [2 pF - 6 pF]
1: Load range = [6 pF - 12 pF]

- GPMC_NCS1 bit CONTROL.CONTROL_PROG_IO0[13] selects GPMC_NCS1 I/O drive strength.
  0: Load range = [2 pF - 6 pF]
  1: Load range = [6 pF - 12 pF]
- GPMC_NCS2 bit CONTROL.CONTROL_PROG_IO0[12] selects GPMC_NCS2 I/O drive strength.
  0: Load range = [2 pF - 6 pF]
  1: Load range = [6 pF - 12 pF]
- GPMC_NCS3 bit CONTROL.CONTROL_PROG_IO0[11] selects GPMC_NCS3 I/O drive strength.
  0: Load range = [2 pF - 6 pF]
  1: Load range = [6 pF - 12 pF]
- GPMC_NCS4 bit CONTROL.CONTROL_PROG_IO0[10] selects GPMC_NCS4 I/O drive strength.
  0: Load range = [2 pF - 6 pF]
  1: Load range = [6 pF - 12 pF]
- GPMC_NCS5 bit CONTROL.CONTROL_PROG_IO0[9] selects GPMC_NCS5 I/O drive strength.
  0: Load range = [2 pF - 6 pF]
  1: Load range = [6 pF - 12 pF]
- GPMC_NCS6 bit CONTROL.CONTROL_PROG_IO0[8] selects GPMC_NCS6 I/O drive strength.
  0: Load range = [2 pF - 6 pF]
  1: Load range = [6 pF - 12 pF]
- GPMC_NCS7 bit CONTROL.CONTROL_PROG_IO0[7] selects GPMC_NCS7 I/O drive strength.
  0: Load range = [2 pF - 6 pF]
  1: Load range = [6 pF - 12 pF]
- GPMC_CLK bit CONTROL.CONTROL_PROG_IO0[6] selects GPMC_CLK I/O drive strength.
  0: Load range = [2 pF - 6 pF]
  1: Load range = [6 pF - 12 pF]
- GPMC_NBE0_CLE bit CONTROL.CONTROL_PROG_IO0[5] selects GPMC_NBE0_CLE I/O drive strength.
  0: Load range = [2 pF - 6 pF]
  1: Load range = [6 pF - 12 pF]
- GPMC_NBE1 bit CONTROL.CONTROL_PROG_IO0[4] selects GPMC_NBE1 I/O drive strength.
  0: Load range = [2 pF - 6 pF]
  1: Load range = [6 pF - 12 pF]
- GPMC_NWP bit CONTROL.CONTROL_PROG_IO0[3] selects GPMC_NWP I/O drive strength.
  0: Load range = [2 pF - 6 pF]
  1: Load range = [6 pF - 12 pF]
- GPMC_WAIT1 bit CONTROL.CONTROL_PROG_IO0[2] selects GPMC_WAIT1 I/O drive strength.
  0: Load range = [2 pF - 6 pF]
  1: Load range = [6 pF - 12 pF]
- GPMC_WAIT2 bit CONTROL.CONTROL_PROG_IO0[1] selects GPMC_WAIT2 I/O drive strength.
  0: Load range = [2 pF - 6 pF]
  1: Load range = [6 pF - 12 pF]
- GPMC_WAIT3 bit CONTROL.CONTROL_PROG_IO0[0] selects GPMC_WAIT3 I/O drive strength.
  0: Load range = [2 pF - 6 pF]
  1: Load range = [6 pF - 12 pF]

7.5.1.14 MCBSP2 I/O Drive Strength Selection

- MCBSP2_MIN_CTL field CONTROL.CONTROL_PROG_IO1[7:6] selects FSX, CLKX, DR and DX I/O drive strength.
  0: 8 mA
  1: 4 mA
  2: 6 mA
  3: 2 mA

7.5.1.15 MCSPI1 I/O Drive Strength Selection

- MCSPI1_MIN_CTL field CONTROL.CONTROL_PROG_IO1[5:4] selects CLK, SOMI, SIMO and CS0 I/O drive strength.
  0: 8 mA
  1: 4 mA
  2: 6 mA
  3: 2 mA
- MCSPI1_CS1 field CONTROL.CONTROL_PROG_IO1[3:2] selects CS1 I/O drive strength.
  0: 8 mA
  1: 4 mA
  2: 6 mA
  3: 2 mA
- MCSPI1_CS2 field CONTROL.CONTROL_PROG_IO1[1:0] selects CS2 I/O drive strength.
  0: 8 mA
  1: 4 mA
  2: 6 mA
  3: 2 mA

7.5.1.16 Force MPU Writes to Be Nonposted

The MPUFORCEWRNP bit CONTROL.CONTROL_DEVCONF1[9] bit is for debugging only. When this bit is set to 1 (for debugging), all writes are forced to nonposted and the cache attributes are ignored. By default, this bit should be set to 0 (posted writes). For the best performance, keep this bit at 0.

7.5.2 Extended-Drain I/Os and PBIAS Cells Programming Guide

Note: The device can be supplied by an external power IC. Texas Instruments provides such a global solution to its customers with the TWL4030 or TWL4040 power IC.

If the device is associated with the TWL4030 or TWL4040 power IC, before using the MMC/SD/SDIO1 interface, the software must program the TWL4030 or TWL4040 to enable the VMMC1 or VSIM (for the MMC/SD/SDIO1 module or muxed GPIO I/Os) and the LDO and to provide a 1.8-V/3.0-V voltage. This is done by software through the I^2C interface that links the device and TWL4030 or TWL4040 IC.

If the application does not want the selected interface running at 3.0 V, software users must then assert the VMODE signal to low for 1.8-V activity: in this case, the PBIAS is connected to ground. (See Figure 7-27.)

CAUTION

The power supply voltage generated from the companion power IC and applied on the vdds_mmc1 or vdds_sim device pins must be programmed according to the recommendations for the power supply range of the selected I/O module.

Table 7-59 lists the control signal with the corresponding control bits from the CONTROL.CONTROL_PBIAS_LITE register to configure the PBIAS and the extended-drain I/O cells. These signals can be software-controlled.

Table 7-59. Control Signals

Control Signal	Bit for MMC/SD/SDIO1 Module Using PBIAS0 Cell	Bit for MMC/SD/SDIO1 Module Using PBIAS1 Cell	Reset Value
PWRDNZ	CONTROL.CONTROL_PBIAS_LITE[1] PBIASLITEPWRDNZ0	CONTROL.CONTROL_PBIAS_LITE[9] PBIASLITEPWRDNZ1	0
VMODE	CONTROL.CONTROL_PBIAS_LITE[0] PBIASLITEVMODE0	CONTROL.CONTROL_PBIAS_LITE[8] PBIASLITEVMODE1	1
SUPPLY_HIGH	CONTROL.CONTROL_PBIAS_LITE[7] PBIASLITESUPPLYHIGH0	CONTROL.CONTROL_PBIAS_LITE[15] PBIASLITESUPPLYHIGH1	0
SPEEDCTRL	CONTROL.CONTROL_PBIAS_LITE[2] PBIASSPEEDCTRL0	CONTROL.CONTROL_PBIAS_LITE[10] PBIASSPEEDCTRL1	0
VMODEERROR	CONTROL.CONTROL_PBIAS_LITE[3] PBIASLITEVMODEERROR0	CONTROL.CONTROL_PBIAS_LITE[11] PBIASLITEVMODEERROR1	0

Each PBIAS cell supports two ranges of I/O power supply: 1.8 V, typical for low-voltage applications, and 3.0 V, typical for high-voltage applications. For each supply voltage range, the cell generates suitable bias voltage (PBIAS) for extended-drain PMOS devices.

CAUTION

A PBIAS cell must be programmed according to peripheral power supply voltage. See Table 7-60.

Table 7-60. Voltage Configuration(1)

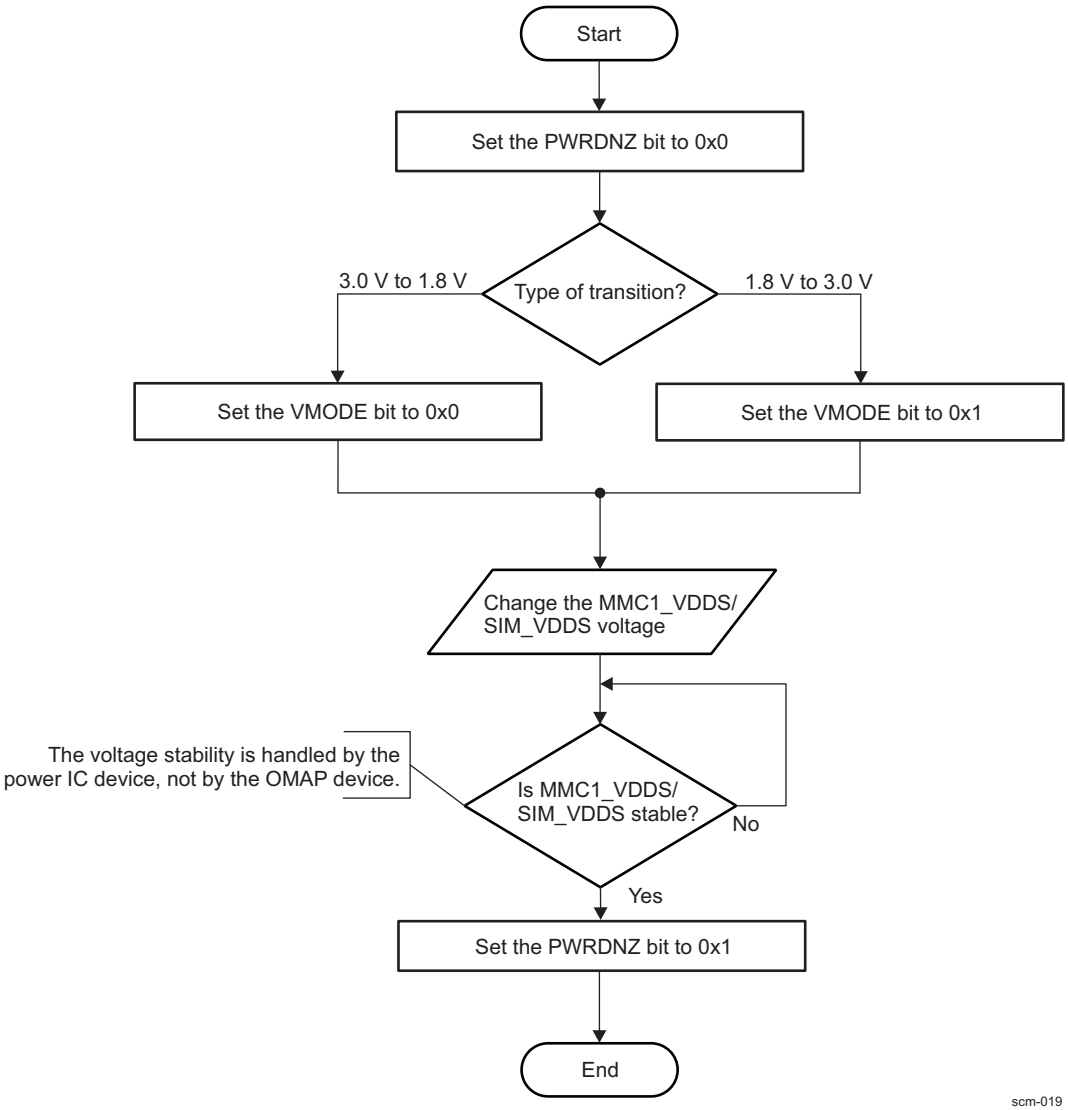
PBIASLITEVMODE Configuration	MMC1_VDDS/SIM_VDDS Voltage	Reset Value
1.8 V	1.8 V	Normal 1.8-V operation
1.8 V	3.0 V	Damaging configuration(2)
3.0 V	1.8 V	Degraded functionality(2)
3.0 V	3.0 V	Normal 3.0-V operation

(1) For damaging configuration, hardware system protection is provided to prevent deterioration of the associated extended-drain I/Os.

(2) It is forbidden to use these modes.

Figure 7-27 describes the programming flow to go from 3.0 V to 1.8 V, and vice versa.

Figure 7-27. Flow Chart



The PBIAS output is the same as MMC1_VDDS/SIM_VDDS when the corresponding PBIAS cell PWRDNZ bit is LOW. Once the MMC1_VDDS/SIM_VDDS supply settles, the software releases the PWRNDZ (pulls it HIGH). This then starts up the PBIAS cell work to generate the PBIAS voltage. During the complete process the MMC/SD/SDIO1 I/Os cannot be used for transmitting data.

Note: In the case of a damaging configuration, hardware system protection prevents deterioration of the associated extended-drain I/Os.

CAUTION

The following are critical requirements for the cell:

- The VMODE bit must be defined before the PWRNDZ bit is made HIGH (cell is brought out of PWRNDZ).
- The default state of VMODE bit must be HIGH (to indicate 3.0-V operation).
- PWRNDZ bit must be kept LOW when the MMC1_VDDS/SIM_VDDS supply is ramping up (PWRNDZ bit is not required to be kept LOW during ramp down of the supply). This could be damaging.

7.5.2.1 PBIAS Error Generation

Table 7-61 summarizes the generation of the PBIAS error interrupt (PBIAS0_ERROR and PBIAS1_ERROR) depending on the various CONTROL.CONTROL_PBIAS_LITE bits control. The shaded row is a potential condition that could cause a reliability issue if not detected. To prevent this reliability issue, the PBIAS voltage is kept at MMC1_VDDS/SIM_VDDS level when the PBIAS error signal is HIGH.

Table 7-61. PBIAS Error Signal Truth Table

VMODE	PWRDNZ	SUPPLY_HIGH	PBIAS ERROR
X	X	X	0
0	0	X	0
0	1	0	0
0	1	1	1
X	1	X	1
1	0	X	0
1	1	0	1
1	1	1	0

Note: PBIAS ERROR = 1: VMODE level not same as SUPPLY_HIGH

PBIAS ERROR = 0: VMODE level same or VMODE not considered

The PBIAS errors are merged and connected to the MPU subsystem interrupt controller.

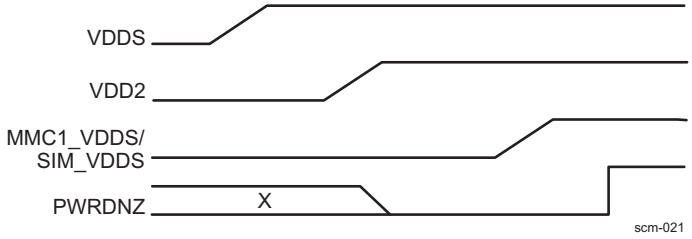
The CONTROL.CONTROL_PBIAS_LITE[i] VMODEERROR bits (with i = 3 or 11) also indicates if this kind of error occurs.

7.5.2.2 Critical Timing Requirements

It is crucial that the PBIAS and I/O cell-related PWRDNZ bits are deasserted (made 1 from 0) only after MMC1_VDDS/SIM_VDDS is stable. The device supports only the power-up sequence in which VDDS ramps up before VDD2. However, MMC1_VDDS/SIM_VDDS must come up after both VDD2 and VDDS.

Figure 7-28 shows the expected behavior of PWRNDZ bit with regard to supply ramp up. It also shows the case when VDDS ramps up before VDD2.

Figure 7-28. VDDS Ramps Up Before VDD2



Note: These timing requirements apply only when MMC1_VDDS/SIM_VDDS is 3.0 V. If MMC1_VDDS/SIM_VDDS is 1.8 V, VDDS and MMC1_VDDS/SIM_VDDS can be ramped up simultaneously.

7.5.2.3 Speed Control and Voltage Supply State

There are two other control bits in the CONTROL.CONTROL_PBIAS_LITE register:

- SPEEDCTRL bit can be used to reduce dynamic current if fast rise/fall times are not needed.
- SUPPLYHIGH bit signal is used to inform the cell on the value of the MMC1_VDDS/SIM_VDDS signal. (0b0 = 1.8 V and 0b1 = 3.0 V)

7.5.3 Off Mode Preliminary Settings

The following actions must be performed once, and remain valid for all device OFF <-> ON transitions:

- Program a valid device OFF pads configuration, by setting in each CONTROL.CONTROL_PADCONF_X registers all off mode values bits: OFFPULLTYPESELECT (OFF mode pull type), OFFPULLUDENABLE (OFF mode pull enabling), OFFOUTVALUE (OFF mode output value), OFFOUTENABLE (OFF mode output enabling), OFFENABLE (OFF mode pad state override control) .
- Program a valid device ACTIVE pads configuration by setting pertinent bits in each CONTROL.CONTROL_PADCONF_X register (see Section 7.5.4, *Pad Configuration Programming Points*).
- Perform a device ACTIVE pads configuration saving in the save and restore memory from Wake-up Control Module by asserting the STARTSAVE bit CONTROL.CONTROL_PADCONF_OFF[1].
- Set the SCM in smart idle mode by setting the IDLEMODE field CONTROL.CONTROL_SYSCONFIG[4:3] (this will ensure that its clocks can not be cut until the save procedure has completed).
- Enable/disable the wakup-up event detection capability of the pads by setting the WAKEUPENABLE bit CONTROL.CONTROL_PADCONF_X.

Note: When the wake-up detection is enabled for a pad, this pad is configured as input. Therefore, do not forget to write 0b1 in the OFFOUTENABLE bit CONTROL.CONTROL_PADCONF_X to disable the output capability.

For further information, see Chapter 4, *Power, Reset, and Clock Management*.

7.5.4 Pad Configuration Programming Points

To configure the pad, ensure that the following are done:

- Identify signals required on the interface based on the target application.

  Example: To configure the UART1 interface on balls, the required signals are uart1_tx, uart1_rts, uart1_cts, and uart1_rx.
- Choose the pads used for those signals. Some signals could be available on several pads and/or may be multiplexed with other signals needed for another application. See Section 7.4.4.3, *Pad Multiplexing*

Register Fields.

Example: Each UART1 interface signal is available on two pads. These signals are also multiplexed with a McBSP signal.

Assume that the McBSP interface is required in the system. Therefore, for the UART1 interface signals, pads must be used where those signals are not multiplexed with the McBSP signal.

- Identify the pad configuration registers associated with the pads to be used in the application. See Section 7.4.4.3, P*ad Multiplexing Register Fields*.

  Example: Under the previous hypothesis, the pad configuration registers to program are:

  – uart1_cts: CONTROL.CONTROL_PADCONF_DSS_DATA0[15:0]
  – uart1_rts: CONTROL.CONTROL_PADCONF_DSS_DATA0[31:16]
  – uart1_tx: CONTROL.CONTROL_PADCONF_DSS_DATA6[15:0]
  – uart1_rx: CONTROL.CONTROL_PADCONF_DSS_DATA6[31:16]

Note: In this configuration, dss_datan (where n = 0) signals are not available on these balls.

- Configure the MUXMODE field of each pad configuration register (CONTROL.CONTROL_PADCONF_X) associated with the pads used. Section 7.4.4.3, *Pad Multiplexing Register Fields*, lists the entire mode available for each pin. Write the binary value of the mode used in the MUXMODE field of the pad configuration registers.

  Example: UART1 signals are available in mode 2. Therefore, write 0b010 in the corresponding MUXMODE field of the pad configurations registers to be programmed.

- Configure the pull of the pad when used as input. When the pad is used as output, the pull is automatically disabled. Section 7.4.4.3, *Pad Multiplexing Register Fields*, lists the pull available on each pad and it reset value. Set the appropriate value for the PULLTYPESELECT bit of the pad configuration register to set the pull value (0b0 = Pull Down selected, 0b1 = Pull Up selected) and set the PULLUDENABLE bit of the pad configuration register to enable the pull on the pad (0b0 = pull disabled, 0b1 = pull enabled)

  Example: uart1_rts and uart1_tx are output signals; therefore, the pull is automatically disabled on the pad. Because uart1_cts and uart1_rx are input signals, configure the pull for these pads:

  – uart1_cts: Enable pullup (write 0b1 in the PULLTYPESELECT bit and 0b1 in the PULLUDENABLE bit of the corresponding pad configuration register)
  – uart1_rx: Enable pullup (write 0b1 in the PULLTYPESELECT bit and 0b1 in the PULLUDENABLE bit of the corresponding pad configuration register)

- Set the INPUTENABLE bit of the pad configuration register if the pin is used as input.

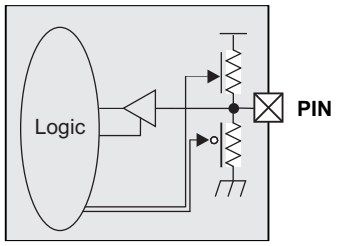
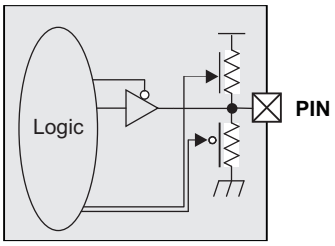
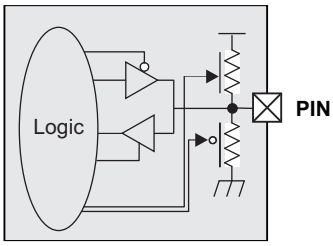
  Example: uart1_rts and uart1_tx are output signals; therefore, clear the INPUTENABLE bit of the corresponding pad configuration register. Because uart1_cts and uart1_rx are input signals, set the INPUTENABLE bit of the corresponding pad configuration register.

Note: The order for setting the previous pad configuration bits is not important.

7.5.5 I/O Power Optimization Guidelines

To optimize I/O power, it is important to avoid unconnected or incorrectly pulled pins. According to the type of pins, the way to reduce power consumption can differ. Figure 7-29 shows the three available pin (or ball) types.

Table 7-29. Pin Types

Input	Output	Bidirectional
		

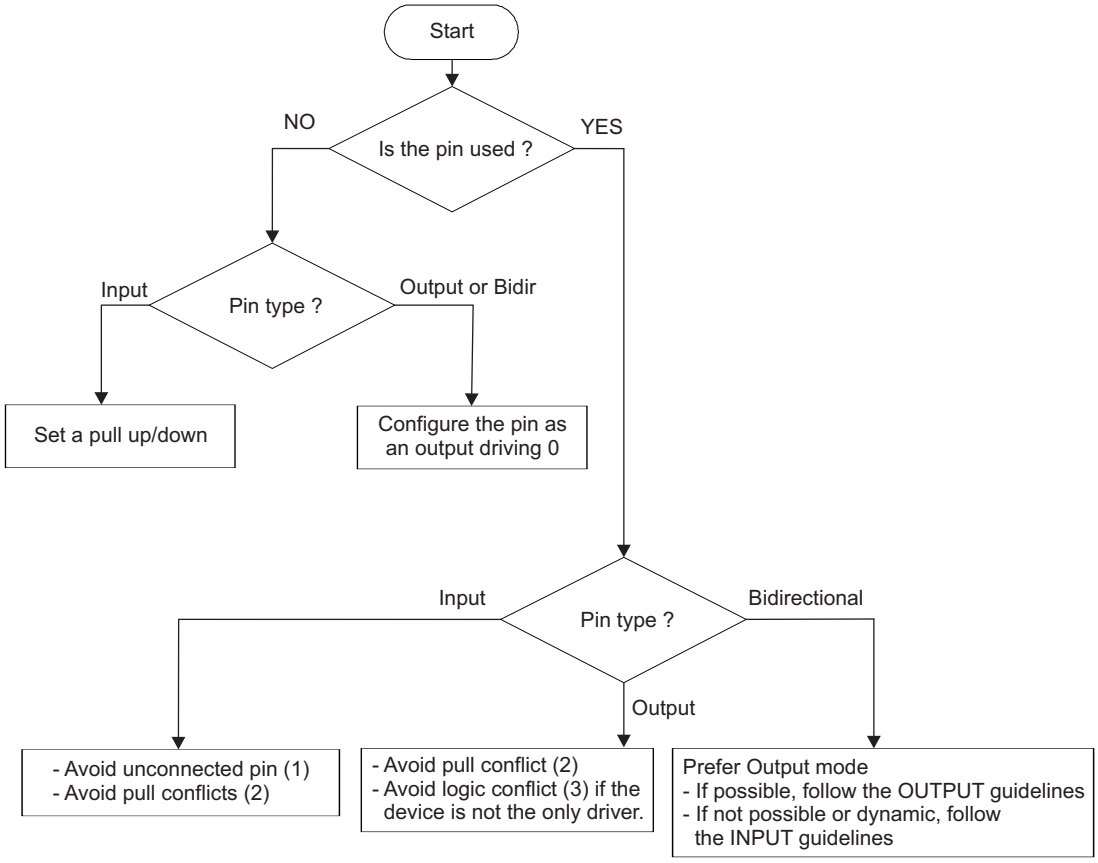
The configuration differs according to the I/O cell types. The following describes some pieces of advice which can be useful to avoid extra current leakage:

- For input pins, use a pull up/down when possible.
- For output pins, check existing pulls to avoid conflicts.
- For bidirectional pins, reconfigure the pin as an ouput driving 0 when possible.

Some I/O configurations involve modifications during the software setup of I/Os and sometimes require several hardware updates.

Figure 7-30 shows how to optimize the power consumption of pads.

Figure 7-30. I/O Power Optimization Flow Chart



The following notes give additional explanation about the pin configuration.

1. To avoid unconnected pins, the configuration depends on its use:
   - If the pin is not driven externally, a pull up/down is required.
   - Otherwise, a pull up/down is not necessary.
2. Pull conflicts occur when when there are different pulls on the same line. In order to correctly configure the pin, avoid external and internal pull together.
3. Logic conflicts consist in different electrical levels at the same time on one line. This can occur when several devices are connected to the same line. The two possible cases are:
   - If no external device drives the line, configure the pin to drive a '0'.
   - If another device drives the line, either the same value has to be driven or the pin has to be disconnected (HZ).

Note: It is advised to use high impedence logical state either on the device or the external component when the line is driven by both components.

The I/O pads are software-controlled by:

- Writing to the CONTROL.CONTROL_PADCONF_X registers in the Control Module for input/output and pull up/down configuration.
- Writing to the GPIOi.GPIO_OE registers in the GPI/O module for input/output configuration.

For more information about how to configure the I/O pads, see Chapter 7, *SCM*.

For more information about the GPI/O module, see Chapter 25, *General-Purpose Interface.*

Note: For a correct configuration of each pin direction (input, output, bidirectional), the CONTROL.CONTROL_PADCONF_X and the GPIOi.GPIO_OE registers must be written.

7.6 SCM Register Manual

This section gives information about all modules and features in the high-tier device. See Section 1.5, *OMAP34xx Family*, to check availability of modules and features. Unavailable module and feature pins are not functional.

7.6.1 SCM Instance Summary

Table 7-62 lists the base address and address space for the SCM instances.

Table 7-62. SCM Instance Summary

Module Name	Base Address	Size
INTERFACE	0x4800 2000	36 bytes
GENERAL	0x4800 2270	767 bytes
GENERAL_WKUP	0x4800 2A5C	31 bytes

A MEM_WKUP (1KB) instance exists; it based at 0x4800 2600. This instance is used in a save-and-restore context. See Table 7-66 for more details.

To configure pads, two instances exist:

- PADCONF (564 bytes) instance based at 0x4800 2030. See Table 7-64 for more details.
- PADCONFS_WKUP (80 bytes) instance based at 0x4800 2A00. See Table 7-67.

7.6.2 SCM Register Summary

Note: All module registers are 8-, 16-, or 32-bit accessible through the L4 interconnect (little endian encoding).

Table 7-63 lists the INTERFACE registers.

Table 7-63. INTERFACE Register Summary

Register Name	Type	Register Width (Bits)	Address Offset	Physical Address
CONTROL_REVISION	R	32	0x0000 0000	0x4800 2000
CONTROL_SYSCONFIG	RW	32	0x0000 0010	0x4800 2010

Table 7-64 lists the PADCONFS registers.

Table 7-64. PADCONFS Register Summary

Register Name	Type	Register Width (Bits)	Address Offset	Physical Address
CONTROL_PADCONF_SDRC_D0	RW	32	0x0000 0000	0x4800 2030
CONTROL_PADCONF_SDRC_D2	RW	32	0x0000 0004	0x4800 2034
CONTROL_PADCONF_SDRC_D4	RW	32	0x0000 0008	0x4800 2038
CONTROL_PADCONF_SDRC_D6	RW	32	0x0000 000C	0x4800 203C
CONTROL_PADCONF_SDRC_D8	RW	32	0x0000 0010	0x4800 2040
CONTROL_PADCONF_SDRC_D10	RW	32	0x0000 0014	0x4800 2044
CONTROL_PADCONF_SDRC_D12	RW	32	0x0000 0018	0x4800 2048
CONTROL_PADCONF_SDRC_D14	RW	32	0x0000 001C	0x4800 204C
CONTROL_PADCONF_SDRC_D16	RW	32	0x0000 0020	0x4800 2050
CONTROL_PADCONF_SDRC_D18	RW	32	0x0000 0024	0x4800 2054

Table 7-64. PADCONFS Register Summary (continued)

Register Name	Type	Register Width (Bits)	Address Offset	Physical Address
CONTROL_PADCONF_SDRC_D20	RW	32	0x0000 0028	0x4800 2058
CONTROL_PADCONF_SDRC_D22	RW	32	0x0000 002C	0x4800 205C
CONTROL_PADCONF_SDRC_D24	RW	32	0x0000 0030	0x4800 2060
CONTROL_PADCONF_SDRC_D26	RW	32	0x0000 0034	0x4800 2064
CONTROL_PADCONF_SDRC_D28	RW	32	0x0000 0038	0x4800 2068
CONTROL_PADCONF_SDRC_D30	RW	32	0x0000 003C	0x4800 206C
CONTROL_PADCONF_SDRC_CLK	RW	32	0x0000 0040	0x4800 2070
CONTROL_PADCONF_SDRC_DQS1	RW	32	0x0000 0044	0x4800 2074
CONTROL_PADCONF_SDRC_DQS3	RW	32	0x0000 0048	0x4800 2078
CONTROL_PADCONF_GPMC_A2	RW	32	0x0000 004C	0x4800 207C
CONTROL_PADCONF_GPMC_A4	RW	32	0x0000 0050	0x4800 2080
CONTROL_PADCONF_GPMC_A6	RW	32	0x0000 0054	0x4800 2084
CONTROL_PADCONF_GPMC_A8	RW	32	0x0000 0058	0x4800 2088
CONTROL_PADCONF_GPMC_A10	RW	32	0x0000 005C	0x4800 208C
CONTROL_PADCONF_GPMC_D1	RW	32	0x0000 0060	0x4800 2090
CONTROL_PADCONF_GPMC_D3	RW	32	0x0000 0064	0x4800 2094
CONTROL_PADCONF_GPMC_D5	RW	32	0x0000 0068	0x4800 2098
CONTROL_PADCONF_GPMC_D7	RW	32	0x0000 006C	0x4800 209C
CONTROL_PADCONF_GPMC_D9	RW	32	0x0000 0070	0x4800 20A0
CONTROL_PADCONF_GPMC_D11	RW	32	0x0000 0074	0x4800 20A4
CONTROL_PADCONF_GPMC_D13	RW	32	0x0000 0078	0x4800 20A8
CONTROL_PADCONF_GPMC_D15	RW	32	0x0000 007C	0x4800 20AC
CONTROL_PADCONF_GPMC_NCS1	RW	32	0x0000 0080	0x4800 20B0
CONTROL_PADCONF_GPMC_NCS3	RW	32	0x0000 0084	0x4800 20B4
CONTROL_PADCONF_GPMC_NCS5	RW	32	0x0000 0088	0x4800 20B8
CONTROL_PADCONF_GPMC_NCS7	RW	32	0x0000 008C	0x4800 20BC
CONTROL_PADCONF_GPMC_NADV_ALE	RW	32	0x0000 0090	0x4800 20C0
CONTROL_PADCONF_GPMC_NWE	RW	32	0x0000 0094	0x4800 20C4
CONTROL_PADCONF_GPMC_NBE1	RW	32	0x0000 0098	0x4800 20C8
CONTROL_PADCONF_GPMC_WAIT0	RW	32	0x0000 009C	0x4800 20CC
CONTROL_PADCONF_GPMC_WAIT2	RW	32	0x0000 00A0	0x4800 20D0
CONTROL_PADCONF_DSS_PCLK	RW	32	0x0000 00A4	0x4800 20D4
CONTROL_PADCONF_DSS_VSYNC	RW	32	0x0000 00A8	0x4800 20D8
CONTROL_PADCONF_DSS_DATA0	RW	32	0x0000 00AC	0x4800 20DC
CONTROL_PADCONF_DSS_DATA2	RW	32	0x0000 00B0	0x4800 20E0
CONTROL_PADCONF_DSS_DATA4	RW	32	0x0000 00B4	0x4800 20E4
CONTROL_PADCONF_DSS_DATA6	RW	32	0x0000 00B8	0x4800 20E8
CONTROL_PADCONF_DSS_DATA8	RW	32	0x0000 00BC	0x4800 20EC
CONTROL_PADCONF_DSS_DATA10	RW	32	0x0000 00C0	0x4800 20F0
CONTROL_PADCONF_DSS_DATA12	RW	32	0x0000 00C4	0x4800 20F4
CONTROL_PADCONF_DSS_DATA14	RW	32	0x0000 00C8	0x4800 20F8
CONTROL_PADCONF_DSS_DATA16	RW	32	0x0000 00CC	0x4800 20FC
CONTROL_PADCONF_DSS_DATA18	RW	32	0x0000 00D0	0x4800 2100
CONTROL_PADCONF_DSS_DATA20	RW	32	0x0000 00D4	0x4800 2104
CONTROL_PADCONF_DSS_DATA22	RW	32	0x0000 00D8	0x4800 2108
CONTROL_PADCONF_CAM_HS	RW	32	0x0000 00DC	0x4800 210C
CONTROL_PADCONF_CAM_XCLKA	RW	32	0x0000 00E0	0x4800 2110

Table 7-64. PADCONFS Register Summary (continued)

Register Name	Type	Register Width (Bits)	Address Offset	Physical Address
CONTROL_PADCONF_CAM_FLD	RW	32	0x0000 00E4	0x4800 2114
CONTROL_PADCONF_CAM_D1	RW	32	0x0000 00E8	0x4800 2118
CONTROL_PADCONF_CAM_D3	RW	32	0x0000 00EC	0x4800 211C
CONTROL_PADCONF_CAM_D5	RW	32	0x0000 00F0	0x4800 2120
CONTROL_PADCONF_CAM_D7	RW	32	0x0000 00F4	0x4800 2124
CONTROL_PADCONF_CAM_D9	RW	32	0x0000 00F8	0x4800 2128
CONTROL_PADCONF_CAM_D11	RW	32	0x0000 00FC	0x4800 212C
CONTROL_PADCONF_CAM_WEN	RW	32	0x0000 0100	0x4800 2130
CONTROL_PADCONF_CSI2_DX0	RW	32	0x0000 0104	0x4800 2134
CONTROL_PADCONF_CSI2_DX1	RW	32	0x0000 0108	0x4800 2138
CONTROL_PADCONF_MCBSP2_FSX	RW	32	0x0000 010C	0x4800 213C
CONTROL_PADCONF_MCBSP2_DR	RW	32	0x0000 0110	0x4800 2140
CONTROL_PADCONF_MMC1_CLK	RW	32	0x0000 0114	0x4800 2144
CONTROL_PADCONF_MMC1_DAT0	RW	32	0x0000 0118	0x4800 2148
CONTROL_PADCONF_MMC1_DAT2	RW	32	0x0000 011C	0x4800 214C
CONTROL_PADCONF_MMC1_DAT4	RW	32	0x0000 0120	0x4800 2150
CONTROL_PADCONF_MMC1_DAT6	RW	32	0x0000 0124	0x4800 2154
CONTROL_PADCONF_MMC2_CLK	RW	32	0x0000 0128	0x4800 2158
CONTROL_PADCONF_MMC2_DAT0	RW	32	0x0000 012C	0x4800 215C
CONTROL_PADCONF_MMC2_DAT2	RW	32	0x0000 0130	0x4800 2160
CONTROL_PADCONF_MMC2_DAT4	RW	32	0x0000 0134	0x4800 2164
CONTROL_PADCONF_MMC2_DAT6	RW	32	0x0000 0138	0x4800 2168
CONTROL_PADCONF_MCBSP3_DX	RW	32	0x0000 013C	0x4800 216C
CONTROL_PADCONF_MCBSP3_CLKX	RW	32	0x0000 0140	0x4800 2170
CONTROL_PADCONF_UART2_CTS	RW	32	0x0000 0144	0x4800 2174
CONTROL_PADCONF_UART2_TX	RW	32	0x0000 0148	0x4800 2178
CONTROL_PADCONF_UART1_TX	RW	32	0x0000 014C	0x4800 217C
CONTROL_PADCONF_UART1_CTS	RW	32	0x0000 0150	0x4800 2180
CONTROL_PADCONF_MCBSP4_CLKX	RW	32	0x0000 0154	0x4800 2184
CONTROL_PADCONF_MCBSP4_DX	RW	32	0x0000 0158	0x4800 2188
CONTROL_PADCONF_MCBSP1_CLKR	RW	32	0x0000 015C	0x4800 218C
CONTROL_PADCONF_MCBSP1_DX	RW	32	0x0000 0160	0x4800 2190
CONTROL_PADCONF_MCBSP_CLKS	RW	32	0x0000 0164	0x4800 2194
CONTROL_PADCONF_MCBSP1_CLKX	RW	32	0x0000 0168	0x4800 2198
CONTROL_PADCONF_UART3_RTS_SD	RW	32	0x0000 016C	0x4800 219C
CONTROL_PADCONF_UART3_TX_IRTX	RW	32	0x0000 0170	0x4800 21A0
CONTROL_PADCONF_HSUSB0_STP	RW	32	0x0000 0174	0x4800 21A4
CONTROL_PADCONF_HSUSB0_NXT	RW	32	0x0000 0178	0x4800 21A8
CONTROL_PADCONF_HSUSB0_DATA1	RW	32	0x0000 017C	0x4800 21AC
CONTROL_PADCONF_HSUSB0_DATA3	RW	32	0x0000 0180	0x4800 21B0
CONTROL_PADCONF_HSUSB0_DATA5	RW	32	0x0000 0184	0x4800 21B4
CONTROL_PADCONF_HSUSB0_DATA7	RW	32	0x0000 0188	0x4800 21B8
CONTROL_PADCONF_I2C1_SDA	RW	32	0x0000 018C	0x4800 21BC
CONTROL_PADCONF_I2C2_SDA	RW	32	0x0000 0190	0x4800 21C0
CONTROL_PADCONF_I2C3_SDA	RW	32	0x0000 0194	0x4800 21C4
CONTROL_PADCONF_MCSPI1_CLK	RW	32	0x0000 0198	0x4800 21C8
CONTROL_PADCONF_MCSPI1_SOMI	RW	32	0x0000 019C	0x4800 21CC

Table 7-64. PADCONFS Register Summary (continued)

Register Name	Type	Register Width (Bits)	Address Offset	Physical Address
CONTROL_PADCONF_MCSPI1_CS1	RW	32	0x0000 01A0	0x4800 21D0
CONTROL_PADCONF_MCSPI1_CS3	RW	32	0x0000 01A4	0x4800 21D4
CONTROL_PADCONF_MCSPI2_SIMO	RW	32	0x0000 01A8	0x4800 21D8
CONTROL_PADCONF_MCSPI2_CS0	RW	32	0x0000 01AC	0x4800 21DC
CONTROL_PADCONF_SYS_NIRQ	RW	32	0x0000 01B0	0x4800 21E0
CONTROL_PADCONF_SAD2D_MCAD0	RW	32	0x0000 01B4	0x4800 21E4
CONTROL_PADCONF_SAD2D_MCAD2	RW	32	0x0000 01B8	0x4800 21E8
CONTROL_PADCONF_SAD2D_MCAD4	RW	32	0x0000 01BC	0x4800 21EC
CONTROL_PADCONF_SAD2D_MCAD6	RW	32	0x0000 01C0	0x4800 21F0
CONTROL_PADCONF_SAD2D_MCAD8	RW	32	0x0000 01C4	0x4800 21F4
CONTROL_PADCONF_SAD2D_MCAD10	RW	32	0x0000 01C8	0x4800 21F8
CONTROL_PADCONF_SAD2D_MCAD12	RW	32	0x0000 01CC	0x4800 21FC
CONTROL_PADCONF_SAD2D_MCAD14	RW	32	0x0000 01D0	0x4800 2200
CONTROL_PADCONF_SAD2D_MCAD16	RW	32	0x0000 01D4	0x4800 2204
CONTROL_PADCONF_SAD2D_MCAD18	RW	32	0x0000 01D8	0x4800 2208
CONTROL_PADCONF_SAD2D_MCAD20	RW	32	0x0000 01DC	0x4800 220C
CONTROL_PADCONF_SAD2D_MCAD22	RW	32	0x0000 01E0	0x4800 2210
CONTROL_PADCONF_SAD2D_MCAD24	RW	32	0x0000 01E4	0x4800 2214
CONTROL_PADCONF_SAD2D_MCAD26	RW	32	0x0000 01E8	0x4800 2218
CONTROL_PADCONF_SAD2D_MCAD28	RW	32	0x0000 01EC	0x4800 221C
CONTROL_PADCONF_SAD2D_MCAD30	RW	32	0x0000 01F0	0x4800 2220
CONTROL_PADCONF_SAD2D_MCAD32	RW	32	0x0000 01F4	0x4800 2224
CONTROL_PADCONF_SAD2D_MCAD34	RW	32	0x0000 01F8	0x4800 2228
CONTROL_PADCONF_SAD2D_MCAD36	RW	32	0x0000 01FC	0x4800 222C
CONTROL_PADCONF_SAD2D_NRESPWRON	RW	32	0x0000 0200	0x4800 2230
CONTROL_PADCONF_SAD2D_ARMNIRQ	RW	32	0x0000 0204	0x4800 2234
CONTROL_PADCONF_SAD2D_SPINT	RW	32	0x0000 0208	0x4800 2238
CONTROL_PADCONF_SAD2D_DMAREQ0	RW	32	0x0000 020C	0x4800 223C
CONTROL_PADCONF_SAD2D_DMAREQ2	RW	32	0x0000 0210	0x4800 2240
CONTROL_PADCONF_SAD2D_NTRST	RW	32	0x0000 0214	0x4800 2244
CONTROL_PADCONF_SAD2D_TDO	RW	32	0x0000 0218	0x4800 2248
CONTROL_PADCONF_SAD2D_TCK	RW	32	0x0000 021C	0x4800 224C
CONTROL_PADCONF_SAD2D_MSTDBY	RW	32	0x0000 0220	0x4800 2250
CONTROL_PADCONF_SAD2D_IDLEACK	RW	32	0x0000 0224	0x4800 2254
CONTROL_PADCONF_SAD2D_SWRITE	RW	32	0x0000 0228	0x4800 2258
CONTROL_PADCONF_SAD2D_SREAD	RW	32	0x0000 022C	0x4800 225C
CONTROL_PADCONF_SAD2D_SBUSFLAG	RW	32	0x0000 0230	0x4800 2260
CONTROL_PADCONF_SDRC_CKE1	RW	32	0x0000 0234	0x4800 2264
CONTROL_PADCONF_ETK_CLK	RW	32	0x0000 05A8	0x4800 25D8
CONTROL_PADCONF_ETK_D0	RW	32	0x0000 05AC	0x4800 25DC
CONTROL_PADCONF_ETK_D2	RW	32	0x0000 05B0	0x4800 25E0
CONTROL_PADCONF_ETK_D4	RW	32	0x0000 05B4	0x4800 25E4
CONTROL_PADCONF_ETK_D6	RW	32	0x0000 05B8	0x4800 25E8
CONTROL_PADCONF_ETK_D8	RW	32	0x0000 05BC	0x4800 25EC
CONTROL_PADCONF_ETK_D10	RW	32	0x0000 05C0	0x4800 25F0
CONTROL_PADCONF_ETK_D12	RW	32	0x0000 05C4	0x4800 25F4
CONTROL_PADCONF_ETK_D14	RW	32	0x0000 05C8	0x4800 25F8

Table 7-65 lists the general registers.

Table 7-65. General Register Summary

Register Name	Type	Register Width (Bits)	Address Offset	Physical Address
CONTROL_PADCONF_OFF	RW	32	0x0000 0000	0x4800 2270
CONTROL_DEVCONF0	RW	32	0x0000 0004	0x4800 2274
RESERVED	R	32	0x0000 0008	0x4800 2278
RESERVED	R	32	0x0000 000C	0x4800 227C
CONTROL_MSUSPENDMUX_0	RW	32	0x0000 0020	0x4800 2290
CONTROL_MSUSPENDMUX_1	RW	32	0x0000 0024	0x4800 2294
CONTROL_MSUSPENDMUX_2	RW	32	0x0000 0028	0x4800 2298
CONTROL_MSUSPENDMUX_3	RW	32	0x0000 002C	0x4800 229C
CONTROL_MSUSPENDMUX_4	RW	32	0x0000 0030	0x4800 22A0
CONTROL_MSUSPENDMUX_5	RW	32	0x0000 0034	0x4800 22A4
CONTROL_DEVCONF1	RW	32	0x0000 0068	0x4800 22D8
CONTROL_CSIRXFE	RW	32	0x0000 006C	0x4800 22DC
CONTROL_PROT_ERR_STATUS	RW	32	0x0000 0074	0x4800 22E4
CONTROL_PROT_ERR_STATUS_DEBUG	RW	32	0x0000 0078	0x4800 22E8
CONTROL_STATUS	R	32	0x0000 0080	0x4800 22F0
CONTROL_GENERAL_PURPOSE_STATUS	R	32	0x0000 0084	0x4800 22F4
CONTROL_RPUB_KEY_H_0	R	32	0x0000 0090	0x4800 2300
CONTROL_RPUB_KEY_H_1	R	32	0x0000 0094	0x4800 2304
CONTROL_RPUB_KEY_H_2	R	32	0x0000 0098	0x4800 2308
CONTROL_RPUB_KEY_H_3	R	32	0x0000 009C	0x4800 230C
CONTROL_RPUB_KEY_H_4	R	32	0x0000 00A0	0x4800 2310
CONTROL_USB_CONF_0	R	32	0x0000 0100	0x4800 2370
CONTROL_USB_CONF_1	R	32	0x0000 0104	0x4800 2374
CONTROL_FUSE_OPP1_VDD1	R	32	0x0000 0110	0x4800 2380
CONTROL_FUSE_OPP2_VDD1	R	32	0x0000 0114	0x4800 2384
CONTROL_FUSE_OPP3_VDD1	R	32	0x0000 0118	0x4800 2388
CONTROL_FUSE_OPP4_VDD1	R	32	0x0000 011C	0x4800 238C
CONTROL_FUSE_OPP5_VDD1	R	32	0x0000 0120	0x4800 2390
CONTROL_FUSE_OPP1_VDD2	RW	32	0x0000 0124	0x4800 2394
CONTROL_FUSE_OPP2_VDD2	RW	32	0x0000 0128	0x4800 2398
CONTROL_FUSE_OPP3_VDD2	RW	32	0x0000 012C	0x4800 239C
CONTROL_FUSE_SR	RW	32	0x0000 0130	0x4800 23A0
CONTROL_IVA2_BOOTADDR	RW	32	0x0000 0190	0x4800 2400
CONTROL_IVA2_BOOTMOD	RW	32	0x0000 0194	0x4800 2404
CONTROL_DEBOBS_0	RW	32	0x0000 01B0	0x4800 2420
CONTROL_DEBOBS_1	RW	32	0x0000 01B4	0x4800 2424
CONTROL_DEBOBS_2	RW	32	0x0000 01B8	0x4800 2428
CONTROL_DEBOBS_3	RW	32	0x0000 01BC	0x4800 242C
CONTROL_DEBOBS_4	RW	32	0x0000 01C0	0x4800 2430
CONTROL_DEBOBS_5	RW	32	0x0000 01C4	0x4800 2434
CONTROL_DEBOBS_6	RW	32	0x0000 01C8	0x4800 2438
CONTROL_DEBOBS_7	RW	32	0x0000 01CC	0x4800 243C
CONTROL_DEBOBS_8	RW	32	0x0000 01D0	0x4800 2440
CONTROL_PROG_IO0	RW	32	0x0000 01D4	0x4800 2444
CONTROL_PROG_IO1	RW	32	0x0000 01D8	0x4800 2448
CONTROL_DSS_DPLL_SPREADING	RW	32	0x0000 01E0	0x4800 2450

Table 7-65. General Register Summary (continued)

Register Name	Type	Register Width (Bits)	Address Offset	Physical Address
CONTROL_CORE_DPLL_SPREADING	RW	32	0x0000 01E4	0x4800 2454
CONTROL_PER_DPLL_SPREADING	RW	32	0x0000 01E8	0x4800 2458
CONTROL_USBHOST_DPLL_SPREADING	RW	32	0x0000 01EC	0x4800 245C
CONTROL_SDRC_SHARING	RW	32	0x0000 01F0	0x4800 2460
CONTROL_SDRC_MCFG0	RW	32	0x0000 01F4	0x4800 2464
CONTROL_SDRC_MCFG1	RW	32	0x0000 01F8	0x4800 2468
CONTROL_MODEM_FW_CONFIGURATION_LOCK	RW	32	0x0000 01FC	0x4800 246C
CONTROL_MODEM_MEMORY_RESOURCES_CONF	RW	32	0x0000 0200	0x4800 2470
CONTROL_MODEM_GPMC_DT_FW_REQ_INFO	RW	32	0x0000 0204	0x4800 2474
CONTROL_MODEM_GPMC_DT_FW_RD	RW	32	0x0000 0208	0x4800 2478
CONTROL_MODEM_GPMC_DT_FW_WR	RW	32	0x0000 020C	0x4800 247C
CONTROL_MODEM_GPMC_BOOT_CODE	RW	32	0x0000 0210	0x4800 2480
CONTROL_MODEM_SMS_RG_ATT1	RW	32	0x0000 0214	0x4800 2484
CONTROL_MODEM_SMS_RG_RDPERM1	RW	32	0x0000 0218	0x4800 2488
CONTROL_MODEM_SMS_RG_WRPERM1	RW	32	0x0000 021C	0x4800 248C
CONTROL_MODEM_D2D_FW_DEBUG_MODE	RW	32	0x0000 0220	0x4800 2490
CONTROL_DPF_OCM_RAM_FW_ADDR_MATCH	RW	32	0x0000 0228	0x4800 2498
CONTROL_DPF_OCM_RAM_FW_REQINFO	RW	32	0x0000 022C	0x4800 249C
CONTROL_DPF_OCM_RAM_FW_WR	RW	32	0x0000 0230	0x4800 24A0
CONTROL_DPF_REGION4_GPMC_FW_ADDR_MATCH	RW	32	0x0000 0234	0x4800 24A4
CONTROL_DPF_REGION4_GPMC_FW_REQINFO	RW	32	0x0000 0238	0x4800 24A8
CONTROL_DPF_REGION4_GPMC_FW_WR	RW	32	0x0000 023C	0x4800 24AC
CONTROL_DPF_REGION1_IVA2_FW_ADDR_MATCH	RW	32	0x0000 0240	0x4800 24B0
CONTROL_DPF_REGION1_IVA2_FW_REQINFO	RW	32	0x0000 0244	0x4800 24B4
CONTROL_DPF_REGION1_IVA2_FW_WR	RW	32	0x0000 0248	0x4800 24B8
CONTROL_DPF_MAD2D_FW_ADDR_MATCH	RW	32	0x0000 02C8	0x4800 2538
CONTROL_DPF_MAD2D_FW_REQINFO	RW	32	0x0000 02CC	0x4800 253C
CONTROL_DPF_MAD2D_FW_WR	RW	32	0x0000 02D0	0x4800 2540
CONTROL_PBIAS_LITE	RW	32	0x0000 02B0	0x4800 2520
CONTROL_TEMP_SENSOR	RW	32	0x0000 02B4	0x4800 2524
CONTROL_CSI	RW	32	0x0000 02C0	0x4800 2530
CONTROL_IDCODE	RW	32	0x0030 7F94	0x4830 A204

Table 7-66 lists the MEM_WKUP registers.

Table 7-66. MEM_WKUP Register Summary

Register Name	Type	Register Width (Bits)	Address Offset	Physical Address
CONTROL_SAVE_RESTORE_MEM	RW	32	0x0600 - 0x09FC	0x4800 2600 - 0x4800 29FC

Table 7-67 lists the PADCONFS_WKUP registers.

Table 7-67. PADCONFS_WKUP Register Summary

Register Name	Type	Register Width (Bits)	Address Offset	Physical Address
CONTROL_PADCONF_I2C4_SCL	RW	32	0x0000 0000	0x4800 2A00

Table 7-67. PADCONFS_WKUP Register Summary (continued)

Register Name	Type	Register Width (Bits)	Address Offset	Physical Address
CONTROL_PADCONF_SYS_32K	RW	32	0x0000 0004	0x4800 2A04
CONTROL_PADCONF_SYS_NRESWARM	RW	32	0x0000 0008	0x4800 2A08
CONTROL_PADCONF_SYS_BOOT1	RW	32	0x0000 000C	0x4800 2A0C
CONTROL_PADCONF_SYS_BOOT3	RW	32	0x0000 0010	0x4800 2A10
CONTROL_PADCONF_SYS_BOOT5	RW	32	0x0000 0014	0x4800 2A14
CONTROL_PADCONF_SYS_OFF_MODE	RW	32	0x0000 0018	0x4800 2A18
CONTROL_PADCONF_JTAG_NTRST	RW	32	0x0000 001C	0x4800 2A1C
CONTROL_PADCONF_JTAG_TMS_TMSC	RW	32	0x0000 0020	0x4800 2A20
CONTROL_PADCONF_JTAG_EMU0	RW	32	0x0000 0024	0x4800 2A24
CONTROL_PADCONF_SAD2D_SWAKEUP	RW	32	0x0000 004C	0x4800 2A4C
CONTROL_PADCONF_JTAG_TDO	RW	32	0x00000050	0x48002A50

Table 7-68 lists the GENERAL_WKUP registers.

Table 7-68. GENERAL_WKUP Register Summary

Register Name	Type	Register Width (Bits)	Address Offset	Physical Address
CONTROL_WKUP_CTRL	RW	32	0x0000 0000	0x4800 2A5C
CONTROL_WKUP_DEBOBS_0	RW	32	0x0000 000C	0x4800 2A68
CONTROL_WKUP_DEBOBS_1	RW	32	0x0000 0010	0x4800 2A6C
CONTROL_WKUP_DEBOBS_2	RW	32	0x0000 0014	0x4800 2A70
CONTROL_WKUP_DEBOBS_3	RW	32	0x0000 0018	0x4800 2A74
CONTROL_WKUP_DEBOBS_4	RW	32	0x0000 001C	0x4800 2A78

7.6.3 SCM Register Description

7.6.3.1 INTERFACE Register Description

Table 7-69 through Table 7-71 describe the interface register bits.

Table 7-69. CONTROL_REVISION

Address Offset	0x00		
Physical address	0x4800 2000	**Instance**	INTERFACE
Description	Control module Revision number		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																								REVISION							

Bits	Field Name	Description	Type	Reset
31:8	RESERVED	Read returns reset value.	R	0x000000
7:0	REVISION	Revision number	R	See (1)

(1) TI internal data

Table 7-70. Register Call Summary for Register CONTROL_REVISION

SCM Register Manual
- SCM Register Summary: [0]

Table 7-71. CONTROL_SYSCONFIG

Address Offset	0x10		
Physical address	0x4800 2010	**Instance**	INTERFACE
Description	Set various parameters relative to the Idle mode of the Control module		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																											IDLEMODE			ENAWAKEUP	SOFTRESET	AUTOIDLE

Bits	Field Name	Description	Type	Reset
31:5	RESERVED	Read returns reset value.	R	0x0000000
4:3	IDLEMODE	Power Management, req/ack control	R/W	0x0
		0x0: Force-idle. An idle request is acknowledged unconditionally		
		0x1: Reserved		
		0x2: Smart-idle. Acknowledgement to an idle request is given based on the internal activity of the module		
		0x3: Reserved		
2	ENAWAKEUP	Wake-up enable. Not used in the module	R	0x0
1	SOFTRESET	Software reset. Not used in the module	R	0x0
0	AUTOIDLE	Internal interface clock gating strategy	R/W	0x0
		0x0: Interface clock is free-running		
		0x1: Automatic interface clock gating strategy is applied, based on the interconnect interface activity		

Table 7-72. Register Call Summary for Register CONTROL_SYSCONFIG

SCM Integration
- Resets: [0]
- Power Management: [1] [2] [3] [4]

SCM Programming Model
- Off Mode Preliminary Settings: [5]

SCM Register Manual
- SCM Register Summary: [6]

7.6.3.2 PADCONFS Register Description

Each 32-bit PADCONF register gathers the configuration of two pads. For example, CONTROL.CONTROL_PADCONF_GPMC_7 is used to configure gpmc_7 pad (bits [15:0]) and gpmc_8 pad (bits [31:16]). See Figure 7-8 for more information about PADCONF registers.

According to the pad type, some features are configurable or not. Table 7-73 gives the description of a fully configurable pad.

Table 7-73. CONTROL_PADCONF_X

Address Offset	0x0000 0000 - 0x0000 05C8		
Physical address	See Table 7-64	**Instance**	SYSC_PADCONFS
Description	Pad configuration register		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
WAKEUPEVENT1	WAKEUPENABLE1	OFFPULLTYPE	OFFPULLUDENABLE1	OFFOUTVALUE1	OFFOUTENABLE1	OFFENABLE1	INPUTENABLE1	RESERVED			PULLTYPESELECT1	PULLUDENABLE1	MUXMODE1			WAKEUPEVENT0	WAKEUPENABLE0	OFFPULLTYPESELECT0	OFFPULLUDENABLE0	OFFOUTVALUE0	OFFOUTENABLE0	OFFENABLE0	INPUTENABLE0	RESERVED			PULLTYPESELECT0	PULLUDENABLE0	MUXMODE0		

Bits	Field Name	Description	Type	Reset
31	WAKEUPEVENT1 See NOTE following this table.	Pad_x wake-up event status latched in the I/O: 0: No wake-up event occurred 1: A wake-up event occurred	R	0
30	WAKEUPENABLE1	Input pad wake-up enable (and OFF mode input enable value) for pad_x: 0: Wake-up detection on is disabled. 1: Wake-up detection on is enabled.	R/W	0
29	OFFPULLTYPE SELECT1	Off mode Pull-Up/Down selection for pad_x: 0: Pull-Down selected 1: Pull-Up selected	R/W	0
28	OFFPULLUDENABLE1	Off mode Pull-Up/Down enable for pad_x: 0: Pull-Up/Down disabled 1: Pull-Up/Down enabled	R/W	0
27	OFFOUTVALUE1	Off mode pad_x output value	R/W	0
26	OFFOUTENABLE1	Off mode pad_x output enable value: (Warning:This is an active low signal.) 0: Output enabled 1: Output disabled	R/W	0
25	OFFENABLE1	Off mode pad_x state override control: 0: Off mode disabled 1: Off mode enabled	R/W	0
24	INPUTENABLE1	Input enable value for pad_x	R/W	1
23:21	RESERVED	Reserved	R	0
20	PULLTYPESELECT1	Pull-Up/Down selection for pad_x: 0: Pull-Down selected 1: Pull-Up selected	R/W	Pad dependent
19	PULLUDENABLE1	Pull-Up/Down enable for pad_x: 0: Pull-Up/Down disabled 1: Pull-Up/Down enabled	R/W	Pad dependent
18:16	MUXMODE1	Functional multiplexing selection for pad_x	R/W	Pad dependent

Bits	Field Name	Description	Type	Reset
15	WAKEUPEVENT0 See NOTE following this table.	Pad_y wake-up event status latched in the I/O: 0: No wake-up event occurred 1: A wake-up event occurred	R	Pad dependent
14	WAKEUPENABLE0	Input pad wake-up enable (and OFF mode input enable value) for pad_y: 0: Wake-up detection on is disabled. 1: Wake-up detection on is enabled.	R/W	0
13	OFFPULLTYPESELECT0	Off mode Pull-Up/Down selection for pad_y: 0: Pull-Down selected 1: Pull-Up selected	R/W	0
12	OFFPULLUDENABLE0	Off mode Pull-Up/Down enable for pad_y: 0: Pull-Up/Down disabled 1: Pull-Up/Down enabled	R/W	0
11	OFFOUTVALUE0	Off mode pad_y output value	R/W	0
10	OFFOUTENABLE0	Off mode pad_y output enable value: (Warning:This is an active low signal.) 0: Off mode enabled. 1: Off mode disabled	R/W	0
9	OFFENABLE0	Off mode pad_y state override control: 0: Off mode disabled. 1: Off mode enabled	R/W	0
8	INPUTENABLE0	Input enable value for pad_y	R/W	1
7:5	RESERVED	Reserved	R	0
4	PULLTYPESELECT0	Pull-Up/Down selection for pad_y: 0: Pull-Down selected 1: Pull-Up selected	R/W	Pad dependent
3	PULLUDENABLE0	Pull-Up/Down enable for pad_y: 0: Pull-Up/Down disabled 1: Pull-Up/Down enabled	R/W	Pad dependent
2:0	MUXMODE0	Functional multiplexing selection for pad_y	R/W	Pad dependent

Table 7-74. Register Call Summary for Register CONTROL_PADCONF_X

SCM Programming Model
• I/O Power Optimization Guidelines: [0] [1]

Note: The WAKEUPEVENT1 of the CAM_D0 register is swapped with the WAKEUPEVENT0 of CAM_D1.

CAUTION

The OFFOUTENABLE and OFFOUTVALUE bits are functional only if the pad configuration supports output mode on at least one MUXMODE. For a pad that supports only the input feature, the OFFOUTENABLE and OFFOUTVALUE bits cannot be configured (they are don't care and read always returns 0).

Note: The Bits field gives the field number for the pairs of pads gathered in each register.

Table 7-75 describes the reset values, the capabilities, and the corresponding register for each pad.

Notes:

- All '-' assume that the corresponding bit is not available. These bits are considered as Reserved.
- The reset value for Reserved bits is 0.
- In the Mux Reset States and PU/PD Reset States columns of Table 7-75, a dash (-) indicates that the field is hardwired to logic 0. No corresponding control block ports are implemented for these bits.

Table 7-75. CONTROL_PADCONF_CAPABILITIES

REGISTER NAME	Pad Name	Physical Address	WakeUpx	OffMode	Input Enable	Reserved	PU/PD	MuxMode
CONTROL_PADCONF_SDRC_D0[15:0]	sdrc_d0	0x4800 2030	--	-----	0b1	0b000	0b00	---
CONTROL_PADCONF_SDRC_D0[31:16]	sdrc_d1	0x4800 2030	--	-----	0b1	0b000	0b00	---
CONTROL_PADCONF_SDRC_D2[15:0]	sdrc_d2	0x4800 2034	--	-----	0b1	0b000	0b00	---
CONTROL_PADCONF_SDRC_D2[31:16]	sdrc_d3	0x4800 2034	--	-----	0b1	0b000	0b00	---
CONTROL_PADCONF_SDRC_D4[15:0]	sdrc_d4	0x4800 2038	--	-----	0b1	0b000	0b00	---
CONTROL_PADCONF_SDRC_D4[31:16]	sdrc_d5	0x4800 2038	--	-----	0b1	0b000	0b00	---
CONTROL_PADCONF_SDRC_D6[15:0]	sdrc_d6	0x4800 203C	--	-----	0b1	0b000	0b00	---
CONTROL_PADCONF_SDRC_D6[31:16]	sdrc_d7	0x4800 203C	--	-----	0b1	0b000	0b00	---
CONTROL_PADCONF_SDRC_D8[15:0]	sdrc_d8	0x4800 2040	--	-----	0b1	0b000	0b00	---
CONTROL_PADCONF_SDRC_D8[31:16]	sdrc_d9	0x4800 2040	--	-----	0b1	0b000	0b00	---
CONTROL_PADCONF_SDRC_D10[15:0]	sdrc_d10	0x4800 2044	--	-----	0b1	0b000	0b00	---
CONTROL_PADCONF_SDRC_D10[31:16]	sdrc_d11	0x4800 2044	--	-----	0b1	0b000	0b00	---
CONTROL_PADCONF_SDRC_D12[15:0]	sdrc_d12	0x4800 2048	--	-----	0b1	0b000	0b00	---
CONTROL_PADCONF_SDRC_D12[31:16]	sdrc_d13	0x4800 2048	--	-----	0b1	0b000	0b00	---
CONTROL_PADCONF_SDRC_D14[15:0]	sdrc_d14	0x4800 204C	--	-----	0b1	0b000	0b00	---
CONTROL_PADCONF_SDRC_D14[31:16]	sdrc_d15	0x4800 204C	--	-----	0b1	0b000	0b00	---
CONTROL_PADCONF_SDRC_D16[15:0]	sdrc_d16	0x4800 2050	--	-----	0b1	0b000	0b00	---
CONTROL_PADCONF_SDRC_D16[31:16]	sdrc_d17	0x4800 2050	--	-----	0b1	0b000	0b00	---
CONTROL_PADCONF_SDRC_D18[15:0]	sdrc_d18	0x4800 2054	--	-----	0b1	0b000	0b00	---
CONTROL_PADCONF_SDRC_D18[31:16]	sdrc_d19	0x4800 2054	--	-----	0b1	0b000	0b00	---
CONTROL_PADCONF_SDRC_D20[15:0]	sdrc_d20	0x4800 2058	--	-----	0b1	0b000	0b00	---
CONTROL_PADCONF_SDRC_D20[31:16]	sdrc_d21	0x4800 2058	--	-----	0b1	0b000	0b00	---
CONTROL_PADCONF_SDRC_D22[15:0]	sdrc_d22	0x4800 205C	--	-----	0b1	0b000	0b00	---
CONTROL_PADCONF_SDRC_D22[31:16]	sdrc_d23	0x4800 205C	--	-----	0b1	0b000	0b00	---
CONTROL_PADCONF_SDRC_D24[15:0]	sdrc_d24	0x4800 2060	--	-----	0b1	0b000	0b00	---
CONTROL_PADCONF_SDRC_D24[31:16]	sdrc_d25	0x4800 2060	--	-----	0b1	0b000	0b00	---
CONTROL_PADCONF_SDRC_D26[15:0]	sdrc_d26	0x4800 2064	--	-----	0b1	0b000	0b00	---
CONTROL_PADCONF_SDRC_D26[31:16]	sdrc_d27	0x4800 2064	--	-----	0b1	0b000	0b00	---
CONTROL_PADCONF_SDRC_D28[15:0]	sdrc_d28	0x4800 2068	--	-----	0b1	0b000	0b00	---
CONTROL_PADCONF_SDRC_D28[31:16]	sdrc_d29	0x4800 2068	--	-----	0b1	0b000	0b00	---
CONTROL_PADCONF_SDRC_D30[15:0]	sdrc_d30	0x4800 206C	--	-----	0b1	0b000	0b00	---
CONTROL_PADCONF_SDRC_D30[31:16]	sdrc_d31	0x4800 206C	--	-----	0b1	0b000	0b00	---
CONTROL_PADCONF_SDRC_CLK[15:0]	sdrc_clk	0x4800 2070	--	-----	0b1	0b000	0b00	---

Table 7-75. CONTROL_PADCONF_CAPABILITIES (continued)

REGISTER NAME	Pad Name	Physical Address	WakeUpx	OffMode	Input Enable	Reserved	PU/PD	MuxMode
CONTROL_PADCONF_SDRC_CLK[31:16]	sdrc_dqs0	0x4800 2070	--	-----	0b1	0b000	0b00	---
CONTROL_PADCONF_SDRC_DQS1[15:0]	sdrc_dqs1	0x4800 2074	--	-----	0b1	0b000	0b00	---
CONTROL_PADCONF_SDRC_DQS1[31:16]	sdrc_dqs2	0x4800 2074	--	-----	0b1	0b000	0b00	---
CONTROL_PADCONF_SDRC_DQS3[15:0]	sdrc_dqs3	0x4800 2078	--	-----	0b1	0b000	0b00	---
CONTROL_PADCONF_SDRC_DQS3[31:16]	gpmc_a1	0x4800 2078	0b00	0b00000	0b1	0b000	0b01	0b111
CONTROL_PADCONF_GPMC_A2[15:0]	gpmc_a2	0x4800 207C	0b00	0b00000	0b1	0b000	0b01	0b111
CONTROL_PADCONF_GPMC_A2[31:16]	gpmc_a3	0x4800 207C	0b00	0b00000	0b1	0b000	0b01	0b111
CONTROL_PADCONF_GPMC_A4[15:0]	gpmc_a4	0x4800 2080	0b00	0b00000	0b1	0b000	0b01	0b111
CONTROL_PADCONF_GPMC_A4[31:16]	gpmc_a5	0x4800 2080	0b00	0b00000	0b1	0b000	0b01	0b111
CONTROL_PADCONF_GPMC_A6[15:0]	gpmc_a6	0x4800 2084	0b00	0b00000	0b1	0b000	0b11	0b111
CONTROL_PADCONF_GPMC_A6[31:16]	gpmc_a7	0x4800 2084	0b00	0b00000	0b1	0b000	0b11	0b111
CONTROL_PADCONF_GPMC_A8[15:0]	gpmc_a8	0x4800 2088	0b00	0b00000	0b1	0b000	0b11	0b111
CONTROL_PADCONF_GPMC_A8[31:16]	gpmc_a9	0x4800 2088	0b00	0b00000	0b1	0b000	0b11	0b111
CONTROL_PADCONF_GPMC_A10[15:0]	gpmc_a10	0x4800 208C	0b00	0b00000	0b1	0b000	0b11	0b111
CONTROL_PADCONF_GPMC_A10[31:16]	gpmc_d0	0x4800 208C	--	0b00000	0b1	0b000	0b11	---
CONTROL_PADCONF_GPMC_D1[15:0]	gpmc_d1	0x4800 2090	--	0b00000	0b1	0b000	0b11	---
CONTROL_PADCONF_GPMC_D1[31:16]	gpmc_d2	0x4800 2090	--	0b00000	0b1	0b000	0b11	---
CONTROL_PADCONF_GPMC_D3[15:0]	gpmc_d3	0x4800 2094	--	0b00000	0b1	0b000	0b11	---
CONTROL_PADCONF_GPMC_D3[31:16]	gpmc_d4	0x4800 2094	--	0b00000	0b1	0b000	0b11	---
CONTROL_PADCONF_GPMC_D5[15:0]	gpmc_d5	0x4800 2098	--	0b00000	0b1	0b000	0b11	---
CONTROL_PADCONF_GPMC_D5[31:16]	gpmc_d6	0x4800 2098	--	0b00000	0b1	0b000	0b11	---
CONTROL_PADCONF_GPMC_D7[15:0]	gpmc_d7	0x4800 209C	--	0b00000	0b1	0b000	0b11	---
CONTROL_PADCONF_GPMC_D7[31:16]	gpmc_d8	0x4800 209C	0b00	0b00000	0b1	0b000	0b11	0b000
CONTROL_PADCONF_GPMC_D9[15:0]	gpmc_d9	0x4800 20A0	0b00	0b00000	0b1	0b000	0b11	0b000
CONTROL_PADCONF_GPMC_D9[31:16]	gpmc_d10	0x4800 20A0	0b00	0b00000	0b1	0b000	0b11	0b000
CONTROL_PADCONF_GPMC_D11[15:0]	gpmc_d11	0x4800 20A4	0b00	0b00000	0b1	0b000	0b11	0b000
CONTROL_PADCONF_GPMC_D11[31:16]	gpmc_d12	0x4800 20A4	0b00	0b00000	0b1	0b000	0b11	0b000
CONTROL_PADCONF_GPMC_D13[15:0]	gpmc_d13	0x4800 20A8	0b00	0b00000	0b1	0b000	0b11	0b000
CONTROL_PADCONF_GPMC_D13[31:16]	gpmc_d14	0x4800 20A8	0b00	0b00000	0b1	0b000	0b11	0b000
CONTROL_PADCONF_GPMC_D15[15:0]	gpmc_d15	0x4800 20AC	0b00	0b00000	0b1	0b000	0b11	0b000
CONTROL_PADCONF_GPMC_D15[31:16]	gpmc_ncs0	0x4800 20AC	--	0b--000	-	0b000	--	---
CONTROL_PADCONF_GPMC_NCS1[15:0]	gpmc_ncs1	0x4800 20B0	0b00	0b00000	0b1	0b000	0b00	0b000
CONTROL_PADCONF_GPMC_NCS1[31:16]	gpmc_ncs2	0x4800 20B0	0b00	0b00000	0b1	0b000	0b11	0b111

Table 7-75. CONTROL_PADCONF_CAPABILITIES (continued)

REGISTER NAME	Pad Name	Physical Address	WakeUpx	OffMode	Input Enable	Reserved	PU/PD	MuxMode
CONTROL_PADCONF_GPMC_NCS3[15:0]	gpmc_ncs3	0x4800 20B4	0b00	0b00000	0b1	0b000	0b11	0b111
CONTROL_PADCONF_GPMC_NCS3[31:16]	gpmc_ncs4	0x4800 20B4	0b00	0b00000	0b1	0b000	0b11	0b111
CONTROL_PADCONF_GPMC_NCS5[15:0]	gpmc_ncs5	0x4800 20B8	0b00	0b00000	0b1	0b000	0b11	0b111
CONTROL_PADCONF_GPMC_NCS5[31:16]	gpmc_ncs6	0x4800 20B8	0b00	0b00000	0b1	0b000	0b11	0b111
CONTROL_PADCONF_GPMC_NCS7[15:0]	gpmc_ncs7	0x4800 20BC	0b00	0b00000	0b1	0b000	0b11	0b111
CONTROL_PADCONF_GPMC_NCS7[31:16]	gpmc_clk	0x4800 20BC	0b00	0b00000	0b1	0b000	0b00	0b000
CONTROL_PADCONF_GPMC_NADV_ALE[15:0]	gpmc_nadv_ale	0x4800 20C0	--	0b--000	-	0b000	--	---
CONTROL_PADCONF_GPMC_NADV_ALE[31:16]	gpmc_noe	0x4800 20C0	--	0b--000	-	0b000	--	---
CONTROL_PADCONF_GPMC_NWE[15:0]	gpmc_nwe	0x4800 20C4	--	0b--000	-	0b000	--	---
CONTROL_PADCONF_GPMC_NWE[31:16]	gpmc_nbe0_cle	0x4800 20C4	0b00	0b00000	0b1	0b000	0b00	0b000
CONTROL_PADCONF_GPMC_NBE1[15:0]	gpmc_nbe1	0x4800 20C8	0b00	0b00000	0b1	0b000	0b01	0b111
CONTROL_PADCONF_GPMC_NBE1[31:16]	gpmc_nwp	0x4800 20C8	0b00	0b00000	0b1	0b000	0b00	0b000
CONTROL_PADCONF_GPMC_WAIT0[15:0]	gpmc_wait0	0x4800 20CC	--	0b00--0	0b1	0b000	0b11	---
CONTROL_PADCONF_GPMC_WAIT0[31:16]	gpmc_wait1	0x4800 20CC	0b00	0b00000	0b1	0b000	0b11	0b111
CONTROL_PADCONF_GPMC_WAIT2[15:0]	gpmc_wait2	0x4800 20D0	0b00	0b00000	0b1	0b000	0b11	0b111
CONTROL_PADCONF_GPMC_WAIT2[31:16]	gpmc_wait3	0x4800 20D0	0b00	0b00000	0b1	0b000	0b11	0b111
CONTROL_PADCONF_DSS_PCLK[15:0]	dss_pclk	0x4800 20D4	0b00	0b00000	0b1	0b000	0b11	0b111
CONTROL_PADCONF_DSS_PCLK[31:16]	dss_hsync	0x4800 20D4	0b00	0b00000	0b1	0b000	0b11	0b111
CONTROL_PADCONF_DSS_VSYNC[15:0]	dss_vsync	0x4800 20D8	0b00	0b00000	0b1	0b000	0b11	0b111
CONTROL_PADCONF_DSS_VSYNC[31:16]	dss_acbias	0x4800 20D8	0b00	0b00000	0b1	0b000	0b01	0b111
CONTROL_PADCONF_DSS_DATA0[15:0]	dss_data0	0x4800 20DC	0b00	0b00000	0b1	0b000	0b01	0b111
CONTROL_PADCONF_DSS_DATA0[31:16]	dss_data1	0x4800 20DC	0b00	0b00000	0b1	0b000	0b01	0b111
CONTROL_PADCONF_DSS_DATA2[15:0]	dss_data2	0x4800 20E0	0b00	0b00000	0b1	0b000	0b01	0b111
CONTROL_PADCONF_DSS_DATA2[31:16]	dss_data3	0x4800 20E0	0b00	0b00000	0b1	0b000	0b01	0b111
CONTROL_PADCONF_DSS_DATA4[15:0]	dss_data4	0x4800 20E4	0b00	0b00000	0b1	0b000	0b01	0b111
CONTROL_PADCONF_DSS_DATA4[31:16]	dss_data5	0x4800 20E4	0b00	0b00000	0b1	0b000	0b01	0b111
CONTROL_PADCONF_DSS_DATA6[15:0]	dss_data6	0x4800 20E8	0b00	0b00000	0b1	0b000	0b01	0b111
CONTROL_PADCONF_DSS_DATA6[31:16]	dss_data7	0x4800 20E8	0b00	0b00000	0b1	0b000	0b01	0b111
CONTROL_PADCONF_DSS_DATA8[15:0]	dss_data8	0x4800 20EC	0b00	0b00000	0b1	0b000	0b01	0b111
CONTROL_PADCONF_DSS_DATA8[31:16]	dss_data9	0x4800 20EC	0b00	0b00000	0b1	0b000	0b01	0b111
CONTROL_PADCONF_DSS_DATA10[15:0]	dss_data10	0x4800 20F0	0b00	0b00000	0b1	0b000	0b01	0b111
CONTROL_PADCONF_DSS_DATA10[31:16]	dss_data11	0x4800 20F0	0b00	0b00000	0b1	0b000	0b01	0b111
CONTROL_PADCONF_DSS_DATA12[15:0]	dss_data12	0x4800 20F4	0b00	0b00000	0b1	0b000	0b01	0b111

Table 7-75. CONTROL_PADCONF_CAPABILITIES (continued)

REGISTER NAME	Pad Name	Physical Address	WakeUpx	OffMode	Input Enable	Reserved	PU/PD	MuxMode
CONTROL_PADCONF_DSS_DATA12[31:16]	dss_data13	0x4800 20F4	0b00	0b00000	0b1	0b000	0b01	0b111
CONTROL_PADCONF_DSS_DATA14[15:0]	dss_data14	0x4800 20F8	0b00	0b00000	0b1	0b000	0b01	0b111
CONTROL_PADCONF_DSS_DATA14[31:16]	dss_data15	0x4800 20F8	0b00	0b00000	0b1	0b000	0b01	0b111
CONTROL_PADCONF_DSS_DATA16[15:0]	dss_data16	0x4800 20FC	0b00	0b00000	0b1	0b000	0b01	0b111
CONTROL_PADCONF_DSS_DATA16[31:16]	dss_data17	0x4800 20FC	0b00	0b00000	0b1	0b000	0b01	0b111
CONTROL_PADCONF_DSS_DATA18[15:0]	dss_data18	0x4800 2100	0b00	0b00000	0b1	0b000	0b01	0b111
CONTROL_PADCONF_DSS_DATA18[31:16]	dss_data19	0x4800 2100	0b00	0b00000	0b1	0b000	0b01	0b111
CONTROL_PADCONF_DSS_DATA20[15:0]	dss_data20	0x4800 2104	0b00	0b00000	0b1	0b000	0b11	0b111
CONTROL_PADCONF_DSS_DATA20[31:16]	dss_data21	0x4800 2104	0b00	0b00000	0b1	0b000	0b01	0b111
CONTROL_PADCONF_DSS_DATA22[15:0]	dss_data22	0x4800 2108	0b00	0b00000	0b1	0b000	0b01	0b111
CONTROL_PADCONF_DSS_DATA22[31:16]	dss_data23	0x4800 2108	0b00	0b00000	0b1	0b000	0b01	0b111
CONTROL_PADCONF_CAM_HS[15:0]	cam_hs	0x4800 210C	0b00	0b00000	0b1	0b000	0b01	0b111
CONTROL_PADCONF_CAM_HS[31:16]	cam_vs	0x4800 210C	0b00	0b00000	0b1	0b000	0b01	0b111
CONTROL_PADCONF_CAM_XCLKA[15:0]	cam_xclka	0x4800 2110	0b00	0b00000	0b1	0b000	0b01	0b111
CONTROL_PADCONF_CAM_XCLKA[31:16]	cam_pclk	0x4800 2110	0b00	0b00000	0b1	0b000	0b01	0b111
CONTROL_PADCONF_CAM_FLD[15:0]	cam_fld	0x4800 2114	0b00	0b00000	0b1	0b000	0b01	0b111
CONTROL_PADCONF_CAM_FLD[31:16]	cam_d0	0x4800 2114	0b00	0b00000	0b1	0b000	0b01	0b111
CONTROL_PADCONF_CAM_D1[15:0]	cam_d1	0x4800 2118	0b00	0b00000	0b1	0b000	0b01	0b111
CONTROL_PADCONF_CAM_D1[31:16]	cam_d2	0x4800 2118	0b00	0b00000	0b1	0b000	0b01	0b111
CONTROL_PADCONF_CAM_D3[15:0]	cam_d3	0x4800 211C	0b00	0b00000	0b1	0b000	0b01	0b111
CONTROL_PADCONF_CAM_D3[31:16]	cam_d4	0x4800 211C	0b00	0b00000	0b1	0b000	0b01	0b111
CONTROL_PADCONF_CAM_D5[15:0]	cam_d5	0x4800 2120	0b00	0b00000	0b1	0b000	0b01	0b111
CONTROL_PADCONF_CAM_D5[31:16]	cam_d6	0x4800 2120	0b00	0b00000	0b1	0b000	0b01	0b111
CONTROL_PADCONF_CAM_D7[15:0]	cam_d7	0x4800 2124	0b00	0b00000	0b1	0b000	0b01	0b111
CONTROL_PADCONF_CAM_D7[31:16]	cam_d8	0x4800 2124	0b00	0b00000	0b1	0b000	0b01	0b111
CONTROL_PADCONF_CAM_D9[15:0]	cam_d9	0x4800 2128	0b00	0b00000	0b1	0b000	0b01	0b111
CONTROL_PADCONF_CAM_D9[31:16]	cam_d10	0x4800 2128	0b00	0b00000	0b1	0b000	0b01	0b111
CONTROL_PADCONF_CAM_D11[15:0]	cam_d11	0x4800 212C	0b00	0b00000	0b1	0b000	0b01	0b111
CONTROL_PADCONF_CAM_D11[31:16]	cam_xclkb	0x4800 212C	0b00	0b00000	0b1	0b000	0b01	0b111
CONTROL_PADCONF_CAM_WEN[15:0]	cam_wen	0x4800 2130	0b00	0b00000	0b1	0b000	0b01	0b111
CONTROL_PADCONF_CAM_WEN[31:16]	cam_strobe	0x4800 2130	0b00	0b00000	0b1	0b000	0b01	0b111
CONTROL_PADCONF_CSI2_DX0[15:0]	csi2_dx0	0x4800 2134	0b00	0b00000	0b1	0b000	0b01	0b111
CONTROL_PADCONF_CSI2_DX0[31:16]	csi2_dy0	0x4800 2134	0b00	0b00000	0b1	0b000	0b01	0b111

Table 7-75. CONTROL_PADCONF_CAPABILITIES (continued)

REGISTER NAME	Pad Name	Physical Address	WakeUpx	OffMode	Input Enable	Reserved	PU/PD	MuxMode
CONTROL_PADCONF_CSI2_DX1[15:0]	csi2_dx1	0x4800 2138	0b00	0b00000	0b1	0b000	0b01	0b111
CONTROL_PADCONF_CSI2_DX1[31:16]	csi2_dy1	0x4800 2138	0b00	0b00000	0b1	0b000	0b01	0b111
CONTROL_PADCONF_MCBSP2_FSX[15:0]	mcbsp2_fsx	0x4800 213C	0b00	0b00000	0b1	0b000	0b01	0b111
CONTROL_PADCONF_MCBSP2_FSX[31:16]	mcbsp2_clkx	0x4800 213C	0b00	0b00000	0b1	0b000	0b01	0b111
CONTROL_PADCONF_MCBSP2_DR[15:0]	mcbsp2_dr	0x4800 2140	0b00	0b00000	0b1	0b000	0b01	0b111
CONTROL_PADCONF_MCBSP2_DR[31:16]	mcbsp2_dx	0x4800 2140	0b00	0b00000	0b1	0b000	0b01	0b111
CONTROL_PADCONF_MMC1_CLK[15:0]	mmc1_clk	0x4800 2144	0b00	0b00000	0b1	0b000	0b01	0b111
CONTROL_PADCONF_MMC1_CLK[31:16]	mmc1_cmd	0x4800 2144	0b00	0b00000	0b1	0b000	0b01	0b111
CONTROL_PADCONF_MMC1_DAT0[15:0]	mmc1_dat0	0x4800 2148	0b00	0b00000	0b1	0b000	0b01	0b111
CONTROL_PADCONF_MMC1_DAT0[31:16]	mmc1_dat1	0x4800 2148	0b00	0b00000	0b1	0b000	0b01	0b111
CONTROL_PADCONF_MMC1_DAT2[15:0]	mmc1_dat2	0x4800 214C	0b00	0b00000	0b1	0b000	0b01	0b111
CONTROL_PADCONF_MMC1_DAT2[31:16]	mmc1_dat3	0x4800 214C	0b00	0b00000	0b1	0b000	0b01	0b111
CONTROL_PADCONF_MMC1_DAT4[15:0]	mmc1_dat4	0x4800 2150	0b00	0b00000	0b1	0b000	0b01	0b111
CONTROL_PADCONF_MMC1_DAT4[31:16]	mmc1_dat5	0x4800 2150	0b00	0b00000	0b1	0b000	0b01	0b111
CONTROL_PADCONF_MMC1_DAT6[15:0]	mmc1_dat6	0x4800 2154	0b00	0b00000	0b1	0b000	0b01	0b111
CONTROL_PADCONF_MMC1_DAT6[31:16]	mmc1_dat7	0x4800 2154	0b00	0b00000	0b1	0b000	0b01	0b111
CONTROL_PADCONF_MMC2_CLK[15:0]	mmc2_clk	0x4800 2158	0b00	0b00000	0b1	0b000	0b01	0b111
CONTROL_PADCONF_MMC2_CLK[31:16]	mmc2_cmd	0x4800 2158	0b00	0b00000	0b1	0b000	0b11	0b111
CONTROL_PADCONF_MMC2_DAT0[15:0]	mmc2_dat0	0x4800 215C	0b00	0b00000	0b1	0b000	0b11	0b111
CONTROL_PADCONF_MMC2_DAT0[31:16]	mmc2_dat1	0x4800 215C	0b00	0b00000	0b1	0b000	0b11	0b111
CONTROL_PADCONF_MMC2_DAT2[15:0]	mmc2_dat2	0x4800 2160	0b00	0b00000	0b1	0b000	0b11	0b111
CONTROL_PADCONF_MMC2_DAT2[31:16]	mmc2_dat3	0x4800 2160	0b00	0b00000	0b1	0b000	0b11	0b111
CONTROL_PADCONF_MMC2_DAT4[15:0]	mmc2_dat4	0x4800 2164	0b00	0b00000	0b1	0b000	0b01	0b111
CONTROL_PADCONF_MMC2_DAT4[31:16]	mmc2_dat5	0x4800 2164	0b00	0b00000	0b1	0b000	0b01	0b111
CONTROL_PADCONF_MMC2_DAT6[15:0]	mmc2_dat6	0x4800 2168	0b00	0b00000	0b1	0b000	0b01	0b111
CONTROL_PADCONF_MMC2_DAT6[31:16]	mmc2_dat7	0x4800 2168	0b00	0b00000	0b1	0b000	0b01	0b111
CONTROL_PADCONF_MCBSP3_DX[15:0]	mcbsp3_dx	0x4800 216C	0b00	0b00000	0b1	0b000	0b01	0b111
CONTROL_PADCONF_MCBSP3_DX[31:16]	mcbsp3_dr	0x4800 216C	0b00	0b00000	0b1	0b000	0b01	0b111
CONTROL_PADCONF_MCBSP3_CLKX[15:0]	mcbsp3_clkx	0x4800 2170	0b00	0b00000	0b1	0b000	0b01	0b111
CONTROL_PADCONF_MCBSP3_CLKX[31:16]	mcbsp3_fsx	0x4800 2170	0b00	0b00000	0b1	0b000	0b01	0b111
CONTROL_PADCONF_UART2_CTS[15:0]	uart2_cts	0x4800 2174	0b00	0b00000	0b1	0b000	0b11	0b111
CONTROL_PADCONF_UART2_CTS[31:16]	uart2_rts	0x4800 2174	0b00	0b00000	0b1	0b000	0b11	0b111
CONTROL_PADCONF_UART2_TX[15:0]	uart2_tx	0x4800 2178	0b00	0b00000	0b1	0b000	0b11	0b111

Table 7-75. CONTROL_PADCONF_CAPABILITIES (continued)

REGISTER NAME	Pad Name	Physical Address	WakeUpx	OffMode	Input Enable	Reserved	PU/PD	MuxMode
CONTROL_PADCONF_UART2_TX[31:16]	uart2_rx	0x4800 2178	0b00	0b00000	0b1	0b000	0b11	0b111
CONTROL_PADCONF_UART1_TX[15:0]	uart1_tx	0x4800 217C	0b00	0b00000	0b1	0b000	0b01	0b111
CONTROL_PADCONF_UART1_TX[31:16]	uart1_rts	0x4800 217C	0b00	0b00000	0b1	0b000	0b01	0b111
CONTROL_PADCONF_UART1_CTS[15:0]	uart1_cts	0x4800 2180	0b00	0b00000	0b1	0b000	0b01	0b111
CONTROL_PADCONF_UART1_CTS[31:16]	uart1_rx	0x4800 2180	0b00	0b00000	0b1	0b000	0b01	0b111
CONTROL_PADCONF_MCBSP4_CLKX[15:0]	mcbsp4_clkx	0x4800 2184	0b00	0b00000	0b1	0b000	0b01	0b111
CONTROL_PADCONF_MCBSP4_CLKX[31:16]	mcbsp4_dr	0x4800 2184	0b00	0b00000	0b1	0b000	0b01	0b111
CONTROL_PADCONF_MCBSP4_DX[15:0]	mcbsp4_dx	0x4800 2188	0b00	0b00000	0b1	0b000	0b01	0b111
CONTROL_PADCONF_MCBSP4_DX[31:16]	mcbsp4_fsx	0x4800 2188	0b00	0b00000	0b1	0b000	0b01	0b111
CONTROL_PADCONF_MCBSP1_CLKR[15:0]	mcbsp1_clkr	0x4800 218C	0b00	0b00000	0b1	0b000	0b01	0b111
CONTROL_PADCONF_MCBSP1_CLKR[31:16]	mcbsp1_fsr	0x4800 218C	0b00	0b00000	0b1	0b000	0b01	0b111
CONTROL_PADCONF_MCBSP1_DX[15:0]	mcbsp1_dx	0x4800 2190	0b00	0b00000	0b1	0b000	0b01	0b111
CONTROL_PADCONF_MCBSP1_DX[31:16]	mcbsp1_dr	0x4800 2190	0b00	0b00000	0b1	0b000	0b01	0b111
CONTROL_PADCONF_MCBSP_CLKS[15:0]	mcbsp_clks	0x4800 2194	0b00	0b00000	0b1	0b000	0b01	0b111
CONTROL_PADCONF_MCBSP_CLKS[31:16]	mcbsp1_fsx	0x4800 2194	0b00	0b00000	0b1	0b000	0b01	0b111
CONTROL_PADCONF_MCBSP1_CLKX[15:0]	mcbsp1_clkx	0x4800 2198	0b00	0b00000	0b1	0b000	0b01	0b111
CONTROL_PADCONF_MCBSP1_CLKX[31:16]	uart3_cts_rctx	0x4800 2198	0b00	0b00000	0b1	0b000	0b11	0b111
CONTROL_PADCONF_UART3_RTS_SD[15:0]	uart3_rts_sd	0x4800 219C	0b00	0b00000	0b1	0b000	0b11	0b111
CONTROL_PADCONF_UART3_RTS_SD[31:16]	uart3_rx_irrx	0x4800 219C	0b00	0b00000	0b1	0b000	0b11	0b111
CONTROL_PADCONF_UART3_TX_IRTX[15:0]	uart3_tx_irtx	0x4800 21A0	0b00	0b00000	0b1	0b000	0b11	0b111
CONTROL_PADCONF_UART3_TX_IRTX[31:16]	hsusb0_clk	0x4800 21A0	0b00	0b00000	0b1	0b000	0b01	0b111
CONTROL_PADCONF_HSUSB0_STP[15:0]	hsusb0_stp	0x4800 21A4	0b00	0b00000	0b1	0b000	0b11	0b111
CONTROL_PADCONF_HSUSB0_STP[31:16]	hsusb0_dir	0x4800 21A4	0b00	0b00000	0b1	0b000	0b01	0b111
CONTROL_PADCONF_HSUSB0_NXT[15:0]	hsusb0_nxt	0x4800 21A8	0b00	0b00000	0b1	0b000	0b01	0b111
CONTROL_PADCONF_HSUSB0_NXT[31:16]	hsusb0_data0	0x4800 21A8	0b00	0b00000	0b1	0b000	0b01	0b111
CONTROL_PADCONF_HSUSB0_DATA1[15:0]	hsusb0_data1	0x4800 21AC	0b00	0b00000	0b1	0b000	0b01	0b111
CONTROL_PADCONF_HSUSB0_DATA1[31:16]	hsusb0_data2	0x4800 21AC	0b00	0b00000	0b1	0b000	0b01	0b111
CONTROL_PADCONF_HSUSB0_DATA3[15:0]	hsusb0_data3	0x4800 21B0	0b00	0b00000	0b1	0b000	0b01	0b111
CONTROL_PADCONF_HSUSB0_DATA3[31:16]	hsusb0_data4	0x4800 21B0	0b00	0b00000	0b1	0b000	0b01	0b111
CONTROL_PADCONF_HSUSB0_DATA5[15:0]	hsusb0_data5	0x4800 21B4	0b00	0b00000	0b1	0b000	0b01	0b111
CONTROL_PADCONF_HSUSB0_DATA5[31:16]	hsusb0_data6	0x4800 21B4	0b00	0b00000	0b1	0b000	0b01	0b111
CONTROL_PADCONF_HSUSB0_DATA7[15:0]	hsusb0_data7	0x4800 21B8	0b00	0b00000	0b1	0b000	0b01	0b111
CONTROL_PADCONF_HSUSB0_DATA7[31:16]	i2c1_scl	0x4800 21B8	0b00	0b00000	0b1	0b000	0b11	---

Table 7-75. CONTROL_PADCONF_CAPABILITIES (continued)

REGISTER NAME	Pad Name	Physical Address	WakeUpx	OffMode	Input Enable	Reserved	PU/PD	MuxMode
CONTROL_PADCONF_I2C1_SDA[15:0]	i2c1_sda	0x4800 21BC	0b00	0b00000	0b1	0b000	0b11	---
CONTROL_PADCONF_I2C1_SDA[31:16]	i2c2_scl	0x4800 21BC	0b00	0b00000	0b1	0b000	0b11	0b111
CONTROL_PADCONF_I2C2_SDA[15:0]	i2c2_sda	0x4800 21C0	0b00	0b00000	0b1	0b000	0b11	0b111
CONTROL_PADCONF_I2C2_SDA[31:16]	i2c3_scl	0x4800 21C0	0b00	0b00000	0b1	0b000	0b11	0b111
CONTROL_PADCONF_I2C3_SDA[15:0]	i2c3_sda	0x4800 21C4	0b00	0b00000	0b1	0b000	0b11	0b111
CONTROL_PADCONF_I2C3_SDA[31:16]	hdq_sio	0x4800 21C4	0b00	0b00000	0b1	0b000	0b11	0b111
CONTROL_PADCONF_MCSPI1_CLK[15:0]	mcspi1_clk	0x4800 21C8	0b00	0b00000	0b1	0b000	0b01	0b111
CONTROL_PADCONF_MCSPI1_CLK[31:16	mcspi1_simo	0x4800 21C8	0b00	0b00000	0b1	0b000	0b01	0b111
CONTROL_PADCONF_MCSPI1_SOMI[15:0]	mcspi1_somi	0x4800 21CC	0b00	0b00000	0b1	0b000	0b01	0b111
CONTROL_PADCONF_MCSPI1_SOMI[31:16]	mcspi1_cs0	0x4800 21CC	0b00	0b00000	0b1	0b000	0b11	0b111
CONTROL_PADCONF_MCSPI1_CS1[15:0]	mcspi1_cs1	0x4800 21D0	0b00	0b00000	0b1	0b000	0b11	0b111
CONTROL_PADCONF_MCSPI1_CS1[31:16]	mcspi1_cs2	0x4800 21D0	0b00	0b00000	0b1	0b000	0b11	0b111
CONTROL_PADCONF_MCSPI1_CS3[15:0]	mcspi1_cs3	0x4800 21D4	0b00	0b00000	0b1	0b000	0b11	0b111
CONTROL_PADCONF_MCSPI1_CS3[31:16]	mcspi2_clk	0x4800 21D4	0b00	0b00000	0b1	0b000	0b01	0b111
CONTROL_PADCONF_MCSPI2_SIMO[15:0]	mcspi2_simo	0x4800 21D8	0b00	0b00000	0b1	0b000	0b01	0b111
CONTROL_PADCONF_MCSPI2_SIMO[31:16]	mcspi2_somi	0x4800 21D8	0b00	0b00000	0b1	0b000	0b01	0b111
CONTROL_PADCONF_MCSPI2_CS0[15:0]	mcspi2_cs0	0x4800 21DC	0b00	0b00000	0b1	0b000	0b11	0b111
CONTROL_PADCONF_MCSPI2_CS0[31:16]	mcspi2_cs1	0x4800 21DC	0b00	0b00000	0b1	0b000	0b01	0b111
CONTROL_PADCONF_SYS_NIRQ[15:0]	sys_nirq	0x4800 21E0	0b00	0b00000	0b1	0b000	0b11	0b111
CONTROL_PADCONF_SYS_NIRQ[31:16]	sys_clkout2	0x4800 21E0	0b00	0b00000	0b1	0b000	0b01	0b111
CONTROL_PADCONF_SAD2D_MCAD0[15:0]	sad2d_mcad0	0x4800 21E4	--	0b00000	0b1	0b000	0b01	0b000
CONTROL_PADCONF_SAD2D_MCAD0[31:16]	sad2d_mcad1	0x4800 21E4	--	0b00000	0b1	0b000	0b01	0b000
CONTROL_PADCONF_SAD2D_MCAD2[15:0]	sad2d_mcad2	0x4800 21E8	--	0b00000	0b1	0b000	0b01	0b000
CONTROL_PADCONF_SAD2D_MCAD2[31:16]	sad2d_mcad3	0x4800 21E8	--	0b00000	0b1	0b000	0b01	0b000
CONTROL_PADCONF_SAD2D_MCAD4[15:0]	sad2d_mcad4	0x4800 21EC	--	0b00000	0b1	0b000	0b01	0b000
CONTROL_PADCONF_SAD2D_MCAD4[31:16]	sad2d_mcad5	0x4800 21EC	--	0b00000	0b1	0b000	0b01	0b000
CONTROL_PADCONF_SAD2D_MCAD6[15:0]	sad2d_mcad6	0x4800 21F0	--	0b00000	0b1	0b000	0b01	0b000
CONTROL_PADCONF_SAD2D_MCAD6[31:16]	sad2d_mcad7	0x4800 21F0	--	0b00000	0b1	0b000	0b01	0b000
CONTROL_PADCONF_SAD2D_MCAD8[15:0]	sad2d_mcad8	0x4800 21F4	--	0b00000	0b1	0b000	0b01	0b000
CONTROL_PADCONF_SAD2D_MCAD8[31:16]	sad2d_mcad9	0x4800 21F4	--	0b00000	0b1	0b000	0b01	0b000
CONTROL_PADCONF_SAD2D_MCAD10[15:0]	sad2d_mcad10	0x4800 21F8	--	0b00000	0b1	0b000	0b01	0b000
CONTROL_PADCONF_SAD2D_MCAD10[31:16]	sad2d_mcad11	0x4800 21F8	--	0b00000	0b1	0b000	0b01	0b000
CONTROL_PADCONF_SAD2D_MCAD12[15:0]	sad2d_mcad12	0x4800 21FC	--	0b00000	0b1	0b000	0b01	0b000

Table 7-75. CONTROL_PADCONF_CAPABILITIES (continued)

REGISTER NAME	Pad Name	Physical Address	WakeUpx	OffMode	Input Enable	Reserved	PU/PD	MuxMode
CONTROL_PADCONF_SAD2D_MCAD12[31:16]	sad2d_mcad13	0x4800 21FC	--	0b00000	0b1	0b000	0b01	0b000
CONTROL_PADCONF_SAD2D_MCAD14[15:0]	sad2d_mcad14	0x4800 2200	--	0b00000	0b1	0b000	0b01	0b000
CONTROL_PADCONF_SAD2D_MCAD14[31:16]	sad2d_mcad15	0x4800 2200	--	0b00000	0b1	0b000	0b01	0b000
CONTROL_PADCONF_SAD2D_MCAD16[15:0]	sad2d_mcad16	0x4800 2204	--	0b00000	0b1	0b000	0b01	0b000
CONTROL_PADCONF_SAD2D_MCAD16[31:16]	sad2d_mcad17	0x4800 2204	--	0b00000	0b1	0b000	0b01	0b000
CONTROL_PADCONF_SAD2D_MCAD18[15:0]	sad2d_mcad18	0x4800 2208	--	0b00000	0b1	0b000	0b01	0b000
CONTROL_PADCONF_SAD2D_MCAD18[31:16]	sad2d_mcad19	0x4800 2208	--	0b00000	0b1	0b000	0b01	0b000
CONTROL_PADCONF_SAD2D_MCAD20[15:0]	sad2d_mcad20	0x4800 220C	--	0b00000	0b1	0b000	0b01	0b000
CONTROL_PADCONF_SAD2D_MCAD20[31:16]	sad2d_mcad21	0x4800 220C	--	0b00000	0b1	0b000	0b01	0b000
CONTROL_PADCONF_SAD2D_MCAD22[15:0]	sad2d_mcad22	0x4800 2210	--	0b00000	0b1	0b000	0b01	0b000
CONTROL_PADCONF_SAD2D_MCAD22[31:16]	sad2d_mcad23	0x4800 2210	--	0b00000	0b1	0b000	0b01	0b000
CONTROL_PADCONF_SAD2D_MCAD24[15:0]	sad2d_mcad24	0x4800 2214	--	0b00000	0b1	0b000	0b01	0b000
CONTROL_PADCONF_SAD2D_MCAD24[31:16]	sad2d_mcad25	0x4800 2214	--	0b00000	0b1	0b000	0b01	0b000
CONTROL_PADCONF_SAD2D_MCAD26[15:0]	sad2d_mcad26	0x4800 2218	--	0b00000	0b1	0b000	0b01	0b000
CONTROL_PADCONF_SAD2D_MCAD26[31:16]	sad2d_mcad27	0x4800 2218	--	0b00000	0b1	0b000	0b01	0b000
CONTROL_PADCONF_SAD2D_MCAD28[15:0]	sad2d_mcad28	0x4800 221C	--	0b00000	0b1	0b000	0b01	0b000
CONTROL_PADCONF_SAD2D_MCAD28[31:16]	sad2d_mcad29	0x4800 221C	--	0b00000	0b1	0b000	0b01	0b000
CONTROL_PADCONF_SAD2D_MCAD30[15:0]	sad2d_mcad30	0x4800 2220	--	0b00000	0b1	0b000	0b01	0b000
CONTROL_PADCONF_SAD2D_MCAD30[31:16]	sad2d_mcad31	0x4800 2220	--	0b00000	0b1	0b000	0b01	0b000
CONTROL_PADCONF_SAD2D_MCAD32[15:0]	sad2d_mcad32	0x4800 2224	--	0b00000	0b1	0b000	0b01	0b000
CONTROL_PADCONF_SAD2D_MCAD32[31:16]	sad2d_mcad33	0x4800 2224	--	0b00000	0b1	0b000	0b01	0b000
CONTROL_PADCONF_SAD2D_MCAD34[15:0]	sad2d_mcad34	0x4800 2228	--	0b00000	0b1	0b000	0b01	0b000
CONTROL_PADCONF_SAD2D_MCAD34[31:16]	sad2d_mcad35	0x4800 2228	--	0b00000	-	0b000	0b01	0b000
CONTROL_PADCONF_SAD2D_MCAD36[15:0]	sad2d_mcad36	0x4800 222C	--	0b00000	0b1	0b000	0b01	0b000
CONTROL_PADCONF_SAD2D_MCAD36[31:16]	sad2d_clk26mi	0x4800 222C	--	0b--000	0b1	0b000	0b01	0b000
CONTROL_PADCONF_SAD2D_NRESPWRON[15:0]	sad2d_nrespwron	0x4800 2230	--	0b--000	0b1	0b000	--	---
CONTROL_PADCONF_SAD2D_NRESPWRON[31:16]	sad2d_nreswarm	0x4800 2230	0b00	0b00000	0b1	0b000	0b11	---
CONTROL_PADCONF_SAD2D_ARMNIRQ[15:0]	sad2d_armnirq	0x4800 2234	0b00	0b--000	0b1	0b000	--	---
CONTROL_PADCONF_SAD2D_ARMNIRQ[31:16]	sad2d_umafiq	0x4800 2234	0b00	0b--000	0b1	0b000	--	---
CONTROL_PADCONF_SAD2D_SPINT[15:0]	sad2d_spint	0x4800 2238	0b00	0b00--0	0b1	0b000	0b01	0b000
CONTROL_PADCONF_SAD2D_SPINT[31:16]	sad2d_frint	0x4800 2238	0b00	0b00--0	0b1	0b000	0b01	0b000
CONTROL_PADCONF_SAD2D_DMAREQ0[15:0]	sad2d_dmareq0	0x4800 223C	--	0b--000	0b1	0b000	--	---
CONTROL_PADCONF_SAD2D_DMAREQ0[31:16]	sad2d_dmareq1	0x4800 223C	--	0b--000	0b1	0b000	--	---

Table 7-75. CONTROL_PADCONF_CAPABILITIES (continued)

REGISTER NAME	Pad Name	Physical Address	WakeUpx	OffMode	Input Enable	Reserved	PU/PD	MuxMode
CONTROL_PADCONF_SAD2D_DMAREQ2[15:0]	sad2d_dmareq2	0x4800 2240	--	0b--000	0b1	0b000	--	---
CONTROL_PADCONF_SAD2D_DMAREQ2[31:16]	sad2d_dmareq3	0x4800 2240	--	0b--000	0b1	0b000	--	---
CONTROL_PADCONF_SAD2D_NTRST[15:0]	sad2d_ntrst	0x4800 2244	--	0b--000	0b1	0b000	--	---
CONTROL_PADCONF_SAD2D_NTRST[31:16]	sad2d_tdi	0x4800 2244	--	0b--000	0b1	0b000	--	---
CONTROL_PADCONF_SAD2D_TDO[15:0]	sad2d_tdo	0x4800 2248	--	0b00--0	0b1	0b000	0b01	---
CONTROL_PADCONF_SAD2D_TDO[31:16]	sad2d_tms	0x4800 2248	--	0b--000	0b1	0b000	--	---
CONTROL_PADCONF_SAD2D_TCK[15:0]	sad2d_tck	0x4800 224C	--	0b--000	0b1	0b000	--	---
CONTROL_PADCONF_SAD2D_TCK[31:16]	sad2d_rtck	0x4800 224C	--	0b00--0	0b1	0b000	0b01	---
CONTROL_PADCONF_SAD2D_MSTDBY[15:0]	sad2d_mstdby	0x4800 2250	0b00	0b00--0	0b1	0b000	0b11	---
CONTROL_PADCONF_SAD2D_MSTDBY[31:16]	sad2d_idlereq	0x4800 2250	--	0b--000	0b1	0b000	--	---
CONTROL_PADCONF_SAD2D_IDLEACK[15:0]	sad2d_idleack	0x4800 2254	--	0b00--0	0b1	0b000	0b11	---
CONTROL_PADCONF_SAD2D_IDLEACK[31:16]	sad2d_mwrite	0x4800 2254	--	0b00000	0b1	0b000	0b01	0b000
CONTROL_PADCONF_SAD2D_SWRITE[15:0]	sad2d_swrite	0x4800 2258	--	0b00000	0b1	0b000	0b00	---
CONTROL_PADCONF_SAD2D_SWRITE[31:16]	sad2d_mread	0x4800 2258	--	0b00--0	0b1	0b000	0b01	---
CONTROL_PADCONF_SAD2D_SREAD[15:0]	sad2d_sread	0x4800 225C	--	0b00000	0b1	0b000	0b00	---
CONTROL_PADCONF_SAD2D_SREAD[31:16]	sad2d_mbusflag	0x4800 225C	--	0b00000	0b1	0b000	0b01	---
CONTROL_PADCONF_SAD2D_SBUSFLAG[15:0]	sad2d_sbusflag	0x4800 2260	--	0b--000	0b1	0b000	--	---
CONTROL_PADCONF_SAD2D_SBUSFLAG[31:16]	sdrc_cke0	0x4800 2260	--	-----	0b1	0b000	0b11	0b111
CONTROL_PADCONF_SDRC_CKE1[15:0]	sdrc_cke1	0x4800 2264	--	-----	0b1	0b000	0b11	0b111
CONTROL_PADCONF_ETK_CLK[15:0]	etk_clk	0x4800 25D8	0b00	0b00000	0b1	0b000	0b11	0b100
CONTROL_PADCONF_ETK_CLK[31:16]	etk_ctl	0x4800 25D8	0b00	0b00000	0b1	0b000	0b11	0b100
CONTROL_PADCONF_ETK_D0[15:0]	etk_d0	0x4800 25DC	0b00	0b00000	0b1	0b000	0b11	0b100
CONTROL_PADCONF_ETK_D0[31:16]	etk_d1	0x4800 25DC	0b00	0b00000	0b1	0b000	0b11	0b100
CONTROL_PADCONF_ETK_D2[15:0]	etk_d2	0x4800 25E0	0b00	0b00000	0b1	0b000	0b11	0b100
CONTROL_PADCONF_ETK_D2[31:16]	etk_d3	0x4800 25E0	0b00	0b00000	0b1	0b000	0b11	0b100
CONTROL_PADCONF_ETK_D4[15:0]	etk_d4	0x4800 25E4	0b00	0b00000	0b1	0b000	0b01	0b100
CONTROL_PADCONF_ETK_D4[31:16]	etk_d5	0x4800 25E4	0b00	0b00000	0b1	0b000	0b01	0b100
CONTROL_PADCONF_ETK_D6[15:0]	etk_d6	0x4800 25E8	0b00	0b00000	0b1	0b000	0b01	0b100
CONTROL_PADCONF_ETK_D6[31:16]	etk_d7	0x4800 25E8	0b00	0b00000	0b1	0b000	0b01	0b100
CONTROL_PADCONF_ETK_D8[15:0]	etk_d8	0x4800 25EC	0b00	0b00000	0b1	0b000	0b01	0b100
CONTROL_PADCONF_ETK_D8[31:16]	etk_d9	0x4800 25EC	0b00	0b00000	0b1	0b000	0b01	0b100
CONTROL_PADCONF_ETK_D10[15:0]	etk_d10	0x4800 25F0	0b00	0b00000	0b1	0b000	0b01	0b100
CONTROL_PADCONF_ETK_D10[31:16]	etk_d11	0x4800 25F0	0b00	0b00000	0b1	0b000	0b01	0b100

Table 7-75. CONTROL_PADCONF_CAPABILITIES (continued)

REGISTER NAME	Pad Name	Physical Address	WakeUpx	OffMode	Input Enable	Reserved	PU/PD	MuxMode
CONTROL_PADCONF_ETK_D12[15:0]	etk_d12	0x4800 25F4	0b00	0b00000	0b1	0b000	0b01	0b100
CONTROL_PADCONF_ETK_D12[31:16]	etk_d13	0x4800 25F4	0b00	0b00000	0b1	0b000	0b01	0b100
CONTROL_PADCONF_ETK_D14[15:0]	etk_d14	0x4800 25F8	0b00	0b00000	0b1	0b000	0b01	0b100
CONTROL_PADCONF_ETK_D14[31:16]	etk_d15	0x4800 25F8	0b00	0b00000	0b1	0b000	0b01	0b100

7.6.3.3 GENERAL Register Description

Table 7-76 through Table 7-236 describe the GENERAL registers bits.

Table 7-76. CONTROL_PADCONF_OFF

Address Offset	0x0000 0000		
Physical address	0x4800 2270	**Instance**	GENERAL
Description	Off mode pad configuration register		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																													WKUPCTRLCLOCKDIV	STARTSAVE	FORCEOFFMODEEN

Bits	Field Name	Description	Type	Reset
31:3	RESERVED	Read returns reset value.	R	0x0000000
2	WKUPCTRLCLOCKDIV	Wkup_ctrl module clock divider	R/W	0x0
		0x0: Clock is divided by 4.		
		0x1: Clock is divided by 2.		
1	STARTSAVE	Start pad configuration registers save mechanism	R/W	0x0
		0x0: Save is not started.		
		0x1: Save is running. This bit is auto-cleared after 8 interface clock cycles.		
0	FORCEOFFMODEEN	Force OFF mode active	R/W	0x0
		0x0: OFF mode is not forced active.		
		0x1: OFF mode is forced active.		

Table 7-77. Register Call Summary for Register CONTROL_PADCONF_OFF

SCM Integration
- Clock: [0]
- Power Management: [1]

SCM Functional Description
- System Off Mode: [2] [3]

SCM Programming Model
- Off Mode Preliminary Settings: [4]

SCM Register Manual
- SCM Register Summary: [5]

Table 7-78. CONTROL_DEVCONF0

Address Offset	0x0000 0004		
Physical address	0x4800 2274	**Instance**	GENERAL
Description	Static device configuration register-0. Module dedicated functions		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SPARE					SPARE	SPARE	MMCSDIO1ADPCLKISEL	RESERVED																MCBSP2_CLKS	RESERVED	MCBSP1_FSR	MCBSP1_CLKR	MCBSP1_CLKS	SENSDMAREQ1	SENSDMAREQ0		

Bits	Field Name	Description	Type	Reset
31:27	SPARE	Spare bits	R/W	0x00
26	SPARE	Spare bit	R/W	0x1
25	SPARE	Spare bit	R/W	0x0
24	MMCSDIO1ADPCLKISEL	MMC/SDI/O Module Input Clock selection 0x0: Input clock is from the external pin 0x1: Internal loop-back, module input clock is copied from the module output clock	R/W	0x0
23:7	RESERVED	Read returns reset value	R	0x000
6	MCBSP2_CLKS	Select the CLKS input for the module McBSP2 Note : There are no external pins McBSP2_CLKR and McBSP2_FSR for the module McBSP2. For this module, CLKR input is from the pin McBSP2_CLKX and FSR input is from the pin McBSP2_FSX 0x0: CLKS is from the PRCM functional clock 0x1: CLKS is from the external pin McBSP_CLKS	R/W	0x0
5	RESERVED	Read returns reset value.	R	0
4	MCBSP1_FSR	Select the FSR input for the module McBSP1 0x0: FSR is from the pin McBSP1_FSR 0x1: FSR is from the pin McBSP1_FSX	R/W	0x0
3	MCBSP1_CLKR	Select the CLKR input for the module McBSP1 0x0: CLKR is from the pin McBSP1_CLKR 0x1: CLKR is from the pin McBSP1_CLKX	R/W	0x0
2	MCBSP1_CLKS	Select the CLKS input for the module McBSP1 0x0: CLKS is from the PRCM functional clock 0x1: CLKS is from the external pin McBSP_CLKS	R/W	0x0
1	SENSDMAREQ1	Set sensitivity on SYS.DMAREQ1 input pin 0x0: Level sensitivity 0x1: Edge sensitivity	R/W	0x0
0	SENSDMAREQ0	Set sensitivity on SYS.DMAREQ0 input pin 0x0: Level sensitivity 0x1: Edge sensitivity	R/W	0x0

Table 7-79. Register Call Summary for Register CONTROL_DEVCONF0

SCM Environment

- SCM Environment: [0]

Table 7-79. Register Call Summary for Register CONTROL_DEVCONF0 (continued)

SCM Functional Description • Static Device Configuration Registers: [1]
SCM Programming Model • McBSP1 Internal Clock: [2] [3] [4] • McBSP2 Internal Clock: [5] • MMC/SD/SDIO1 Module Input Clock Selection: [6] • Setting Sensitivity on sys_ndmareq[6:0] Input Pins: [7] [8]
SCM Register Manual • SCM Register Summary: [9]

Table 7-80. CONTROL_MSUSPENDMUX_0

Address Offset	0x0000 0020		
Physical address	0x4800 2290	**Instance**	GENERAL
Description	MSuspend Control register: control the use of MSuspend signals at module level		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								MCBSP2MSCTRL			MCBSP1MSCTRL			I2C2MSCTRL		I2C1MSCTRL				RESERVED											

Bits	Field Name	Description		Type	Reset
31:24	RESERVED	Read returns reset value.		R	0x00
23:21	MCBSP2MSCTRL	Control McBSP_2 sensitivity to MCU and/or DSP MSuspend signals		RW	0x0
		0x0:	No sensitivity: no MSuspend signal reaches the module		
		0x1:	Sensitivity to MCU MSuspend signals (DSP signal ignored)		
		0x2:	Sensitivity to DSP MSuspend signal (MCU signal ignored)		
		0x3:	Sensitivity to the logical ORed MCU and DSP MSuspend signals		
		0x4:	Sensitivity to the logical ANDed MCU and DSP MSuspend signals		
		0x5:	No sensitivity: no MSuspend signal reaches the module		
		0x6:	No sensitivity: no MSuspend signal reaches the module		
		0x7:	No sensitivity: no MSuspend signal reaches the module		

Bits	Field Name	Description	Type	Reset
20:18	MCBSP1MSCTRL	Control McBSP_1 sensitivity to MCU and/or DSP MSuspend signals 0x0: No sensitivity: no MSuspend signal reaches the module 0x1: Sensitivity to MCU MSuspend signals (DSP signal ignored) 0x2: Sensitivity to DSP MSuspend signal (MCU signal ignored) 0x3: Sensitivity to the logical ORed MCU and DSP MSuspend signals 0x4: Sensitivity to the logical ANDed MCU and DSP MSuspend signals 0x5: No sensitivity: no MSuspend signal reaches the module 0x6: No sensitivity: no MSuspend signal reaches the module 0x7: No sensitivity: no MSuspend signal reaches the module	RW	0x0
17:15	I2C2MSCTRL	Control I2C_2 sensitivity to MCU and/or DSP MSuspend signals 0x0: No sensitivity: no MSuspend signal reaches the module 0x1: Sensitivity to MCU MSuspend signals (DSP signal ignored) 0x2: Sensitivity to DSP MSuspend signal (MCU signal ignored) 0x3: Sensitivity to the logical ORed MCU and DSP MSuspend signals 0x4: Sensitivity to the logical ANDed MCU and DSP MSuspend signals 0x5: No sensitivity: no MSuspend signal reaches the module 0x6: No sensitivity: no MSuspend signal reaches the module 0x7: No sensitivity: no MSuspend signal reaches the module	RW	0x0
14:12	I2C1MSCTRL	Control I2C_1 sensitivity to MCU and/or DSP MSuspend signals 0x0: No sensitivity: no MSuspend signal reaches the module 0x1: Sensitivity to MCU MSuspend signals (DSP signal ignored) 0x2: Sensitivity to DSP MSuspend signal (MCU signal ignored) 0x3: Sensitivity to the logical ORed MCU and DSP MSuspend signals 0x4: Sensitivity to the logical ANDed MCU and DSP MSuspend signals 0x5: No sensitivity: no MSuspend signal reaches the module 0x6: No sensitivity: no MSuspend signal reaches the module 0x7: No sensitivity: no MSuspend signal reaches the module	RW	0x0
11:0	RESERVED	Read returns reset value.	R	0x00

Table 7-81. Register Call Summary for Register CONTROL_MSUSPENDMUX_0

SCM Functional Description

- MPU and/or DSP (IVA2.2) MSuspend Configuration Registers: [0]

SCM Register Manual

- SCM Register Summary: [1]

Table 7-82. CONTROL_MSUSPENDMUX_1

Address Offset	0x0000 0024		
Physical address	0x4800 2294	**Instance**	GENERAL
Description	MSuspend Control register : control the use of MSuspend signals at module level		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED		GPTM7MSCTRL			GPTM6MSCTRL			GPTM5MSCTRL			GPTM4MSCTRL			GPTM3MSCTRL		GPTM2MSCTRL			GPTM1MSCTRL				RESERVED							

Bits	Field Name	Description	Type	Reset
31:30	RESERVED	Read returns reset value.	R	0x0
29:27	GPTM7MSCTRL	Control General Purpose Timer 7 sensitivity to MCU and/or DSP MSuspend signals	RW	0x0
		0x0: No sensitivity: no MSuspend signal reaches the module		
		0x1: Sensitivity to MCU MSuspend signals (DSP signal ignored)		
		0x2: Sensitivity to DSP MSuspend signal (MCU signal ignored)		
		0x3: Sensitivity to the logical ORed MCU and DSP MSuspend signals		
		0x4: Sensitivity to the logical ANDed MCU and DSP MSuspend signals		
		0x5: No sensitivity: no MSuspend signal reaches the module		
		0x6: No sensitivity: no MSuspend signal reaches the module		
		0x7: No sensitivity: no MSuspend signal reaches the module		

Bits	Field Name	Description	Type	Reset
26:24	GPTM6MSCTRL	Control General Purpose Timer 6 sensitivity to MCU and/or DSP MSuspend signals 0x0: No sensitivity: no MSuspend signal reaches the module 0x1: Sensitivity to MCU MSuspend signals (DSP signal ignored) 0x2: Sensitivity to DSP MSuspend signal (MCU signal ignored) 0x3: Sensitivity to the logical ORed MCU and DSP MSuspend signals 0x4: Sensitivity to the logical ANDed MCU and DSP MSuspend signals 0x5: No sensitivity: no MSuspend signal reaches the module 0x6: No sensitivity: no MSuspend signal reaches the module 0x7: No sensitivity: no MSuspend signal reaches the module	RW	0x0
23:21	GPTM5MSCTRL	Control General Purpose Timer 5 sensitivity to MCU and/or DSP MSuspend signals 0x0: No sensitivity: no MSuspend signal reaches the module 0x1: Sensitivity to MCU MSuspend signals (DSP signal ignored) 0x2: Sensitivity to DSP MSuspend signal (MCU signal ignored) 0x3: Sensitivity to the logical ORed MCU and DSP MSuspend signals 0x4: Sensitivity to the logical ANDed MCU and DSP MSuspend signals 0x5: No sensitivity: no MSuspend signal reaches the module 0x6: No sensitivity: no MSuspend signal reaches the module 0x7: No sensitivity: no MSuspend signal reaches the module	RW	0x0
20:18	GPTM4MSCTRL	Control General Purpose Timer 4 sensitivity to MCU and/or DSP MSuspend signals 0x0: No sensitivity: no MSuspend signal reaches the module 0x1: Sensitivity to MCU MSuspend signals (DSP signal ignored) 0x2: Sensitivity to DSP MSuspend signal (MCU signal ignored) 0x3: Sensitivity to the logical ORed MCU and DSP MSuspend signals 0x4: Sensitivity to the logical ANDed MCU and DSP MSuspend signals 0x5: No sensitivity: no MSuspend signal reaches the module 0x6: No sensitivity: no MSuspend signal reaches the module 0x7: No sensitivity: no MSuspend signal reaches the module	RW	0x0

Bits	Field Name	Description	Type	Reset
17:15	GPTM3MSCTRL	Control General Purpose Timer 3 sensitivity to MCU and/or DSP MSuspend signals 0x0: No sensitivity: no MSuspend signal reaches the module 0x1: Sensitivity to MCU MSuspend signals (DSP signal ignored) 0x2: Sensitivity to DSP MSuspend signal (MCU signal ignored) 0x3: Sensitivity to the logical ORed MCU and DSP MSuspend signals 0x4: Sensitivity to the logical ANDed MCU and DSP MSuspend signals 0x5: No sensitivity: no MSuspend signal reaches the module 0x6: No sensitivity: no MSuspend signal reaches the module 0x7: No sensitivity: no MSuspend signal reaches the module	RW	0x0
14:12	GPTM2MSCTRL	Control General Purpose Timer 2 sensitivity to MCU and/or DSP MSuspend signals 0x0: No sensitivity: no MSuspend signal reaches the module 0x1: Sensitivity to MCU MSuspend signals (DSP signal ignored) 0x2: Sensitivity to DSP MSuspend signal (MCU signal ignored) 0x3: Sensitivity to the logical ORed MCU and DSP MSuspend signals 0x4: Sensitivity to the logical ANDed MCU and DSP MSuspend signals 0x5: No sensitivity: no MSuspend signal reaches the module 0x6: No sensitivity: no MSuspend signal reaches the module 0x7: No sensitivity: no MSuspend signal reaches the module	RW	0x0
11:9	GPTM1MSCTRL	Control General Purpose Timer 1 sensitivity to MCU and/or DSP MSuspend signals 0x0: No sensitivity: no MSuspend signal reaches the module 0x1: Sensitivity to MCU MSuspend signals (DSP signal ignored) 0x2: Sensitivity to DSP MSuspend signal (MCU signal ignored) 0x3: Sensitivity to the logical ORed MCU and DSP MSuspend signals 0x4: Sensitivity to the logical ANDed MCU and DSP MSuspend signals 0x5: No sensitivity: no MSuspend signal reaches the module 0x6: No sensitivity: no MSuspend signal reaches the module 0x7: No sensitivity: no MSuspend signal reaches the module	RW	0x0
8:0	RESERVED	Read returns reset value.	R	0x00

Table 7-83. Register Call Summary for Register CONTROL_MSUSPENDMUX_1

SCM Functional Description

- MPU and/or DSP (IVA2.2) MSuspend Configuration Registers: [0]

SCM Register Manual

- SCM Register Summary: [1]

Table 7-84. CONTROL_MSUSPENDMUX_2

Address Offset	0x0000 0028		
Physical Address	0x4800 2298	**Instance**	GENERAL
Description	MSuspend Control register : control the use of MSuspend signals at module level		
Type	RW		

31 30	29 28 27	26 25 24	23 22 21 20	19 18 17 16	15 14 13 12	11 10 9 8	7 6	5 4	3 2	1 0
RESERVED	SYNCTMMSCTRL	RESERVED	WD3MSCTRL	WD2MSCTRL	RESERVED	GPTM11MSCTRL	GPTM10MSCTRL	GPTM9MSCTRL	GPTM8MSCTRL	

Bits	Field Name	Description	Type	Reset
31:30	RESERVED	Read returns reset value.	R	0x0
29:27	SYNCTMMSCTRL	Control Sync Timer32K sensitivity to MCU and/or DSP MSuspend signals 0x0: No sensitivity: no MSuspend signal reaches the module 0x1: Sensitivity to MCU MSuspend signals (DSP signal ignored) 0x2: Sensitivity to DSP MSuspend signal (MCU signal ignored) 0x3: Sensitivity to the logical ORed MCU and DSP MSuspend signals 0x4: Sensitivity to the logical ANDed MCU and DSP MSuspend signals 0x5: No sensitivity: no MSuspend signal reaches the module 0x6: No sensitivity: no MSuspend signal reaches the module 0x7: No sensitivity: no MSuspend signal reaches the module	R/W	0x0
26:24	RESERVED	Read returns reset value.	R	0x0
23:21	WD3MSCTRL	Control Watch Dog 3 sensitivity to MCU and/or DSP MSuspend signals 0x0: No sensitivity: no MSuspend signal reaches the module 0x1: Sensitivity to MCU MSuspend signals (DSP signal ignored) 0x2: Sensitivity to DSP MSuspend signal (MCU signal ignored) 0x3: Sensitivity to the logical ORed MCU and DSP MSuspend signals 0x4: Sensitivity to the logical ANDed MCU and DSP MSuspend signals 0x5: No sensitivity: no MSuspend signal reaches the module 0x6: No sensitivity: no MSuspend signal reaches the module 0x7: No sensitivity: no MSuspend signal reaches the module	R/W	0x1

Bits	Field Name	Description	Type	Reset
20:18	WD2MSCTRL	Control Watch Dog 2 sensitivity to MCU and/or DSP MSuspend signals 0x0: No sensitivity: no MSuspend signal reaches the module 0x1: Sensitivity to MCU MSuspend signals (DSP signal ignored) 0x2: Sensitivity to DSP MSuspend signal (MCU signal ignored) 0x3: Sensitivity to the logical ORed MCU and DSP MSuspend signals 0x4: Sensitivity to the logical ANDed MCU and DSP MSuspend signals 0x5: No sensitivity: no MSuspend signal reaches the module 0x6: No sensitivity: no MSuspend signal reaches the module 0x7: No sensitivity: no MSuspend signal reaches the module	R/W	0x1
17:12	RESERVED	Reserved. Read returns reset value.	R	0x8
11:9	GPTM11MSCTRL	Control General Purpose Timer 11 sensitivity to MCU and/or DSP MSuspend signals 0x0: No sensitivity: no MSuspend signal reaches the module 0x1: Sensitivity to MCU MSuspend signals (DSP signal ignored) 0x2: Sensitivity to DSP MSuspend signal (MCU signal ignored) 0x3: Sensitivity to the logical ORed MCU and DSP MSuspend signals 0x4: Sensitivity to the logical ANDed MCU and DSP MSuspend signals 0x5: No sensitivity: no MSuspend signal reaches the module 0x6: No sensitivity: no MSuspend signal reaches the module 0x7: No sensitivity: no MSuspend signal reaches the module	R/W	0x0
8:6	GPTM10MSCTRL	Control General Purpose Timer 10 sensitivity to MCU and/or DSP MSuspend signals 0x0: No sensitivity: no MSuspend signal reaches the module 0x1: Sensitivity to MCU MSuspend signals (DSP signal ignored) 0x2: Sensitivity to DSP MSuspend signal (MCU signal ignored) 0x3: Sensitivity to the logical ORed MCU and DSP MSuspend signals 0x4: Sensitivity to the logical ANDed MCU and DSP MSuspend signals 0x5: No sensitivity: no MSuspend signal reaches the module 0x6: No sensitivity: no MSuspend signal reaches the module 0x7: No sensitivity: no MSuspend signal reaches the module	R/W	0x0
5:3	GPTM9MSCTRL	Control General Purpose Timer 9 sensitivity to MCU and/or DSP MSuspend signals 0x0: No sensitivity: no MSuspend signal reaches the module 0x1: Sensitivity to MCU MSuspend signals (DSP signal ignored) 0x2: Sensitivity to DSP MSuspend signal (MCU signal ignored) 0x3: Sensitivity to the logical ORed MCU and DSP MSuspend signals 0x4: Sensitivity to the logical ANDed MCU and DSP MSuspend signals 0x5: No sensitivity: no MSuspend signal reaches the module 0x6: No sensitivity: no MSuspend signal reaches the module 0x7: No sensitivity: no MSuspend signal reaches the module	R/W	0x0

Bits	Field Name	Description	Type	Reset
2:0	GPTM8MSCTRL	Control General Purpose Timer 8 sensitivity to MCU and/or DSP MSuspend signals 0x0: No sensitivity: no MSuspend signal reaches the module 0x1: Sensitivity to MCU MSuspend signals (DSP signal ignored) 0x2: Sensitivity to DSP MSuspend signal (MCU signal ignored) 0x3: Sensitivity to the logical ORed MCU and DSP MSuspend signals 0x4: Sensitivity to the logical ANDed MCU and DSP MSuspend signals 0x5: No sensitivity: no MSuspend signal reaches the module 0x6: No sensitivity: no MSuspend signal reaches the module 0x7: No sensitivity: no MSuspend signal reaches the module	R/W	0x0

Table 7-85. Register Call Summary for Register CONTROL_MSUSPENDMUX_2

SCM Functional Description
- MPU and/or DSP (IVA2.2) MSuspend Configuration Registers: [0]

SCM Register Manual
- SCM Register Summary: [1]

Table 7-86. CONTROL_MSUSPENDMUX_3

Address Offset	0x0000 002C		
Physical Address	0x4800 229C	**Instance**	GENERAL
Description	MSuspend Control register : control the use of MSuspend signals at module level		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																															

Bits	Field Name	Description	Type	Reset
31:0	RESERVED	Read returns reset value.	R	0x0

Table 7-87. Register Call Summary for Register CONTROL_MSUSPENDMUX_3

SCM Functional Description
- MPU and/or DSP (IVA2.2) MSuspend Configuration Registers: [0]

SCM Register Manual
- SCM Register Summary: [1]

Table 7-88. CONTROL_MSUSPENDMUX_4

Address Offset	0x0000 0030		
Physical Address	0x4800 22A0	**Instance**	GENERAL
Description	MSuspend Control register: control the use of MSuspend signals at module level		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED			DMAMSCTRL					RESERVED																							

Bits	Field Name	Description	Type	Reset
31:30	RESERVED	Read returns reset value.	R	0x0
29:27	DMAMSCTRL	Control DMA sensitivity to MCU and/or DSP MSuspend signals 0x0: No sensitivity: No MSuspend signal reaches the module 0x1: Sensitivity to MCU MSuspend signals (DSP signal ignored) 0x2: Sensitivity to DSP MSuspend signal (MCU signal ignored) 0x3: Sensitivity to the logical ORed MCU and DSP MSuspend signals 0x4: Sensitivity to the logical ANDed MCU and DSP MSuspend signals 0x5: No sensitivity: No MSuspend signal reaches the module 0x6: No sensitivity: No MSuspend signal reaches the module 0x7: No sensitivity: No MSuspend signal reaches the module	R/W	0x0
26:0	RESERVED	Read returns reset value.	R	0x0000000

Table 7-89. Register Call Summary for Register CONTROL_MSUSPENDMUX_4

SCM Functional Description
- MPU and/or DSP (IVA2.2) MSuspend Configuration Registers: [0]

SCM Register Manual
- SCM Register Summary: [1]

Table 7-90. CONTROL_MSUSPENDMUX_5

Address Offset	0x0000 0034		
Physical Address	0x4800 22A4	**Instance**	GENERAL
Description	MSuspend Control register: control the use of MSuspend signals at module level Not used		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								I2C3MSCTRL			RESERVED													MCBSP5MSCTRL			MCBSP4MSCTRL			MCBSP3MSCTRL		

Bits	Field Name	Description	Type	Reset
31:24	RESERVED	Read returns reset value.	R	0x0
23:21	I2C3MSCTRL	Control I2C-3 Sensitivity to MCU and/or DSP MSuspend Signals 0x0: No sensitivity: No MSuspend signal reaches the module 0x1: Sensitivity to MCU MSuspend signals (DSP signal ignored) 0x2: Sensitivity to DSP MSuspend signal (MCU signal ignored) 0x3: Sensitivity to the logical ORed MCU and DSP MSuspend signals 0x4: Sensitivity to the logical ANDed MCU and DSP MSuspend signals 0x5: No sensitivity: No MSuspend signal reaches the module 0x6: No sensitivity: No MSuspend signal reaches the module 0x7: No sensitivity: No MSuspend signal reaches the module	R/W	0x0
20:9	RESERVED	Read returns reset value.	R	0x0

Bits	Field Name	Description	Type	Reset
8:6	MCBSP5MSCTRL	Control McBSP-5 Sensitivity to MCU and/or DSP MSuspend Signals	R/W	0x0
		0x0: No sensitivity: No MSuspend signal reaches the module		
		0x1: Sensitivity to MCU MSuspend signals (DSP signal ignored)		
		0x2: Sensitivity to DSP MSuspend signal (MCU signal ignored)		
		0x3: Sensitivity to the logical ORed MCU and DSP MSuspend signals		
		0x4: Sensitivity to the logical ANDed MCU and DSP MSuspend signals		
		0x5: No sensitivity: No MSuspend signal reaches the module		
		0x6: No sensitivity: No MSuspend signal reaches the module		
		0x7: No sensitivity: No MSuspend signal reaches the module		
5:3	MCBSP4MSCTRL	Control McBSP-4 Sensitivity to MCU and/or DSP MSuspend Signals	R/W	0x0
		0x0: No sensitivity: No MSuspend signal reaches the module		
		0x1: Sensitivity to MCU MSuspend signals (DSP signal ignored)		
		0x2: Sensitivity to DSP MSuspend signal (MCU signal ignored)		
		0x3: Sensitivity to the logical ORed MCU and DSP MSuspend signals		
		0x4: Sensitivity to the logical ANDed MCU and DSP MSuspend signals		
		0x5: No sensitivity: No MSuspend signal reaches the module		
		0x6: No sensitivity: No MSuspend signal reaches the module		
		0x7: No sensitivity: No MSuspend signal reaches the module		
2:0	MCBSP3MSCTRL	Control McBSP-3 Sensitivity to MCU and/or DSP MSuspend Signals	R/W	0x0
		0x0: No sensitivity: no MSuspend signal reaches the module		
		0x1: Sensitivity to MCU MSuspend signals (DSP signal ignored)		
		0x2: Sensitivity to DSP MSuspend signal (MCU signal ignored)		
		0x3: Sensitivity to the logical ORed MCU and DSP MSuspend signals		
		0x4: Sensitivity to the logical ANDed MCU and DSP MSuspend signals		
		0x5: No sensitivity: No MSuspend signal reaches the module		
		0x6: No sensitivity: No MSuspend signal reaches the module		
		0x7: No sensitivity: No MSuspend signal reaches the module		

Table 7-91. Register Call Summary for Register CONTROL_MSUSPENDMUX_5

SCM Functional Description
- MPU and/or DSP (IVA2.2) MSuspend Configuration Registers: [0]

SCM Register Manual
- SCM Register Summary: [1]

Table 7-92. CONTROL_DEVCONF1

Address Offset	0x0000 0068		
Physical Address	0x4800 22D8	**Instance**	GENERAL
Description	Static device configuration register-1. Module dedicated functions		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								SENSDMAREQ6	SENSDMAREQ5	SENSDMAREQ4	CARKITHSUSB0DATA1AUTOEN	CARKITHSUSB0DATA0AUTOEN	TVOUTBYPASS	RESERVED		I2C3HSMASTER	I2C2HSMASTER	I2C1HSMASTER	TVACEN	RESERVED	MPUFORCEWRNP	SENSDMAREQ3	SENSDMAREQ2	MMCSDIO2ADPCLKISEL	RESERVED	MCBSP5_CLKS	RESERVED	MCBSP4_CLKS	RESERVED	MCBSP3_CLKS

Bits	Field Name	Description	Type	Reset
31:24	RESERVED	Read returns reset value	R	0x0
23	SENSDMAREQ6	Set sensitivity on SYS.nDMAREQ6 input pin 0x0: Level sensitivity 0x1: Edge sensitivity	R/W	0x0
22	SENSDMAREQ5	Set sensitivity on SYS.nDMAREQ5 input pin 0x0: Level sensitivity 0x1: Edge sensitivity	R/W	0x0
21	SENSDMAREQ4	Set sensitivity on SYS.nDMAREQ4 input pin 0x0: Level sensitivity 0x1: Edge sensitivity	R/W	0x0
20	CARKITHSUSB0 DATA1AUTOEN	Enable force from HSUB0 DATA1 pad configuration MuxMode when CARKITEN is generated: 0x0: HSUB0 DATA1 pad MuxMode is driven by HSUB0 DATA1 pad configuration register. 0x1: HSUB0 DATA1 pad MuxMode is forced to 0x2 when CARKITEN signal is active.	R/W	0x0
19	CARKITHSUSB0 DATA0AUTOEN	Enable force from HSUB0 DATA0 pad configuration MuxMode when CARKITEN is generated 0x0: HSUB0 DATA0 pad MuxMode is driven by HSUB0 DATA1 pad configuration register 0x1: HSUB0 DATA0 pad MuxMode is forced to 0x2 when CARKITEN signal is active.	R/W	0x0
18	TVOUTBYPASS	Active high enable Dual 10-bit video DAC TV out bypass	R/W	0x0
17:15	RESERVED	Read returns reset value.	R	0x0
14	I2C3HSMASTER	Active-high enable of I2C3 I/O internal pull-up (used for master component)	R/W	0x0
13	I2C2HSMASTER	Active-high enable of I2C2 I/O internal pull-up (used for master component)	R/W	0x0
12	I2C1HSMASTER	Active-high enable of I2C1 I/O internal pull-up (used for master component)	R/W	0x0
11	TVACEN	TV AC coupled load enable for TV output	R/W	0x0
10	RESERVED	Read returns reset value.	R	0x0
9	MPUFORCEWRNP	Force MPU writes to others to be non posted 0x0: Posted writes 0x1: Non posted writes	R/W	0x0
8	SENSDMAREQ3	Set sensitivity on SYS.nDMAREQ3 input pin 0x0: Level sensitivity 0x1: Edge sensitivity	R/W	0x0

Bits	Field Name	Description	Type	Reset
7	SENSDMAREQ2	Set sensitivity on SYS.nDMAREQ2 input pin 0x0: Level sensitivity 0x1: Edge sensitivity	R/W	0x0
6	MMCSDIO2ADPCLKISEL	MMC/SDIO2 Module Input Clock selection 0x0: Input clock is from the external pin 0x1: Internal loop-back; module input clock is copied from the module output clock	R/W	0x0
5	RESERVED	Read returns reset value.	R	0x0
4	MCBSP5_CLKS	Select the CLKS input for the module McBSP5 0x0: CLKS is from the PRCM functional clock 0x1: CLKS is from the external pin McBSP_CLKS	R/W	0x0
3	RESERVED	Read returns reset value.	R	0x0
2	MCBSP4_CLKS	Select the CLKS input for the module McBSP4 0x0: CLKS is from the PRCM functional clock 0x1: CLKS is from the external pin McBSP_CLKS	R/W	0x0
1	RESERVED	Read returns reset value.	R	0x0
0	MCBSP3_CLKS	Select the CLKS input for the module McBSP3 0x0: CLKS is from the PRCM functional clock 0x1: CLKS is from the external pin McBSP_CLKS	R/W	0x0

Table 7-93. Register Call Summary for Register CONTROL_DEVCONF1

SCM Environment
- SCM Environment: [0]

SCM Functional Description
- Static Device Configuration Registers: [1]

SCM Programming Model
- Force Pad Configuration MuxMode by High-Speed USB: [2] [3]
- Video Driver: [4] [5]
- McBSP3 Internal Clock: [6]
- McBSP4 Internal Clock: [7]
- McBSP5 Internal Clock: [8]
- MMC/SD/SDIO2 Module Input Clock Selection: [9]
- Setting Sensitivity on sys_ndmareq[6:0] Input Pins: [10] [11] [12] [13] [14]
- I2C I/O Internal Pull-up Enable: [15] [16] [17]
- Force MPU Writes to Be Nonposted: [18]

SCM Register Manual
- SCM Register Summary: [19]

Table 7-94. CONTROL_CSIRXFE

Address Offset	0x0000 0006C		
Physical Address	0x4800 22DC	**Instance**	GENERAL
Description	This register makes possible to control some settings of CSIRXFE cells used in the design		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																		CSIB_RESET	CSIB_PWRDNZ	RESERVED	CSIB_SELFORM	RESERVED	CSIB_RESENABLE	CSIB_INV	RESERVED							

Bits	Field Name	Description	Type	Reset
31:14	RESERVED	Read returns reset value.	R	0x0
13	CSIB_RESET	Active Low asynchronous reset signal for CSIb_interface	R/W	0x0
12	CSIB_PWRDNZ	Power Down control for CSIb_interface 0: CSIb differential transceiver is powered down 1: CSIb differential transceiver is active	R/W	0x0
11	RESERVED	Read returns reset value.	R	0x0
10	CSIB_SELFORM	CSI receiver transmission format selection for CSIb_interface 0: Data/clock transmission format. 1: Data/Strobe transmission format	R/W	0x0
9	RESERVED	Read returns reset value.	R	0x0
8	CSIB_RESENABLE	Enable resistor for CSIb_interface 0: CSIb internal resistor is disconnected (use external termination resistor) 1: CSIb internal resistor is enabled	R/W	0x0
7	CSIB_INV	Strobe/Clock inversion control for CSIb_interface 0: CSIb internal data latched on rising edge of clock 1: CSIb internal data latched on falling edge of clock	R/W	0x0
6:0	RESERVED	Read returns reset value.	R/W	0x0

Table 7-95. Register Call Summary for Register CONTROL_CSIRXFE

SCM Functional Description
- Control CSIRXFE Register: [0] [1]

SCM Register Manual
- SCM Register Summary: [2]

Table 7-96. CONTROL_PROT_ERR_STATUS

Address Offset	0x0000 0074		
Physical Address	0x4800 22E4	**Instance**	GENERAL
Description	Protection error status register		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED														L4EMUFWERROR	L4PERIPHFWERROR	D2DFWERROR	RESERVED	SMXAPERTFWERROR	RESERVED	DISPDMAACCERROR	CAMERADMAACCERROR	SYSDMAACCERROR	L4COREFWERROR	IVA2FWERROR	MAD2DFWERROR	SMSFWERROR	SMSFUNCFWERROR	GPMCFWERROR	OCMRAMFWERROR	OCMROMFWERROR	

Bits	Field Name	Description	Type	Reset
31:18	RESERVED	Read returns reset value.	R	0x0
17	L4EMUFWERROR	L4 Emulation Firewall Error 0x0: No L4 Emulation interconnect firewall error 0x1: L4 Emulation interconnect firewall error	R	0x0
16	L4PERIPHFW ERROR	L4 Peripheral Firewall Error 0x0: No L4 Peripheral interconnect firewall error 0x1: L4 Peripheral interconnect firewall error	R	0x0
15	D2DFWERROR	D2D Firewalll error 0x0: No D2D firewall error 0x1: D2D firewall error	R	0x0
14:13	RESERVED	Read returns reset value.	R	0x0
12	SMXAPERTFW ERROR	L3 Register target Firewall error 0x0: No L3 Register Target firewall error 0x1: L3 Register Target firewall error	R	0x0
11	RESERVED	Reserved for non GRP-devices	R	0x0
10	DISPDMAACC ERROR	Disp Dma Access Error 0x0: No access error to DISP DMA protection channels 0x1: Unauthorized access to a DISP DMA protection channel	R	0x0
9	CAMERADMAACC ERROR	Camera Dma Access Error 0x0: No access error to Camera DMA protection channels 0x1: Unauthorized access to a Camera DMA protection channel	R	0x0
8	SYSDMAACC ERROR	sDma Access Error 0x0: No access error to sDMA protection channels 0x1: Unauthorized access to a sDMA protection channel	R	0x0
7	L4COREFW ERROR	L4 Protection Firewall Error 0x0: No error from L4 Core protection firewall 0x1: Error from L4 Core protection firewall	R	0x0
6	IVA2FWERROR	IVA2 Protection Firewall Error 0x0: No error from IVA2 protection firewall 0x1: Error from IVA2 protection firewall	R	0x0
5	MAD2DFW ERROR	MAD2D Firewall Error 0x0: No MAD2D functional firewall error 0x0: No MAD2D functional firewall error	R	0x0
4	SMSFWERROR	SMS Firewall Error 0x0: No SMS firewall error 0x1: SMS firewall error	R	0x0
3	SMSFUNCFW ERROR	SMS Functional Firewall Error 0x0: No SMS functional firewall error 0x1: SMS functional firewall error	R	0x0
2	GPMCFWERROR	GPMC Firewall Error 0x0: No error from GPMC protection firewall 0x1: Error from GPMC protection firewall	R	0x0
1	OCMRAMFW ERROR	On Chip Ram Firewall Error	R	0x0

Bits	Field Name	Description	Type	Reset
		0x0: No error from On Chip RAM protection firewall 0x1: Error from On Chip RAM protectionfirewall		
0	OCMROMFW ERROR	On Chip Rom Firewall Error 0x0: No error from On Chip ROM protection firewall 0x1: Error from On Chip ROM protection firewall	R	0x0

Table 7-97. Register Call Summary for Register CONTROL_PROT_ERR_STATUS

SCM Functional Description

- Protection Status Registers: [0] [1] [2] [3] [4] [5] [6] [7] [8] [9] [10] [11]

SCM Register Manual

- SCM Register Summary: [12]

Table 7-98. CONTROL_PROT_ERR_STATUS_DEBUG

Address Offset	0x0000 0078		
Physical Address	0x4800 22E8	**Instance**	GENERAL
Description	Protection Error Status Debug Register		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED														L4EMUDBGFWERROR	L4PERIPHERALDBGFWERROR	RESERVED			SMXAPERTDBGFWERROR	RESERVED				L4COREDBGFWERROR	IVA2DBGFWERROR	MAD2D2DBGFWERROR	RESERVED	SMSDBGFWERROR	GPMCDBGFWERROR	OCMRAMDBGFWERROR	OCMROMDBGFWERROR

Bits	Field Name	Description	Type	Reset
31:18	RESERVED	Read returns reset value	R	0x0
17	L4EMUDBGFW ERROR	L4 Emulation Debug Firewall Error 0x0: No firewall error in L4 Emulation Debug 0x1: Firewall error in L4 Emulation Debug	R	0x0
16	L4PERIPHERALD BGFWERROR	L4 Peripheral Debug Firewall Error 0x0: No firewall error in L4 Peripheral Debug 0x1: Firewall error in L4 Peripheral Debug	R	0x0
15:13	RESERVED	Read returns reset value	R	0x0
12	SMXAPERT DBGFWERROR	L3 Register target Debug Firewall error 0x0: No firewall error in L3 Register Target Debug 0x1: Firewall error in L3 Register Target Debug	R	0x0
11:8	RESERVED	Read returns reset value.	R	0x0
7	L4COREDBGFW ERROR	L4 Core Debug Firewall Error 0x0: No firewall error in L4 Core Debug 0x1: Firewall error in L4 Core Debug	R	0x0
6	IVA2DBGFW ERROR	IVA2 Debug Firewall Error 0x0: No error from IVA2 debug protection firewall 0x1: Error from IVA2 debug protection firewall	R	0x0

Bits	Field Name	Description	Type	Reset
5	MAD2D2DBGFW ERROR	MAD2D Debug Firewall Error 0x0: No Firewall debug error in MAD2D 0x0: No Firewall debug error in MAD2D	R	0x0
4	RESERVED	Read returns reset value.	R	0x0
3	SMSDBGFW ERROR	SMS Debug Firewall Error 0x0: No Firewall debug error in SMS 0x1: Firewall debug error in SMS	R	0x0
2	GPMCDBGFW ERROR	GPMC Debug Firewall Error 0x0: No error from GPMC debug protection firewall 0x1: Error from GPMC debug protection firewall	R	0x0
1	OCMRAMDBGFW ERROR	On Chip Ram Debug Firewall Error 0x0: No error from On Chip RAM debug protection firewall 0x1: Error from On Chip RAM debug protection firewall	R	0x0
0	OCMROMDBGFW ERROR	On Chip Rom Debug Firewall Error 0x0: No error from On Chip ROM debug protection firewall 0x1: Error from On Chip ROM debug protection firewall	R	0x0

Table 7-99. Register Call Summary for Register CONTROL_PROT_ERR_STATUS_DEBUG

SCM Functional Description
- Protection Status Registers: [0] [1] [2] [3] [4] [5] [6] [7]

SCM Register Manual
- SCM Register Summary: [8]

Table 7-100. CONTROL_STATUS

Address Offset	0x0000 0080		
Physical Address	0x4800 22F0	**Instance**	GENERAL
Description	Control Module Status register: latches system information at reset time		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																					DEVICETYPE			RESERVED						SYS_BOOT							

Bits	Field Name	Description	Type	Reset
31:11	RESERVED	Read returns reset value.	R	0x-
10:8	DEVICETYPE	Device type captured at reset time 0x3: GP device¿ Other values: Reserved	R	0x-
7:6	RESERVED	Read returns reset value.	R	0x-
5:0	SYS_BOOT	sys_boot[5:0] pin values sampled at power-on reset	R	0x-

Table 7-101. Register Call Summary for Register CONTROL_STATUS

SCM Environment
- SCM Environment: [0]

SCM Register Manual
- SCM Register Summary: [1]

Table 7-102. CONTROL_GENERAL_PURPOSE_STATUS

Address Offset	0x0000 0084		
Physical Address	0x4800 22F4	**Instance**	GENERAL
Description	Status bits reflecting chip internal states		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								RESERVED																							SAVEDONE

Bits	Field Name	Description	Type	Reset
31	RESERVED	Reserved. Read returns reset value.	R	0x0
30:1	RESERVED	Reserved	R	0x0
0	SAVEDONE	Pad configuration save status	R	0x0

Table 7-103. Register Call Summary for Register CONTROL_GENERAL_PURPOSE_STATUS

SCM Functional Description
- System Off Mode: [0]

SCM Register Manual
- SCM Register Summary: [1]

Table 7-104. CONTROL_RPUB_KEY_H_0

Address Offset	0x0000 0090		
Physical Address	0x4800 2300	**Instance**	SYSC_GENERAL1
Description	Root_public_key_hash; B-field		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RPKH_31_0																															

Bits	Field Name	Description	Type	Reset
31:0	RPKH_31_0	Root Public Key Hash bits [31:0] Fuse Keys [31:0]	R	0x00000000

Table 7-105. Register Call Summary for Register CONTROL_RPUB_KEY_H_0

SCM Register Manual
- SCM Register Summary: [0]

Table 7-106. CONTROL_RPUB_KEY_H_1

Address Offset	0x0000 0094		
Physical Address	0x4800 2304	**Instance**	SYSC_GENERAL1
Description	Root_public_key_hash; B-field		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RPKH_63_32																															

Bits	Field Name	Description	Type	Reset
31:0	RPKH_63_32	Root Public Key Hash bits [63:32] Fuse Keys [63:32]	R	0x00000000

Table 7-107. Register Call Summary for Register CONTROL_RPUB_KEY_H_1

SCM Register Manual

- SCM Register Summary: [0]

Table 7-108. CONTROL_RPUB_KEY_H_2

Address Offset	0x0000 0098		
Physical Address	0x4800 2308	**Instance**	SYSC_GENERAL1
Description	Root_public_key_hash; B-field		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RPKH_95_64																															

Bits	Field Name	Description	Type	Reset
31:0	RPKH_95_64	Root Public Key Hash bits [95:64] Fuse Keys [95:64]	R	0x00000000

Table 7-109. Register Call Summary for Register CONTROL_RPUB_KEY_H_2

SCM Register Manual

- SCM Register Summary: [0]

Table 7-110. CONTROL_RPUB_KEY_H_3

Address Offset	0x0000 009C		
Physical Address	0x4800 230C	**Instance**	SYSC_GENERAL1
Description	Root_public_key_hash; B-field		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RPKH_127_96																															

Bits	Field Name	Description	Type	Reset
31:0	RPKH_127_96	Root Public Key Hash bits [127:96] Fuse Keys [127:96]	R	0x00000000

Table 7-111. Register Call Summary for Register CONTROL_RPUB_KEY_H_3

SCM Register Manual

- SCM Register Summary: [0]

Table 7-112. CONTROL_RPUB_KEY_H_4

Address Offset	0x0000 00A0		
Physical Address	0x4800 2310	**Instance**	SYSC_GENERAL1
Description	Root_public_key_hash; B-field		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RPKH_159_128																															

Bits	Field Name	Description	Type	Reset
31:0	RPKH_159_128	Root Public Key Hash bits [159:128] Fuse Keys [159:128]	R	0x00000000

Table 7-113. Register Call Summary for Register CONTROL_RPUB_KEY_H_4

SCM Register Manual

- SCM Register Summary: [0]

Table 7-114. CONTROL_USB_CONF_0

Address Offset	0x0000 0100		
Physical Address	0x4800 2370	**Instance**	SYSC_GENERAL1
Description	USB Fuse conf [31:0],USB Product ID [31:16] Vendor ID [15:0]		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
USB_CONF_31_0																															

Bits	Field Name	Description	Type	Reset
31:0	USB_CONF_31_0	USB Fuse conf [31:0],USB Product ID [31:16] Vendor ID [15:0]	R	0x00000000

Table 7-115. Register Call Summary for Register CONTROL_USB_CONF_0

SCM Register Manual

- SCM Register Summary: [0]

Table 7-116. CONTROL_USB_CONF_1

Address Offset	0x0000 0104		
Physical Address	0x4800 2374	**Instance**	SYSC_GENERAL1
Description	USB Fuse conf [63:32], SEQ_DISADAPTCLK[1], USB PHY detection mode [0]		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
USBCONF_63_32																															

Bits	Field Name	Description	Type	Reset
31:0	USBCONF_63_32	USB Fuse conf [63:32], SEQ_DISADAPTCLK[1], USB PHY detection mode [0]	R	0x00000000

Table 7-117. Register Call Summary for Register CONTROL_USB_CONF_1

SCM Register Manual

- SCM Register Summary: [0]

Table 7-118. CONTROL_FUSE_OPP1_VDD1

Address Offset	0x0000 0110		
Physical Address	0x4800 2380	**Instance**	SYSC_GENERAL1
Description	Fuse OPP[95:72]		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								FUSE_OPP_95_72																							

Bits	Field Name	Description	Type	Reset
31:24	RESERVED	Read returns reset value.	R	0x00000000
23:0	FUSE_OPP_95_72	Fuse OPP [95:72]	R	0x00000000

Table 7-119. Register Call Summary for Register CONTROL_FUSE_OPP1_VDD1

SCM Register Manual

- SCM Register Summary: [0]

Table 7-120. CONTROL_FUSE_OPP2_VDD1

Address Offset	0x0000 0114		
Physical Address	0x4800 2384	**Instance**	SYSC_GENERAL1
Description	Fuse OPP [119:96]		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								FUSE_OPP_119_96																							

Bits	Field Name	Description	Type	Reset
31:24	RESERVED	Read returns reset value.	R	0x00000000
23:0	FUSE_OPP_119_96	Fuse OPP [119:96]	R	0x00000000

Table 7-121. Register Call Summary for Register CONTROL_FUSE_OPP2_VDD1

SCM Register Manual

- SCM Register Summary: [0]

Table 7-122. CONTROL_FUSE_OPP3_VDD1

Address Offset	0x0000 0118		
Physical Address	0x4800 2388	**Instance**	SYSC_GENERAL1
Description	Fuse OPP [143:120]		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								FUSE_OPP_143_120																							

Bits	Field Name	Description	Type	Reset
31:24	RESERVED	Read returns reset value.	R	0x00000000
23:0	FUSE_OPP_143_120	Fuse OPP [143:120]	R	0x00000000

Table 7-123. Register Call Summary for Register CONTROL_FUSE_OPP3_VDD1

SCM Register Manual

- SCM Register Summary: [0]

Table 7-124. CONTROL_FUSE_OPP4_VDD1

Address Offset	0x0000 011C		
Physical Address	0x4800 238C	**Instance**	SYSC_GENERAL1
Description	Fuse OPP [167:144]		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								FUSE_OPP_167_144																							

Bits	Field Name	Description	Type	Reset
31:24	RESERVED	Read returns reset value.	R	0x00000000
23:0	FUSE_OPP_167_144	Fuse OPP [167:144]	R	0x00000000

Table 7-125. Register Call Summary for Register CONTROL_FUSE_OPP4_VDD1

SCM Register Manual

- SCM Register Summary: [0]

Table 7-126. CONTROL_FUSE_OPP5_VDD1

Address Offset	0x0000 0120		
Physical Address	0x4800 2390	**Instance**	SYSC_GENERAL1
Description	Fuse OPP [191:168]		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								FUSE_OPP_191_168																							

Bits	Field Name	Description	Type	Reset
31:24	RESERVED	Read returns reset value.	R	0x00000000
23:0	FUSE_OPP_191_168	Fuse OPP [191:168]	R	0x00000000

Table 7-127. Register Call Summary for Register CONTROL_FUSE_OPP5_VDD1

SCM Register Manual

- SCM Register Summary: [0]

Table 7-128. CONTROL_FUSE_OPP1_VDD2

Address Offset	0x0000 0124		
Physical Address	0x4800 2394	**Instance**	SYSC_GENERAL1
Description	Fuse OPP [23:0]		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								FUSE_OPP_23_0																							

Bits	Field Name	Description	Type	Reset
31:24	RESERVED	Read returns reset value.	R	0x00000000
23:0	FUSE_OPP_23_0	Fuse OPP [23:0]	R	0x00000000

Table 7-129. Register Call Summary for Register CONTROL_FUSE_OPP1_VDD2

SCM Register Manual

- SCM Register Summary: [0]

Table 7-130. CONTROL_FUSE_OPP2_VDD2

Address Offset	0x0000 0128		
Physical Address	0x4800 2398	**Instance**	SYSC_GENERAL1
Description	Fuse OPP [47:24]		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								FUSE_OPP_47_24																							

Bits	Field Name	Description	Type	Reset
31:24	RESERVED	Read returns reset value.	R	0x00000000
23:0	FUSE_OPP_47_24	Fuse OPP [47:24]	R	0x00000000

Table 7-131. Register Call Summary for Register CONTROL_FUSE_OPP2_VDD2

SCM Register Manual

- SCM Register Summary: [0]

Table 7-132. CONTROL_FUSE_OPP3_VDD2

Address Offset	0x0000 012C		
Physical Address	0x4800 239C	**Instance**	SYSC_GENERAL1
Description	Fuse OPP [63:48]		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								FUSE_OPP_63_48																							

Bits	Field Name	Description	Type	Reset
31:24	RESERVED	Read returns reset value.	R	0x00000000
23:0	FUSE_OPP_63_48	Fuse OPP [63:48]	R	0x00000000

Table 7-133. Register Call Summary for Register CONTROL_FUSE_OPP3_VDD2

SCM Register Manual

- SCM Register Summary: [0]

Table 7-134. CONTROL_FUSE_SR

Address Offset	0x0000 0130		
Physical Address	0x4800 23A0	**Instance**	SYSC_GENERAL1
Description	Fuse SR1 and SR2		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																FUSE_SR2								FUSE_SR1							

Bits	Field Name	Description	Type	Reset
31:16	RESERVED	Read returns reset value.	R	0x00000000
15:8	FUSE_SR2	Fuse SR 2	R	0x00000000
7:0	FUSE_SR1	Fuse SR 1	R	0x00000000

Table 7-135. Register Call Summary for Register CONTROL_FUSE_SR

SCM Register Manual

- SCM Register Summary: [0]

Table 7-136. CONTROL_IVA2_BOOTADDR

Address Offset	0x0000 0190		
Physical Address	0x4800 2400	**Instance**	GENERAL
Description	This register defines the physical address of the IVA2 boot loader		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BOOTLOADADDR																						RESERVED									

Bits	Field Name	Description	Type	Reset
31:10	BOOTLOADADDR	Physical Address of the IVA2 boot loader. This is an index to a 4kByte page	R/W	0x0
9:0	RESERVED	Reserved. Write 0's for compatibility.	R	0x0

Table 7-137. Register Call Summary for Register CONTROL_IVA2_BOOTADDR

SCM Functional Description

- IVA2.2 Boot Registers: [0] [1] [2]

SCM Register Manual

- SCM Register Summary: [3]

Table 7-138. CONTROL_IVA2_BOOTMOD

Address Offset	0x0000 0194		
Physical Address	0x4800 2404	**Instance**	GENERAL
Description	This register defines the IVA2 bootmode		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																												BOOTMODE			

Bits	Field Name	Description	Type	Reset
31:4	RESERVED	Read returns reset value.	R	0x0
3:0	BOOTMODE	Boot mode of the IVA2	R/W	0x0

Table 7-139. Register Call Summary for Register CONTROL_IVA2_BOOTMOD

SCM Functional Description

- IVA2.2 Boot Registers: [0] [1]

SCM Register Manual

- SCM Register Summary: [2]

Table 7-140. CONTROL_DEBOBS_0

Address Offset	0x0000 01B0		
Physical Address	0x4800 2420	**Instance**	GENERAL
Description	Select the set of signals to be observed for hw_dbg0, hw_dbg1		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								OBSMUX0								RESERVED								OBSMUX1							

Bits	Field Name	Description	Type	Reset
31:23	RESERVED	Read returns reset value.	R	0x00
22:16	OBSMUX0	Select the set of signals to be exported for WKUPOBSMUX0	RW	0x000
15:7	RESERVED	Read returns reset value.	R	0x00
6:0	OBSMUX1	Select the set of signals to be exported for WKUPOBSMUX1	RW	0x000

Table 7-141. Register Call Summary for Register CONTROL_DEBOBS_0

SCM Functional Description
- Description: [0]
- Observability Tables: [1] [2]

SCM Register Manual
- SCM Register Summary: [3]

Table 7-142. CONTROL_DEBOBS_1

Address Offset	0x0000 01B4		
Physical Address	0x4800 2424	**Instance**	GENERAL
Description	Select the set of signals to be observed for hw_dbg2, hw_dbg3		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								OBSMUX2								RESERVED								OBSMUX3							

Bits	Field Name	Description	Type	Reset
31:23	RESERVED	Read returns reset value.	R	0x00
22:16	OBSMUX2	Select the set of signals to be exported for WKUPOBSMUX2	RW	0x000
15:7	RESERVED	Read returns reset value.	R	0x00
6:0	OBSMUX3	Select the set of signals to be exported for WKUPOBSMUX3	RW	0x000

Table 7-143. Register Call Summary for Register CONTROL_DEBOBS_1

SCM Functional Description
- Description: [0]
- Observability Tables: [1] [2]

SCM Register Manual
- SCM Register Summary: [3]

Table 7-144. CONTROL_DEBOBS_2

Address Offset	0x0000 01B8		
Physical Address	0x4800 2428	**Instance**	GENERAL
Description	Select the set of signals to be observed for hw_dbg4, hw_dbg5		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								OBSMUX4								RESERVED								OBSMUX5							

Bits	Field Name	Description	Type	Reset
31:23	RESERVED	Read returns reset value.	R	0x00
22:16	OBSMUX4	Select the set of signals to be exported for WKUPOBSMUX4	RW	0x000
15:7	RESERVED	Read returns reset value.	R	0x00
6:0	OBSMUX5	Select the set of signals to be exported for WKUPOBSMUX5	RW	0x000

Table 7-145. Register Call Summary for Register CONTROL_DEBOBS_2

SCM Functional Description
- Description: [0]
- Observability Tables: [1] [2]

SCM Register Manual
- SCM Register Summary: [3]

Table 7-146. CONTROL_DEBOBS_3

Address Offset	0x0000 01BC		
Physical Address	0x4800 242C	**Instance**	GENERAL
Description	Select the set of signals to be observed for hw_dbg6, hw_dbg7		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								OBSMUX6								RESERVED								OBSMUX7							

Bits	Field Name	Description	Type	Reset
31:23	RESERVED	Read returns reset value.	R	0x00
22:16	OBSMUX6	Select the set of signals to be exported for WKUPOBSMUX6	RW	0x000
15:7	RESERVED	Read returns reset value.	R	0x00
6:0	OBSMUX7	Select the set of signals to be exported for WKUPOBSMUX7	RW	0x000

Table 7-147. Register Call Summary for Register CONTROL_DEBOBS_3

SCM Functional Description
- Description: [0]
- Observability Tables: [1] [2]

SCM Register Manual
- SCM Register Summary: [3]

Table 7-148. CONTROL_DEBOBS_4

Address Offset	0x0000 01C0		
Physical Address	0x4800 2430	**Instance**	GENERAL
Description	Select the set of signals to be observed for hw_dbg8, hw_dbg9		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								OBSMUX8								RESERVED								OBSMUX9							

Bits	Field Name	Description	Type	Reset
31:23	RESERVED	Read returns reset value.	R	0x00
22:16	OBSMUX8	Select the set of signals to be exported for WKUPOBSMUX8	RW	0x000
15:7	RESERVED	Read returns reset value.	R	0x00
6:0	OBSMUX9	Select the set of signals to be exported for WKUPOBSMUX9	RW	0x000

Table 7-149. Register Call Summary for Register CONTROL_DEBOBS_4

SCM Functional Description
- Description: [0]
- Observability Tables: [1] [2]

SCM Register Manual
- SCM Register Summary: [3]

Table 7-150. CONTROL_DEBOBS_5

Address Offset	0x0000 01C4		
Physical Address	0x4800 2434	**Instance**	GENERAL
Description	Select the set of signals to be observed for hw_dbg10, hw_dbg11		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								OBSMUX10								RESERVED								OBSMUX11							

Bits	Field Name	Description	Type	Reset
31:23	RESERVED	Read returns reset value.	R	0x00
22:16	OBSMUX10	Select the set of signals to be exported for WKUPOBSMUX10	RW	0x000
15:7	RESERVED	Read returns reset value.	R	0x00
6:0	OBSMUX11	Select the set of signals to be exported for WKUPOBSMUX11	RW	0x000

Table 7-151. Register Call Summary for Register CONTROL_DEBOBS_5

SCM Functional Description
- Description: [0]
- Observability Tables: [1] [2]

SCM Register Manual
- SCM Register Summary: [3]

Table 7-152. CONTROL_DEBOBS_6

Address Offset	0x0000 01C8		
Physical Address	0x4800 2438	**Instance**	GENERAL
Description	Select the set of signals to be observed for hw_dbg12, hw_dbg13		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								OBSMUX12								RESERVED								OBSMUX13							

Bits	Field Name	Description	Type	Reset
31:23	RESERVED	Read returns reset value.	R	0x00
22:16	OBSMUX12	Select the set of signals to be exported for WKUPOBSMUX12	RW	0x000
15:7	RESERVED	Read returns reset value.	R	0x00
6:0	OBSMUX13	Select the set of signals to be exported for WKUPOBSMUX13	RW	0x000

Table 7-153. Register Call Summary for Register CONTROL_DEBOBS_6

SCM Functional Description
- Description: [0]
- Observability Tables: [1] [2]

SCM Register Manual
- SCM Register Summary: [3]

Table 7-154. CONTROL_DEBOBS_7

Address Offset	0x0000 01CC		
Physical Address	0x4800 243C	**Instance**	GENERAL
Description	Select the set of signals to be observed for hw_dbg14, hw_dbg15		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								OBSMUX14								RESERVED								OBSMUX15							

Bits	Field Name	Description	Type	Reset
31:23	RESERVED	Read returns reset value.	R	0x00
22:16	OBSMUX14	Select the set of signals to be exported for WKUPOBSMUX14	RW	0x000
15:7	RESERVED	Read returns reset value.	R	0x00
6:0	OBSMUX15	Select the set of signals to be exported for WKUPOBSMUX15	RW	0x000

Table 7-155. Register Call Summary for Register CONTROL_DEBOBS_7

SCM Functional Description
- Description: [0]
- Observability Tables: [1] [2]

SCM Register Manual
- SCM Register Summary: [3]

Table 7-156. CONTROL_DEBOBS_8

Address Offset	0x0000 01D0		
Physical Address	0x4800 2440	**Instance**	GENERAL
Description	Select the set of signals to be observed for hw_dbg16, hw_dbg17		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								OBSMUX16								RESERVED								OBSMUX17							

Bits	Field Name	Description	Type	Reset
31:23	RESERVED	Read returns reset value.	R	0x00
22:16	OBSMUX16	Select the set of signals to be exported for WKUPOBSMUX16	RW	0x000
15:7	RESERVED	Read returns reset value.	R	0x00
6:0	OBSMUX17	Select the set of signals to be exported for WKUPOBSMUX17	RW	0x000

Table 7-157. Register Call Summary for Register CONTROL_DEBOBS_8

SCM Functional Description

- Description: [0]
- Observability Tables: [1] [2]

SCM Register Manual

- SCM Register Summary: [3]

Table 7-158. CONTROL_PROG_IO0

Address Offset	0x0000 01D4		
Physical Address	0x4800 2444	**Instance**	GENERAL
Description	Configure drive strength of I/O cells		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SDRC_LOWDATA	SDRC_HIGHDATA	SDRC_ADDRCTR	SDRC_NCS0	SDRC_NCS1	GPMC_A1	GPMC_A2	GPMC_A3	GPMC_A4	GPMC_A5	GPMC_A6	GPMC_A7	GPMC_A8	GPMC_A9	GPMC_A10	GPMC_MIN_CFG	GPMC_D8_D15	GPMC_NCS0	GPMC_NCS1	GPMC_NCS2	GPMC_NCS3	GPMC_NCS4	GPMC_NCS5	GPMC_NCS6	GPMC_NCS7	GPMC_CLK	GPMC_NBE0_CLE	GPMC_NBE1	GPMC_NWP	GPMC_WAIT1	GPMC_WAIT2	GPMC_WAIT3

Bits	Field Name	Description	Type	Reset
31	SDRC_LOWDATA	Drive strength for output load: 0x0: Drive strength set to low 2-6 pF 0x1: Drive strength set to high 6-12 pF	R/W	0x0
30	SDRC_HIGHDATA	Drive strength for output load: 0x0: Drive strength set to low 2-6 pF 0x1: Drive strength set to high 6-12 pF	R/W	0x0
29	SDRC_ADDRCTR	Drive strength for output load: 0x0: Drive strength set to low 2-6 pF 0x1: Drive strength set to high 6-12 pF	R/W	0x0
28	SDRC_NCS0	Drive strength for output load: 0x0: Drive strength set to low 2-6 pF 0x1: Drive strength set to high 6-12 pF	R/W	0x0
27	SDRC_NCS1	Drive strength for output load: 0x0: Drive strength set to low 2-6 pF 0x1: Drive strength set to high 6-12 pF	R/W	0x0
26	GPMC_A1	Drive strength for output load: 0x0: Drive strength set to low 2-6 pF 0x1: Drive strength set to high 6-12 pF	R/W	0x0

Bits	Field Name	Description	Type	Reset
25	GPMC_A2	Drive strength for output load: 0x0: Drive strength set to low 2-6 pF 0x1: Drive strength set to high 6-12 pF	R/W	0x0
24	GPMC_A3	Drive strength for output load: 0x0: Drive strength set to low 2-6 pF 0x1: Drive strength set to high 6-12 pF	R/W	0x0
23	GPMC_A4	Drive strength for output load: 0x0: Drive strength set to low 2-6 pF 0x1: Drive strength set to high 6-12 pF	R/W	0x0
22	GPMC_A5	Drive strength for output load: 0x0: Drive strength set to low 2-6 pF 0x1: Drive strength set to high 6-12 pF	R/W	0x0
21	GPMC_A6	Drive strength for output load: 0x0: Drive strength set to low 2-6 pF 0x1: Drive strength set to high 6-12 pF	R/W	0x0
20	GPMC_A7	Drive strength for output load: 0x0: Drive strength set to low 2-6 pF 0x1: Drive strength set to high 6-12 pF	R/W	0x0
19	GPMC_A8	Drive strength for output load: 0x0: Drive strength set to low 2-6 pF 0x1: Drive strength set to high 6-12 pF	R/W	0x0
18	GPMC_A9	Drive strength for output load: 0x0: Drive strength set to low 2-6 pF 0x1: Drive strength set to high 6-12 pF	R/W	0x0
17	GPMC_A10	Drive strength for output load: 0x0: Drive strength set to low 2-6 pF 0x1: Drive strength set to high 6-12 pF	R/W	0x0
16	GPMC_MIN_CFG	Drive strength for output load: 0x0: Drive strength set to low 2-6 pF 0x1: Drive strength set to high 6-12 pF	R/W	0x0
15	GPMC_D8_D15	Drive strength for output load: 0x0: Drive strength set to low 2-6 pF 0x1: Drive strength set to high 6-12 pF	R/W	0x0
14	GPMC_NCS0	Drive strength for output load: 0x0: Drive strength set to low 2-6 pF 0x1: Drive strength set to high 6-12 pF	R/W	0x1
13	GPMC_NCS1	Drive strength for output load: 0x0: Drive strength set to low 2-6 pF 0x1: Drive strength set to high 6-12 pF	R/W	0x1
12	GPMC_NCS2	Drive strength for output load: 0x0: Drive strength set to low 2-6 pF 0x1: Drive strength set to high 6-12 pF	R/W	0x1
11	GPMC_NCS3	Drive strength for output load: 0x0: Drive strength set to low 2-6 pF 0x1: Drive strength set to high 6-12 pF	R/W	0x1
10	GPMC_NCS4	Drive strength for output load: 0x0: Drive strength set to low 2-6 pF 0x1: Drive strength set to high 6-12 pF	R/W	0x1
9	GPMC_NCS5	Drive strength for output load: 0x0: Drive strength set to low 2-6 pF 0x1: Drive strength set to high 6-12 pF	R/W	0x1
8	GPMC_NCS6	Drive strength for output load: 0x0: Drive strength set to low 2-6 pF 0x1: Drive strength set to high 6-12 pF	R/W	0x1
7	GPMC_NCS7	Drive strength for output load: 0x0: Drive strength set to low 2-6 pF 0x1: Drive strength set to high 6-12 pF	R/W	0x1
6	GPMC_CLK	Drive strength for output load: 0x0: Drive strength set to low 2-6 pF 0x1: Drive strength set to high 6-12 pF	R/W	0x1

Bits	Field Name	Description	Type	Reset
5	GPMC_NBE0_CLE	Drive strength for output load: 0x0: Drive strength set to low 2-6 pF 0x1: Drive strength set to high 6-12 pF	R/W	0x0
4	GPMC_NBE1	Drive strength for output load: 0x0: Drive strength set to low 2-6 pF 0x1: Drive strength set to high 6-12 pF	R/W	0x0
3	GPMC_NWP	Drive strength for output load: 0x0: Drive strength set to low 2-6 pF 0x1: Drive strength set to high 6-12 pF	R/W	0x0
2	GPMC_WAIT1	Drive strength for output load: 0x0: Drive strength set to low 2-6 pF 0x1: Drive strength set to high 6-12 pF	R/W	0x0
1	GPMC_WAIT2	Drive strength for output load: 0x0: Drive strength set to low 2-6 pF 0x1: Drive strength set to high 6-12 pF	R/W	0x0
0	GPMC_WAIT3	Drive strength for output load: 0x0: Drive strength set to low 2-6 pF 0x1: Drive strength set to high 6-12 pF	R/W	0x0

Table 7-159. Register Call Summary for Register CONTROL_PROG_IO0

SCM Programming Model

- SDRC I/O Drive Strength Selection: [0] [1] [2] [3] [4]
- GPMC I/O Drive Strength Selection: [5] [6] [7] [8] [9] [10] [11] [12] [13] [14] [15] [16] [17] [18] [19] [20] [21] [22] [23] [24] [25] [26] [27] [28] [29] [30] [31]

SCM Register Manual

- SCM Register Summary: [32]

Table 7-160. CONTROL_PROG_IO1

Address Offset	0x0000 01D8		
Physical Address	0x4800 2448	**Instance**	GENERAL
Description	Configure drive strength of I/O cells		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																								MCBSP2_MIN_CTL		MCSPI1_MIN_CTRL		MCSP1_CS1		MCSP1_CS2	

Bits	Field Name	Description	Type	Reset
31:8	RESERVED	Read returns reset value	R	0x0
7:6	MCBSP2_MIN_CTL	Drive strength for impedance and current output: 0x0: Drive strength set to 20 Ω, 8 mA 0x2: Drive strength set to 25 Ω, 6 mA 0x1: Drive strength set to 40 Ω, 4 mA 0x3: Drive strength set to 65 Ω, 2 mA	R/W	0x2
5:4	MCSPI1_MIN_CTRL	Drive strength for impedance and current output: 0x0: Drive strength set to 20 Ω, 8 mA 0x2: Drive strength set to 25 Ω, 6 mA 0x1: Drive strength set to 40 Ω, 4 mA 0x3: Drive strength set to 65 Ω, 2 mA	R/W	0x2

Bits	Field Name	Description	Type	Reset
3:2	MCSP1_CS1	Drive strength for impedance and current output: 0x0: Drive strength set to 20 Ω, 8 mA 0x2: Drive strength set to 25 Ω, 6 mA 0x1: Drive strength set to 40 Ω, 4 mA 0x3: Drive strength set to 65 Ω, 2 mA	R/W	0x2
1:0	MCSP1_CS2	Drive strength for impedance and current output: 0x0: Drive strength set to 20 Ω, 8 mA 0x2: Drive strength set to 25 Ω, 6 mA 0x1: Drive strength set to 40 Ω, 4 mA 0x3: Drive strength set to 65 Ω, 2 mA	R/W	0x2

Table 7-161. Register Call Summary for Register CONTROL_PROG_IO1

SCM Programming Model
- MCBSP2 I/O Drive Strength Selection: [0]
- MCSPI1 I/O Drive Strength Selection: [1] [2] [3]

SCM Register Manual
- SCM Register Summary: [4]

Table 7-162. CONTROL_DSS_DPLL_SPREADING

Address Offset	0x0000 01E0		
Physical Address	0x4800 2450	**Instance**	GENERAL
Description	This register controls the EMI Reduction feature for Display_SS/DSI DPLL		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																								DSS_SPREADING_ENABLE_STATUS	RESERVED		DSS_SPREADING_ENABLE	DSS_SPREADING_AMPLITUDE		DSS_SPREADING_RATE	

Bits	Field Name	Description	Type	Reset
31:8	RESERVED	Reads return zero	R	0x000000
7	DSS_SPREADING_ENABLE_ STATUS	Indicates the status of the SSC feature	R	-
6:5	RESERVED	Reads return zero	R	0x0
4	DSS_SPREADING_ENABLE	Enables/disables EMI Reduction feature (Spreading): 0: Modulation stops at the end of the current modulation cycle. 1: Modulation cycle is started.	RW	0x0

Bits	Field Name	Description	Type	Reset
3:2	DSS_SPREADING_ AMPLITUDE	Controls the modulation index. This index (K) is a ratio of Frequency spread (Δf) and spreading rate (f_m) . 00: K = 4 01: K = 6 10: K = 8 11: K = 10	RW	0x0
1:0	DSS_SPREADING_RATE	Controls the rate of frequency modulation: 00: 62.5 to 125 KHz 01: 125 to 250 KHz 10: 250 to 500 KHz 11: 500 to 1000 KHz	RW	0x0

Table 7-163. Register Call Summary for Register CONTROL_DSS_DPLL_SPREADING

SCM Register Manual

- SCM Register Summary: [0]

Table 7-164. CONTROL_CORE_DPLL_SPREADING

Address Offset	0x0000 01E4		
Physical Address	0x4800 2454	**Instance**	GENERAL
Description	This register controls the EMI Reduction feature for CORE domain DPLL		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																								CORE_SPREADING_ENABLE_STATUS	RESERVED		CORE_SPREADING_ENABLE	CORE_SPREADING_AMPLITUDE		CORE_SPREADING_RATE	

Bits	Field Name	Description	Type	Reset
31:8	RESERVED	Reads return zero	R	0x000000
7	CORE_SPREADING_ENABLE_ STATUS	Indicates the status of the SSC feature	R	-
6:5	RESERVED	Reads return zero	R	0x0
4	CORE_SPREADING_ENABLE	Enables/disables EMI Reduction feature (Spreading): 0: Modulation stops at the end of the current modulation cycle. 1: Modulation cycle is started.	RW	0x0

Bits	Field Name	Description	Type	Reset
3:2	CORE_SPREADING_ AMPLITUDE	Controls the modulation index. This index (K) is a ratio of Frequency spread (Δf) and spreading rate (f_m) . 00: K = 4 01: K = 6 10: K = 8 11: K = 10	RW	0x0
1:0	CORE_SPREADING_RATE	Controls the rate of frequency modulation: 00: 62.5 to 125 KHz 01: 125 to 250 KHz 10: 250 to 500 KHz 11: 500 to 1000 KHz	RW	0x0

Table 7-165. Register Call Summary for Register CONTROL_CORE_DPLL_SPREADING

SCM Register Manual

- SCM Register Summary: [0]

Table 7-166. CONTROL_PER_DPLL_SPREADING

Address Offset	0x0000 01E8		
Physical Address	0x4800 2458	**Instance**	GENERAL
Description	This register controls the EMI Reduction feature for PER domain DPLL		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																								PER_SPREADING_ENABLE_STATUS	RESERVED		PER_SPREADING_ENABLE	PER_SPREADING_AMPLITUDE		PER_SPREADING_RATE	

Bits	Field Name	Description	Type	Reset
31:8	RESERVED	Reads return zero	R	0x000000
7	PER_SPREADING_ENABLE_ STATUS	Indicates the status of the Spreading feature	R	-
6:5	RESERVED	Reads return zero	R	0x0
4	PER_SPREADING_ENABLE	Enables/disables EMI Reduction feature (Spreading): 0: Modulation stops at the end of the current modulation cycle. 1: Modulation cycle is started.	RW	0x0

Bits	Field Name	Description	Type	Reset
3:2	PER_SPREADING_ AMPLITUDE	Controls the modulation index. This index (K) is a ratio of the spreading frequency (Δf) and spreading rate (f_m) . 00: K = 4 01: K = 6 10: K = 8 11: K = 10	RW	0x0
1:0	PER_SPREADING_RATE	Controls the rate of frequency modulation: 00: 62.5 to 125 KHz 01: 125 to 250 KHz 10: 250 to 500 KHz 11: 500 to 1000 KHz	RW	0x0

Table 7-167. Register Call Summary for Register CONTROL_PER_DPLL_SPREADING

SCM Register Manual

- SCM Register Summary: [0]

Table 7-168. CONTROL_USBHOST_DPLL_SPREADING

Address Offset	0x0000 01EC		
Physical Address	0x4800 245C	**Instance**	GENERAL
Description	This register controls the EMI Reduction feature for USBHOST DPLL		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																								USBHOST_SPREADING_ENABLE_STATUS	RESERVED		USBHOST_SPREADING_ENABLE	USBHOST_SPREADING_AMPLITUDE		USBHOST_SPREADING_RATE	

Bits	Field Name	Description	Type	Reset
31:8	RESERVED	Reads return zero	R	0x000000
7	USBHOST_SPREADING_ ENABLE_STATUS	Indicates the status of the Spreading feature	R	-
6:5	RESERVED	Reads return zero	R	0x0
4	USBHOST_SPREADING_ ENABLE	Enables/disables EMI Reduction feature (Spreading): 0: Modulation stops at the end of the current modulation cycle. 1: Modulation cycle is started.	RW	0x0

Bits	Field Name	Description	Type	Reset
3:2	USBHOST_SPREADING_AMPLITUDE	Controls the modulation index. This index (K) is a ratio of Frequency spread (Δf) and spreading rate (f_m) . 00: K = 4 01: K = 6 10: K = 8 11: K = 10	RW	0x0
1:0	USBHOST_SPREADING_RATE	Controls the rate of frequency modulation: 00: 62.5 to 125 KHz 01: 125 to 250 KHz 10: 250 to 500 KHz 11: 500 to 1000 KHz	RW	0x0

Table 7-169. Register Call Summary for Register CONTROL_USBHOST_DPLL_SPREADING

SCM Register Manual
- SCM Register Summary: [0]

Table 7-170. CONTROL_SDRC_SHARING

Address Offset	0x0000 01F0		
Physical Address	0x4800 2460	**Instance**	GENERAL
Description	SDRC Sharing configuration register		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED	SDRCSHARINGLOCK	SDRCSHARING																													

Bits	Field Name	Description	Type	Reset
31	RESERVED	Reserved for non-GP devices	Reserved	0x0
30	SDRCSHARINGLOCK	Exported value to SDRC.SDRC_SHARING[30] For more information, see Section 7.4.9, *SDRC Registers*	R/W	0x0
29:0	SDRCSHARING	Exported value to SDRC.SDRC_SHARING[29:0] For more information, see Section 7.4.9, *SDRC Registers*	R/W	0x00002700

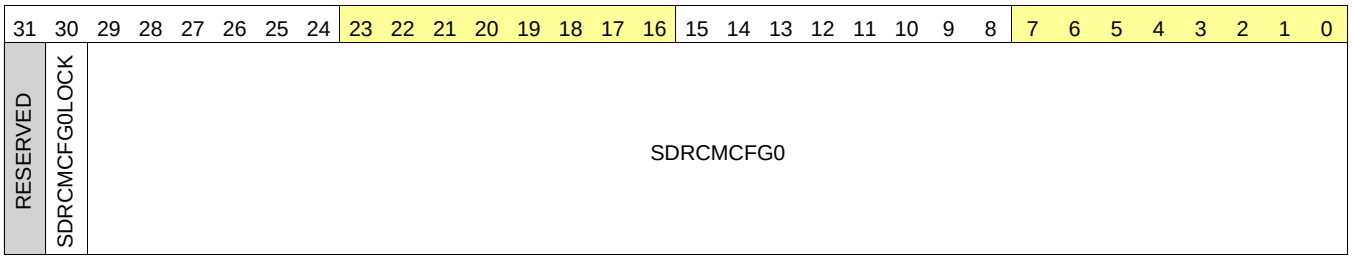
Table 7-171. Register Call Summary for Register CONTROL_SDRC_SHARING

SCM Functional Description
- SDRC Registers: [0] [1] [2]

SCM Register Manual
- SCM Register Summary: [3]

Table 7-172. CONTROL_SDRC_MCFG0

Address Offset	0x0000 01F4		
Physical Address	0x4800 2464	**Instance**	GENERAL
Description	SDRC MCFG Configuration register-0		
Type	RW		



Bits	Field Name	Description	Type	Reset
31	RESERVED	Reserved for non-GP devices	Reserved	0x0
30	SDRCMCFG0LOCK	Exported value to SDRC.SDRC_MCFG_0[30] For more information, see Section 7.4.9, *SDRC Registers*	R/W	0x0
29:0	SDRCMCFG0	Exported value to SDRC.SDRC_MCFG_0[29:0] For more information, see Section 7.4.9, *SDRC Registers*	R/W	0x00300000

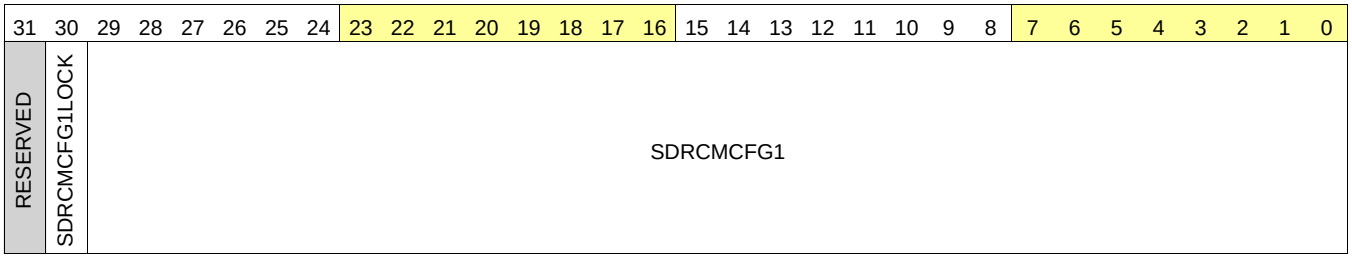
Table 7-173. Register Call Summary for Register CONTROL_SDRC_MCFG0

SCM Functional Description
- SDRC Registers: [0] [1] [2]

SCM Register Manual
- SCM Register Summary: [3]

Table 7-174. CONTROL_SDRC_MCFG1

Address Offset	0x0000 01F8		
Physical Address	0x4800 2468	**Instance**	GENERAL
Description	SDRC MCFG Configuration register-1		
Type	RW		



Bits	Field Name	Description	Type	Reset
31	RESERVED	Reserved for non-GP devices	Reserved	0x0
30	SDRCMCFG1LOCK	Exported value to SDRC.SDRC_MCFG_1[30] For more information, see Section 7.4.9, *SDRC Registers*	R/W	0x0
29:0	SDRCMCFG1	Exported value to SDRC.SDRC_MCFG_1[29:0] For more information, see Section 7.4.9, *SDRC Registers*	R/W	0x00300000

Table 7-175. Register Call Summary for Register CONTROL_SDRC_MCFG1

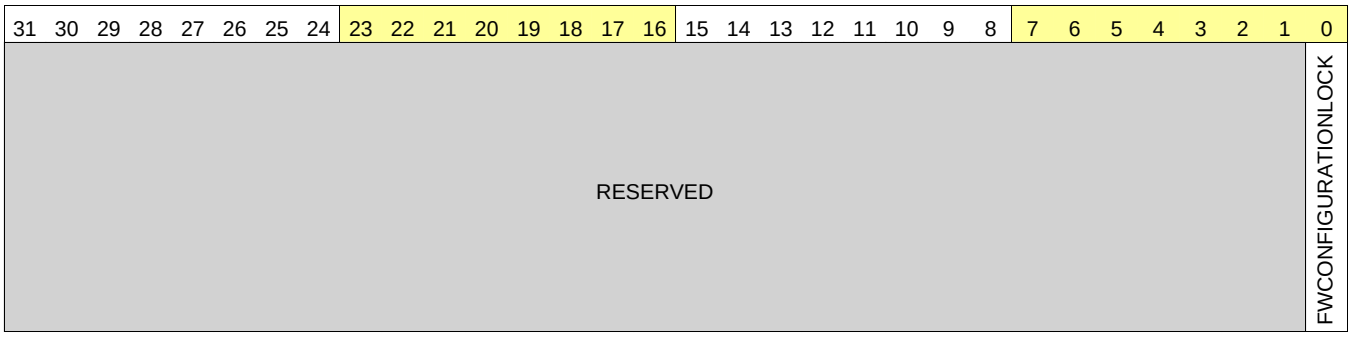
SCM Functional Description
- SDRC Registers: [0] [1] [2]

Table 7-175. Register Call Summary for Register CONTROL_SDRC_MCFG1 (continued)

SCM Register Manual • SCM Register Summary: [3]

Table 7-176. CONTROL_MODEM_FW_CONFIGURATION_LOCK

Address Offset	0x0000 01FC		
Physical Address	0x4800 246C	**Instance**	GENERAL
Description	Allows locking of the modem isolation registers in the SCM until the next battery removal.		
Type	RW		



Bits	Field Name	Description	Type	Reset
31:1	RESERVED	Read returns reset value.	R	0x00000000
0	FWCONFIGURATION LOCK	Allows locking of the modem isolation registers in the SCM until the next battery removal. 0x0: Not locked 0x1: Locked	Refer to Table 7-178	0x0

Table 7-177. Register Call Summary for Register CONTROL_MODEM_FW_CONFIGURATION_LOCK

SCM Register Manual • SCM Register Summary: [0] • GENERAL Register Description: [1] [2] [3] [4] [5] [6] [7] [8] [9] [10]

Table 7-178. Type Value For CONTROL_MODEM_FW_CONFIGURATION_LOCK

CONTROL_MODEM_FW_CONFIGURATION_LOCK[0] FWCONFIGURATIONLOCK bit	0	1
FWCONFIGURATIONLOCK	R/W	R

Table 7-179. CONTROL_MODEM_MEMORY_RESOURCES_CONF

Address Offset	0x0000 0200		
Physical Address	0x4800 2470	**Instance**	GENERAL
Description	Modem Memory Resources Conf		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CMDWTOCMRAMSWITCH	MODEMSTACKMEMORYSIZE				MODEMSMSMEMORYSIZE			MODEMGPMCRESERVEDS2SIZE								MODEMGPMCRESERVEDS1SIZE								MODEMGPMCRESERVEDBASEADDR							

Bits	Field Name	Description	Type	Reset
31	CMDWTOCMRAMSWITCH	Select if CMDWT or OCMRAM is accessible by the Modem	Refer to Table 7-181	0x0
30:27	MODEMSTACKMEMORYSIZE	Configuration of the modem stack memory size	Refer to Table 7-181	0x0
26:22	MODEMSMSMEMORYSIZE	Configuration of the SMS modem memory	Refer to Table 7-181	0x00
21:17	MODEMGPMCRESERVEDS2SIZE	Configuration of the GPMC modem shared section size	Refer to Table 7-181	0x00
16:12	MODEMGPMCRESERVEDS1SIZE	Configuration of the GPMC modem reserved section size	Refer to Table 7-181	0x00
11:0	MODEMGPMCRESERVEDBASEADDR	Configuration of the GPMC base address of the modem reserved section. This base address is configurable in 1Mbyte step.	Refer to Table 7-181	0x000

Table 7-180. Register Call Summary for Register CONTROL_MODEM_MEMORY_RESOURCES_CONF

SCM Register Manual

- SCM Register Summary: [0]

Table 7-181. Type Value For CONTROL_MODEM_MEMORY_RESOURCES_CONF

CONTROL_MODEM_FW_CONFIGURATION_LOCK[0] FWCONFIGURATIONLOCK bit	0	1
CMDWTOCMRAMSWITCH	R/W	R
MODEMSTACKMEMORYSIZE	R/W	R
MODEMSMSMEMORYSIZE	R/W	R
MODEMGPMCRESERVEDS2SIZE	R/W	R
MODEMGPMCRESERVEDS1SIZE	R/W	R
MODEMGPMCRESERVEDBASEADDR	R/W	R

Table 7-182. CONTROL_MODEM_GPMC_DT_FW_REQ_INFO

Address Offset	0x0000 0204		
Physical Address	0x4800 2474	**Instance**	GENERAL
Description	Modem GPMC Default firewall request info register		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																MODEMGPMCDTFWREQINFO															

Bits	Field Name	Description	Type	Reset
31:16	RESERVED	Read returns reset value.	R	0x0000
15:0	MODEMGPMCDTFW REQINFO	Exported values to the GPMC firewall region1 REQ_INFO_PERMISSION field	Refer to Table 7-184	0xFFFF

Table 7-183. Register Call Summary for Register CONTROL_MODEM_GPMC_DT_FW_REQ_INFO

SCM Register Manual

- SCM Register Summary: [0]

Table 7-184. Type Value For CONTROL_MODEM_GPMC_DT_FW_REQ_INFO

CONTROL_MODEM_FW_CONFIGURATION_LOCK[0] FWCONFIGURATIONLOCK bit	0	1
MODEMGPMCDTFWREQINFO	R/W	R

Table 7-185. CONTROL_MODEM_GPMC_DT_FW_RD

Address Offset	0x0000 0208		
Physical Address	0x4800 2478	**Instance**	GENERAL
Description	Modem GPMC Default firewall read permission register		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																MODEMGPMCDTFWRD															

Bits	Field Name	Description	Type	Reset
31:16	RESERVED	Read returns reset value.	R	0x0000
15:0	MODEMGPMCDTFWRD	Exported values to the GPMC firewall region1 READ_PERMISSION field	Refer to Table 7-187	0xFFFF

Table 7-186. Register Call Summary for Register CONTROL_MODEM_GPMC_DT_FW_RD

SCM Register Manual

- SCM Register Summary: [0]

Table 7-187. Type Value For CONTROL_MODEM_GPMC_DT_FW_RD

CONTROL_MODEM_FW_CONFIGURATION_LOCK[0] FWCONFIGURATIONLOCK bit	0	1
MODEMGPMCDTFWRD	R/W	R

Table 7-188. CONTROL_MODEM_GPMC_DT_FW_WR

Address Offset	0x0000 020C		
Physical Address	0x4800 247C	**Instance**	GENERAL
Description	Modem GPMC Default firewall write permission register		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																MODEMGPMCDTFWWR															

Bits	Field Name	Description	Type	Reset
31:16	RESERVED	Read returns reset value.	R	0x0000
15:0	MODEMGPMCDTFWWR	Exported values to the GPMC firewall region1 WRITE_PERMISSION field	Refer to Table 7-190	0xFFFF

Table 7-189. Register Call Summary for Register CONTROL_MODEM_GPMC_DT_FW_WR

SCM Register Manual

- SCM Register Summary: [0]

Table 7-190. Type Value For CONTROL_MODEM_GPMC_DT_FW_WR

CONTROL_MODEM_FW_CONFIGURATION_LOCK[0] FWCONFIGURATIONLOCK bit	0	1
MODEMGPMCDTFWWR	R/W	R

Table 7-191. CONTROL_MODEM_GPMC_BOOT_CODE

Address Offset	0x0000 0210		
Physical Address	0x4800 2480	**Instance**	GENERAL
Description	GPMC Flash Boot Code protection register		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																										GPMCBOOTCODEWRITE	GPMCBOOTCODESIZE						

Bits	Field Name	Description	Type	Reset
31:6	RESERVED	Read returns reset value.	R	0x0000000
5	GPMCBOOTCODEWRITE PROTECTED	When set HIGH the Flash Boot Code area is write protected 0x0: When cleared Flash boot code area is not write protected 0x1: When set Flash boot code area is write protected	Refer to Table 7-193	0
4:0	GPMCBOOTCODESIZE	Size of the Flash boot code to protect	Refer to Table 7-193	0x00

Table 7-192. Register Call Summary for Register CONTROL_MODEM_GPMC_BOOT_CODE

SCM Register Manual

- SCM Register Summary: [0]

Table 7-193. Type Value For CONTROL_MODEM_GPMC_BOOT_CODE

CONTROL_MODEM_FW_CONFIGURATION_LOCK[0] FWCONFIGURATIONLOCK bit	0	1
GPMCBOOTCODEWRITEPROTECTED	R/W	R
GPMCBOOTCODESIZE	R/W	R

Table 7-194. CONTROL_MODEM_SMS_RG_ATT1

Address Offset	0x0000 0214		
Physical Address	0x4800 2484	**Instance**	GENERAL
Description	Modem SMS Default firewall register		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SMSRGATT1																															

Bits	Field Name	Description	Type	Reset
31:0	SMSRGATT1	Exported values to the SMS firewall region1 SMS_RG_ATT1 field	Refer to Table 7-196	0xFFFF FFFF

Table 7-195. Register Call Summary for Register CONTROL_MODEM_SMS_RG_ATT1

SCM Register Manual

- SCM Register Summary: [0]

Table 7-196. Type Value For CONTROL_MODEM_SMS_RG_ATT1

CONTROL_MODEM_FW_CONFIGURATION_LOCK[0] FWCONFIGURATIONLOCK bit	0	1
SMSRGATT1	R/W	R

Table 7-197. CONTROL_MODEM_SMS_RG_RDPERM1

Address Offset	0x0000 0218		
Physical Address	0x4800 2488	**Instance**	GENERAL
Description	Modem SMS Default firewall read permission register		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																SMSRGRDPERM1															

Bits	Field Name	Description	Type	Reset
31:16	RESERVED	Read returns reset value.	R	0x0000
15:0	SMSRGRDPERM1	Exported values to the SMS firewall region1 SMS_RG_RDPERM1 field	Refer to Table 7-199	0xFFFF

Table 7-198. Register Call Summary for Register CONTROL_MODEM_SMS_RG_RDPERM1

SCM Register Manual

- SCM Register Summary: [0]

Table 7-199. Type Value For CONTROL_MODEM_SMS_RG_RDPERM1

CONTROL_MODEM_FW_CONFIGURATION_LOCK[0] FWCONFIGURATIONLOCK bit	0	1
SMSRGRDPERM1	R/W	R

Table 7-200. CONTROL_MODEM_SMS_RG_WRPERM1

Address Offset	0x0000 021C		
Physical Address	0x4800 248C	**Instance**	GENERAL
Description	Modem SMS Default firewall write permission register		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																SMSRGWRPERM1															

Bits	Field Name	Description	Type	Reset
31:16	RESERVED	Read returns reset value.	R	0x0000
15:0	SMSRGWRPERM1	Exported values to the SMS firewall region1 SMS_RG_WRPERM1 field	Refer to Table 7-202	0xFFFF

Table 7-201. Register Call Summary for Register CONTROL_MODEM_SMS_RG_WRPERM1

SCM Register Manual

- SCM Register Summary: [0]

Table 7-202. Type Value For CONTROL_MODEM_SMS_RG_WRPERM1

CONTROL_MODEM_FW_CONFIGURATION_LOCK[0] FWCONFIGURATIONLOCK bit	0	1
SMSRGWRPERM1	R/W	R

Table 7-203. CONTROL_MODEM_D2D_FW_DEBUG_MODE

Address Offset	0x0000 0220		
Physical Address	0x4800 2490	**Instance**	GENERAL
Description	D2D firewall debug mode register		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																															D2DFWDEBUGMODE

Bits	Field Name	Description	Type	Reset
31:1	RESERVED	Read returns reset value.	R	0x00000000
0	D2DFWDEBUGMODE	When set to high the L3-D2D FW is in debug mode.	Refer to Table 7-205	0x0

Table 7-204. Register Call Summary for Register CONTROL_MODEM_D2D_FW_DEBUG_MODE

SCM Register Manual

- SCM Register Summary: [0]

Table 7-205. Type Value For CONTROL_MODEM_D2D_FW_DEBUG_MODE

CONTROL_MODEM_FW_CONFIGURATION_LOCK[0] FWCONFIGURATIONLOCK bit	0	1
D2DFWDEBUGMODE	R/W	R

Table 7-206. CONTROL_DPF_OCM_RAM_FW_ADDR_MATCH

Address Offset	0x0000 0228		
Physical Address	0x4800 2498	**Instance**	SYSC_GENERAL1
Description	OCM RAM Dynamic Power Framework Handing		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								REGIONOCMRAMFWADDRMATCH																									

Bits	Field Name	Description	Type	Reset
31:20	RESERVED	Read returns reset value	R	0x00
19:0	REGIONOCMRAMFW ADDRMATCH	Refer to L3 FW addr_match field	R	0x000000

Table 7-207. Register Call Summary for Register CONTROL_DPF_OCM_RAM_FW_ADDR_MATCH

SCM Register Manual

- SCM Register Summary: [0]

Table 7-208. CONTROL_DPF_OCM_RAM_FW_REQINFO

Address Offset	0x0000 022C		
Physical Address	0x4800 249C	**Instance**	SYSC_GENERAL1
Description	OCM RAM Dynamic Power Framework Handing		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																REGIONOCMRAMFWREQINFO															

Bits	Field Name	Description	Type	Reset
31:16	RESERVED	Read returns reset value	R	0x00
15:0	REGIONOCMRAMFW REQINFO	Refer to L3 FW REQINFO permission field	R	0xFFFF

Table 7-209. Register Call Summary for Register CONTROL_DPF_OCM_RAM_FW_REQINFO

SCM Register Manual

- SCM Register Summary: [0]

Table 7-210. CONTROL_DPF_OCM_RAM_FW_WR

Address Offset	0x0000 0230		
Physical Address	0x4800 24A0	**Instance**	SYSC_GENERAL1
Description	OCM RAM Dynamic Power Framework Handing		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																REGIONOCMRAMFWWR															

Bits	Field Name	Description	Type	Reset
31:16	RESERVED	Read returns reset value	R	0x00000
15:0	REGIONOCMRAMFWWR	Refer to L3 FW WR permission field	R	0xFFFF

Table 7-211. Register Call Summary for Register CONTROL_DPF_OCM_RAM_FW_WR

SCM Register Manual

- SCM Register Summary: [0]

Table 7-212. CONTROL_DPF_REGION4_GPMC_FW_ADDR_MATCH

Address Offset	0x0000 0234		
Physical Address	0x4800 24A4	**Instance**	GENERAL
Description	GPMC Dynamic Power Framework Handing		
Type	RW		

29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
REGION4GPMCFWADDRMATCH																													

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED	REGION4GPMCFWADDRMATCH																														

Bits	Field Name	Description	Type	Reset
31:30	RESERVED	Read returns reset value	R	0x00000000
29:0	REGION4GPMCFW ADDRMATCH	Exported value to L3 FW region 4 GPMC ADDR_MATCH4 field	R	0x00000000

Table 7-213. Register Call Summary for Register CONTROL_DPF_REGION4_GPMC_FW_ADDR_MATCH

SCM Register Manual

- SCM Register Summary: [0]

Table 7-214. CONTROL_DPF_REGION4_GPMC_FW_REQINFO

Address Offset	0x0000 0238		
Physical Address	0x4800 24A8	**Instance**	GENERAL
Description	GPMC Dynamic Power Framework Handing		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																REGION4GPMCFWREQINFO															

Bits	Field Name	Description	Type	Reset
31:16	RESERVED	Read returns reset value.	R	0x0000
15:0	REGION4GPMCFWREQINFO	Exported value to L3 FW region 4 GPMC REQINFO_PERMISSION_4 field	R	0xFFFF

Table 7-215. Register Call Summary for Register CONTROL_DPF_REGION4_GPMC_FW_REQINFO

SCM Register Manual

- SCM Register Summary: [0]

Table 7-216. CONTROL_DPF_REGION4_GPMC_FW_WR

Address Offset	0x0000 023C		
Physical Address	0x4800 24AC	**Instance**	GENERAL
Description	GPMC Dynamic Power Framework Handing		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																REGION4GPMCFWWR															

Bits	Field Name	Description	Type	Reset
31:16	RESERVED	Read returns reset value.	R	0x0000
15:0	REGION4GPMCFWWR	Exported value to L3 FW region 4 GPMC WRITE_PERMISSION_4 field	R	0xFFFF

Table 7-217. Register Call Summary for Register CONTROL_DPF_REGION4_GPMC_FW_WR

SCM Register Manual

- SCM Register Summary: [0]

Table 7-218. CONTROL_DPF_REGION1_IVA2_FW_ADDR_MATCH

Address Offset	0x0000 0240		
Physical Address	0x4800 24B0	**Instance**	GENERAL
Description	IVA2 Dynamic Power Framework Handing		
Type	RW		

23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
REGION1IVA2FWADDRMATCH																							

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								REGION1IVA2FWADDRMATCH																							

Bits	Field Name	Description	Type	Reset
31:24	RESERVED	Read returns reset value	R	0x0
23:0	REGION1IVA2FW ADDRMATCH	Exported value to L3 FW region 1 IVA2 ADDR_MATCH1 field	R	0x0

Table 7-219. Register Call Summary for Register CONTROL_DPF_REGION1_IVA2_FW_ADDR_MATCH

SCM Register Manual

- SCM Register Summary: [0]

Table 7-220. CONTROL_DPF_REGION1_IVA2_FW_REQINFO

Address Offset	0x0000 0244		
Physical Address	0x4800 24B4	**Instance**	GENERAL
Description	IVA2 Dynamic Power Framework Handing		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																REGION1IVA2FWREQINFO															

Bits	Field Name	Description	Type	Reset
31:16	RESERVED	Read returns reset value.	R	0x0
15:0	REGION1IVA2FW REQINFO	Exported value to L3 FW region 1 IVA2 REQINFO_PERMISSION_1 field	R	0xFFFF FFFF

Table 7-221. Register Call Summary for Register CONTROL_DPF_REGION1_IVA2_FW_REQINFO

SCM Register Manual

- SCM Register Summary: [0]

Table 7-222. CONTROL_DPF_REGION1_IVA2_FW_WR

Address Offset	0x0000 0248		
Physical Address	0x4800 24B8	**Instance**	GENERAL
Description	IVA2 Dynamic Power Framework Handing		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																								REGION1IVA2FWWR							

Bits	Field Name	Description	Type	Reset
31:13	RESERVED	Read returns reset value.	R	0x0
12:0	REGION1IVA2FWWR	Exported value to L3 FW region 1 IVA2 WRITE_PERMISSION_1 field	R	0x1FFF

Table 7-223. Register Call Summary for Register CONTROL_DPF_REGION1_IVA2_FW_WR

SCM Register Manual

- SCM Register Summary: [0]

Table 7-224. CONTROL_PBIAS_LITE

Address Offset	0x0000 02B0		
Physical Address	0x4800 2520	**Instance**	GENERAL
Description	This register controls the settings for PBIAS LITE MMC/SD/SDIO1 pins		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																PBIASLITESUPPLYHIGH1	RESERVED			PBIASLITEVMODEERROR1	PBIASSPEEDCTRL1	PBIASLITEPWRDNZ1	PBIASLITEVMODE1	PBIASLITESUPPLYHIGH0	RESERVED			PBIASLITEVMODEERROR0	PBIASSPEEDCTRL0	PBIASLITEPWRDNZ0	PBIASLITEVMODE0

Bits	Field Name	Description	Type	Reset
31:16	RESERVED	Reserved for future use	R	0x0000000
15	PBIASLITESUPPLY HIGH1	Status indicating whether PBIAS1 is supplied by 1.8 V or 3.0 V SIM_VDDS 1: SIM_VDDS = 3.0 V 0: SIM_VDDS = 1.8 V	R	0
14:12	RESERVED	Reserved for future use	R	0
11	PBIASLITEVMODE ERROR1	Status indicating if the software programmed VMODE level matches the SUPPLY_HI output signal 0b0 => VMODE Level same or VMODE level not considered 0b1 => Indicates VMODE_LEVEL not same as SUPPLY_HI_OUT	R	0x0
10	PBIASSPEEDCTRL1	Speed control for PBIAS1 associated I/O cell 0b0 => 26 MHz I/O max speed 0b1 => 52 MHz I/O max speed	RW	0x0
9	PBIASLITEPWRDNZ1	Input signal referenced to VDD2. Software must keep this signal low when SIM_VDDS is ramping up 0b0 => SIM_VDDS ramping up 0b1 => SIM_VDDS stable	RW	0x0
8	PBIASLITEVMODE1	SIM_VDDS voltage level information control from software 0b0 => SIM_VDDS = 1.8 V 0b1 => SIM_VDDS = 3.0 V	RW	0x1
7	PBIASLITESUPPLY HIGH0	Status indicating if PBIAS0 is supplied by 1.8 V or 3.0 V MMC1_VDDS 0b0 => PBIAS0 is supplied by MMC1_VDDS = 1.8 V 0b1 => PBIAS0 is supplied by MMC1_VDDS = 3.0 V	R	0x0
6:4	RESERVED	Read returns reset value.	R	0x0
3	PBIASLITEVMODE ERROR0	Status indicating if the software programmed VMODE level matches the SUPPLY_HI output signal 0b0 => VMODE Level same or VMODE level not considered 0b1 => Indicates VMODE_LEVEL not same as SUPPLY_HI_OUT	R	0x0
2	PBIASSPEEDCTRL0	Speed Control for MMC I/O 0b0 => 26 MHz I/O max speed 0b1 => 52 MHz I/O max speed	RW	0x0

Bits	Field Name	Description	Type	Reset
1	PBIASLITEPWRDNZ0	Input signal referenced to VDD2. Software must keep this signal low when MMC1_VDDS is ramping up 0b0 => MMC1_VDDS ramping up 0b1 => MMC1_VDDS stable	RW	0x0
0	PBIASLITEVMODE0	MMC1_VDDS voltage level information control from software 0b0 => MMC1_VDDS = 1.8 V 0b1 => MMC1_VDDS = 3.0 V	RW	0x1

Table 7-225. Register Call Summary for Register CONTROL_PBIAS_LITE

SCM Functional Description
- Extended-Drain I/O Pin and PBIAS Cell: [0] [1]
- PBIAS LITE Control Register: [2] [3]

SCM Programming Model
- Extended-Drain I/Os and PBIAS Cells Programming Guide: [4] [5] [6] [7] [8] [9] [10] [11] [12] [13] [14]
- PBIAS Error Generation: [15] [16]
- Speed Control and Voltage Supply State: [17]

SCM Register Manual
- SCM Register Summary: [18]

Table 7-226. CONTROL_TEMP_SENSOR

Address Offset	0x0000 02B4		
Physical Address	0x4800 2524	**Instance**	GENERAL
Description	temperature sensor register		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																						CONTCONV	SOC	EOCZ	TEMP						

Bits	Field Name	Description	Type	Reset
31:10	RESERVED	Read returns reset value.	R	0x0
9	CONTCONV	VDD level digital inputs. When high the ADC is in continuous conversion mode 0 : ADC Single Conversion Mode; 1 : ADC Continuous Conversion Mode	RW	0x0
8	SOC	ADC Start of Conversion. A transition to high starts a new ADC conversion cycle	RW	0x0
7	EOCZ	ADC End of Conversion. Active low, when CTRL_ TEMP[6:0] is valid	R	0x0
6:0	TEMP	Temperature data from the ADC. Valid if EOCZ is low	R	0x0

Table 7-227. Register Call Summary for Register CONTROL_TEMP_SENSOR

SCM Functional Description
- Band Gap Voltage and Temperature Sensor: [0] [1]
- Temperature Sensor: [2] [3] [4] [5]
- Temperature Sensor Control Register: [6] [7]

SCM Register Manual
- SCM Register Summary: [8]

Table 7-228. CONTROL_CSI

Address Offset	0x0000 02C0		
Physical Address	0x4800 2530	**Instance**	GENERAL
Description	Controls CSIb reciever trimming setting		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CSI_B_MUX_CTRL	RESERVED							CSI_B_TRIM_FUSE			CSI_B_TRIM_REG					RESERVED															

Bits	Field Name	Description	Type	Reset
31	CSI_B_MUX_CTRL	Override CSIb reciever trimming Fuse setting	RW	0x0
30:26	RESERVED	Read returns reset value.	R	0x0-
25:21	CSI_B_TRIM_FUSE	CSIb reciever trimming Fuse value	R	0x--
20:16	CSI_B_TRIM_REG	CSIb reciever trimming Register control	RW	0x00
15:0	RESERVED	Read returns reset value.	R	0x0-

Table 7-229. Register Call Summary for Register CONTROL_CSI

SCM Functional Description

- CSI Reciever Control Register: [0] [1]

SCM Register Manual

- SCM Register Summary: [2]

Table 7-230. CONTROL_DPF_MAD2D_FW_ADDR_MATCH

Address Offset	0x0000 02C8		
Physical Address	0x4800 2538	**Instance**	GENERAL
Description	MAD2D Dynamic Power Framework Handing		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								REGIONMAD2DFWADDRMATCH																							

Bits	Field Name	Description	Type	Reset
31:27	RESERVED	Read returns reset value	R	0x00
26:0	REGIONMAD2DFWADDRMATCH	Refer to L3 FW addr match field	R	0x000000

Table 7-231. Register Call Summary for Register CONTROL_DPF_MAD2D_FW_ADDR_MATCH

SCM Register Manual

- SCM Register Summary: [0]

Table 7-232. CONTROL_DPF_MAD2D_FW_REQINFO

Address Offset	0x0000 02CC		
Physical Address	0x4800 253C	**Instance**	GENERAL
Description	MAD2D Dynamic Power Framework Handing		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																MAD2DFWREQINFO															

Bits	Field Name	Description	Type	Reset
31:16	RESERVED	Read returns reset value	R	0x00
15:0	MAD2DFWREQINFO	Refer to L3 FW REQINFO permission field	R	0xFFFF

Table 7-233. Register Call Summary for Register CONTROL_DPF_MAD2D_FW_REQINFO

SCM Register Manual

- SCM Register Summary: [0]

Table 7-234. CONTROL_DPF_MAD2D_FW_WR

Address Offset	0x0000 02D0		
Physical Address	0x4800 2540	**Instance**	GENERAL
Description	MAD2D Dynamic Power Framework Handing		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																REGIONMAD2DFWWR															

Bits	Field Name	Description	Type	Reset
31:16	RESERVED	Read returns reset value	R	0x00000
15:0	REGIONMAD2DFWWR	Refer to L3 FW WR permission field	R	0xFFFF

Table 7-235. Register Call Summary for Register CONTROL_DPF_MAD2D_FW_WR

SCM Register Manual

- SCM Register Summary: [0]

Table 7-236. CONTROL_IDCODE

Address Offset	0x307F94		
Physical Address	0x4830 A204	**Instance**	SYSC_GENERAL1
Description	Device IDCODE		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
VERSION				HAWKEYE												TI_IDM															Reserved

Bits	Field Name	Description	Type	Reset
31:28	VERSION	Revision number	R	See (1).
27:12	HAWKEYE	Hawkeye number	R	See (2)
11:1	TI_IDM	Manufacturer identity (TI)	R	See (1).

(1) For more information, see Section 1.5, *OMAP34xx Family*, in Chapter 1, *Introduction*.
(2) TI internal data

Bits	Field Name	Description	Type	Reset
0	Reserved	Read returns reset value.	R	1

Table 7-237. Register Call Summary for Register CONTROL_IDCODE

SCM Register Manual
- SCM Register Summary: [0]

7.6.3.4 MEM_WKUP Register Description

Physical adresses 0x4800 2600 to 0x4800 29FC are memories mapped for save and restore location (1KB).

- 0x4800 2600 - 0x4800 28A0: nonaccessible (pad configuration)
- 0x4800 28A4 - 0x4800 29FC: user accessible

7.6.3.5 PADCONFS_WKUP Register Description

Each 32-bit PADCONF register gathers the configuration of two pads. For example, CONTROL.CONTROL_PADCONF_GPMC_7 is used to configure gpmc_7 pad (bits [15:0]) and gpmc_8 pad (bits [32:16]). See Figure 7-8 for more information about PADCONF registers.

According to the pad type, some features are configurable or not. Table 7-73 gives the description of a fully configurable pad.

Table 7-238 describes the reset values, the capabilities, and the corresponding register for each pad.

Notes:

- All '-' assume that the corresponding bit is not available. These bits are considered as Reserved.
- The reset value for Reserved bits is 0.
- In the Mux Reset States and PU/PD Reset States columns of Table 7-238, a dash (-) indicates that the field is hardwired to logic 0. No corresponding control block ports are implemented for these bits.

Table 7-238. CONTROL_PADCONF_WKUP_CAPABILITIES

REGISTER NAME	Pad Name	Physical Address	WakeUpx	Off Mode	Input Enable	Reserved	PU/PD	Mux Mode
CONTROL_PADCONF_I2C4_SCL[15:0]	i2c4_scl	0x4800 2A00	--	-----	0b1	0b000	0b11	0b000
CONTROL_PADCONF_I2C4_SCL[31:16]	i2c4_sda	0x4800 2A00	--	-----	0b1	0b000	0b11	0b000
CONTROL_PADCONF_SYS_32K[15:0]	sys_32k	0x4800 2A04	--	-----	0b1	0b000	--	---
CONTROL_PADCONF_SYS_32K[31:16]	sys_clkreq	0x4800 2A04	0b00	-----	0b1	0b000	0b00	0b000
CONTROL_PADCONF_SYS_NRESWARM [15:0]	sys_ nreswarm	0x4800 2A08	0b00	-----	0b1	0b000	0b11	0b000
CONTROL_PADCONF_SYS_NRESWARM [31:16]	sys_boot0	0x4800 2A08	0b00	0b--000	0b1	0b000	0b00	0b000
CONTROL_PADCONF_SYS_BOOT1[15:0]	sys_boot1	0x4800 2A0C	0b00	0b--000	0b1	0b000	0b00	0b000
CONTROL_PADCONF_SYS_BOOT1[31:16]	sys_boot2	0x4800 2A0C	0b00	0b--000	0b1	0b000	0b00	0b000
CONTROL_PADCONF_SYS_BOOT3[15:0]	sys_boot3	0x4800 2A10	0b00	0b--000	0b1	0b000	0b00	0b000
CONTROL_PADCONF_SYS_BOOT3[31:16]	sys_boot4	0x4800 2A10	0b00	0b--000	0b1	0b000	0b00	0b000
CONTROL_PADCONF_SYS_BOOT5[15:0]	sys_boot5	0x4800 2A14	0b00	0b--000	0b1	0b000	0b00	0b000
CONTROL_PADCONF_SYS_BOOT5[31:16]	sys_boot6	0x4800 2A14	0b00	0b--000	0b1	0b000	0b00	0b000
CONTROL_PADCONF_SYS_OFF_MODE [15:0]	sys_off_ mode	0x4800 2A18	0b00	-----	0b1	0b000	0b01	0b111
CONTROL_PADCONF_SYS_OFF_MODE [31:16]	sys_clkout1	0x4800 2A18	0b00	-----	0b1	0b000	0b01	0b111
CONTROL_PADCONF_JTAG_NTRST[15:0]	jtag_ntrst	0x4800 2A1C	--	-----	0b1	0b000	0b01	---
CONTROL_PADCONF_JTAG_NTRST[31:16]	jtag_tck	0x4800 2A1C	--	-----	0b1	0b000	0b01	---
CONTROL_PADCONF_JTAG_TMS_TMSC [15:0]	jtag_tms_ tmsc	0x4800 2A20	--	-----	0b1	0b000	0b11	---
CONTROL_PADCONF_JTAG_TMS_TMSC [31:16]	jtag_tdi	0x4800 2A20	--	-----	0b1	0b000	0b11	---
CONTROL_PADCONF_JTAG_EMU0[15:0]	jtag_emu0	0x4800 2A24	0b00	0b--000	0b1	0b000	0b11	0b000
CONTROL_PADCONF_JTAG_EMU0[31:16]	jtag_emu1	0x4800 2A24	0b00	0b--000	0b1	0b000	0b11	0b000
CONTROL_PADCONF_SAD2D_SWAKEUP [15:0]	sad2d_ swakeup	0x4800 2A4C	--	-----	0b1	0b000	0b01	---
CONTROL_PADCONF_SAD2D_SWAKEUP [31:16]	jtag_rtck	0x4800 2A4C	--	0b--000	-	0b000	0b01	---
CONTROL_PADCONF_JTAG_TDO[15:0]	jtag_tdo	0x4800 2A50	--	0b--000	-	0b000	--	---

7.6.3.6 GENERAL_WKUP Register Description

Table 7-239. CONTROL_WKUP_CTRL

Address Offset	0x0000 0000		
Physical Address	0x4800 2A5C	**Instance**	GENERAL_WKUP
Description	USB TXEN polarity control and log modem warm reset source mux sel		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																										GPIO_1_IN_SEL_SAD2D_	RESERVED				MM_FSUSB3_TXEN_N_	MM_FSUSB2_TXEN_N_	MM_FSUSB1_TXEN_N_

Bits	Field Name	Description	Type	Reset
31:6	RESERVED	Reserved for future use.	R	0
5	GPIO_1_IN_SEL_SAD2D_ NRESWARM_IN_SEL	Mux select for GPIO1/SAD2D_NRESWARM bit.	RW	0
4:3	RESERVED	Reserved for future use.	R	0
2	MM_FSUSB3_TXEN_N_ OUT_POLARITY_CTRL	Polarity control for TXEN signal of multimode USB interface, port3. 0: Active low 1: Active high	RW	0
1	MM_FSUSB2_TXEN_N_ OUT_POLARITY_CTRL	Polarity control for TXEN signal of multimode USB interface, port2. 0: Active low 1: Active high	RW	0
0	MM_FSUSB1_TXEN_N_ OUT_POLARITY_CTRL	Polarity control for TXEN signal of multimodel USB interface, port1. 0: Active low 1: Active high	RW	0

Table 7-240. Register Call Summary for Register CONTROL_WKUP_CTRL

SCM Register Manual

- SCM Register Summary: [0]

Table 7-241. CONTROL_WKUP_DEBOBS_0

Address Offset	0x0000 000C		
Physical Address	0x4800 2A68	**Instance**	GENERAL_WKUP
Description	Select the WKUP domain set of signals to be observed for hw_dbg3, hw_dbg2, hw_dbg1, hw_dbg0		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED			OBSMUX3					RESERVED			OBSMUX2					RESERVED			OBSMUX1					RESERVED		OBSMUX0					

Bits	Field Name	Description	Type	Reset
31:29	RESERVED	Read returns reset value.	R	0x0
28:24	OBSMUX3	Select the set of signals to be observed for hw_dbg3	Refer to Table 7-243	0x00
23:21	RESERVED	Read returns reset value.	R	0x0
20:16	OBSMUX2	Select the set of signals to be observed for hw_dbg2	Refer to Table 7-243	0x00
15:13	RESERVED	Read returns reset value.	R	0x0
12:8	OBSMUX1	Select the set of signals to be observed for hw_dbg1	Refer to Table 7-243	0x00
7:5	RESERVED	Read returns reset value.	R	0x0
4:0	OBSMUX0	Select the set of signals to be observed for hw_dbg0	Refer to Table 7-243	0x00

Table 7-242. Register Call Summary for Register CONTROL_WKUP_DEBOBS_0

SCM Functional Description
- Description: [0]
- Observability Tables: [1] [2] [3] [4]

SCM Register Manual
- SCM Register Summary: [5]

Table 7-243. Type Value For CONTROL_WKUP_DEBOBS_0 Register

CONTROL_WKUP_DEBOBS_4[31] WKUPOBSERVABILITYDISABLE bit	0	1
OBSMUX3	RW	R
OBSMUX2	RW	R
OBSMUX1	RW	R
OBSMUX0	RW	R

Table 7-244. CONTROL_WKUP_DEBOBS_1

Address Offset	0x0000 0010		
Physical Address	0x4800 2A6C	**Instance**	GENERAL_WKUP
Description	Select the WKUP domain set of signals to be observed for hw_dbg7, hw_dbg6, hw_dbg5, hw_dbg4		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED			OBSMUX7					RESERVED			OBSMUX6					RESERVED			OBSMUX5					RESERVED			OBSMUX4				

Bits	Field Name	Description	Type	Reset
31:29	RESERVED	Read returns reset value.	R	0x0
28:24	OBSMUX7	Select the set of signals to be observed for hw_dbg7	Refer to Table 7-246	0x00
23:21	RESERVED	Read returns reset value.	R	0x0
20:16	OBSMUX6	Select the set of signals to be observed for hw_dbg6	Refer to Table 7-246	0x00
15:13	RESERVED	Read returns reset value.	R	0x0

Bits	Field Name	Description	Type	Reset
12:8	OBSMUX5	Select the set of signals to be observed for hw_dbg5	Refer to Table 7-246	0x00
7:5	RESERVED	Read returns reset value.	R	0x0
4:0	OBSMUX4	Select the set of signals to be observed for hw_dbg4	Refer to Table 7-246	0x00

Table 7-245. Register Call Summary for Register CONTROL_WKUP_DEBOBS_1

SCM Functional Description
- Description: [0]
- Observability Tables: [1] [2] [3] [4]

SCM Register Manual
- SCM Register Summary: [5]

Table 7-246. Type Value For CONTROL_WKUP_DEBOBS_1 Register

CONTROL_WKUP_DEBOBS_4[31] WKUPOBSERVABILITYDISABLE bit	0	1
OBSMUX7	RW	R
OBSMUX6	RW	R
OBSMUX5	RW	R
OBSMUX4	RW	R

Table 7-247. CONTROL_WKUP_DEBOBS_2

Address Offset	0x0000 0014		
Physical Address	0x4800 2A70	**Instance**	GENERAL_WKUP
Description	Select the WKUP domain set of signals to be observed for hw_dbg11 hw_dbg10, hw_dbg9, hw_dbg8		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED			OBSMUX11					RESERVED			OBSMUX10					RESERVED			OBSMUX9					RESERVED			OBSMUX8				

Bits	Field Name	Description	Type	Reset
31:29	RESERVED	Read returns reset value.	R	0x0
28:24	OBSMUX11	Select the set of signals to be observed for hw_dbg11	Refer to Table 7-249	0x00
23:21	RESERVED	Read returns reset value.	R	0x0
20:16	OBSMUX10	Select the set of signals to be observed for hw_dbg10	Refer to Table 7-249	0x00
15:13	RESERVED	Read returns reset value.	R	0x0
12:8	OBSMUX9	Select the set of signals to be observed for hw_dbg9	Refer to Table 7-249	0x00
7:5	RESERVED	Read returns reset value.	R	0x0
4:0	OBSMUX8	Select the set of signals to be observed for hw_dbg8	Refer to Table 7-249	0x00

Table 7-248. Register Call Summary for Register CONTROL_WKUP_DEBOBS_2

SCM Functional Description
- Description: [0]
- Observability Tables: [1] [2] [3] [4]

SCM Register Manual
- SCM Register Summary: [5]

Table 7-249. Type Value For CONTROL_WKUP_DEBOBS_2 Register

CONTROL_WKUP_DEBOBS_4[31] WKUPOBSERVABILITYDISABLE bit	0	1
OBSMUX11	RW	R
OBSMUX10	RW	R
OBSMUX9	RW	R
OBSMUX8	RW	R

Table 7-250. CONTROL_WKUP_DEBOBS_3

Address Offset	0x0000 0018		
Physical Address	0x4800 2A74	**Instance**	GENERAL_WKUP
Description	Select the WKUP domain set of signals to be observed for hw_dbg15 hw_dbg14, hw_dbg13, hw_dbg12		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED			OBSMUX15					RESERVED			OBSMUX14					RESERVED			OBSMUX13					RESERVED			OBSMUX12				

Bits	Field Name	Description	Type	Reset
31:29	RESERVED	Read returns reset value.	R	0x0
28:24	OBSMUX15	Select the set of signals to be observed for hw_dbg15	Refer to Table 7-252	0x00
23:21	RESERVED	Read returns reset value.	R	0x0
20:16	OBSMUX14	Select the set of signals to be observed for hw_dbg14	Refer to Table 7-252	0x00
15:13	RESERVED	Read returns reset value.	R	0x0
12:8	OBSMUX13	Select the set of signals to be observed for hw_dbg13	Refer to Table 7-252	0x00
7:5	RESERVED	Read returns reset value.	R	0x0
4:0	OBSMUX12	Select the set of signals to be observed for hw_dbg12	Refer to Table 7-252	0x00

Table 7-251. Register Call Summary for Register CONTROL_WKUP_DEBOBS_3

SCM Functional Description
- Description: [0]
- Observability Tables: [1] [2] [3] [4]

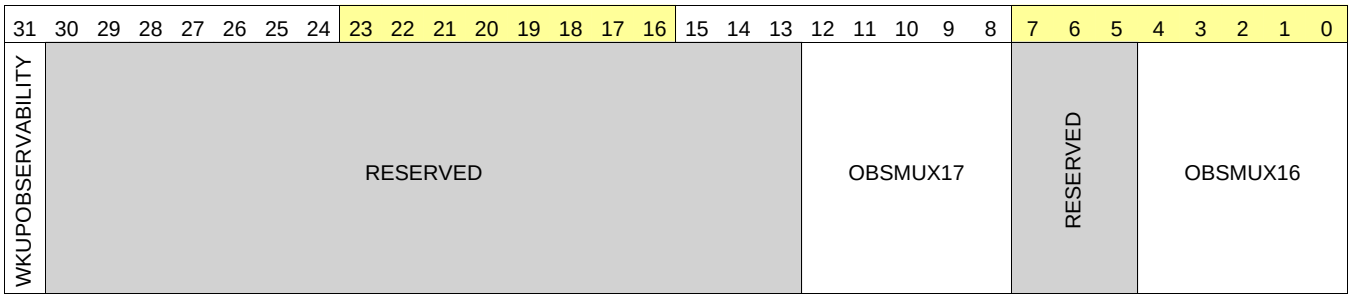
SCM Register Manual
- SCM Register Summary: [5]

Table 7-252. Type Value For CONTROL_WKUP_DEBOBS_3 Register

CONTROL_WKUP_DEBOBS_4[31] WKUPOBSERVABILITYDISABLE bit	0	1
OBSMUX15	RW	R
OBSMUX14	RW	R
OBSMUX13	RW	R
OBSMUX12	RW	R

Table 7-253. CONTROL_WKUP_DEBOBS_4

Address Offset	0x0000 001C		
Physical Address	0x4800 2A78	**Instance**	GENERAL_WKUP
Description	Select the WKUP domain set of signals to be observed for hw_dbg17, hw_dbg16		
Type	RW		



Bits	Field Name	Description	Type	Reset
31	WKUPOBSERVABILITY DISABLE	Control the observability feature 0x0: Observability can be configured through the ObsMux bit field in CONTROL_DEBOBS register 0x1: Observability is disabled. If pads are configured for the 'hardware debug', output is tied low	Refer to Table 7-255	0x0
30:13	RESERVED	Read returns reset value.	R	0x00000
12:8	OBSMUX17	Select the set of signals to be observed for hw_dbg17	Refer to Table 7-255	0x00
7:5	RESERVED	Read returns reset value.	R	0x0
4:0	OBSMUX16	Select the set of signals to be observed for hw_dbg16	Refer to Table 7-255	0x00

Table 7-254. Register Call Summary for Register CONTROL_WKUP_DEBOBS_4

SCM Functional Description

- Description: [0] [1]
- Observability Tables: [2] [3]

SCM Register Manual

- SCM Register Summary: [4]
- GENERAL_WKUP Register Description: [5] [6] [7] [8] [9]

Table 7-255. Type Value For CONTROL_WKUP_DEBOBS_4 Register

CONTROL_WKUP_DEBOBS_4[31] WKUPOBSERVABILITYDISABLE bit	0	1
WKUPOBSERVABILITYDISABLE	R/OCO	R
OBSMUX17	RW	R
OBSMUX16	RW	R

Memory Management Units

This chapter describes the memory management units (MMUs) for the OMAP34xx multimedia device.

Topic **Page**

8.1 MMU Overview 932
8.2 MMU Integration 933
8.3 MMU Functional Description 937
8.4 MMU Basic Programming Model 950
8.5 MMU Register Manual 957

8.1 MMU Overview

The OMAP34xx device contains three memory management units (MMUs):

- Microprocessor unit (MPU) MMU
- Camera MMU
- Image Video and Audio accelerator (IVA2.2) MMU

The camera MMU and IVA2.2 MMU share the same architecture and are both described in this chapter. The MPU MMU, which implements a different architecture, is covered in Chapter 3, *MPU Subsystem*.

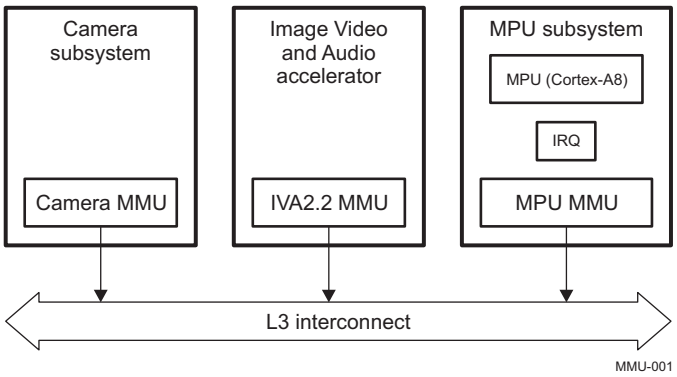
Note: The MMUn prefix provides information about the register instantiation, where n = 1 for the camera MMU, and n = 2 for the IVA2.2 MMU.

The MMU instances include the following main features:

- N entries fully associative translation look-aside buffer (TLB) with N = 8 for the camera MMU and N = 32 for the IVA2.2 MMU
- 1 interrupt line out to the MPU subsystem
- 32-bit virtual addresses, 32-bit physical address
- Mapping size: 4KB and 64KB pages, 1MB section, and 16MB supersection
- Predefined (static) or table-driven (hardware table walker) software translation strategies

Figure 8-1 shows the MMU instances in the OMAP34xx device.

Figure 8-1. OMAP34xx MMU Instances



8.2 MMU Integration

The MMU communicates accesses from the requestor (either the the camera subsystem or the IVA2.2) to the L3 main interconnect, performing virtual to physical address translation. The camera MMU is programmed through the L4-core interconnect. The IVA2.2 MMU is programmed through the L3 interconnect. Both MMU error conditions are signaled as interrupts to the system master processor (that is, the MPU).

Figure 8-2 and Figure 8-3 show the system integration of the camera MMU instance and the IVA2.2 MMU instance.

Figure 8-2. Camera MMU System Integration

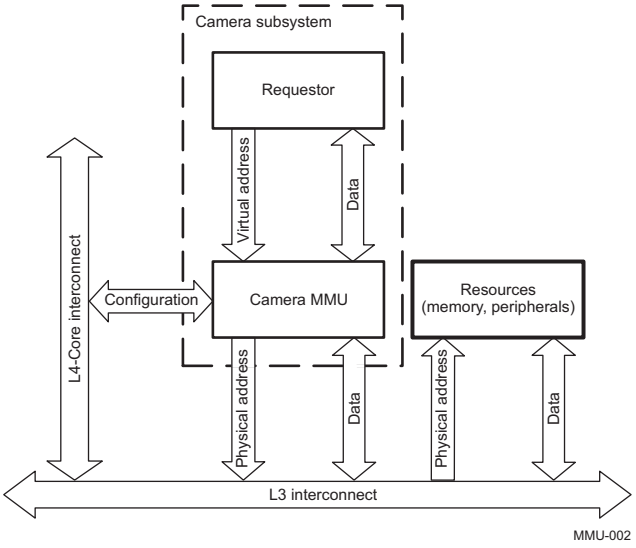
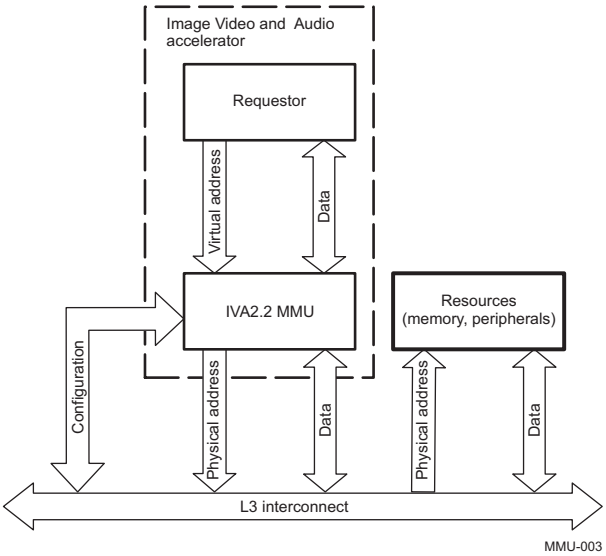


Figure 8-3. IVA2.2 MMU System Integration



8.2.1 Clock Domains

The camera MMU instance has two clock domains: the functional clock domain for the MMU, which is synchronous to the L3 interconnect clock, and the L4 interconnect clock, which is used to configure the MMU instance. Both camera MMU clocks are derived from a common reference clock generated by the power reset and clock management (PRCM) module.

The IVA2.2 MMU instance has only one clock domain: the functional clock domain for the MMU. The IVA MMU clock is generated by the DPLL2 embedded in IVA2.2, but controlled by PRCM registers.

8.2.2 Power Management

The OMAP34xx functional units are grouped into power domains. Each power domain is a section of the device with independent and dedicated power rails. Fifteen different power domains exist. For information about OMAP34xx power management, see Chapter 4, *Power, Reset, and Clock Management.*

The MMU instances belong to different power domains. Table 8-1 shows the correspondence between the MMU instance and power domains.

Table 8-1. Power Domains of the MMU Instances

MMU Instance	Power Domain
MMU1 (camera MMU)	CAM
MMU2 (IVA2.2 MMU)	IVA2

8.2.2.1 System Power Management

As part of the OMAP34xx system-wide power management scheme, each MMU instance supports a communication protocol with the PRCM module that allows the PRCM module to request an MMU instance to enter a low-power state. When the MMU instance acknowledges a low-power mode request from the PRCM module, the clock to the instance is gated off at the PRCM clock generator. Because the clock is disabled at the source, the low-power mode offers lower power consumption than the internal clock gating method in the local power management.

The MMU instance can be configured through the MMUn.MMU_SYSCONFIG[4:3] IDLEMODE field as one of the following acknowledgement modes:

- No-idle mode: The MMU instance never enters the idle state.
- Force-idle mode: The MMU instance immediately enters the idle state after receiving a low-power mode request from the PRCM module. In this mode, the software must ensure that there are no pending interrupts before requesting this mode to go into the idle state; otherwise, an error can occur.
- Smart-idle mode: After receiving a low-power mode request from the PRCM module, the MMU instance enters the idle state only after all interrupts are acknowledged.

Table 8-2 describes the MMU power management modes. For details, see the MMU_SYSCONFIG register.

Table 8-2. Power Domains of the MMU Instances

Power Management Mode Requested by the PRCM	MMUn.MMU_SYSCONFIG[4:3] IDLEMODE field
Force-idle	00
No-idle	01
Smart-idle	10
Reserved	11

8.2.2.2 Module Power Saving

To conserve power, the MMU instance supports an automatic idle mode whenever activity is not detected on the configuration register port of the MMU instance. The automatic idle mode is enabled or disabled through the MMUn.MMU_SYSCONFIG[0] AUTOIDLE bit.

If the MMUn.MMU_SYSCONFIG[0] AUTOIDLE bit is asserted, the automatic idle mode is enabled when activity is not detected on the configuration register port, and the MMU instance clock is disabled internally to the module, thereby reducing power consumption.

When new activity is detected on the configuration register port, the clock is restarted with no latency penalty. After reset, the automatic idle mode is disabled; therefore, it is recommended to enable the automatic idle mode to reduce power consumption.

8.2.3 Reset

The MMU instances are reset together with their respective reset domains. Table 8-3 shows the correspondence between the MMU instances and the reset domains.

Table 8-3. Reset Domains of the MMU Instances

MMU Instance	Reset Domain
MMU1 (camera MMU)	CAM_RST
MMU2 (IVA2.2 MMU)	IVA_RST2

Software reset is applied when the MMUn.MMU_SYSCONFIG[1] SOFTRESET is set to 1. The MMUn.MMU_SYSSTATUS[0] RESETDONE bit can be polled to know the reset status.

When an MMU instance is released from reset, its TLB is empty and the MMU is disabled.

Note: IVA2.2 MMU can be accessed from the L3 interconnect (configuration registers) even if the DSP is still under reset (IVA_RST1 active).

8.2.4 Interrupts

Each MMU instance can generate an interrupt to the MPU on the occurrence of the following predefined set of events:

- Multi-hit fault
  There is more than one TLB entry for the given virtual address.
- Table walk fault
  A table walk data read generated an error.
- Emulation miss
  A TLB miss was caused by an emulation access.
- Translation fault
  No translation is found for the given virtual address. The hardware table walker is enabled but no valid page table entry exists for the requested address.
- TLB miss with table walk disabled
  No translation is found in the TLB for the given virtual address and the table walking logic is disabled.

Each of these events can be individually enabled and disabled using the MMUn.MMU_IRQENABLE register. If an event occurs and is enabled, an interrupt is generated to the MPU. The MPU can use the MMUn.MMU_IRQSTATUS register to find the precise cause of the interrupt.

The MMUn.MMU_FAULT_AD register holds the virtual address of the translation that causes the interrupt. The MMUn.MMU_EMU_FAULT_AD indicates the address of the last emulation event causing an MMU interrupt.

Table 8-4 shows the generated interrupt versus the MMU instance.

Table 8-4. Interrupts of the MMU Instances

MMU Instance	MMU Interrupt Name	MMU Interrupt Mapping
MMU1 (camera MMU)	CAM_IRQ0	M_IRQ_24
MMU2 (IVA2.2 MMU)	IVA2_MMU_IRQ	M_IRQ_28

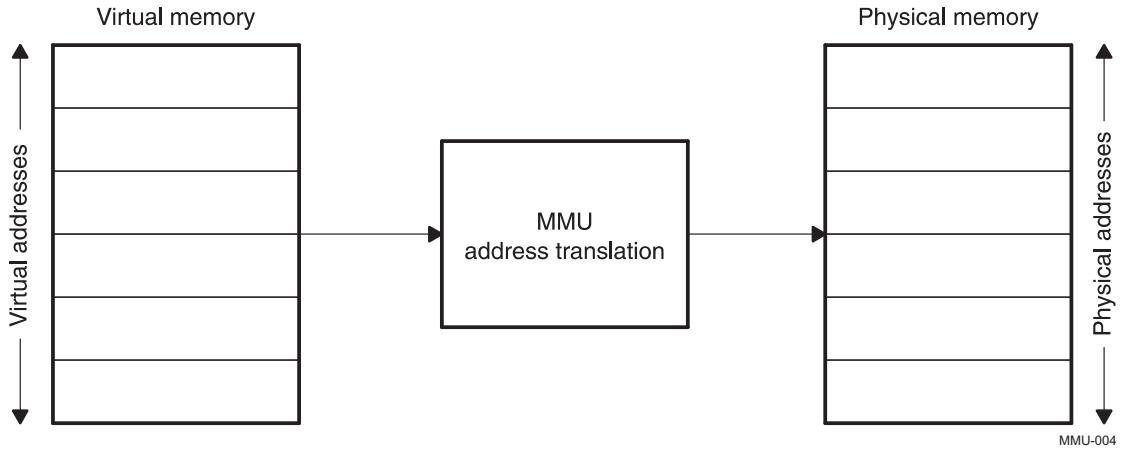
Note: The IVA2_MMU_IRQ (MMU2 instance) interrupt is a dedicated interrupt to the MPU subsystem.
Chapter 12, *Camera Subsystem*, and Chapter 14, *IVA2.2 Subsystem*, outline details about interrupt generation in the camera and IVA2.2 subsystems. For details about the MPU interrupt scheme, see Chapter 10, *Interrupt Controller*.

8.3 MMU Functional Description

The MMUs handle the translation from virtual into physical addresses. The requestor (either the camera subsystem or the DSP mega cell or the EDMA module for IVA2.2) issues virtual addresses to the respective MMU (MMU1 for the camera subsystem and MMU2 for the IVA2.2). The MMU translates these virtual addresses into physical addresses to access the actual resource (memory) when the MMU instance is enable. That is when MMUn.MMU_CNTL[1] MMUENABLE is set to 1.

Figure 8-4 shows the relationship between the physical address, the virtual address, and the MMU.

Figure 8-4. MMU Address Translation



8.3.1 MMU Benefits

The MMU offers two major benefits:

- Memory defragmentation: Fragmented physical memory can be translated into contiguous virtual memory without moving data.
- Memory protection: Illegal, that is, non-allowed accesses to memory locations can be detected and prevented.

Figure 8-5 shows two typical MMU use cases.

Figure 8-5. MMU Usage Examples

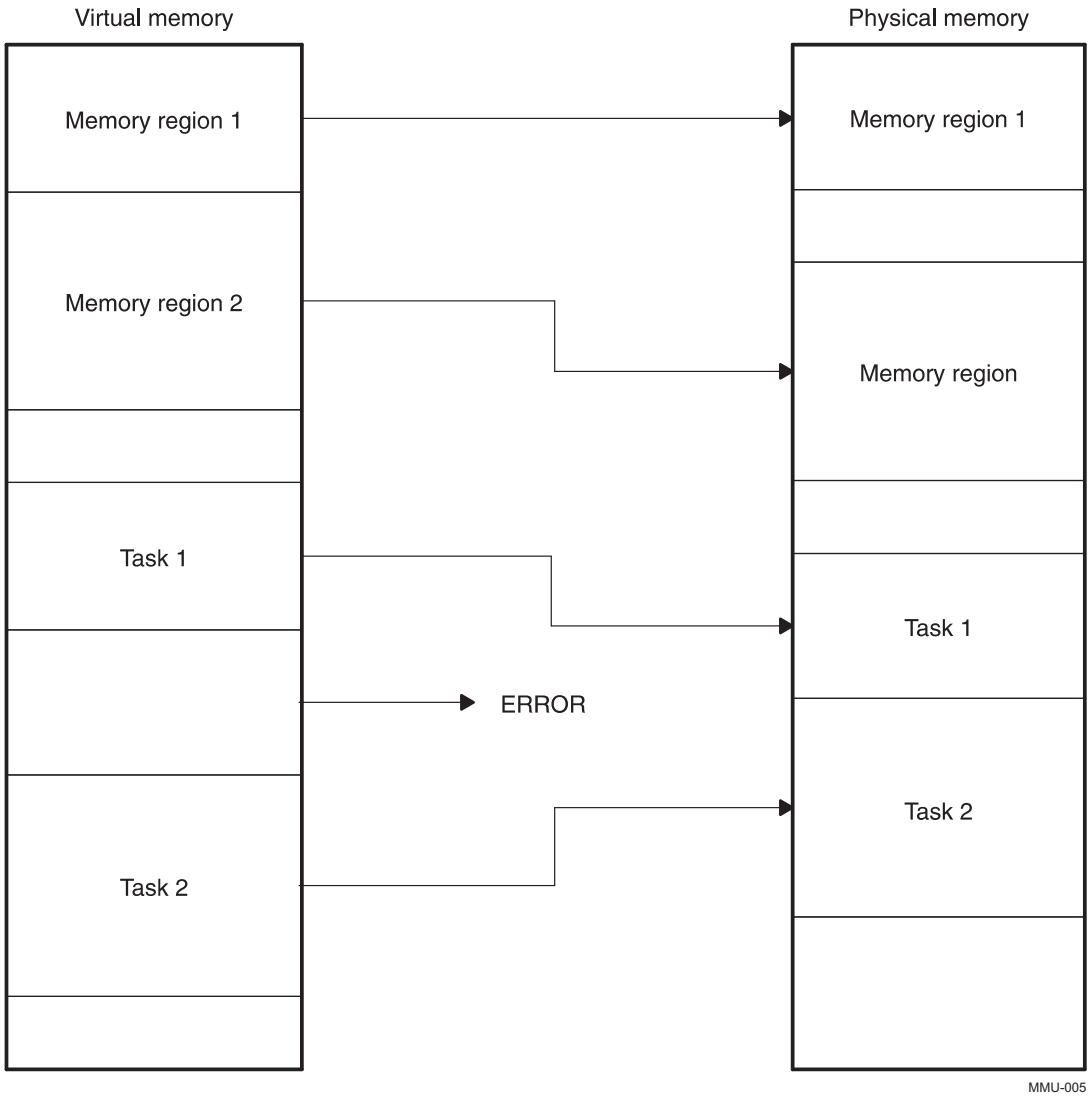


Figure 8-5 shows two benefits of using an MMU. Memory region 1 and memory region 2 are fragmented in physical memory. Using the MMU, they appear as one contiguous memory region in the virtual memory space.

On the other hand, task 1 and task 2 are located adjacent to each other in physical memory. In systems without an MMU, there is a danger that task 1 can accidentally write into the memory area allocated to task 2 and vice versa. Allocating task 1 and task 2 into two separate virtual memory regions prevents this problem, because a region of unmapped memory separates the two tasks. Any erroneous access to this region results in an error that can be detected easily.

8.3.2 MMU Architecture

The MMU translation process is based on translation entries stored in translation tables. One first-level translation table can exist with several optional second-level translation tables.

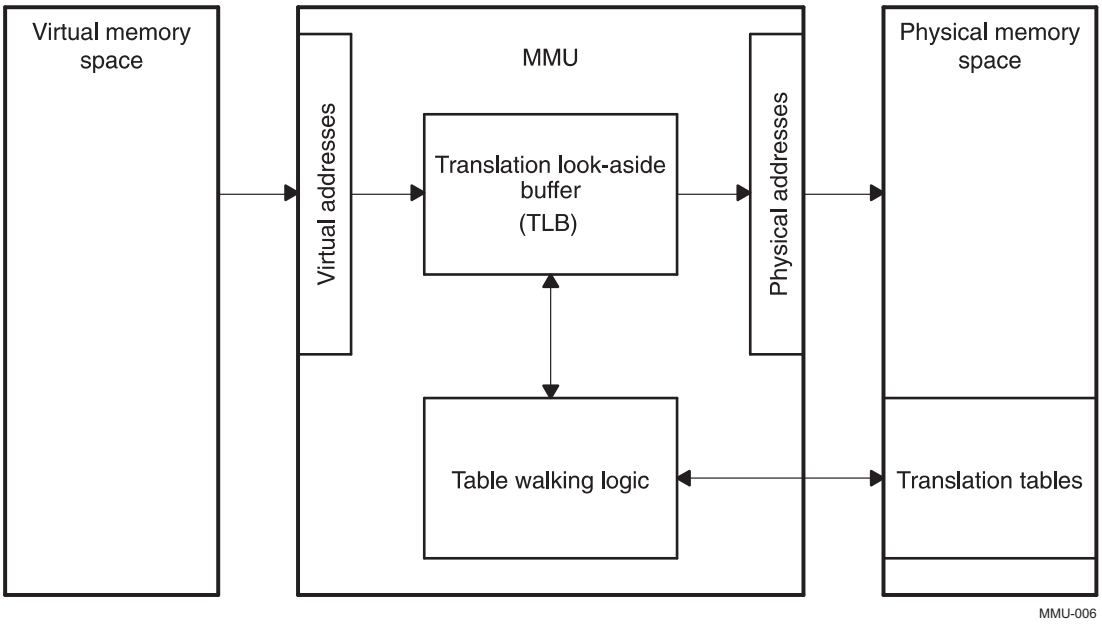
Each table entry describes the translation of one contiguous memory region. For a description of the structure of these tables, see Section 8.3.3, *Translation Tables*.

Two major functional units exist in the MMU to provide address translation automatically based on the table entries:

- The table walker automatically retrieves the correct translation table entry for a requested translation. If two-level translation is used (for the translation of small memory pages), the table walker also automatically reads the required second-level translation table entry. Two-level translation is described in Section 8.3.3.3,*Two-Level Translation*.
- The TLB stores recently used translation entries, acting like a cache of the translation table.

This basic architecture is outlined in Figure 8-6.

Figure 8-6. MMU Architecture

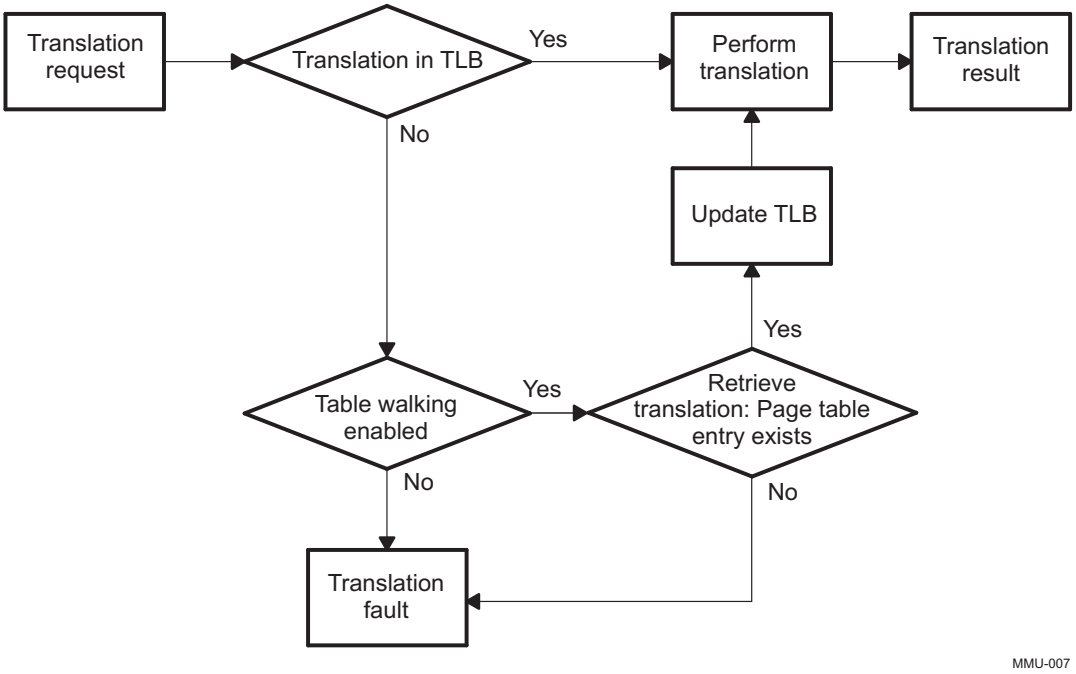


8.3.2.1 MMU Address Translation Process

Whenever an address translation is requested (that is, for every access with the MMU enabled), the MMU first checks whether the translation is already contained in the TLB, which acts like a cache storing recent translations. The TLB can also be programmed manually to ensure that time-critical data can be translated without delay.

If the requested translation is not in the TLB, the table-walking logic retrieves this translation from the translation table(s), and then updates the TLB. The address translation is then performed. Figure 8-7 summarizes the process.

Figure 8-7. Translation Process



8.3.3 Translation Tables

The translation of virtual to physical addresses is based on entries in translation tables that define the following properties:

- Address translation, that is, the correspondence between virtual and physical addresses
- Size of the memory region the entry translates
- Endianness, data access size, and the mixed property of this memory region

The virtual addresses index the translation tables. Each virtual address corresponds to exactly one entry in the translation table.

8.3.3.1 Translation Table Hierarchy

When developing a table-based address translation scheme, one of the most important design parameters is the memory page size described by each translation table entry. MMU instances support 4KB and 64KB pages, a 1MB section, and a 16MB supersection. Using bigger page sizes means a smaller translation table.

Using a smaller page size greatly increases the efficiency of dynamic memory allocation and defragmentation. That is why many operating systems (OSs) can operate on memory blocks as small as 4KB; however, the smaller size implies a more complex table structure.

A quick calculation shows that using 4KB memory pages with one translation table would require one million entries to span the entire 4GB address range. The table itself would be 32MB, a size that is not feasible.

However, using bigger pages greatly reduces the functionality of the OS memory management. Implementing a two-level hierarchy reconciles these two requirements. Within this hierarchy, one first-level translation table describes the translation properties based on 1MB memory regions.

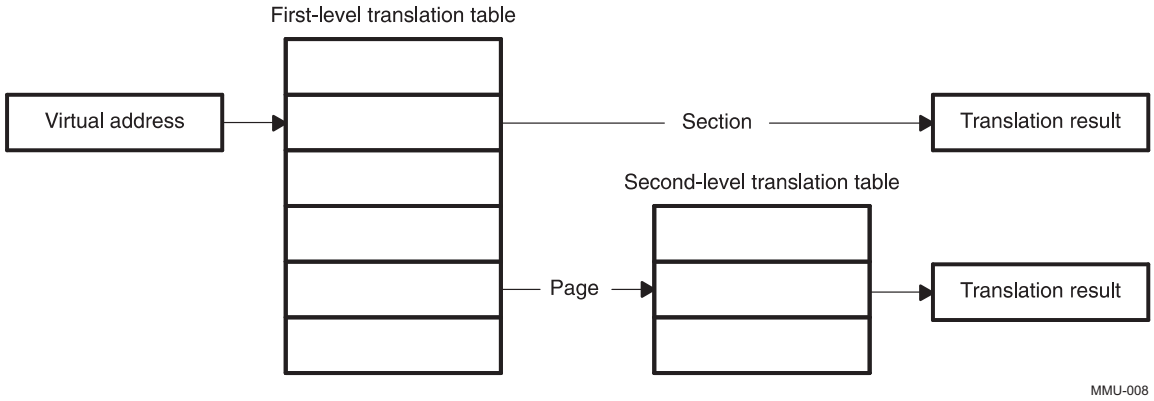
Each of the entries in this first-level translation table can specify the following:

- The translation properties for a big memory section. This memory section can be either 1MB (section) or 16MB (supersection). In this case, all translation parameters are specified in the first-level translation table entry.

- A pointer to a second-level translation table that specifies individual translation properties based on smaller pages within the 1MB page of memory. These pages can be either 64KB (large page) or 4KB (small page). In this case, the actual translation parameters are specified in the second-level translation table entry. The first-level translation table entry specifies only the base address of the second-level translation table.

This hierarchical approach means that additional translation information for smaller pages must be provided only when the pages are actually used. Figure 8-8 shows this hierarchy.

Figure 8-8. Translation Hierarchy



The structure of the first and second-level translation tables and their entries are described in more detail in Section 8.3.3.2, *First-Level Translation Table*, and Section 8.3.3.3, *Two-Level Translation*.

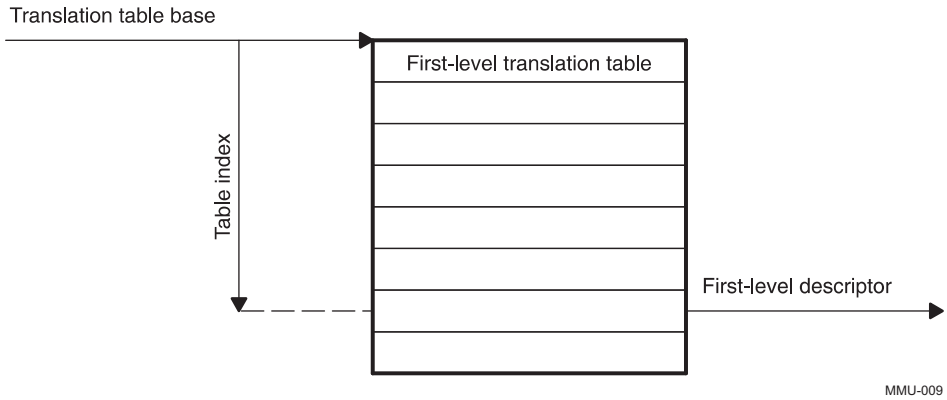
8.3.3.2 First-Level Translation Table

The first-level translation table describes the translation properties for 1MB sections. To describe a 4GB address range requires 4096 32-bit entries (so-called first-level descriptors).

The first-level translation table start address must be aligned on a multiple of the table size with a 128-byte minimum. Consequently, an alignment of at least 16K bytes is required for a complete 4096-entry table; that is, at least the last fourteen address bits must be zero.

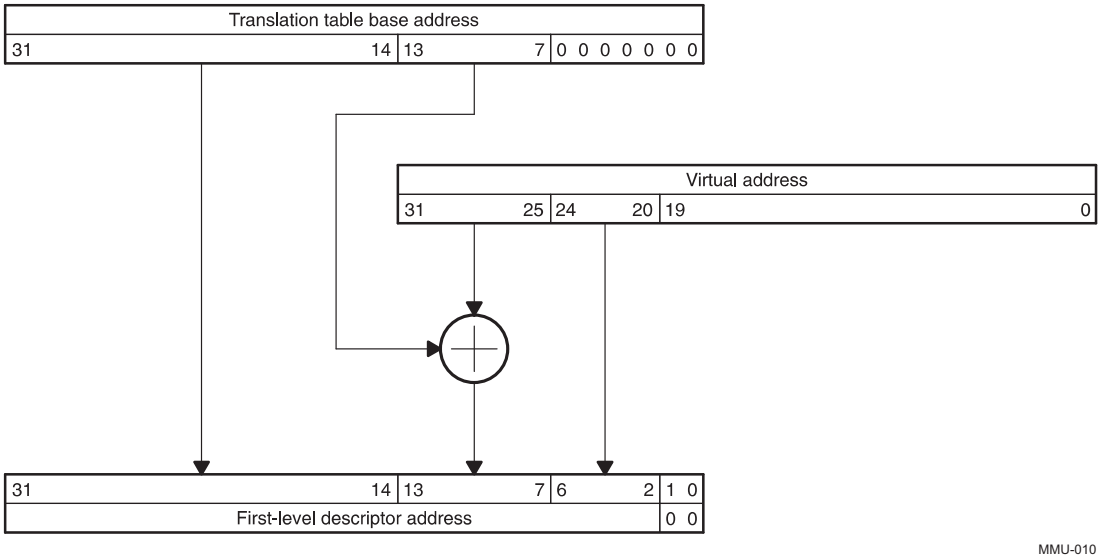
The start address of the first-level translation table is specified by the so-called translation table base. The table is indexed by the upper 12-bits of the virtual address. This mechanism is shown in Figure 8-9.

Figure 8-9. First-Level Descriptor Address Calculation



To summarize, the translation table base and the translation table index together define the first-level descriptor address. Figure 8-10 outlines the precise mechanism used to calculate this address.

Figure 8-10. Detailed First-Level Descriptor Address Calculation



As an example of this mechanism, consider a translation table base address of 0x8000:0000 and a virtual address of 0x1234:5678. In this case, the first-level descriptor address is 0x8000:0000 + (0x123 <<2) = 0x8000:048C.

8.3.3.2.1 First-Level Descriptor Format

Each first-level descriptor provides either the complete address translation for 1MB or 16MB sections or provides a pointer to a second-level translation table for 4KB or 64KB pages. The first-level descriptor format is shown in Table 8-5.

Table 8-5. First-Level Descriptor Format

First-Level Descriptor Format												
31:24	23:20	19	18	17	16	15	14:12	11:10	9:2	1	0	
X										0	0	Fault
Second-Level Translation Table Base Address									X	0	1	Page
Section Base Address		X	0	M	X	E (1)	X	ES	X	1	0	Section
Supersection Base Address	X		1	M	X	E	X	ES	X	1	0	Supersection
X										1	1	Fault

(1) See Table 8-34 for endianness limitations.

M = Mixed region: 0 = Page-based endianness, 1 = Access-based endianness

E = Endianness: 0 = Little endian, 1 = Big endian (endianness is locked on little endian)

ES = Element Size: 00 = 8-bit, 01= 16-bit, 10 = 32-bit, 11 = No endianness conversion

X = Don't care

8.3.3.2.2 First-Level Page Descriptor Format

If a translation granularity smaller than 1MB is required, a two-level translation process is used. In this case, the first-level block descriptor specifies only the start address of a second-level translation table. The second-level translation table entries specify the actual translation properties.

8.3.3.2.3 First-Level Section Descriptor Format

Each section descriptor in the first-level translation table specifies the complete translation properties for a 1MB section or a 16MB supersection.

Note: Supersection descriptors must be repeated 16 times, because each descriptor in the first-level translation table describes 1MB of memory. If an access points to a descriptor which is not initialized, MMU will behave in an unpredictable way.

In addition to the address translation itself, three parameters are specified in the section descriptors:

- Endianness

  The *endianness* parameter specifies whether the memory section uses a big- or little-endian data format. This parameter is locked to little endian. See Table 8-34 for more information.
- Element size

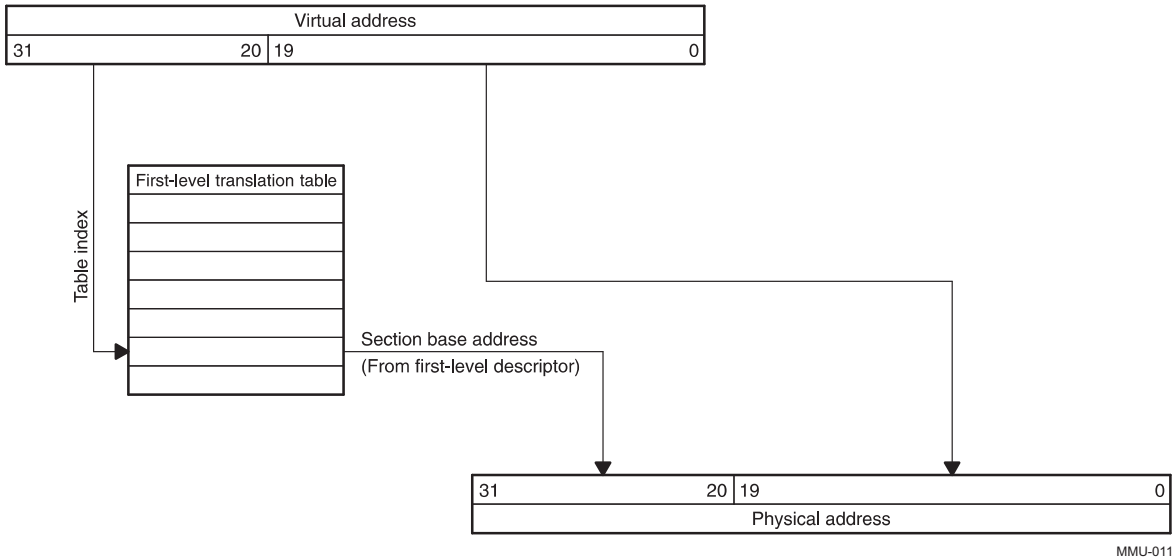
  The *element size* parameter can optionally specify the data access size (8, 16, or 32 bits) for all data items in the defined section.
- Mixed region

  The *mixed region* parameter specifies whether the information about the data access size is detected from the access itself (access-based detection) or if the specified element size parameter is used (page-based detection). For example, the specified element size parameter can be used when several smaller sized accesses are packed into a bigger sized access, such as two 16-bit accesses packed into one 32-bit access. In this case, with no specified data access size, 32 bits would be the access size detected, leading to an incorrect result. To avoid this problem, specify the data access size for the memory section.

8.3.3.2.4 Section Translation Summary

Sections and supersections can be translated based solely on the information in the first-level translation table. Figure 8-11 summarizes the address translation process for a section.

Figure 8-11. Section Translation Summary

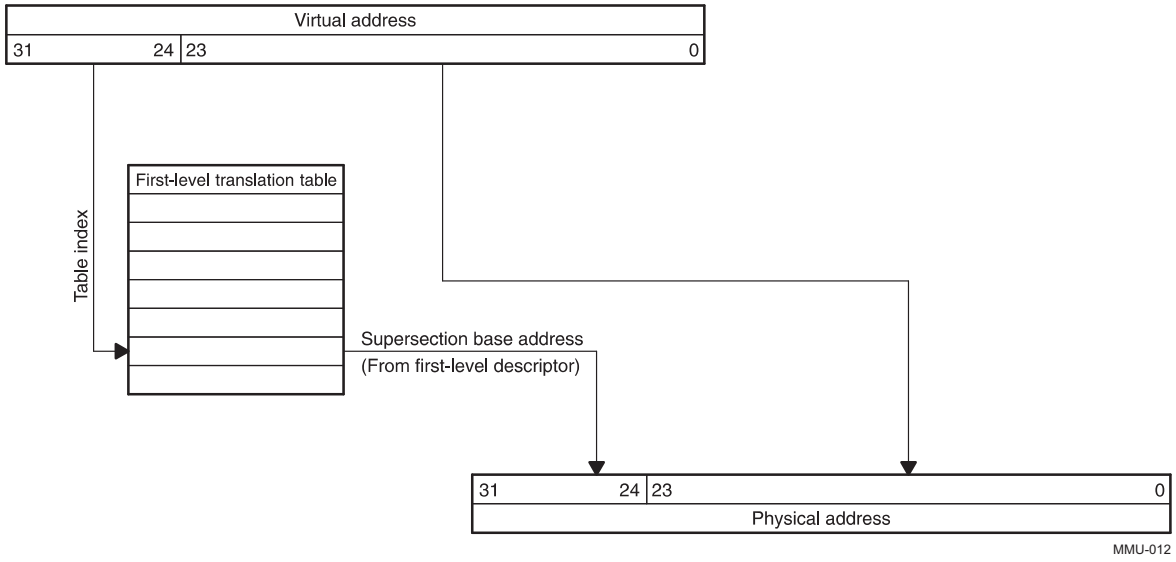


8.3.3.2.5 Supersection Translation Summary

The translation of a supersection is similar to the translation of a section. The difference is that for a supersection only bits 31 to 24 index into the first-level translation table. The last four bits of the table index are implicitly assumed to be zero as there are 16 identical consecutive entries for a supersection.

Figure 8-12 shows the translation mechanism for a supersection.

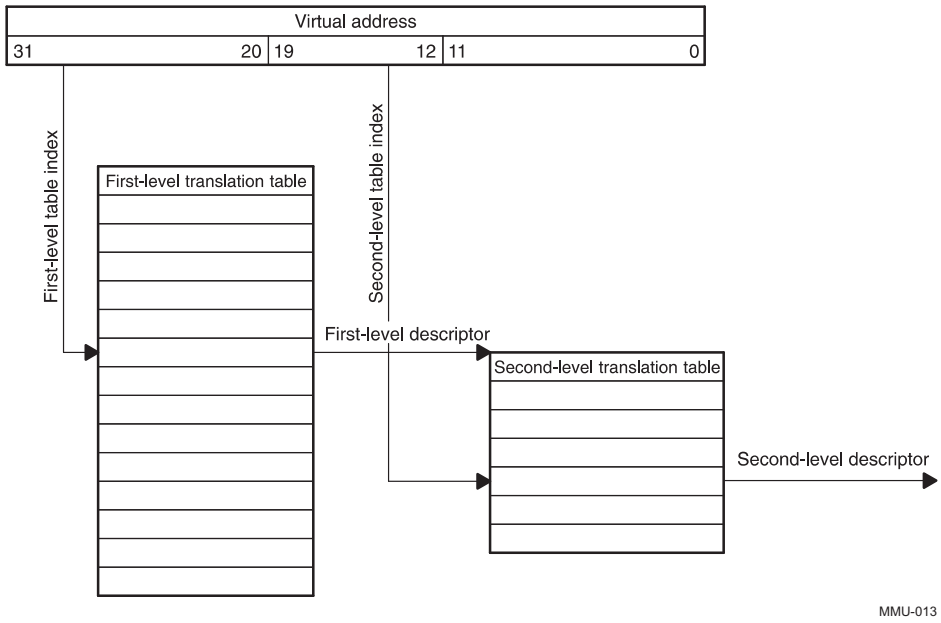
Figure 8-12. Supersection Translation Summary



8.3.3.3 Two-Level Translation

Two-level translation is used when fine-grain granularity is required, that is when memory sections smaller than 1MB are needed. In this case, the first-level descriptor provides a pointer to the base address of a second-level translation table. This second-level table is indexed by bits 19 to 12 of the virtual address. Figure 8-13 shows this indexing mechanism.

Figure 8-13. Two-Level Translation



Each second-level translation table describes the translation of 1MB of address space in pages of 64KB (large page) or 4KB (small page). It consists of 256 second-level descriptors describing 4KB each.

Note: In the case of a large page, the same descriptor must be repeated 16 times. If an access points to a descriptor which is not initialized, MMU will behave in an unpredictable way.

8.3.3.3.1 Second-Level Descriptor Format

Similar to first-level section descriptors, second-level descriptors provide all of the necessary information for the translation of a large or small page. Table 8-6 shows the format of second-level descriptors. The translation parameters (endianness, element size, and mixed region) have the same meaning as those for sections.

Table 8-6. Second-Level Descriptor Format

Second-Level Descriptor Format										
31:16	15:12	11	10	9	8:6	5:4	3:2	1	0	
X								0	0	Fault
Large Page Base Address	X	M	X	E (1)	X	ES	X	0	1	Large Page
Small Page Base Address		M	X	E	X	ES	X	1	X	Small Page

(1) See Table 8-34 for endianness limitations.

M = Mixed region: 0 = Page-based endianness, 1 = Access-based endianness

E = Endianness: 0 = Little-endian, 1 = Big-endian (endianness is locked on little endian)

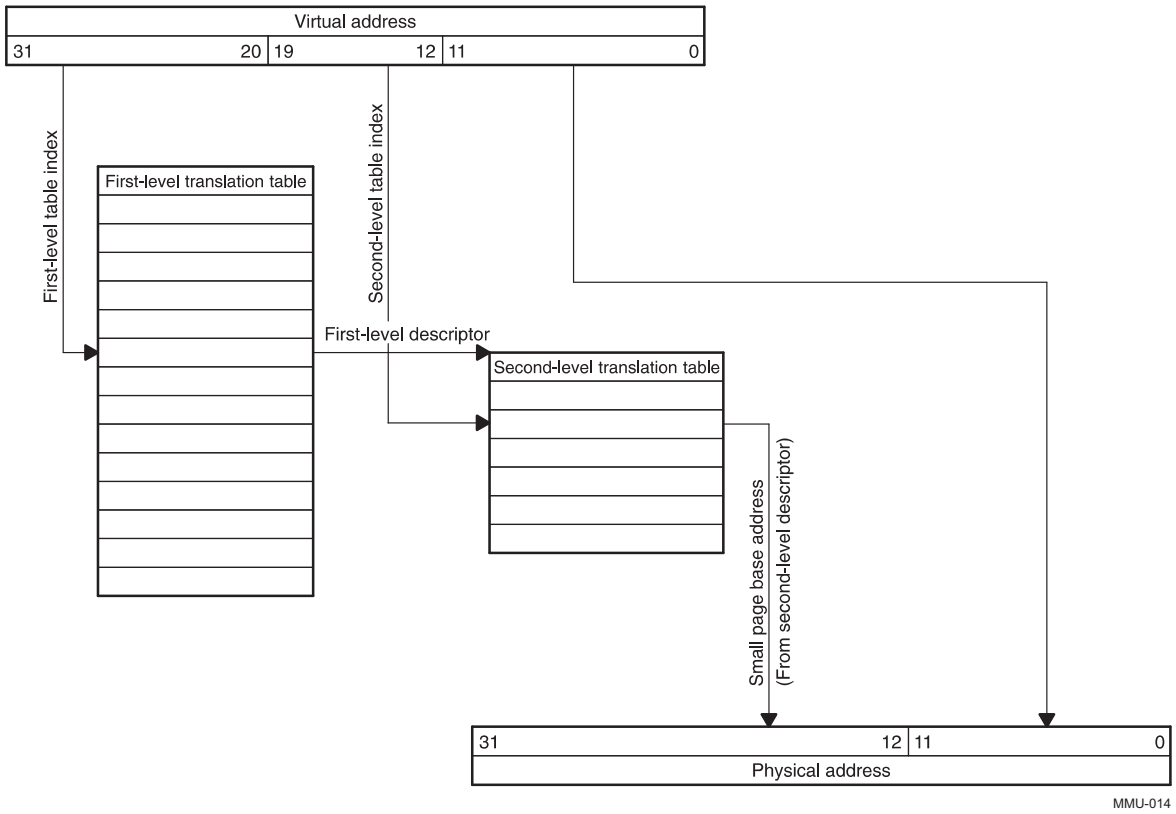
ES = Element Size: 00 = 8-bit, 01= 16-bit, 10 = 32-bit, 11 = No endianness conversion

X = Don't care

8.3.3.3.2 Small Page Translation Summary

Figure 8-14 summarizes the translation process for small pages.

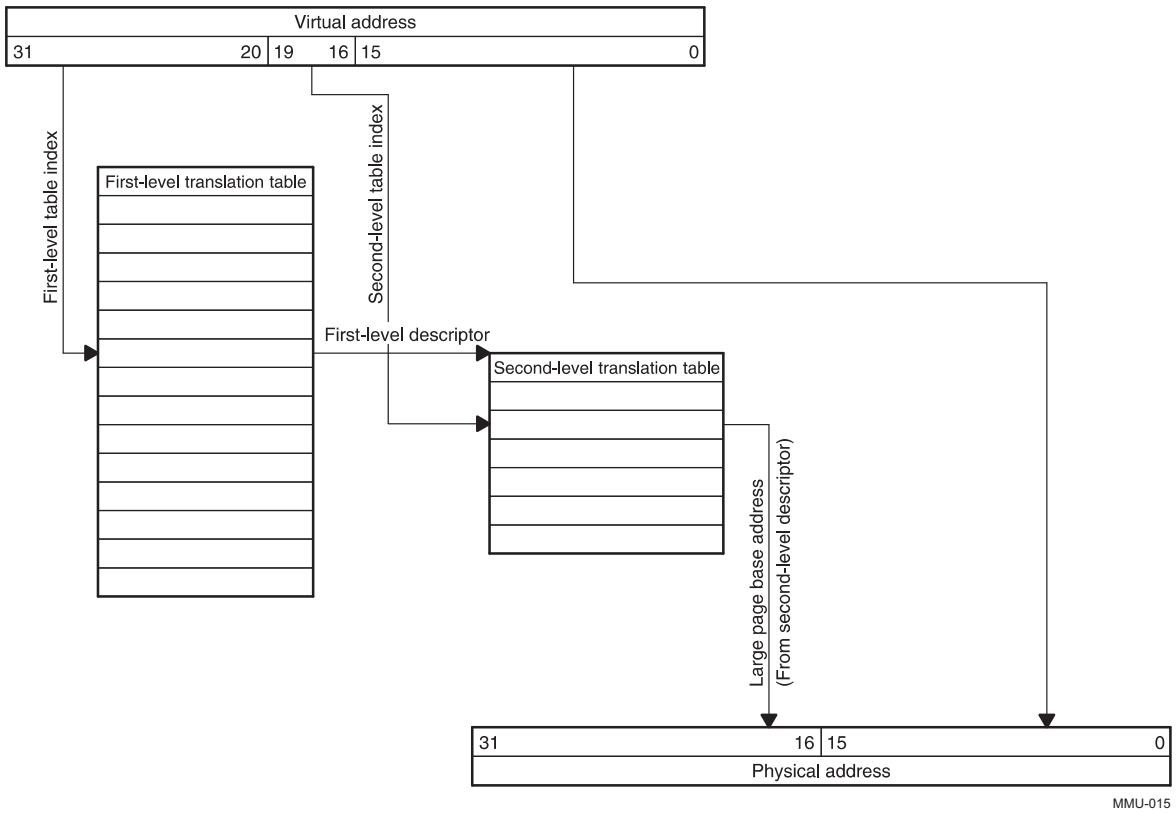
Figure 8-14. Small Page Translation Summary



8.3.3.3.3 Large Page Translation Summary

The translation of a large page is similar to the translation of a small page. The difference is that, for a large page, only bits 19 to 16 index into the second-level translation table. The last four bits of the table index are implicitly assumed to be zero as there are 16 identical consecutive entries for a large page. This is shown in Figure 8-15.

Figure 8-15. Large Page Translation Summary



8.3.4 Translation Lookaside Buffer

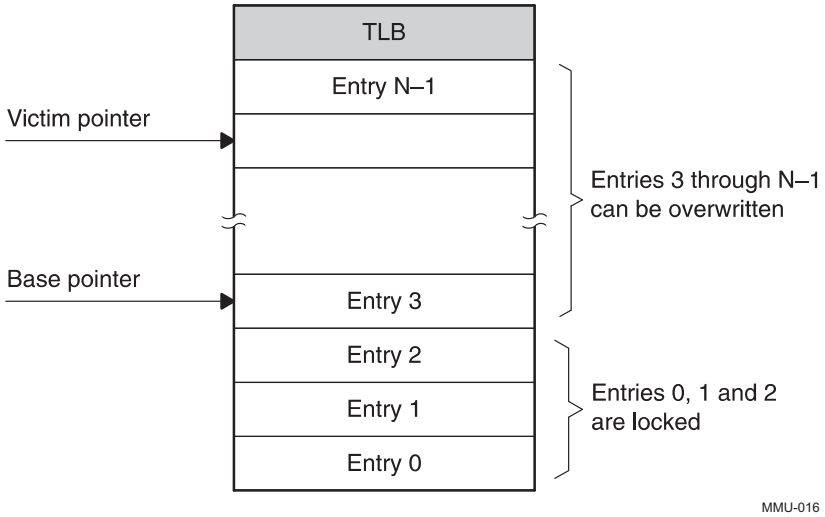
Translating virtual to physical addresses is required for each memory access in systems using an MMU. To accelerate this translation process, a cache, or TLB, holds the result of recent translations.

For every translation, the MMU internal logic first checks whether the requested translation is already cached in the TLB. If the translation is cached, this translation is used; otherwise the translation is retrieved from the translation tables and the TLB is updated. If the TLB is full, one of its entries must be replaced. This entry is selected on a random basis.

The first *n* TLB entries, where *n* < *Total Number N of TLB Entries*, can be protected (locked) against being overwritten by setting the TLB base pointer to *n*. When this mechanism is used, only unprotected entries can be overwritten. The victim pointer indicates the next TLB entry to be written. Figure 8-16 shows an example of TLB with N TLB entries (ranging from 0 to N-1). The base pointer contains the value "3" protecting Entry 0, Entry 1, and Entry 2 and the victim pointer points to the next TLB entry to be updated.

Note: The last TLB entry (Entry N-1, where N = 8 for the camera MMU and N = 32 for the IVA2.2 MMU) always remains unprotected.

Figure 8-16. TLB Entry Lock Mechanism



The table walking logic automatically writes the TLB entries. The entries can also be manually written, which is done typically to ensure that the translation of time-critical data accesses is already present in the TLB so that they execute as fast as possible. The entries must be locked to prevent them from being overwritten.

8.3.4.1 TLB Entry Format

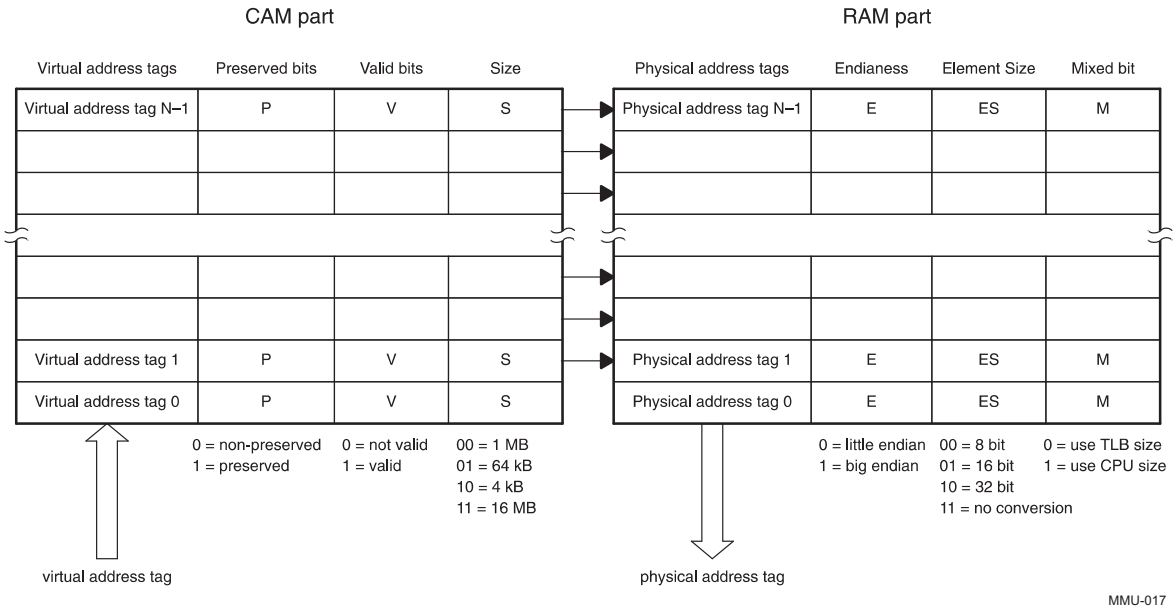
TLB entries consist of two parts:

- The CAM part contains the virtual address tag used to determine if a virtual address translation is in the TLB. The TLB acts like a fully associative cache addressed by the virtual address tag. The CAM part also contains the section/page size, as well as the preserved and the valid parameters. See the MMU_CAM register table for more details.
- The RAM part contains the address translation that belongs to the virtual address tag as well as the endianness, element size, and mixed parameters described in Section 8.3.3.2. See the MMU_RAM register table for more details.

The valid parameter specifies whether an entry is valid or not. The preserved parameter determines the behavior of an entry in the event of a TLB flush. If an entry is set as preserved, it is not deleted when a TLB is flushed, that is when MMU_GFLUSH[0] GLOBALFLUSH is set to 1. Preserved entries must be deleted manually. Section 8.3.3.2 describes the procedure to delete TLB entries.

Figure 8-17 shows the TLB entry structure.

Figure 8-17. TLB Entry Structure



8.3.5 MMU Error Handling

The following types of faults can occur:

- TLB miss with table walker disabled

  No translation is found for the virtual address required. If the hardware table walker is disabled, a fault is generated.
- Translation fault

  No translation is found for the virtual address required (TLB miss). The table walker is enabled but a page table entry does not exist for the given virtual address.
- Table walk fault

  A table walk results in a memory read error.
- Multi-hit fault

  More than one valid entry exists in the TLB for the given virtual address.

When a fault occurs and its corresponding interrupt is enabled, an interrupt is signaled to the MPU. The interrupt service routine (ISR) is then responsible for fault recovery. The requestor is stalled by the MMU while the fault is handled. For example, for a TLB miss, the ISR might load the missing entry into the TLB.

The ISR can determine the cause of the fault interrupt by reading the MMUn.MMU_IRQSTATUS register. The virtual address that caused the fault can be determined by reading the MMUn.MMU_FAULT_AD register.

In the case of a TLB miss, the MMU continues servicing the request as soon as a valid TLB entry is written. In the case of a translation fault, table walk fault, or mult-hit fault, the ISR first addresses the cause of the fault and then releases the MMU by writing to the interrupt status register. The MMU then continues servicing the request.

8.3.6 MMU Instance Design Parameters

The various MMU instances have different design parameters, most notably the size of the virtual address space and the number of TLB entries. Table 8-7 shows the correspondence between the MMU instances and the design parameters.

Table 8-7. Design Parameters of the MMU Instances

MMU Instance	Virtual Address Space	TLB Entries
MMU1 (camera MMU)	4GB	8 entries
MMU2 (IVA2.2 MMU)	4GB	32 entries

8.4 MMU Basic Programming Model

MMU instances handle translation from virtual into physical addresses. Virtual addresses are issued by the Camera or the IVA2.2 subsystems to the MMU, which converts them into physical addresses. These physical addresses correspond to actual memory resource. Refer to Chapter 2, *Memory Mapping* for more information on the device memory mapping.

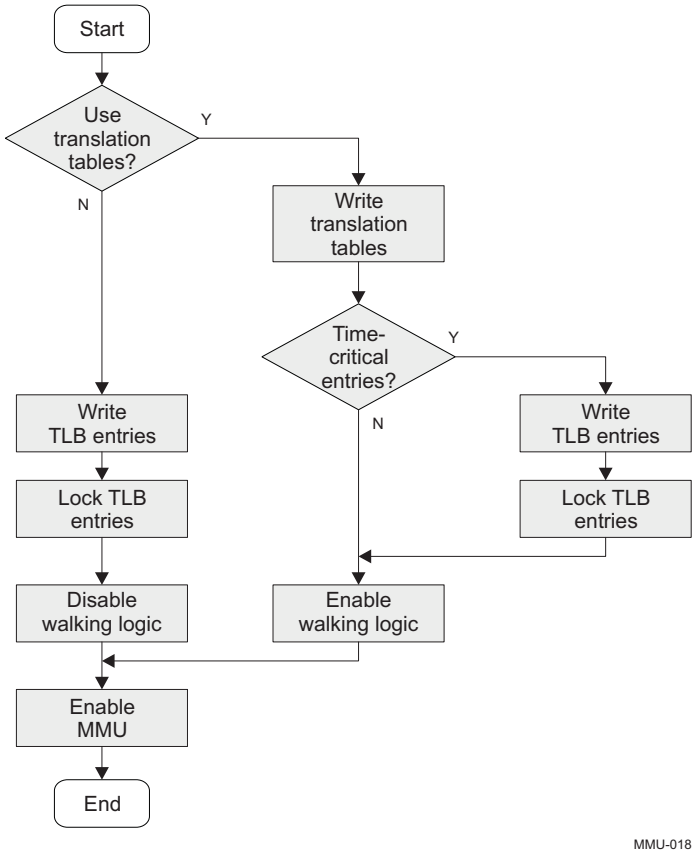
MMU instances can be used dynamically or statically, in other words, MMU can be managed by the MPU software or configured directly.

An MMU is configured dynamically when a software subroutine writes translation tables into the appropriate physical memory space. Translation tables are most likely stored in external SDRAM (see SMS, Section 11.2.4.1, *SDRAM Memory Scheduler*, for more details). They are automatically written by the MPU OS (Symbian™, Linux®,…). Operating System may limit the memory section size to page or sections. A MPU memory management software treats all information concerning addressing. Sub-set tables can be copied into external SDRAM to automatically create a given MMU translation table. Features such as mixed region, endianness, element size are then overwritten in SDRAM to match one's need and customize Camera or IVA2.2 subsystems' need (8-bit exchange in little-endian format for example). Note that 16MB sections and 64KB page table entries must be duplicated 16 times with the same content in translation tables. This process avoids a second read operation when an access points to another element than first one in the table. If they are not copied the MMU will behave in an unpredictable way.

When an MMU is configured statically, translation tables are not used, and TLB entries are written directly by the programmer.

Figure 8-18 shows the configuration strategies: static, dynamic and mixed use of MMU.

Figure 8-18. MMU Configuration Strategies



MMU components must be set up before the MMU can be used for address translation. The following steps are required:

- Initialize translation tables and the table walker logic if translation tables are used.

- Write TLB entries if required.
- Enable MMU.

Next paragraph explains how to configure the IVA2.2 MMU by directly writing the entries in the Translation Look-aside Buffer (TLB). The focus of this example is to explain the required configuration steps rather than to provide the most efficient implementation.

8.4.1 Writing TLB Entries Statically

This example deals with predefined translation strategy, that is, entries are “statically” written in the TLB.

This method avoids the need to write Translation tables in memory and is commonly used for relatively small address spaces. It ensures that the translation of time-critical data accesses execute as fast as possible with entries already present in the TLB. These entries must be locked to prevent them from being overwritten. To setup the MMU follow those steps:

1. Reset MMU: write MMU_SYSCONFIG[1] SOFTRESET = 1 and wait for the system reset completion by polling MMU_SYSSTATUS[0] RESETDONE until it is equal to 1.
2. Set MMU_SYSCONFIG[0] AUTOIDLE bit to enable power saving via automatic interface clock gating.

   After reset the Table walking logic remain disabled (MMU_CNTL[2] TWLENABLE = 0), the TLB is empty and the victim pointer points to the first TLB entry, Entry 0 (MMU_LOCK[8:4] CURRENTVICTIM at 0).

   To initialize a TLB entry, follow those steps:
3. Load the Virtual Address (VATAG), the preserved (P=1) and valid (V=1) bits and the page size (small or large page, section, supersection) into MMU_CAM register.
4. Load the Physical Address (PHYSICALADDRESS), the endianness (ENDIANNESS = 0), element size (ELEMENTSIZE) and mixed page attributes bits (MIXED) into MMU_RAM register.
5. Specify the TLB entry you want to write by setting the MMU_LOCK[8:4] CURRENTVICTIM pointer. Start with TLB Entry 0 and increment this pointer for each subsequent entry you want to write.
6. Load the specified entry in the TLB by setting MMU_LD_TLB[0] LDTLBITEM = 1.
7. Repeat steps 3 to 6 for all entries you want to write.

   Remember to increment the MMU_LOCK[8:4] CURRENTVICTIM pointer with each entry you are writing. To prevent replacement of TLB entries see Section 8.4.1.1, Protecting TLB Entries.

   To enable error handling when more than one valid entry exists in the TLB for the given virtual address or when no translation is found for the virtual address required (with the Table walking logic disabled):
8. Enable Multi-hit fault and TLB miss with table walker disabled interrupts by writing 1 in MMU_IRQENABLE[4] MULTIHITFAULT and MMU_IRQENABLE[0] TLBMISS.
9. To determine the cause of the fault interrupt the interrupt service routine (ISR) can read corresponding MMU_IRQSTATUS bits. The virtual address that caused the fault can be determined by reading MMU_FAULT_AD.

   Note: MMU errors result in a memory stall; the MMU will not process any request until the cause of the error has been addressed.

   To enable memory translations enable MMU (virtual addresses are treated as physical addresses when MMU is disabled):
10. Set MMU_CNTL[1] MMUENABLE = 1 to enable memory translations enable the MMU (virtual addresses are treated as physical addresses when the MMU is disabled.

8.4.1.1 Protecting TLB Entries

The first *n* TLB entries (with *n* < total number of TLB entries) can be protected from being overwritten with new translations. This is useful to ensure that certain commonly used or time-critical translations are always in the TLB and do not require retrieval via the table walking process.

The entry protection mechanism is shown in Figure 8-16. To protect the first n TLB entries, set the MMU1.MMU_LOCK[12:10] BASEVALUE bit field for the camera MMU (MMU2.MMU_LOCK[14:10] BASEVALUE field for the IVA2.2 MMU) to n.

8.4.1.2 Deleting TLB Entries

Two mechanisms exist to delete TLB entries. All unpreserved TLB entries, i.e., TLB entries that were written with the preserved bit set to zero, can be deleted by invoking a TLB flush. Such a TLB flush is invoked by setting the MMUn.MMU_GFLUSH[0] GLOBALFLUSH bit.

Individual TLB entries can be flushed, regardless of the preserved bit setting, by specifying its virtual address in the MMUn.MMU_CAM register and setting the MMUn.MMU_FLUSH_ENTRY[0] FLUSHENTRY bit.

The preserved bit should only be used on protected TLB entries, *as it does not prevent replacement by the table walking logic*.

8.4.1.3 Reading TLB Entries

TLB entries can be read by you to determine the TLB content at runtime. In doing so, the TLB entry number is specified by setting the MMUn.MMU_LOCK[8:4] CURRENTVICTIM pointer. CAM and RAM parts of the TLB entry can then be read in the MMUn.MMU_READ_CAM and MMUn.MMU_READ_RAM registers, respectively.

8.4.2 Programming the MMU Dynamically

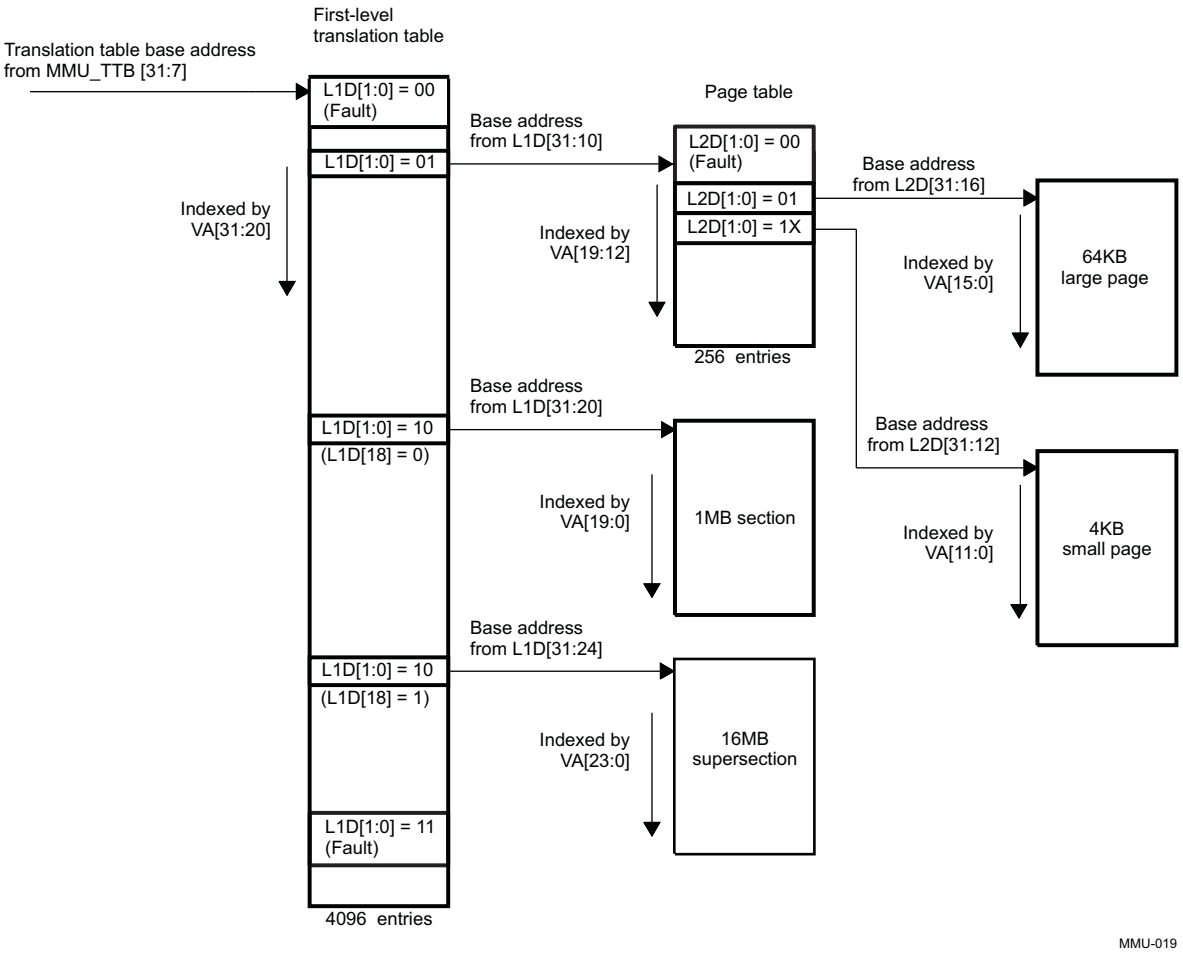
When translation tables are used for MMU address translation they must be properly set up and the table walking logic must be enabled.

The following page sizes are supported:

- Supersection: 16M bytes
- Section: 1M byte
- Large page: 64K bytes
- Small page: 4K bytes

The memory location of these sections or pages (the Physical Address) is found by reading the Translation Table Hierarchy using a combination of the Virtual Address and the Translation Table Base address. Figure 8-19 illustrates the MMUn translation table hierarchy, with first-level descriptors (L1D) and second-level descriptors (L2D).

Figure 8-19. MMUn Translation Table Hierarchy



The first step is to build translation tables (first- and second-level translation tables depending on translation strategy) and place them into memory.

The start address of translation tables must always be aligned according to their size. For example, a 4096-entry first-level translation table must be aligned on a 16KB boundary (4096 Entries * 4 bytes = 16KB); that is, the lower 14 bits of the translation table start address (MMUn.MMU_TTB[13:0]) must be 0. For more details, see Section 8.3.3.2.

After the translation tables have been written to memory the translation table base address (that is, the most significant bits of the first-level translation table start address) must be set. This is done using the MMUn.MMU_TTB[31:7] TTBADDRESS bit field. See Section 8.5 for a complete description of the MMU control registers.

Once the translation tables have been written to memory and the translation table base is set, the table walking logic can be enabled. This is done by setting the MMUn.MMU_CNTL[2] TWLENABLE bit.

8.4.2.1 Programming the MMU Using First- and Second-Level Translation Tables

Translation tables must be set up and written in memory. Each TLB entry contains a memory page size: 1MB section or 16MB supersection. When the page size is smaller, 4KB or 64KB, second-level translation tables are necessary. To set up the MMU, follow the same steps as in the previous example but write the first- and second-level descriptors in physical memory.

The first-level translation table describes the translation properties for 1MB sections. To describe a 4-GB address range with 1-MB sections, 4096 32-bit entries (so-called first-level descriptors) are required. The first-level translation table start address must be aligned on a multiple of the table size with a 128-byte

minimum. Consequently, an alignment of at least 16K bytes is required for a complete 4096-entry table (that is, at least the last 14 address bits must be 0). The start address of the first-level translation table is specified by the so-called translation table base. The table is indexed by the upper 12-bits of the virtual address. Figure 8-20 through Figure 8-23 shows how the physical address is built from the virtual address as a function of the page sizes and hierarchy.

Figure 8-20. Translation of a Supersection

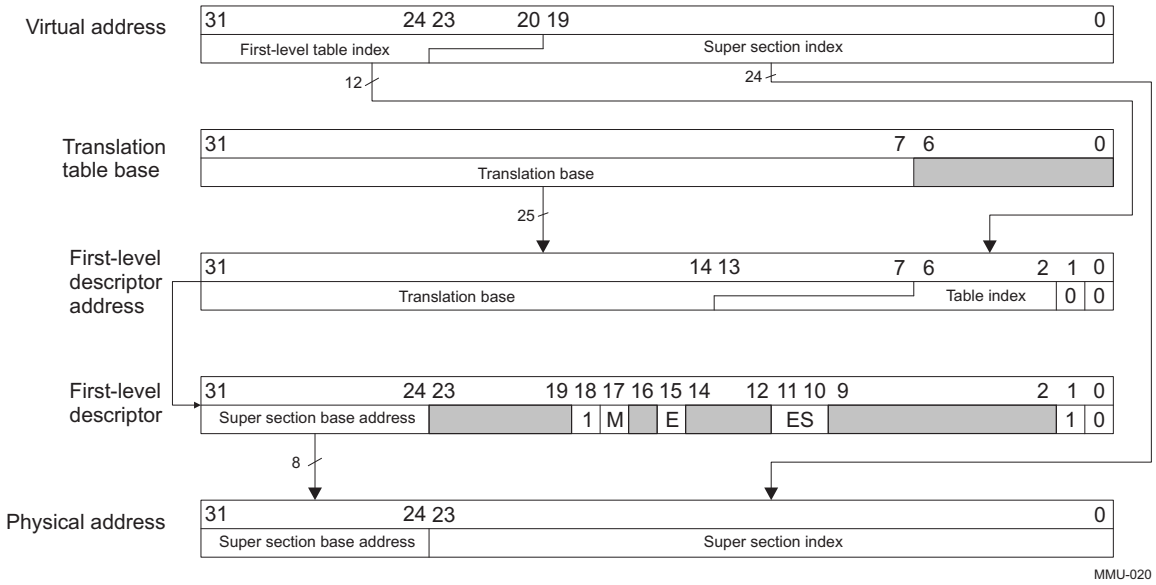


Figure 8-21. Translation of a Section

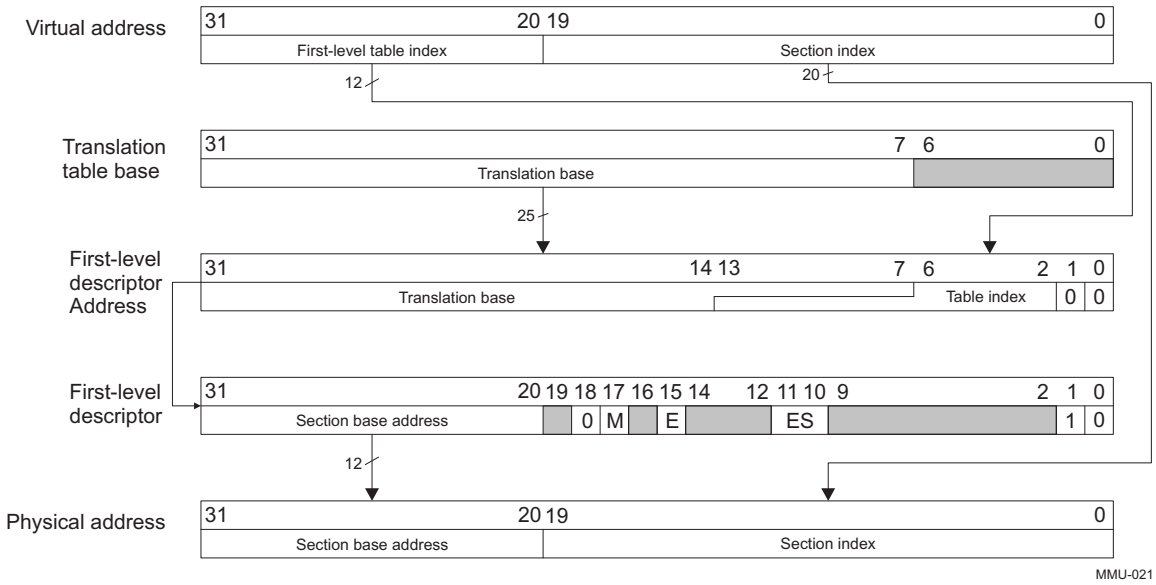


Figure 8-22. Translation of a Large Page Included in a Page Table

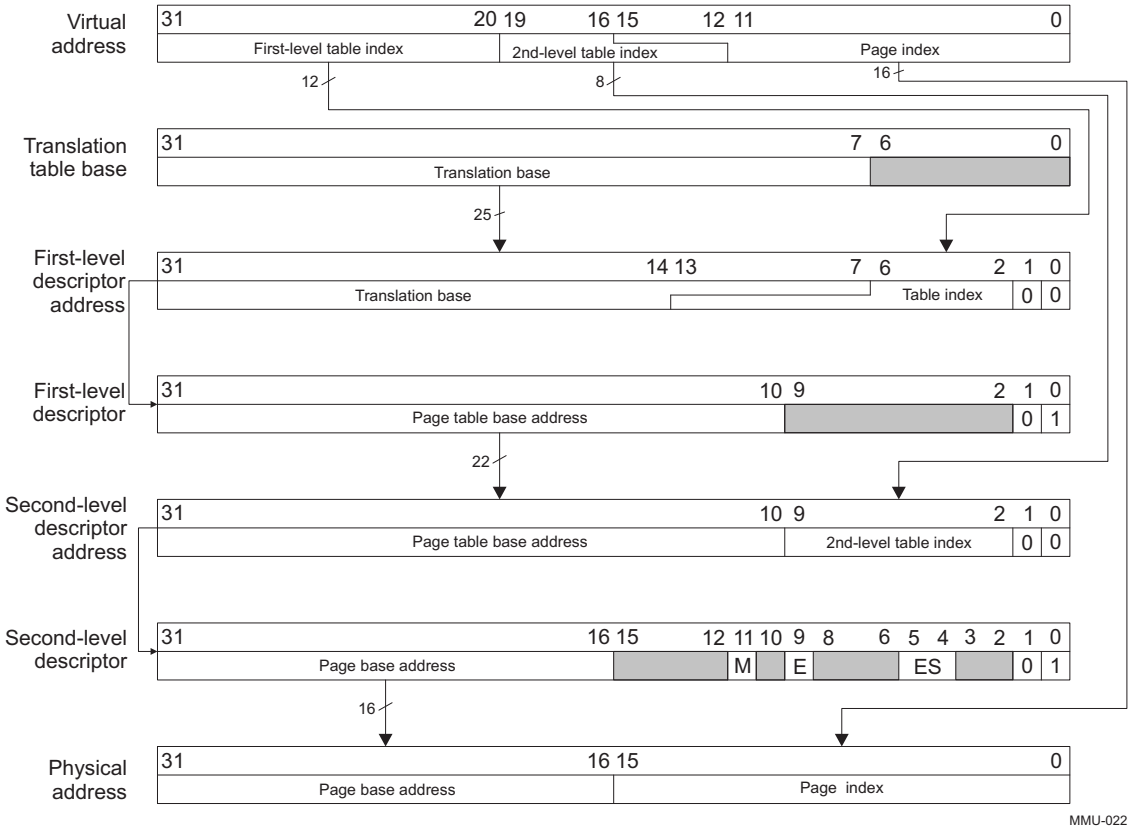
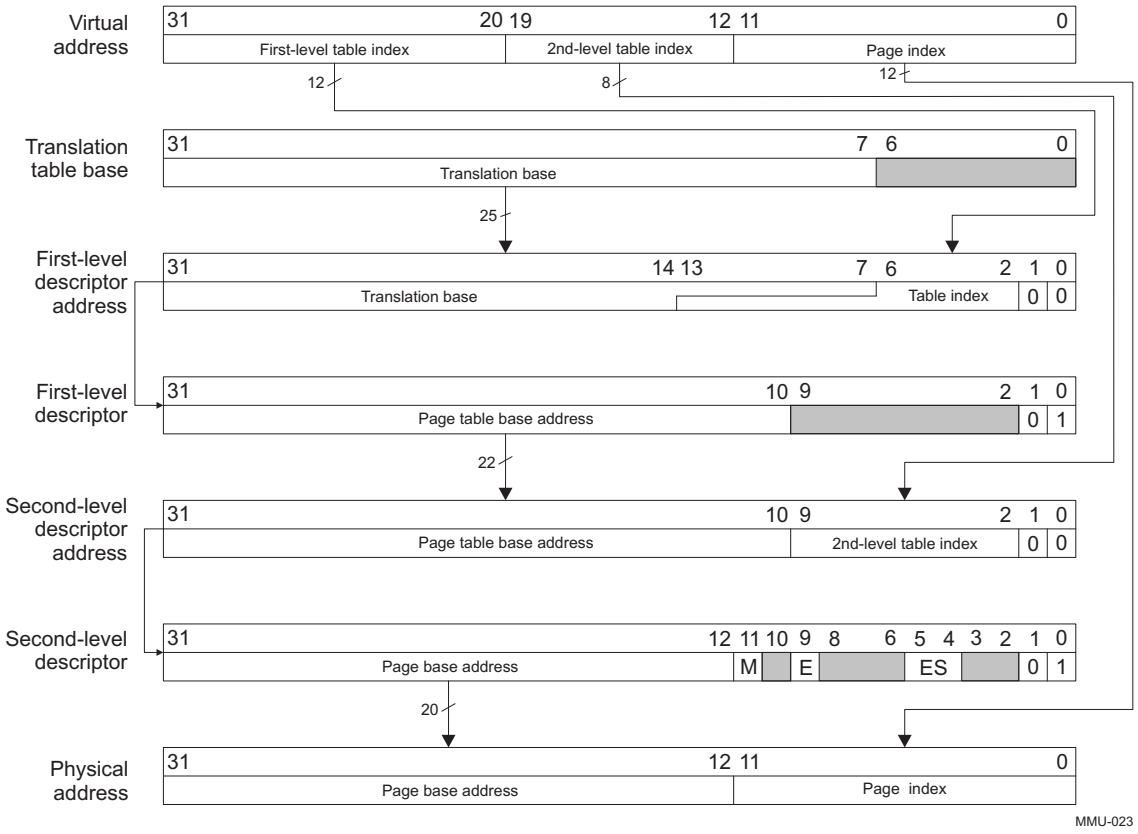


Figure 8-23. Translation of an Extended Small Page Included in a Page Table



8.5 MMU Register Manual

The MMU1 (camera ISP MMU) and MMU2 (IVA2.2 MMU) instances are programmed using two identical sets of configuration registers. Table 8-8 lists the base address of each register set.

Table 8-8. MMU Instance Summary

MMU Instance	Base Address	Size
MMU1 (Camera MMU)	0x480B D400	256 bytes
MMU2 (IVA2.2 MMU)	0x5D00 0000	256 bytes

Note: The MPU MMU is configured using register CP15. For more information, see the public ARM Cortex-A8 TRM.

8.5.1 Register Mapping Summary

Each MMU instance (except the MPU MMU) is programmed using a set of 18 configuration registers.Table 8-9 lists these registers and their access type, access size, and offset against their respective base address.

Note: The registers and their bit fields in this set are identical for the two instances except the MMUn.MMU_LOCK register, which has bit fields for the MMU1 (camera ISP MMU) that are different from the MMU2 (IVA2.2 MMU).

CAUTION

MMU2 instance (IVA2.2 MMU) registers are limited to 32-bit data accesses. Data access of 16 bits and 8 bits are not allowed and can corrupt register content.

Table 8-9. MMU Register Summary

Register Name	Type	Register Width (Bits)	Address Offset	MMU1 (Camera MMU) Physical Address	MMU2 (IVA2.2 MMU) Physical Address
MMU_REVISION	R	32	0x00	0x480B D400	0x5D00 0000
MMU_SYSCONFIG	RW	32	0x10	0x480B D410	0x5D00 0010
MMU_SYSSTATUS	R	32	0x14	0x480B D414	0x5D00 0014
MMU_IRQSTATUS	RW	32	0x18	0x480B D418	0x5D00 0018
MMU_IRQENABLE	RW	32	0x1C	0x480B D41C	0x5D00 001C
MMU_WALKING_ST	R	32	0x40	0x480B D440	0x5D00 0040
MMU_CNTL	RW	32	0x44	0x480B D444	0x5D00 0044
MMU_FAULT_AD	R	32	0x48	0x480B D448	0x5D00 0048
MMU_TTB	RW	32	0x4C	0x480B D44C	0x5D00 004C
MMU_LOCK	RW	32	0x50	0x480B D450	0x5D00 0050
MMU_LD_TLB	RW	32	0x54	0x480B D454	0x5D00 0054
MMU_CAM	RW	32	0x58	0x480B D458	0x5D00 0058
MMU_RAM	RW	32	0x5C	0x480B D45C	0x5D00 005C
MMU_GFLUSH	RW	32	0x60	0x480B D460	0x5D00 0060
MMU_FLUSH_ENTRY	RW	32	0x64	0x480B D464	0x5D00 0064
MMU_READ_CAM	R	32	0x68	0x480B D468	0x5D00 0068

Table 8-9. MMU Register Summary (continued)

Register Name	Type	Register Width (Bits)	Address Offset	MMU1 (Camera MMU) Physical Address	MMU2 (IVA2.2 MMU) Physical Address
MMU_READ_RAM	R	32	0x6C	0x480B D46C	0x5D00 006C
MMU_EMU_FAULT_AD	R	32	0x70	0x480B D470	0x5D00 0070

8.5.2 MMU Register Description

Table 8-10. MMU_REVISION

Address Offset	0x000		
Physical address	0x480B D400 0x5D00 0000	**Instance**	MMU1 (Camera ISP MMU) MMU2 (IVA2.2 MMU)
Description	This register contains the IP revision code.		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																								REV							

Bits	Field Name	Description	Type	Reset
31:8	Reserved	Reads return 0.	R	0x000000
7:0	REV	IP revision [7:4]: Major revision [3:0]: Minor revision Examples: 0x10 for 1.0, 0x21 for 2.1	R	See (1)

(1) TI internal data

Table 8-11. Register Call Summary for Register MMU_REVISION

MMU Register Manual

- Register Mapping Summary: [0]

Table 8-12. MMU_SYSCONFIG

Address Offset	0x010		
Physical address	0x480B D410 0x5D00 0010	**Instance**	MMU1 (Camera ISP MMU) MMU2 (IVA2.2 MMU)
Description	This register contains the various parameters of the interconnect interface.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																						CLOCK ACTIVITY		Reserved				IDLEMODE		Reserved	SOFTRESET	AUTOIDLE

Bits	Field Name	Description	Type	Reset
31:10	Reserved	Reads return 0. Write 0s for future compatibility.	R	0x000000
9:8	CLOCKACTIVITY	Clock activity during wake-up mode Read 0x0: Functional and interconnect clocks can be switched off. Read 0x1, 0x2, 0x3: never happens. Write 0s for future compatibility.	R	0x0
7:5	Reserved	Reads return 0. Write 0s for future compatibility.	R	0x0
4:3	IDLEMODE	Idle mode	RW	0x0

Bits	Field Name	Description	Type	Reset
		0x0: Force idle. Idle request is acknowledged unconditionally. 0x1: No idle. Idle request is never acknowledged. 0x2: Smart idle. Acknowledgement to an idle request is given based on the internal activity of the module. 0x3: Reserved - Do not use		
2	Reserved	Reads return 0. Write 0s for future compatibility.	R	0
1	SOFTRESET	Software reset. This bit is automatically reset by the hardware. During reads, it always returns 0. Read 0x0: Always returns 0 Write 0x0: No functional effect Read 0x1: Never happens Write 0x1: The module is reset.	RW	0
0	AUTOIDLE	Internal interconnect clock gating strategy 0x0: Interconnect clock is free-running. 0x1: Automatic interconnect clock gating strategy is applied, based on the interconnect interface activity.	RW	0

Table 8-13. Register Call Summary for Register MMU_SYSCONFIG

MMU Integration • System Power Management: [0] [1] [2] • Module Power Saving: [3] [4] • Reset: [5]
Basic Programming Model • Writing TLB entries statically: [6] [7]
MMU Register Manual • Register Mapping Summary: [8]

Table 8-14. MMU_SYSSTATUS

Address Offset	0x014		
Physical address	0x480B D414 0x5D00 0014	**Instance**	MMU1 (Camera ISP MMU) MMU2 (IVA2.2 MMU)
Description	This register provides status information about the module, excluding the interrupt status information.		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																								Reserved							RESETDONE

Bits	Field Name	Description	Type	Reset
31:8	Reserved	Reads return 0.	R	0x000000
7:1	Reserved	Reads return 0. Reserved for interconnect-socket status information	R	0x00
0	RESETDONE	Internal reset monitoring 0x0: Internal module reset in ongoing. 0x1: Reset completed	R	-

Table 8-15. Register Call Summary for Register MMU_SYSSTATUS

MMU Integration • Reset: [0]
Basic Programming Model • Writing TLB entries statically: [1]
MMU Register Manual • Register Mapping Summary: [2]

Table 8-16. MMU_IRQSTATUS

Address Offset	0x018		
Physical address	0x480B D418 0x5D00 0018	**Instance**	MMU1 (Camera ISP MMU) MMU2 (IVA2.2 MMU)
Description	This interrupt status register regroups all the status of the module internal events that can generate an interrupt.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																											MULTIHITFAULT	TABLEWALKFAULT	EMUMISS	TRANSLATIONFAULT	TLBMISS

Bits	Field Name	Description	Type	Reset
31:5	Reserved	Reads return 0. Write 0s for future compatibility.	R	0x0000000
4	MULTIHITFAULT	Error due to multiple matches in the TLB	RW	0
		Read 0x0: MultiHitFault false		
		Write 0x0: MultiHitFault status bit unchanged		
		Read 0x1: MultiHitFault is true (pending)		
		Write 0x1: TableWalkFault status bit is reset		
3	TABLEWALKFAULT	Error response received during a table walk	RW	0
		Read 0x0: TableWalkFault false		
		Write 0x0: TableWalkFault status bit unchanged		
		Read 0x1: TableWalkFault is true (pending)		
		Write 0x1: TableWalkFault status bit is reset		
2	EMUMISS	Unrecoverable TLB miss during debug (hardware TWL disabled)	RW	0
		Read 0x0: EMUMiss false		
		Write 0x0: EMUMiss status bit unchanged		
		Read 0x1: EMUMiss is true (pending)		
		Write 0x1: EMUMiss status bit is reset		
1	TRANSLATION FAULT	Invalid descriptor in translation tables (translation fault)	RW	0
		Read 0x0: TranslationFault false		
		Write 0x0: TranslationFault status bit unchanged		
		Read 0x1: TranslationFault is true (pending)		
		Write 0x1: TranslationFault status bit is reset		
0	TLBMISS	Unrecoverable TLB miss (hardware TWL disabled)	RW	0
		Read 0x0: TLBMiss false		
		Write 0x0: TLBMiss status bit unchanged		
		Read 0x1: TLBMiss is true (pending)		

Bits	Field Name	Description	Type	Reset
		Write 0x1: TLBMiss status bit is reset		

Table 8-17. Register Call Summary for Register MMU_IRQSTATUS

MMU Integration • Interrupts: [0]
MMU Functional Description • MMU Error Handling: [1]
Basic Programming Model • Writing TLB entries statically: [2]
MMU Register Manual • Register Mapping Summary: [3]

Table 8-18. MMU_IRQENABLE

Address Offset	0x01C		
Physical address	0x480B D41C 0x5D00 001C	**Instance**	MMU1 (Camera ISP MMU) MMU2 (IVA2.2 MMU)
Description	The interrupt enable register allows the module's internal sources of interrupt to be masked and unmasked on an event-by-event basis..		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																											MULTIHITFAULT	TABLEWALKFAULT	EMUMISS	TRANSLATIONFAULT	TLBMISS

Bits	Field Name	Description	Type	Reset
31:5	Reserved	Reads return 0. Write 0s for future compatibility.	R	0x0000000
4	MULTIHITFAULT	Error due to multiple matches in the TLB	RW	0
		0x0: MultiHitFault interrupt is masked		
		0x1: MultiHitFault event generates an interrupt if occurs		
3	TABLEWALKFAULT	Error response received during a table walk	RW	0
		0x0: TableWalkFault interrupt is masked		
		0x0: TableWalkFault event generates an interrupt if occurs		
2	EMUMISS	Unrecoverable TLB miss during debug (hardware TWL disabled)	RW	0
		0x0: EMUMiss interrupt is masked		
		0x1: EMUMiss event generates an interrupt if occurs		
1	TRANSLATIONFAULT	Invalid descriptor in translation tables (translation fault)	RW	0
		0x0: TranslationFault interrupt is masked		
		0x1: TranslationFault generates an interrupt if occurs		
0	TLBMISS	Unrecoverable TLB miss (hardware TWL disabled)	RW	0
		0x0: TLBMiss interrupt is masked		
		0x1: TLBMiss event generates an interrupt if occurs		

Table 8-19. Register Call Summary for Register MMU_IRQENABLE

MMU Integration • Interrupts: [0]
Basic Programming Model • Writing TLB entries statically: [1] [2]
MMU Register Manual • Register Mapping Summary: [3]

Table 8-20. MMU_WALKING_ST

Address Offset	0x040		
Physical address	0x480B D440 0x5D00 0040	**Instance**	MMU1 (Camera ISP MMU) MMU2 (IVA2.2 MMU)
Description	This register provides status information about the table walking logic.		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																															TWLRUNNING

Bits	Field Name	Description	Type	Reset
31:1	Reserved	Reads return 0.	R	0x00000000
0	TWLRUNNING	Table walking logic is running. 0x0: TWL completed 0x1: TWL running	R	0

Table 8-21. Register Call Summary for Register MMU_WALKING_ST

MMU Register Manual • Register Mapping Summary: [0]

Table 8-22. MMU_CNTL

Address Offset	0x044		
Physical address	0x480B D444 0x5D00 0044	**Instance**	MMU1 (Camera ISP MMU) MMU2 (IVA2.2 MMU)
Description	This register programs the MMU features.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																												EMUTLBUPDATE	TWLENABLE	MMUENABLE	Reserved

Bits	Field Name	Description	Type	Reset
31:4	Reserved	Reads return 0. Write 0s for future compatibility.	R	0x0000000
3	EMUTLBUPDATE	Enable TLB update on emulator table walk 0x0: Emulator TLB update disabled 0x1: Emulator TLB update enabled	RW	0

Bits	Field Name	Description	Type	Reset
2	TWLENABLE	Table walking logic enable 0x0: TWL disabled 0x0: TWL enabled	RW	0
1	MMUENABLE	MMU enable 0x0: MMU disabled 0x1: MMU enabled	RW	0
0	Reserved	Reads return 0. Write 0s for future compatibility.	R	0

Table 8-23. Register Call Summary for Register MMU_CNTL

MMU Functional Description • MMU Functional Description: [0]
Basic Programming Model • Writing TLB entries statically: [1] [2] • Programming the MMU dynamically: [3]
MMU Register Manual • Register Mapping Summary: [4]

Table 8-24. MMU_FAULT_AD

Address Offset	0x048		
Physical address	0x480B D448 0x5D00 0048	**Instance**	MMU1 (Camera ISP MMU) MMU2 (IVA2.2 MMU)
Description	This register contains the virtual address that generated the interrupt.		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FAULTADDRESS																															

Bits	Field Name	Description	Type	Reset
31:0	FAULTADDRESS	Virtual address of the access that generated a fault	R	0x00000000

Table 8-25. Register Call Summary for Register MMU_FAULT_AD

MMU Integration • Interrupts: [0]
MMU Functional Description • MMU Error Handling: [1]
Basic Programming Model • Writing TLB entries statically: [2]
MMU Register Manual • Register Mapping Summary: [3]

Table 8-26. MMU_TTB

Address Offset	0x04C		
Physical address	0x480B D44C 0x5D00 004C	**Instance**	MMU1 (Camera ISP MMU) MMU2 (IVA2.2 MMU)
Description	This register contains the resolution table base address.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TTBADDRESS																								Reserved							

Bits	Field Name	Description	Type	Reset
31:7	TTBADDRESS	Translation table base address	RW	0x0000000
6:0	Reserved	Reads return 0. Write 0s for future compatibility.	R	0x00

Table 8-27. Register Call Summary for Register MMU_TTB

Basic Programming Model
- Programming the MMU dynamically: [0] [1]

MMU Register Manual
- Register Mapping Summary: [2]

Table 8-28. MMU_LOCK

Address Offset	0x050		
Physical address	0x480B D450 0x5D00 0050	**Instance**	MMU1 (Camera ISP MMU) MMU2 (IVA2.2 MMU)
Description	This register locks some of the TLB entries or specifies the TLB entry to be read.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																BASEVALUE						Reserved		CURRENTVICTIM				Reserved			

Bits	Field Name	Description	Type	Reset
31:15	Reserved	Reads return 0. Write 0s for future compatibility.	R	0x00000
14:10	BASEVALUE	Locked entries base value Note: In the Camera MMU instance, BASEVALUE is a 3-bit field, ie. bits 13 and 14 are reserved.	RW	0x00
9	Reserved	Reads return 0. Write 0s for future compatibility.	R	0
8:4	CURRENTVICTIM	Current entry to be updated either by the TWL or by the software or TLB entry to be read Note: In the Camera MMU instance, CURRENTVICTIM is a 3-bit field, ie. bits 7 and 8 are reserved. Write value: TLB entry to be updated by software or TLB entry to be read Read value: TLB entry to be updated by table walk logic	RW	0x00
3:0	Reserved	Reads return 0. Write 0s for future compatibility.	R	0x0

Table 8-29. Register Call Summary for Register MMU_LOCK

Basic Programming Model
- Writing TLB entries statically: [0] [1] [2]
- Protecting TLB Entries: [3] [4]
- Reading TLB Entries: [5]

Table 8-29. Register Call Summary for Register MMU_LOCK (continued)

MMU Register Manual • Register Mapping Summary: [6] [7]

Table 8-30. MMU_LD_TLB

Address Offset	0x054		
Physical address	0x480B D454 0x5D00 0054	**Instance**	MMU1 (Camera ISP MMU) MMU2 (IVA2.2 MMU)
Description	This register loads a TLB entry (CAM+RAM).		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																															LDTLBITEM

Bits	Field Name	Description	Type	Reset
31:1	Reserved	Reads return 0. Write 0s for future compatibility	R	0x00000000
0	LDTLBITEM	Write (load) data in the TLB	RW	0
		Read 0x0: Always returns 0		
		Write 0x0: No functional effect		
		Read 0x1: Never happens		
		Write 0x1: Load TLB data		

Table 8-31. Register Call Summary for Register MMU_LD_TLB

Basic Programming Model • Writing TLB entries statically: [0]
MMU Register Manual • Register Mapping Summary: [1]

Table 8-32. MMU_CAM

Address Offset	0x058		
Physical address	0x480B D458 0x5D00 0058	**Instance**	MMU1 (Camera ISP MMU) MMU2 (IVA2.2 MMU)
Description	This register holds a CAM entry.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
VATAG																Reserved												P	V	PAGESIZE	

Bits	Field Name	Description	Type	Reset
31:12	VATAG	Virtual address tag	RW	0x00000
11:4	Reserved	Reads return 0. Write 0s for future compatibility.	R	0x00
3	P	Preserved bit	RW	0
		0x0: TLB entry can be flushed		
		0x1: TLB entry is protected against flush		
2	V	Valid bit	RW	0

Bits	Field Name	Description	Type	Reset
		0x0: TLB entry is invalid		
		0x1: TLB entry is valid		
1:0	PAGESIZE	Page size	RW	0x0
		0x0: Section (1MB)		
		0x1: Large page (64KB)		
		0x2: Small page (4KB)		
		0x3: Supersection (16MB)		

Table 8-33. Register Call Summary for Register MMU_CAM

MMU Functional Description
- TLB Entry Format: [0]

Basic Programming Model
- Writing TLB entries statically: [1]
- Deleting TLB Entries: [2]

MMU Register Manual
- Register Mapping Summary: [3]
- MMU Register Description: [4]

Table 8-34. MMU_RAM

Address Offset	0x05C		
Physical address	0x480B D45C 0x5D00 005C	**Instance**	MMU1 (Camera ISP MMU) MMU2 (IVA2.2 MMU)
Description	This register holds a RAM entry.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PHYSICALADDRESS																				Reserved		ENDIANNESS		ELEMENTSIZE		MIXED		Reserved			

Bits	Field Name	Description	Type	Reset
31:12	PHYSICALADDRESS	Physical address of the page	RW	0x00000
11:10	Reserved	Reads return 0. Write 0s for future compatibility.	R	0x0
9	ENDIANNESS	Endianness of the page	RW	0
		0x0: Little endian		
		0x1: Big endian - must not be used (locked on little endian)		
8:7	ELEMENTSIZE	Element size of the page (8, 16, 32, no translation)	RW	0x0
		0x0: 8 bits		
		0x1: 16 bits		
		0x2: 32 bits		
		0x3: No translation		
6	MIXED	Mixed page attribute (use CPU element size)	RW	0
		0x0: Use TLB element size		
		0x1: Use CPU element size		
5:0	Reserved	Reads return 0. Write 0s for future compatibility.	R	0x00

Table 8-35. Register Call Summary for Register MMU_RAM

MMU Functional Description • TLB Entry Format: [0]
Basic Programming Model • Writing TLB entries statically: [1]
MMU Register Manual • Register Mapping Summary: [2]

Table 8-36. MMU_GFLUSH

Address Offset	0x060		
Physical address	0x480B D460 0x5D00 0060	**Instance**	MMU1 (Camera ISP MMU) MMU2 (IVA2.2 MMU)
Description	This register flushes all the non-protected TLB entries.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																															GLOBALFLUSH

Bits	Field Name	Description		Type	Reset
31:1	Reserved	Reads return 0. Write 0s for future compatibility.		RW	0x00000000
0	GLOBALFLUSH	Flush all the non-protected TLB entries when set		RW	0
		Read 0x0:	Always returns 0		
		Write 0x0:	No functional effect		
		Read 0x1:	Never happens		
		Write 0x1:	Flush all the non-protected TLB entries		

Table 8-37. Register Call Summary for Register MMU_GFLUSH

MMU Functional Description • TLB Entry Format: [0]
Basic Programming Model • Deleting TLB Entries: [1]
MMU Register Manual • Register Mapping Summary: [2]

Table 8-38. MMU_FLUSH_ENTRY

Address Offset	0x064		
Physical address	0x480B D464 0x5D00 0064	**Instance**	MMU1 (Camera ISP MMU) MMU2 (IVA2.2 MMU)
Description	This register flushes the entry pointed to by the CAM virtual address.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																															FLUSHENTRY

Bits	Field Name	Description	Type	Reset
31:1	Reserved	Reads return 0. Write 0s for future compatibility.	RW	0x00000000
0	FLUSHENTRY	Flush the TLB entry pointed by the virtual address (VATag) in MMU_CAM register, even if this entry is set protected Read 0x0: Always returns 0 Write 0x0: No functional effect Read 0x1: Never happens Write 0x1: Flush all the TLB entries specified by the CAM register	RW	0

Table 8-39. Register Call Summary for Register MMU_FLUSH_ENTRY

Basic Programming Model • Deleting TLB Entries: [0]
MMU Register Manual • Register Mapping Summary: [1]

Table 8-40. MMU_READ_CAM

Address Offset	0x068		
Physical address	0x480B D468 0x5D00 0068	**Instance**	MMU1 (Camera ISP MMU) MMU2 (IVA2.2 MMU)
Description	This register reads CAM data from a CAM entry.		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
VATAG																Reserved								P	V	PAGESIZE					

Bits	Field Name	Description	Type	Reset
31:12	VATAG	Virtual address tag	R	0x00000
11:4	Reserved	Reads return 0.	R	0x00
3	P	Preserved bit 0x0: TLB entry can be flushed 0x1: TLB entry is protected against flush	R	0
2	V	Valid bit 0x0: TLB entry is invalid 0x1: TLB entry is valid	R	0
1:0	PAGESIZE	Page size 0x0: Section (1MB) 0x1: Large page (64KB) 0x2: Small page (4KB) 0x3: Supersection (16MB)	R	0x0

Table 8-41. Register Call Summary for Register MMU_READ_CAM

Basic Programming Model • Reading TLB Entries: [0]
MMU Register Manual • Register Mapping Summary: [1]

Table 8-42. MMU_READ_RAM

Address Offset	0x06C		
Physical address	0x480B D46C 0x5D00 006C	**Instance**	MMU1 (Camera ISP MMU) MMU2 (IVA2.2 MMU)
Description	This register reads RAM data from a RAM entry.		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PHYSICALADDRESS																				Reserved		ENDIANNESS	ELEMENTSIZE		MIXED	Reserved						

Bits	Field Name	Description	Type	Reset
31:12	PHYSICALADDRESS	Physical address of the page	R	0x00000
11:10	Reserved	Reads return 0.	R	0x0
9	ENDIANNESS	Endianness of the page	R	0
		0x0: Little endian		
		0x1: Big endian - must not be used (locked on little endian)		
8:7	ELEMENTSIZE	Element size of the page (8, 16, 32 bits or no translation)	R	0x0
		0x0: 8 bits		
		0x1: 16 bits		
		0x2: 32 bits		
		0x3: No translation		
6	MIXED	Mixed page attribute (use CPU element size)	R	0
		0x0: Use TLB element size		
		0x1: Use CPU element size		
5:0	Reserved	Reads return 0. Write 0s for future compatibility.	R	0x00

Table 8-43. Register Call Summary for Register MMU_READ_RAM

Basic Programming Model

- Reading TLB Entries: [0]

MMU Register Manual

- Register Mapping Summary: [1]

Table 8-44. MMU_EMU_FAULT_AD

Address Offset	0x070		
Physical address	0x480B D470 0x5D00 0070	**Instance**	MMU1 (Camera ISP MMU) MMU2 (IVA2.2 MMU)
Description	This register contains the last virtual address of a fault caused by the debugger.		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EMUFAULTADDRESS																															

Bits	Field Name	Description	Type	Reset
31:0	EMUFAULTADDRESS	Virtual address of the last emulator access that generated a fault	R	0x00000000

Table 8-45. Register Call Summary for Register MMU_EMU_FAULT_AD

MMU Integration • Interrupts: [0]
MMU Register Manual • Register Mapping Summary: [1]

DMA

This chapter describes the system direct memory access (SDMA) of the OMAP34xx multimedia device.

Note: This chapter gives information about all modules and features in the high-tier device. See Section 1.5, *OMAP34xx Family*, to check availability of modules and features. Ensure that DMA requests of unavailable modules and features are masked in DMA subsystems.

Topic **Page**

9.1 SDMA Module Overview 972
9.2 SDMA Controller Environment 974
9.3 SDMA Module Integration 975
9.4 SDMA Functional Description 981
9.5 SDMA Basic Programming Model 997
9.6 SDMA Use Cases and Tips 1003
9.7 SDMA Registers Manual 1011

9.1 SDMA Module Overview

The System Direct Memory Access (SDMA), also called DMA4, performs high-performance data transfers between memories and peripheral devices without microprocessor unit (MPU) support during transfer. A DMA transfer is programmed through a logical DMA channel, which allows the transfer to be optimally tailored to the requirements of the application.

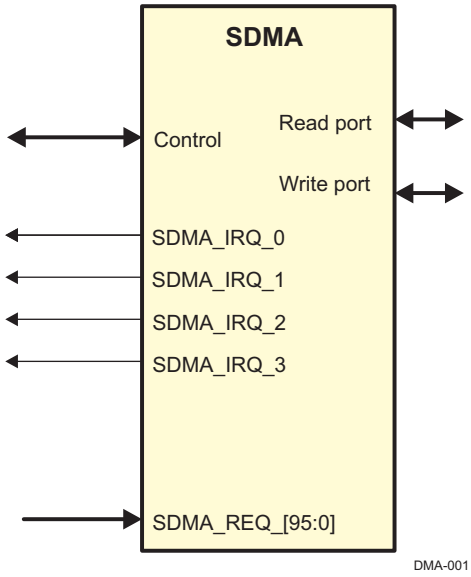
The device also embeds dedicated DMA controllers: the camera image signal processor (ISP) DMA; the enhanced DMA (EDMA), which is embedded in the IVA2.2 subsystem; the display DMA; and the universal serial bus (USB) high-speed (HS) DMA. For more information, see Chapter 12, *Camera ISP Subsystem*; Chapter 14, *IVA Subsystem*; Chapter 15, *Display Subsystem*; and Chapter 24, *High-Speed USB Controllers*.

The DMA controller includes the following main features:

- Data transfer support in either direction between:
  - Memory and memory
  - Memory and peripheral device
- 32 logical DMA channels supporting:
  - Multiple concurrent transfers
  - Independent transfer profile for each channel
  - 8-bit, 16-bit, or 32-bit data element transfer size
  - Software-triggered or hardware-synchronized transfers
  - Flexible source and destination address generation
  - Burst read and write
  - Chained multiple-channel transfers
  - Endianism conversion
- First-come, first-serve DMA scheduling with fixed priority
- Up to 96 DMA requests
- Constant fill
- Transparent copy
- Four programmable interrupt request output lines
- Software or hardware enabling
- FIFO depth: 256 x 32-bits
- Data buffering
- FIFO budget allocation
- Power-management support
- Auto-idle power-saving support
- Implementation of retention flip-flops (RFFs) to support dynamic power saving (DPS) between system power modes without MPU involvement

Figure 9-1 shows an overview of the SDMA module.

Figure 9-1. SDMA Overview



The SDMA module has two ports—one read and one write—and provides multiple logical channel support. A dynamically allocated FIFO queue memory pool provides buffering between the read and write ports.

The MPU configures the SDMA through the L4 interconnect.

9.2 SDMA Controller Environment

9.2.1 Environment Overview

The read and write ports of the SDMA controller are connected to the device through the L3 interconnect. The control port is connected to the L4 interconnect. The SDMA interrupt lines are connected to the interrupt lines of the MPU interrupt controller. The SDMA can handle up to 96 DMA requests, including four external hardware DMA requests.

9.2.2 SDMA Request Scheme

The hardware DMA request line schemes can be either edge-sensitive or transition-sensitive.

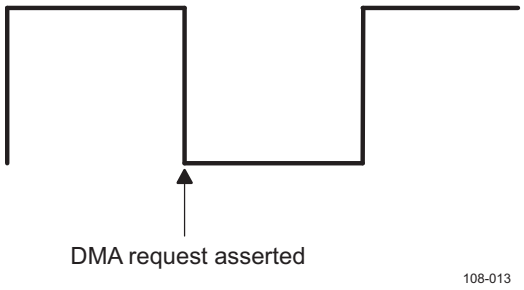
The sensitivity selection of the DMA request line can be configured in the system control module with the following register bits:

- CONTROL.CONTROL_DEVCONF0[0] SENSDMAREQ0 register bit for sys_ndmareq0
- CONTROL.CONTROL_DEVCONF0[1] SENSDMAREQ1 register bit for sys_ndmareq1
- CONTROL.CONTROL_DEVCONF1[7] SENSDMAREQ2 register bit for sys_ndmareq2
- CONTROL.CONTROL_DEVCONF1[8] SENSDMAREQ3 register bit for sys_ndmareq3

The default scheme is transition sensitive. All internal peripherals of the device use the transition-sensitive scheme.

Figure 9-2 shows the DMA request captured on a falling edge in the edge-sensitive scheme.

Figure 9-2. Edge-Sensitive DMA Request Scheme



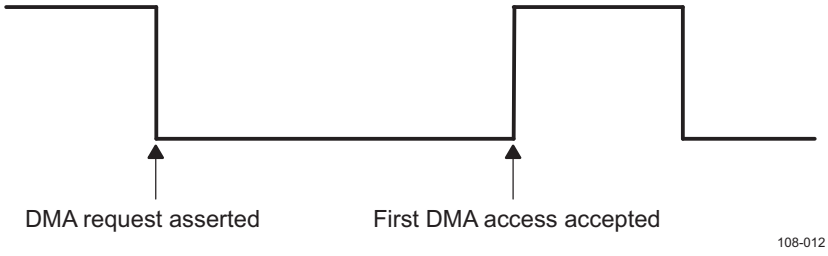
For a transition-sensitive DMA request (see Figure 9-3), the line must be maintained low (asserted) until the first DMA access is complete, after which the line must be maintained high (deasserted) for a minimum of one clock cycle (CORE_L3_ICLK):

When the deassertion time is less than one clock cycle, the SDMA might not detect the deassertion.

When the channel is enabled one cycle after a DMA request is disabled, the channel detects the DMA request and starts the corresponding transfer.

When the channel is enabled two cycles after the DMA request is disabled, the channel does not detect the DMA request.

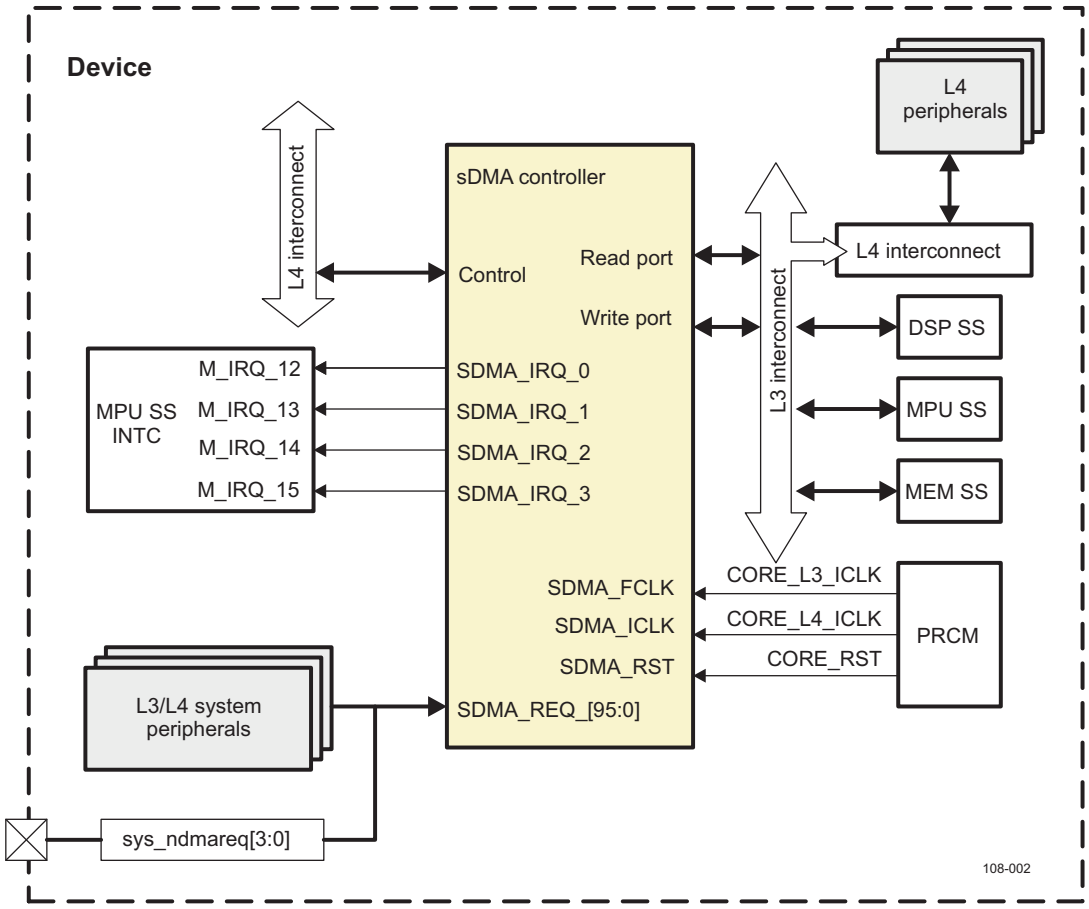
Figure 9-3. Transition-Sensitive DMA Request Scheme



9.3 SDMA Module Integration

Figure 9-4 highlights the SDMA controller integration.

Figure 9-4. SDMA Controller Integration



9.3.1 External SDMA Request Interface Description

The sys_ndmareq pins are optional external DMA requests used by external devices to establish direct hardware synchronization with the SDMA controller. A logical channel can be configured to respond to an external synchronization request. These requests can be configured to either active low on level or falling edge active by the control module.

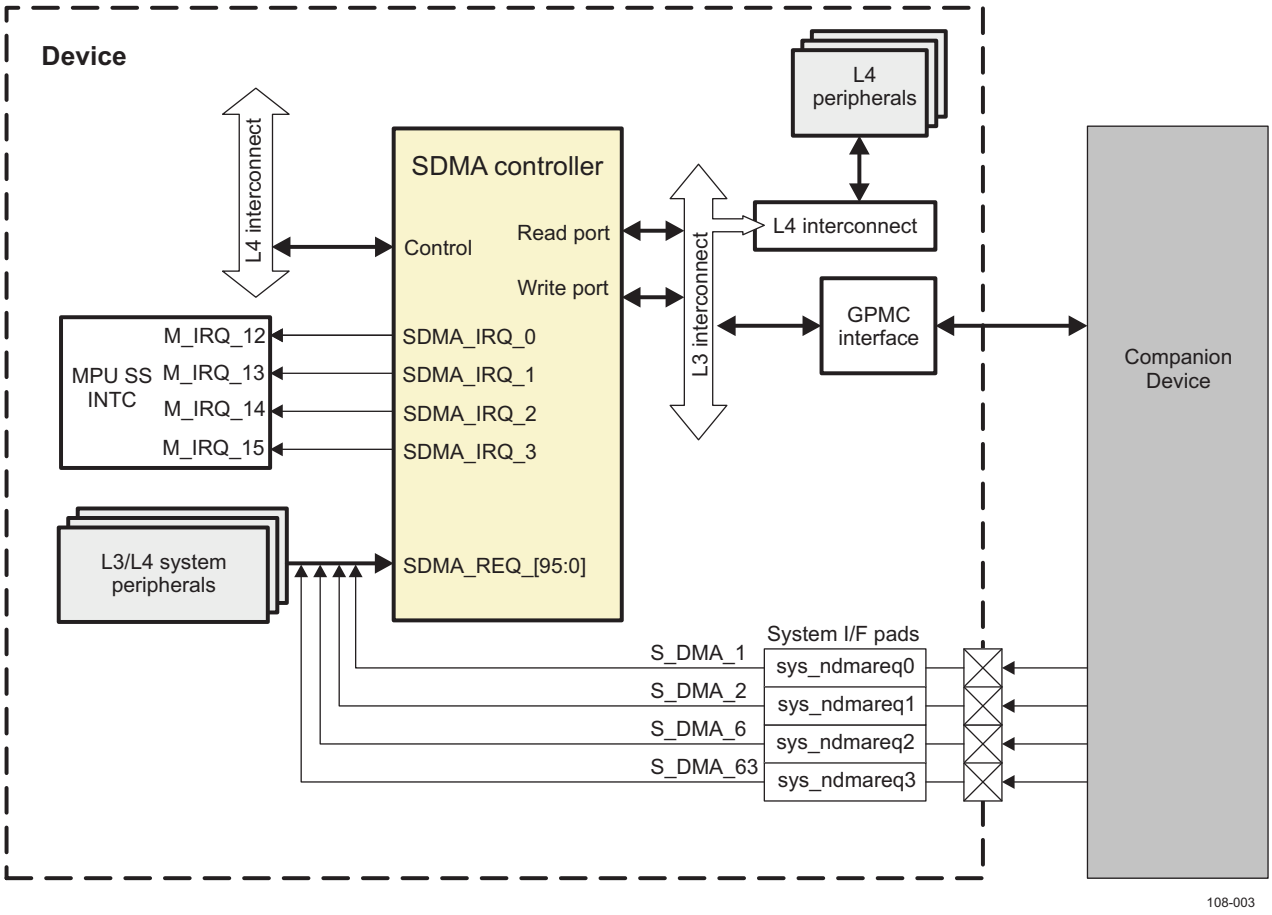
Table 9-1 shows the I/O description of the external DMA request pin.

Table 9-1. Description External DMA Request Pin

Signal Name	IO	Description	Reset
sys_ndmareq0	I	External DMA request to the SDMA controller	Unknown
sys_ndmareq1	I	External DMA request to the SDMA controller	Unknown
sys_ndmareq2	I	External DMA request to the SDMA controller	Unknown
sys_ndmareq3	I	External DMA request to the SDMA controller	Unknown

Figure 9-5 shows the SDMA environment with an example of how to use the external hardware DMA request pins.

Figure 9-5. Example of External DMA Requests Use to the SDMA Controller



For example, an external device can use the external DMA request pins to start a logical channel transfer over the general-purpose memory controller (GPMC) interface. The transfer can be a memory-to-memory transfer in which the source memory is in the external device.

The external DMA request signals are not available on external pins by default after cold reset. See Chapter 7, *System Control Module*, for instructions on multiplexing out the seven signal lines to pins.

9.3.2 Clocking, Reset, and Power-Management Scheme

9.3.2.1 Clocking

The SDMA controller uses two clock domains:

- CORE_L4_ICLK supports the configuration port.
- CORE_L3_ICLK is both a functional clock for all internal logic and for the two master read and write ports.

The SDMA controller supports a software-controlled standby mode with an input clock shutoff. Setting the PRCM.CM_IDLEST1_CORE[2] ST_DMA status bit to 1 allows detection of the SDMA power mode.

For more information about power management and clock idle, see Section 9.4.20, *Power Management*.

9.3.2.2 Resets

9.3.2.2.1 Asynchronous Hardware Reset

The SDMA controller is part of the CORE_RST reset domain.

9.3.2.2.2 Software Reset Through the Configuration Port

The SDMA controller can be reset independently by software through the SDMA.DMA4_OCP_SYSCONFIG[1] SOFTRESET bit. Setting the bit to 1 enables an active software reset that is functionally equivalent to a hardware reset.

Hardware and software resets initialize all of the logic in the SDMA module and the global registers and some of the per-channel registers (such as the enable bit hardware request line number and link bit field) implemented in flip-flops. However, all remaining per-channel registers are memory-based and, therefore, are not reset.

9.3.2.3 Power Domain

The SDMA controller is part of the CORE power domain.

9.3.3 Hardware Requests

9.3.3.1 Interrupts to the MPU Subsystem

DMA4 has four interrupt lines, numbered Lj with j=0..3. Each logical channel can request an interrupt over any line. The attachment of a channel interrupt event to one of these four external lines is programmable. The software determines whether it attaches a channel interrupt to a single IRQ line or to multiple IRQ lines.

There are two different registers per interrupt line:

- SDMA.DMA4_IRQSTATUS_Lj CH_31_0_Lj field shows the status of the different sources of interrupt. If the SDMA.DMA4_IRQENABLE_Lj bit *i* is 1, the channel *i* is the source of interrupt in line *j*. In contrast to the SDMA.DMA4_CSRi registers, the SDMA.DMA4_IRQSTATUS_Lj registers are updated regardless of the corresponding bits in the SDMA.DMA4_IRQENABLE_Lj registers.
- SDMA.DMA4_IRQENABLE_Lj CH_31_0_Lj_EN field masks/unmasks the channel interrupt. If the SDMA.DMA4_IRQENABLE_Lj bit *i* is set to 0, the channel interrupt *i* of the line *j* is masked.

Each logical channel can generate 10 different interrupt events when enabled (that is, set to 1) in the SDMA.DMA4_CICRi register. Each status bit is updated in the SDMA.DMA4_CSRi register only when the corresponding enable bit is enabled in the SDMA.DMA4_CICRi register.

To determine an interrupt source when an interrupt rises on an interrupt line L*j*, you must:

- Identify the channel (LCH*i*) generating the interrupt.

  Read the SDMA.DMA4_IRQSTATUS_Lj.LCH*i* (LCH0 to LCH31). If LCH*i* = 1, channel *i* is the originator of the interrupt.
- Identify the interrupt event.

  Read the LCH*i* SDMA.DMA4_CSRi. For example, if the drop event (the SDMA.DMA4_CSRi[1] DROP bit) is 1, there will be a request collision.

  The interrupt event status bit in the SDMA.DMA4_CSRi register is immediately reset after it is written to 1.

  The interrupt status bit in the SDMA.DMA4_IRQSTATUS_Lj register is cleared after it is written to 1.

Table 9-2 shows how the SDMA module interrupt lines are connected to the interrupt lines of the MPU interrupt controller.

Table 9-2. SDMA Interrupt Mapping to the MPU Subsystem

IRQ	Source	Description
M_IRQ_12	SDMA_IRQ_0	SDMA interrupt request 0
M_IRQ_13	SDMA_IRQ_1	SDMA interrupt request 1
M_IRQ_14	SDMA_IRQ_2	SDMA interrupt request 2
M_IRQ_15	SDMA_IRQ_3	SDMA interrupt request 3

9.3.3.2 DMA Requests to the SDMA Controller

This section gives information about all modules and features in the high-tier device. See Section 1.5, *OMAP34xx Family*, to check availability of modules and features. Ensure that DMA requests of unavailable modules and features are masked in DMA subsystems.

All peripherals internal to the device use the transition-sensitive scheme for DMA requests. For more information on the transition-sensitive scheme, see Section 9.2.2, *SDMA Request Scheme*. Table 9-3 lists the SDMA request mapping.

Table 9-3. SDMA Request Mapping

DMA Request Line	Source	Description
S_DMA_0	Reserved	Reserved
S_DMA_1	SYS_DMA_REQ0	External DMA request 0 (system expansion)
S_DMA_2	SYS_DMA_REQ1	External DMA request 1 (system expansion)
S_DMA_3	GPMC_DMA	GPMC request from prefetch engine
S_DMA_4	Reserved	Reserved
S_DMA_5	DSS_LINE_TRIGGER	Display subsystem—frame update request
S_DMA_6	SYS_DMA_REQ2	External DMA request 2 (system expansion)
S_DMA_7	Reserved	Reserved
S_DMA_8	Reserved	Reserved
S_DMA_9	Reserved	Reserved
S_DMA_10	Reserved	Reserved
S_DMA_11	Reserved	Reserved
S_DMA_12	Reserved	Reserved
S_DMA_13	Reserved	Reserved
S_DMA_14	SPI3_DMA_TX0	McSPI module 3—transmit request channel 0
S_DMA_15	SPI3_DMA_RX0	McSPI module 3—receive request channel 0
S_DMA_16	MCBSP3_DMA_TX	MCBSP module 3—transmit request
S_DMA_17	MCBSP3_DMA_RX	MCBSP module 3—receive request
S_DMA_18	MCBSP4_DMA_TX	MCBSP module 4—transmit request
S_DMA_19	MCBSP4_DMA_RX	MCBSP module 4—receive request
S_DMA_20	MCBSP5_DMA_TX	MCBSP module 5—transmit request
S_DMA_21	MCBSP5_DMA_RX	MCBSP module 5—receive request
S_DMA_22	SPI3_DMA_TX1	McSPI module 3—transmit request channel 1
S_DMA_23	SPI3_DMA_RX1	McSPI module 3—receive request channel 1
S_DMA_24	I2C3_DMA_TX	I^2C module 3—transmit request
S_DMA_25	I2C3_DMA_RX	I^2C module 3—receive request
S_DMA_26	I2C1_DMA_TX	I^2C module 1—transmit request
S_DMA_27	I2C1_DMA_RX	I^2C module 1—receive request
S_DMA_28	I2C2_DMA_TX	I^2C module 2—transmit request
S_DMA_29	I2C2_DMA_RX	I^2C module 2—receive request
S_DMA_30	MCBSP1_DMA_TX	MCBSP module 1—transmit request

Table 9-3. SDMA Request Mapping (continued)

DMA Request Line	Source	Description
S_DMA_31	MCBSP1_DMA_RX	MCBSP module 1—receive request
S_DMA_32	MCBSP2_DMA_TX	MCBSP module 2—transmit request
S_DMA_33	MCBSP2_DMA_RX	MCBSP module 2—receive request
S_DMA_34	SPI1_DMA_TX0	McSPI module 1—transmit request channel 0
S_DMA_35	SPI1_DMA_RX0	McSPI module 1—receive request channel 0
S_DMA_36	SPI1_DMA_TX1	McSPI module 1—transmit request channel 1
S_DMA_37	SPI1_DMA_RX1	McSPI module 1—receive request channel 1
S_DMA_38	SPI1_DMA_TX2	McSPI module 1—transmit request channel 2
S_DMA_39	SPI1_DMA_RX2	McSPI module 1—receive request channel 2
S_DMA_40	SPI1_DMA_TX3	McSPI module 1—transmit request channel 3
S_DMA_41	SPI1_DMA_RX3	McSPI module 1—receive request channel 3
S_DMA_42	SPI2_DMA_TX0	McSPI module 2—transmit request channel 0
S_DMA_43	SPI2_DMA_RX0	McSPI module 2—receive request channel 0
S_DMA_44	SPI2_DMA_TX1	McSPI module 2—transmit request channel 1
S_DMA_45	SPI2_DMA_RX1	McSPI module 2—receive request channel 1
S_DMA_46	MMC2_DMA_TX	MMC/SD2 transmit request
S_DMA_47	MMC2_DMA_RX	MMC/SD2 receive request
S_DMA_48	UART1_DMA_TX	UART module 1—transmit request
S_DMA_49	UART1_DMA_RX	UART module 1—receive request
S_DMA_50	UART2_DMA_TX	UART module 2—transmit request
S_DMA_51	UART2_DMA_RX	UART module 2—receive request
S_DMA_52	UART3_DMA_TX	UART module 3—transmit request
S_DMA_53	UART3_DMA_RX	UART module 3—receive request
S_DMA_54	Reserved	Reserved
S_DMA_55	Reserved	Reserved
S_DMA_56	Reserved	Reserved
S_DMA_57	Reserved	Reserved
S_DMA_58	Reserved	Reserved
S_DMA_59	Reserved	Reserved
S_DMA_60	MMC1_DMA_TX	MMC/SD1 transmit request
S_DMA_61	MMC1_DMA_RX	MMC/SD1 receive request
S_DMA_62	Reserved	Reserved
S_DMA_63	SYS_DMA_REQ3	External DMA request 3 (system expansion)
S_DMA_64	Reserved	Reserved
S_DMA_65	Reserved	Reserved
S_DMA_66	Reserved	Reserved
S_DMA_67	Reserved	Reserved
S_DMA_68	Reserved	Reserved
S_DMA_69	SPI4_DMA_TX0	McSPI module 4—transmit request channel 0
S_DMA_70	SPI4_DMA_RX0	McSPI module 4—receive request channel 0
S_DMA_71	DSS_DMA0	Display subsystem DMA request 0 (DSI)
S_DMA_72	DSS_DMA1	Display subsystem DMA request 1 (DSI)
S_DMA_73	DSS_DMA2	Display subsystem DMA request 2 (DSI)
S_DMA_74	DSS_DMA3	Display subsystem DMA request 3 (DSI or RFBI)
S_DMA_75	Reserved	Reserved

Table 9-3. SDMA Request Mapping (continued)

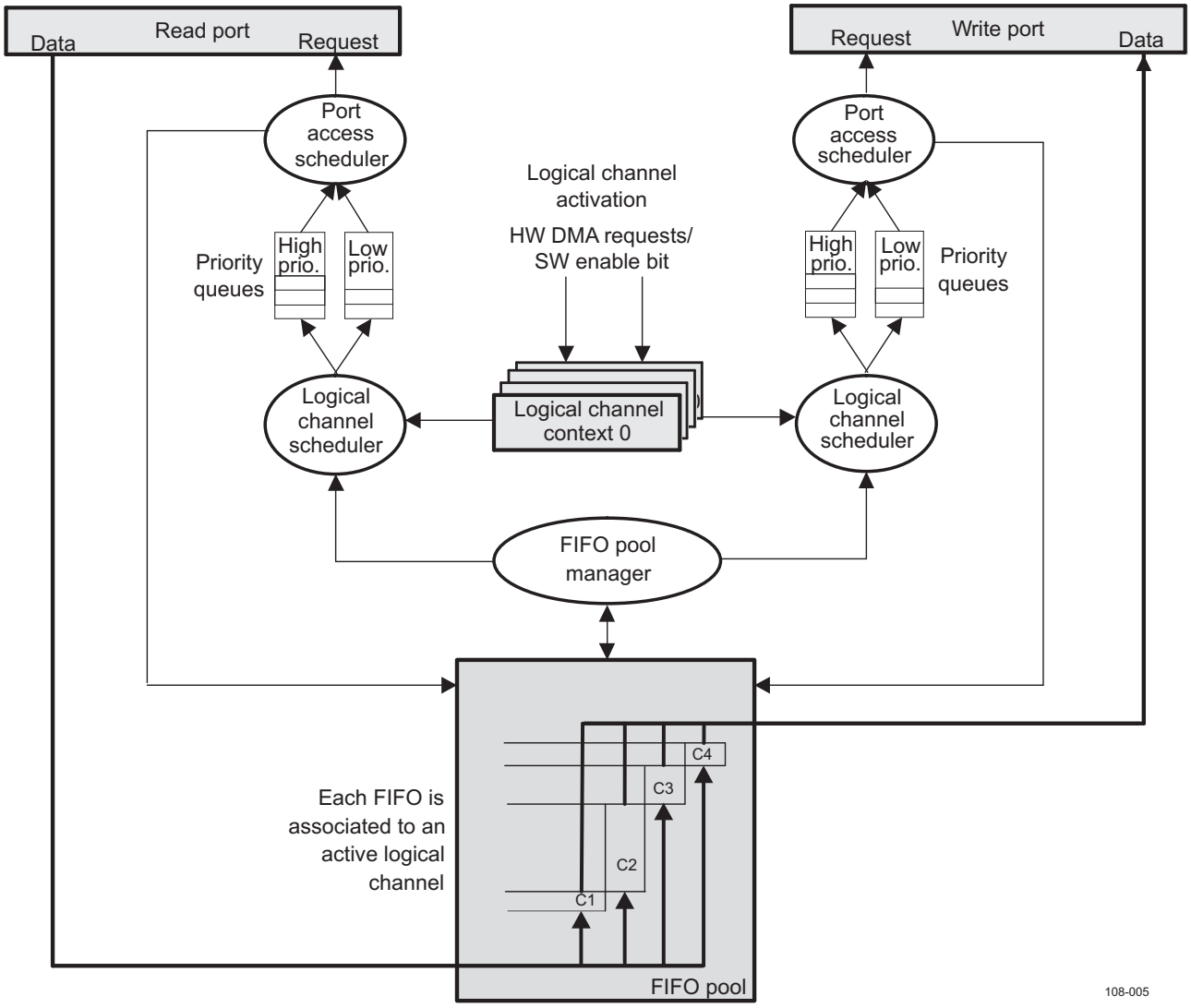
DMA Request Line	Source	Description
S_DMA_76	MMC3_DMA_TX	MMC/SD3 transmit request
S_DMA_77	MMC3_DMA_RX	MMC/SD3 receive request
S_DMA_80 ... S_DMA_96	Reserved	Reserved

9.4 SDMA Functional Description

The SDMA module provides high-performance data transfers between memories and peripheral devices with low processor use. A DMA transfer is programmed through a logical DMA channel, which allows the transfer to be optimally tailored to the requirements of the application.

Figure 9-6 shows the SDMA controller top-level block diagram.

Figure 9-6. SDMA Controller Top-Level Block Diagram



9.4.1 Logical Channel Transfer Overview

As Figure 9-6 shows, the SDMA module has one read port and one write port operating independently of each other. Buffering is provided between the read and write ports through a FIFO queue memory pool that is shared dynamically between the active logical channels.

- Logical channel synchronization

  A logical channel is described as hardware-synchronized when the DMA transfers are triggered by DMA requests from a hardware device. Alternatively, a logical channel is described as nonsynchronized when the DMA transfer is triggered by software.
- Logical channel activation

  A logical channel becomes active as follows:

  – For hardware-synchronized transfers, when the logical channel is enabled and the hardware DMA

request line is asserted
  - For software-triggered (nonsynchronized) transfers, as soon as software enables the logical channel

- Logical channel transfer composition

  A DMA transfer is divided automatically into a number of transactions. Depending on the logical channel context configured, transfer size, the start address alignment, the addressing mode, and the configured maximum burst size, each transaction can be either a single access or a burst of accesses.

- Logical channel scheduling

  When several logical channels are active at the same time, schedulers manage the read and write ports. The scheduling of logical channel transfers is similar for both read and write ports. When a logical channel becomes active, it is added to the tail of a scheduling queue. If more than one logical channel becomes active at the same time, the one with the lower number is queued first. This mechanism provides a first-come, first-serve scheduling scheme between the concurrently active logical channels.

  In addition, each read and write port has a high-priority queue and a low-priority queue. The priority bit in the logical channel SDMA.DMA4_CCRi register determines if a logical channel is queued as high or low priority. The relative weighting of the scheduling of the high-priority queue to the low priority queue is programmable from 1:1 to 1:256 through the DMA global channel register (SDMA.DMA4_GCR).

Note: The SDMA.DMA4_GCR[23:16] ARBITRATION_RATE field is not dependent on the SDMA.DMA4_GCR[13:12] HI_THREAD_RESERVED field. The ARBITRATION_ RATE field is dependent on the SDMA.DMA4_CCRi[26] WRITE_PRIORITY bit and the SDMA.DMA4_CCRi[6] READ_PRIORITY bit.

- Read/write port access scheduling policy

  When either the read or write port becomes available, the port access scheduler selects the next logical channel for which to perform a DMA transaction from either the high- or low-priority queue.

  When the current DMA transaction (single or burst access) is complete and the full DMA transfer is not finished, the logical channel returns to the tail of the queue. Because the port access scheduling is on a per-transaction basis, a logical channel can be queued repeatedly this way several times during a single transfer.

The SDMA module can have up to four outstanding read transactions and two outstanding write transactions in the system interconnect; four read and two write thread IDs exist. For an arbitration cycle to occur, these two conditions must be met:

- At least one channel is requesting.
- At least one free thread ID is available.

On an arbitration cycle, the scheduler grants the highest priority channel that has an active request, allocates the thread ID, and tags this thread as busy. At a given time, a channel cannot be allocated for more than one thread ID.

Note: If more than one channel is active, each channel is given a thread ID for the current service only, not for the whole channel transfer.

When only one channel is active, one thread ID is allocated during the channel transfer. The access can be up to a 16 x 32-bit access (that is, 16 32-bit single access, four bursts of 16bytes, two bursts of 32bytes, or one burst of 64bytes) without rescheduling the channel at the end of each burst transfer.

When nonburst alignment is at the beginning of the transfer, the channel is rescheduled for each smaller access until burst-aligned. When the end of the transfer is not burst-aligned, the channel is rescheduled for each of the remaining smaller accesses.

For a logical channel transfer completion, when the last access is written to the destination, the logical channel becomes inactive. If enabled, an interrupt request is generated (see Section 9.4.12, *Interrupt Generation*).

9.4.2 FIFO Queue Memory Pool

A FIFO queue memory pool provides buffering between the read and write ports. The hardware allocates the space dynamically to a number of FIFO queues, and each queue is associated with an active logical channel.

To avoid a memory pool overflow, if there are fewer entries in the FIFO queue memory pool than are required for the maximum configured source burst size of the next logical channel to be scheduled, the logical channel is returned to the tail of the queue, and the port access scheduler continues to search the queue until it finds a logical channel that can be scheduled.

The maximum FIFO depth that can be allocated to each individual logical channel can be limited globally through the SDMA.DMA4_GCR register. This value should be configured to allow a fair allocation of the memory pool between the active channels.

A logical channel is scheduled if it has not yet reached its allocation limit, even if the access to be performed will exceed this limit. This means that the effective number of entries used by a particular logical channel is limited to the configured maximum entries per channel + channel maximum configured burst size (in words) -1.

9.4.3 Addressing Modes

A DMA transfer block consists of a number of frames (FN). Each frame consists of a number of elements, and each element can have a size of 8, 16, or 32 bits, as follows:

transfer block size = number of frames x number of elements per frame x element size

The FN, number of elements per frame (EN), and size of elements are common for both the source and destination. However, the way in which the data is represented (addressing profile/mode) is independently programmable for the source and destination devices, using one of these four addressing modes:

- Constant: The address remains the same for consecutive element accesses.
- Post-increment: The address increases by the element size (ES), even across consecutive frames.
- Single-index: The address increases by the ES, plus the element index (EI) value minus one (even across consecutive frames).
- Double-index: The address increases by the ES, plus the EI value minus one within a frame. When a full frame is transferred, the address increases by the ES plus the frame index (FI) value minus 1.

The ES, EI, and FI values are expressed in bytes. The EI and FI values can be positive or negative.

When calculating the EI and FI values, it is critical to note that, after an element is accessed, the logical channel address pointer equals the address of the last byte (highest address) of the accessed element. The correct value for the EI or FI should be such that, when added to the logical channel address pointer, results in the address of the first byte (lowest address) of the next element to be accessed.

The EI and FI values must be configured so that the address of each element in the transfer is aligned on an ES boundary.

Consequently, the single-index addressing mode with EI = 1 or double-index addressing mode with EI = 1 and FI = 1 is equivalent to post-increment addressing.

Note: The source and destination start addresses must also be aligned on an ES boundary.

When the address of an element to be accessed is not aligned on an ES boundary, the transfer is stopped and an address error interrupt occurs, if enabled (see Section 9.4.12, *Interrupt Generation*).

The SDMA.DMA4_CFNi register configures the FN in a block.

The SDMA.DMA4_CENi register configures the EN.

The SDMA.DMA4_CSDPi register configures the ES.

The SDMA.DMA4_CSSAi and SDMA.DMA4_CDSAi registers configure the source and destination start addresses.

The SDMA.DMA4_CCRi register configures the source and destination addressing modes.

The SDMA.DMA4_CSEi, SDMA.DMA4_CSFi, SDMA.DMA4_CDEi, and SDMA.DMA4_CDFi registers configure the source EI, source FI, destination EI, and destination FI, respectively.

The addressing profiles are expressed as equations as follows:

Equation 1. Constant addressing:

$$A(n+1) = A(n)$$

Equation 2. Post-increment addressing:

$$A(n+1) = A(n) + ES$$

Equation 3. Single-indexed addressing:

$$A(n+1) = A(n) + ES + (EI - 1)$$

Equation 4. Double-indexed addressing:

When not at the end of a frame or transfer (that is, when the element counter $\neq 0$):

$$A(n+1) = A(n) + ES + (EI - 1)$$

When at the end of a frame but not at the end of the transfer (that is, when the element counter = 0 and the frame counter $\neq 0$):

$$A(n+1) = A(n) + ES + (FI - 1)$$

Calculate the element and frame index as follows:

Equation 5. Element index

$$EI = [(Stride\ EI - 1) * ES] + 1$$

Equation 6. Frame index

$$FI = [(Stride\ FI - 1) * ES] + 1$$

where:

A(n): Byte address of the element n within the transfer.

ES is in bytes, ESε{1, 2, 4}.

EI is in bytes, specified in a configuration register, $-32768 \leq EI \leq 32767$.

Stride EI: The difference in the number of elements between the start of the current element, *n*, to the start of next element, n+1.

Element counter: A counter that is (re)initiated with the number of elements per frame or per transfer. Decreased by 1 for each element transferred. The initial value is configured in the register DMA channel element number, SDMA.DMA4_CENi.

F is in bytes, specified in a configuration register, $-2147483648 \leq FI \leq 2147483647$.

Stride FI: The difference in the number of elements between the start of the last element of the current frame and the beginning of the first element of the next frame.

Frame counter: A counter that is (re)initiated with the FN per transfer. Decreased by 1 for each frame transferred. The initial value is configured in the register DMA channel frame number, SDMA.DMA4_CFNi.

Figure 9-7 shows how a stride EI and FI are defined. When handling complex configurations, using strides can make it easier to calculate EI and FI because you can calculate in elements instead of bytes. (This approach is used in the 90Degrees clockwise image rotation example shown in Figure 9-11.) The double-index addressing example shown in Figure 9-7 has ES = 4, EN = 2, EI = 5, FI = 5, and FN = 2.

Figure 9-7 through Figure 9-10 show examples of addressing mode configurations. Table 9-4 lists parameter values for the examples.

Figure 9-7. Example Showing Double-Index Addressing, Elements, Frames, and Strides

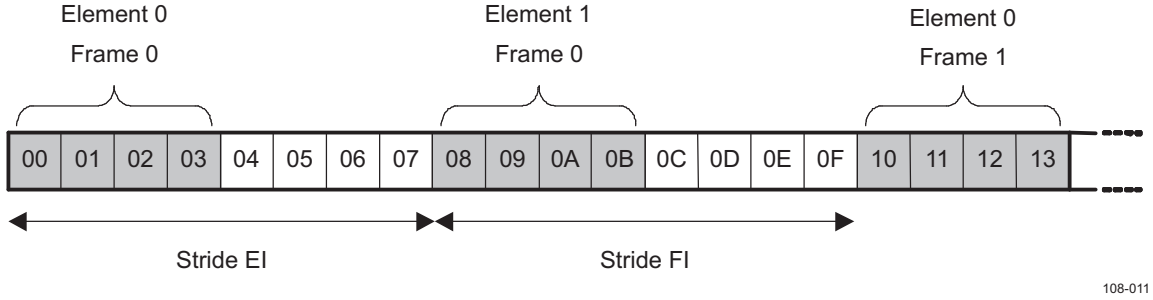


Figure 9-8. Addressing Mode Example (a)

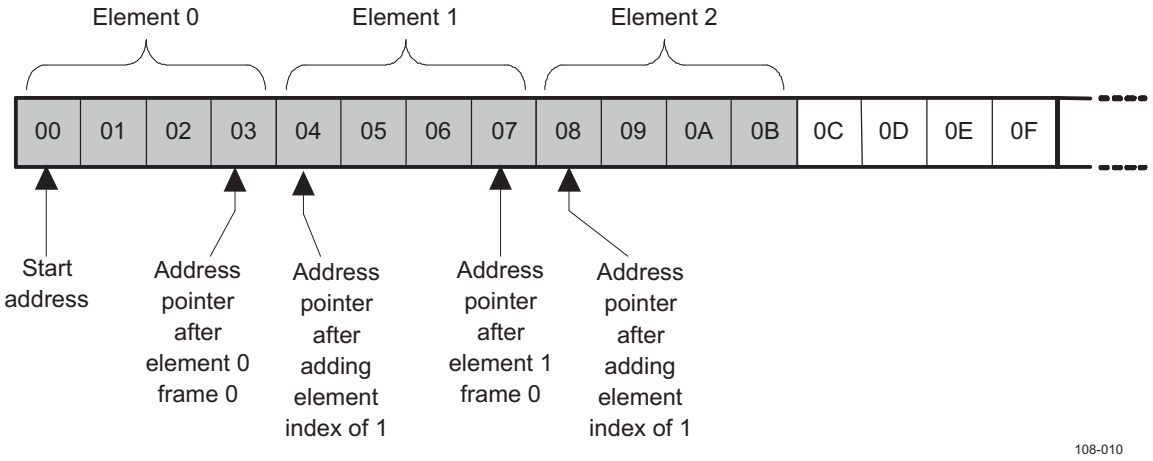


Figure 9-9. Addressing Mode Example (b)

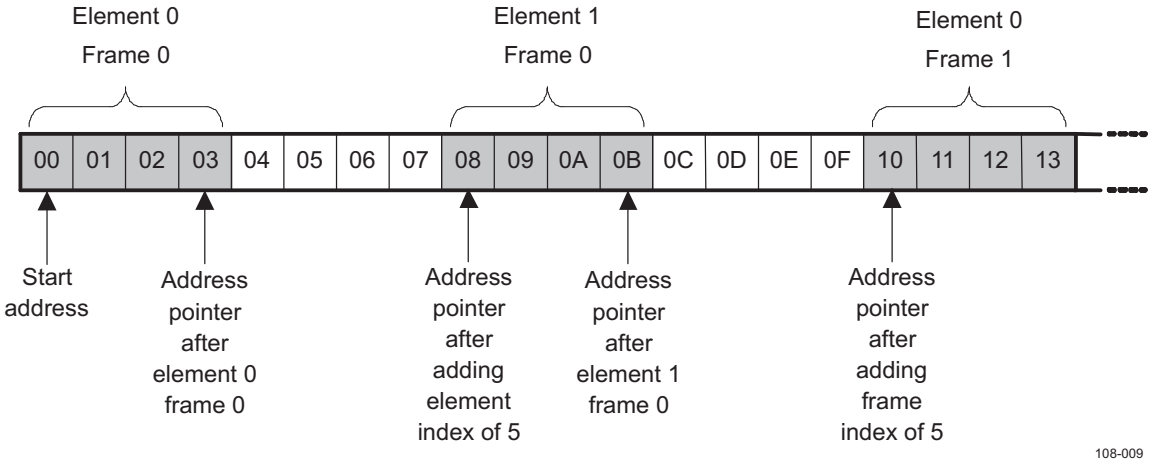


Figure 9-10. Addressing Mode Example (c)

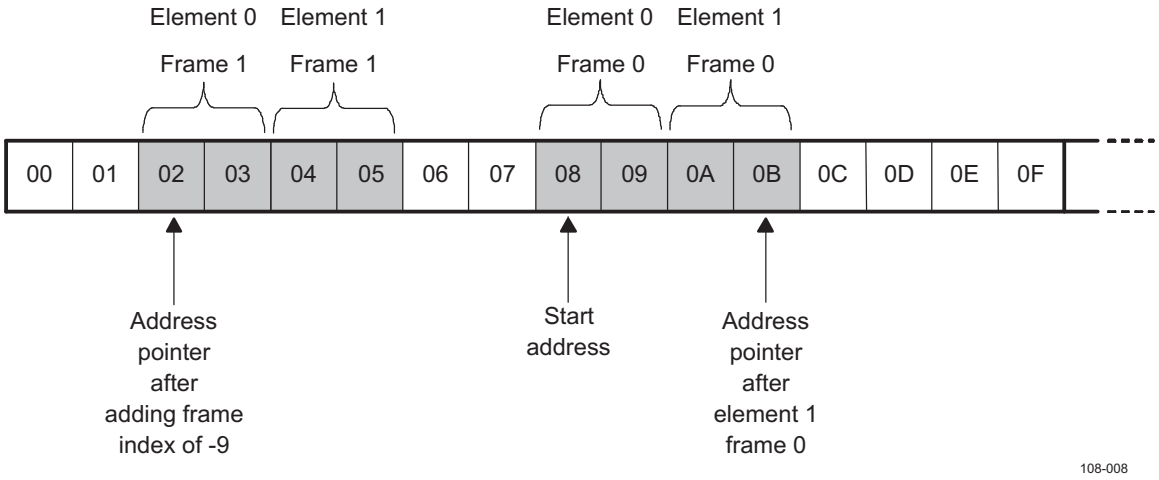


Table 9-4. Parameter Values for Addressing Mode Examples (a), (b), and (c)

Parameter	Example (a)	Example (b)	Example (c)
Addressing mode	Single index (or post-increment)	Double index	Double index
Start address	0	0	8
ES	4 (32-bit)	4 (32-bit)	2 (16-bit)
EN	3	2	2
EI	1	5	1
FN	1	2	2
Frame index	N/A	5	-9

Double indexing can occur either on source (read) or destination (write). Equations for rotation of xx degrees on destination are obtained by taking equations for rotation of (360-xx) degrees on source, and swapping x and y in them. The opposite is also true. Table 9-5 list the equations for rotation for 90, 180 and 270°.

Table 9-5. Equations for Rotation

		90° Rotation	180° Rotation	270° Rotation
Double indexing on destination (write)	Base address	ES*(y-1)	ES*(x*y-1)	ES*y*(x-1)
	Element index (EI)	ES*(y-1)+1	1-2*ES	1-ES*(y+1)
	Frame index (FI)	1 – ES*[(x-1)*y+2]	1-2*ES	1+ES*(x-1)*y
Double indexing on source (read)	Base address	ES*x*(y-1)	ES*(x*y-1)	ES*(x-1)
	Element index (EI)	1-ES*(x+1)	1-2*ES	ES*(x-1)+1
	Frame index (FI)	1+ES*(y-1)*x	1-2*ES	1 – ES*[(y-1)*x+2]

Table 9-6 and Figure 9-11 show the configuration required to perform a 90° clockwise image rotation of a 240 x 160 pixel, 32-bit image. The EI, frame size, and FI values are configured so that the image is rotated line by line starting at the left-hand end of the top line.

Note: The FI value for the destination is negative so that the first pixel of each subsequent line of the source image is written to the correct location at the destination.

Equation 5 and Equation 6 calculate the destination FI and EI. The example assumes that the image lines are stored at consecutive addresses in memory, meaning that both EI and FI on the source side are 1.

Rotations:

Section 9.5.7, *90° Clockwise Image Rotation*, describes how to program this example.

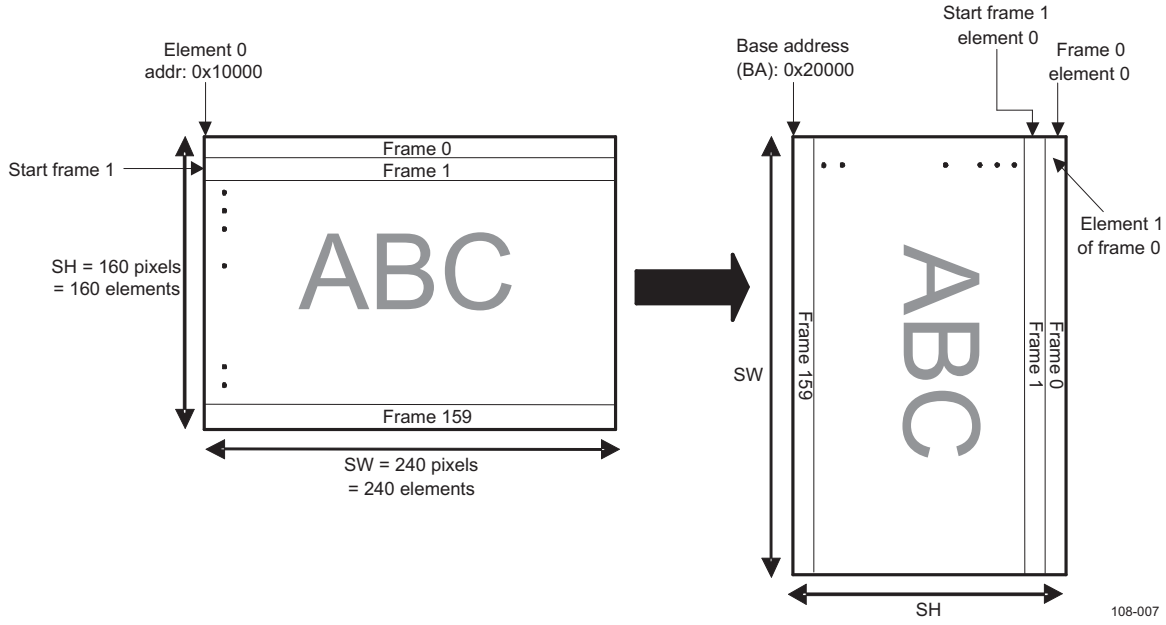
Observe that:

- One pixel = one element
- One line = one DMA frame
- Pixel size = element size = ES

Table 9-6. Example Parameter Values for a 90° Clockwise Image Rotation

Parameter	Source Value	Destination Value
Bits per pixel	32	32
ES	4	4
Image width	SW	SH
Image height	SH	SW
Stride elements (stride EI)	1 element	SH
Stride frames (stride FI)	1 element	-[(SW-1)*SH+1] = –38241 elements
Start address	0x100000	0x200000 + (SH – 1) x ES = 0x20027C
EN	SW	SW
EI	[(Stride EI – 1) * ES] + 1 = 1	[(Stride EI – 1) * ES] + 1 = 637
FN	SH	SH
FI	[(Stride FI – 1) * ES] + 1= 1	[(Stride FI – 1) * ES] + 1 = –152967

Figure 9-11. Example of a 90° Clockwise Image Rotation



9.4.4 Packed Accesses

When the logical channel ES is less than the DMA module read/write port size, and the addressing profile supports it (that is, post-increment mode or single- or double-index mode with EI = 1), the number of elements to transfer in each read/write port access may be maximized by specifying that the source or destination is packed through the channel SDMA.DMA4_CSDPi register. Thus:

- For a read/write port size of 32 bits, the source or destination can be configured as packed for transfer ESs of 8 bits (four elements per access) and 16 bits (two elements per access).

- For a read/write port size of 64 bits, the source or destination can be configured as packed for transfer ESs of 8 bits (eight elements per access), 16 bits (four elements per access), and 32 bits (two elements per access).

Depending on the start address and transfer length, the first or last packed access might be only partially filled. This is indicated to the source or destination using the byte-enable signals.

Note: When a constant addressing mode is specified, only nonpacked accesses are used, and specifying the accesses as packed has no effect.

9.4.5 Burst Transactions

Transfer performance can be improved, where the source or destination and addressing profile supports it, by configuring the logical channel to perform burst transactions consisting of multiple instead of single accesses. The channel can be programmed to use burst sizes equivalent to 16, 32, or 64 bytes through the SDMA.DMA4_CSDPi register, with the read burst size being programmable independently of the write burst size. Typically, the optimal burst size is 64 bytes (16 accesses for a 32-bit read/write port size or 8 accesses for a 64-bit read/write port size).

To obtain the maximum benefit from burst transactions, the source and destination start addresses should be aligned with the burst size. If this is not the case, the start of the transfer can consist of a number of smaller (single or burst) transactions until the first burst size boundary is reached.

Similarly, if the end of the transfer is not aligned on a burst size boundary, the final part of the transfer can consist of a number of smaller transactions.

Note: Except in the constant addressing mode, the source or destination must be specified as packed for burst transactions to occur.

9.4.6 Endianism Conversion

The source and destination are each specified as little-endian or big-endian through the SDMA.DMA4_CSDPi register for the particular logical channel. If the endianism of the source and destination differ, and the logical channel ES is less than the SDMA module read/write port size, an endianism conversion is applied to the data before it is written to the destination.

When transferring data between a source and a destination with different endianism, it is important to specify an ES that is equal to the type of data being transferred to preserve the correct data image at the destination.

In the system, endianism conversion can be performed in more than one place. It is possible to inform the source and/or destination to lock the endianism (that is, to not perform a conversion) through the logical DMA channel SDMA.DMA4_CSDPi register.

9.4.7 Transfer Synchronization

A logical channel can be programmed for either software-triggered or hardware synchronized transfers.

9.4.7.1 Software Synchronization

A transfer is software-triggered when the logical channel is set up and started by software. To specify a software-triggered transfer, set the channel DMA register bits SDMA.DMA4_CCRi[4:0] and SDMA.DMA4_CCRi[20:19] to 0. The transfer starts as soon as the DMA register bit SDMA.DMA4_CCRi[7] is set (that is, enters the scheduling process).

9.4.7.2 Hardware Synchronization

A transfer is hardware-synchronized if the logical channel activation is driven by hardware requests from either the source or destination target. A hardware synchronized transfer is specified by configuring the DMA request line number in the channel SDMA.DMA4_CCRi register to a value that corresponds to the DMA request line from the source or destination that generates the DMA requests. The DMA request numbers to be configured are specified in the DMA request mapping (see Table 9-8).

Specify the DMA request number in the channel DMA register bits SDMA.DMA4_CCRi[4:0] and SDMA.DMA4_CCRi[20:19]. After the DMA register bit SDMA.DMA4_CCRi[7] is set, the logical channel becomes enabled but not activated (that is, it does not enter the scheduling process), which means that channel registers are not updated until the first DMA request is received.

Note: DMA Request Line

A DMA request line must not be shared between concurrently enabled DMA channels. However, a DMA request line can be shared between several chained logical channels.

The channel synchronization control registers are 1-based. For example, to enable the S_DMA_1 request, SDMA.DMA4_CCRi[4:0] SYNCHRO_CONTROL must be set to 0x2 (DMA request number + 1).

For hardware synchronization, the amount of data to be transferred for each assertion of the DMA request line is configured through the frame synchronization (FS) and block synchronization (BS) bits in the logical channel SDMA.DMA4_CCRi register and the DMA register bits SDMA.DMA4_CCRi[5] and SDMA.DMA4_CCRi[18], respectively.

The amount of data can be any of the following:

- A single element transfer: A complete element defined by Data_type. For example, 8/16/32 bits are transferred in response to a DMA request.
- A full frame: A complete frame of several elements is transferred in response to a DMA request.
- A full block (a full channel transfer): A complete block of several frames is transferred in response to a DMA request.
- A full packet (a full channel transfer): A complete packet of several elements is transferred in response to a DMA request.

Packets allow the size of each part of the full DMA transfer to be configured independent of the organization of the data to be transferred (typically a number of elements). This can be useful when the source or destination has a buffer (such as a FIFO queue) with a size unrelated to the frame size of the transfer. The packet size then can be set to the size of the buffer.

Packet transfer must be used only where the source or destination is addressed in constant addressing mode, because FI registers are reused to specify size of the packet.

To support the burst mode, the logical channel must also be configured to use the source-port packed access mode. The packed address mode cannot be used with a constant address mode: it must be configured to use a post-increment address mode.

The packet size is configured based on the source/destination synchronization select bit in the SDMA.DMA4_CCRi register through either the channel SDMA.DMA4_CDFi register (source synchronized) or the SDMA.DMA4_CSFi register (destination synchronized).

When the logical channel transfer block is not an exact multiple of the packet size, the final packet consists of the remaining elements in the transfer, using burst or single accesses to complete the block transfer.

The maximum transfer size, regardless of the packet size, is always as follows:

Block_size = Number_of_Frame_in_Block * Number_of_Element_in_Frame * Element_Size

- Synchronized at the source

The DMA module optimizes the transfer with respect to the number and size of burst transactions for the given source and destination addressing profiles and configured maximum burst sizes. When writing to the destination is slower than reading from the source, data is buffered in the channel FIFO queue. If the transfer is packet-synchronized at the source, the end-of-packet interrupt is disabled (see Section 9.4.11, *Reprogramming an Active Channel*).

For a source synchronized transfer, buffering can be enabled or disabled by setting the SDMA.DMA4_CCRi[25] BUFFERING_DISABLE bit. For a packet source synchronization with buffering disabled and the packed/burst across the packet boundary, the last packed/burst write transaction is split in optimized smaller accesses to complete the packet transfer size. However, for a packet source synchronized transfer with buffering enabled and with the packed/burst across the packet boundary, the DMA module waits for the next DMA request(s) to read enough data to issue an atomic packed/burst write transaction (assuming the address is packed/burst aligned).

Note: Regardless of whether buffering is enabled or not, buffering is not performed between frames. If the packed/burst is across the frame boundary, the last packed/burst write transaction is split into optimized smaller accesses to complete the frame transfer size.

- Synchronized at the destination

  The performance of a hardware-synchronized transfer can be improved by using the prefetch mode, enabled through the channel DMA register bit SDMA.DMA4_CCRi[23]. Data is prefetched on the read port side in advance of the DMA request received and buffered in the FIFO queue. Up to a full transfer block can be prefetched, although this can be limited by the specified maximum channel FIFO queue depth (see Section 9.4.2, *FIFO Queue Memory Pool*).

  Buffering disable is not allowed for a destination-synchronized transfer.

Note: Behavior is undefined when prefetch is enabled and a transfer is synchronized to the source.

Whether buffering is enabled or disabled, the last transaction in the frame or in the block is write nonposted (WNP) even if the write mode is specified as write last nonposted (WLNP; the WRITE_MODE bit field of the SDMA.DMA4_CSDPi register = 0x2). However, in a packet synchronization mode, the last transaction of each packet in the transfer is WNP only if buffering disable is on (even if the write mode is specified as WLNP).

Regardless of whether buffering disable is enabled or disabled, the packet interrupt is not generated in the packet source synchronized mode.

CAUTION

The BUFFERING_DISABLE bit field of the SDMA.DMA4_CCRi register must be filled with an allowed value, as specified in Table 9-7.

Table 9-7. Buffering Disable

	BUFFERING_DISABLE (0: buffering enable, 1: buffering disable)	
Destination synchronized	0	Allowed
	1	Not allowed
Source synchronized	0	Allowed
	1	Allowed

Synchronized transfer monitoring using CDAC (SDMA.DMA4_CDACi):

Context is restored only when the channel becomes active on a DMA request (not at software enable). The channel is software-enabled first, and then a DMA request is asserted followed by the first context restore.

The CDAC register is writable; thus, you can initialize the CDAC to monitor the transfer and determine if the transfer is started or not (see Section 9.5.4, *Synchronized Transfer Monitoring Using CDAC*, for more information).

Note: The CDAC register must be written or read so that the least-significant byte (LSByte) is read first; otherwise, the shadow registers do not update the CDAC registers.

This is not an issue for 32-bit read-write transactions. Nevertheless, for 16-bit transactions, start reading or writing the LSByte first to enable the register update.

9.4.8 Thread Budget Allocation

When several concurrent channels are latency critical and hardware synchronized, a specific latency cannot be ensured until the target is served. This situation occurs when the concurrent channel number is superior to the number of available threads.

Note: Four threads are available on the read port, and two threads are available on the write port.

For a hardware-synchronized transfer (memory to peripheral), a minimum bandwidth for a latency-critical transfer must be ensured to avoid collisions between two hardware requests.

Because it is latency critical, the software user is responsible for the following:

- Programming the synchronized channel as a high-priority channel
- Reserving one or several threads for high-priority channels

The proposed implementation is as follows (see Section 9.5.5):

Prevent the regular channel queue from exceeding more than a programmable (3, 2, or 1) number of threads on the read port and no more than one thread on the write port. This number can be set on the global register SDMA.DMA4_GCR[13:12].

The thread reservation is programmable for maximum use of thread resources for concurrent, low-priority channel transfer. Programmability can also allow a partial throughput control by limiting in software the number of concurrent outstanding requests that break the pipelining.

Depending on the SDMA.DMA4_GCR [13:12] value, the following threadID on the read/write ports are allocated for a high-priority channel:

Read port priority thread reservation:

- SDMA.DMA4_GCR[13:12] = 0x0 => No ThreadID is allocated for high-priority channels.
- SDMA.DMA4_GCR[13:12] = 0x1 => Read ThreadID 0 is allocated for high-priority channels.
- SDMA.DMA4_GCR[13:12] = 0x2 => Read ThreadID 0 and Read ThreadID 1 are allocated for high-priority channels.
- SDMA.DMA4_GCR[13:12] = 0x3 => Read ThreadID 0, Read ThreadID 1, and Read ThreadID 2 are allocated for high-priority channels.

Write port priority thread reservation:

- SDMA.DMA4_GCR[13:12] = 0x0 => No ThreadID is allocated for high-priority channels
- SDMA.DMA4_GCR[13:12] = 0x1 => Write ThreadID 0 is allocated for high-priority channels.
- SDMA.DMA4_GCR[13:12] = 0x2 => Write ThreadID 0 is allocated for high-priority channels.
- SDMA.DMA4_GCR[13:12] = 0x3 => Write ThreadID 0 is allocated for high-priority channels.

Regardless whether or not the enabled channels are of high priority, only the setting of the SDMA.DMA4_GCR[13:12] value forces the thread reservation to these values. Set the appropriate value to avoid losing threads using only regular channels.

To have an independent read and write priority context, a per-channel bit SDMA.DMA4_CCRi[26] is added for write priority, and the previous priority bit becomes read priority bit SDMA.DMA4_CCRi[6].

Note: The device has one priority bit per logical channel, not one per port.

9.4.9 FIFO Budget Allocation

To avoid fully occupying the FIFO with a high-priority transfer while low-priority channels wait in the arbitration queue, two separate FIFO budgets are specified: one for high-priority channels and one for low-priority channels. This is defined in the SDMA.DMA4_GCR register, allowing the user to share the FIFO budget between the low- and high-priority channels. The amount of the FIFO allocated by the low- and high-priority channels is fixed by the value set in the SDMA.DMA4_GCR[15:14] HI_LO_FIFO_BUDGET field. The maximum channel FIFO depth is limited by the HI_LO_FIFO_BUDGET field as follows:

If the channel is low priority:

- When HI_LO_FIFO_BUDGET = 0x1, then low priority cannot exceed 75 percent of the total FIFO.
- When HI_LO_FIFO_BUDGET = 0x2, then low priority cannot exceed 25 percent of the total FIFO.
- When HI_LO_FIFO_BUDGET = 0x3, then low priority cannot exceed 50 percent of the total FIFO.

If channel is high priority

- When HI_LO_FIFO_BUDGET = 0x1, then high priority cannot exceed 25 percent of the total FIFO.
- When HI_LO_FIFO_BUDGET = 0x2, then high priority cannot exceed 75 percent of the total FIFO.
- When HI_LO_FIFO_BUDGET = 0x3, then high priority cannot exceed 50 percent of the total FIFO.

The user is responsible for performing the following equation:

- For a high-priority channel: (Per_Channel_Maximum FIFO Depth + 1) x Number of High Channel =< High Budget FIFO
- For a low-priority channel: (Per_Channel_Maximum FIFO Depth + 1) x Number of Low Channel =< Low Budget FIFO

Note: Ensure that *Number of High Channel* means *Number of Active High-Priority Channel* and that *Number of Low Channel* means *Number of Active Low-Priority Channel*.

9.4.10 Chained Logical Channel Transfers

Chaining multiple logical channels permits transfers consisting of multiple parts to be executed without repeated software intervention. This results in better performance than the alternative of software setting up and starting each transfer separately. Each part of a chained transfer can have the data addressed in a different manner that permits the programming of a variety of complex transfers. For example:

- Interlaced video data with one logical channel configured to transfer the even lines and another logical channel configured to transfer the odd lines
- Protocol headers with a separate DMA4 channel configured to transfer each field in the header

Channels can be chained through each channel SDMA.DMA4_CLNK_CTRLi register. When the transfer for the first channel completes, the next channel in the chain is enabled. The number of channels in the chain that are configured for hardware-synchronized transfers is flexible (although typically it might be all, none, or just the first one). The DMA request line number should be set to 0 to specify that any or all of the channels in a chain are software-triggered or nonsynchronized.

The last channel in a chain can be chained to the first channel to create a continuously looping chain. The continuously looping transfer can be stopped on the fly at a specific channel by disabling the SDMA.DMA4_CLNK_CTRLi[15] ENABLE_LNK bit. The looping transfer stops after the specified channel transfer is complete.

Note: DMA Request Line

A DMA request line must not be shared between concurrently enabled DMA channels. However, a DMA request line can be shared between several chained logical channels.

For more information on the programming model, see Section 9.5, *SDMA Basic Programming Model*.

9.4.11 Reprogramming an Active Channel

A currently active logical DMA channel can be disabled through the SDMA.DMA4_CCRi[7] ENABLE bit. When any ongoing transaction is complete and the read-active and write-active bits in the SDMA.DMA4_CCRi register (SDMA.DMA4_CCRi[9] RD_ACTIVE and SDMA.DMA4_CCRi[10] WR_ACTIVE) are reset, the channel can be reprogrammed for a new transfer.

9.4.12 Interrupt Generation

The SDMA module has four interrupt request output lines, SDMA_IRQ_0 to SDMA_IRQ_3. One or more logical channels can be programmed to generate an interrupt request on any of these lines when any one of the maskable DMA events listed in Table 9-8 occurs.

Table 9-8. Logical DMA Channel Events

Event	Description
End of packet	A packet transfer completed.
End of block	A block transfer completed.
End of frame	A frame transfer completed.
Half of frame	Half of the current frame transferred.
Start of last frame	The first element of the last frame transferred.
Transaction error	A transaction error returned by the interconnect in either the read or write port.
Address error	An attempt was made to perform a DMA access to an address not aligned on an ES boundary.
Supervisor transaction error	An error occurred, for example, when an unauthorized initiator (that is not a supervisor) tries to use a supervisor transfer.
Synchronization error	A new DMA request arrived before completion of the transfer because of the previous DMA request.
Drain end	Drain is completed (SDMA.DMA4_CCRi[10] WR_ACTIVE becomes 0).
System transaction error	An error has occurred, for example, when an unauthorized initiator tries to set a normal channel to a System channel or tries to change a System channel to a normal channel or tries to access (write) any register of a System channel registers.
Drop error	A drop event interrupt is generated when a DMA request is being serviced while a second one is asserted and a third one arrives before the second DMA request is serviced.

The logical DMA channels that generate an interrupt on a particular IRQ output are specified through the SDMA.DMA4_IRQENABLE_Lj register (where *j* is the IRQ number: 0, 1, 2, or 3). The events that generate an interrupt for a particular channel can be configured through the channel SDMA.DMA4_CICRi register.

When an interrupt is detected, the logical DMA channel generating the event can first be identified by reading the SDMA.DMA4_IRQSTATUS_Lj register. The event causing the interrupt then can be identified by reading the interrupt status via the relevant DMA channel SDMA.DMA4_CSRi register.

9.4.13 Packet Synchronization

A packet transfer notion is related to the behavior of some peripheral, which have certain buffering capability and requires to transfer the buffer content once an element number threshold is reached (a hardware DMA request is generated). To associate a frame synchronization to each DMA request is possible, but this limits the maximum transfer size. Indeed the maximum transfer size is proportional to the FIFO depth of the peripheral:
maximum_transfer_size =peripheral_FIFO_depth x number_of_frame_in_block.

The packet synchronization allows to dissociate the transfer size from the FIFO depth of the peripheral. Only Constant addressing mode is allowed on RD port or WR port if source target or destination target is packet synchronized respectively.

Example:
Let's consider a camera interface, which have a FIFO_depth of 128 Words and a FIFO_element_number_threshold of 128 and a picture to transfer with a size 320 lines per 240 columns. If frame synchronization is associated to each DMA request then the maximum transfer size that can be performed is $128x2^{16}$ words. In this case, a frame is 128 words long, which does not fit the size of a line. Then it's not possible to generate an interrupt at the end of line. However with introducing the packet transfer notion, which is related to the peripheral FIFO behavior/structure, the maximum transfer size (maximum_transfer_size = 2^{24} x 2^{16} words) is independent of both peripheral_FIFO_depth and FIFO_element_number_threshold. This allows, making an enough long transfer within one channel context and perform rotation operation on big image format.

The main features of DMA Packet transfer are as follows:

- DMA Packet_Data_Size for each DMA Request: typically this will be Peripheral_element_number_threshold Number of elements in a packet shares the SDMA.DMA4_CSFIi and SDMA.DMA4_CDFIi configuration registers. Indeed if the peripheral is the source target, respectively destination target, that means the used addressing mode is constant, consequently SDMA.DMA4_CSFIi[15:0], respectively SDMA.DMA4_CDFI[15:0], is used to specify the packet data size (PKT_ELNT_NBR) and the bit fields [31:16] are unused. To specify the Packet data size in the SDMA.DMA4_CSFIi or SDMA.DMA4_CDFIi, the user must set the SDMA.DMA4_CCRi[24] SEL_SRC_DST_SYNC respectively to 1 or 0.

Note: The packet size can be a sub-multiple or non sub-multiple of a frame size. If DMA Packet_Data_Size is aligned on DMA channel block data size boundary, DMA will transfer the last data in channel block boundary and stop at block boundary for the last packet DMA request. If the Packet_Data_size is not aligned on the block boundary, the remaining data smaller than a packet size are transferred using burst or single accesses to complete the block.

- DMA Packet_Data_Transfer does not affect DMA channel capabilities in term of packing and bursting.

The Packet synchronization mode is active when SDMA.DMA4_CCRi[5] FS = SDMA.DMA4_CCRi[18] BS = 1. Then

- if SDMA.DMA4_CCRi[24] SEL_SRC_DST_SYNC=0; SDMA.DMA4_CDFIi[15:0] gives the number of element in packet and SDMA.DMA4_CDFIi[31:16] is unused for the packet size.

- if SDMA.DMA4_CCRi[24] SEL_SRC_DST_SYNC=1; SDMA.DMA4_CSFIi[15:0] gives the number of element in packet and SDMA.DMA4_CSFIi[31:16] is unused for the packet size.

Note: The maximum transfer size, regardless of the packet size, is always: Block_size = Number_of_Frame_in_Block x Number_of_Element_in_Frame x Element_Size. If the DMA channel packet/burst access is across packet boundary, DMA hardware automatically splits this packing/burst access into multiple smaller accesses, which will be aligned on packet boundary. Otherwise, the DMA transfers data as usual packing/burst access.

9.4.14 Graphics Acceleration Support

The SDMA supports two graphic acceleration features:

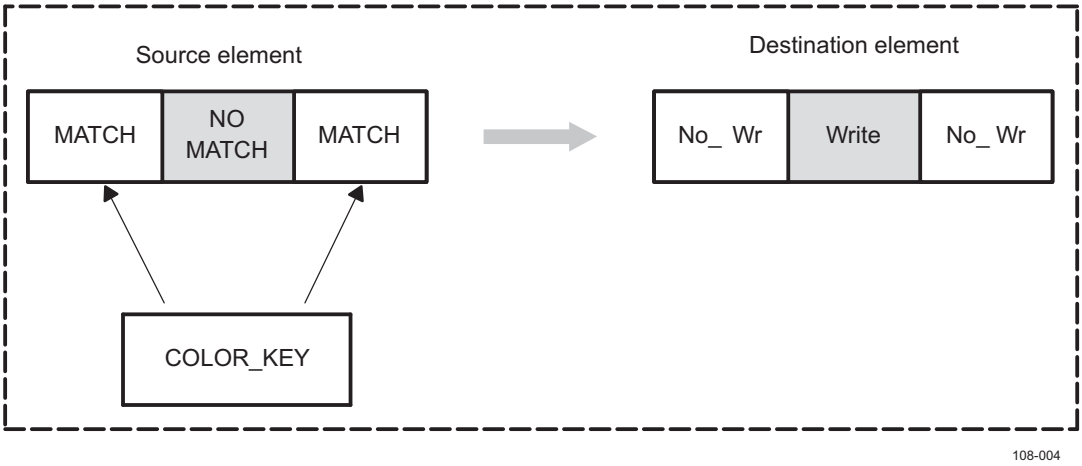
- Transparent copy
- Constant fill

Only one of these features can be enabled at any given time through the SDMA.DMA4_CCRi register for the particular logical DMA channel.

The transparent copy feature enables specification of a particular color through the SDMA.DMA4_COLORi register, so that when it is recognized in the data from the source, it is not copied to the corresponding location in the destination, but instead leaves the data in the corresponding location in the destination as it is.

Figure 9-12 shows the 2-D graphic transparent color block diagram.

Figure 9-12. 2-D Graphic Transparent Color Block Diagram



The constant fill feature provides the ability to specify a particular color through the SDMA.DMA4_COLORi register for every specified location in the destination. In this case, the transfer consists only of writing to the destination without reading from a source.

Both features support 8 bpp (bits per pixel), 16 bpp, and 24 bpp, depending on what is specified as the DMA transfer ES via the SDMA.DMA4_CSDPi register. An ES of 32 bits corresponds to 24 bpp. During a 32-bit (24 bpp) transfer, the most-significant 8 bits ([31:24]) are 0. Both features are compatible with packed and burst transactions.

9.4.15 Supervisor Modes

A logical DMA channel can be configured to operate in supervisor mode through the corresponding bits in the channel SDMA.DMA4_CCRi register. This must be done using supervisor access. Once a channel is configured in supervisor mode, the channel configuration is protected from nonsupervisor accesses. All DMA transactions on a supervisor channel are supervisor transactions.

9.4.16 Posted and Nonposted Writes

A logical channel can be configured in its DMA register bits SDMA.DMA4_CSDPi[17:16] to use one of three write access handshake modes for the destination:

- Nonposted write: Each write must complete before transfer can continue or complete.
- Posted write: Transfer continues without waiting for each write to complete (might improve performance with slow devices).
- Posted with final write nonposted: Transfer continues without waiting for each write to complete, but final write is completed before transfer can complete.

9.4.17 Disabling a Channel During Transfer

When a channel is disabled during a transfer, the channel will undergo an abort, except if the channel was hardware source synchronized with buffering Enabled (SDMA.DMA4_CCRi[25] BUFFERING_DISABLE='0'). In that case, the fifo will be drained in order to avoid losing data. See Section 9.4.18 for details on this feature.

9.4.18 FIFO Draining Mechanism

When a source synchronized channel is disabled during a transfer, then the current hardware request (element/packet/frame/block) service is completed and the channel SDMA.DMA4_CCRi[9] RD_ACTIVE bit is set to 0, which means the channel is not active on the read port. The remaining data in the corresponding disabled channel FIFO is drained onto the write port and transferred to the programmed destination as in normal transfer.

At the end of the draining the SDMA.DMA4_CCRi[10] WR_ACTIVE bit is set to 0 (channel is no more active on the write port) and if the SDMA.DMA4_CICRi[12] DRAIN_END_IE is set to 1, the status bit DMA4_CSRi[12] DRAIN_END is updated and an interrupt is generated.

Once a channel is disabled during a transfer, it needs to wait for SDMA.DMA4_CCRi[9] RD_ACTIVE and SDMA.DMA4_CCRi[10] WR_ACTIVE to become '0' before being re-enabled for a new transfer. The FIFO drain for a channel will happen only in the following cases:

- If the channel is a source synchronized channel and SDMA.DMA4_CCRi[25] BUFFERING_DISABLE='0' and
- If the channel is not a solid fill channel and
- If the channel is not a transparent and copy channel

Note: In case of a self-linked or chain-linked channel, it is user responsibility to disable the SDMA.DMA4_CLNK_CTRLi[15] ENABLE_LINK bit before disabling the channel.

In all other cases, the channel will undergo an abort.

9.4.19 Reset

Following a software or hardware reset, all fields in the logical channel registers have undefined values, except 5 bits in the SDMA.DMA4_CCRi and SDMA.DMA4_CICRi registers. Thus, when programming a channel for the first time, the remaining fields in all channel registers must be configured before enabling the channel.

Note: After a reset, the global registers take their specified reset values.

9.4.20 Power Management

The SDMA module provides two methods to reduce power consumption:

- Interconnect clock auto-idle
- Automatic standby mode

9.4.20.1 Interconnect Clock Auto-Idle

The interconnect clock auto-idle power-saving mode is enabled or disabled in DMA register bit SDMA.DMA4_OCP_SYSCONFIG[0]. When this mode is enabled and there is no activity on the interconnect interface, the interconnect clock is disabled internally to the module to reduce power consumption. When there is new activity on the interconnect interface, the interconnect clock is restarted without any latency penalty. After reset, this mode is disabled by default. Enabling this mode is recommended to reduce power consumption.

9.4.20.2 Automatic Standby Mode

As part of the system-wide power-management scheme, the module can go into a standby mode at the request of the power, reset, and clock management (PRCM) module (for more information, see Chapter 4, *Power, Reset, and Clock Management*).

The module can be configured to one of the following standby modes using the DMA register bits SDMA.DMA4_OCP_SYSCONFIG[13:12]:

- No standby mode: The module never goes into standby mode.

- Force standby mode: The module goes into standby mode only when all the DMA channels are disabled.
- Smart standby mode: The module enters standby mode when:
  - All DMA channels are disabled.
  - No nonsynchronized channel is enabled, no DMA request line is asserted, and no DMA request is pending in the module.

9.5 SDMA Basic Programming Model

9.5.1 Setup Configuration

After a software or hardware reset, program all fields in the logical channel registers to default values for any channels used, because most fields are undefined following reset.

Before programming any DMA transfers, the priority arbitration rate and the maximum FIFO depth must be configured through the SDMA.DMA4_GCR register, and any required interrupts must be enabled through the SDMA.DMA4_IRQENABLE_Lj registers and the logical channel SDMA.DMA4_CICRi registers.

Software clears the SDMA.DMA4_CSRi register and the IRQSTATUS bit for the different interrupt lines before enabling the channel.

9.5.2 Software-Triggered (Nonsynchronized) Transfer

To program a software-triggered DMA transfer:

1. Configure the transfer parameters in the logical DMA channel registers:
   - SDMA.DMA4_CSDPi:
     - Transfer ES (8 bits, 16 bits, or 32 bits) and DMA register bits SDMA.DMA4_CSDPi[1:0]
     - Read and write port access types (single/burst), DMA register bits SDMA.DMA4_CSDPi[8:7] and SDMA.DMA4_CSDPi[15:14]
     - Source and destination endianism, DMA register bits SDMA.DMA4_CSDPi[21] and SDMA.DMA4_CSDPi[19]
     - Write mode (posted or nonposted) and DMA register bits SDMA.DMA4_CSDPi[17:16]
     - Source or destination packed or nonpacked (if the ES is less than the read/write port size), DMA register bits SDMA.DMA4_CSDPi[6] and SDMA.DMA4_CSDPi[13]
   - SDMA.DMA4_CENi: EN
   - SDMA.DMA4_CFNi: FN per transfer block
   - SDMA.DMA4_CSSAi and SDMA.DMA4_CDSAi: Source and destination start address (aligned with transfer ES)
   - SDMA.DMA4_CCRi:
     - Read and write port addressing modes, DMA register bits SDMA.DMA4_CCRi[13:12] and SDMA.DMA4_CCRi[15:14]
     - Priority bit for both read and write ports, DMA register bits SDMA.DMA4_CCRi[6] and SDMA.DMA4_CCRi[26]
     - DMA request number (set to 0 for a software-triggered transfer) and DMA register bits SDMA.DMA4_CCRi[4:0] = 0 and SDMA.DMA4_CCRi[20:19] = 0
   - SDMA.DMA4_CSEi, SDMA.DMA4_CSFi, SDMA.DMA4_CDEi, and SDMA.DMA4_CDFi: Source and destination element and frame indexes (depending on addressing mode)
2. Start the transfer through the enable bit in the channel SDMA.DMA4_CCRi register and DMA register bit SDMA.DMA4_CCRi[7]

The example below perform a DMA transfer on channel 10 of a 240*160 picture from RAM to RAM (0x80C00000 to 0x80F00000) :

```
  UWORD32 RegVal = 0;
DMA4_t *DMA4;

DMA4 = (DMA4_t *)malloc(sizeof(DMA4_t));
```

```
  /* Init. parameters */
  DMA4->DataType = 0x2;                          // DMA4_CSDPi[1:0]
  DMA4->ReadPortAccessType = 0;                  // DMA4_CSDPi[8:7]
  DMA4->WritePortAccessType = 0;                 // DMA4_CSDPi[15:14]
  DMA4->SourceEndiansim = 0;                     // DMA4_CSDPi[21]
  DMA4->DestinationEndianism = 0;                // DMA4_CSDPi[19]
  DMA4->WriteMode = 0;                           // DMA4_CSDPi[17:16]
  DMA4->SourcePacked = 0;                        // DMA4_CSDPi[6]
  DMA4->DestinationPacked = 0;                   // DMA4_CSDPi[13]
  DMA4->NumberOfElementPerFrame = 240;           // DMA4_CENi
  DMA4->NumberOfFramePerTransferBlock = 160;     // DMA4_CFNi
  DMA4->SourceStartAddress = 0x80C00000;         // DMA4_CSSAi
  DMA4->DestinationStartAddress = 0x80F00000;    // DMA4_CDSAi
  DMA4->SourceElementIndex = 1;                  // DMA4_CSEi
  DMA4->SourceFrameIndex = 1;                    // DMA4_CSFi
  DMA4->DestinationElementIndex = 1;             // DMA4_CDEi
  DMA4->DestinationFrameIndex = 1;               // DMA4_CDFi
  DMA4->ReadPortAccessMode = 1;                  // DMA4_CCRi[13:12]
  DMA4->WritePortAccessMode = 1;                 // DMA4_CCRi[15:14]
  DMA4->ReadPriority = 0;                        // DMA4_CCRi[6]
  DMA4->WritePriority = 0;                       // DMA4_CCRi[23]
  DMA4->ReadRequestNumber = 0;                   // DMA4_CCRi[4:0]
  DMA4->WriteRequestNumber = 0;                  // DMA4_CCRi[20:19]



  /* 1) Configure the transfer parametres in the logical DMA registers */
  /*-------------------------------------------------------------------*/

  /* a) Set the data type CSDP[1:0], the Read/Write Port access type CSDP[8:7]/[15:14], the
Source/dest endiansim CSDP[21]/CSDP[19], write mode
  CSDP[17:16], source/dest packed or non-packed CSDP[6]/CSDP[13]*/
  // Read CSDP
  RegVal = DMA4_CSDP_CH10;


  // Build reg
  RegVal = ((RegVal & ~ 0x3) | DMA4->DataType );
  RegVal = ((RegVal & ~(0x3 << 7))  | (DMA4->ReadPortAccessType << 7));
  RegVal = ((RegVal & ~(0x3 << 14)) | (DMA4->WritePortAccessType << 14));
  RegVal = ((RegVal & ~(0x1 << 21)) | (DMA4->SourceEndiansim << 21));
  RegVal = ((RegVal & ~(0x1 << 19)) | (DMA4->DestinationEndianism << 19));
  RegVal = ((RegVal & ~(0x3 << 16)) | (DMA4->WriteMode << 16));
  RegVal = ((RegVal & ~(0x1 << 6))  | (DMA4->SourcePacked << 6));
  RegVal = ((RegVal & ~(0x1 << 13)) | (DMA4->DestinationPacked << 13));

  // Write CSDP
  DMA4_CSDP_CH10 = RegVal;


  /* b) Set the number of element per frame CEN[23:0]*/
  DMA4_CEN_CH10 = DMA4->NumberOfElementPerFrame;

  /* c) Set the number of frame per block CFN[15:0]*/
  DMA4_CFN_CH10 = DMA4->NumberOfFramePerTransferBlock;


  /* d) Set the Source/dest  start address index CSSA[31:0]/CDSA[31:0]*/
  DMA4_CSSA_CH10     = DMA4->SourceStartAddress; // address start
  DMA4_CDSA_CH10     = DMA4->DestinationStartAddress; // address dest


  /* e) Set t1he Read Port adressing mode CCR[13:12], the Write Port adressing mode CCR[15:14],
read/write priority CCR[6]/CCR[26],  the current LCH CCR[20:19]=00 and CCR[4:0]=00000*/
  // Read CCR
  RegVal = DMA4_CCR_CH10;

  // Build reg
  RegVal = ((RegVal & ~(0x3 << 12)) | (DMA4->ReadPortAccessMode  << 12));
  RegVal = ((RegVal & ~(0x3 << 14)) | (DMA4->WritePortAccessMode << 14));
  RegVal = ((RegVal & ~(0x1 << 6))  | (DMA4->ReadPriority << 6));
  RegVal = ((RegVal & ~(0x1 << 26)) | (DMA4->WritePriority << 26));

  RegVal&= 0xFFCFFFE0 ;
```

```
// Write CCR
DMA4_CCR_CH10 = RegVal;


/* f)- Set the source element index CSEI[15:0]*/
DMA4_CSEI_CH10 = DMA4->SourceElementIndex;


/*     - Set the source frame index CSFI[15:0]*/
DMA4_CSFI_CH10 = DMA4->SourceFrameIndex ;

/*     - Set the destination element index CDEI[15:0]*/
DMA4_CDEI_CH10 = DMA4->DestinationElementIndex;

/*     - Set the destination frame index CDFI[31:0]*/
DMA4_CDFI_CH10 = DMA4->DestinationFrameIndex;


/* 2) Start the DMA transfer by Setting the enable bit CCR[7]=1 */
/*-------------------------------------------------------------*/
//write enable bit
DMA4_CCR_CH10      |= 1 << 7; /* start */
```

9.5.3 Hardware-Synchronized Transfer

To monitor a hardware synchronized DMA transfer, initialize the SDMA.DMA4_CDACi register before the software enable.

To configure an LCh to synchronize by element, packet, frame, or block, the frame synchronization SDMA.DMA4_CCRi[5] FS bit and the block synchronization SDMA.DMA4_CCRi[18] BS bit register must be programmed. For all the following synchronized transfers (element, packet, frame or block synchronized transfers) User must set first : SDMA.DMA4_CCRi[24] SEL_SRC_DST_SYNC to 1 when the source triggers on the DMA request and SDMA.DMA4_CCRi[24] SEL_SRC_DST_SYNC to 0 when the Destination triggers on the DMA request. Note: User must take care when setting the SDMA.DMA4_CCRi[23] PREFETCH bit it is in conjunction with SDMA.DMA4_CCRi[24] SEL_SRC_DST_SYNC bit .

- To configure an LCh to transfer one element per DMA request:

1. Set the number of DMA request associated to the current LCH in the SDMA.DMA4_CCRi[20:19] SYNCHRO_CONTROL_UPPER and SDMA.DMA4_CCRi[4:0] SYNCHRO bit field.
2. Set the data type, also referenced as element size (ES), in the SDMA.DMA4_CSDPi[1:0] DATA_TYPE bit field.
3. Set the Read Port access type (single or burst access) in the SDMA.DMA4_CSDPi[8:7] SRC_BURST_EN bit field.
4. Set the Write Port access type (single or burst access) in the SDMA.DMA4_CSDPi[15:14] DST_BURST_EN bit field.
5. Set the Read Port addressing mode in the SDMA.DMA4_CCRi[13:12] SRC_AMODE bit field.
6. Set the Write Port addressing mode in the SDMA.DMA4_CCRi[15:14] DST_AMODE bit field.
7. Set the Read start address in the SDMA.DMA4_CSSAi[31:0] SRC_START_ADRS bit field.
8. Set the Write start address in the SDMA.DMA4_CDSAi[31:0] DST_START_ADRS bit field.
9. Set both FS and BS to 0 in SDMA.DMA4_CCRi[5] FS and SDMA.DMA4_CCRi[18] BS.
10. Set to 1 the channel enable bit SDMA.DMA4_CCRi[7] EN bit.

- To configure an LCh to transfer one frame per DMA request:

1. Set the number of DMA request associated to the current LCH in the SDMA.DMA4_CCRi[20:19] SYNCHRO_CONTROL_UPPER and SDMA.DMA4_CCRi[4:0] SYNCHRO bit field.
2. Set the data type, also referenced as element size (ES), in the SDMA.DMA4_CSDPi[1:0] DATA_TYPE bit field.
3. Set the number of element per frame in the SDMA.DMA4_CENi[23:0] CHANNEL_ELMNT_NBR bit field.
4. Set the Read Port access type (single or burst access) in the SDMA.DMA4_CSDPi[8:7]

SRC_BURST_EN bit field.

5. Set the Write Port access type (single or burst access) in the SDMA.DMA4_CSDPi[15:14] DST_BURST_EN bit field.
6. Set the Read Port addressing mode in the SDMA.DMA4_CCRi[13:12] SRC_AMODE bit field.
7. Set the Write Port addressing mode in the SDMA.DMA4_CCRi[15:14] DST_AMODE bit field.
8. Set the Read start address in the SDMA.DMA4_CSSAi[31:0] SRC_START_ADRS bit field.
9. Set the Write start address in the SDMA.DMA4_CDSAi[31:0] DST_START_ADRS bit field.
10. Set FS to 1 and BS to 0 respectively in SDMA.DMA4_CCRi[5] FS and SDMA.DMA4_CCRi[18] BS.
11. Set to 1 the channel enable bit SDMA.DMA4_CCRi[7] EN bit.

- To configure an LCh to transfer one block per DMA request:

1. Set the number of DMA request associated to the current LCH in the SDMA.DMA4_CCRi[20:19] SYNCHRO_CONTROL_UPPER and SDMA.DMA4_CCRi[4:0] SYNCHRO bit field.
2. Set the data type, also referenced as element size (ES), in the SDMA.DMA4_CSDPi[1:0] DATA_TYPE bit field.
3. Set the number of element per frame in the SDMA.DMA4_CENi[23:0] CHANNEL_ELMNT_NBR bit field.
4. Set in the SDMA.DMA4_CFNi[15:0] CHANNEL_FRAME_NBR bit field the number of frame (transfers), to take place before the LCH is disabled.
5. Set the Read Port access type (single or burst access) in the SDMA.DMA4_CSDPi[8:7] SRC_BURST_EN bit field.
6. Set the Write Port access type (single or burst access) in the SDMA.DMA4_CSDPi[15:14] DST_BURST_EN bit field.
7. Set the Read Port addressing mode in the SDMA.DMA4_CCRi[13:12] SRC_AMODE bit field.
8. Set the Write Port addressing mode in the SDMA.DMA4_CCRi[15:14] DST_AMODE bit field.
9. Set the Read start address in the SDMA.DMA4_CSSAi[31:0] SRC_START_ADRS bit field.
10. Set the Write start address in the SDMA.DMA4_CDSAi[31:0] DST_START_ADRS bit field.
11. Set FS to 0 and BS to 1 respectively in SDMA.DMA4_CCRi[5] FS and SDMA.DMA4_CCRi[18] BS.
12. Set to 1 the channel enable bit SDMA.DMA4_CCRi[7] EN bit.

- To configure an LCh to transfer one packet per DMA request:

1. Set the number of DMA request associated to the current LCH in the SDMA.DMA4_CCRi[20:19] SYNCHRO_CONTROL_UPPER and SDMA.DMA4_CCRi[4:0] SYNCHRO bit field.
2. Set the data type, also referenced as element size (ES), in the SDMA.DMA4_CSDPi[1:0] DATA_TYPE bit field.
3. Set the number of element per packet to transfer: If the packet requestor is in the source, set SDMA.DMA4_CCR.Sel_Src_Dst_Sync to 1 and set the packet element number in the SDMA.DMA4_CSFIi register; else, if the packet requestor is in the destination, set the SDMA.DMA4_CCRi[24] SEL_SRC_DST_SYNC to 0 and set the packet element number in the SDMA.DMA4_CDFIi register.
4. Set the number of element per frame in the SDMA.DMA4_CENi[23:0] CHANNEL_ELMNT_NBR bit field.
5. Set in the SDMA.DMA4_CFNi[15:0] CHANNEL_FRAME_NBR bit field the number of frame (transfers), to take place before the LCH is disabled.
6. Set the element number in the packet in the SDMA.DMA4_CSFIi[15:0] PKT_ELNT_NBR, if constant addressing or post-incremented addressing modes are used in the source side. However, the number of elements in the packet is set in the SDMA.DMA4_CDFIi[15:0] PKT_ELNT_NBR if constant addressing mode is used in the destination side.
7. Set the Read Port access type (single or burst access) in the SDMA.DMA4_CSDPi[8:7] SRC_BURST_EN bit field.
8. Set the Write Port access type (single or burst access) in the SDMA.DMA4_CSDPi[15:14] DST_BURST_EN bit field.
9. Set the Read Port addressing mode in the SDMA.DMA4_CCRi[13:12] SRC_AMODE bit field.
10. Set the Write Port addressing mode in the SDMA.DMA4_CCRi[15:14] DST_AMODE bit field.
11. Set the Read start address in the SDMA.DMA4_CSSAi[31:0] SRC_START_ADRS bit field.

12. Set the Write start address in the SDMA.DMA4_CDSAi[31:0] DST_START_ADRS bit field.
13. Set FS to 1 and BS to 1 respectively in SDMA.DMA4_CCRi[5] FS and SDMA.DMA4_CCRi[18] BS.
14. Set to 1 the channel enable bit SDMA.DMA4_CCRi[7] EN bit.

Note: It is possible to stop a transfer by disabling the channel. This is done by resetting the ENABLE bit in the SDMA.DMA4_CCRi register.

9.5.4 Synchronized Transfer Monitoring Using CDAC

The SDMA.DMA4_CDACi register is writable and non-initialized (value undefined). It can be initialized to monitor a transfer by applying the following programming model:

1. Write 0 in the SDMA.DMA4_CDACi
2. Enable the channel.
3. If timeout occurs, read SDMA.DMA4_CDACi
4. If SDMA.DMA4_CDACi!= SDMA.DMA4_CDACi reset value:
   Then transfer starts. User can then rely on SDMA.DMA4_CCENi and SDMA.DMA4_CCFNi element and frame counters.
   Else, if SDMA.DMA4_CDACi = SDMA.DMA4_CDACi reset value:
   Then transfer does not start.

9.5.5 Concurrent Software and Hardware Synchronization

This section describes thread allocation only, not the entire transfer. Because synchronized transfers are latency critical, you must allocate a thread on the synchronized target side at least.

Even for multiple concurrent channels, thread reservation guarantees that as soon as an HW DMA request comes in, the read/write scheduler finds available thread(s) to initiate a channel schedule and issue a read/write transaction.

Consider these six concurrent channels:

- Channels 0/1/2/3 are dedicated to memory-memory transfer: they are software triggered and not synchronized.
- Channel 4 is dedicated to memory→peripheral transfer, hardware triggered, and synchronized on the write side.
- Channel 5 is dedicated to peripheral→memory transfer, hardware triggered, and synchronized on the read side.

1. Allow thread reservation for priority channel 4 and channel 5:
   Reserve one thread (Read ThreadID 0) on the read port: Set SDMA.DMA4_GCR[13:12] = 0x1.
   Reserve one thread (Write ThreadID 0) on the write port: Set SDMA.DMA4_GCR[13:12] = 0x1.
2. Specify channel priority:
   Channel 4 is a write high priority channel: Set SDMA.DMA4_CCRi[26] = 1.
   Channel 5 is a read high priority channel: Set SDMA.DMA4_CCRi[6] = 1.

9.5.6 Chained Transfer

A chained DMA transfer can be programmed as follows:

1. Configure the transfer parameters for each logical DMA channel in the chain as in step 1 for either the synchronized or non-synchronized transfers above.
2. For each channel in the chain, configure the SDMA.DMA4_CLNK_CTRLi register as follows:
   - Next logical DMA channel number (for a looping chained transfer link last channel to first channel number), in DMA register bits SDMA.DMA4_CLNK_CTRLi[4:0].
   - Include the logical channel to the chain and enable link by setting the DMA register bit SDMA.DMA4_CLNK_CTRLi[15].
   - For a non-looping chain, the last logical channel in the chain must have the DMA register bit

SDMA.DMA4_CLNK_CTRLi[15] set to 0 to indicate the end of the chain.

3. Enable the transfer via the enable bit in the first logical channel DMA register bit SDMA.DMA4_CCRi[7]. All other channels in the chain must be configured as disabled. Each channel is enabled automatically in turn when the previous logical channel transfer completes. A non-synchronized transfer starts immediately; a hardware-synchronized transfer starts when the DMA request line corresponding to the first DMA channel in the chain is asserted.

To stop a looping chained transfer, disable the NEXTLCH_ID bit, DMA register bit SDMA.DMA4_CLNK_CTRLi[15](ENABLE_LNK bit set to 0x0), of the final channel transfer.

In the RAM to RAM copy example, to copy in loop it's possible to link channel 10 on itself. The following line can be added in the channel configuration :

```
/* g) Set link for loop */
DMA4_CLINK_CTRL_CH10 = 0x0000800A;
```

9.5.7 90° Clockwise Image Rotation

The 90° clockwise image rotation example described in Section 9.4.3, *Addressing Modes*, can be programmed as follows:

1. Configure the transfer parameters in the logical DMA channel registers:
   - SDMA.DMA4_CSDPi:
     - Transfer ES = 32-bit (32 bpp), DMA register bits SDMA.DMA4_CSDPi[1:0]
     - Read and write port access types = maximum burst size supported by memory device, DMA register bits SDMA.DMA4_CSDPi[8:7] and SDMA.DMA4_CSDPi[15:14]
     - Source and destination endianism, DMA register bits SDMA.DMA4_CSDPi[21] and SDMA.DMA4_CSDPi[19]
     - Write mode = posted with last element nonposted, DMA register bits SDMA.DMA4_CSDPi[17:16]
     - Source and destination packed = Yes (although destination writes will not benefit because EI>1), DMA register bits SDMA.DMA4_CSDPi[6] and SDMA.DMA4_CSDPi[13]
   - SDMA.DMA4_CENi: EN = 240
   - SDMA.DMA4_CFNi: FN per transfer block = 160
   - SDMA.DMA4_CSSAi: Source start address = 0x100000
   - SDMA.DMA4_CDSAi: destination start address = 0x20013E
   - SDMA.DMA4_CCRi:
     - Read and write port addressing modes = double-index addressing mode for both or post-increment addressing on source and double-index addressing on destination, DMA register bits SDMA.DMA4_CCRi[13:12] and SDMA.DMA4_CCRi[15:14]
     - Low or high priority, DMA register bit SDMA.DMA4_CCRi[6]
     - DMA request number = 0 (for software-triggered transfer), DMA register bits SDMA.DMA4_CCRi[4:0] and SDMA.DMA4_CCRi[20:19]
   - SDMA.DMA4_CSEi: Source EI = 1
   - SDMA.DMA4_CSFi: Source frame index = 1
   - SDMA.DMA4_CDEi: destination EI = 637
   - SDMA.DMA4_CDFi: destination frame index = –152967
2. Start the transfer via the enable bit in the channel SDMA.DMA4_CCRi register.

Below are the parameters to perform this rotation from 0x80C00000 RAM address to 0x80F00000, with the same code as in Section 9.5.2:

```
 /* Init. parameters */
DMA4->DataType = 0x2;                       // DMA4_CSDPi[1:0]
DMA4->ReadPortAccessType = 0x3;             // DMA4_CSDPi[8:7]
DMA4->WritePortAccessType = 0x3;            // DMA4_CSDPi[15:14]
DMA4->SourceEndiansim = 0;                  // DMA4_CSDPi[21]
DMA4->DestinationEndianism = 0;             // DMA4_CSDPi[19]
DMA4->WriteMode = 0x2;                      // DMA4_CSDPi[17:16]
DMA4->SourcePacked = 0x1;                   // DMA4_CSDPi[6]
DMA4->DestinationPacked = 0x1;              // DMA4_CSDPi[13]
DMA4->NumberOfElementPerFrame = 240;        // DMA4_CENi
```

```
DMA4->NumberOfFramePerTransferBlock = 160;    // DMA4_CFNi
DMA4->SourceStartAddress = 0x80C00000;        // DMA4_CSSAi
DMA4->DestinationStartAddress = 0x80F00000;   // DMA4_CDSAi
DMA4->SourceElementIndex = 1;                 // DMA4_CSEi
DMA4->SourceFrameIndex = 1;                   // DMA4_CSFi
DMA4->DestinationElementIndex = 637;          // DMA4_CDEi
DMA4->DestinationFrameIndex = -152967;        // DMA4_CDFi
DMA4->ReadPortAccessMode = 0x3;               // DMA4_CCRi[13:12]
DMA4->WritePortAccessMode = 0x3;              // DMA4_CCRi[15:14]
DMA4->ReadPriority = 0;                       // DMA4_CCRi[6]
DMA4->WritePriority = 0;                      // DMA4_CCRi[23]
DMA4->ReadRequestNumber = 0;                  // DMA4_CCRi[4:0]
DMA4->WriteRequestNumber = 0;                 // DMA4_CCRi[20:19]
```

9.5.8 Graphic Operations

- Transparent copy:
  - 1. Set the SDMA.DMA4_CCRi[17] Transparent_Copy_Enable bit field to 1.
  - 2. Set the SDMA.DMA4_CCRi[16] Constant_Fill_Enable bit field to 0.
  - 3. Set the value of key the color in the SDMA.DMA4_COLORi[15:0] color_key bit field.

  To perform this graphic operation, the following lines can be added to the example of Section 9.5.2

```
 DMA4_CCR_CH10 &= ~(0x1 << 16);
DMA4_CCR_CH10 |= 0x1 << 17;
DMA4_COLOR_CH10 = 0x00000003;
```

- Solid Color fill:
  - 1. Set the SDMA.DMA4_CCRi[16] Constant_Fill_Enable bit field to 1.
  - 2. Set the SDMA.DMA4_CCRi[17] Transparent_Copy_Enable bit field to 0.
  - 3. Set the value of key the color in the DMA4_COLORi[15:0] solid_color bit field.

  To perform this graphic operation, the following lines can be added to the example of Section 9.5.2

```
DMA4_CCR_CH10 &= ~(0x1 << 17);
DMA4_CCR_CH10 |= 0x1 << 16;
DMA4_COLOR_CH10 = 0x00000003;
```

9.6 SDMA Use Cases and Tips

9.6.1 Camcorder Use Case: How to Configure SDMA to Handle Transfers With McBSP2 and MMC to External DRAM

9.6.1.1 Introduction

In this use case, the SDMA manages:

- The audio stream between the McBSP and the external DRAM
- The audio stream between the external DRAM and the MMC buffer
- The video stream between the external DRAM and the MMC buffer

The following sections describe how to configure the SDMA controller with MMC and McBSP in this use case.

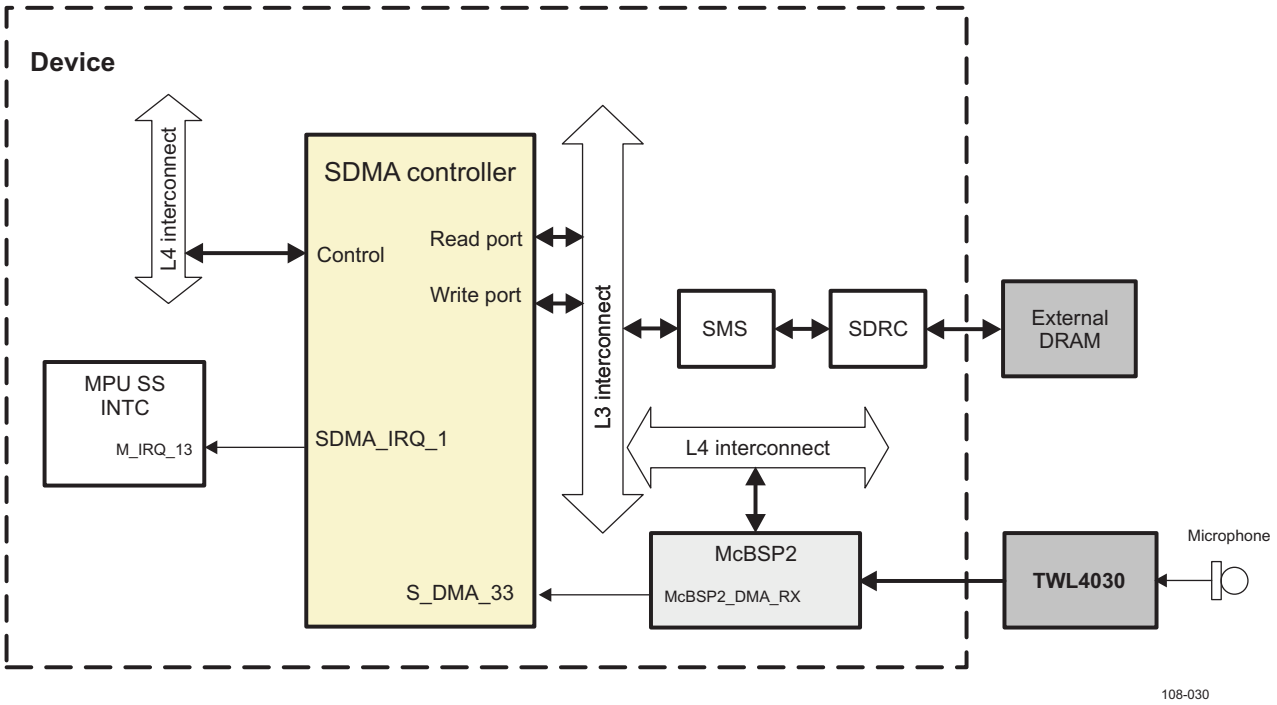
9.6.1.2 SDMA Configuration to Transfer Data Between the McBSP and External DRAM

9.6.1.2.1 Overview

The SDMA gets data from the McBSP_DRR register and copies it into three rolling buffers in the external DRAM.

Figure 9-13 is an overview of the audio path.

Figure 9-13. Overview



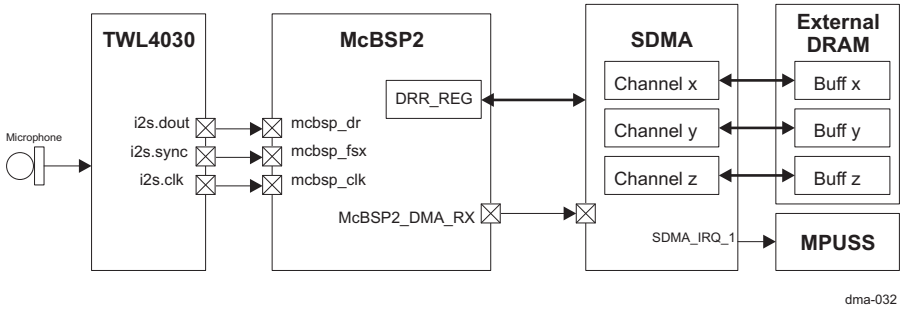
For the camcorder use case, the DMA transfer data format is:

- Frame size = 2048 x 32 bits
- Three DMA channels to manage three transfers to three memory buffers
- Interrupt generated at the end of each frame transfer
- Transfer triggered by McBSP2

9.6.1.2.2 Environment

The DMA accesses MCBSP2_DRR_REG through the interconnect and receives a DMA request from this McBSP module for driving transfers. The DMA accesses the external DRAM through the interconnect. The McBSP is directly connected to the TWL4030 device. Figure 9-14 shows the environment of this scenario.

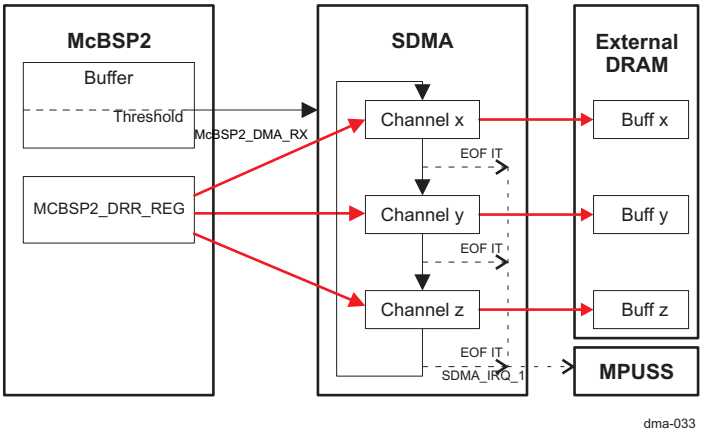
Figure 9-14. Environment



9.6.1.2.3 Data Path

The transfer is started when the DMA transfer is enabled and the MCBSP2_DMA_REQ is asserted (that is, when the MCBSP FIFO threshold is reached). The DMA starts copying the first frame into the first memory buffer. When a whole frame has been transferred (2048 data), an interrupt is asserted to the MPU, and then the second channel is enabled and fills the second buffer, and so on. Figure 9-15 shows the data flow.

Figure 9-15. Data Flow



9.6.1.2.4 Programming Flow

To establish a configuration for the DMA transfers, the following points have been considered.

The McBSP2_DRR_REG register is 32 bits wide, but in the camcorder use case, only the 16 LSBs are used (monophonic 16-bit-wide audio channel).

The McBSP receives data in a FIFO buffer. As long as the occupied locations level in this read buffer is greater than or equal to the threshold value + 1 , the DMA request is asserted. After transferring the configured (THRSH1_REG value + 1) number of words, the receive DMA request is deasserted and then reasserted when the conditions are met again. Therefore, the DMA can receive at least threshold value data, with a threshold maximum value = 1280 elements = size of McBSP2 FIFO.

The frame synchronization transfer mode of the SDMA can be used with McBSP for frames with a size equal to the threshold. In the use case, however, the audio data frame is equal to 2048 elements, which is greater than the maximum transfer length possible with the McBSP.

Because the SDMA cannot be configured in frame synchronization mode, packet synchronization mode must be used. This mode enables to specify the maximum size of a transfer and to divide a frame into as many packet as needed. It is then possible to configure a frame of n elements and generate an interrupt at the end-of-frame transfer, while the transfer is segmented into packets.

The management of the three memory buffers is done by configuring three DMA channels linked between themselves: channel x is linked to channel y, channel y is linked to channel z, and channel z is linked to channel x. The three channels have the same configuration, except for the destination address, which corresponds to the three buffer addresses. Channels 11, 12, and 13 are used. Each channel generates the end-of-frame interrupts on the L1 interrupt line of the SDMA, to warn the MPU of the end of a frame transfer.

With these considerations, the DMA register of the three channels (i = 11, 12, and13) is configured as follows:

1. Channel source destination parameters: DMA4_CSDPi
   - DMA4_CSDPi[1:0] DATA_TYPE = 0x1: Read 16-bit elements from the McBSP_DRR_REG register.
   - DMA4_CSDPi[5:2] RD_ADD_TRSLT = 0: Not useful
   - DMA4_CSDPi[6] SRC_PACKED = 0x0: Cannot pack source data
   - DMA4_CSDPi[8:7] SRC_BURST_EN = 0x0: Cannot burst source
   - DMA4_CSDPi[12:9] WR_ADD_TRSLT = 0: Undefined
   - DMA4_CSDPi[13] DST_PACKED = 0x1: Pack two 16-bit elements in one 32-bit packet to optimize transfer.
   - DMA4_CSDPi[15:14] DST_BURST_EN = 0x3: Burst at 16x32-bit
   - DMA4_CSDPi[17:16] WRITE_MODE = 0x1: Write posted
   - DMA4_CSDPi[18] DST_ENDIAN_LOCK = 0x0: Endianness adapt
   - DMA4_CSDPi[19] DST_ENDIAN = 0x0: Little endian type at destination

- DMA4_CSDPi[20] SRC_ENDIAN_LOCK = 0x0: Endianness adapt
- DMA4_CSDPi[21] SRC_ENDIAN = 0x0: Little endian type at source
- DMA4_CSDPi[31:22] RESERVED = 0x0: For future compatibility

2. Channel control register: DMA4_CCRi
   - DMA4_CCRi[4:0] SYNCHRO_CONTROL = DmaReq & 0x1F = 0x2: 5 first bits of McBSP2_DMA_RX
   - DMA4_CCRi[5] FS = 1: Packet mode with BS = 0x1
   - DMA4_CCRi[6] READ_PRIORITY = 0x0: Low priority on read side
   - DMA4_CCRi[7] ENABLE = 0x0: The logical channel is disabled.
   - DMA4_CCRi[8] SUSPEND_SENSITIVE = 0
   - DMA4_CCRi[9] RD_ACTIVE: Read status, read-only access
   - DMA4_CCRi[10] WR_ACTIVE: Write status, read-only access
   - DMA4_CCRi[11] RESERVED = 0: Write 0s for future compatibility.
   - DMA4_CCRi[13:12] SRC_AMODE = 0x0: Constant address mode; DMA always reads McBSP2_DRR_REG.
   - DMA4_CCRi[15:14] DST_AMODE = 0x1: Post-incremented address mode
   - DMA4_CCRi[16] CONST_FILL_ENABLE = 0x0: Constant fill mode is disabled.
   - DMA4_CCRi[17] TRANSPARENT_COPY_ENABLE = 0x0: Transparent copy mode is disabled.
   - DMA4_CCRi[18] BS = 0x1: Packet mode with FS = 0x1
   - DMA4_CCRi[20:19] SYNCHRO_CONTROL_UPPER = 0x1: Two MSBs of McBSP2_DMA_RX
   - DMA4_CCRi[22] SUPERVISOR = 0x0: Supervisor mode is disabled.
   - DMA4_CCRi[23] PREFETCH = 0x0: Prefetch mode is disabled, cannot prefetch a constant adressing source.
   - DMA4_CCRi[24] SEL_SRC_DST_SYNC = 0x1: Transfer is triggered by the source. The packet element number is specified in the DMA4_CSFI register.
   - DMA4_CCRi[25] BUFFERING_DISABLE = 0x0
   - DMA4_CCRi[26] WRITE_PRIORITY = 0x0: Channel has low priority on Write side during the arbitration process.
   - DMA4_CCRi[31:27] RESERVED = 0x0: Write 0s for future compatibility.
3. Channel parameters: DMA4_CENi, DMA4_CFNi, DMA4_CSSAi, DMA4_CDSAi, DMA4_CSEi, DMA4_CSFi, DMA4_CDEi
   - DMA4_CENi[23:0] CHANNEL_ELMNT_NBR = 2048: 2048 elements
   - DMA4_CFNi[15:0] CHANNEL_FRAME_NBR = 1: One frame
   - DMA4_CSSAi[31:0] SRC_START_ADRS = 0x49022000: Channel source start address = MCBSP2_DRR_REG
   - DMA4_CDSAi[31:0] DST_START_ADRS= X, Y or Z: Channel destination start address in external DRAM, where X, Y, and Z represent the addresses of the three buffers
   - DMA4_CSEi[31:0] CHANNEL_SRC_ELMNT_INDEX = 1: Channel source element index
   - DMA4_CSFi[15:0] 16BIT_PKT_ELNT_NBR = 0x80: 16-bit Packet size = MCBSP_FIFO_THRESHOLD + 1
   - DMA4_CDEi[15:0] CHANNEL_DST_ELMNT_INDEX = 0x1: Channel destination element index
4. Channel linking: DMA4_CLNK_CTRLi
   - DMA4_CLNK_CTRL11[31:0] = 0x0000800C: Channel 11 is linked to channel 12.
   - DMA4_CLNK_CTRL12[31:0] = 0x0000800D: Channel 12 is linked to channel 13.
   - DMA4_CLNK_CTRL13[31:0] = 0x0000800B: Channel 13 is linked to channel 11.
5. Interrupt management: DMA4_CICRi and DMA4_IRQENABLE_Lj
   - DMA4_CICRi[31:0] = 0x00000008: Enables the end of frame interrupt and disables others (the value of these registers is unknown after a reset; it is necessary to clear other bits)
   - DMA4_IRQENABLE_L1[31:0] = 0x00022800; interrupts of channels 11, 12, and 13 are unmasked on IRQ line L1.
6. Launch transfer: DMA4_CCRi[7] ENABLE = 0x1: Enables the three channels, starting with channel 13, then 12, and finally 11; the last one triggers the others.

Table 9-9 lists the registers of DMA channel 11 after a first transfer.

Table 9-9. Registers Print

Register Name	Address	Value	Value Description
DMA4_IRQENABLE_L1	0x4805601c	0x00022800	Interrupts of channels 11, 12, and 13 are unmasked on IRQ line L1.
DMA4_CICR11	0x480564a8	0x00000008	End-of-frame interrupt enabled
DMA4_CCR11	0x480564a0	0x010c40a2	Channel control register
DMA4_CSDP11	0x480564b0	0x0001e001	Channel source destination parameters
DMA4_CEN11	0x480564b4	0x00000800	Channel element number
DMA4_CFN11	0x480564b8	0x00000001	Channel frame number
DMA4_CSSA11	0x480564bc	0x49022000	Channel source start address
DMA4_CDSA11	0x480564c0	0x81e00000	Channel destination start address
DMA4_CSFI11	0x480564c8	0x00000080	Packet size
DMA4_CSAC11	0x480564d4	0x49022000	Source address counter (read only)
DMA4_CDAC11	0x480564d8	0x81e01000	Destination address counter (read only)
DMA4_CCEN11	0x480564dc	0x00000800	Channel current transferred element number in the current frame (read only)
DMA4_CCFN11	0x480564e0	0x00000001	Channel current transferred frame number in the current transfer (read only)
DMA4_CLNK_CTRL11	0x480564a4	0x0000800C	Channel link control register: Link to channel 12

Channels 12 and 13 have a similar configuration; only the destination address and channel link change.

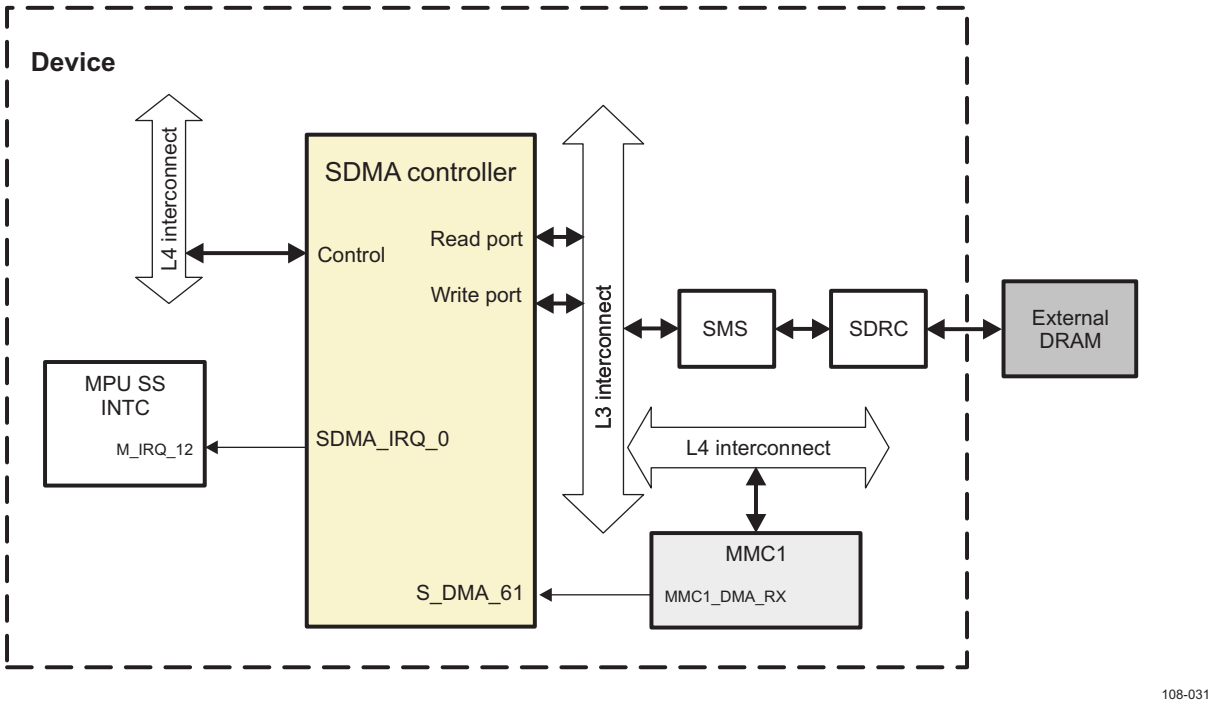
9.6.1.3 SDMA Configuration to Transfer Data Between MMC and External DRAM

9.6.1.3.1 Overview

The SDMA gets data from the MMC.MMCHS_DATA register and copies it into the external DRAM. The SDMA accesses the MMCHS_DATA through the interconnect and receives a DMA request from the MMC module for driving transfers. The DMA accesses the external DRAM through the interconnect. The transfer is started when the DMA transfer is enabled and the MMC1_DMA_REQ is asserted (that is, when the MMC buffer is ready).

Figure 9-16 is an overview of the path.

Figure 9-16. Overview



For the camcorder use case, the DMA transfer data format is:

- Frame size = 4096 x 32 bits
- packet size = 512 x 32 bits (512 elements ready in MMC when a DMA request is asserted)
- DMA channel to manage the transfer
- Interrupt generated a the end of each frame transfer
- Transfer triggered by MMC

9.6.1.3.2 Programming Flow

To establish a configuration for the DMA transfers, the following points have been considered.

The MMCHS_DATA register is 32 bits wide.

The SDMA is configured in packet synchronization mode. This mode enables to specify the transfer of any size frame and then to divide the frame into as many packet as needed. It is therefore possible to configure a frame of n elements and generate an interrupt at the end-of-frame transfer, while the transfer is segmented into packets.

The MMC puts its received data into a 1K-byte ping-pong buffer. When 512 x 32 bits are ready in this buffer, the DMA request is asserted. Then the DMA controller transfers the defined amount of elements divided into packets of 512 x 32-bit element (PKT_ELNT_NBR) transfers.

In the use case, the size of the transfer is set at 4096 x 32 bits, chosen after performance benches.

With these considerations, the DMA register of channel i is configured as follows:

1. Channel source destination parameters: DMA4_CSDPi
   - DMA4_CSDPi[1:0] DATA_TYPE = 0x2: Read 32-bit elements from the MMCHS_DATA register.
   - DMA4_CSDPi[5:2] RESERVED: Write 0s for future compatibility. Read returns 0.
   - DMA4_CSDPi[6] SRC_PACKED = 0x0: Cannot pack source data
   - DMA4_CSDPi[8:7] SRC_BURST_EN = 0x0: Cannot burst source
   - DMA4_CSDPi[12:9] RESERVED: Write 0s for future compatibility. Read returns 0.
   - DMA4_CSDPi[13] DST_PACKED = 0x0: No packing
   - DMA4_CSDPi[15:14] DST_BURST_EN = 0x3: Burst at 16x32 bits

- DMA4_CSDPi[17:16] WRITE_MODE = 0x1: Write posted
- DMA4_CSDPi[18] DST_ENDIAN_LOCK = 0x0: Endianness adapt
- DMA4_CSDPi[19] DST_ENDIAN = 0x0: Little Endian type at destination
- DMA4_CSDPi[20] SRC_ENDIAN_LOCK = 0x0: Endianness adapt
- DMA4_CSDPi[21] SRC_ENDIAN = 0x0: Little endian type at source
- DMA4_CSDPi[31:22] RESERVED = 0x0: For future compatibility

2. Channel control register: DMA4_CCRi
   - DMA4_CCRi[4:0] SYNCHRO_CONTROL = DmaReq & 0x1F = 0x1E: 5 first bits of MMC1_DMA_RX(62)
   - DMA4_CCRi[5] FS = 1: Packet mode with BS = 0x1
   - DMA4_CCRi[6] READ_PRIORITY = 0x0: Low priority on read side
   - DMA4_CCRi[7] ENABLE = 0x0: The logical channel is disabled.
   - DMA4_CCRi[8] SUSPEND_SENSITIVE = 0
   - DMA4_CCRi[9] RD_ACTIVE: Read status, read-only access
   - DMA4_CCRi[10] WR_ACTIVE: Write status, read-only access
   - DMA4_CCRi[11] RESERVED = 0: Write 0s for future compatibility.
   - DMA4_CCRi[13:12] SRC_AMODE = 0x0: Constant address mode; DMA always reads the MMCHS_DATA.
   - DMA4_CCRi[15:14] DST_AMODE = 0x1: Post-incremented address mode
   - DMA4_CCRi[16] CONST_FILL_ENABLE = 0x0: Constant fill mode is disabled.
   - DMA4_CCRi[17] TRANSPARENT_COPY_ENABLE = 0x0: Transparent copy mode is disabled.
   - DMA4_CCRi[18] BS = 0x1: Packet mode with FS = 0x1
   - DMA4_CCRi[20:19] SYNCHRO_CONTROL_UPPER = 0x1: Two MSB of MMC1_DMA_RX(62)
   - DMA4_CCRi[22] SUPERVISOR = 0x0: Supervisor mode is disabled.
   - DMA4_CCRi[23] PREFETCH = 0x0: Prefetch mode is disabled; cannot prefetch a constant adressing source.
   - DMA4_CCRi[24] SEL_SRC_DST_SYNC = 0x1: Transfer is triggered by the source. The packet element number is specified in the DMA4_CSFI register.
   - DMA4_CCRi[25] BUFFERING_DISABLE = 0x0
   - DMA4_CCRi[26] WRITE_PRIORITY = 0x0: Channel has low priority on write side during the arbitration process.
   - DMA4_CCRi[31:27] RESERVED = 0x0: Write 0s for future compatibility.
3. Channel parameters: DMA4_CENi, DMA4_CFNi, DMA4_CSSAi, DMA4_CDSAi, DMA4_CSEi, DMA4_CSFi, DMA4_CDEi
   - DMA4_CENi[23:0] CHANNEL_ELMNT_NBR = 4096: 4096 elements
   - DMA4_CFNi[15:0] CHANNEL_FRAME_NBR = 1: One frame
   - DMA4_CSSAi[31:0] SRC_START_ADRS = 0x4809c120: Channel source start address = MMCHS_DATA
   - DMA4_CDSAi[31:0] DST_START_ADRS= X: Channel destination start address in external DRAM
   - DMA4_CSEi[31:0] CHANNEL_SRC_ELMNT_INDEX = 1: Channel source element index
   - DMA4_CSFi[15:0] 16BIT_PKT_ELNT_NBR = 0x200: Packet size = number odd data available in MMC after DMA req
   - DMA4_CDEi[15:0] CHANNEL_DST_ELMNT_INDEX = 0x1: Channel destination element Index
4. Interrupt management: DMA4_CICRi and DMA4_IRQENABLE_Lj
   - DMA4_CICRi[31:0] = 0x00000008: Enables the end-of-frame interrupt and disables others (the value of these registers is unknown after a reset; it is necessary to clear other bits)
   - DMA4_IRQENABLE_L0[31:0] = 0xi: Interrupt of channel i is unmasked on IRQ line L0.
5. Launch transfer: DMA4_CCRi[7] ENABLE = 0x1: Enables the channel

Table 9-10 lists the DMA channel 2 registers after a first transfer.

Table 9-10. Registers Print

Register Name	Address	Value	Value Description
DMA4_IRQENABLE_L0	0x48056018	0x00000003	Interrupt of channel 2 is unmasked on IRQ line L0
DMA4_CICR2	0x48056148	0x00000008	End-of-frame interrupt enabled
DMA4_CCR2	0x48056140	0x010c403e	Channel control register

Table 9-10. Registers Print (continued)

Register Name	Address	Value	Value Description
DMA4_CSDP2	0x48056150	0x0001c003	Channel source destination parameters
DMA4_CEN2	0x48056154	0x00000080	Channel element number
DMA4_CFN2	0x48056158	0x00000001	Channel frame number
DMA4_CSSA2	0x4805615c	0x4809c120	Channel source start address
DMA4_CDSA2	0x48056160	0x8071aafc	Channel destination start address
DMA4_CSFI2	0x48056168	0x00000080	Packet size
DMA4_CSAC2	0x48056174	0x49022000	Source address counter (read only)
DMA4_CDAC2	0x48056178	0x8071acfc	Destination address counter (read only)
DMA4_CCEN2	0x4805617c	0x00000080	Channel current transferred element number in the current frame (read only)
DMA4_CCFN12	0x48056180	0x00000000	Channel current transferred frame number in the current transfer (read only)
DMA4_CLNK_CTRL2	0x48056144	0x00000000	Channel link control register: No link

9.7 SDMA Registers Manual

This section provides a global view of the memory mapping as seen from the MPU for SDMA. Table 9-11 show the DMA register base address.

9.7.1 SDMA Instance Summary

Table 9-11. SDMA Instances Summary

Module Name	Base Address	Size
SDMA	0x4805 6000	4K bytes

9.7.2 SDMA Register Summary

Table 9-12 lists all DMA4 controller registers and their physical addresses. Table 9-13 through Table 9-67 describe the individual register bits.

Index *i* represents the logical channel number (i = 0 to 31). The offset address for some registers is calculated from channel *c* number. For example, register SDMA.DMA4_CCR10 (channel 10) has an offset address of 10*0x60 = 0x3C0, and so a physical address of 0x4800 A080 + 0x3C0 = 0x4800 A440.

Index *j* represents the interrupt line number (j = 0 to 3) The offset address for some registers is calculated from channel *c*number. For example, register SDMA.DMA4_IRQSTATUS_L3 (line 3) has an offset address of 3*0x4 = 0xC, and so a physical address of 0x4800 A008 + 0xC = 0x4800 A014.

Table 9-12. SDMA Register Summary

Register Name	Type	Register Width (Bits)	Address Offset	SDMA Physical Address
DMA4_REVISION	R	32	0x0000 0000	0x4805 6000
DMA4_IRQSTATUS_Lj	RW	32	0x0000 0008 + (j * 0x4)	0x4805 6008 + (j * 0x4)
DMA4_IRQENABLE_Lj	RW	32	0x0000 0018 + (j * 0x4)	0x4805 6018 + (j * 0x4)
DMA4_SYSSTATUS	R	32	0x0000 0028	0x4805 6028
DMA4_OCP_SYSCONFIG	RW	32	0x0000 002C	0x4805 602C
DMA4_CAPS_0	R	32	0x0000 0064	0x4805 6064
DMA4_CAPS_2	R	32	0x0000 006C	0x4805 606C
DMA4_CAPS_3	R	32	0x0000 0070	0x4805 6070
DMA4_CAPS_4	R	32	0x0000 0074	0x4805 6074
DMA4_GCR	RW	32	0x0000 0078	0x4805 6078
DMA4_CCRi	RW	32	0x0000 0080 + (i * 0x60)	0x4805 6080 + (i * 0x60)
DMA4_CLNK_CTRLi	RW	32	0x0000 0084 + (i * 0x60)	0x4805 6084 + (i * 0x60)
DMA4_CICRi	RW	32	0x0000 0088 + (i * 0x60)	0x4805 6088 + (i * 0x60)
DMA4_CSRi	RW	32	0x0000 008C + (i * 0x60)	0x4805 608C + (i * 0x60)
DMA4_CSDPi	RW	32	0x0000 0090 + (i * 0x60)	0x4805 6090 + (i * 0x60)
DMA4_CENi	RW	32	0x0000 0094 + (i * 0x60)	0x4805 6094 + (i * 0x60)
DMA4_CFNi	RW	32	0x0000 0098 + (i * 0x60)	0x4805 6098 + (i * 0x60)
DMA4_CSSAi	RW	32	0x0000 009C + (i * 0x60)	0x4805 609C + (i * 0x60)
DMA4_CDSAi	RW	32	0x0000 00A0 + (i * 0x60)	0x4805 60A0 + (i * 0x60)
DMA4_CSEIi	RW	32	0x0000 00A4 + (i * 0x60)	0x4805 60A4 + (i * 0x60)
DMA4_CSFIi	RW	32	0x0000 00A8 + (i * 0x60)	0x4805 60A8 + (i * 0x60)
DMA4_CDEIi	RW	32	0x0000 00AC + (i * 0x60)	0x4805 60AC + (i * 0x60)
DMA4_CDFIi	RW	32	0x0000 00B0 + (i * 0x60)	0x4805 60B0 + (i * 0x60)
DMA4_CSACi	R	32	0x0000 00B4 + (i * 0x60)	0x4805 60B4 + (i * 0x60)
DMA4_CDACi	RW	32	0x0000 00B8 + (i * 0x60)	0x4805 60B8 + (i * 0x60)
DMA4_CCENi	R	32	0x0000 00BC + (i * 0x60)	0x4805 60BC + (i * 0x60)
DMA4_CCFNi	R	32	0x0000 00C0 + (i * 0x60)	0x4805 60C0 + (i * 0x60)
DMA4_COLORi	RW	32	0x0000 00C4 + (i * 0x60)	0x4805 60C4 + (i * 0x60)

9.7.3 SDMA Register Description

This section describes the registers used in the DMA4 controller.

Note: Some registers have no reset value (marked with -) because of hardware implementation in memory. Software must ensure the correct programming of these registers, if needed.

The shadow registers are used to read run time registers such as CCEN, CCFN, CDAC, or CSAC. Typically, when accessed in 8-bit or 16-bit access for two consecutive accesses, the value of the previous registers may change. This shadow register is used to hold the whole value to allow the next access to recover the remaining 24 bits or 16 bits.

The CSAC, CDAC, CCEN, and CCFN registers must be written or read in a way that enables the LSByte; otherwise, the shadow registers do not update the register.

There is no issue for 32-bit read-write transactions. For 16-bit transactions, read or write the LSByte first to enable the register update.

Table 9-13 through Table 9-67 describe the DMA register bits.

Table 9-13. DMA4_REVISION

Address Offset	0x0000 0000		
Physical Address	0x4805 6000	**Instance**	SDMA
Description	This register contains the DMA revision code		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																								REV							

Bits	Field Name	Description	Type	Reset
31:8	RESERVED	Reserved. Write 0s for future compatibility. Read returns 0.	RW	0x000000
7:0	REV	[7:4] DMA4 major revision code [3:0] DMA4 minor revision code	R	TI internal data

Table 9-14. Register Call Summary for Register DMA4_REVISION

SDMA Registers Manual

- SDMA Register Summary: [0]

Table 9-15. DMA4_IRQSTATUS_Lj

Address Offset	0x0000 0008 + (j * 0x4)	**Index**	j = 0 to 3
Physical Address	0x4805 6008 + (j * 0x4)	**Instance**	SDMA
Description	The interrupt status register regroups all the status of the DMA4 channels that can generate an interrupt over line Lj.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CH_31_0_Lj																															

Bits	Field Name	Description	Type	Reset
31:0	CH_31_0_Lj	Channel 31 Interrupt on Lj: When an interrupt is seen on the line Lj the status of a interrupting channel i is read in the bit field i. Read 0x0: Channel Interrupt Lj false Write 0x0: Channel Interrupt Lj status bit unchanged Read 0x1: Channel Interrupt Lj true (pending) Write 0x1: Channel Interrupt Lj status bit is reset	RW	0x00000000

Table 9-16. Register Call Summary for Register DMA4_IRQSTATUS_Lj

SDMA Module Integration • Interrupts to the MPU Subsystem: [0] [1] [2] [3]
SDMA Functional Description • Interrupt Generation: [4]
SDMA Registers Manual • SDMA Register Summary: [5]

Table 9-17. DMA4_IRQENABLE_Lj

Address Offset	0x0000 0018 + (j * 0x4)	**Index**	j = 0 to 3
Physical Address	0x4805 6018 + (j * 0x4)	**Instance**	SDMA
Description	The interrupt enable register allows to mask/unmask the module internal sources of interrupt, on line Lj		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CH_31_0_Lj_EN																															

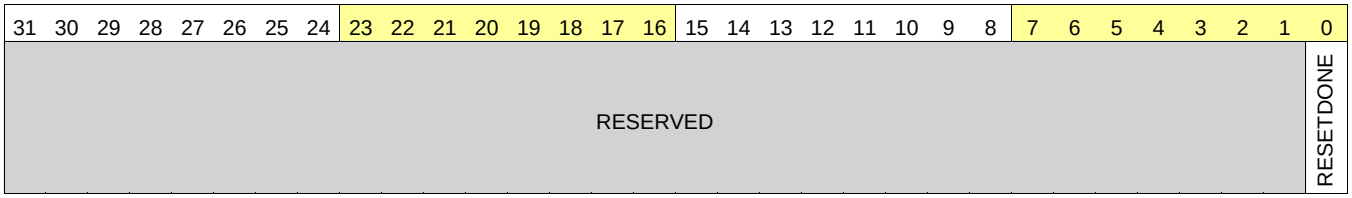
Bits	Field Name	Description	Type	Reset
31:0	CH_31_0_Lj_EN	Channel Interrupt on Lj mask/unmask : to Mask/Unmask a channel i interrupt on Lj the user writes 0/1 on the bit field i. 0x0: Channel Interrupt Lj is masked 0x1: Channel Interrupt Lj generates an interrupt when it occurs	RW	0x00000000

Table 9-18. Register Call Summary for Register DMA4_IRQENABLE_Lj

SDMA Module Integration • Interrupts to the MPU Subsystem: [0] [1] [2] [3]
SDMA Functional Description • Interrupt Generation: [4]
SDMA Basic Programming Model • Setup Configuration: [5]
SDMA Use Cases and Tips • SDMA Configuration to Transfer Data Between the McBSP and External DRAM: [6] • SDMA Configuration to Transfer Data Between MMC and External DRAM: [7]
SDMA Registers Manual • SDMA Register Summary: [8]

Table 9-19. DMA4_SYSSTATUS

Address Offset	0x0000 0028		
Physical Address	0x4805 6028	**Instance**	SDMA
Description	The register provides status information about the module excluding the interrupt status information (see interrupt status register)		
Type	R		



Bits	Field Name	Description	Type	Reset
31:1	RESERVED	Reserved for module-specific status information	RW	0x00000000
0	RESETDONE	Internal reset monitoring 0x0: Internal module reset is on-going 0x1: Reset completed	R	0x1

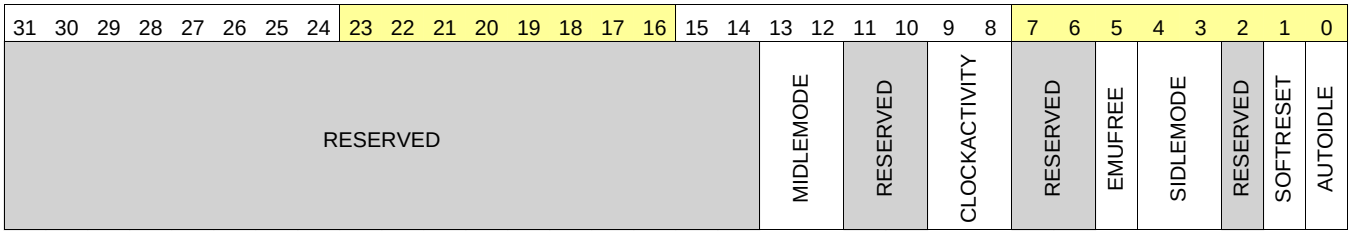
Table 9-20. Register Call Summary for Register DMA4_SYSSTATUS

SDMA Registers Manual

- SDMA Register Summary: [0]

Table 9-21. DMA4_OCP_SYSCONFIG

Address Offset	0x0000 002C		
Physical Address	0x4805 602C	**Instance**	SDMA
Description	This register controls the various parameters of the OCP interface		
Type	RW		



Bits	Field Name	Description	Type	Reset
31:14	RESERVED	Write 0s for future compatibility, Reads return 0	RW	0x00000
13:12	MIDLEMODE	Read write power management, standby/wait control 0x0: Force-standby: MStandby is asserted only when all the DMA channels are disabled. 0x1: No-Standby: MStandby is never asserted. 0x2: Smart-Standby: MStandby is asserted if at least one of the following two conditions is satisfied: 1. All the channels are disabled, OR 2. There is no non-synchronized channel enabled AND [if hardware synchronized channel is enabled, then no DMA request input is asserted and no requests are pending to be serviced]. 0x3: reserved for second smart-standby mode if needed	RW	0x0
11:10	RESERVED	Reserved for clocks activities extension	RW	0x0

Bits	Field Name	Description	Type	Reset
9:8	CLOCKACTIVITY	Clocks activities during wake-up Bit 8: OCP interface clock 0 OCP clock can be switched-off Bit 9: Functional clock 0 Functional clock can be switched-off	R	0x0
7:6	RESERVED	Reserved. Write 0s for future compatibility. Read returns 0.	RW	0x0
5	EMUFREE	Enable sensitivity to MSuspend 0x0: DMA4 freezes its internal logic upon MSuspend assertion 0x1: DMA4 ignores the MSuspend input	RW	0x0
4:3	SIDLEMODE	Configuration port power management, Idle req/ack control 0x0: Force-idle. An idle request is acknowledged unconditionally 0x1: No-idle. An idle request is never acknowledged 0x2: Smart-idle. Idle acknowledge is given by DMA4 if all of the following conditions are true: • All channels are disabled. • If a hardware-synchronized channel is enabled, no DMA request input is asserted and no request are pending to be serviced. • All transactions are complete on all the DMA ports. • No interrupts are pending to be serviced. 0x3: Reserved. Do not use.	RW	0x0
2	RESERVED	Write 0s for future compatibility, Reads return 0	RW	0x0
1	SOFTRESET	Software reset. Set this bit to 1 to trigger a module reset. The bit is automatically reset by the hardware. During reads, it always returns 0. 0x0: No effect 0x1: Reset	RW	0x0
0	AUTOIDLE	Internal OCP clock gating strategy 0x0: OCP clock is free running 0x1: Automatic OCP clock gating strategy is applied, based on the OCP interface activity.	RW	0x0

Table 9-22. Register Call Summary for Register DMA4_OCP_SYSCONFIG

SDMA Module Integration • Resets: [0]
SDMA Functional Description • Interconnect Clock Auto-Idle: [1] • Automatic Standby Mode: [2]
SDMA Registers Manual • SDMA Register Summary: [3]

Table 9-23. DMA4_CAPS_0

Address Offset	0x0000 0064		
Physical Address	0x4805 6064	**Instance**	SDMA
Description	DMA Capabilities Register 0 LSW		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED												CONST_FILL_CPBLTY	TRANSPARENT_BLT_CPBLTY	RESERVED																			

Bits	Field Name	Description	Type	Reset
31:20	RESERVED	Reserved. Write 0s for future compatibility. Read returns 0.	RW	0x000
19	CONST_FILL_CPBLTY	Constant_Fill_Capability 0x0: No LCH supports constant fill copy 0x1: any LCH supports constant fill copy	R	0x1
18	TRANSPARENT_BLT_CPBLTY	Transparent_BLT_Capability 0x0: No LCH supports transparent BLT copy 0x1: any LCH supports transparent BLT copy	R	0x1
17:0	RESERVED	Reserved. Write 0s for future compatibility. Read returns 0.	RW	0x00000

Table 9-24. Register Call Summary for Register DMA4_CAPS_0

SDMA Registers Manual

- SDMA Register Summary: [0]

Table 9-25. DMA4_CAPS_2

Address Offset	0x0000 006C		
Physical Address	0x4805 606C	**Instance**	SDMA
Description	DMA Capabilities Register 2		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																								SEPARATE_SRC_AND_DST_	DST_DOUBLE_INDEX_ADRS_	DST_SINGLE_INDEX_ADRS_	DST_POST_INCRMNT_ADRS_	DST_CONST_ADRS_CPBLTY	SRC_DOUBLE_INDEX_ADRS_	SRC_SINGLE_INDEX_ADRS_	SRC_POST_INCREMENT_ADRS_	SRC_CONST_ADRS_CPBLTY

Bits	Field Name	Description	Type	Reset
31:9	RESERVED	Reserved. Write 0s for future compatibility. Read returns 0.	RW	0x000000
8	SEPARATE_SRC_AND_DST_INDEX_CPBLTY	Separate_source/destination_index_capability 0x0: Does not support separate src/dst index for 2D addressing 0x1: Supports separate src/dest index for 2D addressing	R	0x1
7	DST_DOUBLE_INDEX_ADRS_CPBLTY	Destination_double_index_address_capability 0x0: Does not support double index address mode on the destination port 0x1: Supports double index address mode on the destination port	R	0x1
6	DST_SINGLE_INDEX_ADRS_CPBLTY	Destination_single_index_address_capability 0x0: Does not support single index address mode on the destination port 0x1: Supports single index address mode on the destination port	R	0x1
5	DST_POST_INCRMNT_ADRS_CPBLTY	Destination_post_increment_address_capability 0x0: Does not supports post-increment address mode in the destination port 0x1: Supports post-increment address mode in the destination port	R	0x1
4	DST_CONST_ADRS_CPBLTY	Destination_constant_address_capability 0x0: Does not supports constant address mode in the destination port 0x1: Supports constant address mode in the destination port	R	0x1
3	SRC_DOUBLE_INDEX_ADRS_CPBLTY	Source_double_index_address_capability 0x0: Does not support double index address mode on the source port 0x1: Supports double index address mode on the source port	R	0x1
2	SRC_SINGLE_INDEX_ADRS_CPBLTY	Source_single_index_address_capability 0x0: Does not support single index address mode on the source port 0x1: Supports single index address mode in the source port	R	0x1
1	SRC_POST_INCREMENT_ADRS_CPBLTY	Source_post_increment_address_capability 0x0: Does not supports post-increment address mode in the source port 0x1: Supports post-increment address mode in the source port	R	0x1
0	SRC_CONST_ADRS_CPBLTY	Source_constant_address_capability 0x0: Does not supports constant address mode in the source port 0x1: Supports constant address mode in the source port	R	0x1

Table 9-26. Register Call Summary for Register DMA4_CAPS_2

SDMA Registers Manual

- SDMA Register Summary: [0]

Table 9-27. DMA4_CAPS_3

Address Offset	0x0000 0070		
Physical Address	0x4805 6070	**Instance**	SDMA
Description	DMA Capabilities Register 3		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																								BLOCK_SYNCHR_CPBLTY	PKT_SYNCHR_CPBLTY	CHANNEL_CHAINING_CPBLTY	CHANNEL_INTERLEAVE_CPBLTY	RESERVED		FRAME_SYNCHR_CPBLTY	ELMNT_SYNCHR_CPBLTY

Bits	Field Name	Description	Type	Reset
31:8	RESERVED	Reserved. Write 0s for future compatibility. Read returns 0.	RW	0x000000
7	BLOCK_SYNCHR_CPBLTY	Block_synchronization_capability 0x0: Does not support synchronization transfer on block boundary 0x1: Supports synchronization transfer on block boundary	R	0x1
6	PKT_SYNCHR_CPBLTY	Packet_synchronization_capability 0x0: Does not support synchronization transfer on packet boundary 0x1: Supports synchronization transfer on packet boundary	R	0x1
5	CHANNEL_CHAINING_CPBLTY	Channel_Chaining_capability 0x0: Does not support Channel Chaining capability 0x1: Supports Channel Chaining capability	R	0x1
4	CHANNEL_INTERLEAVE_CPBLTY	Channel_interleave_capability 0x0: Does not support Channel interleave capability 0x1: Supports Channel_interleave capability	R	0x1
3:2	RESERVED		RW	0x0
1	FRAME_SYNCHR_CPBLTY	Frame_synchronization_capability 0x0: Does not support synchronization transfer on Frame boundary 0x1: Supports synchronization transfer on Frame boundary	R	0x1
0	ELMNT_SYNCHR_CPBLTY	Element_synchronization_capability 0x0: Does not support synchronization transfer on Element boundary 0x1: Supports synchronization transfer on Element boundary	R	0x1

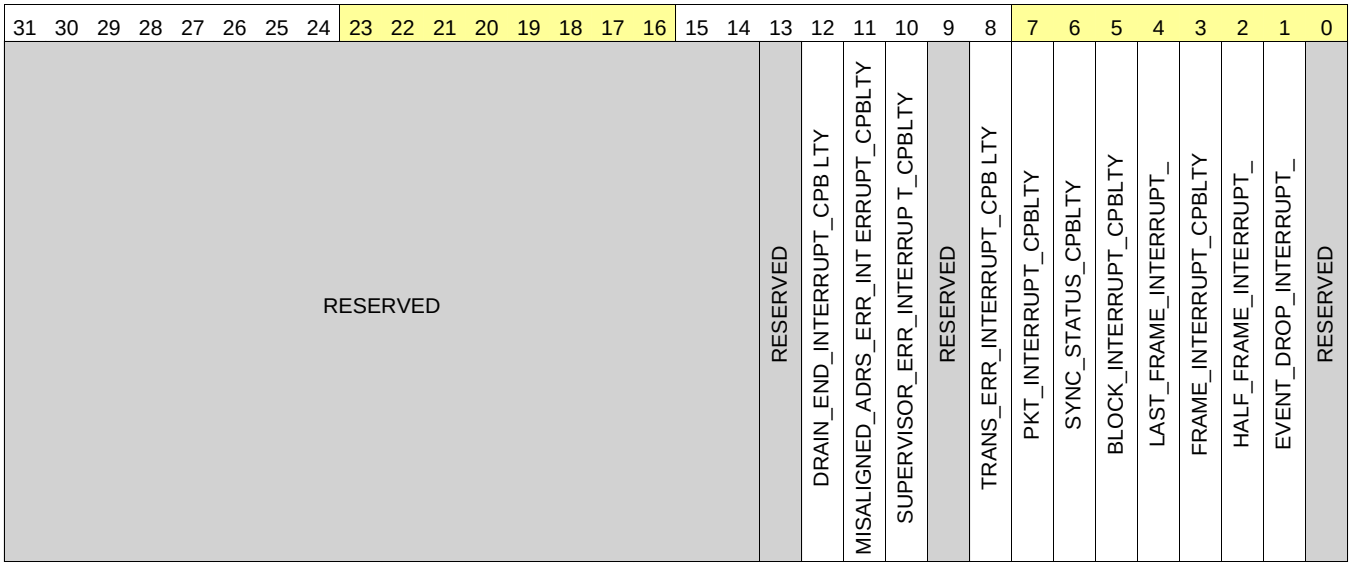
Table 9-28. Register Call Summary for Register DMA4_CAPS_3

SDMA Registers Manual

- SDMA Register Summary: [0]

Table 9-29. DMA4_CAPS_4

Address Offset	0x0000 0074		
Physical Address	0x4805 6074	**Instance**	SDMA
Description	DMA Capabilities Register 4		
Type	R		



Bits	Field Name	Description	Type	Reset
31:14	RESERVED	Reserved. Write 0s for future compatibility. Read returns 0.	RW	0x000000
13	RESERVED	Reserved	R	1
12	DRAIN_END_INTERRUPT_CPB LTY	Drain end detection capability	R	1
11	MISALIGNED_ADRS_ERR_INT ERRUPT_CPBLTY	Misaligned error detection capability	R	1
10	SUPERVISOR_ERR_INTERRUP T_CPBLTY	Supervisor error detection capability	R	1
9	RESERVED	Reserved for non-GP devices	R	1
8	TRANS_ERR_INTERRUPT_CPB LTY	Transaction error detection capability	R	1
7	PKT_INTERRUPT_CPBLTY	End of Packet detection capability 0x0: Does not support end of packet interrupt generation capability 0x1: Supports end of packet interrupt generation capability	R	0x1
6	SYNC_STATUS_CPBLTY	Sync_status_capability 0x0: Does not support synchronized transfer status bit generation 0x1: Supports synchronized transfer status bit generation	R	0x1
5	BLOCK_INTERRUPT_ CPBLTY	End of block detection capability. 0x0: Does not support end of block interrupt generation capability 0x1: Supports end of block interrupt generation capability	R	0x1
4	LAST_FRAME_INTERRUPT_ CPBLTY	Start of last frame detection capability. 0x0: Does not support last frame interrupt generation capability 0x1: Supports last frame interrupt generation capability	R	0x1

Bits	Field Name	Description	Type	Reset
3	FRAME_INTERRUPT_CPBLTY	End of frame detection capability. 0x0: Does not support end of frame interrupt generation capability 0x1: Supports end of frame interrupt generation capability	R	0x1
2	HALF_FRAME_INTERRUPT_CPBLTY	Detection capability of the half of frame end. 0x0: Does not support half of frame interrupt generation capability 0x1: Supports half of frame interrupt generation capability	R	0x1
1	EVENT_DROP_INTERRUPT_CPBLTY	Request collision detection capability. 0x0: Does not support event drop interrupt generation capability 0x1: Supports event drop interrupt generation capability	R	0x1
0	RESERVED	Reserved. Write 0s for future compatibility. Read returns 0.	RW	0x0

Table 9-30. Register Call Summary for Register DMA4_CAPS_4

SDMA Registers Manual

- SDMA Register Summary: [0]

Table 9-31. DMA4_GCR

Address Offset	0x0000 0078		
Physical Address	0x4805 6078	**Instance**	SDMA
Description	FIFO sharing between high and low priority channel. The Maximum per channel FIFO depth is bounded by the low and high channel FIFO budget. The high respectively low priority channels maximum burst size must be less than the min (high respectively low priority channel FIFO budget , per channel maximum FIFO depth)		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								ARBITRATION_RATE								HI_LO_FIFO_BUDGET		HI_THREAD_RESERVED		RESERVED				MAX_CHANNEL_FIFO_DEPTH							

Bits	Field Name	Description	Type	Reset
31:24	RESERVED	Reserved. Write 0s for future compatibility. Read returns 0.	RW	0x00
23:16	ARBITRATION_RATE	Arbitration switching rate between prioritized and regular channel queues	RW	0x01
15:14	HI_LO_FIFO_BUDGET	Allow to have a separate Global FIFO budget for high and low priority channels. For Hi priority Channel: (Per_channel_Maximum FIFO depth + 1) x Number of active High priority Channel =< High Budget FIFO For Low priority channel: (Per_channel_Maximum FIFO depth + 1) x Number of active Low priority Channel =< Low Budget FIFO 0x0: no fixed budget for neither higher nor lower priority channel 0x1: 75% of FIFO for low priority and 25% for high priority channels 0x2: 25% of FIFO for low priority and 75% for high priority channels 0x3: 50% of FIFO for low priority and 50% for high priority channels	RW	0x0
13:12	HI_THREAD_ RESERVED	Allow thread reservation for high priority channel on both read and write ports. 0x0: No ThreadID is reserved on the Read Port for high priority channels. No ThreadID is reserved on the Write Port for high priority channels. 0x1: Read Port ThreadID 0 is reserved for high priority channels. Write Port ThreadID 0 is reserved for high priority channels. 0x2: Read port ThreadID 0 and ThreadID 1 are reserved for high priority channels. Write Port ThreadID 0 is reserved for high priority channels. 0x3: Read PortThreadID 0, ThreadID 1 and ThreadID 2 are reserved for high priority channels. Write Port ThreadID 0 is reserved for high priority channels.	RW	0x0
11:8	RESERVED	Reserved. Write 0s for future compatibility. Read returns 0.	RW	0x0
7:0	MAX_CHANNEL_ FIFO_DEPTH	Maximum FIFO depth allocated to one logical channel. Maximum FIFO depth can not be 0x0. It should be at least 0x1 or greater. Note that If channel limit is less than destination burst size enough data will not be accumulated in the data FIFO and it will never be sent out on the WR port. The burst size should be less than the FIFO limit specified in this bit field.	RW	0x10

Table 9-32. Register Call Summary for Register DMA4_GCR

SDMA Functional Description
- Logical Channel Transfer Overview: [0] [1] [2]
- FIFO Queue Memory Pool: [3]
- Thread Budget Allocation: [4] [5] [6] [7] [8] [9] [10] [11] [12] [13] [14]
- FIFO Budget Allocation: [15] [16]

SDMA Basic Programming Model
- Setup Configuration: [17]
- Concurrent Software and Hardware Synchronization: [18] [19]

SDMA Registers Manual
- SDMA Register Summary: [20]

Table 9-33. DMA4_CCRi

Address Offset	0x0000 0080 + (i* 0x60)	**Index**	i = 0 to 31
Physical Address	0x4805 6080 + (i* 0x60)	**Instance**	SDMA
Description	Channel Control Register		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED					WRITE_PRIORITY	BUFFERING_DISABLE	SEL_SRC_DST_SYNC	PREFETCH	SUPERVISOR	RESERVED	SYNCHRO_CONTROL_UPPER		BS	TRANSPARENT_COPY_ENABLE	CONST_FILL_ENABLE	DST_AMODE		SRC_AMODE		RESERVED	WR_ACTIVE	RD_ACTIVE	SUSPEND_SENSITIVE	ENABLE	READ_PRIORITY	FS	SYNCHRO_CONTROL				

Bits	Field Name	Description	Type	Reset
31:27	RESERVED	Reserved. Write 0s for future compatibility. Read returns 0.	RW	0x00
26	WRITE_PRIORITY	Channel priority on the Write side 0x0: Channel has low priority on the Write side during the arbitration process 0x1: Channel has high priority on Write sided during the arbitration process	RW	0x0
25	BUFFERING_DISABLE	This bit allows to disable the default buffering functionality when transfer is source synchronized. 0x0: buffering is enable across element/packet when source is synchronized to element, packet, frame or blocks 0x1: buffering is disabled across element/packet when source is synchronized to element, packet, frame or blocks	RW	0x-
24	SEL_SRC_DST_SYNC	Specifies that element, packet, frame or block transfer (depending on CCR.bs and CCR.fs) is triggered by the source or the destination on the DMA request 0x0: Transfer is triggered by the destination. If synch on packet the packet element number is specified in the CDFI register 0x1: Transfer is triggered by the source. If synchronized on packet the packet element number is specified in the CSFI register	RW	0x-
23	PREFETCH	Enables the prefetch mode 0x0: Prefetch mode is disabled. When Sel_Src_Dst_Sync=1 transfers are buffered and pipelined between DMA requests 0x1: Prefetch mode is enabled. Prefetch mode is active only when destination is synchronized. It is SW user responsibility not to have at the same time Prefetch=1 when Sel_Src_Dst_Sync=1. This mode is not supported	RW	0x0
22	SUPERVISOR	Enables the supervisor mode 0x0: Supervisor mode is disabled 0x1: Supervisor mode is enabled	RW	0x0
21	RESERVED	Reserved for non-GP devices	RW	0x0
20:19	SYNCHRO_CONTROL_ UPPER	Channel Synchronization control upper (used in conjunction with the 5 bits of synchro channel DMA4_CCRi[4:0]) Used in conjunction, as two msb, with the five bits of the synchro channel bit field.	RW	0x0
18	BS	Block synchronization This bit used with the fs to see how the DMA request is serviced in a synchronized transfer	RW	0x-

Bits	Field Name	Description	Type	Reset
17	TRANSPARENT_COPY_ENABLE	Transparent copy enable 0x0: Transparent copy mode is disabled 0x1: Transparent copy mode is enabled	RW	0x-
16	CONST_FILL_ENABLE	Constant fill enable 0x0: Constant fill mode is disabled 0x1: Constant fill mode is enabled	RW	0x0
15:14	DST_AMODE	Selects the addressing mode on the Write Port of a channel. 0x0: Constant address mode 0x1: Post-incremented address mode 0x2: Single index address mode 0x3: Double index address mode	RW	0x-
13:12	SRC_AMODE	Selects the addressing mode on the Read Port of a channel. 0x0: Constant address mode 0x1: Post-incremented address mode 0x2: Single index address mode 0x3: Double index address mode	RW	0x-
11	RESERVED	Reserved. Write 0s for future compatibility. Read returns 0.	RW	0x0
10	WR_ACTIVE	Indicates if the channel write context is active or not 0x0: Channel is not active on the write port 0x1: Channel is active on the write port	R	0x0
9	RD_ACTIVE	Indicates if the channel read context is active or not 0x0: Channel is not active on the read port 0x1: Channel is currently active on the read port	R	0x0
8	SUSPEND_SENSITIVE	Logical channel suspend enable bit 0x0: The channel ignores the MSuspend even if EMUFree is set to 0. 0x1: If EMUFree is set to 0 and MSuspend comes in then all current OCP services (single transaction or burst transaction as specified in the corresponding CSDP register) have to be completed before stopping processing any more transactions	RW	0x0
7	ENABLE	Logical channel enable. It is SW responsibility to clear the CSR register and the IRQSTATUS bit for the different interrupt lines before enabling the channel. 0x0: The logical channel is disabled 0x1: The logical channel is enabled	RW	0x0
6	READ_PRIORITY	Channel priority on the read side 0x0: Channel has low priority on the Read side during the arbitration process 0x1: Channel has high priority on read sided during the arbitration process	RW	0x0
5	FS	Frame synchronization This bit used with the BS to see how the DMA request is serviced in a synchronized transfer FS=0 and BS=0: An element is transferred once a DMA request is made. FS=0 and BS=1: An entire block is transferred once a DMA request is made. FS=1 and BS=0: An entire frame is transferred once a DMA request is made. FS=1 and BS=1: A packet is transferred once a DMA request is made. All these different transfers can be interleaved on the port with other DMA requests.	RW	0x-

Bits	Field Name	Description	Type	Reset
4:0	SYNCHRO_CONTROL	Channel synchronization control This bit field used with the second_level_synchro_control_upper (as two msb) 0000000 : Is reserved for non synchronized LCH transfer xxxxxxx (from 1 to 127)There are 127 possible DMA request to assign to any LCH. **Note:** The channel synchronization control registers are 1-based. For example, to enable the S_DMA_1 request, SDMA.DMA4_CCR[4:0] SYNCHRO_CONTROL must be set to 0x2 (DMA request number + 1).	RW	0x0

Table 9-34. Register Call Summary for Register DMA4_CCRi

SDMA Functional Description

- Logical Channel Transfer Overview: [0] [1] [2]
- Addressing Modes: [3]
- Software Synchronization: [4] [5] [6]
- Hardware Synchronization: [7] [8] [9] [10] [11] [12] [13] [14] [15] [16] [17] [18]
- Thread Budget Allocation: [19] [20]
- Reprogramming an Active Channel: [21] [22] [23] [24]
- Interrupt Generation: [25]
- Packet Synchronization: [26] [27] [28] [29] [30]
- Graphics Acceleration Support: [31]
- Supervisor Modes: [32]
- Disabling a Channel During Transfer: [33]
- FIFO Draining Mechanism: [34] [35] [36] [37] [38]
- Reset: [39]

SDMA Basic Programming Model

- Software-Triggered (Nonsynchronized) Transfer: [40] [41] [42] [43] [44] [45] [46] [47] [48]
- Hardware-Synchronized Transfer: [49] [50] [51] [52] [53] [54] [55] [56] [57] [58] [59] [60] [61] [62] [63] [64] [65] [66] [67] [68] [69] [70] [71] [72] [73] [74] [75] [76] [77] [78] [79] [80] [81] [82] [83] [84]
- Concurrent Software and Hardware Synchronization: [85] [86]
- Chained Transfer: [87]
- °90 Clockwise Image Rotation: [88] [89] [90] [91] [92] [93] [94]
- Graphic Operations: [95] [96] [97] [98]

SDMA Use Cases and Tips

- SDMA Configuration to Transfer Data Between the McBSP and External DRAM: [99] [100] [101] [102] [103] [104] [105] [106] [107] [108] [109] [110] [111] [112] [113] [114] [115] [116] [117] [118] [119] [120]
- SDMA Configuration to Transfer Data Between MMC and External DRAM: [121] [122] [123] [124] [125] [126] [127] [128] [129] [130] [131] [132] [133] [134] [135] [136] [137] [138] [139] [140] [141] [142]

SDMA Registers Manual

- SDMA Register Summary: [143]
- SDMA Register Description: [144]

Table 9-35. DMA4_CLNK_CTRLi

Address Offset	0x0000 0084 + (i* 0x60)	**Index**	i = 0 to 31
Physical Address	0x4805 6084 + (i* 0x60)	**Instance**	SDMA
Description	Channel Link Control Register		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																ENABLE_LNK	RESERVED							NEXTLCH_ID							

Bits	Field Name	Description	Type	Reset
31:16	RESERVED	Reserved. Write 0s for future compatibility. Read returns 0.	RW	0x0000
15	ENABLE_LNK	Enables or disable the channel linking. 0x0: Channel linking mode is disabled When set on the fly to 0 the current channel will complete the transfer and stops the chain linking 0x1: Channel linking mode is enabled. The logical channel defined in the NextLCH_ID is enabled at the end of the current transfer	RW	0x0
14:5	RESERVED	Reserved. Write 0s for future compatibility. Read returns 0.	RW	0x000
4:0	NEXTLCH_ID	Defines the NextLCh_ID, which is used to build logical channel chaining queue.	RW	0x–

Table 9-36. Register Call Summary for Register DMA4_CLNK_CTRLi

SDMA Functional Description
- Chained Logical Channel Transfers: [0] [1]
- FIFO Draining Mechanism: [2]

SDMA Basic Programming Model
- Chained Transfer: [3] [4] [5] [6] [7]

SDMA Use Cases and Tips
- SDMA Configuration to Transfer Data Between the McBSP and External DRAM: [8]

SDMA Registers Manual
- SDMA Register Summary: [9]

Table 9-37. DMA4_CICRi

Address Offset	0x0000 0088 + (i* 0x60)	**Index**	i = 0 to 31
Physical Address	0x4805 6088 + (i* 0x60)	**Instance**	SDMA
Description	Channel Interrupt Control Register		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																RESERVED			DRAIN_IE	MISALIGNED_ERR_IE	SUPERVISOR_ERR_IE	RESERVED	TRANS_ERR_IE	PKT_IE	RESERVED	BLOCK_IE	LAST_IE	FRAME_IE	HALF_IE	DROP_IE	RESERVED

Bits	Field Name	Description	Type	Reset
31:16	RESERVED	Reserved. Write 0s for future compatibility. Read returns 0.	RW	0x0000
15:13	RESERVED	Reserved. Write 0s for future compatibility. Read returns 0.	RW	0x-
12	DRAIN_IE	Enables the end of draining interrupt 0x0: Disables end of channel draining interrupt 0x1: Enable end of channel draining interrupt	RW	0x0
11	MISALIGNED_ERR_IE	Enables the address misaligned error event interrupt 0x0: Disables the misaligned address error event interrupt 0x1: Enables the misaligned address error event interrupt	RW	0x-
10	SUPERVISOR_ERR_IE	Enables the supervisor transaction error event interrupt 0x0: Disables the supervisor transaction error event interrupt 0x1: Enables the supervisor transaction error event interrupt	RW	0x1
9	RESERVED	Reserved for non-GP devices	RW	0x1

Bits	Field Name	Description	Type	Reset
8	TRANS_ERR_IE	Enables the transaction error event interrupt 0x0: Disables the transaction error event interrupt 0x1: Enables the transaction error event interrupt	RW	0x-
7	PKT_IE	Enables the end of Packet interrupt 0x0: Disables the end of Packet transfer interrupt 0x1: Enables the end of Packet transfer interrupt	RW	0x-
6	RESERVED	Reserved. Write 0s for future compatibility. Read returns 0.	RW	0x0
5	BLOCK_IE	Enables the end of block interrupt 0x0: Disables the end of block interrupt 0x1: Enables the end of block interrupt	RW	0x-
4	LAST_IE	Last frame interrupt enable (start of last frame) 0x0: Disables the last frame interrupt 0x1: Enables the last frame interrupt	RW	0x-
3	FRAME_IE	Frame interrupt enable (end of frame) 0x0: Disables the end of frame interrupt 0x1: Enables the end of frame interrupt	RW	0x-
2	HALF_IE	Enables or disables the half frame interrupt. 0x0: Disables the half frame interrupt 0x1: Enables the half frame interrupt	RW	0x-
1	DROP_IE	Synchronization event drop interrupt enable (request collision) 0x0: Disables the event drop interrupt 0x1: Enables the event drop interrupt	RW	0x0
0	RESERVED	Reserved. Write 0s for future compatibility. Read returns 0.	RW	0x0

Table 9-38. Register Call Summary for Register DMA4_CICRi

SDMA Module Integration • Interrupts to the MPU Subsystem: [0] [1]
SDMA Functional Description • Interrupt Generation: [2] • FIFO Draining Mechanism: [3] • Reset: [4]
SDMA Basic Programming Model • Setup Configuration: [5]
SDMA Use Cases and Tips • SDMA Configuration to Transfer Data Between the McBSP and External DRAM: [6] [7] • SDMA Configuration to Transfer Data Between MMC and External DRAM: [8] [9]
SDMA Registers Manual • SDMA Register Summary: [10]

Table 9-39. DMA4_CSRi

Address Offset	0x0000 008C + (i* 0x60)	**Index**	i = 0 to 31
Physical Address	0x4805 608C + (i* 0x60)	**Instance**	SDMA
Description	Channel Status Register		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																RESERVED			DRAIN_END	MISALIGNED_ADRS_ERR	SUPERVISOR_ERR	RESERVED	TRANS_ERR	PKT	SYNC	BLOCK	LAST	FRAME	HALF	DROP	RESERVED

Bits	Field Name	Description	Type	Reset
31:16	RESERVED	Reserved. Write 0s for future compatibility. Read returns 0.	RW	0x0000
15:13	RESERVED	Reserved. Write 0s for future compatibility. Read returns 0.	RW	0x0
12	DRAIN_END	End of channel draining Read 0x0: Status bit unchanged Write 0x0: No drain end in the current transfer Read 0x1: The current channel draining is completed Write 0x1: Status bit is reset	RW	0x0
11	MISALIGNED_ ADRS_ERR	Misaligned address error event Read 0x0: No address error Write 0x0: Status bit unchanged Read 0x1: An address error has been occurred Write 0x1: Status bit is reset	RW	0x0
10	SUPERVISOR_ERR	Supervisor transaction error event Read 0x0: No supervisor transaction error Write 0x0: Status bit unchanged Read 0x1: A supervisor transaction error has been occurred Write 0x1: Status bit is reset	RW	0x0
9	RESERVED	Reserved for non-GP devices	RW	0x0
8	TRANS_ERR	Transaction error event Read 0x0: No transaction error Write 0x0: Status bit unchanged Read 0x1: A transaction error has been occurred Write 0x1: Status bit is reset	RW	0x0
7	PKT	End of Packet transfer Read 0x0: The current packet transfer has not been finished Write 0x0: Status bit unchanged Read 0x1: The current packet has been transferred Write 0x1: Status bit is reset	RW	0x0
6	SYNC	Synchronization status of a channel. Read 0x0: Logical channel is not scheduled or servicing a non synchronized DMA request. Write 0x0: Status bit unchanged Read 0x1: Logical channel is servicing a synchronized DMA request Write 0x1: Status bit is reset	RW	0x0
5	BLOCK	End of block event Read 0x0: The current block transfer has not been finished Write 0x0: Status bit unchanged Read 0x1: The current block has been transferred Write 0x1: Status bit is reset	RW	0x0

Bits	Field Name	Description	Type	Reset
4	LAST	Last frame (start of last frame) Read 0x0: The start of the last frame to transfer is not reached Write 0x0: Status bit unchanged Read 0x1: The start of the last frame to transfer is reached Write 0x1: Status bit is reset	RW	0x0
3	FRAME	End of frame event Read 0x0: The end of current transferred frame is not reached Write 0x0: Status bit unchanged Read 0x1: The end of current transferred frame is reached Write 0x1: Status bit is reset	RW	0x0
2	HALF	Half of frame event. Read 0x0: The half of current transferred frame is not reached Write 0x0: Status bit unchanged Read 0x1: The half of current transferred frame is reached Write 0x1: Status bit is reset	RW	0x0
1	DROP	Synchronization event drop occurred during the transfer Read 0x0: No synchronization collision Write 0x0: Status bit unchanged Read 0x1: A synchronization collision has been occurred Write 0x1: Status bit is reset	RW	0x0
0	RESERVED	Reserved. Write 0s for future compatibility. Read returns 0.	RW	0x0

Table 9-40. Register Call Summary for Register DMA4_CSRi

SDMA Module Integration • Interrupts to the MPU Subsystem: [0] [1] [2] [3] [4]
SDMA Functional Description • Interrupt Generation: [5] • FIFO Draining Mechanism: [6]
SDMA Basic Programming Model • Setup Configuration: [7]
SDMA Registers Manual • SDMA Register Summary: [8]

Table 9-41. DMA4_CSDPi

Address Offset	0x0000 0090 + (i* 0x60)	**Index**	i = 0 to 31
Physical Address	0x4805 6090 + (i* 0x60)	**Instance**	SDMA
Description	Channel Source Destination Parameters		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED										SRC_ENDIAN	SRC_ENDIAN_LOCK	DST_ENDIAN	DST_ENDIAN_LOCK	WRITE_MODE		DST_BURST_EN		DST_PACKED	RESERVED				SRC_BURST_EN	SRC_PACKED	RESERVED					DATA_TYPE	

Bits	Field Name	Description	Type	Reset
31:22	RESERVED	Reserved. Write 0s for future compatibility. Read returns 0.	RW	0x000
21	SRC_ENDIAN	Channel source endianness control	RW	0x-
		0x0: Source has Little Endian type		
		0x1: Source has Big Endian type		
20	SRC_ENDIAN_LOCK	Endianness Lock	RW	0x-
		0x0: Endianness adapt		
		0x1: Endianness lock		
19	DST_ENDIAN	Channel Destination endianness control	RW	0x-
		0x0: Destination has Little Endian type		
		0x1: Destination has Big Endian type		
18	DST_ENDIAN_LOCK	Endianness Lock	RW	0x-
		0x0: Endianness adapt		
		0x1: Endianness lock		
17:16	WRITE_MODE	Used to enable writing mode without posting or with posting	RW	0x-
		0x0: Write nonposted (WRNP)		
		0x1: Write (Posted)		
		0x2: All transaction are mapped on the Write command as posted except for the last transaction in the transfer mapped on a Write nonposted		
		0x3: Undefined		
15:14	DST_BURST_EN	Used to enable bursting on the Write Port. Smaller burst size than the programmed burst size is also allowed	RW	0x0
		0x0: Single access		
		0x1: 16 bytes or 4x32-bit/2x64-bit burst access		
		0x2: 32 bytes or 8x32-bit/4x64-bit burst access		
		0x3: 64 bytes or 16x32-bit/8x64-bit burst access		
13	DST_PACKED	Destination receives packed data.	RW	0x-
		0x0: The destination target is nonpacked		
		0x1: The destination target is packed		
12:9	RESERVED	Reserved. Write 0s for future compatibility. Read returns 0.	RW	0x-
8:7	SRC_BURST_EN	Used to enable bursting on the Read Port. Smaller burst size than the programmed burst size is also allowed	RW	0x-
		0x0: Single access		
		0x1: 16 bytes or 4x32-bit/2x64-bit burst access		
		0x2: 32 bytes or 8x32-bit/4x64-bit burst access		
		0x3: 64 bytes or 16x32-bit/8x64-bit burst access		
6	SRC_PACKED	Source provides packed data.	RW	0x-
		0x0: The source target is nonpacked		
		0x1: The source target is packed		
5:2	RESERVED	Reserved. Write 0s for future compatibility. Read returns 0.	RW	0x-
1:0	DATA_TYPE	Defines the type of the data moved in the channel.	RW	0x-
		0x0: 8 bits scalar		
		0x1: 16 bits scalar		
		0x2: 32 bits scalar		
		0x3: Undefined		

Table 9-42. Register Call Summary for Register DMA4_CSDPi

SDMA Functional Description

- Addressing Modes: [0]
- Packed Accesses: [1]
- Burst Transactions: [2]
- Endianism Conversion: [3] [4]
- Hardware Synchronization: [5]
- Graphics Acceleration Support: [6]
- Posted and Nonposted Writes: [7]

SDMA Basic Programming Model

- Software-Triggered (Nonsynchronized) Transfer: [8] [9] [10] [11] [12] [13] [14] [15] [16]
- Hardware-Synchronized Transfer: [17] [18] [19] [20] [21] [22] [23] [24] [25] [26] [27] [28]
- °90 Clockwise Image Rotation: [29] [30] [31] [32] [33] [34] [35] [36] [37]

SDMA Use Cases and Tips

- SDMA Configuration to Transfer Data Between the McBSP and External DRAM: [38] [39] [40] [41] [42] [43] [44] [45] [46] [47] [48] [49] [50] [51]
- SDMA Configuration to Transfer Data Between MMC and External DRAM: [52] [53] [54] [55] [56] [57] [58] [59] [60] [61] [62] [63] [64] [65]

SDMA Registers Manual

- SDMA Register Summary: [66]

Table 9-43. DMA4_CENi

Address Offset	0x0000 0094 + (i* 0x60)	**Index**	i = 0 to 31
Physical Address	0x4805 6094 + (i* 0x60)	**Instance**	SDMA
Description	Channel Element Number		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								CHANNEL_ELMNT_NBR																									

Bits	Field Name	Description	Type	Reset
31:24	RESERVED	Reserved. Write 0s for future compatibility. Read returns 0.	RW	0x00
23:0	CHANNEL_ELMNT_NBR	Number of elements within a frame (unsigned) to transfer	RW	0x------

Table 9-44. Register Call Summary for Register DMA4_CENi

SDMA Functional Description

- Addressing Modes: [0] [1]

SDMA Basic Programming Model

- Software-Triggered (Nonsynchronized) Transfer: [2]
- Hardware-Synchronized Transfer: [3] [4] [5]
- °90 Clockwise Image Rotation: [6]

SDMA Use Cases and Tips

- SDMA Configuration to Transfer Data Between the McBSP and External DRAM: [7] [8]
- SDMA Configuration to Transfer Data Between MMC and External DRAM: [9] [10]

SDMA Registers Manual

- SDMA Register Summary: [11]

Table 9-45. DMA4_CFNi

Address Offset	0x0000 0098 + (i* 0x60)	**Index**	i = 0 to 31
Physical Address	0x4805 6098 + (i* 0x60)	**Instance**	SDMA
Description	Channel Frame Number		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																CHANNEL_FRAME_NBR															

Bits	Field Name	Description	Type	Reset
31:16	RESERVED	Reserved. Write 0s for future compatibility. Read returns 0.	RW	0x0000
15:0	CHANNEL_FRAME_NBR	Number of frames within the block to be transferred (unsigned)	RW	0x----

Table 9-46. Register Call Summary for Register DMA4_CFNi

SDMA Functional Description
- Addressing Modes: [0] [1]

SDMA Basic Programming Model
- Software-Triggered (Nonsynchronized) Transfer: [2]
- Hardware-Synchronized Transfer: [3] [4]
- °90 Clockwise Image Rotation: [5]

SDMA Use Cases and Tips
- SDMA Configuration to Transfer Data Between the McBSP and External DRAM: [6] [7]
- SDMA Configuration to Transfer Data Between MMC and External DRAM: [8] [9]

SDMA Registers Manual
- SDMA Register Summary: [10]

Table 9-47. DMA4_CSSAi

Address Offset	0x0000 009C + (i* 0x60)	**Index**	i = 0 to 31
Physical Address	0x4805 609C + (i* 0x60)	**Instance**	SDMA
Description	Channel Source Start Address		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SRC_START_ADRS																															

Bits	Field Name	Description	Type	Reset
31:0	SRC_START_ADRS	32 bits of the source start address	RW	0x--------

Table 9-48. Register Call Summary for Register DMA4_CSSAi

SDMA Functional Description
- Addressing Modes: [0]

SDMA Basic Programming Model
- Software-Triggered (Nonsynchronized) Transfer: [1]
- Hardware-Synchronized Transfer: [2] [3] [4] [5]
- °90 Clockwise Image Rotation: [6]

SDMA Use Cases and Tips
- SDMA Configuration to Transfer Data Between the McBSP and External DRAM: [7] [8]
- SDMA Configuration to Transfer Data Between MMC and External DRAM: [9] [10]

SDMA Registers Manual
- SDMA Register Summary: [11]

Table 9-49. DMA4_CDSAi

Address Offset	0x0000 00A0 + (i* 0x60)	**Index**	i = 0 to 31
Physical Address	0x4805 60A0 + (i* 0x60)	**Instance**	SDMA
Description	Channel Destination Start Address		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DST_START_ADRS																															

Bits	Field Name	Description	Type	Reset
31:0	DST_START_ADRS	32 bits of the destination start address	RW	0x--------

Table 9-50. Register Call Summary for Register DMA4_CDSAi

SDMA Functional Description
- Addressing Modes: [0]

SDMA Basic Programming Model
- Software-Triggered (Nonsynchronized) Transfer: [1]
- Hardware-Synchronized Transfer: [2] [3] [4] [5]
- °90 Clockwise Image Rotation: [6]

SDMA Use Cases and Tips
- SDMA Configuration to Transfer Data Between the McBSP and External DRAM: [7] [8]
- SDMA Configuration to Transfer Data Between MMC and External DRAM: [9] [10]

SDMA Registers Manual
- SDMA Register Summary: [11]

Table 9-51. DMA4_CSEIi

Address Offset	0x0000 00A4 + (i* 0x60)	**Index**	i = 0 to 31
Physical Address	0x4805 60A4 + (i* 0x60)	**Instance**	SDMA
Description	Channel Source Element Index (Signed)		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																CHANNEL_SRC_ELMNT_INDEX															

Bits	Field Name	Description	Type	Reset
31:16	RESERVED	Reserved. Write 0s for future compatibility. Read returns 0.	RW	0x0000
15:0	CHANNEL_SRC_ ELMNT_INDEX	Channel source element index	RW	0x----

Table 9-52. Register Call Summary for Register DMA4_CSEIi

SDMA Registers Manual
- SDMA Register Summary: [0]

Table 9-53. DMA4_CSFIi

Address Offset	0x0000 00A8 + (i* 0x60)	**Index**	i = 0 to 31
Physical Address	0x4805 60A8 + (i* 0x60)	**Instance**	SDMA
Description	Channel Source Frame Index (Signed) or 16-bit Packet size		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CH_SRC_FRM_INDEX_OR_																															

Bits	Field Name	Description	Type	Reset
31:0	CH_SRC_FRM_INDEX_OR_ 16BIT_PKT_ELNT_NBR	Channel source frame index value if source address is in double index mode. Or if fs=bs=1 and DMA_CCR[SEL_SRC_DST_SYNC]=1; the bit field [15:0] gives the number of element in packet. The field [31:16] is unused for the packet size.	RW	0x--------

Table 9-54. Register Call Summary for Register DMA4_CSFIi

SDMA Functional Description
- Packet Synchronization: [0] [1] [2] [3] [4]

SDMA Basic Programming Model
- Hardware-Synchronized Transfer: [5] [6]

SDMA Registers Manual
- SDMA Register Summary: [7]

Table 9-55. DMA4_CDEIi

Address Offset	0x0000 00AC + (i* 0x60)	**Index**	i = 0 to 31
Physical Address	0x4805 60AC + (i* 0x60)	**Instance**	SDMA
Description	Channel Destination Element Index (Signed)		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																CHANNEL_DST_ELMNT_INDEX															

Bits	Field Name	Description	Type	Reset
31:16	RESERVED	Reserved. Write 0s for future compatibility. Read returns 0.	RW	0x0000
15:0	CHANNEL_DST_ELMNT_INDEX	Channel destination element index	RW	0x----

Table 9-56. Register Call Summary for Register DMA4_CDEIi

SDMA Registers Manual
- SDMA Register Summary: [0]

Table 9-57. DMA4_CDFIi

Address Offset	0x0000 00B0 + (i* 0x60)	**Index**	i = 0 to 31
Physical Address	0x4805 60B0 + (i* 0x60)	**Instance**	SDMA
Description	Channel Destination Frame Index (Signed) or 16-bit Packet size		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CH_DST_FRM_IDX_OR_16BIT_																															

Bits	Field Name	Description	Type	Reset
31:0	CH_DST_FRM_IDX_OR_16BIT_PKT_ELNT_NBR	Channel destination frame index value if destination address is in double index mode. Or if fs=bs=1 and DMA_CCR[SEL_SRC_DST_SYNC]=0; the bit field [15:0] gives the number of element in packet. The field [31:16] is unused for the packet size..	RW	0x--------

Table 9-58. Register Call Summary for Register DMA4_CDFIi

SDMA Functional Description • Packet Synchronization: [0] [1] [2] [3]
SDMA Basic Programming Model • Hardware-Synchronized Transfer: [4] [5]
SDMA Registers Manual • SDMA Register Summary: [6]

Table 9-59. DMA4_CSACi

Address Offset	0x0000 00B4 + (i* 0x60)	**Index**	i = 0 to 31
Physical Address	0x4805 60B4 + (i* 0x60)	**Instance**	SDMA
Description	Channel Source Address Value. User has to access this register only in 32-bit access. If accessed in 8-bit or 16bit data may be corrupted.		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SRC_ELMNT_ADRS																															

Bits	Field Name	Description	Type	Reset
31:0	SRC_ELMNT_ADRS	Current source address counter value	R	0x--------

Table 9-60. Register Call Summary for Register DMA4_CSACi

SDMA Registers Manual • SDMA Register Summary: [0]

Table 9-61. DMA4_CDACi

Address Offset	0x0000 00B8 + (i* 0x60)	**Index**	i = 0 to 31
Physical Address	0x4805 60B8 + (i* 0x60)	**Instance**	SDMA
Description	Channel Destination Address Value. User has to access this register only in 32-bit access. If accessed in 8-bit or 16bit data may be corrupted.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DST_ELMNT_ADRS																															

Bits	Field Name	Description	Type	Reset
31:0	DST_ELMNT_ADRS	Current destination address counter value	RW	0x--------

Table 9-62. Register Call Summary for Register DMA4_CDACi

SDMA Functional Description • Hardware Synchronization: [0]
SDMA Basic Programming Model • Hardware-Synchronized Transfer: [1] • Synchronized Transfer Monitoring Using CDAC: [2] [3] [4] [5] [6] [7] [8]

Table 9-62. Register Call Summary for Register DMA4_CDACi (continued)

SDMA Registers Manual • SDMA Register Summary: [9]

Table 9-63. DMA4_CCENi

Address Offset	0x0000 00BC + (i* 0x60)	**Index**	i = 0 to 31
Physical Address	0x4805 60BC + (i* 0x60)	**Instance**	SDMA
Description	Channel Current Transferred Element Number in the current frame. User has to access this register only in 32-bit access. If accessed in 8-bit or 16bit data may be corrupted.		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								CURRENT_ELMNT_NBR																									

Bits	Field Name	Description	Type	Reset
31:24	RESERVED	Reserved. Write 0s for future compatibility. Read returns 0.	RW	0x00
23:0	CURRENT_ELMNT_NBR	Channel current transferred element number in the current frame	R	0x------

Table 9-64. Register Call Summary for Register DMA4_CCENi

SDMA Basic Programming Model • Synchronized Transfer Monitoring Using CDAC: [0]
SDMA Registers Manual • SDMA Register Summary: [1]

Table 9-65. DMA4_CCFNi

Address Offset	0x0000 00C0 + (i* 0x60)	**Index**	i = 0 to 31
Physical Address	0x4805 60C0 + (i* 0x60)	**Instance**	SDMA
Description	Channel Current Transferred Frame Number in the current transfer. User has to access this register only in 32-bit access. If accessed in 8-bit or 16bit data may be corrupted.		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																CURRENT_FRAME_NBR															

Bits	Field Name	Description	Type	Reset
31:16	RESERVED	Reserved. Write 0s for future compatibility. Read returns 0.	RW	0x0000
15:0	CURRENT_FRAME_NBR	Channel current transferred frame number in the current transfer	R	0x----

Table 9-66. Register Call Summary for Register DMA4_CCFNi

SDMA Basic Programming Model • Synchronized Transfer Monitoring Using CDAC: [0]
SDMA Registers Manual • SDMA Register Summary: [1]

Table 9-67. DMA4_COLORi

Address Offset	0x0000 00C4 + (i* 0x60)	**Index**	i = 0 to 31
Physical Address	0x4805 60C4 + (i* 0x60)	**Instance**	SDMA
Description	Channel DMA COLOR KEY /SOLID COLOR		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								CH_BLT_FRGRND_COLOR_																									

Bits	Field Name	Description	Type	Reset
31:24	RESERVED	Reserved. Write 0s for future compatibility. Read returns 0.	RW	0x–
23:0	CH_BLT_FRGRND_COLOR_ OR_SOLID_COLOR_PTRN	Color key or solid color pattern: The pattern is replicated according to the data type. If the data-type is 8-bit the pattern is replicated 4 times to fill the register in order to enhance processing when data is packed at the graphic module input. The same reasoning for 16-bit data-type.	RW	0x------

Table 9-68. Register Call Summary for Register DMA4_COLORi

SDMA Functional Description • Graphics Acceleration Support: [0] [1]
SDMA Basic Programming Model • Graphic Operations: [2] [3]
SDMA Registers Manual • SDMA Register Summary: [4]

Interrupt Controller

This chapter gives an overview of the interrupt controllers (INTCs) and describes in detail the MPU subsystem interrupt controller (MPU_INTC) module used in the OMAP34xx stand-alone multimedia device.

Note: This chapter gives information about all modules and features in the high-tier device. See Section 1.5, *OMAP34xx Family*, to check availability of modules and features. Ensure that interrupts of unavailable modules and features are masked in MPU/IVA subsystems.

Topic **Page**

10.1 Interrupt Controller Overview 1038
10.2 Interrupt Controller Environment 1040
10.3 MPU Subsystem INTCPS Integration 1041
10.4 Interrupt Controller Functional Description 1045
10.5 Interrupt Basic Programming Model 1049
10.6 Interrupt Controller Register Manual 1057

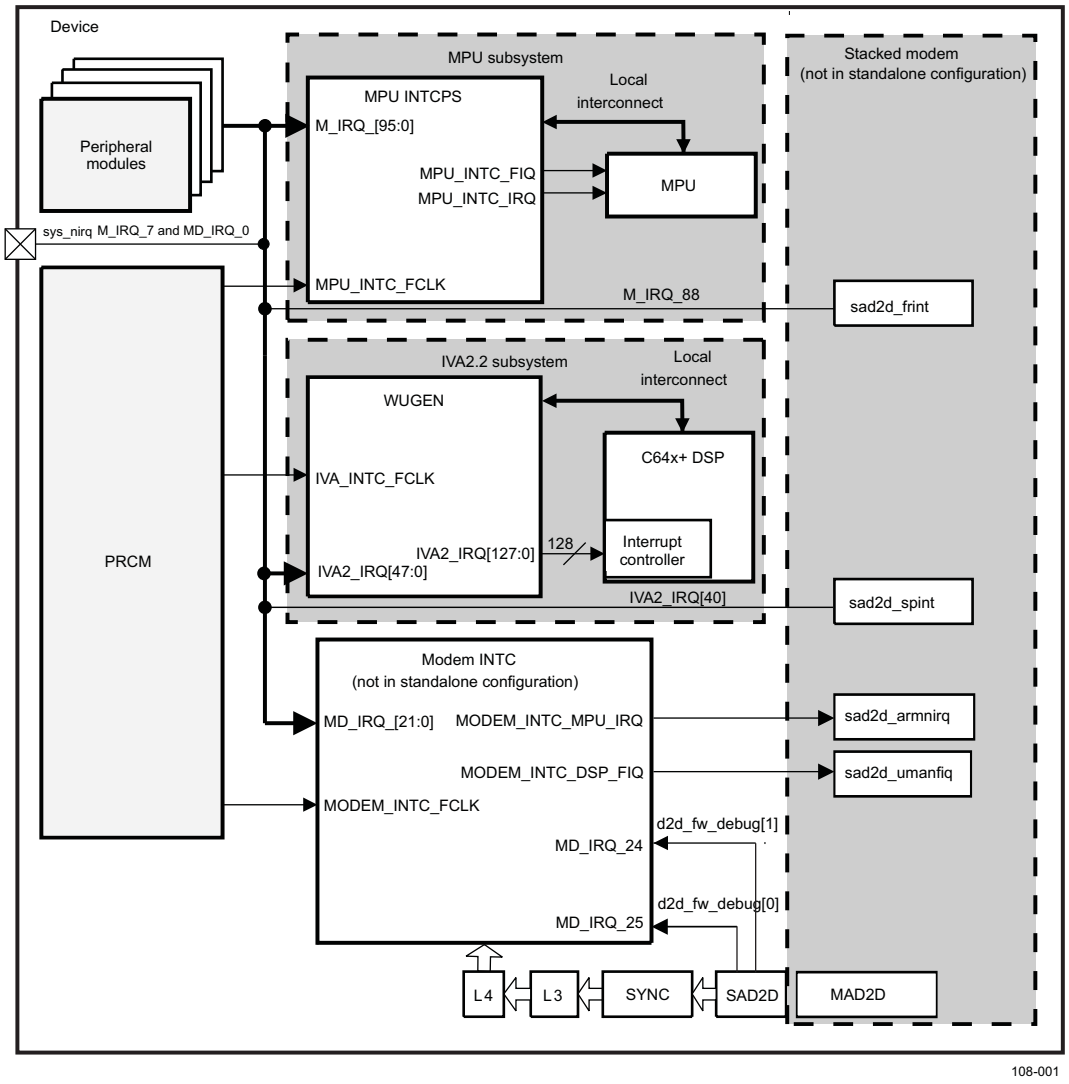
10.1 Interrupt Controller Overview

The device provides three interrupt controller (INTC) modules:

- **MPU subsystem INTC (INTCPS):** This module handles all MPU-related events, using Priority Threshold. It communicates with the public ARM Cortex-A8 processor using a private local interconnect, and runs at half the speed of the processor.
- **IVA2.2 subsystem INTC:** This module is a specific combination of WUGEN (wake-up generator) and the C64x+ DSP interrupt controller (IC). It is used in the device, but is not described in detail in this chapter. For detailed information about this INTC, see Chapter 14, *IVA2.2 Subsystem*.
- **Modem INTC:** This module is an L4-mapped INTC that allows the regrouping of all the interrupts sent to the modem subsystem in stacked mode. It is seen as a level 2 INTC by the MPU and IVA2.2 subsystems. This modem INTC is integrated in the stand-alone device, but it is used only with the stacked modem. Therefore, it is not described in detail in this TRM, which focuses on the stand-alone device.

Figure 10-1 shows the internal interrupt scheme with optional modem INTC (not used in the stand-alone configuration).

Figure 10-1. Interrupt Controllers Highlight



10.2 Interrupt Controller Environment

The INTC can handle two types of interrupts originating from an external device:

- sys_nirq interrupt inputs:

  The MPU INTC and modem INTC handle external interrupts through a dedicated sys_nirq interrupt line that connects the two INTC modules with a TWL4030 power IC, respectively. An interrupt can generate a system wake-up event.

  If the system is idle and the external interrupt is masked, the interrupt cannot wake up the system. Like other interrupt lines, the external interrupt is active at low level and is acknowledged by the software according to the common programming model.

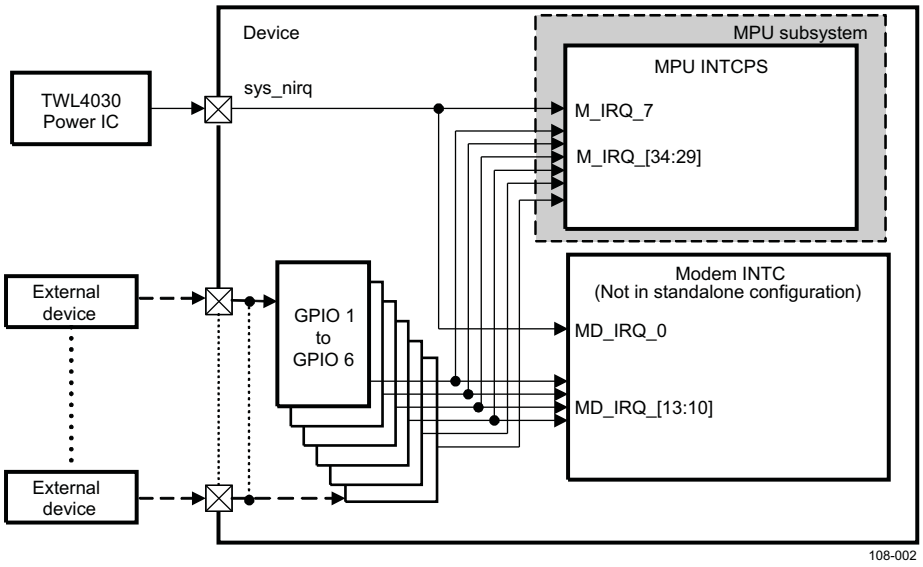
Note: If the CORE power domain is in retention (CSWR or OSWR) or off mode, the interrupt requests (internal or external), the MPU INTC, and the modem INTC (in the CORE domain) have no effect. The CORE power domain does not wake up, and the interrupt is not signaled to the MPU/modem.

- GPIO interrupt inputs:

  External devices can also use GPIO modules to generate interrupts to the MPU and the modem. There are six dedicated interrupt lines to the MPU INTC and four dedicated interrupt inputs to the modem INTC. One interrupt line is associated with each GPIO module. Each GPIO module can generate a single interrupt whenever there is at least one event in any one of the configured 32 GPIO inputs. For more information about GPIO features, see Chapter 25, *General-Purpose Interface*.

Figure 10-2 shows the relationship between the device and external interrupts.

Figure 10-2. Interrupts From External Devices



Note: The modem INTC cannot use GPIO5 and GPIO6 as interrupt sources.

The features specific to INTCPS are:

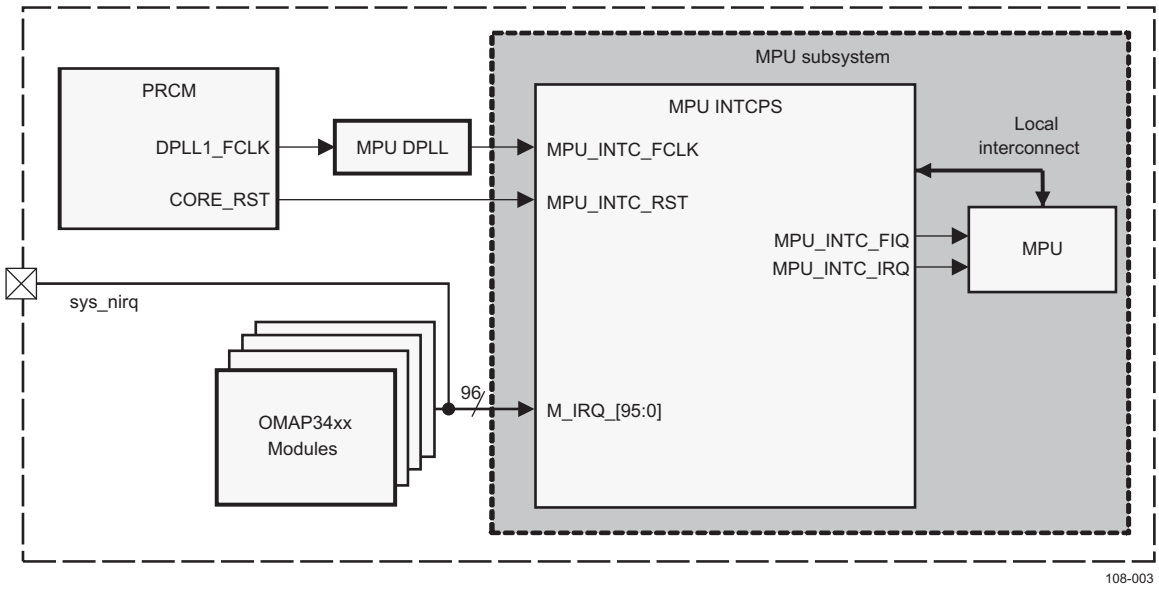
- Up to 96 level-sensitive interrupt inputs
- Individual priority (up to 64) for each interrupt input
- Interrupt lines connected to internal module interrupts
- One incoming interrupt line from an external device

10.3 MPU Subsystem INTCPS Integration

The INTCPS module is the interface between incoming interrupts and the two interrupt inputs of the MPU. It can handle up to 96 request inputs that can be configured as MPU FIQ or IRQ interrupt requests.

Figure 10-3 shows the integration of the INTCPS in the MPU subsystem.

Figure 10-3. MPU Subsystem INTCPS Integration



The MPU subsystem INTCPS is directly connected to the MPU by an MPU peripheral port. Consequently, the MPU subsystem INTCPS is accessible and visible only by the MPU.

10.3.1 Clocking, Reset, and Power Management Scheme

10.3.1.1 MPU Subsystem INTC Clocks

The MPU subsystem INTCPS runs at half the rate of the MPU functional clock (see Chapter 3, *MPU Subsystem*).

The interface clock used for register access runs at the rate of the interconnect bus clock (equal to the rate of the MPU interface clock; see Chapter 4, *Power, Reset, and Clock Management*).

The synchronizer clock allows external asynchronous interrupts to be resynchronized before they are masked.

Table 10-1 lists the MPU subsystem INTC clock rates.

Table 10-1. MPU Subsystem INTC Clock Rates

Clock	Frequency	Name	Comments
Functional	ARM_FCLK	MPU_INTC_FCLK	Source is the MPU DPLL.
Interface	ARM_FCLK	MPU_INTC_ICLK	Source is the PRCM module.
Synchronizer	MPU_INTC_FCLK	Synchronizer clock (module internal clock)	Source is the MPU_INTC_FCLK.

10.3.1.2 Hardware and Software Reset

Table 10-2 lists the MPU subsystem INTC resets.

Table 10-2. Hardware and Software Reset

Type	Name	Source	Activation	Domain
Hardware	CORE_RST	PRCM	Active low	CORE
Software	SOFTRESET	MPU_INTC.INTCPS_SYSCONFIG[1] SOFTRESET bit	Active at 1	MPU INTC internal

10.3.1.3 Power Management

The MPU subsystem INTC belongs to the CORE power domain. As part of CORE power domain, it is sensitive to a CORE_RST issued by the PRCM. For more information about the CORE power domain implementation and CORE_RST signal, see Chapter 4, *Power, Reset, and Clock Management*.

The MPU INTC clocks come from the MPU DPLL. For more information about these clocks control, see Chapter 3, *MPU Subsystem*.

10.3.2 Interrupt Request Lines

Table 10-3 lists the incoming and outgoing interrupt lines of the INTCPS.

Table 10-3. Interrupt Lines Incoming and Outgoing

Type	Number	Name	Mapping	Comments
Interrupt request inputs	Up to 96	M_IRQ_[95:0]	See Table 10-4	Inputs to INTCPS module, source from various modules.
Interrupt request outputs	2	MPU_INTC_FIQ	MPU_INTC_FIQ	Outgoing to MPU Fast Interrupt
		MPU_INTC_IRQ	MPU_INTC_IRQ	Outgoing to MPU Normal Interrupt

Note: Interrupt request signals are active at low level.

CAUTION

A single interrupt source can be physically mapped to multiple INTCs (MPU subsystem, IVA2.2 subsystem, and modem). With multiple-mapped interrupts, it is strongly recommended each interrupt source be unmasked in only one INTC at a time.

This section gives information about all modules and features in the high-tier device. See Section 1.5, *OMAP34xx Family*, to check availability of modules and features. Ensure that interrupts of unavailable modules and features are masked in MPU subsystem.

Table 10-4 lists interrupt mappings to the MPU subsystem.

Table 10-4. Interrupt Mapping to the MPU Subsystem(1)

IRQ	Source	Description
M_IRQ_0	EMUINT	MPU emulation(2)
M_IRQ_1	COMMTX	MPU emulation(2)
M_IRQ_2	COMMRX	MPU emulation(2)
M_IRQ_3	BENCH	MPU emulation(2)
M_IRQ_4	MCBSP2_ST_IRQ	Sidetone MCBSP2 overflow
M_IRQ_5	MCBSP3_ST_IRQ	Sidetone MCBSP3 overflow
M_IRQ_6	Reserved	Reserved

(1) All the IRQ signals are active at low level.
(2) These interrupts are internally generated within the MPU subsystem.

Table 10-4. Interrupt Mapping to the MPU Subsystem (continued)

IRQ	Source	Description
M_IRQ_7	sys_nirq	External source (active low)
M_IRQ_8	Reserved	Reserved
M_IRQ_9	SMX_DBG_IRQ	L3 interconnect error for debug
M_IRQ_10	SMX_APP_IRQ	L3 interconnect error for application
M_IRQ_11	PRCM_MPU_IRQ	PRCM module IRQ
M_IRQ_12	SDMA_IRQ_0	System DMA request 0 (3)
M_IRQ_13	SDMA_IRQ_1	System DMA request 1 (3)
M_IRQ_14	SDMA_IRQ_2	System DMA request 2
M_IRQ_15	SDMA_IRQ_3	System DMA request 3
M_IRQ_16	MCBSP1_IRQ	McBSP module 1 IRQ (3)
M_IRQ_17	MCBSP2_IRQ	McBSP module 2 IRQ (3)
M_IRQ_18	This information is not available in public domain	
M_IRQ_19	This information is not available in public domain	
M_IRQ_20	GPMC_IRQ	General-purpose memory controller module
M_IRQ_21	SGX_IRQ	2D/3D graphics module
M_IRQ_22	MCBSP3_IRQ	McBSP module 3 (3)
M_IRQ_23	MCBSP4_IRQ	McBSP module 4 (3)
M_IRQ_24	CAM_IRQ0	Camera interface request 0
M_IRQ_25	DSS_IRQ	Display subsystem module (3)
M_IRQ_26	MAIL_U0_MPU_IRQ	Mailbox user 0 request
M_IRQ_27	MCBSP5_IRQ	McBSP module 5 (3)
M_IRQ_28	IVA2_MMU_IRQ	IVA2 MMU
M_IRQ_29	GPIO1_MPU_IRQ	GPIO module 1 (3)(4)
M_IRQ_30	GPIO2_MPU_IRQ	GPIO module 2 (3)(4)
M_IRQ_31	GPIO3_MPU_IRQ	GPIO module 3 (3)(4)
M_IRQ_32	GPIO4_MPU_IRQ	GPIO module 4 (3)(4)
M_IRQ_33	GPIO5_MPU_IRQ	GPIO module 5 (3)
M_IRQ_34	GPIO6_MPU_IRQ	GPIO module 6 (3)
M_IRQ_35	Reserved	Reserved
M_IRQ_36	WDT3_IRQ	Watchdog timer module 3 overflow
M_IRQ_37	GPT1_IRQ	General-purpose timer module 1
M_IRQ_38	GPT2_IRQ	General-purpose timer module 2
M_IRQ_39	GPT3_IRQ	General-purpose timer module 3
M_IRQ_40	GPT4_IRQ	General-purpose timer module 4
M_IRQ_41	GPT5_IRQ	General-purpose timer module 5 (3)
M_IRQ_42	GPT6_IRQ	General-purpose timer module 6 (3)
M_IRQ_43	GPT7_IRQ	General-purpose timer module 7 (3)
M_IRQ_44	GPT8_IRQ	General-purpose timer module 8 (3)
M_IRQ_45	GPT9_IRQ	General-purpose timer module 9
M_IRQ_46	GPT10_IRQ	General-purpose timer module 10
M_IRQ_47	GPT11_IRQ	General-purpose timer module 11
M_IRQ_48	SPI4_IRQ	McSPI module 4
M_IRQ_49	Reserved	Reserved
M_IRQ_50	Reserved	Reserved
M_IRQ_51	Reserved	Reserved

(3) Shared with the IVA2.2 interrupt controller
(4) Shared with the modem interrupt controller

Table 10-4. Interrupt Mapping to the MPU Subsystem (continued)

IRQ	Source	Description
M_IRQ_52	Reserved	Reserved
M_IRQ_53	Reserved	Reserved[3]
M_IRQ_54	MCBSP4_IRQ_TX	McBSP module 4 transmit[3]
M_IRQ_55	MCBSP4_IRQ_RX	McBSP module 4 receive[3]
M_IRQ_56	I2C1_IRQ	I^2C module 1
M_IRQ_57	I2C2_IRQ	I^2C module 2
M_IRQ_58	HDQ_IRQ	HDQ/1-Wire®
M_IRQ_59	McBSP1_IRQ_TX	McBSP module 1 transmit[3]
M_IRQ_60	McBSP1_IRQ_RX	McBSP module 1 receive[3]
M_IRQ_61	I2C3_IRQ	I^2C module 3
M_IRQ_62	McBSP2_IRQ_TX	McBSP module 2 transmit[3]
M_IRQ_63	McBSP2_IRQ_RX	McBSP module 2 receive[3]
M_IRQ_64	Reserved	Reserved
M_IRQ_65	SPI1_IRQ	McSPI module 1
M_IRQ_66	SPI2_IRQ	McSPI module 2
M_IRQ_67	Reserved	Reserved
M_IRQ_68	Reserved	Reserved
M_IRQ_69	Reserved	Reserved
M_IRQ_70	Reserved	Reserved
M_IRQ_71	Reserved	Reserved
M_IRQ_72	UART1_IRQ	UART module 1
M_IRQ_73	UART2_IRQ	UART module 2
M_IRQ_74	UART3_IRQ	UART module 3 (also infrared)[3]
M_IRQ_75	PBIAS_IRQ	Merged interrupt for PBIASlite1 and 2
M_IRQ_76	OHCI_IRQ	OHCI controller HSUSB MP Host Interrupt
M_IRQ_77	EHCI_IRQ	EHCI controller HSUSB MP Host Interrupt
M_IRQ_78	TLL_IRQ	HSUSB MP TLL Interrupt
M_IRQ_79	Reserved	Reserved
M_IRQ_80	Reserved	Reserved
M_IRQ_81	MCBSP5_IRQ_TX	McBSP module 5 transmit[3]
M_IRQ_82	MCBSP5_IRQ_RX	McBSP module 5 receive[3]
M_IRQ_83	MMC1_IRQ	MMC/SD module 1
M_IRQ_84	Reserved	Reserved
M_IRQ_85	Reserved	Reserved
M_IRQ_86	MMC2_IRQ	MMC/SD module 2
M_IRQ_87	MPU_ICR_IRQ	MPU ICR
M_IRQ_88	D2DFRINT	From 3G coprocessor hardware when used in stacked modem configuration
M_IRQ_89	MCBSP3_IRQ_TX	McBSP module 3 transmit[3]
M_IRQ_90	MCBSP3_IRQ_RX	McBSP module 3 receive[3]
M_IRQ_91	SPI3_IRQ	McSPI module 3
M_IRQ_92	HSUSB_MC_NINT	High-Speed USB OTG controller
M_IRQ_93	HSUSB_DMA_NINT	High-Speed USB OTG DMA controller
M_IRQ_94	MMC3_IRQ	MMC/SD module 3
M_IRQ_95	Reserved	Reserved

10.4 Interrupt Controller Functional Description

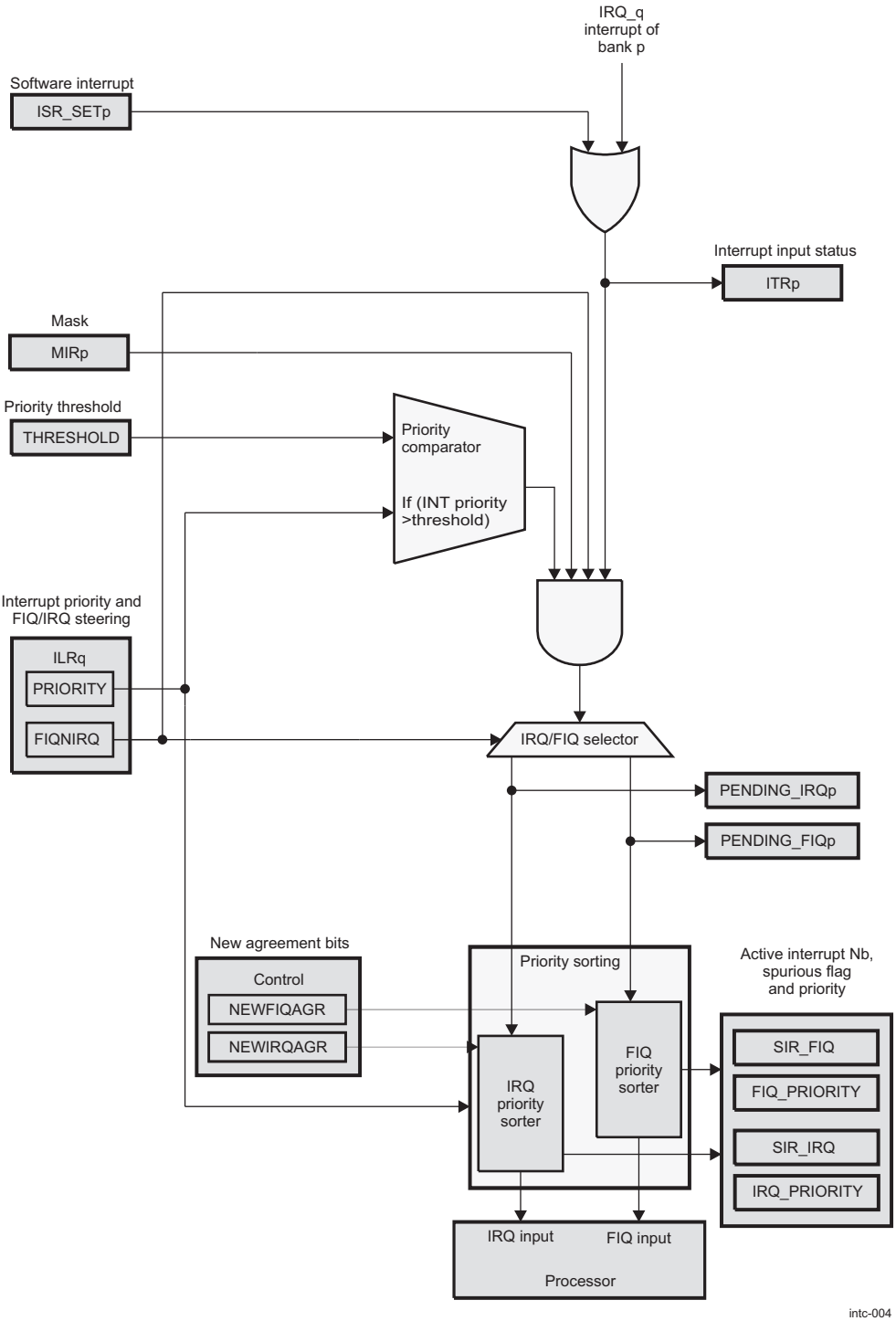
The main features of the INTCPS are:

- Individual priority (up to 64 levels) for each interrupt input
- Ability for each interrupt to be steered to either FIQ or IRQ
- Independent priority sorting for FIQ and IRQ; FIQ sorting is processed concurrently with IRQ sorting.
- Priority masking: Interrupts can be masked based on the priority threshold register.
- Atomic bit set and clear capability for interrupt mask and software interrupt registers
- Power-management and wake-up support
- Auto-idle power-saving support

The INTCPS processes incoming interrupts by masking and priority sorting, then it generates the interrupt requests to the MPU.

Figure 10-4 shows the top-level view of the interrupt processing.

Figure 10-4. Top-Level Block Diagram



10.4.1 Interrupt Processing

10.4.1.1 Input Selection

The INTCPS supports only level-sensitive incoming interrupt detection. A peripheral asserting an interrupt maintains it until software has handled the interrupt and instructed the peripheral to deassert the interrupt.

A software interrupt is generated if the corresponding bit in the MPU_INTC.INTCPS_ISR_SETn register is set (register bank number: n = [0,2] for the MPU subsystem INTCPS, 96 incoming interrupt lines are supported). The software interrupt clears when the corresponding bit in the MPU_INTC.INTCPS_ISR_CLEARn register is written. Typical use of this feature is software debugging.

10.4.1.2 Masking

10.4.1.2.1 Individual Masking

Detection of interrupts on each incoming interrupt line can be enabled or disabled independently by the MPU_INTC.INTCPS_MIRn interrupt mask register. In response to an unmasked incoming interrupt, the INTCPS can generate one of two types of interrupt requests to the processor:

- IRQ: low-priority interrupt request
- FIQ: fast interrupt request

The type of interrupt request is determined by the MPU_INTC.INTCPS_ILRm[0] FIQNIRQ bit (m= [0,95]).

The current incoming interrupt status before masking is readable from the MPU_INTC.INTCPS_ITRn register. After masking and IRQ/FIQ selection, and before priority sorting is done, the interrupt status is readable from the MPU_INTC.INTCPS_PENDING_IRQn and MPU_INTC.INTCPS_PENDING_FIQn registers.

10.4.1.2.2 Priority Masking

To enable faster processing of high-priority interrupts, a programmable priority masking threshold is provided (the MPU_INTC.INTCPS_THRESHOLD[7:0] PRIORITYTHRESHOLD field). This priority threshold allows preemption by higher priority interrupts; all interrupts of lower or equal priority than the threshold are masked. However, priority 0 can never be masked by this threshold; a priority threshold of 0 is treated the same way as priority 1.

PRIORITY and PRIORITYTHRESHOLD fields values can be set between 0x0 and 0x3F; 0x0 is the highest priority and 0x3F is the lowest priority.

When priority masking is not necessary, a priority threshold value of 0xFF disables the priority threshold mechanism. This value is also the reset default for backward compatibility with previous versions of the INTCPS.

10.4.1.3 Priority Sorting

A priority level (0 being the highest) is assigned to each incoming interrupt line. Both the priority level and the interrupt request type are configured by the MPU_INTC.INTCPS_ILRm register. If more than one incoming interrupt with the same priority level and interrupt request type occur simultaneously, the highest-numbered interrupt is serviced first.

When one or more unmasked incoming interrupts are detected, the INTCPS separates between IRQ and FIQ using the corresponding MPU_INTC.INTCPS_ILRm[0] FIQNIRQ bit. The result is placed in INTCPS_PENDING_IRQn or INTCPS_PENDING_FIQn

If no other interrupts are currently being processed, INTCPS asserts IRQ/FIQ and starts the priority computation. Priority sorting for IRQ and FIQ can execute in parallel.

Each IRQ/FIQ priority sorter determines the highest priority interrupt number. Each priority number is placed in the corresponding MPU_INTC.INTCPS_SIR_IRQ[6:0] ACTIVEIRQ field or MPU_INTC.INTCPS_SIR_FIQ[6:0] ACTIVEFIQ field. The value is preserved until the corresponding MPU_INTC.INTCPS_CONTROL NEWIRQAGR or NEWFIQAGR bit is set.

Once the interrupting peripheral device has been serviced and the incoming interrupt deasserted, the user must write to the appropriate NEWIRQAGR or NEWFIQAGR bit to indicate to the INTCPS the interrupt has been handled. If there are any pending unmasked incoming interrupts for this interrupt request type, the INTCPS restarts the appropriate priority sorter; otherwise, the IRQ or FIQ interrupt line is deasserted.

10.4.2 Register Protection

If the MPU_INTC.INTCPS_PROTECTION[0] PROTECTION bit is set, access to the INTCPS registers is restricted to the supervisor mode. Access to the MPU_INTC.INTCPS_PROTECTION register is always restricted to privileged mode.

10.4.3 Module Power Saving

The INTCPS provides an auto-idle function in its three clock domains:

- Interface clock
- Functional clock
- Synchronizer clock

The interface clock auto-idle power-saving mode is enabled if the MPU_INTC.INTCPS_SYSCONFIG[0] AUTOIDLE bit is set to 1. When this mode is enabled and there is no activity on the bus interface, the interface clock is disabled internally to the module, thus reducing power consumption. When there is new activity on the bus interface, the interface clock restarts without any latency penalty. After reset, this mode is disabled, by default.

The functional clock auto-idle power-saving mode is enabled if the MPU_INTC.INTCPS_IDLE[0] FUNCIDLE bit is set to 0. When this mode is enabled and there is no active interrupt (IRQ or FIQ interrupt being processed or generated) or no pending incoming interrupt, the functional clock is disabled internally to the module, thus reducing power consumption. When a new unmasked incoming interrupt is detected, the functional clock restarts and the INTCPS processes the interrupt. If this mode is disabled, the interrupt latency is reduced by one cycle. After reset, this mode is enabled, by default.

The synchronizer clock allows external asynchronous interrupts to be resynchronized before they are masked. The synchronizer input clock has an auto-idle power-saving mode enabled if the MPU_INTC.INTCPS_IDLE[1] TURBO bit is set to 1. If the auto-idle mode is enabled, the standby power is reduced, but the IRQ or FIQ interrupt latency increases from four to six functional clock cycles. This feature can be enabled dynamically according to the requirements of the device. After reset, this mode is disabled, by default.

To reduce power consumption of the modem INTC, which is not use in the stand-alone device, modem INTC clocks must be configured in auto-idle mode. Therefore, both the MPU_INTC.INTCPS_SYSCONFIG[0] AUTOIDLE and MPU_INTC.INTCPS_IDLE[1] TURBO bits must be set to 1 during initialization.

10.4.4 Interrupt Latency

The IRQ/FIQ interrupt generation takes four INTCPS functional clock cycles (plus or minus one cycle) if the MPU_INTC.INTCPS_IDLE[1] TURBO bit is set to 0. If the TURBO bit is set to 1, the interrupt generation takes six cycles, but power consumption is reduced while waiting for an interrupt.

These latencies can be reduced by one cycle by disabling functional clock auto-idle (MPU_INTC.INTCPS_IDLE[0] FUNCIDLE bit set to 1), but power consumption is increased, so the benefit is minimal. For information about power saving, see Section 10.4.3.

To minimize interrupt latency when an unmasked interrupt occurs, the IRQ or FIQ interrupt is generated before priority sorting completion. The priority sorting takes 10 functional clock cycles, which is less than the minimum number of cycles required for the MPU to switch to the interrupt context after reception of the IRQ or FIQ event.

Any read of the MPU_INTC.INTCPS_SIR_IRQ or MPU_INTC.INTCPS_SIR_FIQ register during the priority sorting process stalls until priority sorting is complete and the relevant register is updated. However, the delay between the interrupt request being generated and the interrupt service routine being executed is such that priority sorting always completes before the MPU_INTC.INTCPS_SIR_IRQ or MPU_INTC.INTCPS_SIR_FIQ register is read.

10.5 Interrupt Basic Programming Model

10.5.1 Initialization Sequence

1. Program the MPU_INTC.INTCPS_SYSCONFIG register: If necessary, enable the interface clock autogating by setting the AUTOIDLE bit.
2. Program the MPU_INTC.INTCPS_IDLE register: If necessary, disable functional clock autogating or enable synchronizer autogating by setting the FUNCIDLE bit or TURBO bit accordingly.
3. Program the MPU_INTC.INTCPS_ILRm register for each interrupt line: Assign a priority level and set the FIQNFIQ bit for an FIQ interrupt (by default, interrupts are mapped to IRQ and priority is 0x0 [highest]).
4. Program the MPU_INTC.INTCPS_MIRn register: Enable interrupts (by default, all interrupt lines are masked).

Note: To program the MPU_INTC.INTCPS_MIRn register, the MPU_INTC.INTCPS_MIR_SETn and MPU_INTC.INTCPS_MIR_CLEARn registers are provided to facilitate the masking, even if it is possible for backward-compatibility to write directly to the MPU_INTC.INTCPS_MIRn register.

10.5.2 MPU INTC Processing Sequence

After the MPU_INTC.INTCPS_MIRn and MPU_INTC.INTCPS_ILRm registers are configured to enable and assign priorities to incoming interrupts, the interrupt is processed as explained in the following subsections.

IRQ and FIQ processing sequences are quite similar, the differences for the FIQ sequence are shown after a '/' character in **bold** characters in the text or the code below.

1. One or more unmasked incoming interrupts (M_IRQ_n signals) are received and IRQ or FIQ outputs (MPU_INTC_IRQ/**FIQ**) are not currently asserted.
2. If the MPU_INTC.INTCPS_ILRm[0] FIQNIRQ bit is set to 0, the MPU_INTC_IRQ output signal is generated. If the FIQNIRQ bit is set to 1, the MPU_INTC_FIQ output signal is generated.
3. The INTC performs the priority sorting and updates the MPU_INTC.INTCPS_SIR_IRQ[6:0] ACTIVEIRQ /**MPU_INTC.INTCPS_SIR_FIQ[6:0] ACTIVEFIQ** field with the current interrupt number.
4. During priority sorting, if the IRQ/**FIQ** is enabled at the host processor side, the host processor automatically saves the current context and executes the ISR as follows:

Note: The ARM host processor automatically performs the following actions in pseudo code.

```
LR = PC + 4                     /* return link                      */
SPSR = CPSR                     /* Save CPSR before execution       */
CPSR[5] = 0                     /* Execute in ARM state             */
CPSR[7] = 1                     /* Disable IRQ                      */
CPSR[8] = 1                     /* Disable Imprecise Data Aborts    */
CPSR[9] = CP15_reg1_EEbit       /* Endianness on exception entry    */
if interrupt == IRQ then
    CPSR[4:0] = 0b10010         /* Enter IRQ mode                   */
    if high vectors configured then
         PC = 0xFFFF0018
    else
         PC = 0x00000018              /* execute interrupt vector     */
else if interrupt == FIQ then
```

```
    CPSR[4:0] = 0b10001             /* Enter FIQ mode               */
    CPSR[6] = 1                     /* Disable FIQ                  */
    if high vectors configured then
        PC = 0xFFFF001C
    else
        PC = 0x0000001C             /* execute interrupt vector      */
endif
```

5. The ISR saves the remaining context, identifies the interrupt source by reading the ACTIVEIRQ/**ACTIVEFIQ** field, and jumps to the relevant subroutine handler as follows:

> **CAUTION**
>
> The code in steps 5 and 7 is an assembly code compatible with ARM architecture V6 and V7. This code is developed for the Texas Instruments Code Composer Studio tool set. It is a draft version, only tested on an emulated environment.

```
; INTCPS_SIR_IRQ/INTCPS_SIR_FIQ register address
INTCPS_SIR_IRQ_ADDR/INTCPS_SIR_FIQ_ADDR .word 0x48200040/0x48200044

; ACTIVEIRQ bit field mask to get only the bit field
ACTIVEIRQ_MASK .equ 0x7F
_IRQ_ISR/_FIQ_ISR:

       ; Save the critical context
       STMFD SP!, {R0-R12, LR}            ; Save working registers and the Link register

       MRS R11, SPSR                      ; Save the SPSR into R11

       ; Get the number of the highest priority active IRQ/FIQ
       LDR R10, INTCPS_SIR_IRQ_ADDR/INTCPS_SIR_FIQ_ADDR
       LDR R10, [R10]                     ; Get the INTCPS_SIR_IRQ/INTCPS_SIR_FIQ register
       AND R10, R10, #ACTIVEIRQ_MASK      ; Apply the mask to get the active IRQ number


      ; Jump to relevant subroutine handler
      LDR PC, [PC, R10, lsl #2]           ; PC base address points this instruction + 8

      NOP                                 ; To index the table by the PC

      ; Table of handler start addresses
      .word IRQ0handler ;For IRQ0 of BANK0
      .word IRQ1handler
      .word IRQ2handler
```

6. The subroutine handler executes code specific to the peripheral generating the interrupt by handling the event and deasserting the interrupt condition at the peripheral side.

```
; IRQ0 subroutine
IRQ0handler:

    ; Save working registers
    STMFD SP!, {R0-R1}

    ; Now read-modify-write the peripheral module status register
    ; to de-assert the M_IRQ_0 interrupt signal

    ; De-Assert the peripheral interrupt

    MOV R0, #0x7                    ; Mask for 3 flags
    LDR R1, MODULE0_STATUS_REG_ADDR ; Get the address of the module Status Register
    STR R0, [R1]                    ; Clear the 3 flags

    ; Restore working registers    LDMFD SP!, {R0-R1}
    ; Jump to the end part of the ISR
```

```
    B IRQ_ISR_end/FIQ_ISR_end
```

7. After the return of the subroutine, the ISR sets the NEWIRQAGR/**NEWFIQAGR** bit to enable the processing of subsequent pending IRQs/**FIQs** and to restore ARM context in the following code. Because the writes are posted on an Interconnect bus, to be sure that the preceding writes are done before enabling IRQs/**FIQs**, a Data Synchronization Barrier is used. This operation ensure that the IRQ/**FIQ** line is de-asserted before IRQ/**FIQ** enabling. After that, the INTC processes any other pending interrupts or deasserts the MPU_INTC_IRQ/**MPU_INTC_FIQ** signal if there is no interrupt.

```
; INTCPS_CONTROL register address
INTCPS_CONTROL_ADDR .word 0x48200048;

NEWIRQAGR/NEWFIQAGR bit mask to set only the NEWIRQAGR/NEWFIQAGR bit
NEWIRQAGR_MASK/NEWFIQAGR_MASK .equ 0x01/0x02

IRQ_ISR_end/FIQ_ISR_end:

   ; Allow new IRQs/FIQs at INTC side
   ; The INTCPS_CONTROL register is a write only register so no need to write back others bits


   MOV R0, #NEWIRQAGR_MASK/NEWFIQAGR_MASK   ; Get the NEWIRQAGR/NEWFIQAGR bit position
   LDR R1, INTCPS_CONTROL_ADDR
   STR R0, [R1]                                          ; Write the NEWIRQAGR/NEWFIQAGR
bit to allow new IRQs/FIQs

   ; Data Synchronization Barrier
   MOV R0, #0
   MCR P15, #0, R0, C7, C10, #4

   ; restore critical context
   MSR SPSR, R11                                         ; Restore the SPSR from R11
   LDMFD SP!, {R0-R12, LR}                               ; Restore working registers and
Link register

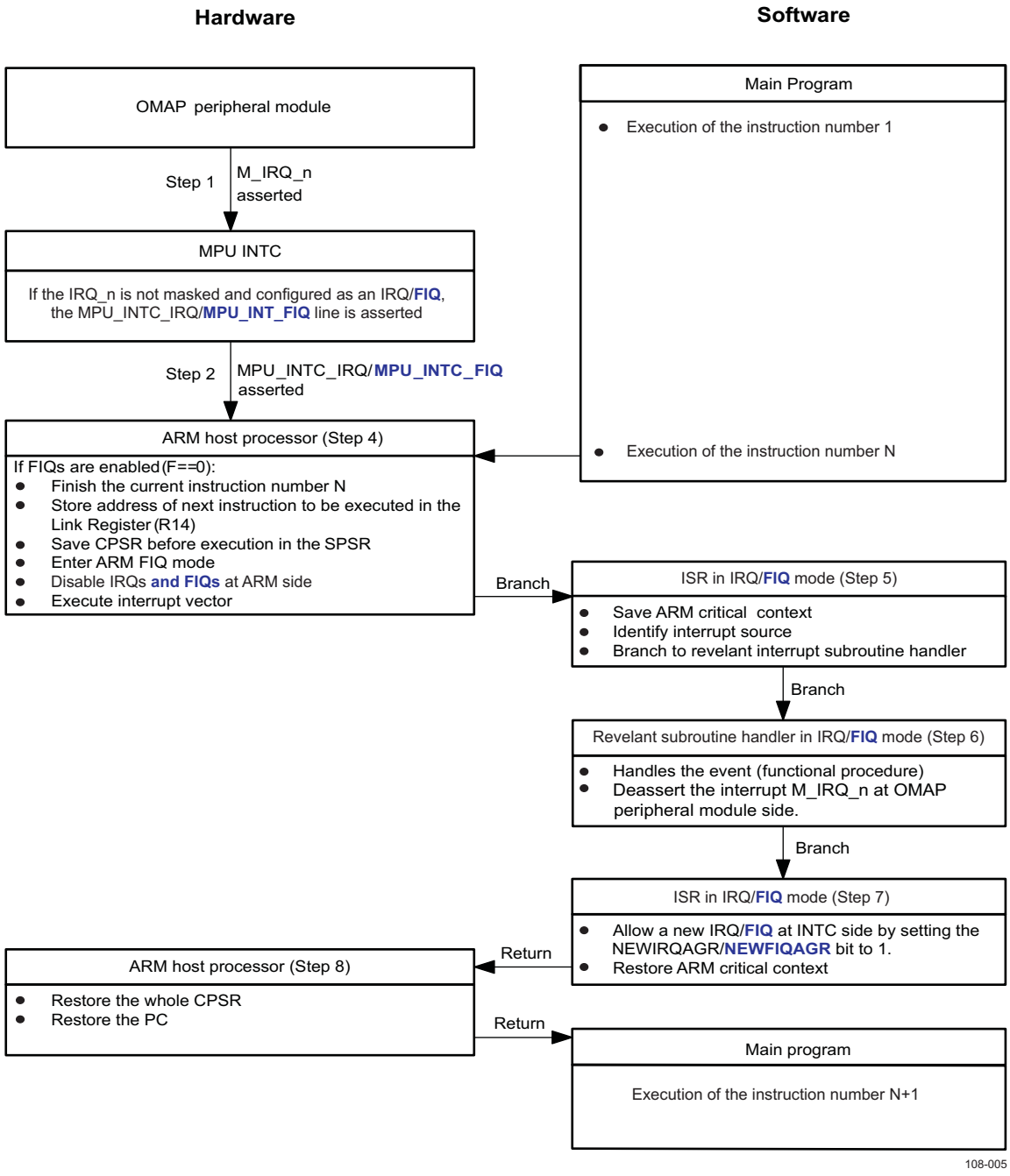
   ; Return after handling the interrupt
   SUBS PC, LR, #4
```

8. After the ISR return, the ARM automatically restores its context as follows:

```
CPSR = SPSR
PC = LR
```

Figure 10-5 shows the IRQ/**FIQ** processing sequence from the originating device peripheral module to the main program interruption.

Figure 10-5. IRQ/FIQ Processing Sequence



Note: The differences between the IRQ and the FIQ sequence are highlighted in blue and bold characters.

The priority sorting mechanism is frozen during an interrupt processing sequence. If an interrupt condition occurs during this time, the interrupt is not lost. It is sorted when the NEWIRQAGR/**NEWFIQAGR** bit is set (priority sorting is reactivated).

10.5.3 MPU INTC Preemptive Processing Sequence

Preemptive interrupts, also called nested interrupts, can reduce the latencies for higher priority interrupts. A preemptive ISR can be suspended by a higher priority interrupt. Thus, the higher priority interrupt can be served immediately.

Nested interrupts must be used carefully to avoid using corrupted data. Programmers must save corruptible registers and enable IRQ or FIQ at ARM side.

IRQ and FIQ processing sequences are quite similar, the differences for the FIQ sequence are shown after a '/' character in **bold** characters in the text or the code below.

To enable IRQ/**FIQ** preemption by higher priority IRQs/**FIQs**, programers can follow this procedure to write the ISR:

At the beginning of an IRQ/**FIQ** ISR:

1. Save the ARM critical context registers.
2. Save the MPU_INTC.INTCPS_THRESHOLD PRIORITYTHRESHOLD field before modifying it.
3. Read the active interrupt priority in the MPU_INTC.INTCPS_IRQ_PRIORITY IRQPRIORITY / **MPU_INTC.INTCPS_FIQ_PRIORITY FIQPRIORITY** field and write it to the PRIORITYTHRESHOLD(1) field.
4. Read the active interrupt number in the MPU_INTC.INTCPS_SIR_IRQ[6:0] ACTIVEIRQ / **MPU_INTC.INTCPS_SIR_FIQ[6:0] ACTIVEFIQ** field to identify the interrupt source.
5. Write 1 to the appropriate MPU_INTC.INTCPS_CONTROL NEWIRQAGR **and**(2) **NEWFIQAGR** bit while an interrupt is still processing to allow only higher priority interrupts to preempt.
6. Because the writes are posted on an Interconnect bus, to be sure that the preceding writes are done before enabling IRQs/**FIQs**, a Data Synchronization Barrier is used. This operation ensure that the IRQ line is de-asserted before IRQ/**FIQ** enabling.
7. Enable IRQ/**FIQ** at ARM side.
8. Jump to the relevant subroutine handler.

The sample code below shows the previous steps:

CAUTION

The code below is an assembly code compatible with ARM architecture V6 and V7. This code is developed for the Texas Instruments Code Composer Studio tool set. It is a draft version, only tested on an emulated environment.

```
;  bit field mask to get only the bit field
ACTIVEPRIO_MASK .equ 0x3F

_IRQ_ISR:

      ; Step 1 : Save the critical context
      STMFD SP!, {R0-R12, LR}                  ; Save working registers
      MRS R11, SPSR                            ; Save the SPSR into R11

      ; Step 2 : Save the INTCPS_THRESHOLD register into R12
      LDR R0, INTCPS_THRESHOLD_ADDR
      LDR R12, [R0]
```

(1) The priority-threshold mechanism is enabled automatically when writing a priority in the range of 0x00 to 0x3F while reading it from the IRQPRIORITY and FIQPRIORITY fields. Writing a value of 0xFF (reset default) disables the priority-threshold mechanism. Values between 0x3F and 0xFF must not be used.
When the hardware-priority threshold is in use, the priorities of interrupts selected as FIQ or IRQ become linked; otherwise, they are independent. When they are linked, all FIQ priorities must be set higher than all IRQ priorities to maintain the relative priority of FIQ over IRQ.

(2) When handling FIQs using the priority-threshold mechanism, both NEWFIQAGR and NEWIRQAGR bits must be written at the same time to ensure that the new priority threshold is applied while an IRQ sort is in progress. This IRQ will not have been seen by the ARM, as it will have been masked on entry to the FIQ ISR. However, the source of the IRQ remains active and it is finally processed when the priority threshold falls to a priority sufficiently low to allow it to be processed. The precaution of writing to New FIQ Agreement is not required during an IRQ ISR, as FIQ sorting is not affected (providing all FIQ priorities are higher than all IRQ priorities).

```
      ; Step 3 : Get the priority of the highest priority active IRQ
      LDR R1, INTCPS_IRQ_PRIORITY_ADDR/INTCPS_FIQ_PRIORITY_ADDR
      LDR R1, [R1]                              ; Get the
INTCPS_IRQ_PRIORITY/INTCPS_FIQ_PRIORITY register
      AND R1, R1, #ACTIVEPRIO_MASK              ; Apply the mask to get the priority of the IRQ

      STR R1, [R0]                              ; Write it to the INTCPS_THRESHOLD register

      ; Step 4 : Get the number of the highest priority active IRQ
      LDR R10, INTCPS_SIR_IRQ_ADDR/INTCPS_SIR_FIQ_ADDR
      LDR R10, [R10]                            ; Get the INTCPS_SIR_IRQ/INTCPS_SIR_FIQ register

      AND R10, R10, #ACTIVEIRQ_MASK             ; Apply the mask to get the active IRQ number

      ; Step 5 : Allow new IRQs and FIQs at INTC side
      MOV R0, #0x1/0x3         ; Get the NEWIRQAGR and NEWFIQAGR bit position
      LDR R1, INTCPS_CONTROL_ADDR
      STR R0, [R1]                              ; Write the NEWIRQAGR and NEWFIQAGR bit

      ; Step 6 : Data Synchronization Barrier
      MOV R0, #0    MCR P15, #0, R0, C7, C10, #4

      ; Step 7 : Read-modify-write the CPSR to enable IRQs/FIQs at ARM side
      MRS R0, CPSR                              ; Read the status register
      BIC R0, R0, #0x80/0x40   ; Clear the I/F bit
      MSR CPSR, R0                              ; Write it back to enable IRQs

      ; Step 8 : Jump to relevant subroutine handler
      LDR PC, [PC, R10, lsl #2]                 ; PC base address points this instruction + 8
      NOP                                       ; To index the table by the PC

      ; Table of handler start addresses
      .word IRQ0handler ;IRQ0 BANK0
      .word IRQ1handler
      .word IRQ2handler
```

After the return of the relevant IRQ/**FIQ** subroutine handle :

1. Disable IRQs/**FIQs** at ARM side.
2. Restore the MPU_INTC.INTCPS_THRESHOLD PRIORITYTHRESHOLD field.
3. Restore the ARM critical context registers.

The sample code below shows the three previous steps:

CAUTION

The code below is an assembly code compatible with ARM architecture V6 and V7. This code is developed for the Texas Instruments Code Composer Studio tool set. It is a draft version, only tested on an emulated environment.

```
IRQ_ISR_end:

     ; Step 1 : Read-modify-write the CPSR to disable IRQs/FIQs at ARM side
     MRS R0, CPSR                                    ; Read the CPSR
     ORR R0, R0, #0x80/0x40             ; Set the I/F bit
     MSR CPSR, R0                                    ; Write it back to disable IRQs

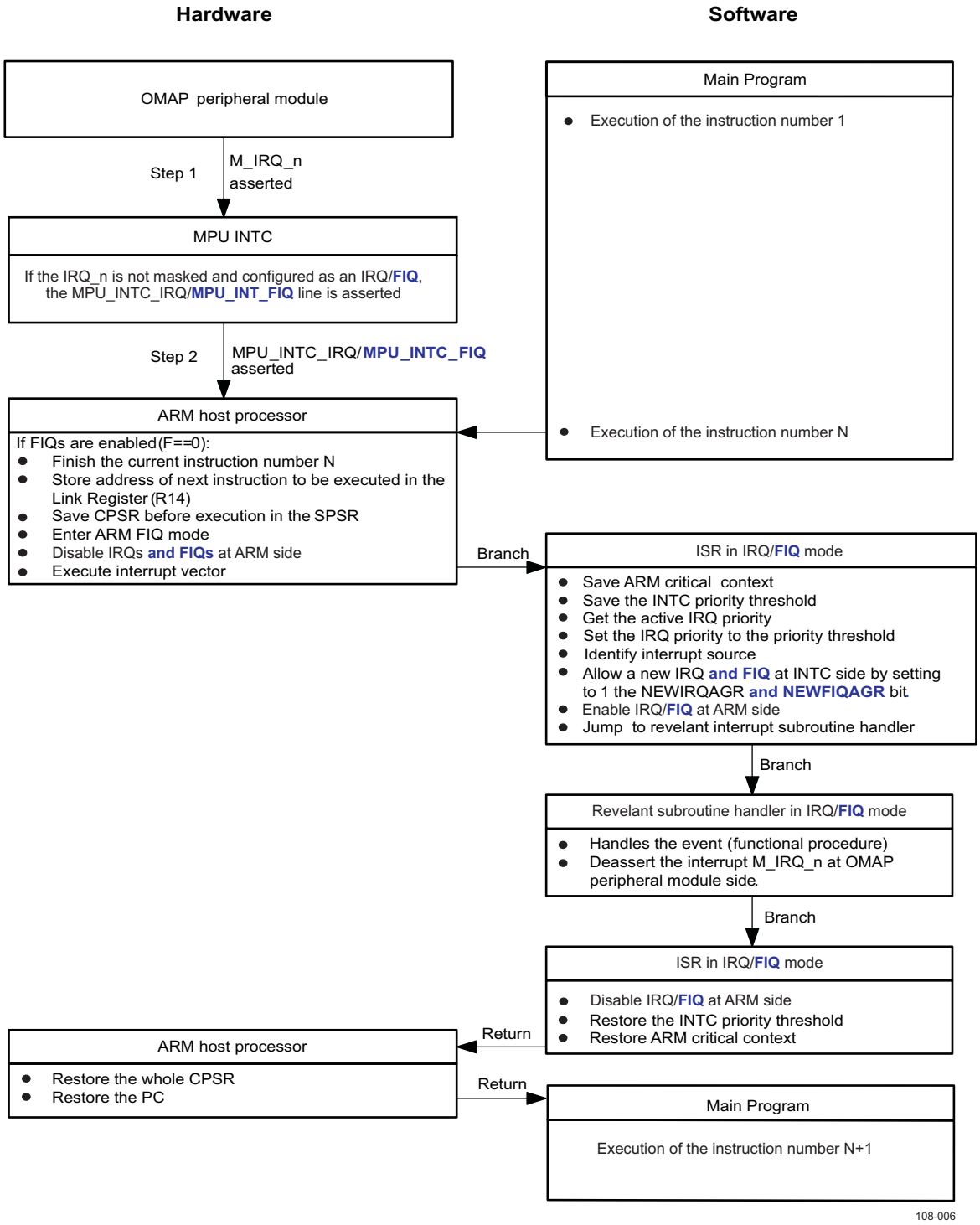
     ; Step 2 : Restore the INTCPS_THRESHOLD register from R12
     LDR R0, INTCPS_THRESHOLD_ADDR
     STR R12, [R0]

     ; Step 3 : Restore critical context
     MSR SPSR, R11                                   ; Restore the SPSR from R11
     LDMFD SP!, {R0-R12, LR}                         ; Restore working registers and Link
register

     ; Return after handling the interrupt
     SUBS PC, LR, #4
```

Figure 10-6 shows the nested IRQ/**FIQ** processing sequence from the originating device peripheral module to the main program interruption.

Figure 10-6. Nested IRQ/FIQ Sequence



Note: The differences between the IRQ and the FIQ sequence are highlighted in blue and bold characters.

10.5.4 MPU INTC Spurious Interrupt Handling

The spurious flag indicates whether the result of the sorting (a window of 10 INTC functional clock cycles after the interrupt assertion) is invalid. The sorting is invalid if:

- The interrupt that triggered the sorting is no longer active during the sorting.
- A change in the mask has affected the result during the sorting time.

As a result, the values in the MPU_INTC.INTCPS_MIRn, MPU_INTC.INTCPS_ILRm, or MPU_INTC.INTCPS_MIR_SETn registers must not be changed while the corresponding interrupt is asserted. If these registers are changed within the 10-cycle window after the interrupt assertion, only the active interrupt input that triggered the sort can be masked before its turn in the sort. The resulting values of the following registers become invalid:

- MPU_INTC.INTCPS_SIR_IRQ
- MPU_INTC.INTCPS_SIR_FIQ
- MPU_INTC.INTCPS_IRQ_PRIORITY
- MPU_INTC. INTCPS_FIQ_PRIORITY

This condition is detected for both IRQ and FIQ, and the invalid status is flagged across the SPURIOUSIRQFLAG (see Note 1) and SPURIOUSFIQFLAG (see Note 2) bit fields in the SIR and PRIORITY registers. A 0 indicates valid and a 1 indicates invalid interrupt number and priority. The invalid indication can be tested in software as a false register value.

Notes:

1. The MPU_INTC.INTCPS_SIR_IRQ[31:7] SPURIOUSIRQFLAG bit field is a copy of the MPU_INTC.INTCPS_IRQ_PRIORITY[31:7] SPURIOUSIRQFLAG bit field.
2. The MPU_INTC.INTCPS_SIR_FIQ[31:7] SPURIOUSFIQFLAG bit field is a copy of the MPU_INTC.INTCPS_FIQ_PRIORITY[31:7] SPURIOUSFIQFLAG bit field.

10.6 Interrupt Controller Register Manual

10.6.1 Instance Summary

Table 10-5 lists the base address and address space for the INTC instances.

Table 10-5. INTC Instance Summary

Module Name	Base Address	Size
MPU INTC	0x4820 0000	4K bytes
Modem INTC	0x480C 7000	4K bytes

10.6.2 Register Summary

CAUTION

MPU INTC and Modem INTC registers are limited to 32-bit and 16-bit data accesses; 8-bit data access is not allowed and can corrupt the register content.

In Section 10.6.3, each register from MPU_INTC.INTCPS_ITRn to MPU_INTC.INTCPS_PENDING_FIQn contains 32 bits, 1 bit for each interrupt (in ascending order: bit 0 of the MPU_INTC.INTCPS_ITR0 register applies to interrupt line 0; bit 0 of the MPU_INTC.INTCPS_ITR1 register applies to interrupt line 32).

Table 10-6. MPU INTC Register Summary

Register Name	Type	Register Width (Bits)	Address Offset	MPU INTC Physical Address
INTCPS_REVISION	R	32	0x0000 0000	0x4820 0000
INTCPS_SYSCONFIG	RW	32	0x0000 0010	0x4820 0010
INTCPS_SYSSTATUS	R	32	0x0000 0014	0x4820 0014
INTCPS_SIR_IRQ	R	32	0x0000 0040	0x4820 0040
INTCPS_SIR_FIQ	R	32	0x0000 0044	0x4820 0044
INTCPS_CONTROL	RW	32	0x0000 0048	0x4820 0048
INTCPS_PROTECTION	RW	32	0x0000 004C	0x4820 004C
INTCPS_IDLE	RW	32	0x0000 0050	0x4820 0050
INTCPS_IRQ_PRIORITY	RW	32	0x0000 0060	0x4820 0060
INTCPS_FIQ_PRIORITY	RW	32	0x0000 0064	0x4820 0064
INTCPS_THRESHOLD	RW	32	0x0000 0068	0x4820 0068
INTCPS_ITRn (1)	R	32	0x0000 0080 + (0x20 * n)	0x4820 0080 + (0x20 * n)
INTCPS_MIRn (1)	RW	32	0x0000 0084 + (0x20 * n)	0x4820 0084 + (0x20 * n)
INTCPS_MIR_CLEARn (1)	W	32	0x0000 0088 + (0x20 * n)	0x4820 0088 + (0x20 * n)
INTCPS_MIR_SETn (1)	W	32	0x0000 008C + (0x20 * n)	0x4820 008C + (0x20 * n)
INTCPS_ISR_SETn (1)	RW	32	0x0000 0090 + (0x20 * n)	0x4820 0090 + (0x20 * n)
INTCPS_ISR_CLEARn (1)	W	32	0x0000 0094 + (0x20 * n)	0x4820 0094 + (0x20 * n)
INTCPS_PENDING_IRQn (1)	R	32	0x0000 0098 + (0x20 * n)	0x4820 0098 + (0x20 * n)
INTCPS_PENDING_FIQn (1)	R	32	0x0000 009C + (0x20 * n)	0x4820 009C + (0x20 * n)
INTCPS_ILRm (2)	RW	32	0x0000 0100 + (0x4 * m)	0x4820 0100 + (0x4 * m)

(1) n = 0 to 2
(2) m = 0 to 95

Table 10-7. Modem INTC Register Summary

Register Name	Type	Register Width (Bits)	Address Offset	Modem INTC Physical Address
INTC_SYSCONFIG	RW	32	0x0000 0010	0x480C 7010
INTC_IDLE	RW	32	0x0000 0050	0x480C 7050

10.6.3 MPU INTC Register Descriptions

Table 10-8 through Table 10-46 describe the MPU INTC registers.

Table 10-8. INTCPS_REVISION

Address Offset	0x000		
Physical Address	0x4820 0000	**Instance**	MPU INTC
Description	This register contains the IP revision code.		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																								REV							

Bits	Field Name	Description	Type	Reset
31:8	Reserved	Read returns reset value.	R	0x000000
7:0	REV	IP revision [7:4] Major revision [3:0] Minor revision Examples: 0x10 for 1.0, 0x21 for 2.1	R	See (1)

(1) TI internal data

Table 10-9. Register Call Summary for Register INTCPS_REVISION

Interrupt Controller Register Manual

- Register Summary: [0]

Table 10-10. INTCPS_SYSCONFIG

Address Offset	0x010		
Physical Address	0x4820 0010	**Instance**	MPU INTC
	0x480C 8010		Modem INTC
Description	This register controls various parameters of the module interface.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																														SOFTRESET	AUTOIDLE

Bits	Field Name	Description	Type	Reset
31:2	Reserved	Write 0s for future compatibility. Read returns reset value.	R	0x00000000
1	SOFTRESET	Software reset. Set this bit to trigger a module reset. The bit is automatically reset by the hardware. Read returns 0. Write 0x0: No functional effect Write 0x1: The module is reset.	RW	0
0	AUTOIDLE	Internal interface clock gating strategy 0x0: Interface clock is free-running.	RW	0

Bits	Field Name	Description	Type	Reset
		0x1: Automatic interface clock gating strategy is applied, based on the interface bus activity.		

Table 10-11. Register Call Summary for Register INTCPS_SYSCONFIG

MPU Subsystem INTCPS Integration • Hardware and Software Reset: [0]
Interrupt Controller Functional Description • Module Power Saving: [1] [2]
Interrupt Basic Programming Model • Initialization Sequence: [3]
Interrupt Controller Register Manual • Register Summary: [4]

Table 10-12. INTCPS_SYSSTATUS

Address Offset	0x014		
Physical Address	0x4820 0014	**Instance**	MPU INTC
Description	This register provides status information about the module.		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																															RESETDONE

Bits	Field Name	Description	Type	Reset
31:1	Reserved	Read returns reset value.	R	0x00000000
0	RESETDONE	Internal reset monitoring	R	-
		Read 0x0: Internal module reset is ongoing.		
		Read 0x1: Reset complete		

Table 10-13. Register Call Summary for Register INTCPS_SYSSTATUS

Interrupt Controller Register Manual • Register Summary: [0]

Table 10-14. INTCPS_SIR_IRQ

Address Offset	0x040		
Physical Address	0x4820 0040	**Instance**	MPU INTC
Description	This register supplies the currently active IRQ interrupt number.		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SPURIOUSIRQFLAG																								ACTIVEIRQ							

Bits	Field Name	Description	Type	Reset
31:7	SPURIOUSIRQFLAG	Spurious IRQ flag	R	0x1FFFFFF
6:0	ACTIVEIRQ	Active IRQ number	R	0x00

Table 10-15. Register Call Summary for Register INTCPS_SIR_IRQ

Interrupt Controller Functional Description
- Priority Sorting: [0]
- Interrupt Latency: [1] [2]

Interrupt Basic Programming Model
- MPU INTC Processing Sequence: [3]
- MPU INTC Preemptive Processing Sequence: [4]
- MPU INTC Spurious Interrupt Handling: [5] [6]

Interrupt Controller Register Manual
- Register Summary: [7]

Table 10-16. INTCPS_SIR_FIQ

Address Offset	0x044		
Physical Address	0x4820 0044	**Instance**	MPU INTC
Description	This register supplies the currently active FIQ interrupt number.		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SPURIOUSFIQFLAG																								ACTIVEFIQ							

Bits	Field Name	Description	Type	Reset
31:7	SPURIOUSFIQFLAG	Spurious FIQ flag	R	0x1FFFFFF
6:0	ACTIVEFIQ	Active FIQ number	R	0x00

Table 10-17. Register Call Summary for Register INTCPS_SIR_FIQ

Interrupt Controller Functional Description
- Priority Sorting: [0]
- Interrupt Latency: [1] [2]

Interrupt Basic Programming Model
- MPU INTC Processing Sequence: [3]
- MPU INTC Preemptive Processing Sequence: [4]
- MPU INTC Spurious Interrupt Handling: [5] [6]

Interrupt Controller Register Manual
- Register Summary: [7]

Table 10-18. INTCPS_CONTROL

Address Offset	0x048		
Physical Address	0x4820 0048	**Instance**	MPU INTC
Description	This register contains the new interrupt agreement bits.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																														NEWFIQAGR	NEWIRQAGR

Bits	Field Name	Description	Type	Reset
31:2	Reserved	Write 0s for future compatibility. Read returns reset value.	R	0x00000000
1	NEWFIQAGR	Reset FIQ output and enable new FIQ generation. Write 0x0: No functional effect	W	-

Bits	Field Name	Description	Type	Reset
		Write 0x1: Reset FIQ output and enable new FIQ generation.		
0	NEWIRQAGR	New IRQ generation	W	-
		Write 0x0: No functional effect		
		Write 0x1: Reset IRQ output and enable new IRQ generation.		

Table 10-19. Register Call Summary for Register INTCPS_CONTROL

Interrupt Controller Functional Description • Priority Sorting: [0]
Interrupt Basic Programming Model • MPU INTC Preemptive Processing Sequence: [1]
Interrupt Controller Register Manual • Register Summary: [2]

Table 10-20. INTCPS_PROTECTION

Address Offset	0x04C		
Physical Address	0x4820 004C	**Instance**	MPU INTC
Description	This register controls protection of the other registers. It can be accessed only in supervisor mode, regardless of the current value of the protection bit.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																															PROTECTION

Bits	Field Name	Description	Type	Reset
31:1	Reserved	Write 0s for future compatibility. Read returns reset value.	R	0x00000000
0	PROTECTION	Protection mode	RW	0
		0x0: Protection mode is disabled (default).		
		0x1: Protection mode is enabled. When enabled, all the MPU INTC registers are accessible only in privileged mode.		

Table 10-21. Register Call Summary for Register INTCPS_PROTECTION

Interrupt Controller Functional Description • Register Protection: [0] [1]
Interrupt Controller Register Manual • Register Summary: [2]

Table 10-22. INTCPS_IDLE

Address Offset	0x050		
Physical Address	0x4820 0050	**Instance**	MPU INTC
	0x480C 8050		Modem INTC
Description	This register controls the functional clock auto-idle and the synchronizer clock auto-gating.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																														TURBO	FUNCIDLE

Bits	Field Name	Description	Type	Reset
31:2	Reserved	Write 0s for future compatibility. Read returns reset value.	R	0x00000000
1	TURBO	Input synchronizer clock auto-gating	RW	0
		0x0: Input synchronizer clock is free-running (default).		
		0x1: Input synchronizer clock is auto-gated based on interrupt input activity.		
0	FUNCIDLE	Functional clock idle mode	RW	0
		0x0: Functional clock gating strategy is applied (default).		
		0x1: Functional clock is free-running.		

Table 10-23. Register Call Summary for Register INTCPS_IDLE

Interrupt Controller Functional Description • Module Power Saving: [0] [1] [2] • Interrupt Latency: [3] [4]
Interrupt Basic Programming Model • Initialization Sequence: [5]
Interrupt Controller Register Manual • Register Summary: [6]

Table 10-24. INTCPS_IRQ_PRIORITY

Address Offset	0x060		
Physical Address	0x4820 0060	**Instance**	MPU INTC
Description	This register supplies the currently active IRQ priority level.		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SPURIOUSIRQFLAG																								IRQPRIORITY							

Bits	Field Name	Description	Type	Reset
31:6	SPURIOUSIRQFLAG	Spurious IRQ flag	R	0x3FFFFFF
5:0	IRQPRIORITY	Current IRQ priority	R	0x00

Table 10-25. Register Call Summary for Register INTCPS_IRQ_PRIORITY

Interrupt Basic Programming Model • MPU INTC Preemptive Processing Sequence: [0] • MPU INTC Spurious Interrupt Handling: [1] [2]
Interrupt Controller Register Manual • Register Summary: [3]

Table 10-26. INTCPS_FIQ_PRIORITY

Address Offset	0x064		
Physical Address	0x4820 0064	**Instance**	MPU INTC
Description	This register supplies the currently active FIQ priority level.		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SPURIOUSFIQFLAG																								FIQPRIORITY							

Bits	Field Name	Description	Type	Reset
31:6	SPURIOUSFIQFLAG	Spurious FIQ flag	R	0x3FFFFFF
5:0	FIQPRIORITY	Current FIQ priority	R	0x00

Table 10-27. Register Call Summary for Register INTCPS_FIQ_PRIORITY

Interrupt Basic Programming Model
- MPU INTC Preemptive Processing Sequence: [0]
- MPU INTC Spurious Interrupt Handling: [1] [2]

Interrupt Controller Register Manual
- Register Summary: [3]

Table 10-28. INTCPS_THRESHOLD

Address Offset	0x068		
Physical Address	0x4820 0068	**Instance**	MPU INTC
Description	This register sets the priority threshold.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																								PRIORITYTHRESHOLD							

Bits	Field Name	Description		Type	Reset
31:8	Reserved	Write 0s for future compatibility. Read returns reset value.		R	0x000000
7:0	PRIORITYTHRESHOLD	Priority threshold		RW	0xFF
		Write 0xFF:	Priority threshold disabled		
		Write 0x0 to 0x3F:	Priority threshold enabled		

Table 10-29. Register Call Summary for Register INTCPS_THRESHOLD

Interrupt Controller Functional Description
- Masking: [0]

Interrupt Basic Programming Model
- MPU INTC Preemptive Processing Sequence: [1] [2]

Interrupt Controller Register Manual
- Register Summary: [3]

Table 10-30. INTCPS_ITRn

Address Offset	0x080 + (0x20 * n)	**Index**	n = 0 to 2
Physical Address	0x4820 0080 + (0x20 * n)	**Instance**	MPU INTC
Description	This register shows the raw interrupt input status before masking.		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ITR																															

Bits	Field Name	Description	Type	Reset
31:0	ITR	Interrupt status before masking	R	Depends on interrupt inputs

Table 10-31. Register Call Summary for Register INTCPS_ITRn

Interrupt Controller Functional Description
- Masking: [0]

Interrupt Controller Register Manual
- Register Summary: [1] [2]

Table 10-32. INTCPS_MIRn

Address Offset	0x084 + (0x20 * n)	**Index**	n = 0 to 2
Physical Address	0x4820 0084 + (0x20 * n)	**Instance**	MPU INTC
Description	This register contains the interrupt mask.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MIR																															

Bits	Field Name	Description	Type	Reset
31:0	MIR	Interrupt mask 0x1: The interrupt is masked 0x0: The interrupt is unmasked	RW	0xFFFFFFFF

Table 10-33. Register Call Summary for Register INTCPS_MIRn

Interrupt Controller Functional Description
- Masking: [0]

Interrupt Basic Programming Model
- Initialization Sequence: [1] [2] [3]
- MPU INTC Processing Sequence: [4]
- MPU INTC Spurious Interrupt Handling: [5]

Interrupt Controller Register Manual
- Register Summary: [6]

Table 10-34. INTCPS_MIR_CLEARn

Address Offset	0x088 + (0x20 * n)	**Index**	n = 0 to 2
Physical Address	0x4820 0088 + (0x20 * n)	**Instance**	MPU INTC
Description	This register is used to clear the interrupt mask bits.		
Type	W		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MIRCLEAR																															

Bits	Field Name	Description	Type	Reset
31:0	MIRCLEAR	Clear the interrupt mask bits. Read returns 0. Write 0x1: Clears the MIR mask bit to 0 Write 0x0: No functional effect	W	0x00000000

Table 10-35. Register Call Summary for Register INTCPS_MIR_CLEARn

Interrupt Basic Programming Model
- Initialization Sequence: [0]

Interrupt Controller Register Manual
- Register Summary: [1]

Table 10-36. INTCPS_MIR_SETn

Address Offset	0x08C + (0x20 * n)	**Index**	n = 0 to 2
Physical Address	0x4820 008C + (0x20 * n)	**Instance**	MPU INTC
Description	This register is used to set the interrupt mask bits.		
Type	W		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MIRSET																															

Bits	Field Name	Description	Type	Reset
31:0	MIRSET	Mask the interrupt bits. Read returns 0. Write 0x0: No functional effect Write 0x1: Sets the MIR mask bit to 1.	W	0x00000000

Table 10-37. Register Call Summary for Register INTCPS_MIR_SETn

Interrupt Basic Programming Model
- Initialization Sequence: [0]
- MPU INTC Spurious Interrupt Handling: [1]

Interrupt Controller Register Manual
- Register Summary: [2]

Table 10-38. INTCPS_ISR_SETn

Address Offset	0x090 + (0x20 * n)	**Index**	n = 0 to 2
Physical Address	0x4820 0090 + (0x20 * n)	**Instance**	MPU INTC
Description	This register is used to set the software interrupt bits. It is also used to read the currently active software interrupts.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ISRSET																															

Bits	Field Name	Description	Type	Reset
31:0	ISRSET	Set the software interrupt bits. Read returns the currently active software interrupts.	RW	0x00000000
		Write 0x0: No functional effect		
		Write 0x1: Sets the software interrupt bits to 1.		

Table 10-39. Register Call Summary for Register INTCPS_ISR_SETn

Interrupt Controller Functional Description
- Input Selection: [0]

Interrupt Controller Register Manual
- Register Summary: [1]

Table 10-40. INTCPS_ISR_CLEARn

Address Offset	0x094 + (0x20 * n)	**Index**	n = 0 to 2
Physical Address	0x4820 0094 + (0x20 * n)	**Instance**	MPU INTC
Description	This register is used to clear the software interrupt bits.		
Type	W		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ISRCLEAR																															

Bits	Field Name	Description	Type	Reset
31:0	ISRCLEAR	Clear the software interrupt bits. Read returns 0.	W	0x00000000
		Write 0x0: No functional effect		
		Write 0x1: Clears the software interrupt bits to 0.		

Table 10-41. Register Call Summary for Register INTCPS_ISR_CLEARn

Interrupt Controller Functional Description
- Input Selection: [0]

Interrupt Controller Register Manual
- Register Summary: [1]

Table 10-42. INTCPS_PENDING_IRQn

Address Offset	0x098 + (0x20 * n)	**Index**	n = 0 to 2
Physical Address	0x4820 0098 + (0x20 * n)	**Instance**	MPU INTC
Description	This register contains the IRQ status after masking.		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PENDINGIRQ																															

Bits	Field Name	Description	Type	Reset
31:0	PENDINGIRQ	IRQ status after masking.	R	0x00000000

Table 10-43. Register Call Summary for Register INTCPS_PENDING_IRQn

Interrupt Controller Functional Description
- Masking: [0]
- Priority Sorting: [1]

Interrupt Controller Register Manual
- Register Summary: [2]

Table 10-44. INTCPS_PENDING_FIQn

Address Offset	0x09C + (0x20 * n)	**Index**	n = 0 to 2
Physical Address	0x4820 009C + (0x20 * n)	**Instance**	MPU INTC
Description	This register contains the FIQ status after masking.		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PENDINGFIQ																															

Bits	Field Name	Description	Type	Reset
31:0	PENDINGFIQ	FIQ status after masking.	R	0x00000000

Table 10-45. Register Call Summary for Register INTCPS_PENDING_FIQn

Interrupt Controller Functional Description
- Masking: [0]
- Priority Sorting: [1]

Interrupt Controller Register Manual
- Register Summary: [2] [3]

Table 10-46. INTCPS_ILRm

Address Offset	0x100 + (0x4 * m)	**Index**	m = 0 to 95
Physical Address	0x4820 0100 + (0x4 * m)	**Instance**	MPU INTC
Description	These registers contain the priority for the interrupts and the FIQ/IRQ steering.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																								PRIORITY						Reserved	FIQNIRQ

Bits	Field Name	Description	Type	Reset
31:8	Reserved	Write 0s for future compatibility. Read returns reset value.	R	0x000000
7:2	PRIORITY	Interrupt priority	RW	0x00
1	Reserved	Write 0 for future compatibility. Read returns reset value.	R	0
0	FIQNIRQ	Interrupt IRQ FIQ mapping. Read returns reset value. Write 0x0: Interrupt is routed to IRQ. Write 0x1: Interrupt is routed to FIQ.	RW	0

Table 10-47. Register Call Summary for Register INTCPS_ILRm

Interrupt Controller Functional Description • Masking: [0] • Priority Sorting: [1] [2]
Interrupt Basic Programming Model • Initialization Sequence: [3] • MPU INTC Processing Sequence: [4] [5] • MPU INTC Spurious Interrupt Handling: [6]
Interrupt Controller Register Manual • Register Summary: [7]

10.6.4 Modem INTC Register Descriptions

Table 10-48 and Table 10-50 describe useful modem INTC registers.

Table 10-48. INTC_SYSCONFIG

Address Offset	0x010		
Physical Address	0x480C 7010	**Instance**	Modem INTC
Description	This register controls various parameters of the module interface.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																														SOFTRESET	AUTOIDLE

Bits	Field Name	Description	Type	Reset
31:2	Reserved	Write 0s for future compatibility. Read returns reset value.	R	0x00000000
1	SOFTRESET	Software reset. Set this bit to trigger a module reset. The bit is automatically reset by the hardware. Read returns 0. 0x0: No functional effect 0x1: The module is reset.	RW	0
0	AUTOIDLE	Internal interface clock gating strategy 0x0: Interface clock is free-running. 0x1: Automatic interface clock gating strategy is applied, based on the interface bus activity.	RW	0

Table 10-49. Register Call Summary for Register INTC_SYSCONFIG

Interrupt Controller Register Manual • Register Summary: [0]

Table 10-50. INTC_IDLE

Address Offset	0x050		
Physical Address	0x480C 7050	**Instance**	Modem INTC
Description	This register controls the functional clock auto-idle and the synchronizer clock auto-gating.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																														TURBO	FUNCIDLE

Bits	Field Name	Description	Type	Reset
31:2	Reserved	Write 0s for future compatibility. Read returns reset value.	R	0x00000000
1	TURBO	Input synchronizer clock auto-gating	RW	0
		0x0: Input synchronizer clock is free-running (default).		
		0x1: Input synchronizer clock is auto-gated based on interrupt input activity.		
0	FUNCIDLE	Functional clock idle mode	RW	0
		0x0: Functional clock gating strategy is applied (default).		
		0x1: Functional clock is free-running.		

Table 10-51. Register Call Summary for Register INTC_IDLE

Interrupt Controller Register Manual

- Register Summary: [0]

Memory Subsystem

This chapter describes the memory subsystem of the OMAP34xx multimedia device.

Topic **Page**

11.1 General-Purpose Memory Controller 1072
11.2 SDRAM Controller (SDRC) Subsystem 1183
11.3 On-Chip Memory Subsystem..... 1303

11.1 General-Purpose Memory Controller

11.1.1 General-Purpose Memory Controller Overview

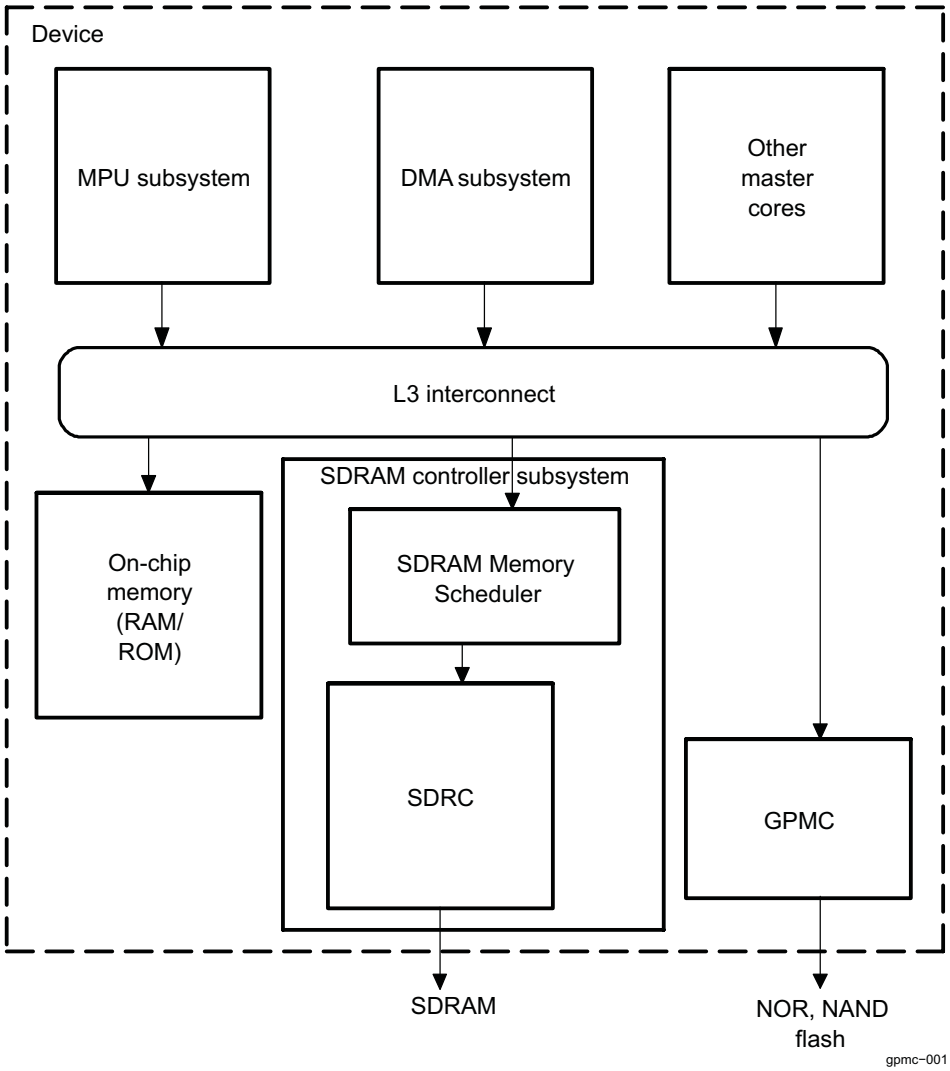
The general-purpose memory controller (GPMC) is the OMAP™2 unified memory controller (UMC) dedicated to interfacing external memory devices:

- Asynchronous SRAM-like memories and application-specific integrated circuit (ASIC) devices
- Asynchronous, synchronous, and page mode (only available in non-muxed mode) burst NOR flash devices
- NAND flash
- Pseudo-SRAM devices

Note: Page mode is only available in non-muxed mode. The non-muxed mode is described in this chapter even though its use is very limited (address space limited to 2 KBytes).

Figure 11-1 shows the environment of the GPMC.

Figure 11-1. GPMC Environment



11.1.1.1 GPMC Features

The GPMC is the OMAP2 16-bit external memory controller. The GPMC data access engine provides a flexible programming model for communication with all standard memories. The GPMC supports various accesses:

- Asynchronous read/write access
- Asynchronous read page access (4, 8, 16 Word16)
- Synchronous read/write access
- Synchronous read/write burst access without wrap capability (4, 8, 16 Word16)
- Synchronous read/write burst access with wrap capability (4, 8, 16 Word16)
- Address/data-multiplexed access
- Little- and big-endian access

The GPMC can communicate with a wide range of external devices:

- External asynchronous or synchronous 8-bit wide memory or device (non-burst device)
- External asynchronous or synchronous 16-bit wide memory or device
- External 16-bit nonmultiplexed device with limited address range (2 Kbytes)
- External 16-bit address/data-multiplexed NOR flash device
- External 8-bit and 16-bit NAND flash device
- External 16-bit pseudo-static random access memory (pSRAM) device

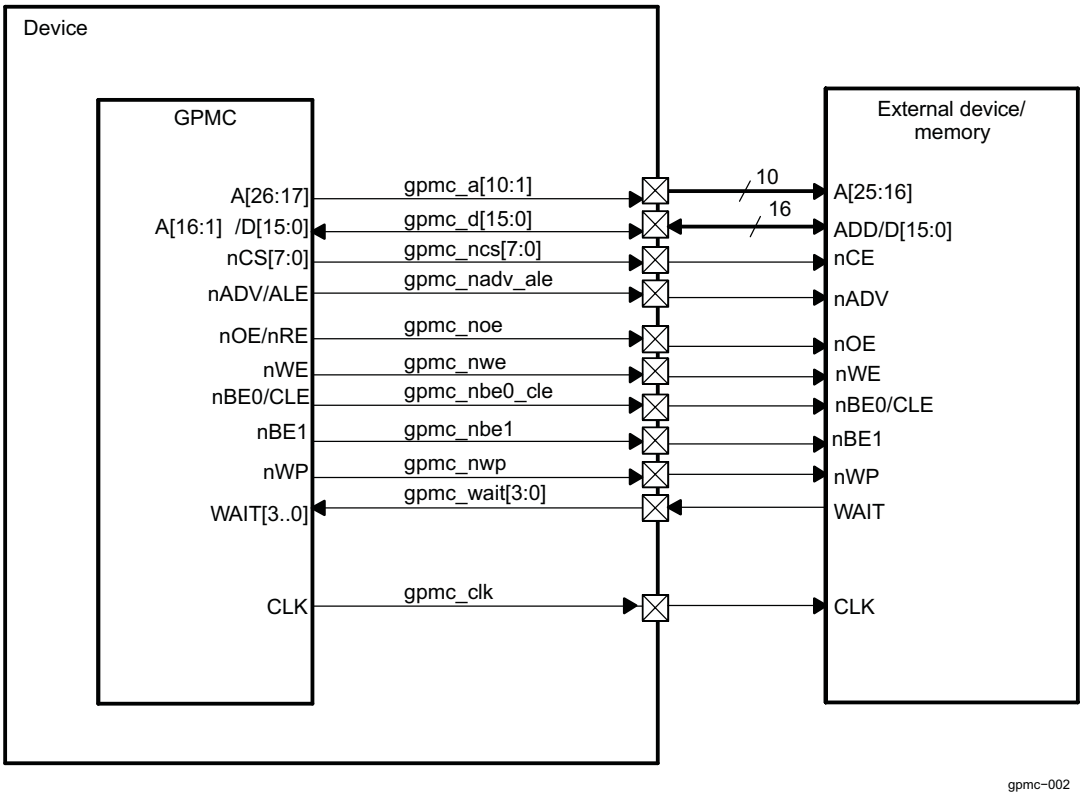
The GPMC supports up to eight chip-select regions of programmable size, and programmable base addresses in a total address space of 1 Gbyte.

11.1.2 GPMC Environment

Figure 11-2 and Figure 11-3 show two GPMC external connection options:

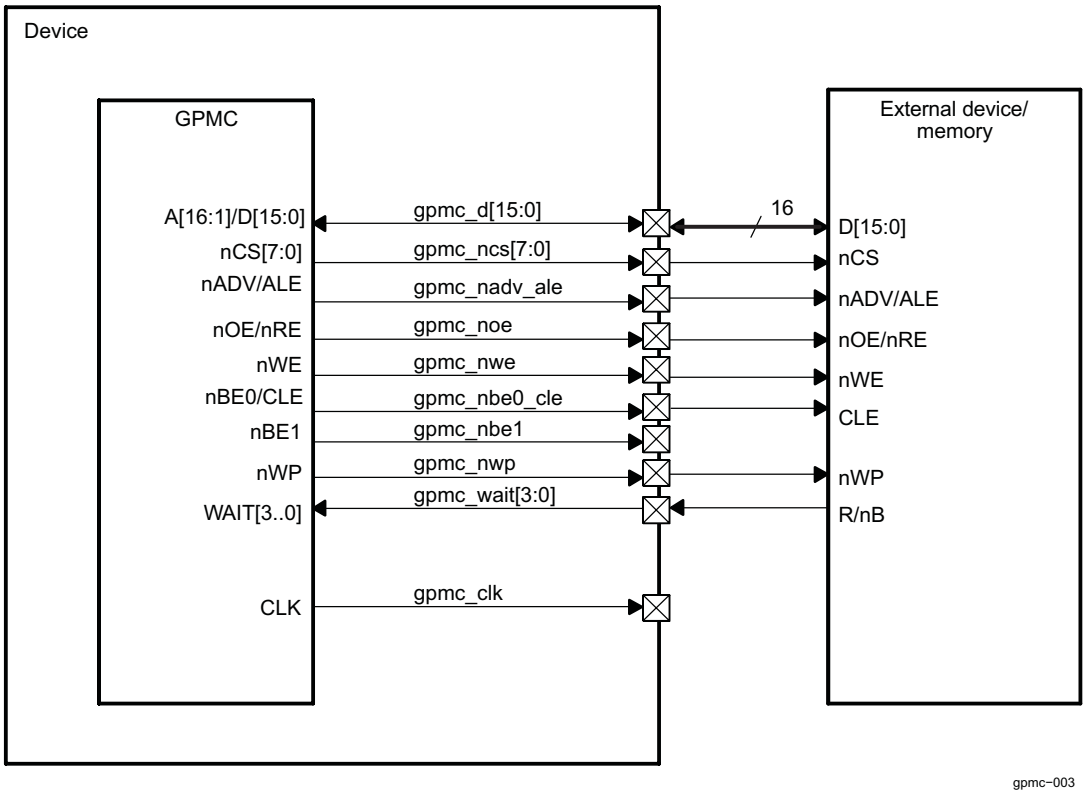
- GPMC to 16-bit address/data-multiplexed memory
  Figure 11-2 shows a connection between the GPMC and a 16-bit synchronous address/data-multiplexed external memory device.
- GPMC to 16-bit NAND device
  Figure 11-3 shows a connection between the GPMC and a 16-bit NAND device.

Figure 11-2. GPMC to 16-Bit Address/Data-Multiplexed Memory



Note: The OMAP device does not provide the A0 byte address line required for random-byte addressable 8-bit wide device interfacing (for multiplexed and nonmultiplexed protocol). Hence, an 8-bit device must be connected to the D[7:0] / gpmc_d[7:0] data bus (rather than D[15:8] / gpmc_d[15:8]) of the GPMC controller. This limits the use of 8-bit wide device interfacing to byte-alias access.

Figure 11-3. GPMC to 16-Bit NAND Device



Note: The OMAP device does not provide the A0 byte address line required for random-byte addressable 8-bit wide device interfacing (for multiplexed and nonmultiplexed protocol). Hence, an 8-bit device must be connected to the D[7:0]/gpmc_d[7:0] data bus (rather than D[15:8]/gpmc_d[15:8]) of the GPMC controller. This limits the use of 8-bit wide device interfacing to byte-alias accesses.

Table 11-1 lists the GPMC subsystem I/O pins.

Table 11-1. GPMC I/O Description

Pin Name	I/O	Description
gpmc_a[10:1]	O	Address
gpmc_d[15:0]	I/O	Data
gpmc_ncs[7:0]	O	Chip-selects (active low)
gpmc_clk	I/O	Clock (1)
gpmc_nadv_ale	O	Address valid (active low). Also used as address latch enable (active high) for NAND protocol memories.
gpmc_noe_nre	O	Output enable (active low). Also used as read enable (active low) for NAND protocol memories.
gpmc_nwe	O	Write enable (active low)
gpmc_nbe0_cle	O	Lower-byte enable (active low). Also used as command latch enable for NAND protocol memories.
gpmc_nbe1	O	Byte 1 enable (active low)
gpmc_nwp	O	Write protect (active low)
gpmc_wait[3:0]	I	External wait signal for NOR and NAND protocol memories

(1) This output signal is also used as re-timing input

Table 11-1. GPMC I/O Description (continued)

Pin Name	I/O	Description
gpmc_io_dir	O	gpmc_d[15:0] signal direction control: Low during transmit (for write access: data OUT from GPMC to memory), High during receive (for read access: data IN from memory to GPMC)

Table 11-2 shows the use of address and data GPMC controller pins based on the type of external device.

Table 11-2. GPMC Pin Multiplexing Options

GPMC Pin	Multiplexed Address Data 16-Bit Device	Nonmultiplexed Address Data 16-Bit Device With LIMITED-ADDRESS Bit Enabled	16-Bit NAND Device	8-Bit NAND Device
gpmc_a[10]	A26	A10	Not used	Not used
gpmc_a[9]	A25	A9	Not used	Not used
gpmc_a[8]	A24	A8	Not used	Not used
gpmc_a[7]	A23	A7	Not used	Not used
gpmc_a[6]	A22	A6	Not used	Not used
gpmc_a[5]	A21	A5	Not used	Not used
gpmc_a[4]	A20	A4	Not used	Not used
gpmc_a[3]	A19	A3	Not used	Not used
gpmc_a[2]	A18	A2	Not used	Not used
gpmc_a[1]	A17	A1	Not used	Not used
gpmc_d[15]	A16/D15	D15	D15	Not used
gpmc_d[14]	A15/D14	D14	D14	Not used
gpmc_d[13]	A14/D13	D13	D13	Not used
gpmc_d[12]	A13/D12	D12	D12	Not used
gpmc_d[11]	A12/D11	D11	D11	Not used
gpmc_d[10]	A11/D10	D10	D10	Not used
gpmc_d[9]	A10/D9	D9	D9	Not used
gpmc_d[8]	A9/D8	D8	D8	Not used
gpmc_d[7]	A8/D7	D7	D7	D7
gpmc_d[6]	A7/D6	D6	D6	D6
gpmc_d[5]	A6/D5	D5	D5	D5
gpmc_d[4}	A5/D4	D4	D4	D4
gpmc_d[3]	A4/D3	D3	D3	D3
gpmc_d[2]	A3/D2	D2	D2	D2
gpmc_d[1]	A2/D1	D1	D1	D1
gpmc_d[0]	A1/D0	D0	D0	D0

Enabling the GPMC.GPMC_CONFIG[1] LIMITEDADDRESS bit forces A[26:11] to 1 on the GPMC I/O side. Thus, only devices with 2 Kbytes of addressing space can be accessed using gpmc_a[10:1].

With all device types, the GPMC does not drive unnecessary address lines. They stay at their reset value of 0x00.

Address mapping supports address/data-multiplexed 16-bit wide devices:

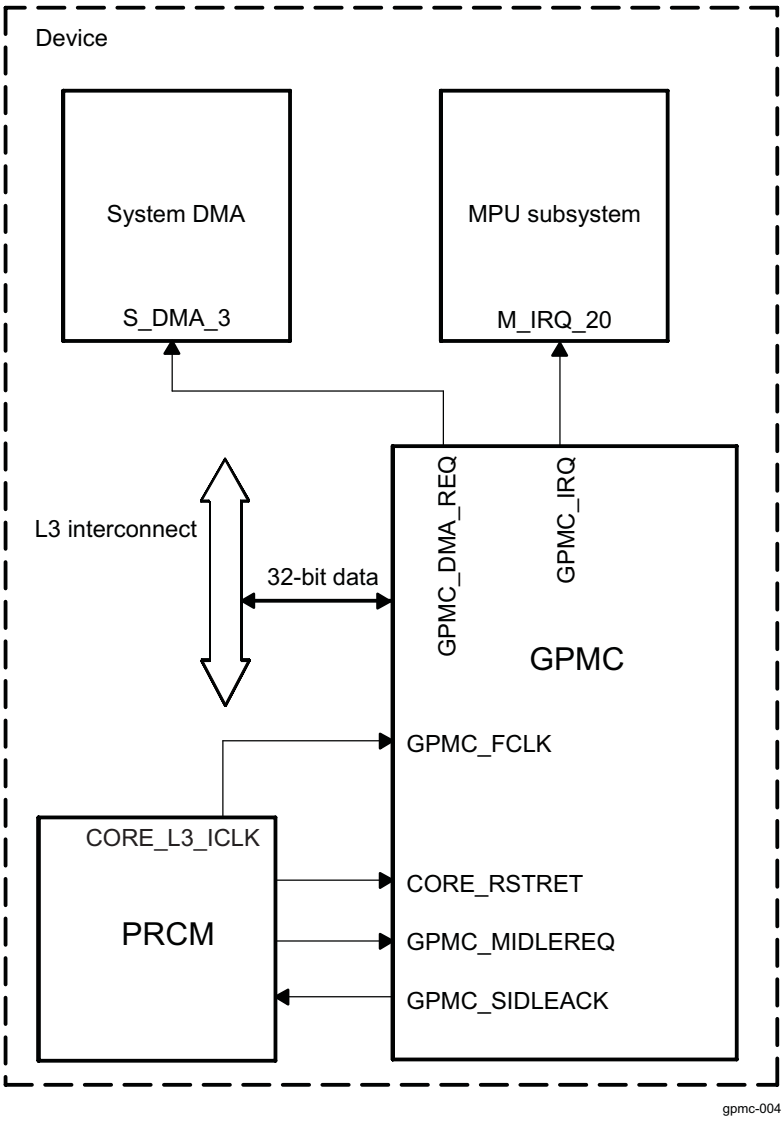
- To minimize the number of IC pins required for the external memory connection, the NOR flash memory controller supports multiplexed address and data memory devices without adding logic externally.
- Multiplexing mode can be selected through the GPMC.GPMC_CONFIG1_i[9] MUXADDDATA bit (i = 0 to 7).
- Asynchronous page mode is not supported for multiplexed address and data devices.

11.1.3 GPMC Integration

11.1.3.1 Description

Figure 11-4 shows how the GPMC interacts with other modules in the OMAP device.

Figure 11-4. GPMC Integration in the OMAP Device



11.1.3.2 Clocking, Reset, and Power-Management Scheme

11.1.3.2.1 Clocking

The GPMC use a single clock, GPMC_FCLK, which comes internally from the power, reset, and clock-management (PRCM) module and runs at the L3 interconnect frequency. Its source is the PRCM module, CORE_L3_ICLK output. CORE_L3_ICLK belongs to the L3 interconnect clock domain.

For details, see Chapter 4, *Power, Reset, and Clock Management.*

GPMC_CLK is the external clock provided to the attached synchronous memory or device. The GPMC_CLK clock frequency is the GPMC_FCLK clock frequency divided by 1, 2, 3, or 4, depending on the GPMC.GPMC_CONFIG1_i[1:0] GPMCFCLKDIVIDER bit field (where i = 0 to 7).

Note: When the GPMC is configured for synchronous mode, the GPMC_CLK signal (which is an output) must also be set as an input (CONTROL.CONTROL_PADCONF_GPMC_NCS7[24] INPUTENABLE1 = 1). GPMC_CLK is looped back through the output and input buffers of the corresponding GPMC_CLK pad at the OMAP boundary. The looped-back clock is used to synchronize the sampling of the memory signals.

11.1.3.2.2 Hardware Reset

A global reset of the GPMC occurs through activation of the CORE_RSTRET signal (CORE power domain) controlled by the Power, Reset and Clock Management module (see Chapter 4, *Power, Reset, and Clock Management*).
The CORE_RSTRET signal is activated during IC global power-on and global warm reset, and it resets the controller state machine and configuration registers.

11.1.3.2.3 Software Reset

GPMC modules can be reset under software control through the GPMC.GPMC_SYSCONFIG[1] SOFTRESET bit. When software reset bit is set, all registers and the finite state-machine (FSM) are reset immediately and unconditionally. The GPMC_SYSSTATUS[0] RESETDONE bit can be polled to check reset status.

11.1.3.2.4 Power Domain, Power Saving, and Reset Management

GPMC power is supplied by the CORE power domain, and GPMC power management complies with system power-management guidelines.

The GPMC reduces power consumption through auto-idle mode and the idle request/acknowledge process, both of which are configurable:

- Dynamic auto-idle (configurable through the GPMC.GPMC_SYSCONFIG[0] AUTOIDLE bit): To reduce power consumption, the GPMC internally disables the functional clock when no requests are pending and no accesses are ongoing.
- Idle request/acknowledge (one of three idle modes configurable through the GPMC.GPMC_SYSCONFIG[4:3] IDLEMODE field):
  - Force-idle: Immediately on receiving an idle request from the PRCM module, the GPMC sends an idle request/acknowledge to let the PRCM module correctly cut the GPMC source clock.
  - No-idle: The GPMC never goes to idle mode.
  - Smart-idle (strongly recommended): The GPMC goes to idle mode when all ongoing transactions are complete.

For detailed information about power management, see Chapter 4, *Power, Reset, and Clock Management*.

11.1.3.2.5 Hardware Requests

The GPMC uses two hardware requests as shown in Figure 11-4 :

- One interrupt request goes from GPMC (GPMC_IRQ) to the microprocessor unit (MPU) subsystem : M_IRQ_20.
- One DMA request goes from GPMC (GPMC_DMA_REQ) to the system DMA (sDMA) : S_DMA_3.

11.1.3.3 GPMC Address and Data Bus

The current application supports GPMC connection to address/data-multiplexed memory and a NAND device. Connection to a nonmultiplexed address/data memory is supported with an address range of only 2 Kbytes.

Depending on the GPMC configuration on each chip-select, address and data-bus lines that are not required for a particular access protocol are not updated (changed from current value) and are not sampled when input (input data bus).

The current application supports GPMC connection to address/data-multiplexed memory, address/data-nonmultiplexed memory with limited address (2 Kbytes), and a NAND device:

- When the GPMC.GPMC_CONFIG[1] LIMITEDADDRESS bit is set to 1, only gpmc_a[10:1] address lines are used. This limits the memory support to 2K-byte addressable memories.
- For address/data-multiplexed NOR devices, the address is multiplexed on the data bus.
- 8-bit wide NOR devices do not use GPMC I/O: gpmc_d[15:8] for data (they are used for address if needed).
- 16-bit wide NAND devices do not use GPMC I/O: gpmc_a[10:1].
- 8-bit wide NAND devices do not use GPMC I/O: gpmc_a[10:1] and GPMC I/O: gpmc_d[15:8].

CAUTION

Before trying to access a chip-select configured with a nonmultiplexed protocol, set the LIMITEDADDRESS bit control.

11.1.3.3.1 GPMC I/O Configuration Setting (in Default Pinout Mode 0)

Note: In this section, the i in GPMC_CONFIG1_i stands for the GPMC chip-select i where i = 0 to 7.

The address/data-nonmultiplexed device, which is limited to a 2K-byte address range, is selected by programming the following register fields:

- GPMC.GPMC_CONFIG1_i[11:10] DEVICETYPE field = 0x00
- GPMC.GPMC_CONFIG1_i[9] MUXADDDATA bit = 0
- GPMC.GPMC_CONFIG[1] LIMITEDADDRESS bit = 1

Note: The LIMITEDADDRESS field applies only to address/data-nonmultiplexed devices; it has no effect on other device types (address/data-multiplexed, NAND).

To select the address/data-multiplexed device, program the following register fields:

- GPMC.GPMC_CONFIG1_i[11:10] DEVICETYPE field = 0b00
- GPMC.GPMC_CONFIG1_i[9] MUXADDDATA bit = 1

To select the NAND device, program the following register field:

- GPMC.GPMC_CONFIG1_i[11:10] DEVICETYPE field = 0b10
- GPMC.GPMC_CONFIG1_i[9] MUXADDDATA bit = 0

11.1.3.3.2 GPMC CS0 Default Configuration at IC Reset

To ensure a correct external boot with a GPMC access from IC reset time on CS0, several external pins are sampled:

- The sys_boot[4:0] pins (OMAP device boundary) define the sequence of interfaces and devices to use for booting.
- The sys_boot[5] pin defines which group of booting sequences is preferred: memory booting (sys_boot[5] = 0) or peripheral booting (sys_boot[5] = 1).
- Three additional pins are used to configure reset values in the GPMC.GPMC_CONFIG1_i register (where i = 0):
  - The bootwaiten input pin (GPMC boundary) enables the monitoring on chip-select 0 of the WAIT pin at IC reset release time for read accesses. The input pin is used to configure the GPMC.GPMC_CONFIG1_i[22] WAITREADMONITORING bit (where i = 0). Its value comes from the BOOT_WAIT_ENABLE signal generated by the system control module (SCM). When sys_boot[5:0] = 0b111111, the BOOT_WAIT_ENABLE signal is activated, causing the wait pin to be monitored for read access.
  - The bootdevicesize input pin (GPMC boundary) defines the size of the attached device on chip-select 0 and is used to configure the GPMC.GPMC_CONFIG1_i[13:12] DEVICESIZE bits (where i = 0). A BOOT_DEVICE_SIZE signal is propagated from the SCM. Its value is fixed at 0x1 at IC reset, causing a 16-bit wide external memory to be used.
  - The cs0muxdevice input pin (GPMC boundary) selects whether the attached device to chip-select 0 is a multiplexed address and data device or not. The input pin is used to configure the GPMC.GPMC_CONFIG1_i[9] MUXADDDATA bit (where i = 0). A CS0_MUX_DEVICE signal is propagated from the SCM. Its value is fixed at 0x1 at IC reset, causing the attached device to be address/data-multiplexed.
  - The waitselectpin input pin selects the WAIT signal at IC reset release time between WAIT0 input pin or WAIT1 input pin. At IC reset release time, these two pins have different polarity.

CAUTION

Using the internal boot code, the entire CS0 configuration can be modified before the first CS0 access. This modification of internal boot code is necessary for two external devices:

- NAND device attached to CS0
- Nonmultiplexed 2-Kbyte address range device attached to CS0

At reset time, the IC may boot from the internal ROM or from the memory attached to the GPMC chip-select 0. This selection is made outside the GPMC.

Reset values of the timing control parameters are defined to cope with direct boot on address and data multiplexed NOR Flash device, on non-multiplexed NOR Flash device or on any asynchronous device with large timing margins assuming a low GPMC_FCLK frequency (for example, 19.2Mhz) at boot time.

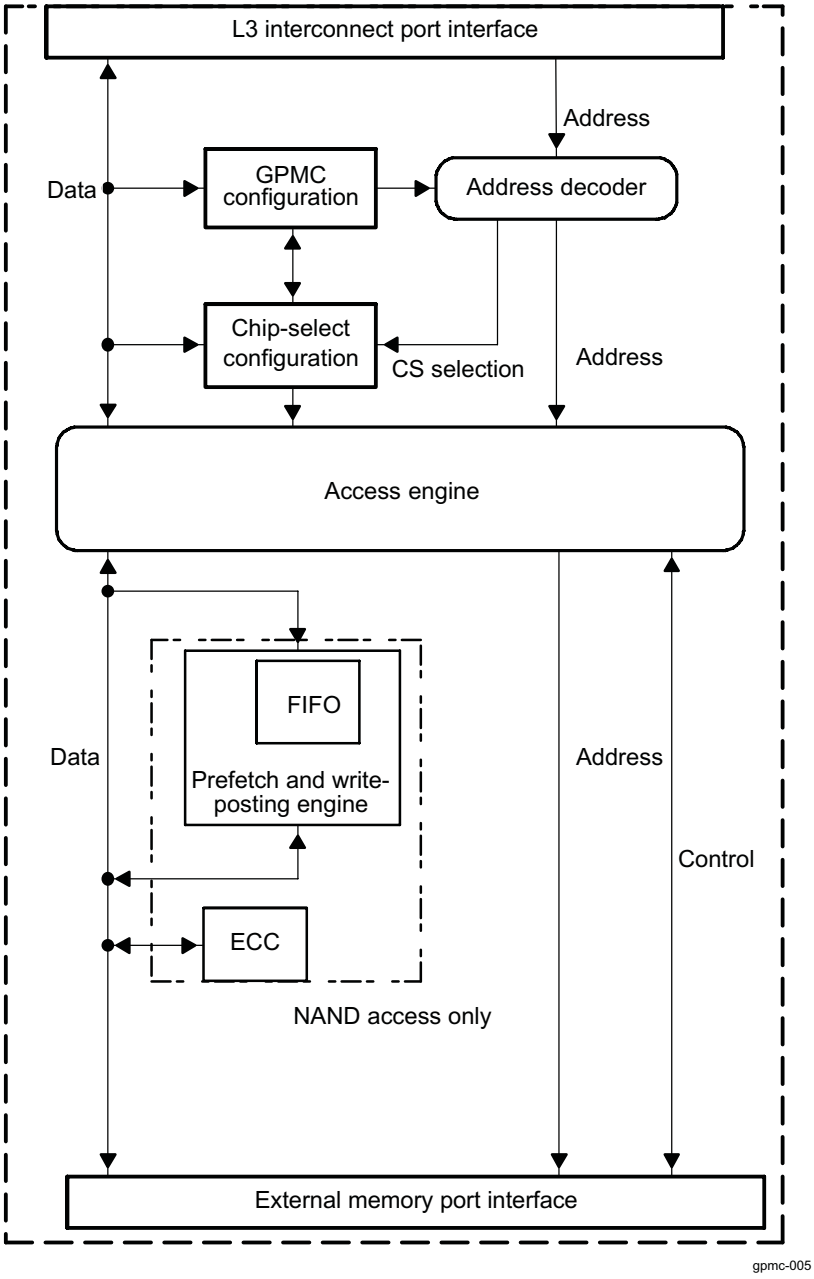
11.1.4 GPMC Functional Description

11.1.4.1 Description

As Figure 11-5 shows, the GPMC consists of six blocks:

- L3 interconnect port interface
- Address decoder, GPMC configuration, and chip-select configuration register file
- Access engine
- Prefetch and write-posting engine
- Error correction code engine (ECC)
- External device/memory port interface

Figure 11-5. GPMC Functional Diagram



The GPMC can access various external devices through the L3 Interconnect. The flexible programming model allows a wide range of attached device types and access schemes.

Based on the programmed configuration bit fields stored in the GPMC registers, the GPMC is able to generate all control signals timing depending on the attached device and access type.

Given the chip-select decoding and its associated configuration registers, the GPMC selects the appropriate device type control signals timing.

11.1.4.2 L3 Interconnect Interface

The GPMC L3 interconnect interface is a pipelined interface including an 8×32-bit word write buffer.

Any OMAP system host can issue external access requests through the GPMC.

The OMAP system can issue the following requests through this interface:

- One 8-bit/16-bit/32-bit interconnect access (read/write)
- Two incrementing 32-bit interconnect accesses (read/write)
- Two wrapped 32-bit interconnect accesses (read/write)
- Four incrementing 32-bit interconnect accesses (read/write)
- Four wrapped 32-bit interconnect accesses (read/write)
- Eight incrementing 32-bit interconnect accesses (read/write)
- Eight wrapped 32-bit interconnect accesses (read/write)

Only linear burst transactions are supported; interleaved burst transactions are not supported. Only power-of-two-length precise bursts 2×32, 4×32, or 8×32 with the burst base address aligned on the total burst size are supported (this limitation applies to incrementing bursts only).

This interface also provides one interrupt and one DMA request line, for specific event control.

It is recommended to program the ATTACHEDDEVICEPAGELENGTH field (GPMC_CONFIG1_i[24:23]) according to the effective attached device page length and to enable WRAPBURST bit (GPMC_CONFIG1_i[31]) if the attached device supports wrapping burst.

However, it is possible to emulate wrapping burst on a non-wrapping memory by providing relevant addresses within the page or splitting transactions. Bursts larger than the memory page length are chopped into multiple bursts transactions. Due to the alignment requirements, a page boundary is never crossed.

11.1.4.3 Address Decoder, GPMC Configuration, and Chip-Select Configuration Register File

Address-decoding logic selects for chip-selects according to the address request and the content of the chip-select base address register file, which includes a set of global GPMC configuration registers and eight sets of chip-select configuration registers.

The GPMC configuration register file is memory-mapped and can be read or written with byte, 16-bit word, or 32-bit word accesses. The register file should be configured as a noncacheable, nonbufferable region to prevent any desynchronization between host execution (write request) and the completion of register configuration (write completed with register updated). Section 11.1.7 of this chapter provides the GPMC register locations. For the map of GPMC memory locations, see Chapter 2, *Memory Mapping*.

After the chip-select is configured, the access engine accesses the external device, drives the external interface control signals, and applies the interface protocol based on user-defined timing parameters and settings.

11.1.4.4 Error Correction Code Engine

The GPMC includes an error correction code (ECC) calculation engine that allows ECC calculation during data read or data program (write) operations.Two ECC algorithms are available depending on GPMC_ECC_CONFIG[16] ECCALGORITHM settings: Hamming code or BCH code (Bose-ChaudhurI-Hocquenghem).

The GPMC does not directly handle the error code correction itself. During writes, the GPMC computes parity bits. During reads, the GPMC provides enough information for the processor to correct errors without reading the data buffer all over again.

The Hamming code ECC is based on a 2-dimensional (row and column) bit parity accumulation. This parity accumulation is either accomplished on the programmed number of bytes or Word16s read from the memory device, or written to the memory device in stream mode.

Because the ECC engine includes only one accumulation context, it can be allocated to only one chip-select at a time through the GPMC.GPMC_ECC_CONFIG[3:1] ECCCS bit field.

See Section 11.1.5.14.3 for more information on ECC calculation.

11.1.4.5 Prefetch and Write-Posting Engine

The prefetch and write-posting engine is a simplified embedded-access requester that presents requests to the access engine on a user-defined chip-select target. The access engine interleaves these requests with any request coming from the L3 interface; as a default the prefetch and write-posting engine has the lowest priority.

The prefetch and write-posting engine is dedicated to data-stream access (as opposed to random data access); thus, it is primarily dedicated to NAND support. The engine does not include an address generator; the request is limited to chip-select target identification. It includes a 64-byte FIFO associated with a DMA request synchronization line, for optimal DMA-based use.

For more information about prefetch and write-posting engine programming, see Section 11.1.5.14.4, *Prefetch and Write-Posting Engine*.

11.1.4.6 External Device/Memory Port Interface

The external port interface controls all address, data, and control signals required for communication with GPMC-supported devices and memories.

11.1.5 GPMC Basic Programming Model

The GPMC basic programming model offers maximum flexibility to support various access protocols for each of the eight configurable chip-selects. Use optimal chip-select settings, based on the characteristics of the external device:

- Different protocols can be selected to support generic asynchronous or synchronous random-access devices (NOR flash, SRAM) or to support specific NAND devices.
- The address and the data bus can be multiplexed on the same external bus.
- Read and write access can be independently defined as asynchronous or synchronous.
- System requests (byte, Word16, burst) are performed through single or multiple accesses. External access profiles (single, multiple with optimized burst length, native- or emulated-wrap) are based on external device characteristics (supported protocol, bus width, data buffer size, native-wrap support).
- System burst read or write requests are synchronous-burst (multiple-read or multiple-write). When neither burst nor page mode is supported by external memory or ASIC devices, system burst read or write requests are translated to successive single synchronous or asynchronous accesses (single reads or single writes). 8-bit wide devices are supported only in single-synchronous or asynchronous read or write mode.
- To simulate a programmable internal-wait state, an external wait pin can be monitored to dynamically control external access at the beginning (initial access time) of and during a burst access.

Each control signal is controlled independently for each chip-select. The internal functional clock of the GPMC (GPMC_FCLK) is used as a time reference to specify the following:

- Read- and write-access duration
- Most GPMC external interface control-signal assertion and deassertion times
- Data-capture time during read access
- External wait-pin monitoring time
- Duration of idle time between accesses, when required

11.1.5.1 Chip-Select Base Address and Region Size Configuration

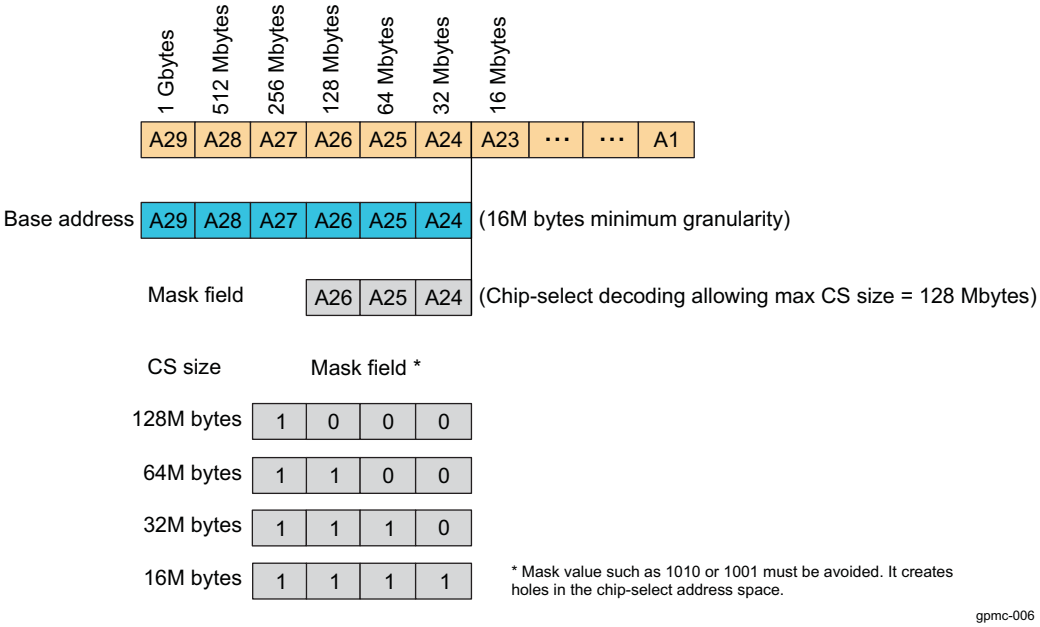
Any external memory or ASIC device attached to the GPMC external interface can be accessed by any OMAP system host within the GPMC 1-GB contiguous address space. For details, see Chapter 2, *Memory Mapping*.

The GPMC 1-GB address space can be divided into a maximum of eight chip-select regions with programmable base address and programmable CS size. The CS size is programmable from 16MB to 128MB (must be a power-of-2) and is defined by the mask field. Attached memory smaller than the programmed CS region size is accessed through the entire CS region (aliasing).

Each chip-select has a 6-bit base address encoding and a 4-bit decoding mask, which must be programmed according to the following rules:

- The programmed chip-select region base address must be aligned on the chip-select region size address boundary and is limited to a power-of-2 address value. During access decoding, the register base address value is used for address comparison with the address-bit line mapping as described in Figure 11-6 (with A0 as the OMAP system byte-address line). Base address is programmed through the GPMC_CONFIG7_i[5:0] BASEADDRESS bit field
- The register mask is used to exclude some address lines from the decoding. A register mask bit field set to 0 suppresses the associated address line from the address comparison (incoming address bit line is don't care). The register mask value must be limited to the subsequent value, based on the desired chip-select region size. Any other value has an undefined result. When multiple chip-select regions with overlapping addresses are enabled concurrently, access to these chip-select regions is cancelled and a GPMC access error is posted. The mask field is programmed through the GPMC_CONFIG7_i[11:8] MASKADDRESS bit field.

Figure 11-6. Chip-Select Address Mapping and Decoding Mask



Chip-select configuration (base and mask address or any protocol and timing settings) must be performed while the associated chip-select is disabled through the GPMC.GPMC_CONFIG7_i[6] CSVALID bit (where i stands for the GPMC chip-select value, i = 0 to 7). In addition, a chip-select configuration can be disabled only if there is no ongoing access to that chip-select. This requires activity monitoring of the prefetch or write-posting engine if the engine is active on the chip-select. Also, the write buffer state must be monitored to wait for any posted write completion to the chip-select.

Conversely, before trying to access a chip-select, software must ensure that the chip-select is enabled. To account for prefetch engine effects, after the chip-select-enable instruction, an NOP instruction (equivalent to 64 bits) must be executed before the chip-select is accessed.

Any access attempted to a nonvalid GPMC address region (CSVALID disabled or address decoding outside a valid chip-select region) is not propagated to the external interface and a GPMC access error is posted. In case of chip-selects overlapping, an error is generated and no access will occur on either chip-select.

Chip-select 0 is the only chip-select region enabled after either a power-up or a GPMC reset.

CAUTION

Although the GPMC interface can drive up to 8 chip-selects, the frequency specified for this interface is for a specific load. If this load is exceeded, the maximum frequency cannot be reached. One solution is to implement a board with buffers, to allow the slowest device to maintain the total load on the lines.

11.1.5.2 Access Protocol Configuration

11.1.5.2.1 Supported Devices

The access protocol of each chip-select can be independently specified through the GPMC.GPMC_CONFIG1_i[11:10] DEVICETYPE parameter (where i =0 to 7) for:

- Random-access synchronous or asynchronous memory like NOR flash, SRAM
- NAND flash asynchronous devices

Note: NAND flash interfacing requires the parameter settings of generic chip-select 0. For more information about the NAND flash GPMC basic programming model and NAND support, see Section 11.1.5.14, *NAND Device Basic Programming Model*, and Section 11.1.5.14.1, *NAND Memory Device in Byte or Word 16 Stream Mode*.

11.1.5.2.2 Access Size Adaptation and Device Width

Each chip-select can be independently configured through the GPMC.GPMC_CONFIG1_i[13:12] DEVICESIZE field (i = 0 to 7) to interface with a 16-bit wide device or an 8-bit wide device. System requests with data width greater than the external device data bus width are split into successive accesses according to both the external device data-bus width and little-endian data organization.

Note: The OMAP device does not provide the A0 byte address line required for random-byte addressable 8-bit wide device interfacing (for both multiplexed and nonmultiplexed protocol). It limits the use of 8-bit wide device interfacing to byte-alias accesses. This limitation is not applicable to NAND device interfacing (8-bit wide or 16-bit wide devices).

11.1.5.2.3 Address/Data-Multiplexing Interface

For random synchronous or asynchronous memory interfacing (DEVICETYPE = 0b00), an address- and data-multiplexing protocol can be selected through the GPMC.GPMC_CONFIG1_i[9] MUXADDDATA bit (i = 0 to 7). The nADV signal must be used as the external device address latch control signal. For the associated chip-select configuration, nADV assertion and deassertion time and nOE assertion time must be set to the appropriate value to meet the address latch setup/hold time requirements of the external device. See Section 11.1.3, *GPMC Integration*.

Note: This address/data-multiplexing interface is not applicable to NAND device interfacing. NAND devices require a specific address, command, and data multiplexing protocol. See Section 11.1.5.14, *NAND Device Basic Programming Model*.

11.1.5.2.4 Address and Data Bus

See Section 11.1.3.3, *GPMC Address and Data Bus*.

11.1.5.2.5 Asynchronous and Synchronous Access

For each chip-select configuration, the read access can be specified as either asynchronous or synchronous access through the GPMC.GPMC_CONFIG1_i[29] READTYPE bit (i = 0 to 7). For each chip-select configuration, the write access can be specified as either synchronous or asynchronous access through the GPMC.GPMC_CONFIG1_i[27] WRITETYPE bit (i = 0 to 7).

Asynchronous and synchronous read (write) access time and related control signals are controlled through timing parameters that refer to GPMC_FCLK. The primary difference of synchronous mode is the availability of a configurable clock interface (GPMC_CLK) to control the external device. Synchronous mode also affects data-capture and wait-pin monitoring schemes in read access.

For details about asynchronous and synchronous access, see the descriptions of GPMC_CLK, RdAccessTime, WrAccessTime, and wait-pin monitoring.

For more information about timing-parameter settings, see the sample timing diagrams in this chapter.

Note: The address bus and nBE[1:0] are fixed for the duration of a synchronous burst read access, but they are updated for each beat of an asynchronous page-read access.

11.1.5.2.6 Page and Burst Support

Each chip-select can be configured to process system single or burst requests into successive single accesses or asynchronous page/synchronous burst accesses, with appropriate access size adaptation.

Depending on the external device page or burst capability, read and write accesses can be independently configured through the GPMC. The GPMC_CONFIG1_i[30] READMULTIPLE and GPMC.GPMC_CONFIG1_i[28] WRITMULTIPLE bits (i = 0 to 7) are associated with the READTYPE and WRITETYPE parameters.

Notes:

- Asynchronous write page mode is not supported.
- 8-bit wide device support is limited to nonburstable devices (READMULTIPLE and WRITEMULTIPLE are don't care).
- Not applicable to NAND device interfacing.

11.1.5.2.7 System Burst Versus External Device Burst Support

The OMAP system can issue the following requests to the GPMC:

- Byte, Word16, Word32 requests (byte enable controlled). This is always a single request from the interconnect point of view.
- Incrementing fixed-length bursts of two words, four words, and eight words
- Wrapped (critical word access first) fixed-length burst of two, four, or eight words

To process a system request with the optimal protocol, the READMULTIPLE (and READTYPE) and WRITEMULTIPLE (and WRITETYPE) parameters must be set according to the burstable capability (synchronous or asynchronous) of the attached device.

The GPMC access engine issues only fixed-length burst. The maximum length that can be issued is defined per CS by the GPMC.GPMC_CONFIG1_i[24:23] ATTACHEDDEVICEPAGELENGTH field (i = 0 to 7). When the ATTACHEDDEVICEPAGELENGTH value is less than the system burst request length (including the appropriate access size adaptation according to the device width), the GPMC splits the system burst request into multiple burst beats. Within the specified 4-, 8-, or 16-word value, the ATTACHEDDEVICEPAGELENGTH field value must correspond to the maximum-length burst supported by the memory device configured in fixed-length burst mode (as opposed to continuous burst mode).

To get optimal performance from memory devices that natively support 16 Word16-length-wrapping burst capability (critical word access first), the ATTACHEDDEVICEPAGELENGTH parameter must be set to 16 words and the GPMC.GPMC_CONFIG1_i[31] WRAPBURST bit (i = 0 to 7) must be set to 1. Similarly DEVICEPAGELENGTH is set to 4 and 8 for memories supporting respectively 4 and 8 Word16-length-wrapping burst.

When the memory device does not offer (or is not configured to offer) native 16 Word16-length-wrapping burst, the WRAPBURST parameter must be cleared, and the GPMC access engine emulates the wrapping burst by issuing the appropriate burst sequences according to the ATTACHEDDEVICEPAGELENGTH value.

When the memory device does not support native-wrapping burst, there is usually no difference in behavior between a fixed burst length mode and a continuous burst mode configuration (except for a potential power increase from a memory-speculative data prefetch in a continuous burst read). However, even though continuous burst mode is compatible with GPCM behavior, because the GPMC access engine issues only fixed-length burst and does not benefit from continuous burst mode, it is best to configure the memory device in fixed-length burst mode.

The memory device maximum-length burst (configured in fixed-length burst wrap or nonwrap mode) usually corresponds to the memory device data buffer size. Memory devices with a minimum of 16 half-word buffers are the most appropriate (especially with wrap support), but memory devices with smaller buffer size (4 or 8) are also supported, assuming that the GPMC.GPMC_CONFIG1_i[24:23] ATTACHEDDEVICEPAGELENGTH field is set accordingly to 4 or 8 words.

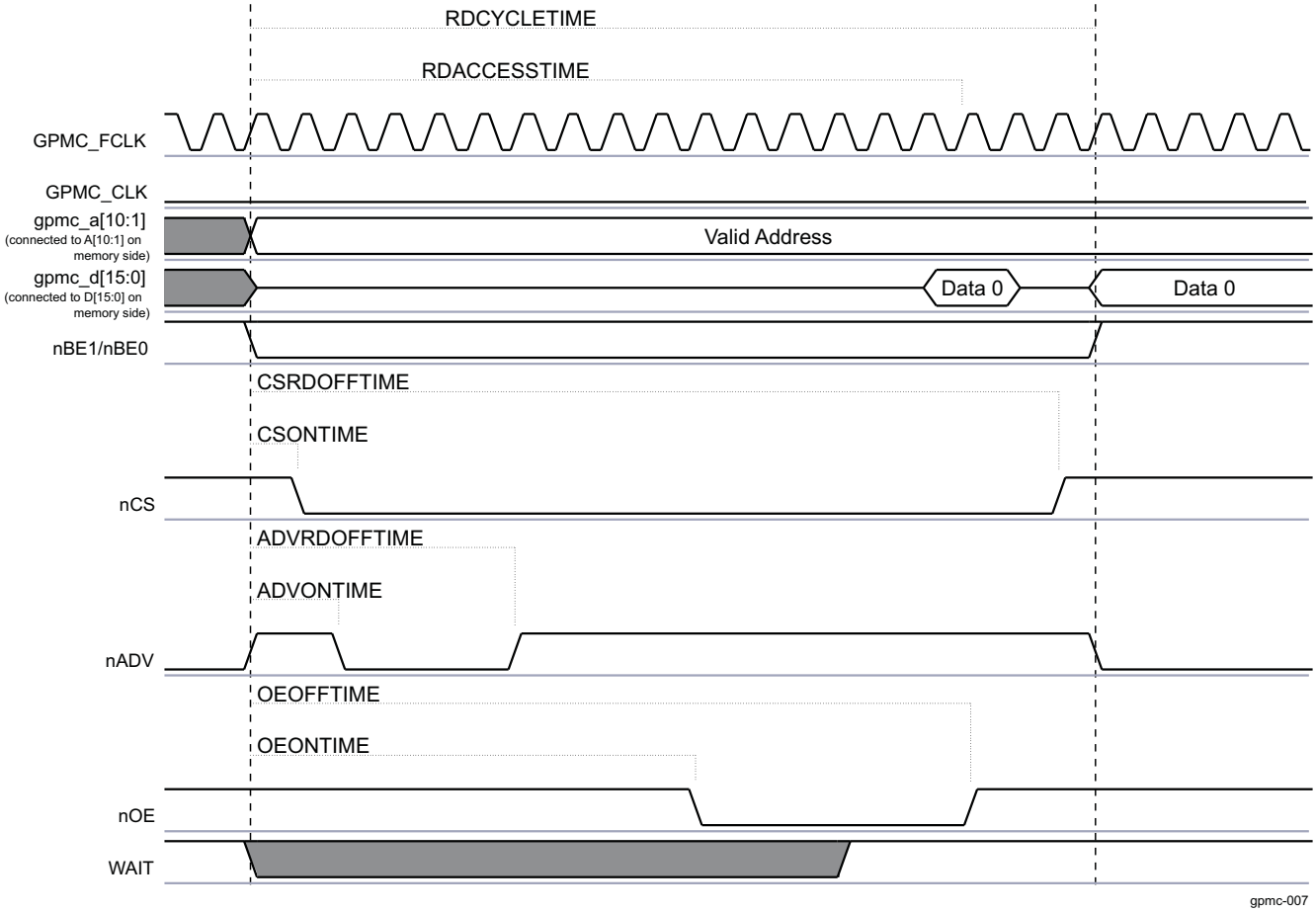
The OMAP system issues only requests with addresses or starting addresses for nonwrapping burst requests; that is, the request size boundary is aligned. In case of an eight-word-wrapping burst, the wrapping address always occurs on the eight-words boundary. As a consequence, all words requested must be available from the memory data buffer when the buffer size is equal to or greater than the ATTACHEDDEVICEPAGELENGTH value. This usually means that data can be read from or written to the buffer at a constant rate (number of cycles between data) without wait states between data accesses. If the memory does not behave this way (nonzero wait state burstable memory), wait-pin monitoring must be enabled to dynamically control data-access completion within the burst beat.

Note: When the system burst request length is less than the ATTACHEDDEVICEPAGELENGTH value, the GPMC proceeds with the required accesses.

11.1.5.3 Timing Setting

The GPMC is a signal generator that offers the maximum flexibility to support various access protocols. Most of the timing parameters of the protocol access used by the GPMC to communicate with attached memories or devices are programmable on a chip-select basis. Assertion and deassertion times of control signals are defined to match the attached memory or device timing specifications and to get maximum performance during accesses. For example, the timing diagram in Figure 11-7 shows an asynchronous single-read access performed on an asynchronous device. For more information about GPMC_CLK and GPMC_FCLK, see Section 11.1.5.3.6.

Figure 11-7. Asynchronous Single Read on a Nonmultiplexed Address/Data Device



11.1.5.3.1 Read Cycle Time and Write Cycle Time (RDCYCLETIME/WRCYCLETIME)

The GPMC.GPMC_CONFIG5_i[4:0] RDCYCLETIME and GPMC.GPMC_CONFIG5_i[12:8] WRCYCLETIME fields (i = 0 to 7) define the address bus and byte enables valid times for read and write accesses. To ensure a correct duty cycle of GPMC_CLK between accesses, RDCYCLETIME and WRCYCLETIME are expressed in GPMC_FCLK cycles and must be multiples of the GPMC_CLK cycle.

When either RDCYCLETIME or WRCYCLETIME completes, if they are not already deasserted, all control signals (NCS, nADV/ALE, nOE/RE, nWE, and BE0/CLE) are deasserted to their reset values, regardless of their deassertion time parameters.

An exception to this forced deassertion occurs when a pipelined request to the same chip-select or to a different chip-select is pending. In such a case, it is not necessary to deassert a control signal with deassertion time parameters equal to the cycle-time parameter. This exception to forced deassertion prevents any unnecessary glitchy transition. This requirement also applies to BE signals, thus avoiding an unnecessary BE glitch transition when pipelining requests.

If no inactive cycles are required between successive accesses to the same or to a different chip-select (GPMC.GPMC_CONFIG6_i[7] CYCLE2CYCLESAMECSEN = 0 or GPMC.GPMC_CONFIG6_i[6] CYCLE2CYCLEDIFFCSEN = 0, where i = 0 to 7), and if assertion-time parameters associated with the pipelined access are equal to 0, asserted control signals (nCS, nADV/ALE, nBE0/CLE, nWE, and nOE/RE) are kept asserted. This applies to any read/write to read/write access combination.

If inactive cycles are inserted between successive accesses, that is, CYCLE2CYCLESAMECSEN = 1 or CYCLE2CYCLEDIFFCSEN = 1, the control signals are forced to their respective default reset values for the number of GPMC_FCLK cycles defined in CYCLE2CYCLEDELAY:

- The RDCYCLETIME and WRCYCLETIME bit fields are programmable in the GPMC.GPMC_CONFIG5_i register, i = 0 to 7.
- The RDCYCLETIME and WRCYCLETIME bit fields can be set from 0 to 31 GPMC_FCLK cycles with a granularity of 1 for GPMC.GPMC_CONFIG1_i[4] TIMEPARAGRANULARITY set to 0.
- The RDCYCLETIME and WRCYCLETIME bit fields can be set from 0 to 62 GPMC._FCLK cycles with a granularity of 2 for GPMC.GPMC_CONFIG1_i[4] TIMEPARAGRANULARITY set to 1.

11.1.5.3.2 nCS: Chip-Select Signal Control Assertion/Deassertion Time (CSONTIME/CSRDOFFTIME/CSWROFFTIME/CSEXTRADELAY)

The GPMC.GPMC_CONFIG2_i[3:0] CSONTIME field (where i = 0 to 7) defines the nCS signal-assertion time relative to the start access time. It is common for read and write accesses.

For a read access, the GPMC.GPMC_CONFIG2_i[12:8] CSRDOFFTIME field defines the nCS signal deassertion time relative to start access time.

For a write access, the GPMC.GPMC_CONFIG2_i[20:16] CSWROFFTIME field defines the nCS signal deassertion time relative to start access time.\

CSONTIME, CSRDOFFTIME and CSWROFFTIME parameters are applicable to synchronous and asynchronous modes. CSONTIME can be used to control an address and byte enable setup time before chip-select assertion. CSRDOFFTIME and CSWROFFTIME can be used to control an address and byte enable hold time after chip-select deassertion.

nCS signal transitions as controlled through CSONTIME, CSRDOFFTIME, and CSWROFFTIME can be delayed by half a GPMC_FCLK period by enabling the GPMC.GPMC_CONFIG2_i[7] CSEXTRADELAY bit. This half of a GPMC_FCLK period provides more granularity on the nCS assertion and deassertion time to guarantee proper setup and hold time relative to GPMC_CLK. CSEXTRADELAY is especially useful in configurations where GPMC_CLK and GPMC_FCLK have the same frequency, but can be used for all GPMC configurations. If asserted, CSEXTRADELAY applies to all parameters controlling nCS transitions.

The CSEXTRADELAY bit must be used carefully to avoid control-signal overlap between successive accesses to different chip-selects. This implies the need to program the RDCYCLETIME and WRCYCLETIME bit fields to be greater than the nCS signal-deassertion time, including the extra half-GPMC_FCLK-period delay.

11.1.5.3.3 nADV/ALE: Address Valid/Address Latch Enable Signal Control Assertion/Deassertion Time (ADVONTIME/ADVRDOFFTIME/ADVWROFFTIME/ADVEXTRADELAY)

The GPMC.GPMC_CONFIG3_i[3:0] ADVONTIME field (where i = 0 to 7) defines the nADV/ALE signal-assertion time relative to start access time. It is common to read and write accesses.

For a read access, the GPMC.GPMC_CONFIG3_i[12:8] ADVRDOFFTIME field defines the nADV/ALE signal-deassertion time relative to start access time.

For a write access, the GPMC.GPMC_CONFIG3_i[20:16] ADVWROFFTIME field defines the nADV/ALE signal-deassertion time relative to start access time.

ADVONTIME can be used to control an address and byte enable valid setup time control before nADV/ALE assertion. ADVRDOFFTIME and ADVWROFFTIME can be used to control an address and byte enable valid hold time control after nADV/ALE de-assertion. ADVRDOFFTIME and ADVWROFFTIME are applicable to both synchronous and asynchronous modes.

nADV/ALE signal transitions as controlled through ADVONTIME, ADVRDOFFTIME, and ADVWROFFTIME can be delayed by half a GPMC_FCLK period by enabling the GPMC.GPMC_CONFIG3_i[7] ADVEXTRADELAY bit. This half of a GPMC_FCLK period provides more granularity on nADV/ALE assertion and deassertion time to guarantee proper setup and hold time relative to GPMC_CLK. The ADVEXTRADELAY configuration parameter is especially useful in configurations where GPMC_CLK and GPMC_FCLK have the same frequency, but can be used for all GPMC configurations. If asserted, ADVEXTRADELAY applies to all parameters controlling nADV/ALE transitions.

ADVEXTRADELAY must be used carefully to avoid control-signal overlap between successive accesses to different chip-selects. This implies the need to program the RDCYCLETIME and WRCYCLETIME bit fields to be greater than nADV/ALE signal-deassertion time, including the extra half-GPMC_FCLK-period delay.

See Section 11.1.5.14 for more details about ADVONTIME, ADVRDOFFTIME, and ADVWROFFTIME use for command (CLE) and address latch enable (ALE) use for a NAND flash interface.

11.1.5.3.4 nOE/nRE: Output Enable/Read Enable Signal Control Assertion/Deassertion Time (OEONTIME/OEOFFTIME/OEEXTRADELAY)

The GPMC.GPMC_CONFIG4_i[3:0] OEONTIME field (where i = 0 to 7) defines the nOE/nRE signal assertion time relative to start access time. It is applicable only to read accesses.

The GPMC.GPMC_CONFIG4_i[12:8] OEOFFTIME field defines the nOE/nRE signal deassertion time relative to start access time. It is applicable only to read accesses.

OEONTIME and OEOFFTIME parameters are applicable to synchronous and asynchronous modes. OEONTIME can be used to control an address and byte enable valid setup time control before nOE/nRE assertion. OEOFFTIME can be used to control an address and byte enable valid hold time control after nOE/nRE assertion.

The nOE/RE signal transitions as controlled through OEONTIME, and OEOFFTIME can be delayed by half a GPMC_FCLK period by enabling the GPMC.GPMC_CONFIG4_i[7] OEEXTRADELAY bit. This half of a GPMC_FCLK period provides more granularity on nOE/RE assertion and deassertion time to guaranty proper setup and hold time relative to GPMC_CLK. If asserted, OEEXTRADELAY applies to all parameters controlling nOE/nRE transitions.

OEEXTRADELAY must be used carefully, to avoid control-signal overlap between successive accesses to different chip-selects. This implies the need to program RDCYCLETIME and WRCYCLETIME to be greater than nOE/RE signal-deassertion time, including the extra half-GPMC_FCLK-period delay.

nOE/nRE is not asserted during a write cycle.

Note: When the GPMC generates a read access to an address-/data-multiplexed device, it drives the address bus until nOE assertion time.

11.1.5.3.5 nWE: Write Enable Signal Control Assertion/Deassertion Time (WEONTIME/WEOFFTIME/WEEXTRADELAY)

The GPMC.GPMC_CONFIG4_i[19:16] WEONTIME field (where i = 0 to 7) defines the nWE signal-assertion time relative to start access time. It applies only to write accesses.

The GPMC.GPMC_CONFIG4_i[28:24] WEOFFTIME field defines the nWE signal-deassertion time relative to start access time. It applies only to write accesses.

WEONTIME can be used to control an address and byte enable valid setup time control before nWE assertion. WEOFFTIME can be used to control an address and byte enable valid hold time control after nWE assertion.

nWE signal transitions as controlled through WEONTIME, and WEOFFTIME can be delayed by half a GPMC_FCLK period by enabling the GPMC.GPMC_CONFIG4_i[23] WEEXTRADELAY bit. This half of a GPMC_FCLK period provides more granularity on nWE assertion and deassertion time to guaranty proper setup and hold time relative to GPMC_CLK. If asserted, WEEXTRADELAY applies to all parameters controlling nWE transitions.

The WEEXTRADELAY bit must be used carefully to avoid control-signal overlap between successive accesses to different chip-selects. This implies the need to program the WRCYCLETIME bit field to be greater than the nWE signal-deassertion time, including the extra half-GPMC_FCLK-period delay.

nWE is not asserted during a read cycle.

11.1.5.3.6 GPMC_CLK

GPMC_CLK is the external clock provided to the attached synchronous memory or device.

- The GPMC_CLK clock frequency is the GPMC_FCLK functional clock frequency divided by 1, 2, 3, or 4, depending on the GPMC.GPMC_CONFIG1_i[1:0] GPMCFCLKDIVIDER bit field (where i = 0 to 7), with a guaranteed 50-percent duty cycle.
- The GPMC_CLK clock is only activated when the access in progress is defined as synchronous (read or write access).
- The GPMC.GPMC_CONFIG1_i[26:25] CLKACTIVATIONTIME field (i = 0 to 7) defines the number of GPMC_FCLK cycles from start access time to GPMC_CLK activation.
- The GPMC_CLK clock is stopped when cycle time completes and is asserted low between accesses.
- The GPMC_CLK clock is kept low when access is defined as asynchronous.
- When cycle time completes, the GPMC_CLK may be high because of the GPMCFCLKDIVIDER bit field. To ensure correct stoppage of the GPMC_CLK clock within the 50-percent required duty cycle, it is the user's responsibility to extend the RDCYCLETIME or WRCYCLETIME value.
- When the GPMC is configured for synchronous mode, the GPMC_CLK signal (which is an output) must also be set as an input (CONTROL.CONTROL_PADCONF_GPMC_NCS7[24] INPUTENABLE1 = 1). GPMC_CLK is looped back through the output and input buffers of the corresponding GPMC_CLK pad at the OMAP boundary. The looped-back clock is used to synchronize the sampling of the memory signals.

Note: To ensure a correct external clock cycle, the following rules must be applied:

- (RDCYCLETIME CLKACTIVATIONTIME) must be a multiple of (GPMCFCLKDIVIDER + 1).
- The PAGEBURSTACCESSTIME value must be a multiple of (GPMCFCLKDIVIDER + 1).

11.1.5.3.7 GPMC_CLK and Control Signals Setup and Hold

Control-signal transition (assertion and deassertion) setup and hold values with respect to the GPMC_CLK edge can be controlled in the following ways:

- For the GPMC_CLK signal, the GPMC.GPMC_CONFIG1_i[26:25] CLKACTIVATIONTIME field (i = 0 to 7) allows setup and hold control of control-signal assertion time.
- The use of a divided GPMC_CLK allows setup and hold control of control-signal assertion and deassertion times.
- When GPMC_CLK runs at the GPMC_FCLK frequency so that GPMC_CLK edge and control-signal transitions refer to the same GPMC_FCLK edge, the control-signal transitions can be delayed by half of a GPMC_FCLK period to provide minimum setup and hold times. This half-GPMC_FCLK delay is enabled with the CSEXTRADELAY, ADVEXTRADELAY, OEEXTRADELAY, or WEEXTRADELAY parameter. This delay must be used carefully to prevent control-signal overlap between successive accesses to different chip-selects. This implies that the RDCYCLETIME and WRCYCLETIME are greater than the last control-signal deassertion time, including the extra half-GPMC_FCLK cycle.

11.1.5.3.8 Access Time (RDACCESSTIME / WRACCESSTIME)

The read access time and write access time durations can be programmed independently allowing nOE and GPMC data capture timing parameters to be independent of nWE and memory device data capture timing parameters.
RDACCESSTIME is programmed in the GPMC.GPMC_CONFIG5_i[20:16] bit field (i = 0 to 7).
WRACCESSTIME is programmed in the GPMC.GPMC_CONFIG6_i[28:24] bit field (i = 0 to 7).
RDACCESSTIME and WRACCESSTIME can be set from 0 to 31 GPMC_FCLK cycles with a granularity of one (GPMC_CONFIG1_i[4] TIMEPARAGRANULARITY = 0).
RDACCESSTIME and WRACCESSTIME can be set from 0 to 62 GPMC_FCLK cycles with a granularity of two (GPMC_CONFIG1_i[4] TIMEPARAGRANULARITY = 1).

11.1.5.3.8.1 Access Time on Read Access

In asynchronous read mode, for single and paged accesses, RDACCESSTIME field (i = 0 to 7) defines the number of GPMC_FCLK cycles from start access time to the GPMC_FCLK rising edge used for the first data capture. RDACCESSTIME must be programmed to the rounded greater GPMC_FCLK cycle value of the read access time of the attached memory device.

In synchronous read mode, for single or burst accesses, RDACCESSTIME defines the number of GPMC_FCLK cycles from start access time to the GPMC_FCLK rising edge corresponding to the GPMC_CLK rising edge used for the first data capture.

GPMC_CLK which is sent to the memory device for synchronization with the GPMC controller, is internally retimed to correctly latch the returned data. RDCYCLETIME must be greater than RDACCESSTIME in order to let the GPMC latch the last return data using the internally retimed GPMC_CLK.

The external WAIT signal can be used in conjunction with RDACCESSTIME to control the effective GPMC data-capture GPMC_FCLK edge on read access in both asynchronous mode and synchronous mode. For details about wait monitoring, see Section 11.1.5.4.

11.1.5.3.8.2 Access Time on Write Access

In asynchronous write mode, the GPMC_CONFIG6_i[28:24] WRACCESSTIME timing parameter is not used to define the effective write access time. Instead, it is used as a WAIT invalid timing window, and must be set to a correct value so that the gpmc_wait pin is at a valid state two GPMC_CLK cycles before WRACCESSTIME completes. For details about wait monitoring, see Section 11.1.5.4.

In synchronous write mode , for single or burst accesses, WRACCESSTIME defines the number of GPMC_FCLK cycles from start access time to the GPMC_CLK rising edge used by the memory device for the first data capture.

The external WAIT signal can be used in conjunction with WRACCESSTIME to control the effective memory device data capture GPMC_CLK edge for a synchronous write access. For details about wait monitoring, see Section 11.1.5.4.

11.1.5.3.9 Page Burst Access Time (PAGEBURSTACCESSTIME)

PAGEBURSTACCESSTIME is programmed in the GPMC.GPMC_CONFIG5_i[27:24] bit field (i = 0 to 7). PAGEBURSTACCESSTIME can be set from 0 to 15 GPMC_FCLK cycles with a granularity of one (GPMC.GPMC_CONFIG1_i[4] TIMEPARAGRANULARITY set to 0), or from 0 to 30 GPMC_FCLK cycles with a granularity of two (TIMEPARAGRANULARITY set to 1).

11.1.5.3.9.1 Page Burst Access Time on Read Access

In asynchronous page read mode, the delay between successive word captures in a page is controlled through the PAGEBURSTACCESSTIME bit field. The PAGEBURSTACCESSTIME parameter must be programmed to the rounded greater GPMC_FCLK cycle value of the read access time of the attached device.

In synchronous burst read mode, the delay between successive word captures in a burst is controlled through the PAGEBURSTACCESSTIME field.

The external WAIT signal can be used in conjunction with PAGEBURSTACCESSTIME to control the effective GPMC data capture GPMC_FCLK edge on read access. For details about wait monitoring, see Section 11.1.5.4.

11.1.5.3.9.2 Page Burst Access Time on Write Access

Asynchronous page write mode is not supported. PAGEBURSTACCESSTIME is irrelevant in this case.

In synchronous burst write mode, PAGEBURSTACCESSTIME controls the delay between successive memory device word captures in a burst.

The external WAIT signal can be used in conjunction with PAGEBURSTACCESSTIME to control the effective memory-device data capture GPMC_CLK edge in synchronous write mode. For details about wait monitoring, see Section 11.1.5.4.

11.1.5.3.10 Bus Keeping Support

At the end-cycle time of a read access, if no other access is pending, the GPMC drives the bus with the last data read after RDCYCLETIME completion time to prevent bus floating and reduce power consumption.

After a write access, if no other access is pending, the GPMC keeps driving the data bus after WRCYCLETIME completes with the same data to prevent bus floating and power consumption.

11.1.5.4 WAIT Pin Monitoring Control

GPMC access time can be dynamically controlled using an external gpmc_wait pin when the external device access time is not deterministic and cannot be defined and controlled only using the GPMC internal RDACCESSTIME, WRACCESSTIME and PAGEBURSTACCESSTIME wait state generator.

The GPMC four input wait pins: gpmc_wait3, gpmc_wait2, gpmc_wait1, and gpmc_wait0. These four pins allow direct plugin and control of external devices with different wait-pin polarity. They also allow the overlap of wait-pin assertion from different devices without affecting access to devices for which the wait pin is not asserted.

- The GPMC.GPMC_CONFIG1_i[17:16] WAITPINSELECT field (i = 0 to 7) selects which input gpmc_wait pin is used for the device attached to the corresponding chip-select.
- The polarity of the wait pin is defined through the WAITxPINPOLARITY bit of the GPMC.GPMC_CONFIG register. A wait pin configured to be active low means that low level on the WAIT signal indicates that the data is not ready and that the data bus is invalid. When WAIT is inactive, data is valid.

The GPMC access engine can be configured by CS to monitor the wait pin of the external memory device or not, based on the access type: read or write.

- The GPMC.GPMC_CONFIG1_i[22] WAITREADMONITORING bit defines whether the wait pin should be monitored during read accesses or not.
- The GPMC.GPMC_CONFIG1_i[21] WAITWRITEMONITORING bit defines whether the wait pin should be monitored during write accesses or not.

The GPMC access engine can be configured to monitor the wait pin of the external memory device asynchronously or synchronously with the GPMC_CLK clock, depending on the access type: synchronous or asynchronous (the GPMC.GPMC_CONFIG1_i[29] READTYPE and GPMC.GPMC_CONFIG1_i[27] WRITETYPE bits).

11.1.5.4.1 Wait Monitoring During an Asynchronous Read Access

When wait-pin monitoring is enabled for read accesses (WAITREADMONITORING), the effective access time is a logical AND combination of the RDACCESSTIME timing completion and the wait-deasserted state.

During asynchronous read accesses with wait-pin monitoring enabled, the wait pin must be at a valid level (asserted or deasserted) for at least two GPMC clock cycles before RDACCESSTIME completes, to ensure correct dynamic access-time control through wait-pin monitoring. The advance pipelining of the two GPMC clock cycles is the result of the internal synchronization requirements for the WAIT signal.

In this context, RDACCESSTIME is used as a WAIT invalid timing window and is set to such a value that the wait pin is at a valid state two GPMC clock cycles before RDACCESSTIME completes.

Similarly, during a multiple-access cycle (for example, asynchronous read page mode), the effective access time is a logical AND combination of PAGEBURSTACCESSTIME timing completion and the wait-deasserted state. Wait-monitoring pipelining is also applicable to multiple accesses (access within a page).

- WAIT monitored as active freezes the CYCLETIME counter. For an access within a page, when the CYCLETIME counter is by definition in a lock state, WAIT monitored as asserted extends the current access time in the page. Control signals are kept in their current state. The data bus is considered invalid, and no data are captured during this clock cycle.

- WAIT monitored as inactive unfreezes the CYCLETIME counter. For an access within a page, when the CYCLETIME counter is by definition in a lock state, WAIT monitored as inactive completes the current access time and starts the next access phase in the page. The data bus is considered valid, and data are captured during this clock cycle. In case of a single access or if this was the last access in a multiple-access cycle, all signals are controlled according to their related control timing value and according to the CYCLETIME counter status.

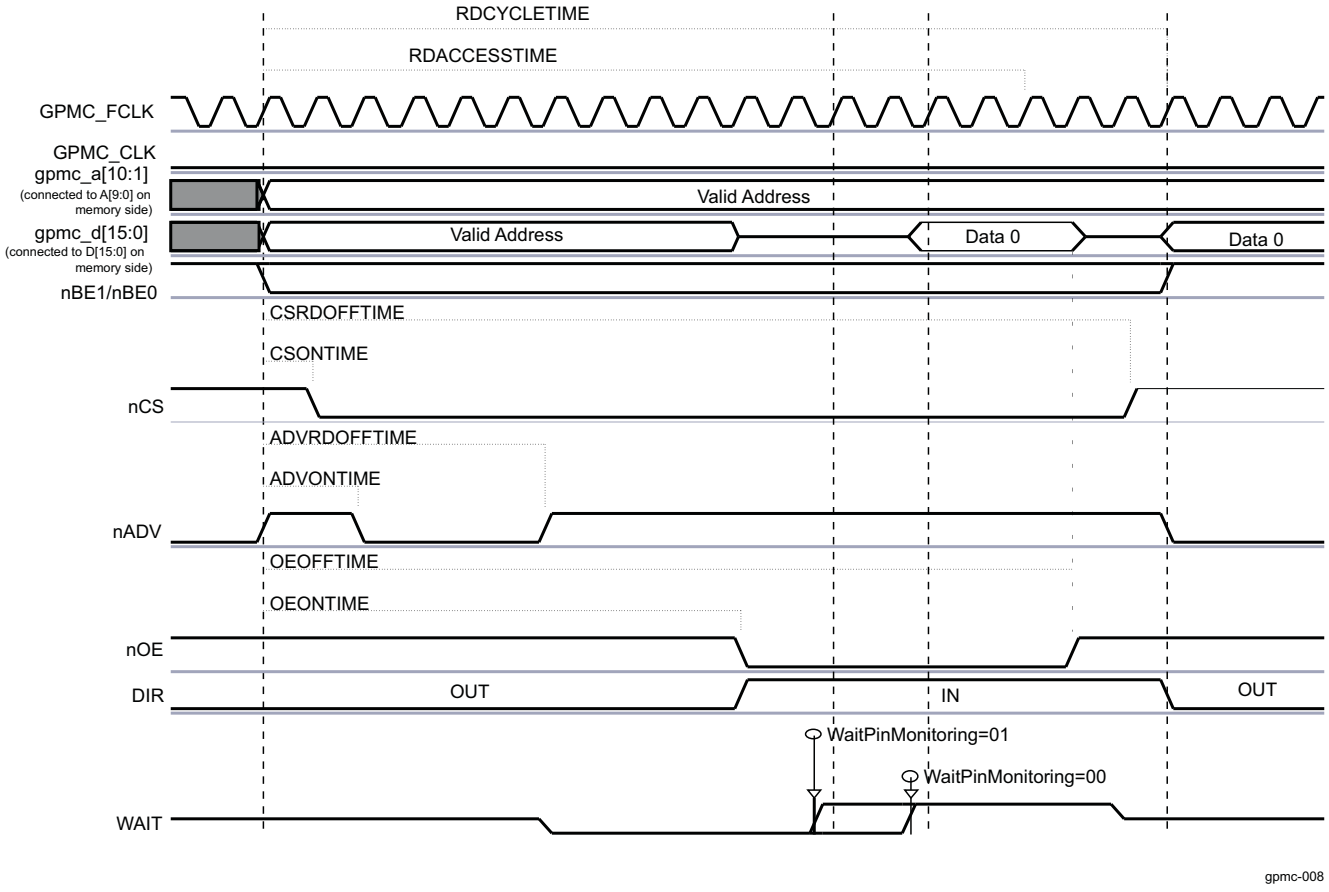
When a delay larger than two GPMC clocks must be observed between wait-pin deactivation time and data valid time (including the required GPMC and the device data setup time), an extra delay can be added between wait-pin deassertion time detection and effective data-capture time and the effective unlock of the CYCLETIME counter. This extra delay can be programmed in the GPMC.GPMC_CONFIG1_i[19:18] WAITMONITORINGTIME field (i = 0 to 7).

Notes:

- The WAITMONITORINGTIME parameter does not delay the wait-pin active or inactive detection, nor does it modify the two GPMC clocks pipelined detection delay.
- This extra delay is expressed as a number of GPMC_CLK clock cycles, even though the access is defined as asynchronous, and no GPMC_CLK clock is provided to the external device. Still, GPMCFCLKDIVIDER is used as a divider for the GPMC clock, so it must be programmed to define the correct WAITMONITORINGTIME delay.

Figure 11-8 shows wait behavior during an asynchronous single read access.

Figure 11-8. Wait Behavior During an Asynchronous Single Read Access (GPMCFCLKDivider = 1)



Note: The WAIT signal is active low. WAITMONITORINGTIME = 00, 01.

11.1.5.4.2 Wait Monitoring During an Asynchronous Write Access

When wait-pin monitoring is enabled for write accesses (GPMC.GPMC_CONFIG1_i[21] WAITWRITEMONITORING bit = 0x1), the WAIT-invalid timing window is defined by the WRACCESSTIME field. WRACCESSTIME must be set so that the wait pin is at a valid state two GPMC clock cycles before WRACCESSTIME completes. The advance pipelining of the two GPMC clock cycles is the result of the internal synchronization requirements for the WAIT signal.

- WAIT monitored as active freezes the CYCLETIME counter. This informs the GPMC that the data bus is not captured by the external device. The control signals are kept in their current state. The data bus still drives the data.
- WAIT monitored as inactive unfreezes the CYCLETIME counter. This informs that the data bus is correctly captured by the external device. All signals, including the data bus, are controlled according to their related control timing value and to the CYCLETIME counter status.

When a delay larger than two GPMC clock cycles must be observed between wait-pin deassertion time and the effective data write into the external device (including the required GPMC data setup time and the device data setup time), an extra delay can be added between wait-pin deassertion time detection and effective data write time into the external device and the effective unfreezing of the CYCLETIME counter. This extra delay can be programmed in the GPMC.GPMC_CONFIG1_i[19:18] WAITMONITORINGTIME fields (i = 0 to 7).

Notes:

- The WAITMONITORINGTIME parameter does not delay the wait-pin assertion or deassertion detection, nor does it modify the two GPMC clock cycles pipelined detection delay.
- This extra delay is expressed as a number of GPMC_CLK clock cycles, even though the access is defined as synchronous, and even though no clock is provided to the external device. Still, GPMC_CONFIG1_i[1:0] GPMCFCLKDIVIDER is used as a divider for the GPMC clock and so it must be programmed to define the correct WAITMONITORINGTIME delay.

11.1.5.4.3 Wait Monitoring During a Synchronous Read Access

During synchronous accesses with wait-pin monitoring enabled, the wait pin is captured synchronously with GPMC_CLK, using the rising edge of this clock.

The WAIT signal can be programmed to apply to the same clock cycle it is captured in. Alternatively, it can be sampled one or two GPMC_CLK cycles ahead of the clock cycle it applies to. This pipelining is applicable to the entire burst access, and to all data phase in the burst access. This WAIT pipelining depth is programmed in the GPMC.GPMC_CONFIG1_i[19:18] WAITMONITORINGTIME field (where i = 0 to 7), and is expressed as a number of GPMC_CLK clock cycles.

In synchronous mode, when wait-pin monitoring is enabled (GPMC.GPMC_CONFIG1_i[22] WAITREADMONITORING bit), the effective access time is a logical AND combination of the RDACCESSTIME timing completion and the WAIT deasserted-state detection.

Depending on the programmed WAITMONITORINGTIME value, the wait pin should be at a valid level, either asserted or deasserted:

- In the same clock cycle the data is valid if WAITMONITORINGTIME = 0 (at RDACCESSTIME completion)
- In the WAITMONITORINGTIME $\times$ (GPMCFCLKDIVIDER + 1) GPMC_FCLK clock cycles before RDACCESSTIME completion if WAITMONITORINGTIME $\neq$ 0

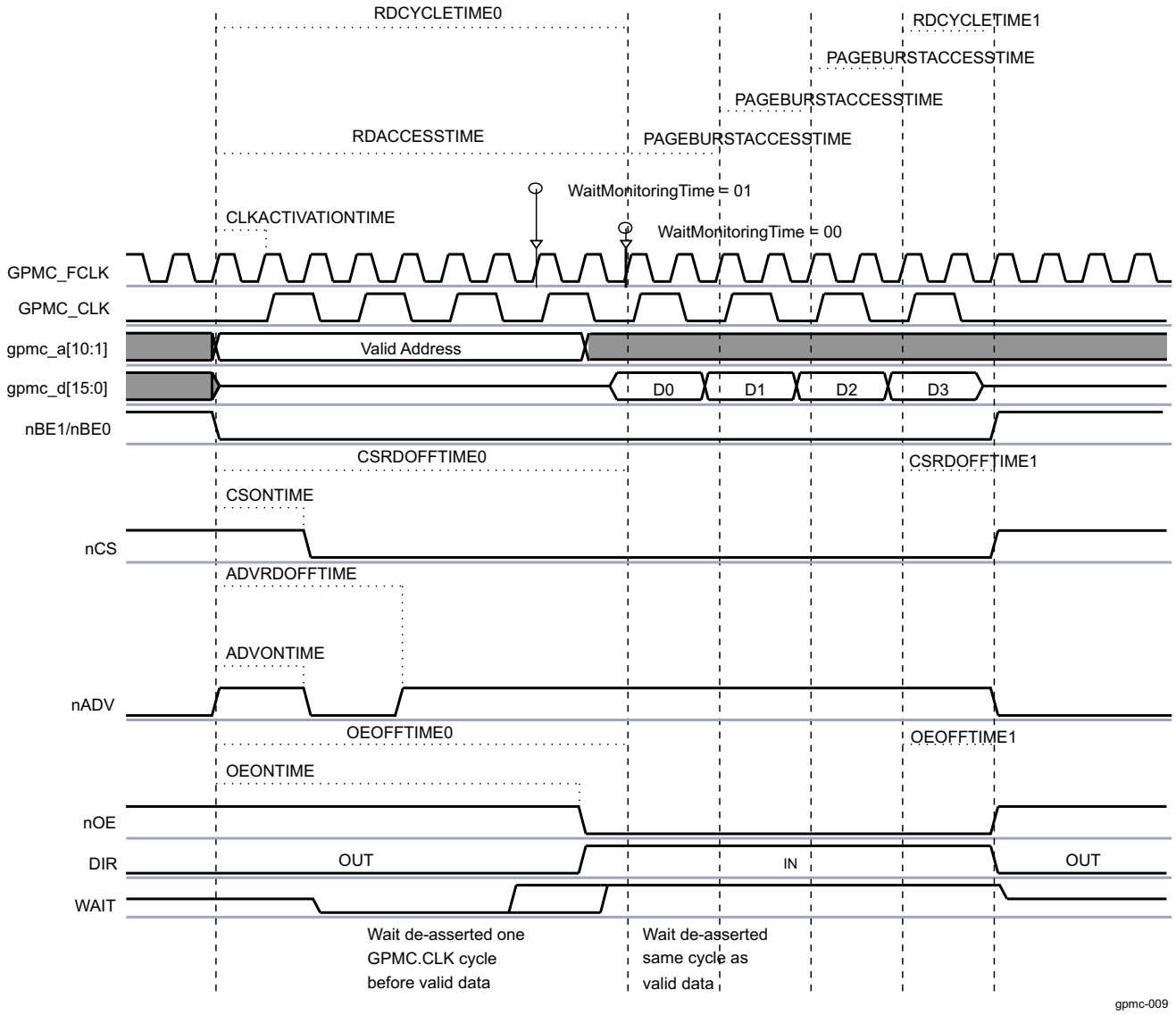
Similarly, during a multiple-access cycle (burst mode), the effective access time is a logical AND combination of PAGEBURSTACCESSTIME timing completion and the wait-inactive state. The Wait pipelining depth programming applies to the whole burst access.

- WAIT monitored as active freezes the CYCLETIME counter. For an access within a burst (when the CYCLETIME counter is by definition in a lock state), WAIT monitored as active extends the current access time in the burst. Control signals are kept in their current state. The data bus is considered invalid, and no data are captured during this clock cycle.

- WAIT monitored as inactive unfreezes the CYCLETIME counter. For an access within a burst (when the CYCLETIME counter is by definition in lock state), WAIT monitored as inactive completes the current access time and starts the next access phase in the burst. The data bus is considered valid, and data are captured during this clock cycle. In a single access or if this was the last access in a multiple-access cycle, all signals are controlled according to their relative control timing value and the CYCLETIME counter status.

Figure 11-9 shows wait behavior during a synchronous read burst access.

Figure 11-9. Wait Behavior During a Synchronous Read Burst Access



Note: The WAIT signal is active low. WAITMONITORINGTIME = 00, 01.

11.1.5.4.4 Wait Monitoring During a Synchronous Write Access

During synchronous accesses with wait-pin monitoring enabled (the WAITWRITEMONITORING bit), the wait pin is captured synchronously with GPMC_CLK, using the rising edge of this clock.

If enabled, external wait-pin monitoring can be used in combination with WRACCESSTIME to control the effective memory device GPMC_CLK capture edge.

Wait-monitoring pipelining depth is similar to synchronous read access:

- At WRACCESSTIME completion if WAITMONITORINGTIME = 0
- The WAITMONITORINGTIME × (GPMCFCLKDIVIDER + 1) GPMC_FCLK cycles before WRACCESSTIME completion if WAITMONITORINGTIME ≠ 0.

Wait-monitoring pipelining definition applies to whole burst accesses:

- WAIT monitored as active freezes the CYCLETIME counter. For accesses within a burst, when the CYCLETIME counter is by definition in a lock state, WAIT monitored as active indicates that the data bus is not being captured by the external device. Control signals are kept in their current state. The data bus is kept in its current state.
- WAIT monitored as inactive unfreezes the CYCLETIME counter. For accesses within a burst, when the CYCLETIME counter is by definition in a lock state, WAIT monitored as inactive indicates the effective data capture of the bus by the external device and starts the next access of the burst. In case of a single access or if this was the last access in a multiple access cycle, all signals, including the data bus, are controlled according to their related control timing value and the CYCLETIME counter status.

Note: Wait monitoring is supported for all configurations except for GPMC_CONFIG1_i[19:18] WAITMONITORINGTIME = 0x 0 (where i = 0 to 7) for write bursts with a clock divider of 1 or 2 (GPMC_CONFIG1_i[1:0] GPMCFCLKDIVIDER bit field equal to 0x0 or 0x1, respectively).

11.1.5.4.5 WAIT With NAND Device

For details about the use of the wait pin for communication with a NAND flash external device, see Section 11.1.5.14.2, *NAND Device-Ready Pin*.

11.1.5.4.6 Idle Cycle Control Between Successive Accesses

11.1.5.4.6.1 Bus Turnaround (BUSTURNAROUND)

To prevent data-bus contention, an access that follows a read access to a slow memory/device (that is, control the nCS/nOE de-assertion to data bus in high-impedance delay) must be delayed.

The bus turnaround is a time-out counter starting after nCS or nOE de-assertion time (whichever occurs first) and delays the next access start-cycle time. It is programmed trhough the GPMC.GPMC_CONFIG6_i[3:0] BUSTURNAROUND bit field (where i = 0 to 7).

After a read access to a chip-select with a non zero BUSTURNAROUND, the next access is delayed until the BUSTURNAROUND delay completes, if the next access is one of the following:

- A write access to any chip-select (same or different from the chip-select data was read from)
- A read access to a different chip-select from the chip-select data was read access from
- A read or write access to a chip-select associated with an address/data-multiplexed device

Another way to prevent bus contention is to define an earlier nCS or nOE deassertion time for slow devices or to extend the value of RDCYCLETIME. Doing this prevents bus contention, but affects all accesses of this specific chip-select.

11.1.5.4.6.2 Idle Cycles Between Accesses to Same Chip-Select (CYCLE2CYCLESAMECSEN, CYCLE2CYCLEDELAY)

Some devices require a minimum chip-select signal inactive time between accesses. The GPMC.GPMC_CONFIG6_i[7] CYCLE2CYCLESAMECSEN bit (i = 0 to 7) enables insertion of a minimum number of GPMC_FCLK cycles, defined by the GPMC.GPMC_CONFIG6_i[11:8] CYCLE2CYCLEDELAY field, between successive accesses of any type (read or write) to the same chip-select.

If CYCLE2CYCLESAMECSEN is enabled, any subsequent access to the same chip-select is delayed until its CYCLE2CYCLEDELAY completes. The CYCLE2CYCLEDELAY counter starts when CSRDOFFTIME/CSWROFFTIME completes.

The same applies to successive accesses occurring during Word32 or burst accesses split into successive single accesses when the single-access mode is used (GPMC_CONFIG1_i[30] READMULTIPLE = 0 or GPMC_CONFIG1_i[28] WRITEMULTIPLE = 0).

All control signals are kept in their default states during these idle GPMC_FCLK cycles. This prevents back-to-back accesses to the same chip-select without idle cycles between accesses.

11.1.5.4.6.3 Idle Cycles Between Accesses to Different Chip-Select (CYCLE2CYCLEDIFFCSEN, CYCLE2CYCLEDELAY)

Because of the pipelined behavior of the system, successive accesses to different chip-selects can occur back-to-back with no idle cycles between accesses. Depending on the control signals (nCS, nADV/ALE, nBE0/CLE, nOE/RE, nWE) assertion and de-assertion timing parameters and on the IC timing parameters, some control signals assertion times may overlap between the successive accesses to different CS. Similarly, some control signals (WE, OE/RE) may not respect required transition times.

To work around the overlapping and to observe the required control-signal transitions, a minimum of CYCLE2CYCLEDELAY inactive cycles is inserted between the access being initiated to this chip-select and the previous access ending for a different chip-select. This applies to any type of access (read or write).

If GPMC_CONFIG6_i[6] CYCLE2CYCLEDIFFCSEN is enabled, the chip-select access is delayed until CYCLE2CYCLEDELAY cycles have expired since the end of a previous access to a different chip-select. CYCLE2CYCLEDELAY count starts at CSRDOFFTIME/CSWROFFTIME completion. All control signals are kept inactive during the idle GPMC_FCLK cycles.

Note: CYCLE2CYCLESAMECSEN and CYCLE2CYCLEDIFFCSEN should be set in the GPMC_CONFIG6_i registers to insert idle cycles between accesses on this chip-select and after accesses to a different chip-select, respectively.

The CYCLE2CYCLEDELAY delay runs in parallel with the BUSTURNAROUND delay. BUSTURNAROUND is a timing parameter defined for the ending chip-select access, whereas CYCLE2CYCLEDELAY is a timing parameter defined for starting chip-select access. The effective minimum delay between successive accesses is based on the larger delay timing parameter and on the access type combination, since bus turnaround does not apply to all access types. See Section 11.1.5.4.6.1 for more details on bus turnaround.

Table 11-3 describes the configuration required for idle cycle insertion.

Table 11-3. Idle Cycle Insertion Configuration

1st Access Type	BUSTURN AROUND Timing Parameter	Second Access Type	Chip-Select	Add/Data Multiplexed	CYCLE2 CYCLE SAMECSEN Parameter	CYCLE2 CYCLE DIFFCSEN Parameter	Idle Cycle Insertion Between the Two Accesses
R/W	= 0	R/W	Any	Any	0	x	No idle cycles are inserted if the two accesses are well pipelined.
R	> 0	R	Same	Nonmuxed	x	0	No idle cycles are inserted if the two accesses are well pipelined.
R	> 0	R	Different	Nonmuxed	0	0	BTA cycles are inserted.
R	> 0	R/W	Any	Muxed	0	0	BTA cycles are inserted.
R	> 0	W	Any	Any	0	0	BTA cycles are inserted.
W	> 0	R/W	Any	Any	0	0	No idle cycles are inserted if the two accesses are well pipelined.
R/W	= 0	R/W	Same	Any	1	x	CYCLE2CYCLEDELAY cycles are inserted.
R/W	= 0	R/W	Different	Any	x	1	CYCLE2CYCLEDELAY cycles are inserted.

Table 11-3. Idle Cycle Insertion Configuration (continued)

1st Access Type	BUSTURN AROUND Timing Parameter	Second Access Type	Chip-Select	Add/Data Multiplexed	CYCLE2 CYCLE SAMECSEN Parameter	CYCLE2 CYCLE DIFFCSEN Parameter	Idle Cycle Insertion Between the Two Accesses
R/W	> 0	R/W	Same	Any	1	x	CYCLE2CYCLEDELAY cycles are inserted. If BTA idle cycles already apply on these two back-to-back accesses, the effective delay is max (BUSTURNAROUND, CYCLE2CYCLEDELAY).
R/W	> 0	R/W	Different	Any	x	1	CYCLE2CYCLEDELAY cycles are inserted. If BTA idle cycles already apply on these two back-to-back accesses, the effective delay is maximum (BUSTURNAROUND, CYCLE2CYCLEDELAY).

11.1.5.4.7 Slow Device Support (TIMEPARAGRANULARITY Parameter)

All access-timing parameters can be multiplied by 2 by setting the GPMC.GPMC_CONFIG1_i[4] TIMEPARAGRANULARITY bit (where i stands for the GPMC chip-select value, i = 0 to 7). Increasing all access timing parameters allows support of slow devices.

11.1.5.5 gpmc_io_dir Pin

The gpmc_io_dir pin is used to control I/O direction on the GPMC data bus gpmc_d[15:0]. Depending on top-level pad multiplexing, this signal can be output and used externally to the OMAP device, if required.

The gpmc_io_dir pin is low during transmit (OUT) and high during receive (IN).

For write accesses, the gpmc_io_dir pin stays OUT from start-cycle time to end-cycle time.

For read accesses, the gpmc_io_dir pin goes from OUT to IN at nOE assertion time and stays IN until:

- BUSTURNAROUND is enabled:
  - The gpmc_io_dir pin goes from IN to OUT at end-cycle time plus programmable bus turnaround time.
- BUSTURNAROUND is disabled:
  - After an asynchronous read access, the gpmc_io_dir pin goes from IN to OUT at RDACCESSTIME + 1 GPMC_FCLK cycle or when RDCYCLETIME completes, whichever occurs last.
  - After a synchronous read access, the gpmc_io_dir pin goes from IN to OUT at RDACCESSTIME + 2 GPMC_FCLK cycles or when RDCYCLETIME completes, whichever occurs last.

Because of the bus-keeping feature of the GPMC, after a read or write access and with no other accesses pending, the default value of the gpmc_io_dir pin is OUT (see Section 11.1.5.3.10, *Bus Keeping Support*).

To prevent unnecessary toggling, the gpmc_io_dir pin stays IN between two successive read accesses to a nonmultiplexed device (address mapping supports nonmultiplexed 16-bit wide devices with limited address (2 Kbytes)).

Figure 11-10 shows address mapping in nonmultiplexed mode with a limited address range (A[10:1]).

11.1.5.6 Reset

No reset signal is sent to the external memory device by the GPMC. The PRCM specifications provide more information about external-device reset.

The PRCM module provides an input pin, global_rst_n, to the GPMC:

- The global_rst_n pin is activated during OMAP warm reset and cold reset.
- The global_rst_n pin initializes the internal state-machine and the internal configuration registers.

11.1.5.7 WRITE PROTECT (nWP)

When connected to the attached memory device, the WRITE PROTECT signal can enable or disable the lockdown function of the attached memory.

The gpmc_nwp output pin value is controlled through the GPMC.GPMC_CONFIG[4] WRITEPROTECT bit, which is common to all CS.

11.1.5.8 BYTE ENABLE (nBE1/nBE0)

BYTE ENABLE signals (nBE1/nBE0) are:

- Valid (asserted or nonasserted according to the incoming system request) from access start to access completion for asynchronous and synchronous single accesses
- Asserted low from access start to access completion for asynchronous and synchronous multiple read accesses
- Valid (asserted or nonasserted, according to the incoming system request) synchronously to each written data for synchronous multiple write accesses

11.1.5.9 Asynchronous Access Description

In asynchronous operations:

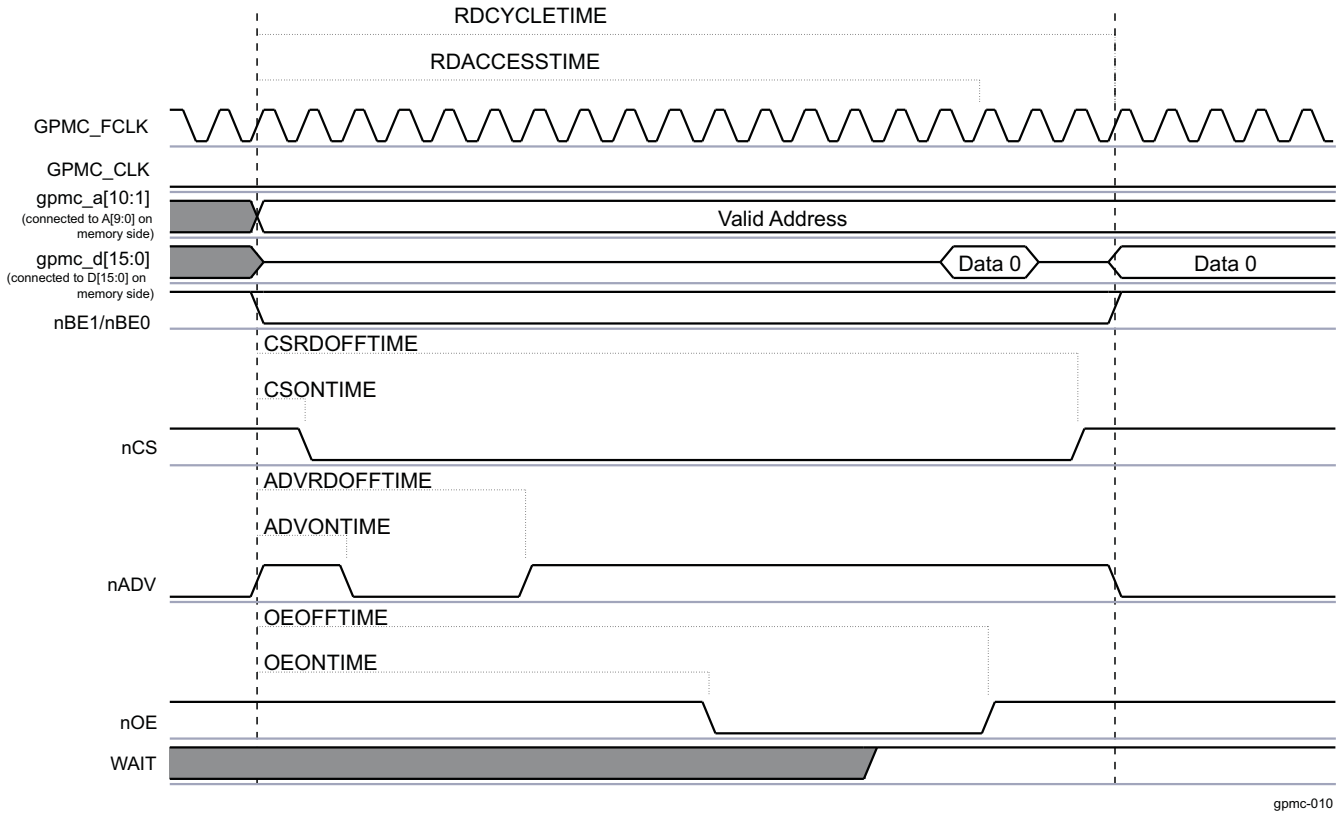
- GPMC_CLK is not provided outside the GPMC.
- GPMC_CLK is kept low.

11.1.5.9.1 Asynchronous Single Read

11.1.5.9.1.1 Asynchronous Single Read Operation on a Nonmultiplexed Device

Figure 11-10 shows an asynchronous single read operation on a nonmultiplexed device.

Figure 11-10. Asynchronous Single Read on an Address/Data-Nonmultiplexed Device



In the following section i stands for the chip-select number, i = 0 to 7.

- GPMC.GPMC_CONFIG[1] LIMITEDADDRESS set to 1 (A26-A11 are not modified during an external memory access)
- GPMC.GPMC_CONFIG1_i register settings:
  - GPMC_CONFIG1_i[30] READMULTIPLE bit at 0 (read single access)
  - GPMC_CONFIG1_i[29] READTYPE bit at 0 (asynchronous read)
  - GPMC_CONFIG1_i[9] MUXADDDATA bit at 0 (non multiplexed device)
- Chip-select signal nCS:
  - nCS assertion time is controlled by the GPMC_CONFIG2_i[3:0] CSONTIME field. It controls the address setup time to nCS assertion.
  - nCS deassertion time is controlled by the GPMC_CONFIG2_i[12:8] CSRDOFFTIME field. It controls the address hold time from nCS deassertion.
- Address valid signal nADV:
  - nADV assertion time is controlled by the GPMC_CONFIG3_i[3:0] ADVONTIME field.
  - nADV deassertion time is controlled by the GPMC_CONFIG3_i[12:8] ADVRDOFFTIME field.
- Output enable signal nOE:
  - nOE assertion indicates a read cycle.
  - nOE assertion time is controlled by the GPMC_CONFIG4_i[3:0] OEONTIME field.
  - nOE deassertion time is controlled by the GPMC_CONFIG4_i[12:8] OEOFFTIME field.
- Read data is latched when RDACCESSTIME completes. Access time is defined in the GPMC.GPMC_CONFIG5_i[20:16] RDACCESSTIME field.
- The end of the access is defined by the RDCYCLETIME parameter. The read cycle time is defined in the GPMC.GPMC_CONFIG5_i[4:0] RDCYCLETIME field.

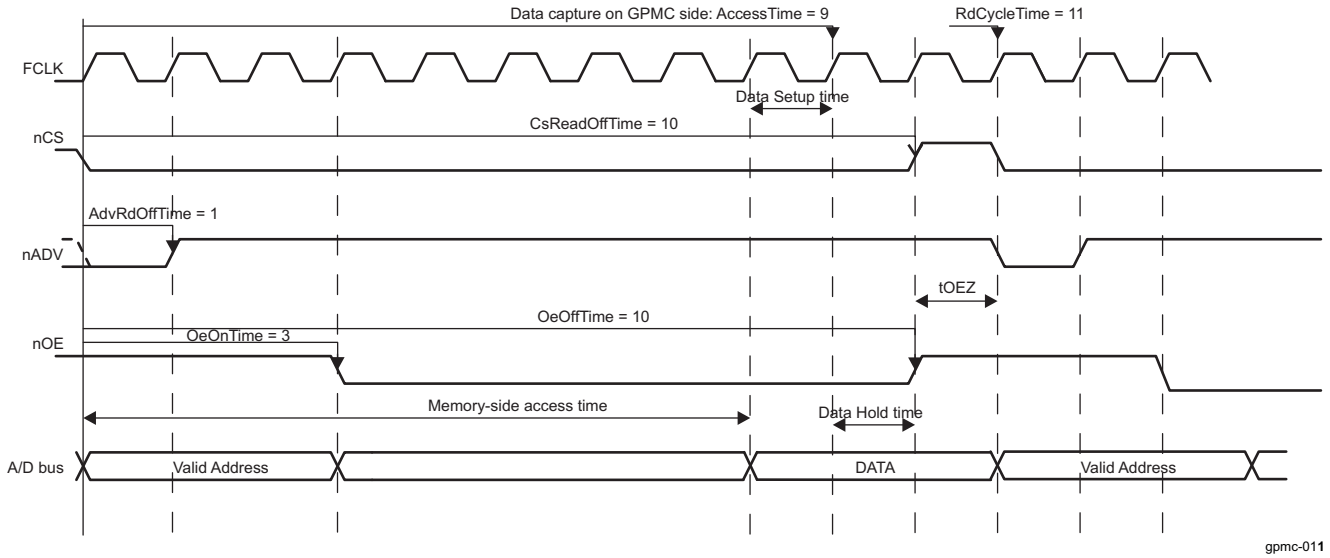
- Direction signal DIR: DIR goes from OUT to IN at the same time that nOE is asserted.

After a read operation, if no other access (read or write) is pending, the data bus is driven with the previous read value. See Section 11.1.5.3.10, *Bus Keeping Support* for more details.

11.1.5.9.1.2 Asynchronous Single-Read Operation on an Address/Data Multiplexed Device

Figure 11-11 shows an asynchronous single read operation on an address/data-multiplexed device.

Figure 11-11. Asynchronous Single Read on an Address/Data-Multiplexed Device



When the GPMC generates a read access to an address/data-multiplexed device, it drives the address bus until nOE assertion time. For details, see Section 11.1.5.2.3, *Address/Data-Multiplexing Interface*.

GPMC.GPMC_CONFIG1_i register settings (i = 0 to 7):

- READMULTIPLE bit at 0 (read single access)
- READTYPE bit at 0 (read asynchronous)
- MUXADDDATA bit at 1 (address/data-multiplexed device)

Address bits ([16:1] from a GPMC perspective, [15:0] from an external device perspective) are placed on the address/data bus, and the remaining address bits [25:16] are placed on the address bus. The address phase ends at nOE assertion, when the DIR signal goes from OUT to IN.

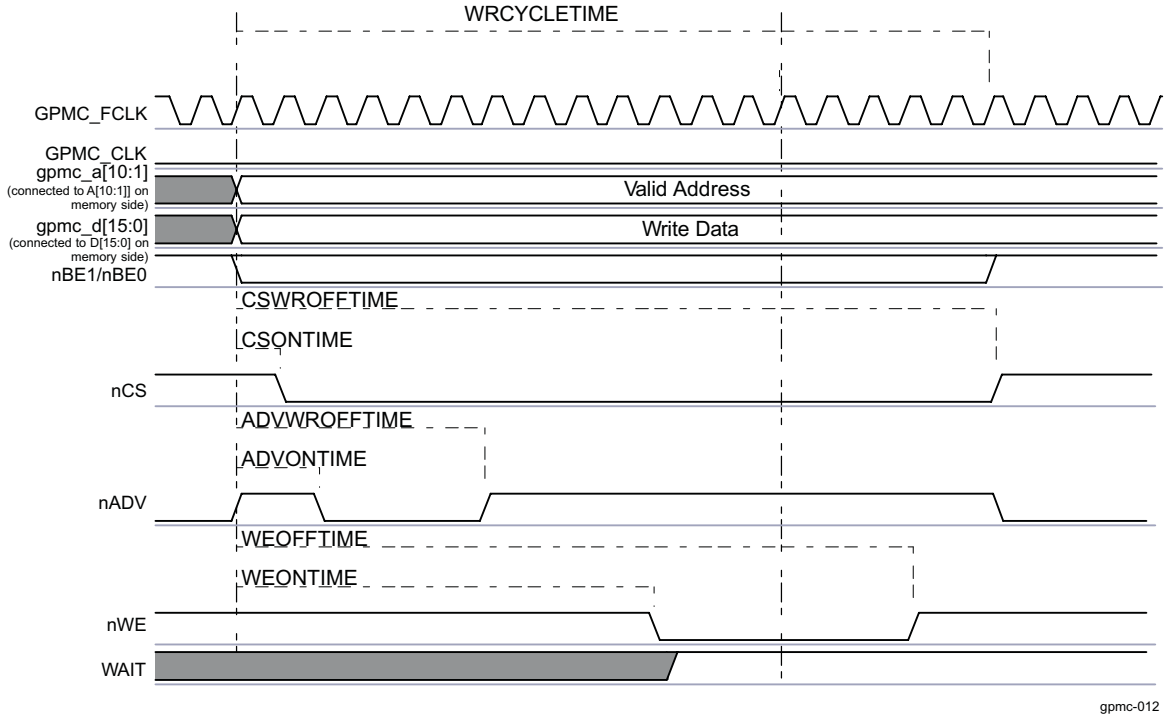
The nCS, nADV, nOE, and DIR signals are controlled in the same way as nonmultiplexed accesses.

11.1.5.9.2 Asynchronous Single Write

11.1.5.9.2.1 Asynchronous Single Write Operation on a Nonmultiplexed Device

Figure 11-12 shows an asynchronous single write operation on a nonmultiplexed device.

Figure 11-12. Asynchronous Single Write on an Address/Data-Nonmultiplexed Device



In the following section i stands for the chip-select number, i = 0 to 7.

- GPMC.GPMC_CONFIG[1] LIMITEDADDRESS set to 1 (A26-A11 are not modified during an external memory access)
- GPMC.GPMC_CONFIG1_i register settings:
  - WRITEMULTIPLE bit at 0 (write single access)
  - WRITETYPE bit at 0 (write asynchronous)
  - MUXADDDATA bit at 0 (nonaddress/data-multiplexed device)
- Chip-select signal nCS:
  - nCS assertion time is controlled by the GPMC.GPMC_CONFIG2_i[3:0] CSONTIME field and ensures address setup time to nCS assertion.
  - nCS deassertion time is controlled by the GPMC.GPMC_CONFIG2_i[20:16] CSWROFFTIME field and ensures address hold time to nCS deassertion.
- Address valid signal nADV:
  - nADV assertion time is controlled by the GPMC.GPMC_CONFIG3_i[3:0] ADVONTIME field.
  - nADV deassertion time is controlled by the GPMC.GPMC_CONFIG3_i[20:16] ADVWROFFTIME field.

Address and data are driven on their corresponding buses at start-of-cycle time.

- Write enable signal nWE:
  - nWE assertion indicates a write cycle.
  - nWE assertion time is controlled by the GPMC.GPMC_CONFIG4_i[19:16] WEONTIME field.
  - nWE deassertion time is controlled by the GPMC.GPMC_CONFIG4_i[28:24] WEOFFTIME field.
- Direction signal DIR:
  DIR signal is OUT during the entire access.

- The end of the access is defined by the WRCYCLETIME parameter.
  This write-cycle time is defined in the GPMC.GPMC_CONFIG5_i[12:8] WRCYCLETIME field.

After a write operation, if no other access (read or write) is pending, the data bus keeps its previous value. See Section 11.1.5.3.10, *Bus Keeping Support*.

In the GPMC, when a 16-bit wide device is attached to the controller, a Word32 write access is split into two Word16 write accesses. For more information about GPMC access size and type adaptation, see Section 11.1.5.2.7, *System Burst Versus External Device Burst Support*.

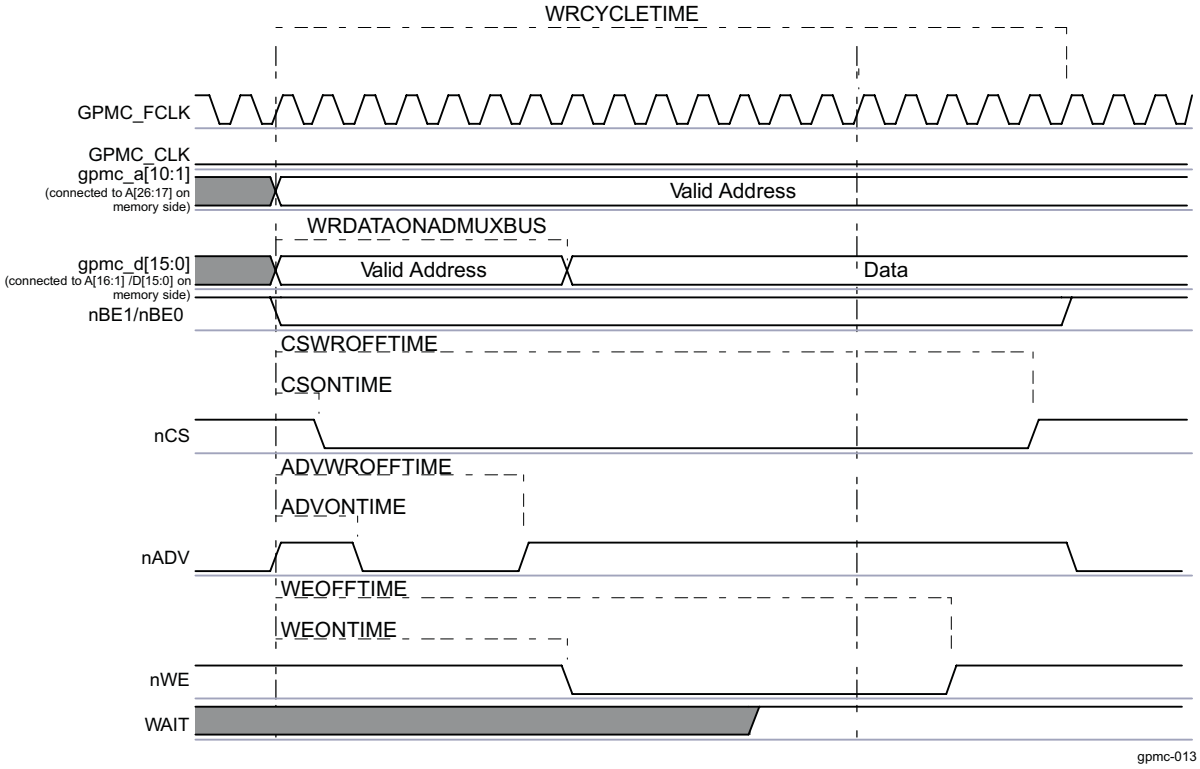
Between two successive accesses, if an nCS pulse is needed:

- The GPMC.GPMC_CONFIG6_i[11:8] CYCLE2CYCLEDELAY field can be programmed with GPMC.GPMC_CONFIG6_i[7] CYCLE2CYCLESAMECSEN enabled.
- The CSWROFFTIME and CSONTIME parameters also allow a chip-select pulse, but this affects all other types of access.

11.1.5.9.2.2 Asynchronous Single Write Operation on an Address/Data-Multiplexed Device

Figure 11-13 shows an asynchronous single write operation on an address/data-multiplexed device.

Figure 11-13. Asynchronous Single Write on an Address/Data-Multiplexed Device



When the GPMC generates a write access to an address/data-multiplexed device, it drives the address on the address/data muxed bus until WRDATAONADMUXBUS time, and then it drives the data. For more information, see Section 11.1.5.2.3, *Address/Data-Multiplexing Interface*.

GPMC.GPMC_CONFIG1_i register settings (i = 0 to 7):

- WRITEMULTIPLE bit at 0 (write single access)
- WRITETYPE bit at 0 (write asynchronous)
- MUXADDDATA bit at 1 (address/data-multiplexed device)

Address bits [16:1] are placed on the address/data bus at the start of cycle time, and the remaining address bits [26:17] are placed on the address bus.

The nCS, nADV, and nWE signals are controlled in the same way as nonmultiplexed accesses.

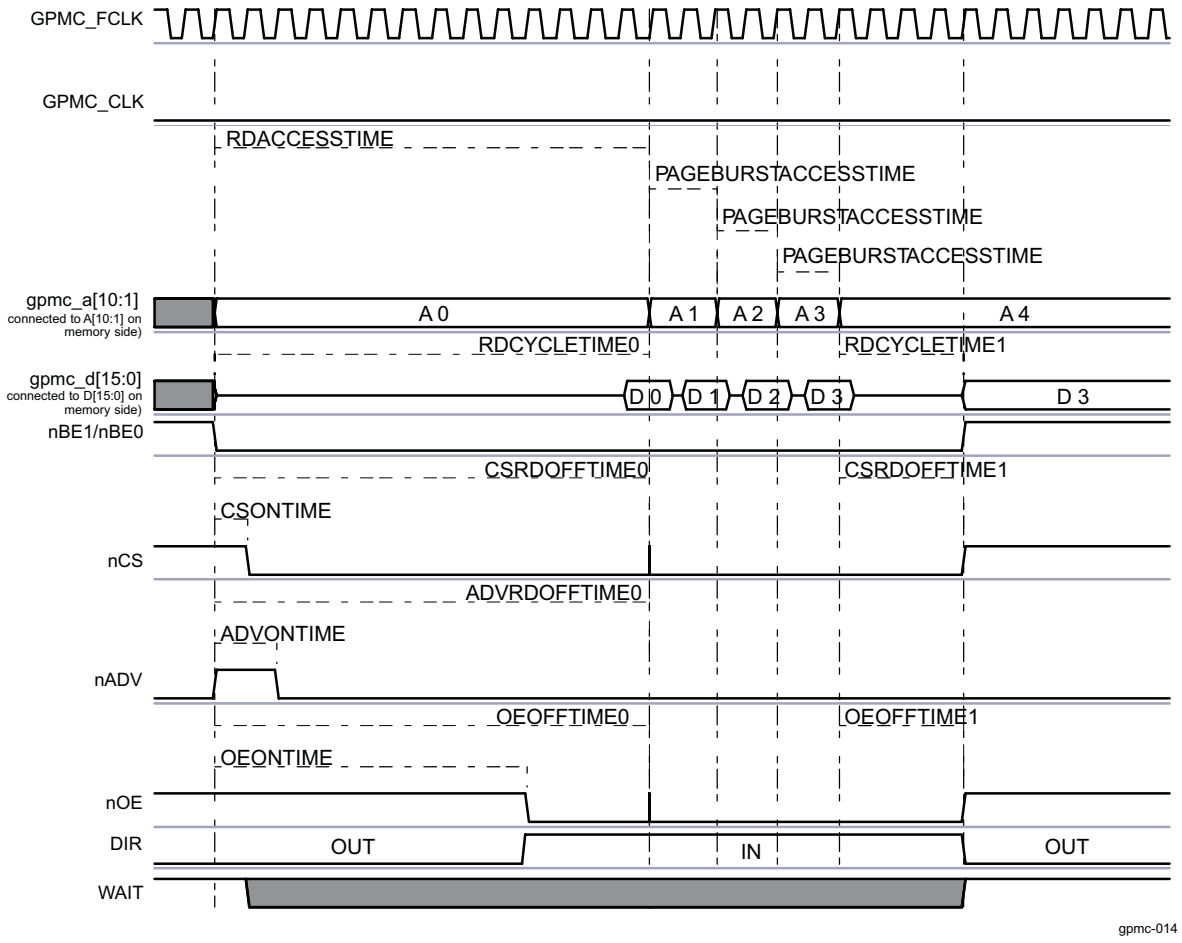
Data is driven on the address/data bus at a GPMC_CONFIG6_i[19:16] WRDATAONADMUXBUS time.

Note: Write multiple access in asynchronous mode is not supported. If WRITEMULTIPLE is enabled with WRITETYPE as asynchronous, the GPMC processes single asynchronous accesses.

11.1.5.9.3 Asynchronous Multiple (Page Mode) Read

Figure 11-14 shows an asynchronous multiple read operation.

Figure 11-14. Asynchronous Multiple (Page Mode) Read



Note: The WAIT signal is active low.

In the following section i stands for the chip-select number, i = 0 to 7.

For read access with GPMC.GPMC_CONFIG1_i register settings:

- READMULTIPLE bit at 1 (read multiple access)
- READTYPE bit at 0 (read asynchronous)
- MUXADDDATA bit at 0 (non-address/data-multiplexed device). The page mode is not supported by address/data-multiplexed devices.

In Figure 11-14, two Word32 read host accesses on the GPMC configured with READMULTIPLE = 1, READTYPE = 0, and MUXADDDATA = 0 are merged into one multiple-read access (page mode of four Word16) on the attached device.

When RDACCESSTIME completes, control-signal timings are frozen during the multiple data transactions, corresponding to PAGEBURSTACCESSTIME multiplied by the number of remaining data transactions.

- Chip-select signal nCS:
  - nCS assertion time is controlled by the GPMC.GPMC_CONFIG2_i[3:0] CSONTIME field and ensures the address setup time to nCS assertion.
  - nCS deassertion time is controlled by the GPMC.GPMC_CONFIG2_i[12:8] CSRDOFFTIME field and ensures the address hold time to nCS deassertion.
- Address valid signal nADV:
  - nADV assertion time is controlled by the GPMC.GPMC_CONFIG3_i[3:0] ADVONTIME field.
  - nADV deassertion time is controlled by the GPMC.GPMC_CONFIG3_i[12:8] ADVRDOFFTIME field.
- Output enable signal nOE:
  - nOE assertion indicates a read cycle.
  - nOE assertion time is controlled by the GPMC.GPMC_CONFIG4_i[3:0] OEONTIME field.
  - nOE deassertion time is controlled by the GPMC.GPMC_CONFIG4_i[12:8] OEOFFTIME field.
- Initial latency for the first read data is controlled by the RDACCESSTIME parameter.

  The access time is defined in the GPMC.GPMC_CONFIG5_i[20:16] RDACCESSTIME field.

During consecutive accesses, the GPMC increments the address after each data read completes.

- Delay between successive read data in the page is controlled by the PAGEBURSTACCESSTIME parameter:
  - This timing is defined in the GPMC.GPMC_CONFIG5_i[27:24] PAGEBURSTACCESSTIME field.
  - Depending on the device page length, the GPMC can control device page crossing during a burst request and insert initial RDACCESSTIME latency. Note that page crossing is only possible with a new burst access, meaning a new initial access phase is initiated.
- Total access time (RDCYCLETIME) corresponds to RDACCESSTIME plus the address hold time starting from the nCS deassertion plus the time from RDACCESSTIME to CSRDOFFTIME.
  - The read cycle time is defined in the GPMC.GPMC_CONFIG5_i[4:0] RDCYCLETIME field.
  - In Figure 11-14, the RDCYCLETIME programmed value equals RDCYCLETIME0 (before paged accesses) + RDCYCLETIME1 (after paged accesses).
- Direction signal DIR:

  DIR goes from OUT to IN at the same time as nOE assertion time.

After a read operation, if no other access (read or write) is pending, the data bus is driven with the previous read value. See Section 11.1.5.3.10, *Bus Keeping Support*.

11.1.5.10 Synchronous Access

In synchronous operations:

- The GPMC_CLK clock is provided outside the GPMC when accessing the memory device.
- The GPMC_CLK clock is derived from the GPMC_FCLK clock using the GPMC.GPMC_CONFIG1_i[1:0] GPMCFCLKDIVIDER field (where i = 0 to 7).
- The GPMC.GPMC_CONFIG1_i[26:25] CLKACTIVATIONTIME field specifies that the GPMC_CLK is provided outside the GPMC 0, 1, or 2 GPMC_FCLK cycles after start access time until CycleTime completes.
- When the GPMC is configured for synchronous mode, the GPMC_CLK signal (which is an output) must also be set as an input (CONTROL.CONTROL_PADCONF_GPMC_NCS7[24] INPUTENABLE1 = 1). GPMC_CLK is looped back through the output and input buffers of the corresponding GPMC_CLK pad at OMAP boundary. The looped-back clock is used to synchronize the sampling of the memory signals.

11.1.5.10.1 Synchronous Single Read

Figure 11-15 and Figure 11-16 show a synchronous single-read operation with GPMCFCLKDIVIDER equal to 0 and 1, respectively.

Figure 11-15. Synchronous Single Read (GPMCFCLKDIVIDER = 0)

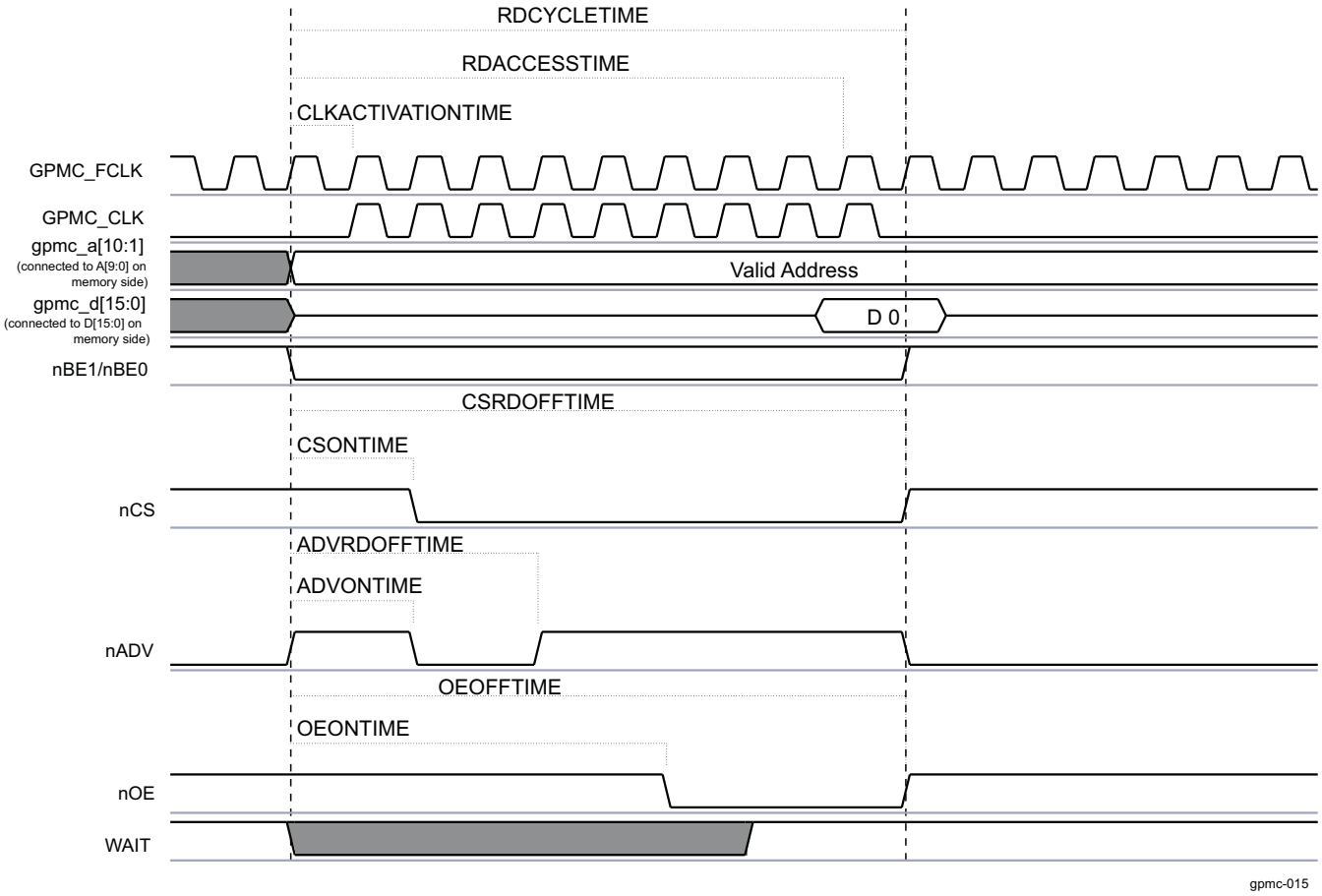
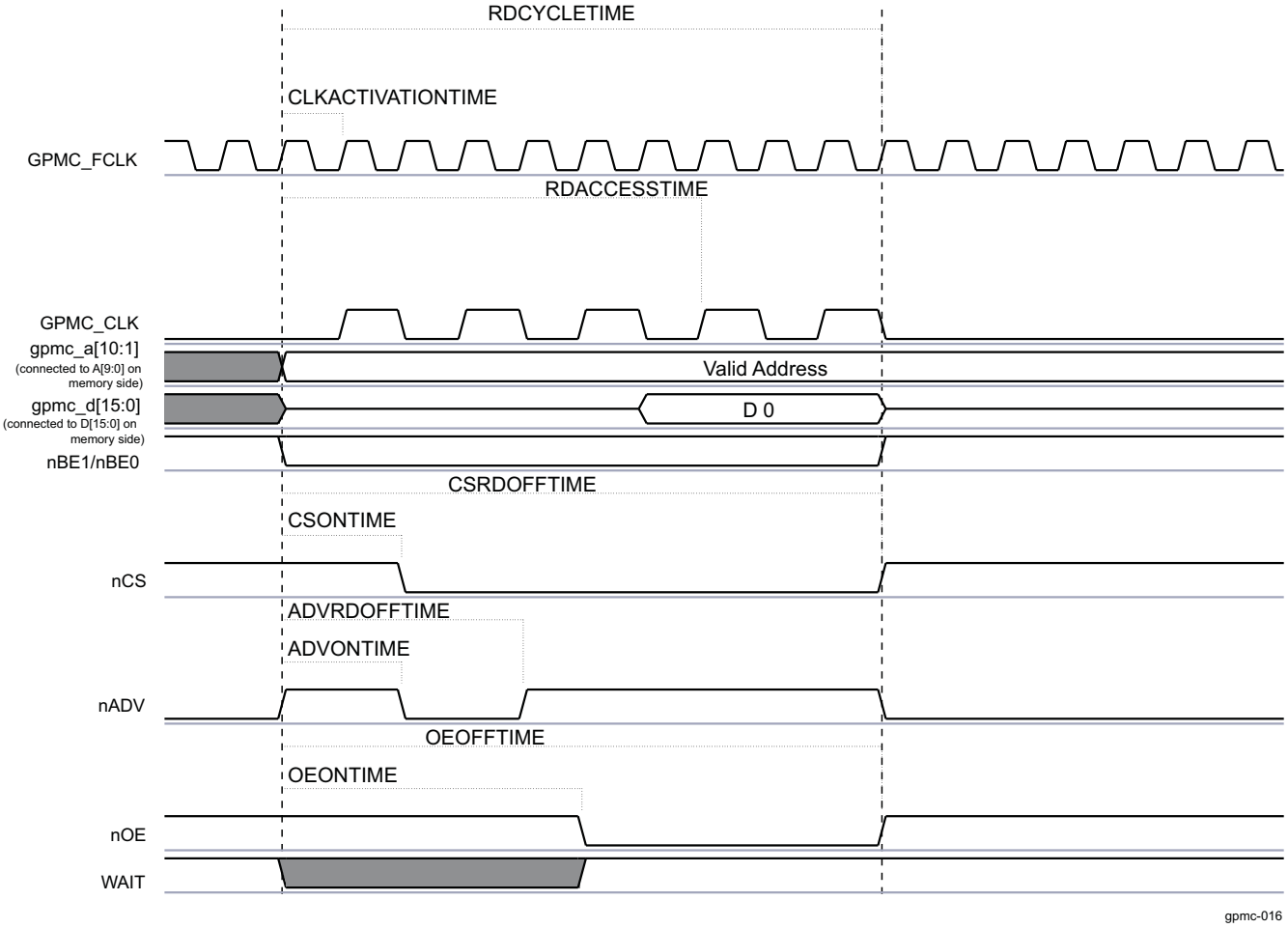


Figure 11-16. Synchronous Single Read (GPMCFCLKDIVIDER = 1)



In the following section i stands for the chip-select number, i = 0 to 7.

- GPMC.GPMC_CONFIG1_i register settings:
  - READMULTIPLE bit at 0 (read single access)
  - READTYPE bit at 1 (read synchronous)
  - MUXADDDATA bit at 0 (non-address/data-multiplexed device)
- Chip-select signal nCS:
  - nCS assertion time is controlled by the GPMC.GPMC_CONFIG2_i[3:0] CSONTIME field and ensures address setup time to nCS assertion.
  - nCS deassertion time is controlled by the GPMC.GPMC_CONFIG2_i[12:8] CSRDOFFTIME field and ensures address hold time to nCS deassertion.
- Address valid signal nADV:
  - nADV assertion time is controlled by the GPMC.GPMC_CONFIG3_i[3:0] ADVONTIME field.
  - nADV deassertion time is controlled by the GPMC.GPMC_CONFIG3_i[12:8] ADVRDOFFTIME field.
- Output enable signal nOE:
  - nOE assertion indicates a read cycle.
  - nOE assertion time is controlled by the GPMC.GPMC_CONFIG4_i[3:0] OEONTIME field.
  - nOE deassertion time is controlled by the GPMC.GPMC_CONFIG4_i[12:8] OEOFFTIME field.
- Initial latency for the first read data is controlled by GPMC.GPMC_CONFIG5_i[20:16] RDACCESSTIME or by monitoring the WAIT signal.
- Total access time (GPMC.GPMC_CONFIG5_i[4:0] RDCYCLETIME) corresponds to RDACCESSTIME

plus the address hold time from nCS deassertion, plus time from RDACCESSTIME to CSWROFFTIME.

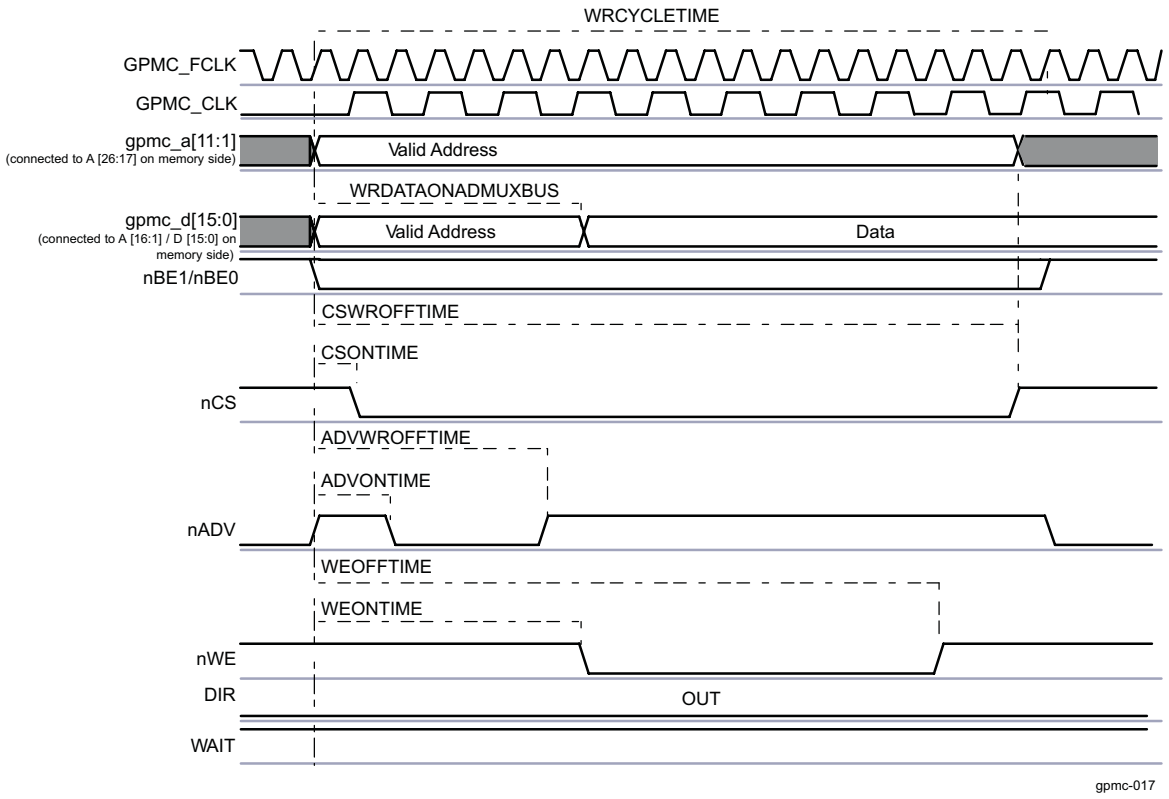
- Direction signal DIR:

  DIR goes from OUT to IN at the same time as nOE assertion.

After a read operation, if no other access (read or write) is pending, the data bus is driven with the previous read value. See Section 11.1.5.3.10, *Bus Keeping Support*.

11.1.5.10.2 Synchronous Single Write

Figure 11-17. Synchronous Single Write on an Address/Data-Multiplexed Device



Note: The WAIT signal is active low.

When the GPMC generates a write access to an address/data-multiplexed device, it drives the data bus until WRDATAONADMUXBUS time (GPMC_CONFIG6_i[19:16]).

The GPMC.GPMC_CONFIG1_i register settings (i = 0 to 7) are as follows:

- WRITEMULTIPLE bit at 0 (write single access)
- WRITETYPE bit at 1 (write synchronous)
- MUXADDDATA bit at 1 (address/data-multiplexed device)

Address bits [16:1] are placed on the address/data bus at cycle-start time, and the remaining address bits [26:17] are placed on the address bus.

The address phase ends at WRDATAONADMUXBUS.

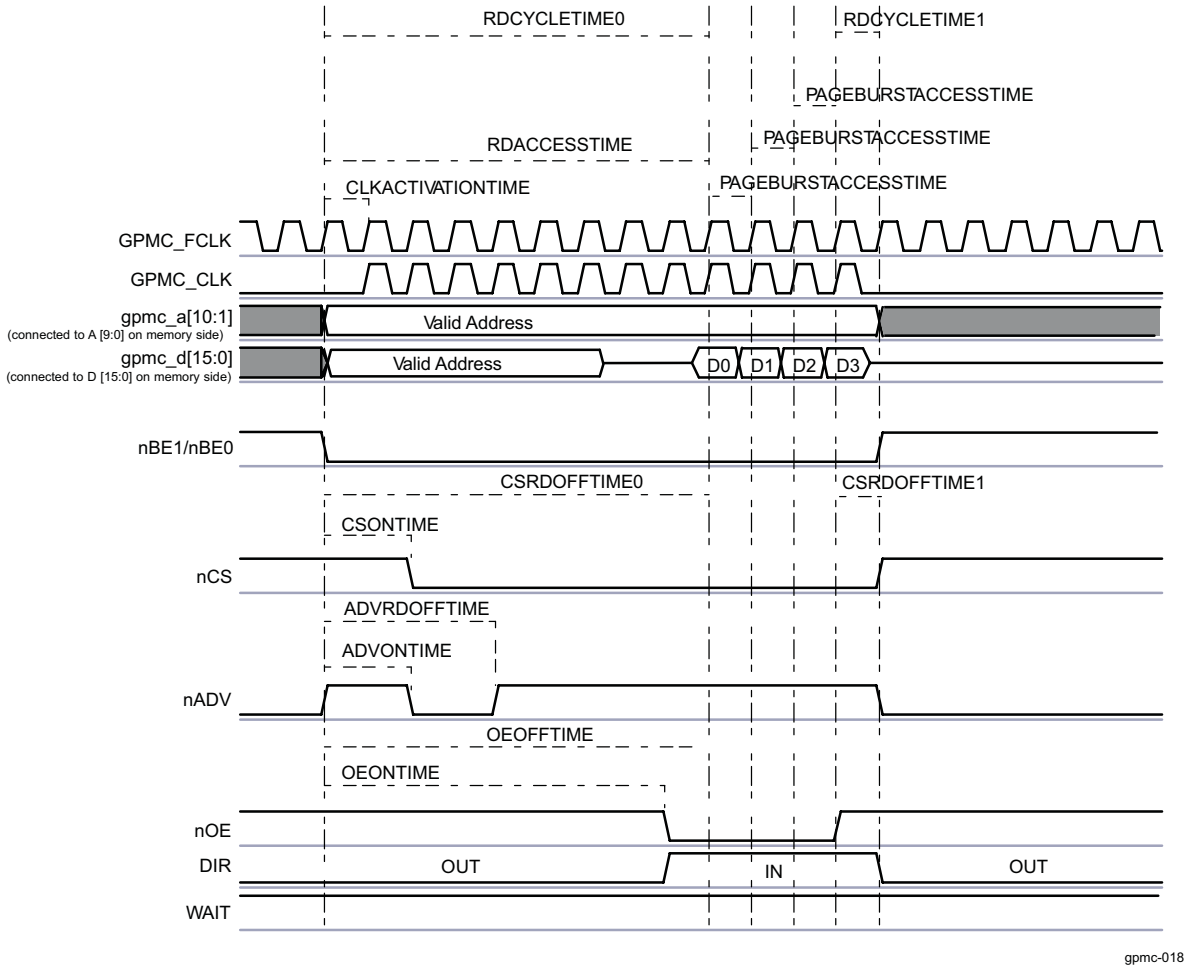
The nCS, nADV, and nWE signals are controlled in the same way as nonmultiplexed accesses.

First data of the burst is driven on the address/data bus at WRDATAONADMUXBUS time.

11.1.5.10.3 Synchronous Multiple (Burst) Read (4-, 8-, 16-Word16 Burst With Wraparound Capability)

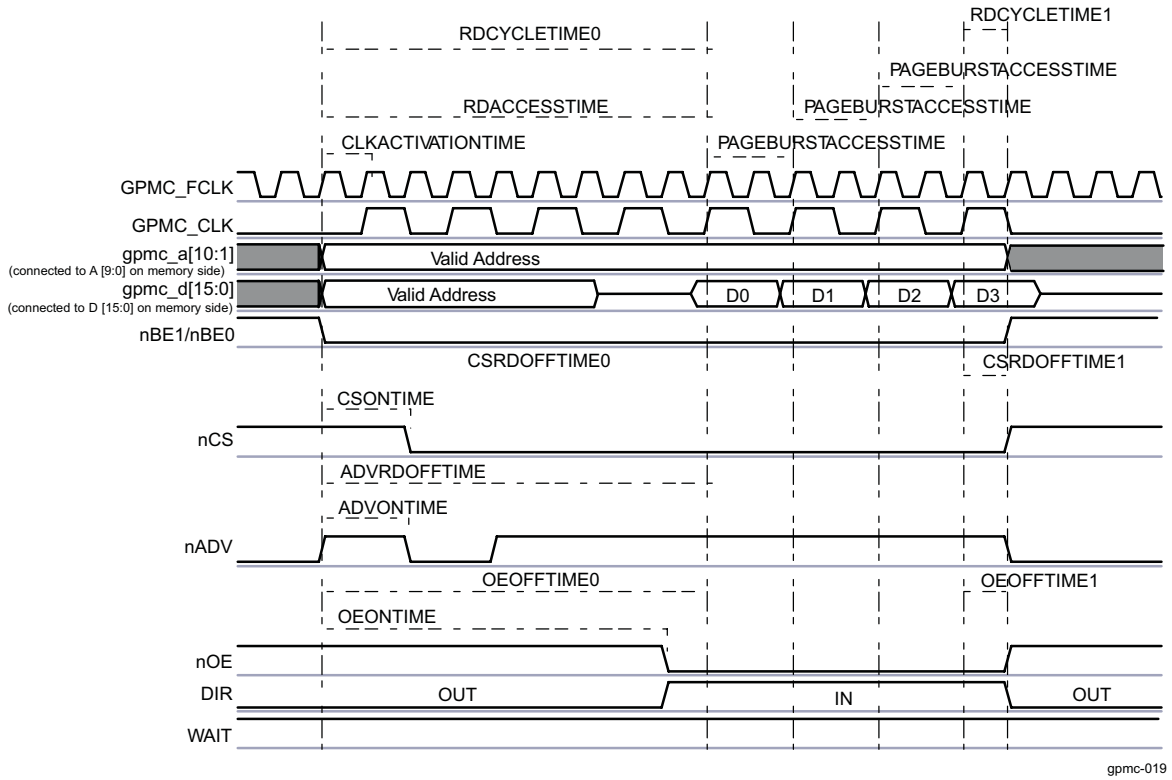
Figure 11-18 and Figure 11-19 show a synchronous multiple read operation with GPMCFCLKDivider equal to 0 and 1, respectively.

Figure 11-18. Synchronous Multiple (Burst) Read (GPMCFCLKDIVIDER = 0)



Note: The WAIT signal is active low.

Figure 11-19. Synchronous Multiple (Burst) Read (GPMCFCLKDIVIDER = 1)



NOTE: The WAIT signal is active low.

In the following section i stands for the chip-select number, i = 0 to 7.

- GPMC.GPMC_CONFIG1_i register settings:
  - READMULTIPLE bit at 1 (read multiple access)
  - READTYPE bit at 1 (read synchronous)
  - MUXADDDATA bit at 0 (nonaddress/data-multiplexed device)

When RDACCESSTIME completes, control-signal timings are frozen during the multiple data transactions, corresponding to PAGEBURSTACCESSTIME multiplied by the number of remaining data transactions.

- Chip-select signal nCS:
  - nCS assertion time is controlled by the GPMC.GPMC_CONFIG2_i[3:0] CSONTIME field and ensures address setup time to nCS assertion.
  - nCS deassertion time is controlled by the GPMC.GPMC_CONFIG2_i[12:8] CSRDOFFTIME field and ensures address hold time to nCS deassertion.
- Address valid signal nADV:
  - nADV assertion time is controlled by the GPMC.GPMC_CONFIG3_i[3:0] ADVONTIME field.
  - nADV deassertion time is controlled by the GPMC.GPMC_CONFIG3_i[12:8] ADVRDOFFTIME field.
- Output enable signal nOE:
  - nOE assertion indicates a read cycle.
  - nOE assertion time is controlled by the GPMC.GPMC_CONFIG4_i[3:0] OEONTIME field.
  - nOE deassertion time is controlled by the GPMC.GPMC_CONFIG4_i[12:8] OEOFFTIME field.
- Initial latency for the first read data is controlled by GPMC.GPMC_CONFIG5_i[20:16] RDACCESSTIME or by monitoring the WAIT signal.
- Successive read data are provided by the memory device each one or two GPMC_CLK cycles. The PAGEBURSTACCESSTIME parameter must be set accordingly with GPMCFCLKDIVIDER and the memory-device internal configuration.

Depending on the device page length, the GPMC can control device page crossing during a new burst request and purposely insert initial latency.

- Total access time (RDCYCLETIME) corresponds to RDACCESSTIME plus the address hold time from nCS deassertion, plus the time from RDACCESSTIME to CSRDOFFTIME.
  - RDCYCLETIME is defined in the GPMC.GPMC_CONFIG5_i register.
  - In Figure 11-19, the RDCYCLETIME programmed value equals RDCYCLETIME0 + RDCYCLETIME1.
- Direction signal DIR:

  DIR goes from OUT to IN at the same time as nOE assertion.

After a read operation, if no other access (read or write) is pending, the data bus is driven with the previous read value. See Section 11.1.5.3.10, *Bus Keeping Support*.

- Burst wraparound

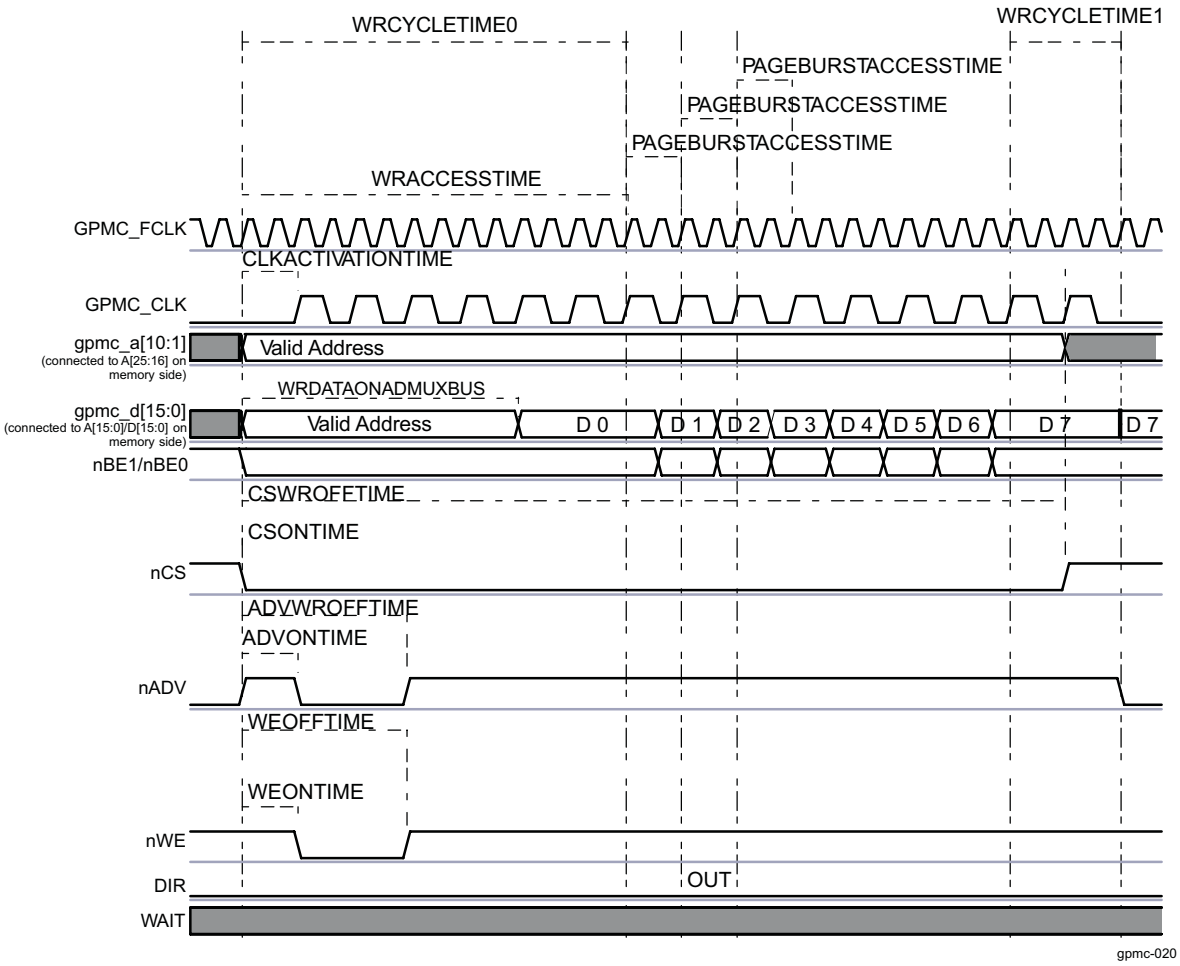
  The GPMC.GPMC_CONFIG1_i[31] WRAPBURST bit allows a 4-, 8-, or 16-Word16 linear burst access to wrap within its burst-length boundary.

11.1.5.10.4 Synchronous Multiple (Burst) Write

Burst write mode provides synchronous single or consecutive accesses.

Figure 11-20 shows a synchronous multiple (burst) write.

Figure 11-20. Synchronous Multiple (Burst) Write



NOTE: The WAIT signal is active low.

In the following section i stands for the chip-select number, i = 0 to 7.

- GPMC.GPMC_CONFIG1_i register settings:
  - WRITEMULTIPLE bit at 1 (write multiple access)
  - WRITETYPE bit at 1 (write synchronous)
  - MUXADDDATA bit at 1 (address/data-multiplexed device)

When WRACCESSTIME completes, control-signal timings are frozen during the multiple data transactions, corresponding to the PAGEBURSTACCESSTIME multiplied by the number of remaining data transactions.

- Chip-select signal nCS:
  - nCS assertion time is controlled by the GPMC.GPMC_CONFIG2_i[3:0] CSONTIME field and ensures address setup time to nCS assertion.
  - nCS deassertion time controlled by the GPMC.GPMC_CONFIG2_i[20:16] CSWROFFTIME field and ensures address hold time to nCS deassertion.
- Address valid signal nADV:
  - nADV assertion time is controlled by the GPMC.GPMC_CONFIG3_i[3:0] ADVONTIME field.
  - nADV deassertion time is controlled by the GPMC.GPMC_CONFIG3_i[20:16] ADVWROFFTIME field.
- Write enable signal nWE:
  - nWE assertion indicates a write cycle.
  - nWE assertion time is controlled by the GPMC.GPMC_CONFIG4_i[19:16] WEONTIME field.
  - nWE deassertion time is controlled by the GPMC.GPMC_CONFIG4_i[28:24] WEOFFTIME field.

Note: The nWE falling edge must not be used to control the time when the burst first data is driven in the address / data bus because some new devices require the nWE signal at low during the address phase.

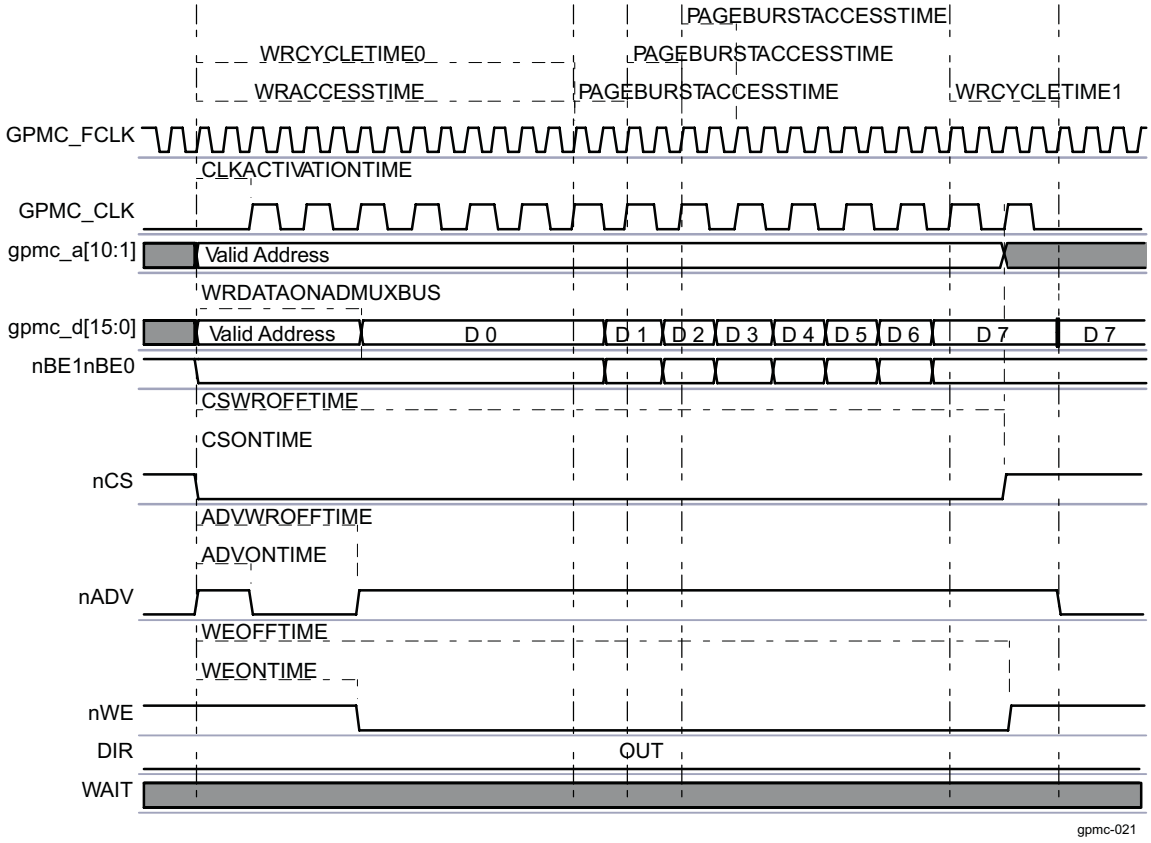
- First write data is driven by the GPMC at WRDATAONADMUXBUS (GPMC_CONFIG6_i[19:16]), when in address/data mux configuration. The next write data of the burst is driven on the bus at WRACCESSTIME + 1 during PAGEBURSTACCESSTIME GPMC_FCLK cycles. The last data of the synchronous burst write is driven until WRCYCLETIME completes.
  - WRACCESSTIME is defined in the GPMC.GPMC_CONFIG5_i register.
  - The PAGEBURSTACCESSTIME parameter must be set accordingly with GPMCFCLKDIVIDER and the memory-device internal configuration.
- Total access time (WRCYCLETIME) corresponds to WRACCESSTIME plus the address hold time from nCS deassertion, plus time from WRACCESSTIME to CSWROFFTIME.
  - WRCYCLETIME is defined in the GPMC.GPMC_CONFIG5_i register.
  - In Figure 11-20, the WRCYCLETIME programmed value equals WRCYCLETIME0 + WRCYCLETIME1.
- Direction signal DIR:

  DIR is OUT during the entire access.

After a write operation, if no other access (read or write) is pending, the data bus keeps the previous value. See Section 11.1.5.3.10, *Bus Keeping Support*.

Figure 11-21 shows the same synchronous burst write access when the chip-select is configured in address/data-multiplexed mode.

Figure 11-21. Synchronous Multiple Write (Burst Write) in Address/Data-Multiplexed Mode



The first data of the burst is driven on the a/d bus at GPMC_CONFIG6_i[19:16] WRDATAONADMUXBUS.

11.1.5.11 pSRAM Basic Programming Model

pSRAM devices are SRAM-pin-compatible low-power memories that contain a self-refreshed DRAM memory array. These devices can be accessed by the GPMC with the GPMC.GPMC_CONFIG1_i[11:10] DEVICETYPE field (i = 0 to 7).

The pSRAM devices support the following operations:

- Asynchronous single read
- Asynchronous page read
- Asynchronous single write
- Synchronous single read and write
- Synchronous burst read
- Synchronous burst write

pSRAM devices must be powered up and initialized in a predefined manner according to the specifications of the attached device.

pSRAM devices can be programmed to use either mode: fixed or variable latency. pSRAM devices can either automatically schedule autorefresh operations, which force the GPMC to use its WAIT signal capability when read or write operations occur during an internal self-refresh operation, or pSRAM devices automatically include the autorefresh operation in the access time. These devices do not require additional WAIT signal capability or a minimum nCS high pulse width between consecutive accesses to ensure that the correct internal refresh operation is scheduled.

In both pSRAM cases, the GPMC configuration for this chip-select must be set according to the specifications of the attached device.

11.1.5.12 Error Handling

When an error occurs in the GPMC, the error information is stored in the GPMC.GPMC_ERR_TYPE register and the address of the illegal access is stored in the GPMC.GPMC_ERR_ADDRESS register. The GPMC only keeps the first error abort information until the GPMC.GPMC_ERR_TYPE register is reset. Subsequent accesses that cause errors are not logged until the error is cleared by hardware with the GPMC.GPMC_ERR_TYPE[0] ERRORVALID bit.

- ERRORNOTSUPPADD occurs when an incoming system request address decoding does not match any valid chip-select region, or if two chip-select regions are defined as overlapped, or if a register file access is tried outside the valid address range of 1 Kbyte.
- ERRORNOTSUPPMCMD occurs when an unsupported command request is decoded at the L3 interconnect interface.
- ERRORTIMEOUT: A time-out mechanism prevents the system from hanging. The start value of the 9-bit time-out counter is defined in the GPMC.GPMC_TIMEOUT_CONTROL register and enabled with the GPMC.GPMC_TIMEOUT_CONTROL[0] TIMEOUTENABLE bit. When enabled, the counter starts at start-cycle time until it reaches 0 and data is not responded to from memory, then a time-out error occurs. When data are sent from memory, this counter is reset to its start value. With multiple accesses (asynchronous page mode or synchronous burst mode), the counter is reset to its start value for each data access within the burst.

The GPMC does not generate interrupts on these errors. True abort to the MPU or interrupt generation is handled at interconnect level.

11.1.5.13 Boot Configuration

See Section 11.1.3.3.2, *GPMC CS0 Default Configuration at IC Reset*.

11.1.5.14 NAND Device Basic Programming Model

NAND (8-bit and 16-bit) memory devices using a classical NAND asynchronous address/data-multiplexing scheme can be supported on any chip-select with the appropriate asynchronous configuration settings.

As for any other type of memory compatible with the GPMC interface, accesses to a chip-select allocated to a NAND device can be interleaved with accesses to chip-selects allocated to other external devices. This interleaved capability limits the system to chip enable dont care NAND devices since the chip-select allocated to the NAND device has to be de-asserted if accesses to other chip-selects are requested.

11.1.5.14.1 NAND Memory Device in Byte or Word16 Stream Mode

NAND devices require correct command and address programming before data array read or write accesses. The GPMC does not include specific hardware to translate a random address system request into a NAND-specific multiphase access. In that sense, GPMC NAND support, as opposed to random memory-map device support, is data-stream-oriented (byte or Word16).

The GPMC NAND programming model relies on a software driver for address and command formatting with the correct data address pointer value according to the block and page structure. Because of NAND structure and protocol interface diversity, the GPMC does not support automatic command and address phase programming, and software drivers must access the NAND device ID to ensure that correct command and address formatting are used for the identified device.

NAND device data read and write accesses are achieved through an asynchronous read or write access. The associated chip-select signal timing control must be programmed according to the NAND device timing specification.

Any chip-select region can be qualified as a NAND region to constrain the nADV/ALE signal as Address Latch Enable (ALE active high, default state value at low) during address program access, and the nBE0/CLE signal as Command Latch Enable (CLE active high, default state value at low) during command program access. GPMC address lines are not used (the previous value is not changed) during NAND access.

11.1.5.14.1.1 Chip-Select Configuration for NAND Interfacing in Byte or Word Stream Mode

The GPMC.GPMC_CONFIG7_i register (where i = 0 to 7) associated with a NAND device region interfaced in byte or word stream mode can be initialized with a minimum size of 16 Mbytes, because any address location in the chip-select memory region can be used to access a NAND data array. The NAND Flash protocol specifies an address sequence where address bits are passed through the data bus in a series of write accesses with the ALE pin asserted. After this address phase, all operations are streamed and the system requests address is irrelevant.

CAUTION

To allow correct command, address, and data-access controls, the GPMC.GPMC_CONFIG1_i register associated with a NAND device region must be initialized in asynchronous read and write modes with the parameters listed in Table 11-4. Failure to comply with these settings corrupts the NAND interface protocol.

Table 11-4. Chip-Select Configuration for NAND Interfacing

Bit Field	Register	Value	Comments
WRAPBURST	GPMC_CONFIG1_i[31] (1)	0	No wrap
READMULTIPLE	GPMC_CONFIG1_i[30]	0	Single access
READTYPE	GPMC_CONFIG1_i[29]	0	Asynchronous mode
WRITEMULTIPLE	GPMC_CONFIG1_i[28]	0	Single access
WRITETYPE	GPMC_CONFIG1_i[27]	0	Asynchronous mode
CLKACTIVATIONTIME	GPMC_CONFIG1_i[26:25]	00	
ATTACHEDDEVICEPAGELENGTH	GPMC_CONFIG1_i[24:23]	Don't care	Single-access mode
WAITREADMONITORING	GPMC_CONFIG1_i[22]	0	Wait not monitored by GPMC access engine
WAITWRITEMONITORING	GPMC_CONFIG1_i[21]	0	Wait not monitored by GPMC access engine
WAITMONITORINGTIME	GPMC_CONFIG1_i[19:18]	Don't care	Wait not monitored by GPMC access engine
WAITPINSELECT	GPMC_CONFIG1_i[17:16]		Select which wait is monitored by edge detectors
DEVICESIZE	GPMC_CONFIG1_i[13:12]	0b00 or 0b01	8- or 16-bit interface
DEVICETYPE	GPMC_CONFIG1_i[11:10]	0b10	NAND device in stream mode
MUXADDDATA	GPMC_CONFIG1_i[9]	0	Nonmultiplexed mode
TIMEPARAGRANULARITY	GPMC_CONFIG1_i[4]	0	Timing achieved with best GPMC clock granularity
GPMCFCLKDIVIDER	GPMC_CONFIG1_i[1:0]	Don't care	Asynchronous mode

(1) i = 0 to 7

The GPMC.GPMC_CONFIG1_i to GPMC.GPMC_CONFIG4_i register (where i = 0 to 7) associated with a NAND device region must be initialized with the correct control-signal timing value according to the NAND device timing parameters.

11.1.5.14.1.2 NAND Device Command and Address Phase Control

NAND devices require multiple address programming phases. The CPU software driver is responsible for issuing the correct number of command and address program accesses, according to the device command set and the device address-mapping scheme.

NAND device-command and address-phase programming is achieved through write requests to the GPMC.GPMC_NAND_COMMAND_i and GPMC.GPMC_NAND_ADDRESS_i register locations (i = 0 to 7) with the correct command and address values. These locations are mapped in the associated chip-select register region. The associated chip-select signal timing control must be programmed according to the NAND device timing specification.

Command and address values are not latched during the access and cannot be read back at the register location.

- Only write accesses must be issued to these locations, but the GPMC does not discard any read access. Accessing a NAND device with nOE and CLE or ALE asserted (read access) can produce undefined results.
- Write accesses to the GPMC.GPMC_NAND_COMMAND_i register location and to the GPMC.GPMC_NAND_ADDRESS_i register location must be posted for faster operations (i = 0 to 7). The GPMC.GPMC_CONFIG[0] NANDFORCEPOSTEDWRITE bit enables write accesses to these locations as posted, even if they are defined as nonposted.

A write buffer is used to store write transaction information before the external device is accessed:

- Up to eight consecutive posted write accesses can be accepted and stored in the write buffer.
- For nonposted write, the pipeline is one deep.
- An GPMC.GPMC_STATUS[0] EMPTYWRITEBUFFERSTATUS bit stores the empty status of the write buffer.

GPMC.GPMC_NAND_COMMAND_i and GPMC.GPMC_NAND_ADDRESS_i (i = 0 to 7) are Word32 locations, which means any Word32 or Word16 access is split into 4- or 2-byte accesses if an 8-bit wide NAND device is attached. For multiple-command phase or multiple-address phase, the software driver can use Word32 or Word16 access to these registers, but it must account for the splitting and little-endian ordering scheme. When only one byte command or address phase is required, only byte write access to GPMC.GPMC_NAND_COMMAND_i and GPMC.GPMC_NAND_ADDRESS_i can be used, and any of the four byte locations of the registers are valid.

The same applies to a GPMC.GPMC_NAND_COMMAND_i and a GPMC.GPMC_NAND_ADDRESS_i (i = 0 to 7) Word32 write access to a 16-bit wide NAND device (split into two Word16 accesses). In the case of a Word16 write access, the MSByte of the Word16 value must be set according to the NAND device requirement (usually 0). Either Word16 location or any one of the four byte locations of the registers is valid.

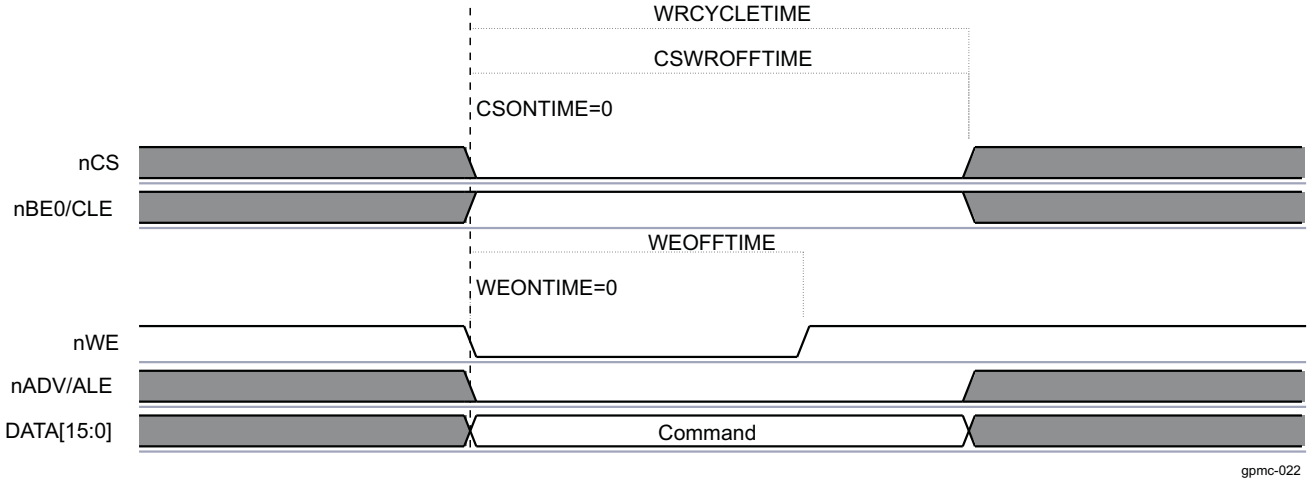
11.1.5.14.1.3 Command Latch Cycle

Writing data at the GPMC.GPMC_NAND_COMMAND_i location (i = 0 to 7) places the data as the NAND command value on the bus, using a regular asynchronous write access.

- nCE is controlled by the CSONTIME and CSWROFFTIME timing parameters.
- CLE is controlled by the ADVONTIME and ADVWROFFTIME timing parameters.
- nWE is controlled by the WEONTIME and WEOFFTIME timing parameters.
- ALE and nRE (nOE) are maintained inactive.

Figure 11-22 shows the NAND command latch cycle.

Figure 11-22. NAND Command Latch Cycle



Note: CLE is shared with the nBE0 output signal and has an inverted polarity from BE0. The NAND qualifier deals with this. During the asynchronous NAND data access cycle, nBE0 (also nBE1) must not toggle, because it is shared with CLE.

NAND flash memories do not use byte enable signals.

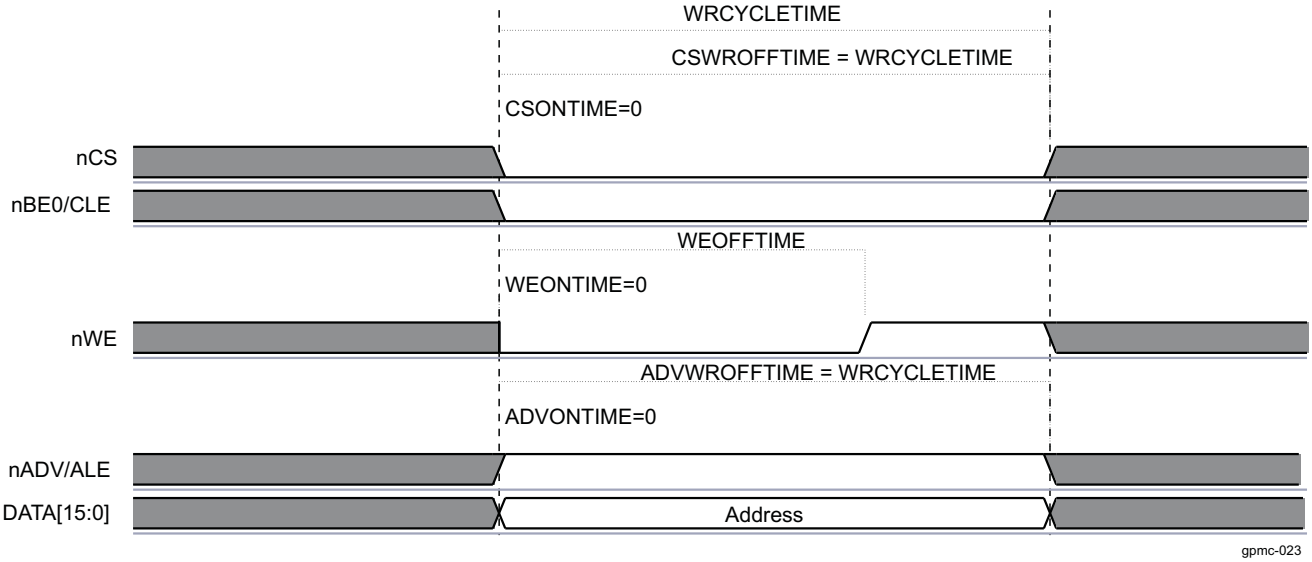
11.1.5.14.1.4 Address Latch Cycle

Writing data at the GPMC.GPMC_NAND_ADDRESS_i location (i = 0 to 7) places the data as the NAND partial address value on the bus, using a regular asynchronous write access.

- nCS is controlled by the CSONTIME and CSWROFFTIME timing parameters.
- ALE is controlled by the ADVONTIME and ADVWROFFTIME timing parameters.
- nWE is controlled by the WEONTIME and WEOFFTIME timing parameters.
- CLE and nRE (nOE) are maintained inactive.

Figure 11-23 shows the NAND address latch cycle.

Figure 11-23. NAND Address Latch Cycle



Note: ALE is shared with the nADV output signal and has an inverted polarity from ADV. The NAND qualifier deals with this. During the asynchronous NAND data access cycle, ALE is kept stable.

11.1.5.14.1.5 NAND Device Data Read and Write Phase Control in Stream Mode

NAND device data read and write accesses are achieved through a read or write request to the chip-select-associated memory region at any address location in the region or through a read or write request to the GPMC.GPMC_NAND_DATA_i location (i = 0 to 7) mapped in the chip-select-associated control register region. GPMC.GPMC_NAND_DATA_i is not a true register, but an address location to enable nRE or nWE signal control. The associated chip-select signal timing control must be programmed according to the NAND device timing specification.

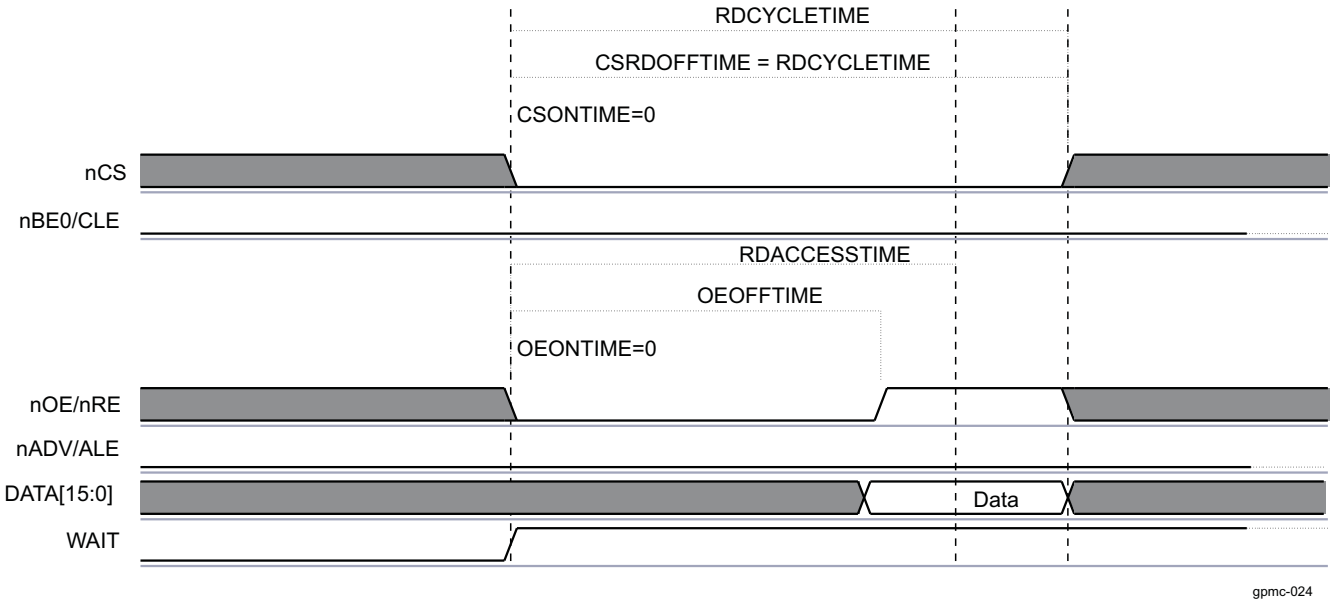
Reading data from the GPMC.GPMC_NAND_DATA_i location or from any location in the associated chip-select memory region activates an asynchronous read access.

- nCS is controlled by the CSONTIME and CSRDOFFTIME timing parameters.
- nRE is controlled by the OEONTIME and OEOFFTIME timing parameters.
- To take advantage of nRE high-to-data invalid minimum timing value, the RDACCESSTIME can be set so that data are effectively captured after nRE deassertion. This allows optimization of NAND read access cycle time completion. For optimal timing parameter settings, see the NAND device and OMAP IC timing parameters.

ALE, CLE, and nWE are maintained inactive.

Figure 11-24 shows the NAND data read cycle.

Figure 11-24. NAND Data Read Cycle

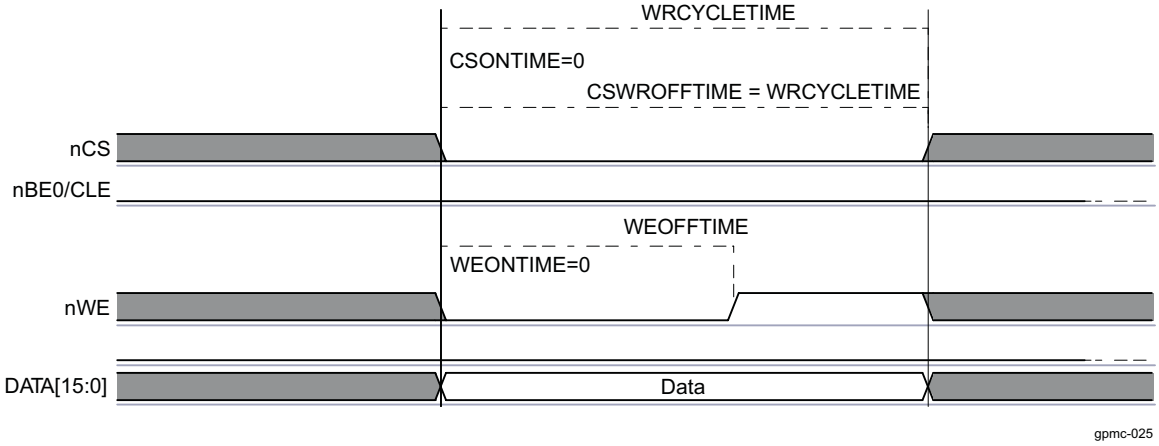


Writing data to the GPMC.GPMC_NAND_DATA_i location or to any location in the associated chip-select memory region activates an asynchronous write access.

- nCS is controlled by the CSONTIME and CSWROFFTIME timing parameters.
- nWE is controlled by the WEONTIME and WEOFFTIME timing parameters.
- ALE, CLE, and nRE (nOE) are maintained inactive.

Figure 11-25 shows the NAND data write cycle.

Figure 11-25. NAND Data Write Cycle



11.1.5.14.1.6 NAND Device General Chip-Select Timing Control Requirement

For most NAND devices, read data access time is dominated by nCS-to-data-valid timing and has faster nRE-to-data-valid timing. Successive accesses with nCS deassertions between accesses are affected by this timing constraint. Because accesses to a NAND device can be interleaved with other chip-select accesses, there is no certainty that nCS always stays low between two accesses to the same chip-select. Moreover, an nCS deassertion time between the same chip-select NAND accesses is likely to be required as follows: the nCS deassertion requires programming CYCLETIME and RDACCESSTIME according to the nCS-to-data-valid critical timing.

To get full performance from NAND read and write accesses, the prefetch engine can dynamically reduce RDCYCLETIME, WRCYCLETIME, RDACCESSTIME, WRACCESSTIME, CSRDOFFTIME, CSWROFFTIME, ADVRDOFFTIME, ADVWROFFTIME, OEOFFTIME, and WEOFFTIME on back-to-back NAND accesses (to the same memory) and suppress the minimum nCS high pulse width between accesses. For more information about optimal prefetch engine access, see Section 11.1.5.14.4, *Prefetch and Write-Posting Engine*.

Some NAND devices require minimum write-to-read idle time, especially for device-status read accesses following status-read command programming (write access). If such write-to-read transactions are used, a minimum nCS high pulse width must be set. For this, CYCLE2CYCLESAMECSEN and CYCLE2CYCLEDELAY must be set according to the appropriate timing requirement to prevent any timing violation.

NAND devices usually have an important nRE high to data bus in tristate mode. This requires a bus turnaround setting (BUSTURNAROUND = 1) so that the next access to a different chip-select is delayed until the BUSTURNAROUND delay completes. Back-to-back NAND read accesses to the same NAND flash are not affected by the programmed bus turnaround delay.

11.1.5.14.1.7 Read and Write Access Size Adaptation

11.1.5.14.1.7.1 8-Bit Wide NAND Device

Host Word16 and Word32 read and write access requests to a chip-select associated with an 8-bit wide NAND device are split into successive read and write byte accesses to the NAND memory device. Byte access is ordered according to little-endian organization. A NAND 8-bit wide device must be interfaced on the D0D7 interface bus lane. GPMC data accesses are justified on this bus lane when the chip-select is associated with an 8-bit wide NAND device.

11.1.5.14.1.7.2 16-Bit Wide NAND Device

Host Word32 read and write access requests to a chip-select associated with a 16-bit wide NAND device are split into successive read and write Word16 accesses to the NAND memory device. Word16 access is ordered according to little-endian organization.

Host byte read and write access requests to a 16-bit wide NAND device are completed as 16-bit accesses on the device itself, because there is no byte-addressing capability on 16-bit wide NAND devices. This means that the NAND device address pointer is incremented on a Word16 basis and not on a byte basis. For a read access, only the requested byte is given back to the host, but the remaining byte is not stored or saved by the GPMC, and the next byte or Word16 read access gets the next Word16 NAND location. For a write access, the invalid byte part of the Word16 is driven to FF, and the next byte or Word16 write access programs the next Word16 NAND location.

Generally, byte access to a 16-bit wide NAND device should be avoided, especially when ECC calculation is enabled. 8-bit or 16-bit ECC-based computations are corrupted by a byte read to a 16-bit wide NAND device, because the nonrequested byte is considered invalid on a read access (not captured on the external data bus; FF is fed to the ECC engine) and is set to FF on a write access.

Host requests (read/write) issued in the chip-select memory region are translated in successive single or split accesses (read/write) to the attached device. Therefore, incrementing 32-bit burst requests are translated in multiple 32-bit sequential accesses following the access adaptation of the 32-bit to 8- or 16-bit device.

11.1.5.14.2 NAND Device-Ready Pin

The NAND memory device provides a ready pin to indicate data availability after a block/page opening and to indicate that data programming is complete. The ready pin can be connected to one of the four WAIT GPMC input pins; data read accesses must not be tried when the ready pin is sampled inactive (device is not ready) even if the associated chip-select WAITREADMONITORING bit field is set. The duration of the NAND device busy state after the block/page opening is so long (up to 50 µs) that accesses occurring when the ready pin is sampled inactive can stall GPMC access and eventually cause a system time-out.

Note: If a read access to a NAND flash is done using the wait monitoring mode, the device is blocked during a page opening, and so is the GPMC. If the correct settings are used, other chip-selects can be used while the memory processes the page opening command.

To avoid a time-out caused by a block/page opening delay in NAND flash, disable the wait pin monitoring for read and write accesses (that is, set the GPMC.GPMC_CONFIG1_i[21] WAITWRITEMONITORING and GPMC.GPMC_CONFIG1_i[22] WAITREADMONITORING bits to 0, where i = 0 to 7), and use one of the following methods instead:

- Use software to poll the WAITnSTATUS bit (n = 0 to 3) of the GPMC_STATUS register.
- Configure an interrupt that is generated on the WAIT signal change (through the GPMC.GPMC_IRQENABLE register bits[11:8]).

Even if the READWAITMONITORING bit is not set, the external memory nR/B pin status is captured in the programmed WAIT bit in the GPMC_STATUS register.

The READWAITMONITORING bit method must be used fo rmemories other than NAND flash, if they require the use of a WAIT signal.

11.1.5.14.2.1 Ready Pin Monitored by Software Polling

The ready signal state can be monitored through the GPMC.GPMC_STATUS WAITxSTATUS bit (x = 0 to 3). The software must monitor the ready pin only when the signal is declared valid. See the NAND device timing parameters to set the correct software temporization to monitor ready only after the invalid window is complete from the last read command written to the NAND device.

11.1.5.14.2.2 Ready Pin Monitored by Hardware Interrupt

Each gpmc_wait input pin can generate an interrupt when a wait-to-no-wait transition is detected. Depending on whether the GPMC.GPMC_CONFIG WAITxPINPOLARITY bits (x = 0 to 3) is active low or active high, the wait-to-no-wait transition is a low-to-high external WAIT signal transition or a high-to-low external WAIT signal transition, respectively.

The wait transition pin detector must be cleared before any transition detection. This is done by writing 1

to the WAITxEDGEDETECTIONSTATUS bit (x = 0 to 3) of the GPMC.GPMC_IRQSTATUS register according to the gpmc_wait pin used for the NAND device-ready signal monitoring. To detect a wait-to-no-wait transition, the transition detector requires a wait active time detection of a minimum of two GPMC_FCLK cycles. Software must incorporate precautions to clear the wait transition pin detector before wait (busy) time completes.

A wait-to-no-wait transition detection can issue a GPMC interrupt if the WAITxEDGEDETECTIONENABLE bit in the GPMC.GPMC_IRQENABLE register is set and if the WAITxEDGEDETECTIONSTATUS bit field in the GPMC.GPMC_IRQSTATUS register is set.

The WAITMONITORINGTIME field does not affect wait-to-no-wait transition time detection.

It is also possible to poll the WAITxEDGEDETECTIONSTATUS bit field in the GPMC.GPMC_IRQSTATUS register according to the gpmc_wait pin used for the NAND device ready signal monitoring.

11.1.5.14.3 ECC Calculator

The general-purpose memory controller includes an error code correction (ECC) calculator circuitry that enables on-the-fly ECC calculation during data read or data program (that is, write) operations.

The user can choose from two different algorithms with different error correction capabilities: Hamming code (for 1-bit error code correction), and BCH code (for 4- or 8-bit error correction) through the GPMC_ECC_CONFIG[16] ECCALGORITHM bit. Only one ECC context can be active at any given time through the GPMC_ECC_CONFIG[3:1] ECCCS bit. Even if two CSs use different ECC algorithms, one the Hamming code and the other a BCH code, they must define separate ECC contexts, because some of the ECC registers are common to all types of algorithms.

11.1.5.14.3.1 Hamming Code

All references to ECC in this section refer to the 1-bit error correction Hamming code.

The ECC is based on a two-dimensional (row and column) bit parity accumulation known as Hamming code. The parity accumulation is done for a programmed number of bytes or Word16 read from the memory device or written to the memory device in stream mode.

There is no automatic error detection or correction, and it is the software NAND driver responsibility to read the multiple ECC calculation results, compare them to the expected code value, and take the appropriate corrective actions according to the error handling strategy (ECC storage in spare byte, error correction on read, block invalidation).

The ECC engine includes a single accumulation context. It can be allocated to a single designated chip-select at a time and parallel computations on different chip-selects are not possible. Since it is allocated to a single chip-select, the ECC computation is not affected by interleaved GPMC accesses to other chip-selects and devices. The ECC accumulation is sequentially processed in the order of data read from or written to the memory on the designated chip-select. The ECC engine does not differentiate read accesses from write accesses and does not differentiate data from command or status information. It is the software responsibility to make sure only relevant data are passed to the NAND flash memory while the ECC computation engine is active.

The starting NAND page location must be programmed first, followed by an ECC accumulation context reset with an ECC enabling, if required. The NAND device accesses discussed in the following sections must be limited to data read or write until the specified number of ECC calculations is completed.

11.1.5.14.3.1.1 ECC Result Register and ECC Computation Accumulation Size

The GPMC includes up to nine ECC result registers (GPMC.GPMC_ECCj_RESULT, j = 1 to 9) to store ECC computation results when the specified number of bytes or Word16s has been computed.

The ECC result registers are used sequentially; one ECC result is stored in one ECC result register on the list, the next ECC result is stored in the next ECC result register on the list, and so forth, until the last ECC computation. The GPMC.GPMC_ECCj_RESULT register value is valid only when the programmed number of bytes or Word16s has been accumulated, which means that the same number of bytes or Word16s has been read from or written to the NAND device in sequence.

The GPMC.GPMC_ECC_CONTROL[3:0] ECCPOINTER field must be set to the correct value to select the ECC result register to be used first in the list for the incoming ECC computation process. The ECCPointer can be read to determine which ECC register is used in the next ECC result storage for the ongoing ECC computation. The GPMC.GPMC_ECCj_RESULT register value (j = 1 to 9) can be considered valid when ECCPOINTER equals j + 1. When GPMC.GPMC_ECCj_RESULT (where j = 9) is updated, ECCPOINTER is frozen at 10, and ECC computing is stopped (ECCENABLE = 0).

The ECC accumulator must be reset before any ECC computation accumulation process. The GPMC.GPMC_ECC_CONTROL[8] ECCCLEAR bit must be set to 1 (nonpersistent bit) to clear the accumulator and all ECC result registers.

For each ECC result (each GPMC.GPMC_ECCj_RESULT register, j = 1 to 9), the number of bytes or Word16s used for ECC computing accumulation can be selected from between two programmable values.

The ECCjRESULTSIZE bits (j = 1 to 9) in the GPMC.GPMC_ECC_SIZE_CONFIG register select which programmable size value (ECCSIZE0 or ECCSIZE1) must be used for this ECC result (stored in GPMC.GPMC_ECCj_RESULT).

The ECCSIZE0 and ECCSIZE1 fields allow selection of the number of bytes or Word16s used for ECC computation accumulation. Any even values from 2 to 512 are allowed.

Flexibility in the number of ECCs computed and the number of bytes or Word16s used in the successive ECC computations enables different NAND page error-correction strategies. Usually based on 256 or 512 bytes and on 128 or 256 Word16, the number of ECC results required is a function of the NAND device page size. Specific ECC accumulation size can be used when computing the ECC on the NAND spare byte.

For example, with a 2-Kbyte data page 8-bit-wide NAND device, eight ECCs accumulated on 256 bytes can be computed and added to one extra ECC computed on the 24 spare bytes area where the eight ECC results used for comparison and correction with the computed data page ECC are stored. The GPMC then provides nine GPMC.GPMC_ECCj_RESULT registers (j= 1 to 9) to store the results. In this case, ECCSIZE0 is set to 256, and ECCSIZE1 is set to 24; the ECC[1:8]RESULTSIZE bits are set to 0, and the ECC9RESULTSIZE bit is set to 1.

11.1.5.14.3.1.2 ECC Enabling

The GPMC.GPMC_ECC_CONFIG[3:0] ECCCS field selects the allocated chip-select. The GPMC.GPMC_ECC_CONFIG[0] ECCENABLE bit enables ECC computation on the next detected read or write access to the selected chip-select.

The ECCPOINTER, ECCCLEAR, ECCSIZE, ECCjRESULTSIZE (where j = 1 to 9), ECC16B, and ECCCS fields must not be changed or cleared while an ECC computation is in progress.

The ECC accumulator and ECC result register must not be changed or cleared while an ECC computation is in progress.

Table 11-5 describes the ECC enable settings.

Table 11-5. ECC Enable Settings

Bit Field	Register	Value	Comments
ECCCS	GPMC_ECC_CONFIG	0–7	Selects the chip-select where ECC is computed
ECC16B	GPMC_ECC_CONFIG	0/1	Selects column number for ECC calculation
ECCCLEAR	GPMC_ECC_CONTROL	0–7	Clears all ECC result registers
ECCPOINTER	GPMC_ECC_CONTROL	0–7	A write to this bit field selects the ECC result register where the first ECC computation is stored. Set to 1 by default.
ECCSIZE1	GPMC_ECC_SIZE_CONFIG	0x00–0xFF	Defines ECCSIZE1
ECCSIZE0	GPMC_ECC_SIZE_CONFIG	0x00–0xFF	Defines ECCSIZE0
ECCjRESULTSIZE (j from 1 to 9)	GPMC_ECC_SIZE_CONFIG	0/1	Selects the size of ECCn result register
ECCENABLE	GPMC_ECC_CONFIG	1	Enables the ECC computation

11.1.5.14.3.1.3 ECC Computation

The ECC algorithm is a multiple parity bit accumulation computed on the odd and even bit streams extracted from the byte or Word 16 streams. The parity accumulation is split into row and column accumulations, as shown in Figure 11-26 and Figure 11-27. The intermediate row and column parities are used to compute the upper level row and column parities. Only the final computation of each parity bit is used for ECC comparison and correction.

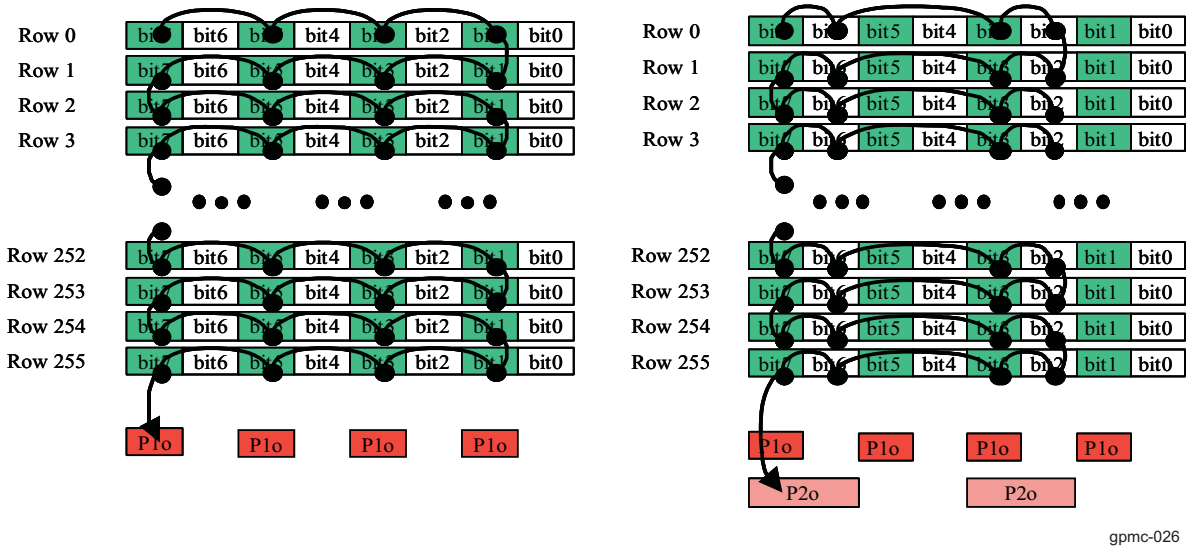
P1o = bit7 XOR bit5 XOR bit3 XOR bit1 on each byte of the data stream
P1e = bit6 XOR bit4 XOR bit2 XOR bit0 on each byte of the data stream

P2o = bit7 XOR bit6 XOR bit3 XOR bit2 on each byte of the data stream
P2e = bit5 XOR bit4 XOR bit1 XOR bit0 on each byte of the data stream

P4o = bit7 XOR bit6 XOR bit5 XOR bit4 on each byte of the data stream
P4e = bit3 XOR bit2 XOR bit1 XOR bit0 on each byte of the data stream

Each column parity bit is XORed with the previous accumulated value.

Figure 11-26. Hamming Code Accumulation Algorithm (1/2)

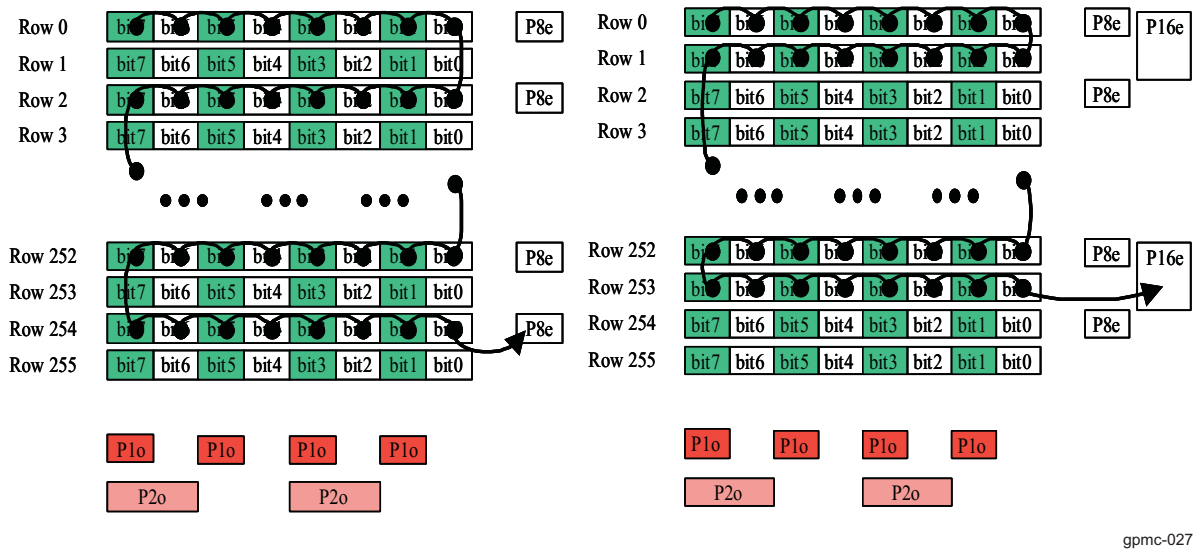


For line parities, the bits of each new data are XORed together, and line parity bits are computed as described below:

P8e = row0 XOR row2 XOR row4 XOR XOR row254
P8o = row1 XOR row3 XOR row5 XOR XOR row255

P16e = row0 XOR row1 XOR row4 XOR row5 XOR XOR row252 XOR row 253
P16o = row2 XOR row3 XOR row6 XOR row7 XOR XOR row254 XOR row 255

Figure 11-27. Hamming Code Accumulation Algorithm (2/2)



Unused parity bits in the result registers are set to 0.

Figure 11-28 shows ECC computation for a 256-byte data stream (read or write). The result includes six column parity bits (P1o-P2o-P4o for odd parities, and P1e-P2e-P4e for even parities) and sixteen row parity bits (P8o-P16o-P32o--P1024o for odd parities, and P8e-P16e-P32e--P1024e for even parities).

Figure 11-28. ECC Computation for a 256-Byte Data Stream (Read or Write)

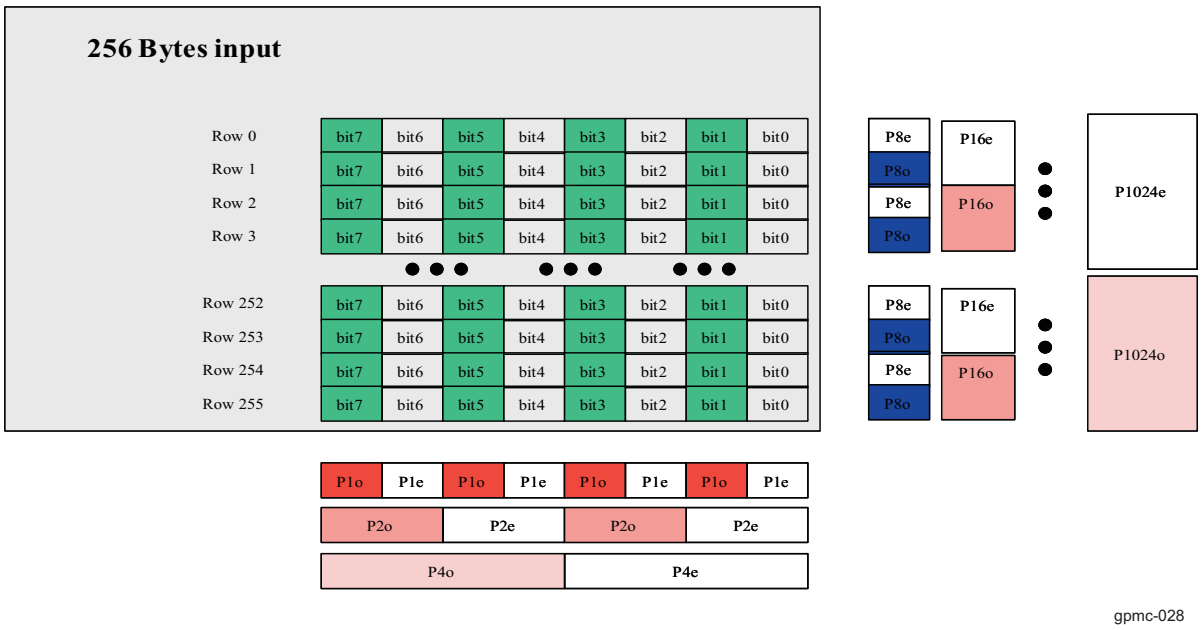
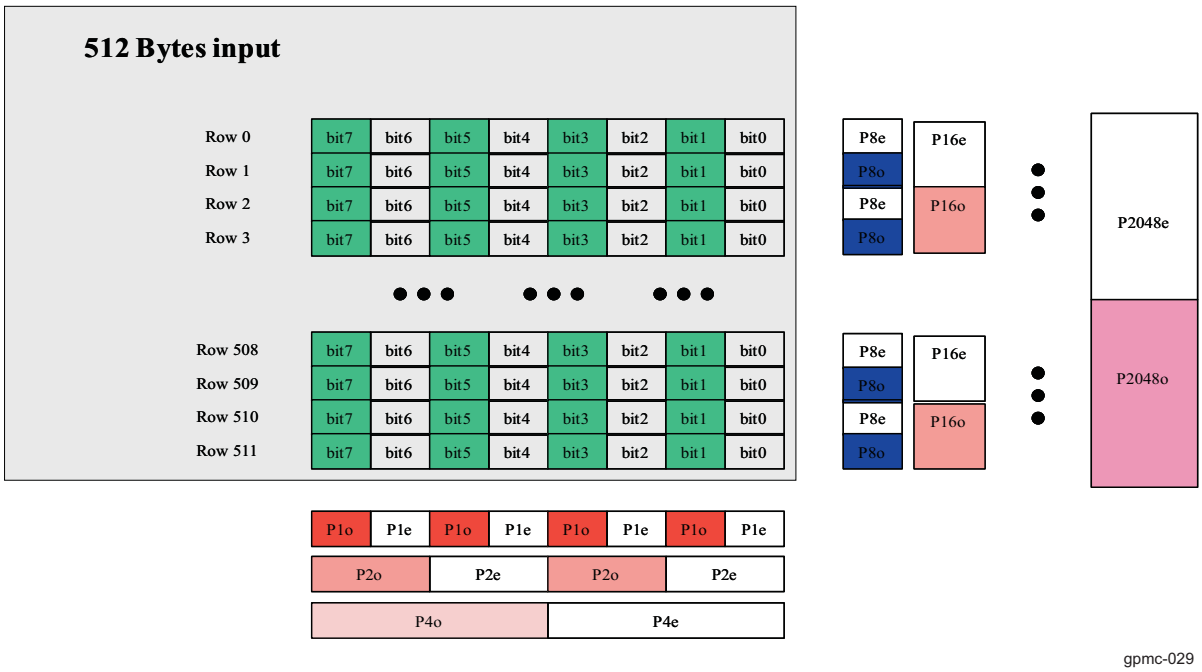


Figure 11-29 shows ECC computation for a 512-byte data stream (read or write). The result includes six column parity bits (P1o-P2o-P4o for odd parities, and P1e-P2e-P4e for even parities) and eighteen row parity bits (P8o-P16o-P32o--P1024o- - P2048o for odd parities, and P8e-P16e-P32e--P1024e- P2048e for even parities).

Figure 11-29. ECC Computation for a 512-Byte Data Stream (Read or Write)



For a 2-Kbytes page, four 512-byte ECC calculations plus one for the spare area are required. Results are stored in the GPMC_ECCj_RESULT registers (j = 1 to 9).

11.1.5.14.3.1.4 ECC Comparison and Correction

To detect an error, the computed ECC result must be XORed with the parity value stored in the spare area of the accessed page.

- If the result of this logical XOR is all 0s, no error is detected and the read data is correct.
- If every second bit in the parity result is a 1, one bit is corrupted and is located at bit address (P2048o, P1024o, P512o, P256o, P128o, P64o, P32o, P16o, P8o, P4o, P2o, P1o). The software must correct the corresponding bit.
- If only one bit in the parity result is 1, it is an ECC error and the read data is correct.

11.1.5.14.3.1.5 ECC Calculation Based on 8-Bit Word

The 8-bit based ECC computation is used for 8-bit wide NAND device interfacing.

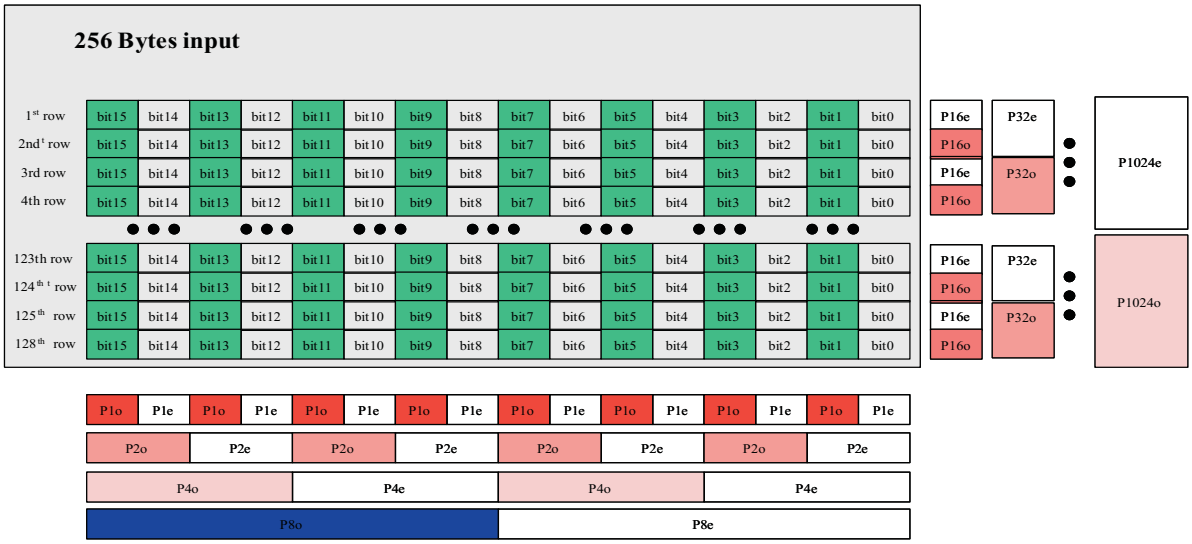
The 8-bit based ECC computation can be used for 16-bit wide NAND device interfacing to get backward compatibility on the error-handling strategy used with 8-bit wide NAND devices. In this case, the 16-bit wide data read from or written to the NAND device is fragmented into 2 bytes. According to little-endian access, the least significant bit (LSB) of the 16-bit wide data is ordered first in the byte stream used for 8-bit based ECC computation.

11.1.5.14.3.1.6 ECC Calculation Based on 16-Bit Word

ECC computation based on a 16-bit word is used for 16-bit wide NAND device interfacing. This ECC computation is not supported when interfacing an 8-bit wide NAND device, and the GPMC.GPMC_ECC_CONFIG[7] ECC16B bit must be set to 0 when interfacing an 8-bit wide NAND device.

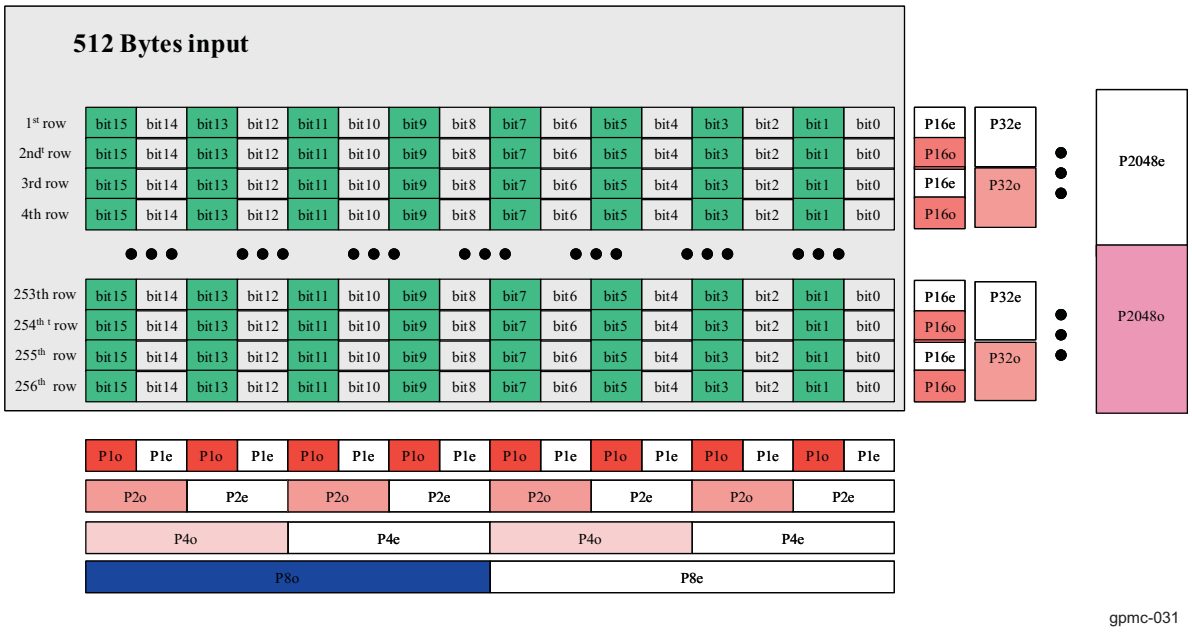
The parity computation based on 16-bit words affects the row and column parity mapping. The main difference is that the odd and even parity bits P8o and P8e are computed on rows for an 8-bit based ECC while there are computed on columns for a 16-bit based ECC. Figure 11-30 and Figure 11-31 show a 128 Word 16 ECC computation scheme and a 256 Word16 ECC computation scheme.

Figure 11-30. 128 Word16 ECC Computation



gpmc-030

Figure 11-31. 256 Word16 ECC Computation



gpmc-031

11.1.5.14.3.2 BCH Code

All references to ECC in this section refer to the 4- or 8-bit error correction BCH code.

11.1.5.14.3.2.1 Requirements

1. Read and write accesses to a NAND flash take place by whole pages, in a predetermined sequence: first, the data byte page itself, then some spare bytes, including the BCH ECC (and other information). The NAND IC can cache a full page, including spares, for read and write accesses.

   Typical page write sequence:

   - Sequential write to NAND cache of main data + spare data, for a page. ECC is calculated on the fly. Calculated ECC may be inserted on the fly in the spares, or replaced by dummy accesses.
   - When the calculated ECC is replaced by dummy accesses, it must be written to the cache in a

second, separate phase. The ECC module is disabled during that time.

- NAND writes its cache line (page) to the array.

Typical page read sequence:

- Sequential read of a page. ECC is calculated on-the-fly.
- ECC module buffers status determines the presence of errors.

2. Accesses to several memories may be interleaved by the GPMC, but only one of those memories can be a NAND using the BCH engine at a time; in other words, only one BCH calculation (for example, for a single page) can be on-going at any time. Note also that the sequential nature of NAND accesses guarantees that the data is always written / read out in the same order. BCH-relevant accesses are selected by the GPMCs chip-select.
3. Each page may hold up to 4 Kbytes of data, spare bytes not included. This means up to 8 × 512-byte BCH messages. Since all the data is written / read out first, followed by the BCH ECC, this means that the BCH engine must be able to hold 8 104-bit remainders or syndromes (or smaller, 52-bit ones) at the same time.

   The BCH module has the capacity to store all remainders internally. After the page start, an internal counter is used to detect the 512-byte sector boundaries. On those boundaries, the current remainder is stored and the divider reset for the next calculation. At the end of the page, the BCH module contains all remainders.
4. NAND access cycles hold 8 or 16 bits of data each (1 or 2 bytes); Each NAND cycle takes at least 4 cycles of the GPMCs internal clock. This means the NAND flash timing parameters must define a RDCYCLETIME and a WRCYCLETIME of at least 4 clock cycles after optimization when using the BCH calculator.
5. The spare area is assumed to be large enough to hold the BCH ECC, that is, to have at least a message of 13 bytes available per 512-byte sector of data. The zone of unused spare area by the ECC may or may not be protected by the same ECC scheme, by extending the BCH message beyond 512 bytes (maximum codeword is 1023-byte long, ECC included, which leaves a lot of space to cover some spares bytes).

11.1.5.14.3.2.2 Memory-Mapping of the BCH Codeword

BCH encoding considers a block of data to protect as a polynomial message M(x). In our standard case, 512 bytes of data (that is, 2^{12} bits = 4096 bits) are seen as a polynomial of degree 2^{12} - 1 = 4095, with parameters ranging from M0 to M4095. For 512 bytes of data, 52 bits are required for 4-bit error correction, and 104 bits are required for 8-bit error correction. The ECC is a remainder polynomial R(x) of degree 103 (or 51, depending on the selected mode). The complete codeword C(x) is the concatenation of M(x) and R(x) as shown in Table 11-6.

Table 11-6. Flattened BCH Codeword Mapping (512 Bytes + 104 Bits)

	Message M(x)			ECC R(x)		
Bit number	M4095	...	M0	R103	...	R0

If the message is extended by the addition of spare bytes to be protected by the same ECC, the principle is still valid. For example, a 3-byte extension of the message gives a polynomial message M(x) of degree ((512 + 3) * 8) - 1 = 4119, for a total of 3+13 = 16 spare bytes of spare, all protected as part of the same codeword.

The message and the ECC bits are manipulated and mapped in the GPMC byte-oriented system. The ECC bits are stored in GPMC_BCH_RESULT0_i, GPMC_BCH_RESULT1_i, GPMC_BCH_RESULT2_i, and GPMC_BCH_RESULT3_i (where i = 0 to 7).

11.1.5.14.3.2.2.1 Memory-Mapping of the Data Message

The data message mapping shall follow the following rules:

- Bit endianness within a byte is little-endian, that is, the bytes LS bit is also the lowest-degree polynomial parameter: a byte b7-b0 (with b0 the LS bit) represents a segment of polynomial b7 * $x^{(7+i)}$ + b6 * $x^{(6+i)}$ + ... + b0 * x^{i}

- The message is mapped in the NAND starting with the highest-order parameters, that is, in the lowest addresses of a NAND page.
- Byte endianness within the NANDs 16-bit words is big endian. This means that the same message mapped in 8- and 16-bit memories has the same content at the same byte address.

Note: The BCH module has no visibility over actual addresses. The most important point is the sequence of data word the BCH sees. However, the NAND page is always scanned incrementally in read and write accesses, which produces the mapping patterns described in the following.

Table 11-7 and Table 11-8 describe the mapping of the same 512-byte vector (typically a BCH message) in the NAND memory space. The byte "address" is only an offset modulo 512 (0x200), because the same page may contain several contiguous 512-byte sectors (BCH blocks). The LSB and MSB are respectively the bits M0 and M(2^{12}-1) of the codeword mapping given previously. In both cases the data vectors are aligned; that is, their boundaries coincide with the RAMs data word boundaries.

Table 11-7. Aligned Message Byte Mapping in 8-bit NAND

Byte Offset	8-Bit Word
0x000	(msb) Byte 511 (0x1FF)
0x001	Byte 510 (0x1FE)
...	...
0x1FF	Byte 0 (0x0) (lsb)

Table 11-8. Aligned Message Byte Mapping in 16-bit NAND

Byte Offset	16-Bit Word MSB	16-Bit Word LSB
0x000	Byte 510 (0x1FE)	(msb) Byte 511 (0x1FF)
0x002	Byte 508 (0x1FC)	Byte 509 (0x1FD)
...	...	...
0x1FE	Byte 0 (0x0)	(lsb) Byte 1 (0x1)

Table 11-9 through Table 11-14 show the mapping in memory of arbitrarily-sized messages, starting on access (byte or 16-bit word) boundaries for more clarity. The message may actually start and stop on arbitrary nibbles. A nibble is a 4-bit entity. The unused nibbles are not discarded; they can still be used by the BCH module, but as part of the next message section (for example, on another sector ECC).

Table 11-9. Aligned Nibble Mapping of Message in 8-bit NAND

Byte Offset	8-Bit Word	
	4-Bit Most Significant Nibble	4-Bit Less Significant Nibble
1	(msb) Nibble S-1	Nibble S-2
2	Nibble S-3	Nibble S-4
...	...	...
S/2 - 2	Nibble 3	Nibble 2
S/2 - 1	Nibble 1	Nibble 0 (lsb)

Table 11-10. Misaligned Nibble Mapping of Message in 8-bit NAND

Byte Offset	8-Bit Word	
	4-Bit Most Significant Nibble	4-Bit Less Significant Nibble
1	(msb) Nibble S-1	Nibble S-2
2	Nibble S-3	Nibble S-4
...	...	...
(S+1)/2 - 2	Nibble 2	Nibble 1
(S+1)/2 - 1	Nibble 0 (lsb)	

Table 11-11. Aligned Nibble Mapping of Message in 16-bit NAND

Byte Offset	16-Bit Word			
	4-Bit Most Significant Nibble		4-Bit Less Significant Nibble	
0	Nibble S-3	Nibble S-4	(msb) Nibble S-1	Nibble S-2
2	Nibble S-7	Nibble S-8	Nibble S-5	Nibble S-6
...	...		...	
S/2 - 4	Nibble 5	Nibble 4	Nibble 7	Nibble 6
S/2 - 2	Nibble 1	Nibble 0 (lsb)	Nibble 3	Nibble 2

Table 11-12. Misaligned Nibble Mapping of Message in 16-bit NAND (1 Unused Nibble)

Byte Offset	16-Bit Word			
	4-Bit Most Significant Nibble		4-bit less significant Nibble	
0	Nibble S-3	Nibble S-4	(msb) Nibble S-1	Nibble S-2
2	Nibble S-7	Nibble S-8	Nibble S-5	Nibble S-6
...	...		...	
(S+1)/2 - 4	Nibble 4	Nibble 3	Nibble 6	Nibble 5
(S+1)/2 - 2	Nibble 0 (lsb)		Nibble 2	Nibble 1

Table 11-13. Misaligned Nibble Mapping of Message in 16-bit NAND (2 Unused Nibbles)

Byte Offset	16-Bit Word			
	4-Bit Most Significant Nibble		4-Bit Less Significant Nibble	
0	Nibble S-3	Nibble S-4	(msb) Nibble S-1	Nibble S-2
2	Nibble S-7	Nibble S-8	Nibble S-5	Nibble S-6
...	...		...	
(S+2)/2 - 4	Nibble 3	Nibble 2	Nibble 5	Nibble 4
(S+2)/2 - 2			Nibble 1	Nibble 0 (lsb)

Table 11-14. Misaligned Nibble Mapping of Message in 16-bit NAND (3 Unused Nibbles)

Byte Offset	16-Bit Word			
	4-Bit Most Significant Nibble		4-Bit Less Significant Nibble	
0	Nibble S-3	Nibble S-4	(msb) Nibble S-1	Nibble S-2
2	Nibble S-7	Nibble S-8	Nibble S-5	Nibble S-6
...	...		...	
(S+3)/2 - 4	Nibble 2	Nibble 1	Nibble 4	Nibble 3
(S+3)/2 - 2			Nibble 0 (lsb)	

Many cases exist other than the ones previously given; for example, where the message does not start on a word boundary.

11.1.5.14.3.2.2.2 Memory-Mapping of the ECC

The ECC (or remainder) is presented by the BCH module as a single 104-bit (or 52-bit), little-endian vector. It is up to the software to fetch those 13 bytes (or 6 bytes) from the modules interface, and then store them to the NAND spare area (page write) or to an intermediate buffer for comparison with the stored ECC (page read). There are no constraints on the ECC mapping inside the spare area: it is software-controlled.

However, it is advised to maintain a coherence in the respective formats of the message or the ECC remainder once they have been read out of the NAND. The error correction algorithm works from the complete codeword (concatenated message and remainder) once an error as been detected. The creation of this codeword must be made as straightforward as possible.

There are cases where the same NAND access contains both data and the ECC protecting that data. This is the case when the data/ECC boundary (which can be on any nibble) does not coincide with an access boundary. The ECC is calculated on-the-fly following the write. In that case, the write must also contain part of the ECC because it is impossible to insert the ECC on-the-fly. Instead:

- During the initial page write (BCH encoding), the ECC is replaced by dummy bits. The BCH encoder is by definition turned OFF during the ECC section, so the BCH result is unmodified.
- During a second phase, the ECC is written to the correct location, next to the actual data.
- The completed line buffer is then written to the NAND array.

11.1.5.14.3.2.2.3 Wrapping Modes

For a given wrapping mode, the module automatically goes through a specific number of sections, as data is being fed into the module. For each section, the BCH core can be enabled (in which case the data is fed to the BCH divider) or not (in which case the BCH simply counts to the end of the section). When enabled, the data is added to the ongoing calculation for a given sector number (for example, number 0).

Wrapping modes are described below. To get a better understanding and see the real-life read and write sequences implemented with each mode, see Section 11.1.5.14.3.2.3.

For each mode:

- A sequence describes the mode in pseudo-language, with the size and the buffer used for ECC processing (if ON) for each section. The programmable lengths are size, size0, and size1.
- A checksum condition is given. If the checksum condition is not respected for a given mode, the module behavior is unpredictable. S is the number of sectors in the page; size0 and size1 are the section sizes programmed for the mode, in nibbles.

Wrapping modes 8, 9, 10, and 11 insert a 1-nibble padding where the BCH processing is OFF. This is intended for t = 4 ECC, where ECC is 6 bytes long and the ECC area is expected to include (at least) one unused nibble to remain byte-aligned.

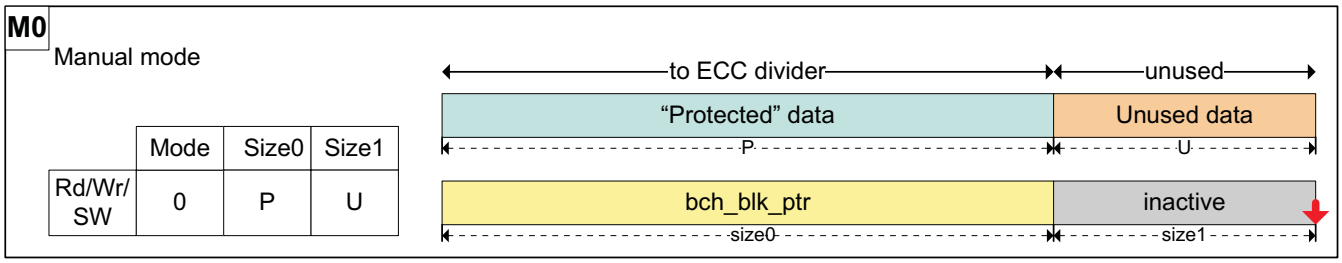
11.1.5.14.3.2.2.3.1 Manual Mode (0x0)

This mode is intended for short sequences, added manually to a given buffer through the software data port input. A complete page may be built out of several such sequences.

To process an arbitrary sequence of 4-bit nibbles, accesses to the software data port shall be made, containing the appropriate data. If the sequence end does not coincide with an access boundary (for example, to process 5 nibbles = 20 bits in 16-bit access mode) and those nibbles need to be skipped, a number of unused nibbles shall be programmed in size1 (in the same example: 5 nibbles to process + 3 to discard = 8 nibbles = exactly 2 x 16-bit accesses: we must program size0 = 5, size1 = 3).

Note: In the following figures size and size0 are the same parameter.

Figure 11-32. Manual Mode Sequence and Mapping



Section processing sequence:

- One time with buffer
  - size0 nibbles of data, processing ON
  - size1 nibbles of unused data, processing OFF

Checksum: size0 + size1 nibbles must fit in a whole number of accesses.

11.1.5.14.3.2.2.3.2 Mode 0x1

Page processing sequence:

- Repeat with buffer 0 to S-1
  - 512-byte data, processing ON
- Repeat with buffer 0 to S-1
  - size0 nibbles spare, processing ON
  - size1 nibbles spare, processing OFF

Checksum: Spare area size (nibbles) = S - (size0 + size1)

11.1.5.14.3.2.2.3.3 Mode 0xA (10)

Page processing sequence:

- Repeat with buffer 0 to S-1
  - 512-byte data, processing ON
- Repeat with buffer 0 to S-1
  - size0 nibbles spare, processing ON
  - 1 nibble pad spare, processing OFF
  - size1 nibbles spare, processing OFF

Checksum: Spare area size (nibbles) = S - (size0 + 1 + size1)

11.1.5.14.3.2.2.3.4 Mode 0x2

Page processing sequence:

- Repeat with buffer 0 to S-1
  - 512-byte data, processing ON
- Repeat with buffer 0 to S-1
  - size0 nibbles spare, processing OFF
  - size1 nibbles spare, processing ON

Checksum: Spare area size (nibbles) = S - (size0 + size1)

11.1.5.14.3.2.2.3.5 Mode 0x3

Page processing sequence:

- Repeat with buffer 0 to S-1
  - 512-byte data, processing ON
- One time with buffer 0
  - size0 nibbles spare, processing ON
- Repeat with buffer 0 to S-1
  - size1 nibbles spare, processing ON

Checksum: Spare area size (nibbles) = size0 + (S - size1)

11.1.5.14.3.2.2.3.6 Mode 0x7

Page processing sequence:

- Repeat with buffer 0 to S-1
  - 512-byte data, processing ON
- One time with buffer 0
  - size0 nibbles spare, processing ON
- Repeat S times (no buffer used)
  - size1 nibbles spare, processing OFF

Checksum: Spare area size (nibbles) = size0 + (S - size1)

11.1.5.14.3.2.2.3.7 Mode 0x8

Page processing sequence:

- Repeat with buffer 0 to S-1
  - 512-byte data, processing ON
- One time with buffer 0
  - size0 nibbles spare, processing ON
- Repeat with buffer 0 to S-1
  - 1 nibble padding spare, processing OFF
  - size1 nibbles spare, processing ON

Checksum: Spare area size (nibbles) = size0 + (S - (1+size1))

11.1.5.14.3.2.2.3.8 Mode 0x4

Page processing sequence:

- Repeat with buffer 0 to S-1
  - 512-byte data, processing ON
- One time (no buffer used)
  - size0 nibbles spare, processing OFF
- Repeat with buffer 0 to S-1
  - size1 nibbles spare, processing ON

Checksum: Spare area size (nibbles) = size0 + (S - size1)

11.1.5.14.3.2.2.3.9 Mode 0x9

Page processing sequence:

- Repeat with buffer 0 to S-1
  - 512-byte data, processing ON
- One time (no buffer used)

  - size0 nibbles spare, processing OFF
- Repeat with buffer 0 to S-1
  - 1 nibble padding spare, processing OFF
  - size1 nibbles spare, processing ON

Checksum: Spare area size (nibbles) = size0 + (S - (1+size1))

11.1.5.14.3.2.2.3.10 Mode 0x5

Page processing sequence:

- Repeat with buffer 0 to S-1
  - 512-byte data, processing ON
- Repeat with buffer 0 to S-1
  - size0 nibbles spare, processing ON
- Repeat with buffer 0 to S-1
  - size1 nibbles spare, processing ON

Checksum: Spare area size (nibbles) = S - (size0 + size1)

11.1.5.14.3.2.2.3.11 Mode 0xB (11)

Page processing sequence:

- Repeat with buffer 0 to S-1
  - 512-byte data, processing ON
- Repeat with buffer 0 to S-1
  - size0 nibbles spare, processing ON
- Repeat with buffer 0 to S-1
  - 1 nibble padding spare, processing OFF
  - size1 nibbles spare, processing ON

Checksum: Spare area size (nibbles) = S - (size0 + 1 + size1)

11.1.5.14.3.2.2.3.12 Mode 0x6

Page processing sequence:

- Repeat with buffer 0 to S-1
  - 512-byte data, processing ON
- Repeat with buffer 0 to S-1
  - size0 nibbles spare, processing ON
- Repeat S times (no buffer used)
  - size1 nibbles spare, processing OFF

Checksum: Spare area size (nibbles) = S - (size0 + size1)

11.1.5.14.3.2.3 Supported NAND Page Mappings and ECC Schemes

The following rules apply throughout the entire mapping description:

- Main data area (sectors) size is hardcoded to 512 bytes.
- Spare area size is programmable.
- All page sections (of main area data bytes, protected spare bytes, unprotected spare bytes, and ECC) are defined as explained in Section 11.1.5.14.3.2.2.1.

Each one of the following sections shows a NAND page mapping example (per-sector spare mappings, pooled spare mapping, per-sector spare mapping, with ECC separated at the end of the page).

In the mapping diagrams, sections that belong to the same BCH codeword have the same color (blue or green); unprotected sections are not covered (orange) by the BCH scheme.

Below each mapping diagram, a write (encoding) and read (decoding: syndrome generation) sequence is given, with the number of the active buffers at each point in time (yellow). In the inactive zones (grey), no computing is taking place but the data counter is still active.

A table on the left summarizes the mode, size0, size1 parameters to program for respectively write and read processing of a page, with the given mapping, where :

- P is the size of spare byte section Protected by the ECC (in nibbles)
- U is the size of spare byte section Unprotected by the ECC (in nibbles)
- E is the size of the ECC itself (in nibbles)
- S is the number of Sectors per page (2 in the current diagrams)

Each time the processing of a BCH block is complete (ECC calculation for write/encoding, syndrome generation for read/decoding, indicated by red arrows), the update pointer is pulsed. Note that the processing for block 0 can be the first or the last to complete, depending on the NAND page mapping and operation (read or write). All examples show a page size of 1kByte + spares; that is, S = 2 sectors of 512 bytes. The same principles can be extended to larger pages by adding more sectors.

The actual BCH codeword size is used during the error location work to restrict the search range: by definition, errors can only happen in the codeword that was actually written to the NAND, and not in the mathematical codeword of $n = 2^{13} - 1 = 8191$ bits. That codeword (higher-order bits) is all-zero and implicit during computations.

The actual BCH codeword size depends on the mode, programmed sizes, and sector number (all sizes in nibbles):

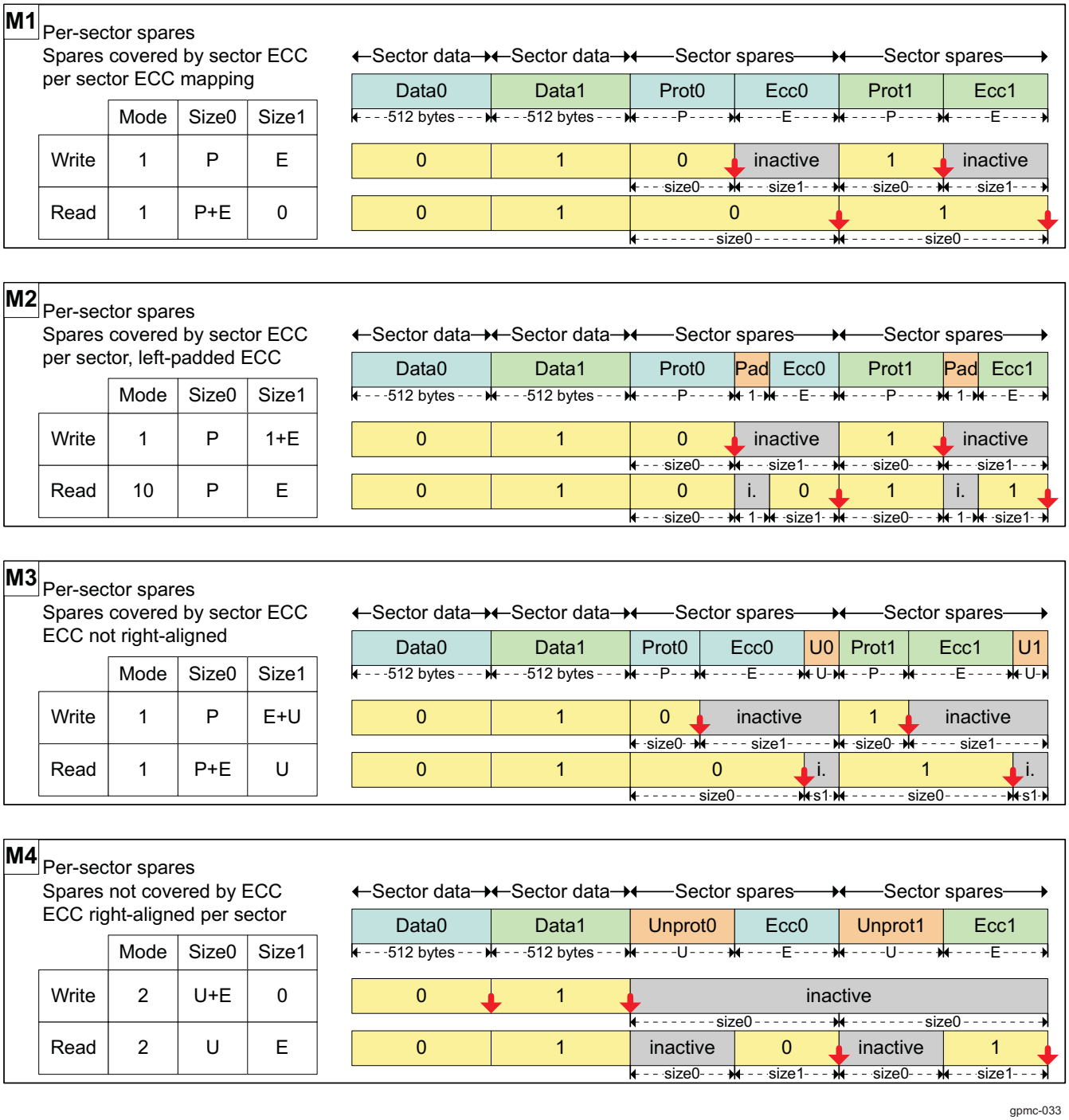
- Spares mapped and protected per sector (below: see M1-M2-M3-M9-M10):
  - all sectors: (512) + P + E
- Spares pooled and protected by sector 0 (below: see M5-M6):
  - sector 0 codeword: (512) + P + E
  - other sectors: (512) + E
- Unprotected spares (below: see M4-M7-M8-M11-M12):
  - all codewords (512) + E

11.1.5.14.3.2.3.1 Per-Sector Spare Mappings

In the schemes in Figure 11-33, each 512-byte sector of the main area has its own dedicated section of the spare area. The spare area of each sector is composed of :

- ECC, which must be located after the data it protects
- Other data, which may or may not be protected by the sector ECC

Figure 11-33. NAND Page Mapping and ECC: Per-Sector Schemes



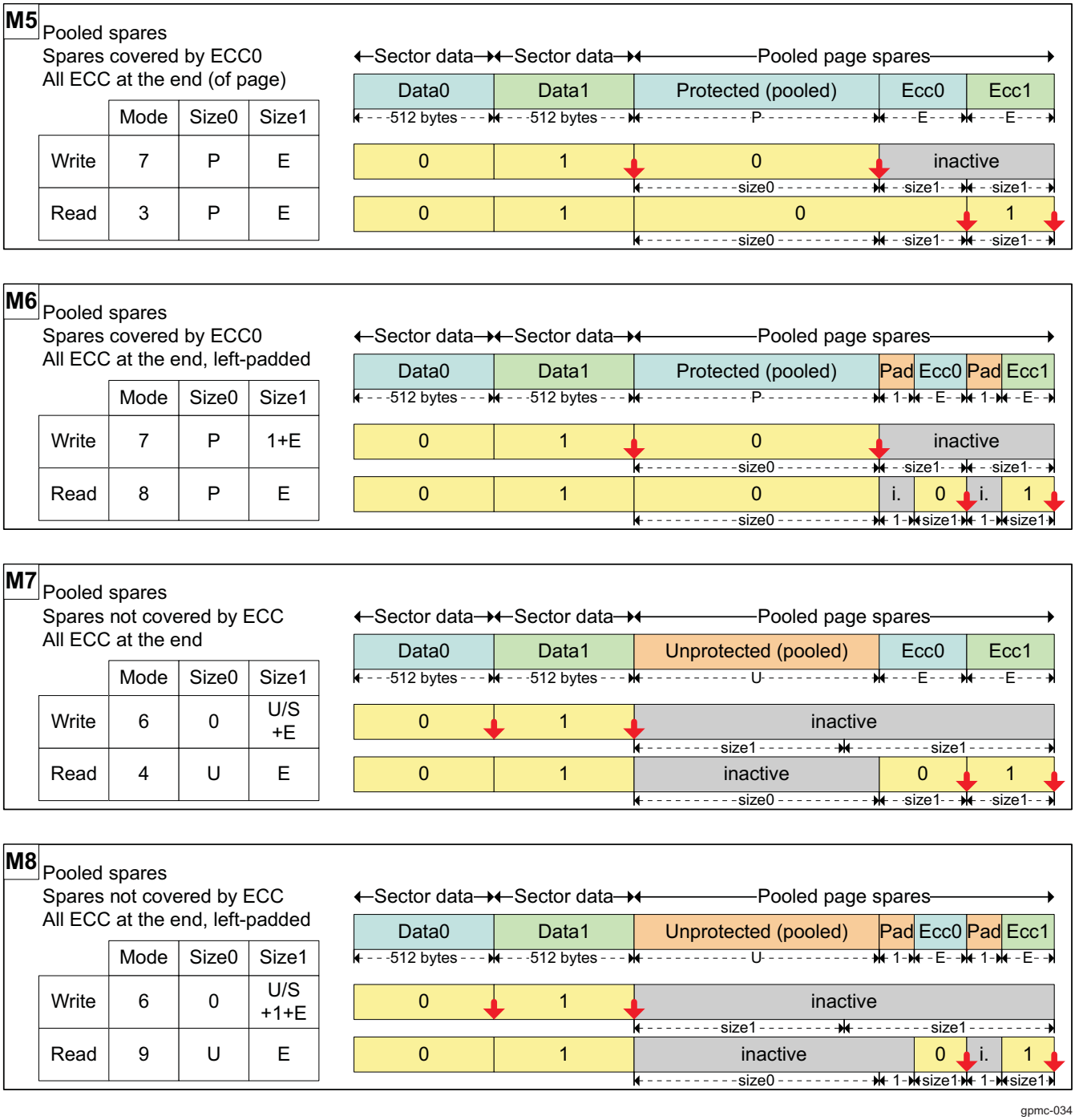
gpmc-033

11.1.5.14.3.2.3.2 Pooled Spare Mapping

In the schemes in Figure 11-34, the spare area is pooled for the page.

- The ECC of each sector is aligned at the end of the spare area.
- The non-ECC spare data may or may not be covered by the ECC of sector 0.

Figure 11-34. NAND Page Mapping and ECC: Pooled Spare Schemes



11.1.5.14.3.2.3.3 Per-Sector Spare Mapping, With ECC Separated at the End of the Page

In the schemes in Figure 11-35, each 512-byte sector of the main area is associated with two sections of the spare area.

- ECC section, all aligned at the end of the page
- Other data section, aligned before the ECCs, each of which may or may not be protected by its sectors ECC

Figure 11-35. NAND Page Mapping and ECC: Per-Sector Schemes, With Separate ECC



11.1.5.14.4 Prefetch and Write-Posting Engine

NAND device data access cycles are usually much slower than the MCU system frequency; such NAND read or write accesses issued by the processor will impact the overall system performance, especially considering long read or write sequences required for NAND page loading or programming. To minimize this effect on system performance, the GPMC includes a prefetch and write-posting engine, which can be used to read from or write to any chip-select location in a buffered manner.

The prefetch and write-posting engine uses an embedded 64 bytes (32 Word16) FIFO to prefetch data

from the NAND device in read mode (prefetch mode) or to store host data to be programmed into the NAND device in write mode (write-posting mode). The FIFO draining and filling (read and write) can be controlled either by the MCU through interrupt synchronization (an interrupt is triggered whenever a programmable threshold is reached) or the sDMA through DMA request synchronization, with a programmable request byte size in both prefetch or posting mode.

The prefetch and write-posting engine includes a single memory pool. Therefore, only one mode, read or write, can be used at any given time. In other words, the prefetch and write-posting engine is a single-context engine that can be allocated to only one chip-select at a time for a read prefetch or a write-posting process.

The engine does not support atomic command and address phase programming and is limited to linear memory read or write access. In consequence, it is limited to NAND data-stream access. The engine relies on the MCU NAND software driver to control block and page opening with the correct data address pointer initialization, before the engine can read from or write to the NAND memory device.

Once started, the engine data reads and writes sequencing is solely based on FIFO location availability and until the total programmed number of bytes is read or written.

Any host-concurrent accesses to a different chip-select are correctly interleaved with ongoing engine accesses. The engine has the lowest priority access so that host accesses to a different chip-select do not suffer a large latency.

A round-robin arbitration scheme can be enabled to ensure minimum bandwidth to the prefetch and write-posting engine in the case of back-to-back direct memory requests to a different chip-select. If the GPMC.GPMC_PREFETCH_CONFIG1[23] PFPWENROUNDROBIN bit is enabled, the arbitration grants the prefetch and write posting engine access to the GPMC bus for a number of requests programmed in the GPMC.GPMC_PREFETCH_CONFIG1[19:16] PFPWWEIGHTEDPRIO field.

The prefetch and write-posting engine is dedicated to data-stream access (as opposed to random data access). The engine does not include an address generator, and the request is limited to chip-select target identification. The prefetch/write-posting engine read or write request is routed to the access engine with the chip-select destination ID. After the required arbitration phase, the access engine processes the request as a single access with the data access size equal to the device size specified in the corresponding chip-select configuration.

Note: The destination chip-select configuration must be set to the NAND protocol-compatible configuration for which address lines are not used (the address bus is not changed from its current value). Selecting a different chip-select configuration can produce undefined behavior.

11.1.5.14.4.1 General Basic Programming Model

The engine can be configured only if the GPMC.GPMC_PREFETCH_CONTROL[0] STARTENGINE bit is de-asserted.

The engine must be correctly configured in prefetch or write-posting mode and must be linked to a NAND chip-select before it can be started. The chip-select is linked using the GPMC.GPMC_PREFETCH_CONFIG1[26:24] ENGINECSSELECTOR field.

In both prefetch and write-posting modes, the engine respectivelly uses byte or Word16 access requests for an 8- or 16-bit wide NAND device attached to the linked chip-select. The FIFOTHRESHOLD and TRANSFERCOUNT fields must be programmed accordingly as a number of bytes or a number of Word16.

When the GPMC.GPMC_PREFETCH_CONFIG1[7] ENABLEENGINE bit is set, the FIFO entry on the L3 interconnect port side is accessible at any address in the associated chip-select memory region. When the ENABLEENGINE bit is set, any host access to this chip-select is rerouted to the FIFO input. Directly accessing the NAND device linked to this chip-select from the host is still possible through the GPMC.GPMC_NAND_COMMAND_i, GPMC.GPMC_NAND_ADDRESS_i, and GPMC.GPMC_NAND_DATA_i registers (where i = 0 to 7).

The FIFO entry on the L3 interconnect port can be accessed with Byte, Word16, or Word32 access size, according to little-endian format, even though the FIFO input is 32-bit wide.

The FIFO control is made easier through the use of interrupts or DMA requests associated with the FIFOTHRESHOLD bit field. The GPMC.GPMC_PREFETCH_STATUS[30:24] FIFOPOINTER field monitors the number of available bytes to be read in prefetch mode or the number of free empty slots which can be written in write-posting mode. The GPMC.GPMC_PREFETCH_STATUS[13:0] COUNTVALUE field monitors the number of remaining bytes to be read or written by the engine according to the TRANSFERCOUNT value. The FIFOPOINTER and COUNTVALUE bit fields are always expressed as a number of bytes even if a 16-bit wide NAND device is attached to the linked chip-select.

In prefetch mode, when the FIFOPOINTER equals 0, that is, the FIFO is empty, a host read access receives the byte last read from the FIFO as its response. In case of Word32 or Word16 read accesses, the last byte read from the FIFO is copied the required number of times to fit the requested word size. In write-posting mode, when the FIFOPOINTER equals 0 (that is, the FIFO is full, a host write overwrites the last FIFO byte location). There is no underflow or overflow error reporting in the GPMC.

11.1.5.14.4.2 Prefetch Mode

The prefetch mode is selected when the GPMC.GPMC_PREFETCH_CONFIG1[0] ACCESSMODE bit is cleared.

The MCU NAND software driver must issue the block and page opening (READ) command with the correct data address pointer initialization before the engine can be started to read from the NAND memory device. The engine is started by asserting the GPMC.GPMC_PREFETCH_CONTROL[0] STARTENGINE bit. The STARTENGINE bit automatically clears when the prefetch process completes.

If required, the ECC calculator engine must be initialized (configured, reset, and enabled) before the prefetch engine is started, so that the ECC is correctly computed on all data read by the prefetch engine.

When the GPMC.GPMC_PREFETCH_CONFIG1[3] SYNCHROMODE bit is cleared, the prefetch engine starts requesting data as soon as the STARTENGINE bit is set. If using this configuration, the host must monitor the NAND device-ready pin so that it only sets the STARTENGINE bit when the NAND device is in a ready state, meaning data is valid for prefetching.

When the GPMC.GPMC_PREFETCH_CONFIG1[3] SYNCHROMODE bit is set, the prefetch engine starts requesting data when an active to inactive wait signal trnasition is detected. The transition detector must be cleared before any transition detection; see Section 11.1.5.14.2.2. The GPMC.GPMC_PREFETCH_CONFIG1[5:4] WAITPINSELECTOR field selects which gpmc_wait pin edge detector triggers the prefetch engine in this synchronized mode.

If the STARTENGINE bit is set after the NAND address phase (page opening command), the engine is effectively started only after the actual NAND address phase completion. To prevent GPMC stall during this NAND address phase, set the STARTENGINE bit field before NAND address phase completion when in synchronized mode. The prefetch engine will start when an active to inactive wait signal transition is detected. The STARTENGINE bit is automatically cleared on prefetch process completion.

The prefetch engine issues a read request to ensure that the FIFO is always filled with as much data as acceptable, until the programmed GPMC.GPMC_PREFETCH_CONFIG2[13:0] TRANSFERCOUNT field is completed.

Table 11-15. Prefetch Mode Configuration

Bit Field	Register	Value	Comments
STARTENGINE	GPMC_PREFETCH_CONTROL[0]	0	Prefetch engine can be configured only if STARTENGINE is set to 0.
ENGINECSSELECTOR	GPMC_PREFETCH_CONFIG1[26:24]	0 to 7	Selects the chip-select associated with a NAND device where the prefetch engine is active.
ACCESSMODE	GPMC_PREFETCH_CONFIG1[0]	0	Selects prefetch mode
FIFOTHRESHOLD	GPMC_PREFETCH_CONFIG1[14:8]		Selects the maximum number of bytes read or written by the host on DMA or interrupt request
TRANSFERCOUNT	GPMC_PREFETCH_CONFIG2[13:0]		Selects the number of bytes to be read or written by the engine to the selected chip-select

Table 11-15. Prefetch Mode Configuration (continued)

Bit Field	Register	Value	Comments
SYNCHROMODE	GPMC_PREFETCH_CONFIG1[3]	0/1	Selects when the engine starts the access to the chip-select
WAITPINSELECT	GPMC_PREFETCH_CONFIG1[17:16]	0 to 3	(If SynchroMode = 1) Selects wait pin edge detector
ENABLEOPTIMIZEDACCESS	GPMC_PREFETCH_CONFIG1[27]	0/1	See Section 11.1.5.14.4.6.
CYCLEOPTIMIZATION	GPMC_PREFETCH_CONFIG1[30:28]		
ENABLEENGINE	GPMC_PREFETCH_CONFIG1[7]	1	Engine enabled
STARTENGINE	GPMC_PREFETCH_CONTROL[0]	1	Starts the prefetch engine

11.1.5.14.4.3 FIFO Control in Prefetch Mode

The FIFO can be drained directly by the MPU or by an sDMA channel.

In MPU draining mode, the FIFO status can be monitored through the GPMC.GPMC_PREFETCH_STATUS[30:24] FIFOPOINTER field or through the GPMC.GPMC_PREFETCH_STATUS[16] FIFOTHRESHOLDSTATUS bit. The FIFOPOINTER indicates the current number of available data to be read; FIFOTHRESHOLDSTATUS set to 1 indicates that at least FIFOTHRESHOLD bytes are available from the FIFO.

An interrupt can be triggered by the GPMC if the GPMC.GPMC_IRQENABLE[0] FIFOEVENTENABLE bit is set. The FIFO interrupt event is logged, and the GPMC.GPMC_IRQSTATUS[0] FIFOEVENTSTATUS bit is set. To clear the interrupt, the MPU must read all the available bytes, or at least enough bytes to get below the programmed FIFO threshold, and the FIFOEVENTSTATUS bit must be cleared to enable further interrupt events. The FIFOEVENTSTATUS bit must always be reset prior to asserting the FIFOEVENTENABLE bit to clear any out-of-date logged interrupt event. This interrupt generation must be enabled after enabling the STARTENGINE bit.

Prefetch completion can be monitored through the GPMC.GPMC_PREFETCH_STATUS[13:0] COUNTVALUE field. COUNTVALUE indicates the number of currently remaining data to be requested according to the TRANSFERCOUNT value. An interrupt can be triggered by the GPMC when the prefetch process is complete (that is, COUNTVALUE equals 0) if the GPMC.GPMC_IRQENABLE[1] TERMINALCOUNTEVENTENABLE bit is set. At prefetch completion, thehe TERMINALCOUNT interrupt event is also logged, and the GPMC.GPMC_IRQSTATUS[1] TERMINALCOUNTSTATUS bit is set. To clear the interrupt, the MPU must clear the TERMINALCOUNTSTATUS bit. The TERMINALCOUNTSTATUS bit must always be cleared prior to asserting the TERMINALCOUNTEVENTENABLE bit to clear any out-of-date logged interrupt event.

Note: The COUNTVALUE value is only valid when the prefetch engine is active (started), and an interrupt is triggered only when COUNTVALUE reaches 0 (that is, when the prefetch engine automatically goes from an active to inactive state).

The number of bytes to be prefetched (programmed in TRANSFERCOUNT) must be a multiple of the programmed FIFOTHRESHOLD to trigger the correct number of interrupts allowing a deterministic and transparent FIFO control. If this guideline is respected, the number of ISR accesses is always required and the FIFO is always empty after the last interrupt is trigerred. In other cases, the TERMINALCOUNT interrupt must be used to read the remaining bytes in the FIFO (the number of remaining bytes being lower than the FIFOTHRESHOLD value).

In DMA draining mode, the GPMC.GPMC_PREFETCH_CONFIG1[2] DMAMODE bit must be set so that the GPMC issues a DMA hardware request when at least FIFOTHRESHOLD bytes are ready to be read from the FIFO. The DMA channel owning this DMA request must be programmed so that the number of bytes programmed in FIFOTHRESHOLD is read from the FIFO during the DMA request process. The DMA request is kept active until this number of bytes has effectively been read from the FIFO, and no other DMA request can be issued until the ongoing active request is complete.

In prefetch mode, the TERMINALCOUNT event is also a source of DMA requests if the number of bytes to be prefetched is not a multiple of FIFOTHRESHOLD, the remaining bytes in the FIFO can be read by the DMA channel using the last DMA request. This assumes that the number of remaining bytes to be read is known and controlled through the DMA channel programming model.

Any potentially active DMA request is cleared when the prefetch engine goes from inactive to active prefetch (the STARTENGINE bit is set to 1). The associated DMA channel must always be enabled by the MPU after setting the STARTENGINE bit so that the out-of-date active DMA request does not trigger spurious DMA transfers.

11.1.5.14.4.4 Write-Posting Mode

The write-posting mode is selected when the GPMC.GPMC_PREFETCH_CONFIG1[0] ACCESSMODE bit is set.

The MCU NAND software driver must issue the correct address pointer initialization command (page program) before the engine can start writing data into the NAND memory device. The engine starts when the GPMC.GPMC_PREFETCH_CONTROL[0] STARTENGINE bit is set to 1. The STARTENGINE bit clears automatically when posting completes. When all data have been written to the NAND memory device, the MCU NAND software driver must issue the second cycle program command and monitor the status for programming process completion (adding ECC handling, if required).

If used, the ECC calculator engine must be started (configured, reset, and enabled) before the posting engine is started so that the ECC parities are properly calculated on all data written by the prefetch engine to the associated chip-select.

In write-posting mode, the GPMC.GPMC_PREFETCH_CONFIG1[3] SYNCHROMODE bit must be cleared so that posting starts as soon as the STARTENGINE bit is set and the FIFO is not empty.

If the STARTENGINE bit is set after the NAND address phase (page program command), the STARTENGINE setting is effective only after the actual NAND command completion. To prevent GPMC stall during this NAND command phase, set the STARTENGINE bit field before the NAND address completion and ensure that the associated DMA channel is enabled after the NAND address phase.

The posting engine issues a write request when valid data are available from the FIFO and until the programmed GPMC.GPMC_PREFETCH_CONFIG2[13:0] TRANSFERCOUNT accesses have been completed.

The STARTENGINE bit clears automatically when posting completes. When all data have been written to the NAND memory device, the MCU NAND software driver must issue the second cycle program command and monitor the status for programming process completion. The closing program command phase must only be issued when the full NAND page has been written into the NAND flash write buffer, including the spare area data and the ECC parities, if used.

Table 11-16. Write-Posting Mode Configuration

Bit Field	Register	Value	Comments
STARTENGINE	GPMC_PREFETCH_CONTROL[0]	0	Write-posting engine can be configured only if STARTENGINE is set to 0.
ENGINECSSELECTOR	GPMC_PREFETCH_CONFIG1[26:24]	0 to 7	Selects the chip-select associated with a NAND device where the prefetch engine is active
ACCESSMODE	GPMC_PREFETCH_CONFIG1[0]	1	Selects write-posting mode
FIFOTHRESHOLD	GPMC_PREFETCH_CONFIG1[14:8]		Selects the maximum number of bytes read or written by the host on DMA or interrupt request
TRANSFERCOUNT	GPMC_PREFETCH_CONFIG2[13:0]		Selects the number of bytes to be read or written by the engine from/to the selected chip-select
SYNCHROMODE	GPMC_PREFETCH_CONFIG1[3]	0	Engine starts the access to chip-select as soon as STARTENGINE is set.
ENABLEOPTIMIZEDACCESS	GPMC_PREFETCH_CONFIG1[27]	0/1	See Section 11.1.5.14.4.6.
CYCLEOPTIMIZATION	GPMC_PREFETCH_CONFIG1[30:28]		
ENABLEENGINE	GPMC_PREFETCH_CONFIG1[7]	1	Engine enabled

Table 11-16. Write-Posting Mode Configuration (continued)

Bit Field	Register	Value	Comments
STARTENGINE	GPMC_PREFETCH_CONTROL[0]	1	Starts the prefetch engine

11.1.5.14.4.5 FIFO Control in Write-Posting Mode

The FIFO can be filled directly by the MPU or by an sDMA channel.

In MPU filling mode, the FIFO status can be monitored through the FIFOPOINTER or through the GPMC.GPMC_PREFETCH_STATUS[16] FIFOTHRESHOLDSTATUS bit. FIFOPOINTER indicates the current number of available free byte places in the FIFO, and the FIFOTHRESHOLDSTATUS bit, when set, indicates that at least FIFOTHRESHOLD free byte places are available in the FIFO.

An interrupt can be issued by the GPMC if the GPMC.GPMC_IRQENABLE[0] FIFOEVENTENABLE bit is set. When the interrupt is fired, the GPMC.GPMC_IRQSTATUS[0] FIFOEVENTSTATUS bit is set. To clear the interrupt, the MPU must write enough bytes to fill the FIFO, or enough bytes to get below the programmed threshold, and the FIFOEVENTSTATUS bit must be cleared to get further interrupt events. The FIFOEVENTSTATUS bit must always be cleared prior to asserting the FIFOEVENTENABLE bit to clear any out-of-date logged interrupt event. This interrupt must be enabled after enabling the STARTENGINE bit.

The posting completion can be monitored through the GPMC.GPMC_PREFETCH_STATUS[13:0] COUNTVALUE field. COUNTVALUE indicates the current number of remaining data to be written based on the TRANSFERCOUNT value. An interrupt is issued by the GPMC when the write-posting process completes (that is, COUNTVALUE equal to 0) if the GPMC.GPMC_IRQENABLE[1] TERMINALCOUNTEVENTENABLE bit is set. When the interrupt is fired, the GPMC.GPMC_IRQSTATUS[1] TERMINALCOUNTSTATUS bit is set. To clear the interrupt, the MPU must clear the TERMINALCOUNTSTATUS bit. The TERMINALCOUNTSTATUS bit must always be cleared prior to asserting the TERMINALCOUNTEVENTENABLE bit to clear any out-of-date logged interrupt event.

Note: The COUNTVALUE value is valid only if the write-posting engine is active and started, and an interrupt is issued only when COUNTVALUE reaches 0 (that is, when the posting engine automatically goes from active to inactive).

In DMA filling mode, the DMAMode bit field in the GPMC.GPMC_PREFETCH_CONFIG1[2] DMAMODE bit must be set so that the GPMC issues a DMA hardware request when at least FIFOTHRESHOLD bytes-free places are available in the FIFO. The DMA channel owning this DMA request must be programmed so that a number of bytes equal to the value programmed in the FIFOTHRESHOLD bit field are written into the FIFO during the DMA access. The DMA request remains active until the associated number of bytes has effectively been written into the FIFO, and no other DMA request can be issued until the ongoing active request has been completed.

Any potentially active DMA request is cleared when the prefetch engine goes from inactive to active prefetch (STARTENGINE set to 1). The associated DMA channel must always be enabled by the MPU after setting the STARTENGINE bit so that an out-of-date active DMA request does not trigger spurious DMA transfers.

In write-posting mode, the DMA or the MPU fill the FIFO with no consideration to the associated byte enables. Any byte stored in the FIFO is written into the memory device.

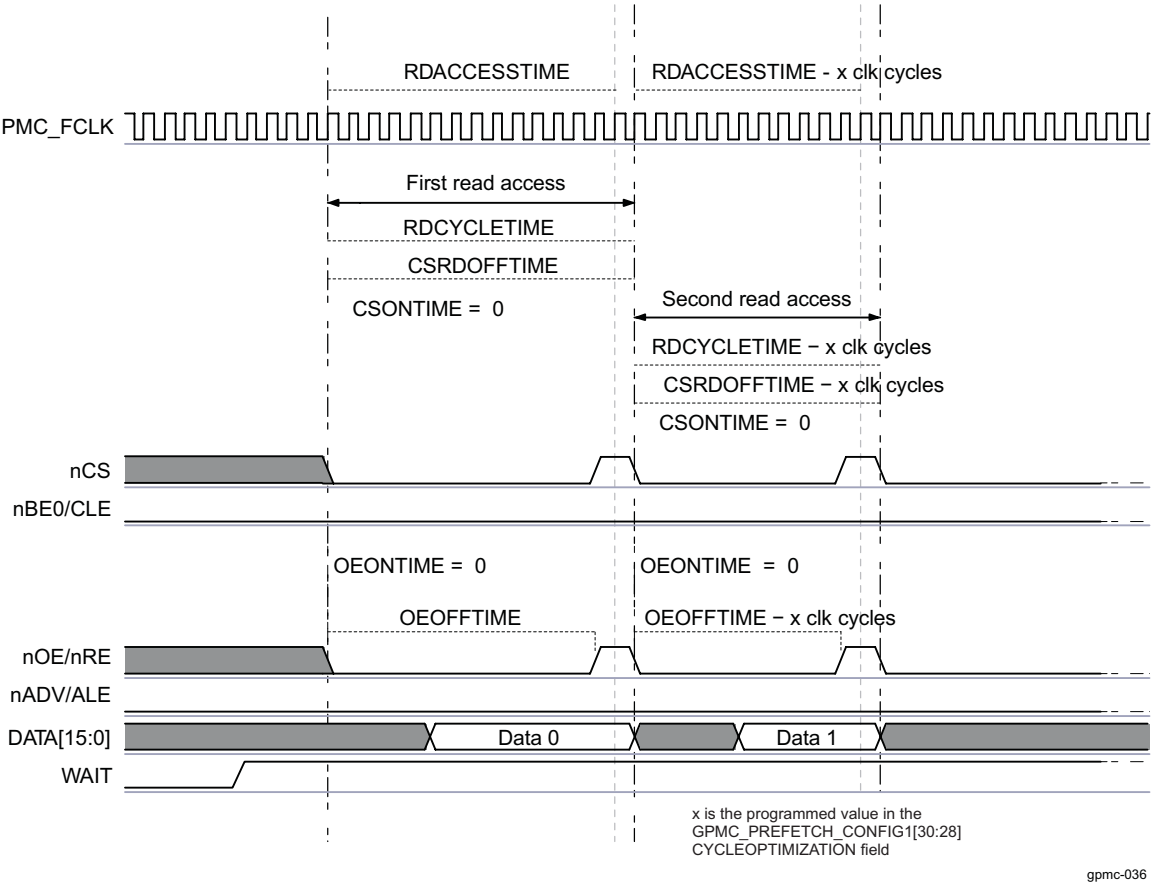
11.1.5.14.4.6 Optimizing NAND Access Using the Prefetch and Write-Posting Engine

Access time to a NAND memory device can be optimized for back-to-back accesses if the associated nCS signal is not deasserted between accesses. The GPMC access engine can track prefetch engine accesses to optimize the access timing parameter programmed for the allocated chip-select, if no accesses to other chip-selects (that is, interleaved accesses) occur. Similarly, the access engine also eliminates the CYCLE2CYCLEDELAY even if CYCLE2CYCLESAMECSEN is set. This capability is limited to the prefetch and write-posting engine accesses, and MPU accesses to a NAND memory device (through the defined chip-select memory region or through the GPMC.GPMC_NAND_DATA_i location, i = 0 to 7) are never optimized.

The GPMC.GPMC_PREFETCH_CONFIG1[27] ENABLEOPTIMIZEDACCESS bit must be set to enable optimized accesses. To optimize access time, the GPMC.GPMC_PREFETCH_CONFIG1[30:28] CYCLEOPTIMIZATION field defines the number of GPMC_FCLK cycles to be suppressed from the RDCYCLETIME, WRCYCLETIME, RDACCESSTIME, WRACCESSTIME, CSOFFTIME, ADVOFFTIME, OEOFFTIME, and WEOFFTIME timing parameters.

Figure 11-36 highlights that, in the case of back-to-back accesses to the NAND flash through the prefetch engine, CYCLE2CYCLESAMECSEN is forced to 0 when using optimized accesses. The first access uses the regular timing settings for this chip-select. All accesses after this one use settings reduced by x clock cycles, x being defined by the GPMC_PREFETCH_CONFIG1[30:28] CYCLEOPTIMIZATION field.

Figure 11-36. NAND Read Cycle Optimization Timing Description



11.1.5.14.4.7 Interleaved Accesses Between Prefetch and Write-Posting Engine and Other Chip-Selects

Any on-going read or write access from the prefetch and write-posting engine is completed before an access to any other chip-select can be initiated. As a default, the arbiter uses a fixed-priority algorithm, and the prefetch and write-posting engine has the lowest priority. The maximum latency added to access starting time in this case equals the RDCYCLETIME or WRCYCLETIME (optimized or not) plus the requested BUSTURNAROUND delay for bus turnaround completion programmed for the chip-select to which the NAND device is connected to.

Alternatively, a round-robin arbitration can be used to prioritize accesses to the external bus. This arbitration scheme is enabled by setting the GPMC.GPMC_PREFETCH_CONFIG1[23] PFPWENROUNDROBIN bit. When a request to another chip-select is received while the prefetch and write-posting engine is active, priority is given to the new request. The request processed thereafter is the prefetch and write-posting engine request, even if another interconnect request is passed in the mean time. The engine keeps control of the bus for an additional number of requests programmed in the GPMC.GPMC_PREFETCH_CONFIG1[19:16] PFPWWEIGHTEDPRIO bit field. Control is then passed to the direct interconnect request.

As an example, the round-robin arbitration scheme is selected with PFPWWEIGHTEDPRIO set to 0x2. Considering the prefetch and write-posting engine and the interconnect interface are always requesting access to the external interface, the GPMC grants priority to the direct interconnect access for one request. The GPMC then grants priority to the engine for three requests, and finnaly back to the direct interconnect access, until the arbiter is reset when one of the two initiators stops initiating requests.

11.1.6 GPMC Use Cases and Tips

11.1.6.1 How to Set GPMC Timing Parameters for Typical Accesses

11.1.6.1.1 External Memory Attached to the GPMC Module

As discussed in the introduction to this chapter, the GPMC module supports the following external memory types:

- Asynchronous or synchronous, 8-bit or 16-bit-width memory or device
- 16-bit address/data-multiplexed or not multiplexed NOR flash device
- 8- or 16-bit NAND flash device

The following examples show how to calculate GPMC timing parameters by showing a typical parameter setup for the access to be performed.

The example is based on a 512-Mb multiplexed NOR flash memory with the following characteristics:

- Type: NOR flash (address/data-multiplexed mode)
- Size: 512M bits
- Data Bus: 16 bits wide
- Speed: 104-MHz clock frequency
- Read access time: 80 ns

11.1.6.1.2 Typical GPMC Setup

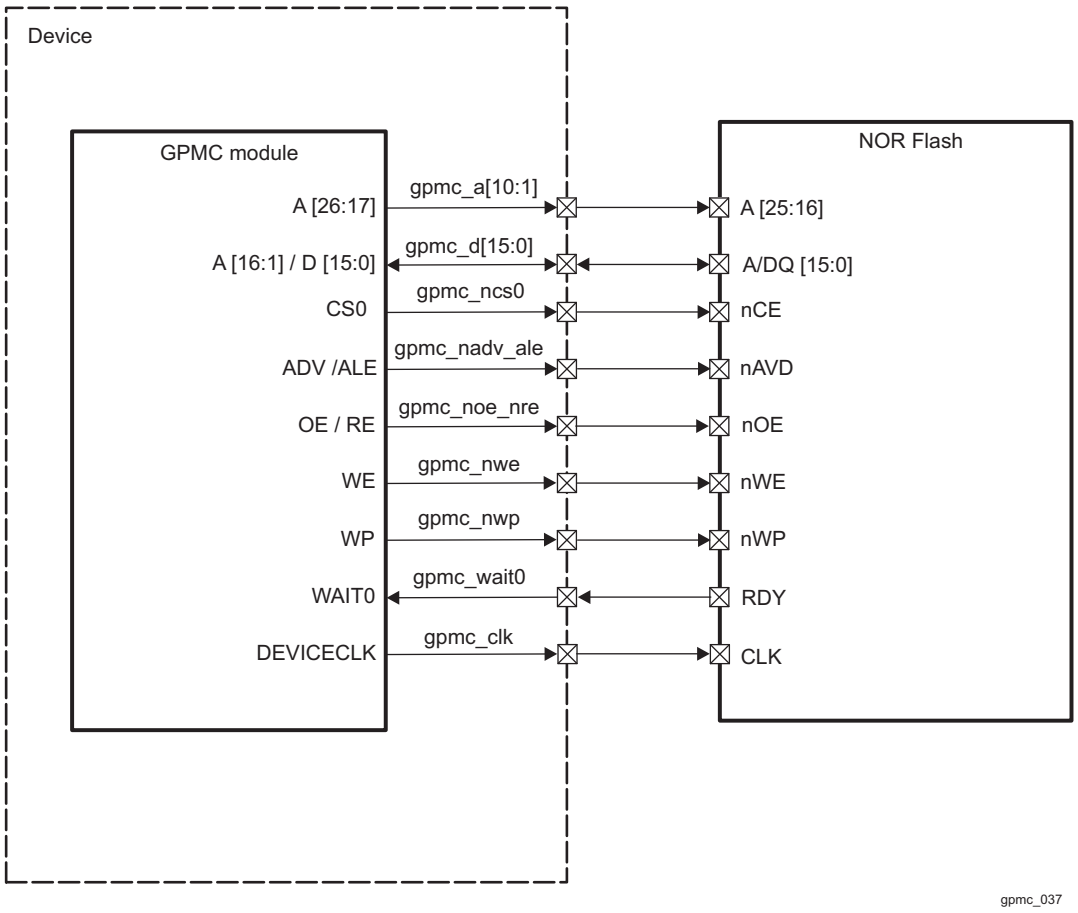
Table 11-17 lists some of the I/Os of the GPMC module.

Table 11-17. GPMC Signals

Signal Name	I/O	Description
GPMC_FCLK	Internal	Functional and interface clock. Acts as the time reference.
gpmc_clk	I/O	External clock provided to the external device for synchronous operations
gpmc_a[10: 1]	O	Address
gpmc_d[15: 0]	I/O	Data-multiplexed with addresses A[16:1] on memory side
gpmc_ncs	O	Chip-select
gpmc_nadv_ale	O	Address valid enable
gpmc_noe_nre	O	Output enable (read access only)
gpmc_nwe	O	Write enable (write access only)
gpmc_wait[3:0]	I	Ready signal from memory device. Indicates when valid burst data is ready to be read

Figure 11-37 shows the typical connection between the GPMC module and an attached NOR flash memory.

Figure 11-37. GPMC Connection to an External NOR Flash Memory



The following sections demonstrate how to calculate GPMC parameters for three access types:

- Synchronous burst read
- Asynchronous read
- Asynchronous single write

11.1.6.1.2.1 GPMC Configuration for Synchronous Burst Read Access

The clock runs at 104 MHz (f = 104 MHz; T = 9, 615 ns).

Table 11-18 shows the timing parameters (on the memory side) that determine the parameters on the GPMC side.

Table 11-19 shows how to calculate timings for the GPMC using the memory parameters.

Figure 11-38 shows the synchronous burst read access.

Table 11-18. Useful Timing Parameters on the Memory Side

AC Read Characteristics on the Memory Side	Description	Duration (ns)
tCES	nCS setup time to clock	0
tACS	Address setup time to clock	3
tIACC	Synchronous access time	80

Table 11-18. Useful Timing Parameters on the Memory Side (continued)

AC Read Characteristics on the Memory Side	Description	Duration (ns)
tBACC	Burst access time valid clock to output delay	5,2
tCEZ	Chip-select to High-Z	7
tOEZ	Output enable to High-Z	7
tAVC	nADV setup time	6
tAVD	nAVD pulse	6
tACH	Address hold time from clock	3

The following terms, which describe the timing interface between the controller and its attached device, are used to calculate the timing parameters on the GPMC side:

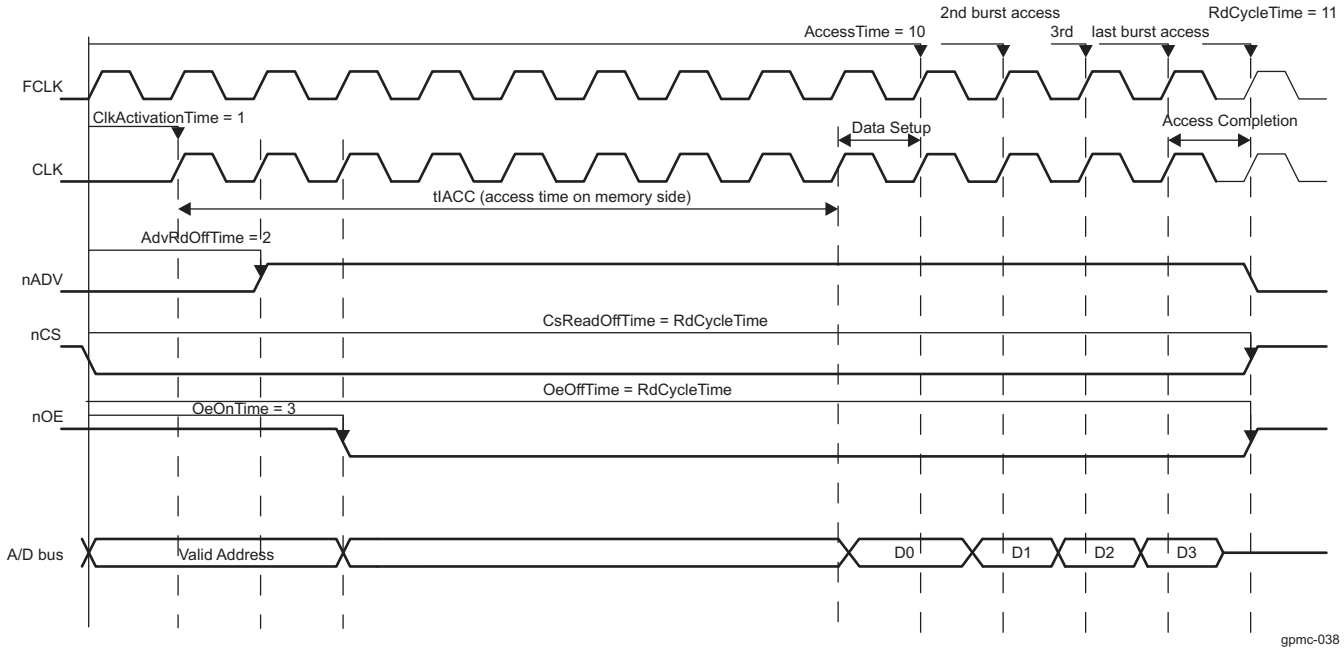
- Read access time (GPMC side): Time required to activate the clock + read access time requested on the memory side + data setup time required for optimal capture of a burst of data
- Data setup time (GPMC side): Ensures a good capture of a burst of data (as opposed to taking a burst of data out). One burst of data is processed in one clock cycle (T = 9,615 ns). The read access time between two bursts of data is tBACC = 5,2 ns. Therefore, data setup time is a clock period - tBACC = 4,415 ns of data setup.
- Access completion (GPMC side): (Different from page burst access time) Time required between the last burst access and access completion: nCS/nOE hold time (nCS and nOE must be released at the end of an access. These signals are held to allow the access to complete).
- Read cycle time (GPMC side): Read access time + access completion
- Write cycle time for burst access: Not supported for NOR flash memory

Table 11-19. Calculating GPMC Timing Parameters

Parameter Name on GPMC Side	Formula	Duration (ns)	Number of Clock Cycles (F = 104 MHz)	GPMC Register Configurations
ClkActivation Time	min (tCES, tACS)	3	1	CLKACTIVATIONTIME = 0x1
RdAccessTime	roundmax (ClkActivationTime + tIACC + DataSetupTime)	94,03: (9,615 + 80 + 4,415)	10 : roundmax (94,03 / 9,615)	ACCESSTIME = 0x0A
PageBurst AccessTime	roundmax (tBACC)	roundmax (5,2)	1	PAGEBURSTACCESSTIME = 0x1
RdCycleTime	AccessTime + max (tCEZ, tOEZ)	101, 03: (94, 03 + 7)	11	RDCYCLETIME = 0x0B
CsOnTime	tCES	0	0	CSONTIME = 0x0
CsReadOffTime	RdCycleTime	-	11	CSRDOFFTIME = 0x0B
AdvOnTime	tAVC (1)	0	0	ADVONTIME = 0x0
AdvRdOffTime	tAVD + tAVC (2)	12	2	ADVRDOFFTIME = 0x02
OeOnTime (3)	(ClkActivationTime + tACH) < OeOnTime < (ClkActivationTime + tIACC)	-	3 for instance.	OEONTIME = 0x3
OeOffTime	RdCycleTime	-	11	OEOFFTIME = 0x0B

(1) The external clock provided to the NOR flash is not yet available.
(2) AdvRdOffTime - AdvOnTime = tAVD; thus, AdvRdOffTime = tAVD + AdvOnTime = tAVD + tAVC.
(3) OeOnTime must guarantee that addresses are available. It must not exceed the availability of the first burst of data.

Figure 11-38. Synchronous Burst Read Access (Timing Parameters in Clock Cycles)



11.1.6.1.2.2 GPMC Configuration for Asynchronous Read Access

The clock runs at 104 MHz (f = 104 MHz; T = 9, 615 ns).

Table 11-20 shows the timing parameters (on the memory side) that determine the parameters on the GPMC side.

Table 11-21 shows how to calculate timings for the GPMC using the memory parameters.

Figure 11-39 shows the asynchronous read access.

Table 11-20. AC Characteristics for Asynchronous Read Access

AC Read Characteristics on the Memory Side	Description	Duration (ns)
tCE	Read Access time from nCS low	80
tAAVDS	Address setup time to rising edge of nADV	3
tAVDP	nADV low time	6
tCAS	nCS setup time to nADV	0
tOE	Output enable to output valid	6
tOEZ	Output enable to High-Z	7

Use the following formula to calculate the RdCycleTime parameter for this typical access:

RdCycleTime = RdAccessTime + AccessCompletion = RdAccessTime + 1 clock cycle + tOEZ:

- First, on the memory side, the external memory makes the data available to the output bus. This is the memory-side read access time defined in Table 11-20: the number of clock cycles between the address capture (nADV rising edge) and the data valid on the output bus.

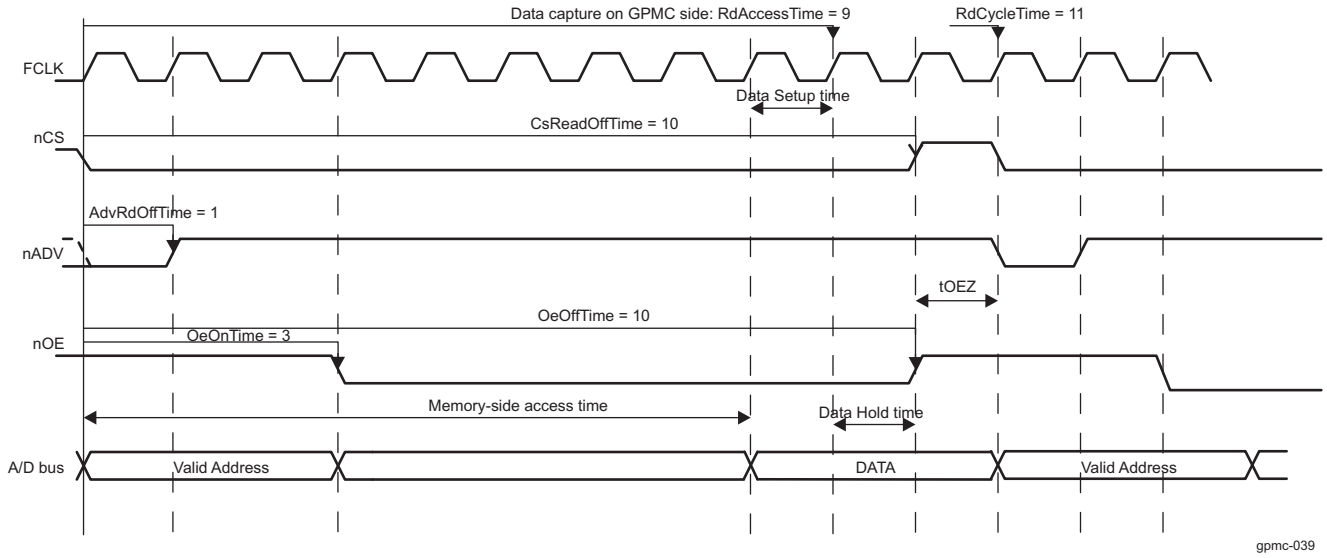
  The GPMC side requires some hold to allow the data to be captured correctly and the access to be finished.
- To read the data correctly, the GPMC must capture it with enough data setup time; the GPMC module captures the data on the next rising edge. This is access time on the GPMC side.
- There must also be a data hold time for correctly reading the data (checking that there is no nOE/nCS deassertion while reading the data). This data hold time is 1 clock cycle (AccessTime + 1).

- To complete the access, nOE/nCS signals are driven to High-Z. AccessTime + 1 + tOEZ is the read cycle time.
- Addresses can now be relatched and a new read cycle begun.

Table 11-21. GPMC Timing Parameters for Asynchronous Read Access

Parameter Name on GPMC side	Formula	Duration (ns)	Number of Clock Cycles (F = 104 MHz)	GPMC Register Configurations
ClkActivationTime	n/a (asynchronous mode)			
AccessTime	round max (tCE)	80	9	ACCESSTIME = 0x09
PageBurstAccess Time	n/a (single access)			
RdCycleTime	AccessTime + 1cycle + tOEZ	96, 615	11	RDCYCLETIME = 0x0B
CsOnTime	tCAS	0	0	CSONTIME = 0x0
CsReadOffTime	AccessTime + 1 cycle	89, 615	10	CSRDOFFTIME = 0x0A
AdvOnTime	tAAVDS	3	1	ADVONTIME = 0X1
AdvRdOffTime	tAAVDS + tAVDP	9	1	ADVRDOFFTIME = 0x01
OeOnTime	OeOnTime >= AdvRdOffTime (multiplexed mode)	-	3 for instance	OEONTIME = 0x3
OeOffTime	AccessTime + 1cycle	89, 615	10	OEOFFTIME = 0x0A

Figure 11-39. Asynchronous Single Read Access (Timing Parameters in Clock Cycles)



11.1.6.1.2.3 GPMC Configuration for Asynchronous Single Write Access

The clock runs at 104 MHz: (f = 104 MHz; T = 9, 615 ns).

Table 11-23 shows how to calculate timings for the GPMC using the memory parameters.

Table 11-22 shows the timing parameters (on the memory side) that determine the parameters on the GPMC side.

Figure 11-40 shows the synchronous burst write access.

Table 11-22. AC Characteristics for Asynchronous Single Write (Memory Side)

AC Characteristics on the Memory Side	Description	Duration (ns)
tWC	Write cycle time	60

Table 11-22. AC Characteristics for Asynchronous Single Write (Memory Side) (continued)

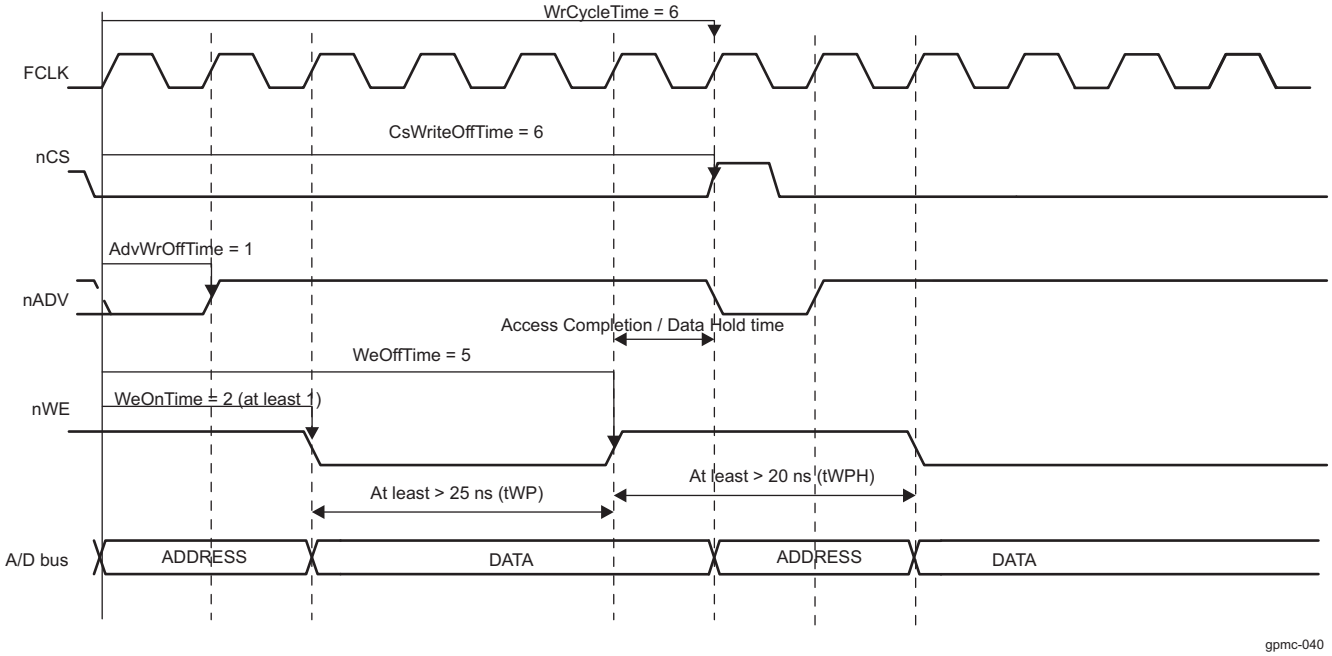
AC Characteristics on the Memory Side	Description	Duration (ns)
tAVDP	nADV low time	6
tWP	Write pulse width	25
tWPH	Write pulse width high	20
tCS	nCS setup time to nWE	3
tCAS	nCS setup time to nADV	0
tAVSC	nADV setup time	3

For asynchronous single write access, write cycle time is WrCycleTime = WeOffTime + AccessCompletion = WeOffTime + 1. For the AccessCompletion: 1 cycle is required for data hold time (nCS deassertion).

Table 11-23. GPMC Timing parameters for Asynchronous Single Write

Parameter Name on GPMC side	Formula	Duration (ns)	Number of Clock Cycles (F = 104 MHz)	GPMC Registers Configuration
ClkActivationTime	n/a (asynchronous mode)			
AccessTime	Applicable only to WAITMONITORING (the value is the same as for read access)			
PageBurstAccessTime	n/a (single access)			
WrCycleTime	WeOffTime + AccessCompletion	57, 615	6	WRCYCLETIME = 0x06
CsOnTime	tCAS	0	0	CSONTIME = 0x0
CsWrOffTime	WeOffTime + 1	57, 615	6	CSWROFFTIME = 0x06
AdvOnTime	tAVSC	3	1	ADVONTIME = 0x1
AdvWrOffTime	tAVSC + tAVDP	9	1	ADVWROFFTIME = 0x01
WeOnTime	tCS	3	1	WEONTIME = 0x1
WeOffTime	tCS + tWP + tWPH	48	5	WEOFFTIME = 0x05

Figure 11-40. Asynchronous Single Write Access (Timing Parameters in Clock Cycles)



11.1.6.2 How to Choose a Suitable Memory to Use With the GPMC

This section explains how to select a suitable memory device to interface with the GPMC controller.

11.1.6.2.1 Supported Memories or Devices

NAND flash and NOR flash architectures are the two flash technologies. The GPMC supports various types of external memory or device, basically any one that supports NAND or NOR protocols:

- 8- and 16-bit width asynchronous or synchronous memory or device (8-bit: non burst device only)
- 16-bit address and data multiplexed NOR flash devices (pSRAM, OneNAND™, ...)
- 8- and 16-bit NAND flash device

Note: Nonmultiplexed NOR flash devices are supported by the GPMC, but their use is highly limited. Because only ten address pins are available on the GPMC interface, the maximum device size supported is 2KB.

11.1.6.2.1.1 Memory Pin Multiplexing

This section describes the interfacing differences of the supported GPMC memories.

Table 11-24. Supported Memory Interfaces

Function	16-Bit Address/Data Muxed pSRAM or NOR Flash [1]	OneNAND	16-bit NAND	8-bit NAND
gpmc_a10	A26			
gpmc_a9	A25			
gpmc_a8	A24			
gpmc_a7	A23			
gpmc_a6	A22			
gpmc_a5	A21			
gpmc_a4	A20			
gpmc_a3	A19			
gpmc_a2	A18			
gpmc_a1	A17			
gpmc_d15	D15 or A16		IO15	
gpmc_d14	D14 or A15		IO14	
gpmc_d13	D13 or A14		IO13	
gpmc_d12	D12 or A13		IO12	
gpmc_d11	D11 or A12		IO11	
gpmc_d10	D10 or A11		IO10	
gpmc_d9	D9 or A10		IO9	
gpmc_d8	D8 or A9		IO8	
gpmc_d7	D7 or A8		IO7	
gpmc_d6	D6 or A7		IO6	
gpmc_d5	D5 or A6		IO5	
gpmc_d4	D4 or A5		IO4	
gpmc_d3	D3 or A4		IO3	
gpmc_d2	D2 or A3		IO2	
gpmc_d1	D1 or A2		IO1	
gpmc_d0	D0 or A1		IO0	
gpmc_clk	CLK			
gpmc_ncs0	nCS0 (chip select)		nCE0 (chip enable)	

(1) Addresses seen from the OMAP side. When interfacing to the external IC, A1 is connected to the memory A0, A2 to the memory A1, and so on.

Table 11-24. Supported Memory Interfaces (continued)

Function	16-Bit Address/Data Muxed pSRAM or NOR Flash (1)	OneNAND	16-bit NAND	8-bit NAND
gpmc_ncs1	nCS1		nCE1	
gpmc_ncs2	nCS2		nCE2	
GPMC_ncs3	nCS3		nCE3	
GPMC_ncs4	nCS4		nCE4	
GPMC_ncs5	nCS5		nCE5	
GPMC_ncs6	nCS6		nCE6	
GPMC_ncs7	nCS7		nCE7	
gpmc_nadv_ale	nADV (address valid)		ALE (address latch enable)	
gpmc_noe	nOE (output enable)		nRE (read enable)	
gpmc_nwe	nWE (write enable)		nWE (write enable)	
gpmc_nbe0_cle	nBE0 (byte enable)		CLE (command latch enable)	
gpmc_nbe1	nBE1			
gpmc_nwp	nWP (write protect)		nWP (write protect)	
gpmc_wait0	WAIT0		R/nB0 (ready/busy)	
gpmc_wait1	WAIT1		R/nB1	
gpmc_wait2	WAIT2		R/nB2	
gpmc_wait3	WAIT3		R/nB3	

11.1.6.2.1.2 NAND Interface Protocol

NAND flash architecture, introduced in 1989, is a flash technology. NAND is a page-oriented memory device (that is, read and write accesses are done by pages). NAND achieves density by sharing common areas of the storage transistor, which creates strings of serially connected transistors (in NOR devices, each transistor stands alone). Because of its high density, NAND is best suited to devices requiring high capacity data storage, such as pictures, music, or data files. Because of its nonvolatility, NAND is a good storage solution for many applications where mobility, low power, and speed are key factors. Low pin count and simple interface are other advantages of NAND.

Table 11-25 summarizes the level of the NAND interface signals applied to external devices or memories.

Table 11-25. NAND Interface Bus Operations Summary

Bus operation	CLE	ALE	nCE	nWE (1)	nRE (1)	nWP
Read (cmd input)	H	L	L	RE	H	x
Read (add input)	L	H	L	RE	H	x
Write (cmd input)	H	L	L	RE	H	H
Write (add input)	L	H	L	RE	H	H
Data input	L	L	L	RE	H	H
Data output	L	L	L	H	FE	x
Busy (during read)	x	x	H (2)	H (2)	H (2)	x
Busy (during program)	x	x	x	x	x	H
Busy (during erase)	x	x	x	x	x	H
Write protect	x	x	x	x	x	L
Stand-by	x	x	H	x	x	H/L (3)

(1) RE stands for rising edge, FE stands for falling edge.
(2) Can be either nCE high, or WE and nRE high
(3) nWP must be biased to CMOS high or CMOS low for standby.

11.1.6.2.1.3 NOR Interface Protocol

NOR flash architecture, introduced in 1988, is a flash technology. Unlike NAND, which is a sequential access device, NOR is directly addressable (that is, it is designed to be a random access device). NOR is best suited to devices used to store and run code or firmware, usually in small capacities. While NOR has fast read capabilities, it has slow write and erase functions compared to NAND architecture.

Table 11-26 summarizes the level of the NOR interface signals applied to external devices or memories.

Table 11-26. NOR Interface Bus Operations Summary

Bus operation	CLK	nADV	nCS	nOE	nWE	WAIT	DQ[15:0]
Read (asynchronous)	x	L	L	L	H	Asserted	Output
Read (synchronous)	Running	L	L	L	H	Driven	Output
Read (burst suspend)	Halted	x	L	H	H	Active	Output
Write	x	L	L	H	L	Asserted	Input
Output disable	x	x	L	H	H	Asserted	High-Z
Standby	x	x	H	x	x	High-Z	High-Z

11.1.6.2.1.4 Other Technologies

Other supported device types interact with the GPMC through the NOR interface protocol.

OneNAND is a high density and low-power memory device. OneNAND is based on a single- or multilevel-cell NAND core with SRAM and logic, and interfaces as a synchronous NOR flash; it also has synchronous write capability. It reads faster than conventional NAND and writes faster than conventionnal NOR flash. Therefore, it is appropriate for both mass storage and code storage.

pSRAM is a low-power memory device for mobile applications. pSRAM is based on the DRAM cell with internal refresh and address control features, and interfaces as a synchronous NOR flash; it also has synchronous write capability.

11.1.6.2.1.5 Supported Protocols

The GPMC supports the following interface protocols when communicating with external memory or external devices:

- Asynchronous read/write access
- Asynchronous read page access (4-8-16 Word16)
- Synchronous read/write access
- Synchronous read burst access without wrap capability (4-8-16 Word16)
- Synchronous read burst access with wrap capability (4-8-16 Word16)

11.1.6.2.2 GPMC Features and Settings

This section lists the GPMC features and settings:

- Supported device type: up to eight NAND or NOR protocol external memories or devices
- Operating voltage: 1.8 V
- Maximum operating frequency provided externally: up to 100 MHz (single device) with an L3-clock of 100 MHz. Up to 83 MHz (L3-clock divided by two) with an L3-clock of 166 MHz.
- Maximum GPMC addressing capability: 1 GB divided into eight chip-selects
- Maximum supported memory size: 128 MB (must be a power-of-2)
- Minimum supported memory size: 16 MB (must be a power-of-2). Aliasing occurs when addressing smaller memories.
- Data path to external memory or device: 8- and 16-bit wide
- Burst and page access: burst of 4-8-16 Word16
- Supports bus keeping and bus turnaround

11.1.7 GPMC Register Manual

11.1.7.1 GPMC Instance Summary

Table 11-27 describes the GPMC instance.

Table 11-27. GPMC Instance Summary

Module Name	Base Address	Size
GPMC	0x6E00 0000	16 MB

11.1.7.2 GPMC Register Summary

Table 11-28 is a summary of the GPMC registers.

Table 11-28. GPMC Registers Mapping Summary

Register Name	Type	Register Width (Bits)	Address Offset	Physical Address
GPMC_REVISION	R	32	0x0000 0000	0x6E00 0000
GPMC_SYSCONFIG	RW	32	0x0000 0010	0x6E00 0010
GPMC_SYSSTATUS	R	32	0x0000 0014	0x6E00 0014
GPMC_IRQSTATUS	RW	32	0x0000 0018	0x6E00 0018
GPMC_IRQENABLE	RW	32	0x0000 001C	0x6E00 001C
GPMC_TIMEOUT_CONTROL	RW	32	0x0000 0040	0x6E00 0040
GPMC_ERR_ADDRESS	RW	32	0x0000 0044	0x6E00 0044
GPMC_ERR_TYPE	RW	32	0x0000 0048	0x6E00 0048
GPMC_CONFIG	RW	32	0x0000 0050	0x6E00 0050
GPMC_STATUS	RW	32	0x0000 0054	0x6E00 0054
GPMC_CONFIG1_i (1)	RW	32	0x0000 0060 + (0x0000 0030 * i)	0x6E00 0060 + (0x0000 0030 * i)
GPMC_CONFIG2_i (1)	RW	32	0x0000 0064 + (0x0000 0030 * i)	0x6E00 0064 + (0x0000 0030 * i)
GPMC_CONFIG3_i (1)	RW	32	0x0000 0068 + (0x0000 0030 * i)	0x6E00 0068 + (0x0000 0030 * i)
GPMC_CONFIG4_i (1)	RW	32	0x0000 006C + (0x0000 0030 * i)	0x6E00 006C + (0x0000 0030 * i)
GPMC_CONFIG5_i (1)	RW	32	0x0000 0070 + (0x0000 0030 * i)	0x6E00 0070 + (0x0000 0030 * i)
GPMC_CONFIG6_i (1)	RW	32	0x0000 0074 + (0x0000 0030 * i)	0x6E00 0074 + (0x0000 0030 * i)
GPMC_CONFIG7_i (1)	RW	32	0x0000 0078 + (0x0000 0030 * i)	0x6E00 0078 + (0x0000 0030 * i)
GPMC_NAND_COMMAND_i (1)	W	32	0x0000 007C + (0x0000 0030 * i)	0x6E00 007C + (0x0000 0030 * i)
GPMC_NAND_ADDRESS_i (1)	W	32	0x0000 0080 + (0x0000 0030 * i)	0x6E00 0080 + (0x0000 0030 * i)
GPMC_NAND_DATA_i (1)	RW	32	0x0000 0084 + (0x0000 0030 * i)	0x6E00 0084 + (0x0000 0030 * i)
GPMC_PREFETCH_CONFIG1	RW	32	0x0000 01E0	0x6E00 01E0
GPMC_PREFETCH_CONFIG2	RW	32	0x0000 01E4	0x6E00 01E4
GPMC_PREFETCH_CONTROL	RW	32	0x0000 01EC	0x6E00 01EC
GPMC_PREFETCH_STATUS	RW	32	0x0000 01F0	0x6E00 01F0
GPMC_ECC_CONFIG	RW	32	0x0000 01F4	0x6E00 01F4
GPMC_ECC_CONTROL	RW	32	0x0000 01F8	0x6E00 01F8
GPMC_ECC_SIZE_CONFIG	RW	32	0x0000 01FC	0x6E00 01FC
GPMC_ECCj_RESULT(2) where k = j – 1(3)	RW	32	0x0000 0200 + (0x0000 0004 * k)	0x6E00 0200 + (0x0000 0004 * k)
GPMC_BCH_RESULT0_i (1)	RW	32	0x0000 0240 + (0x0000 0010 * i)	0x6E00 0240 + (0x0000 0010 * i)
GPMC_BCH_RESULT1_i (1)	RW	32	0x0000 0244 + (0x0000 0010 * i)	0x6E00 0244 + (0x0000 0010 * i)
GPMC_BCH_RESULT2_i (1)	RW	32	0x0000 0248 + (0x0000 0010 * i)	0x6E00 0248 + (0x0000 0010 * i)
GPMC_BCH_RESULT3_i (1)	RW	32	0x0000 024C + (0x0000 0010 * i)	0x6E00 024C + (0x0000 0010 * i)
GPMC_BCH_SWDATA	RW	32	0x0000 02D0	0x6E00 02D0

(1) i = 0 to 7
(2) j = 1 to 9
(3) k = 0 to 8

11.1.7.3 GPMC Register Description

This section provides a description of GPMC registers.

Note: All GPMC registers are aligned to 32-bit address boundaries. All register file accesses, except to GPMC_NAND_DATA_i register, are little endian. If the GPMC_NAND_DATA_i register location is accessed, the endianness is access-dependent.

Table 11-29. GPMC_REVISION

Address Offset	0x0000 0000		
Physical Address	0x6E00 0000	**Instance**	GPMC
Description	This register contains the IP revision code.		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																								REV							

Bits	Field Name	Description	Type	Reset
31:8	Reserved	Reads return 0.	R	0x000000
7:0	REV	IP revision [7:4] Major revision [3:0] Minor revision Examples: 0x10 for 1.0, 0x21 for 2.1	R	See (1)

(1) TI internal data

Table 11-30. Register Call Summary for Register GPMC_REVISION

General-Purpose Memory Controller

- GPMC Register Summary: [0]

Table 11-31. GPMC_SYSCONFIG

Address Offset	0x0000 0010		
Physical Address	0x6E00 0010	**Instance**	GPMC
Description	This register controls the various parameters of the Interconnect.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																											IDLEMODE		RESERVED	SOFTRESET	AUTOIDLE

Bits	Field Name	Description	Type	Reset
31:5	RESERVED	Write 0s for future compatibility. Reads return 0s.	RW	0x0000000
4:3	IDLEMODE	0x0: Force-idle. An idle request is acknowledged unconditionally 0x1: No-idle. An idle request is never acknowledged 0x2: Smart-idle. Acknowledgment to an idle request is given based on the internal activity of the module 0x3: Do not use	RW	0x0
2	RESERVED	Write 0 for future compatibility Reads returns 0	RW	0x0

Bits	Field Name	Description	Type	Reset
1	SOFTRESET	Software reset. Set this bit to 1 triggers a module reset. This bit is automatically reset by hardware. During reads, it always returns 0. 0x0: Normal mode 0x1: The module is reset	RW	0x0
0	AUTOIDLE	Internal Interface clock gating strategy 0x0: Interface clock is free-running 0x1: Automatic Interface clock gating strategy is applied, based on the Interconnect activity	RW	0x0

Table 11-32. Register Call Summary for Register GPMC_SYSCONFIG

General-Purpose Memory Controller

- Clocking, Reset, and Power Management Scheme: [0] [1] [2]
- GPMC Register Summary: [3]

Table 11-33. GPMC_SYSSTATUS

Address Offset	0x0000 0014		
Physical Address	0x6E00 0014	**Instance**	GPMC
Description	This register provides status information about the module, excluding the interrupt status information		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																								RESERVED							RESETDONE

Bits	Field Name	Description	Type	Reset
31:8	RESERVED	Reads returns 0	R	0x000000
7:1	RESERVED	Reads returns 0 (reserved for Interconnect-socket status information)	R	0x00
0	RESETDONE	Internal reset monitoring 0x0: Internal module reset in ongoing 0x1: Reset completed	R	0x-

Table 11-34. Register Call Summary for Register GPMC_SYSSTATUS

General-Purpose Memory Controller

- Clocking, Reset, and Power Management Scheme: [0]
- GPMC Register Summary: [1]

Table 11-35. GPMC_IRQSTATUS

Address Offset	0x0000 0018		
Physical Address	0x6E00 0018	**Instance**	GPMC
Description	This interrupt status register regroups all the status of the module internal events that can generate an interrupt.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																				WAIT3EDGEDETECTIONSTATUS	WAIT2EDGEDETECTIONSTATUS	WAIT1EDGEDETECTIONSTATUS	WAIT0EDGEDETECTIONSTATUS	RESERVED						TERMINALCOUNTSTATUS	FIFOEVENTSTATUS

Bits	Field Name	Description	Type	Reset
31:12	RESERVED	Write 0s for future compatibility. Read returns 0s.	RW	0x00000
11	WAIT3EDGEDETECTION STATUS	Status of the Wait3 Edge Detection interrupt Read 0x0: A transition on WAIT3 input pin has not been detected Write 0x0: WAIT3EDGEDETECTIONSTATUS bit unchanged Read 0x1: A transition on WAIT3 input pin has been detected Write 0x1: WAIT3EDGEDETECTIONSTATUS bit is reset	RW	0x0
10	WAIT2EDGEDETECTION STATUS	Status of the Wait2 Edge Detection interrupt Read 0x0: A transition on WAIT2 input pin has not been detected Write 0x0: WAIT2EDGEDETECTIONSTATUS bit unchanged Read 0x1: A transition on WAIT2 input pin has been detected Write 0x1: WAIT2EDGEDETECTIONSTATUS bit is reset	RW	0x0
9	WAIT1EDGEDETECTION STATUS	Status of the Wait1 Edge Detection interrupt Read 0x0: A transition on WAIT1 input pin has not been detected Write 0x0: WAIT1EDGEDETECTIONSTATUS bit unchanged Read 0x1: A transition on WAIT1 input pin has been detected Write 0x1: WAIT1EDGEDETECTIONSTATUS bit is reset	RW	0x0
8	WAIT0EDGEDETECTION STATUS	Status of the Wait0 Edge Detection interrupt Read 0x0: A transition on WAIT0 input pin has not been detected Write 0x0: WAIT0EDGEDETECTIONSTATUS bit unchanged Read 0x1: A transition on WAIT0 input pin has been detected Write 0x1: WAIT0EDGEDETECTIONSTATUS bit is reset	RW	0x0
7:2	RESERVED	Write 0s for future compatibility. Read returns 0s.	RW	0x00
1	TERMINALCOUNTSTATUS	Status of the TerminalCountEvent interrupt Read 0x0: Indicates that CountValue is greater than 0 Write 0x0: TERMINALCOUNTSTATUS bit unchanged Read 0x1: Indicates that CountValue is equal to 0 Write 0x1: TERMINALCOUNTSTATUS bit is reset	RW	0x0

Bits	Field Name	Description	Type	Reset
0	FIFOEVENTSTATUS	Status of the FIFOEvent interrupt Read 0x0: Indicates than less than FIFOThreshold bytes are available in prefetch mode and less than FIFOThreshold bytes free places are available in write-posting mode. Write 0x0: FIFOEVENTSTATUS bit unchanged Read 0x1: Indicates than at least FIFOThreshold bytes are available in prefetch mode and at least FIFOThreshold bytes free places are available in write-posting mode. Write 0x1: FIFOEVENTSTATUS bit is reset	RW	0x0

Table 11-36. Register Call Summary for Register GPMC_IRQSTATUS

General-Purpose Memory Controller

- NAND Device Basic Programming Model: [0] [1] [2] [3] [4] [5] [6]
- GPMC Register Summary: [7]

Table 11-37. GPMC_IRQENABLE

Address Offset	0x0000 001C		
Physical Address	0x6E00 001C	**Instance**	GPMC
Description	The interrupt enable register allows to mask/unmask the module internal sources of interrupt, on a event-by-event basis.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																				WAIT3EDGEDETECTIONENABLE	WAIT2EDGEDETECTIONENABLE	WAIT1EDGEDETECTIONENABLE	WAIT0EDGEDETECTIONENABLE	RESERVED						TERMINALCOUNTEVENTENABLE	FIFOEVENTENABLE

Bits	Field Name	Description	Type	Reset
31:12	RESERVED	Write 0s for future compatibility. Read returns 0s.	RW	0x00000
11	WAIT3EDGEDETECTION ENABLE	Enables the Wait3 Edge Detection interrupt 0x0: Wait3EdgeDetection interrupt is masked 0x1: Wait3EdgeDetection event generates an interrupt if occurs	RW	0x0
10	WAIT2EDGEDETECTION ENABLE	Enables the Wait2 Edge Detection interrupt 0x0: Wait2EdgeDetection interrupt is masked 0x1: Wait2EdgeDetection event generates an interrupt if occurs	RW	0x0
9	WAIT1EDGEDETECTION ENABLE	Enables the Wait1 Edge Detection interrupt 0x0: Wait1EdgeDetection interrupt is masked 0x1: Wait1EdgeDetection event generates an interrupt if occurs	RW	0x0
8	WAIT0EDGEDETECTION ENABLE	Enables the Wait0 Edge Detection interrupt 0x0: Wait0EdgeDetection interrupt is masked 0x1: Wait0EdgeDetection event generates an interrupt if occurs	RW	0x0
7:2	RESERVED	Write 0s for future compatibility. Read returns 0s.	RW	0x00

Bits	Field Name	Description	Type	Reset
1	TERMINALCOUNTEVENTENABLE	Enables TerminalCountEvent interrupt issuing in pre-fetch or write-posting mode 0x0: TerminalCountEvent interrupt is masked 0x1: TerminalCountEvent interrupt is not masked	RW	0x0
0	FIFOEVENTENABLE	Enables the FIFOEvent interrupt 0x0: FIFOEvent interrupt is masked 0x1: FIFOEvent interrupt is not masked	RW	0x0

Table 11-38. Register Call Summary for Register GPMC_IRQENABLE

General-Purpose Memory Controller

- NAND Device Basic Programming Model: [0] [1] [2] [3] [4] [5]
- GPMC Register Summary: [6]

Table 11-39. GPMC_TIMEOUT_CONTROL

Address Offset	0x0000 0040		
Physical Address	0x6E00 0040	**Instance**	GPMC
Description	The GPMC_TIMEOUT_CONTROL register allows the user to set the start value of the timeout counter		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																TIMEOUTSTARTVALUE												RESERVED			TIMEOUTENABLE

Bits	Field Name	Description	Type	Reset
31:13	RESERVED	Write 0s for future compatibility. Read returns 0s.	RW	0x00000
12:4	TIMEOUTSTARTVALUE	Start value of the time-out counter 0x000: Zero GPMC_FCLK cycle 0x001: One GPMC_FCLK cycle ... 0x1FF: 511 GPMC_FCLK cycles	RW	0x1FF
3:1	RESERVED	Write 0s for future compatibility. Read returns 0s.	RW	0x0
0	TIMEOUTENABLE	Enable bit of the TimeOut feature 0x0: TimeOut feature is disabled 0x1: TimeOut feature is enabled	RW	0x0

Table 11-40. Register Call Summary for Register GPMC_TIMEOUT_CONTROL

General-Purpose Memory Controller

- Error Handling: [0] [1]
- GPMC Register Summary: [2]
- GPMC Register Description: [3]

Table 11-41. GPMC_ERR_ADDRESS

Address Offset	0x0000 0044		
Physical Address	0x6E00 0044	**Instance**	GPMC
Description	The GPMC_ERR_ADDRESS register stores the address of the illegal access when an error occurs		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								ILLEGALADD																							

Bits	Field Name	Description	Type	Reset
31	RESERVED	Write 0s for future compatibility. Read returns 0.	RW	0x0
30:0	ILLEGALADD	Address of illegal access A30: 0 for memory region, 1 for GPMC register region A29-A0: 1 GBytes max	R	0x00000000

Table 11-42. Register Call Summary for Register GPMC_ERR_ADDRESS

General-Purpose Memory Controller

- Error Handling: [0]
- GPMC Register Summary: [1]
- GPMC Register Description: [2]

Table 11-43. GPMC_ERR_TYPE

Address Offset	0x0000 0048		
Physical Address	0x6E00 0048	**Instance**	GPMC
Description	The GPMC_ERR_TYPE register stores the type of error when an error occurs		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																					ILLEGALMCMD			RESERVED			ERRORNOTSUPPADD	ERRORNOTSUPPMCMD	ERRORTIMEOUT	RESERVED	ERRORVALID

Bits	Field Name	Description	Type	Reset
31:11	RESERVED	Write 0s for future compatibility. Read returns 0s.	RW	0x000000
10:8	ILLEGALMCMD	System Command of the transaction that caused the error	R	0x0
7:5	RESERVED	Write 0s for future compatibility. Read returns 0s.	RW	0x0
4	ERRORNOTSUPPADD	Not supported Address error 0x0: No error occurs 0x1: The error is due to a non supported Address	R	0x0
3	ERRORNOTSUPPMCMD	Not supported Command error 0x0: No error occurs 0x1: The error is due to a non supported Command	R	0x0

Bits	Field Name	Description	Type	Reset
2	ERRORTIMEOUT	Time-out error 0x0: No error occurs 0x1: The error is due to a time out	R	0x0
1	RESERVED	Write 0s for future compatibility. Read returns 0.	RW	0x0
0	ERRORVALID	Error validity status - Must be explicitly cleared with a write 1 transaction 0x0: All error fields no longer valid 0x1: Error detected and logged in the other error fields	RW	0x0

Table 11-44. Register Call Summary for Register GPMC_ERR_TYPE

General-Purpose Memory Controller

- Error Handling: [0] [1] [2]
- GPMC Register Summary: [3]
- GPMC Register Description: [4]

Table 11-45. GPMC_CONFIG

Address Offset	0x0000 0050		
Physical Address	0x6E00 0050	**Instance**	GPMC
Description	The configuration register allows global configuration of the GPMC		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																				WAIT3PINPOLARITY	WAIT2PINPOLARITY	WAIT1PINPOLARITY	WAIT0PINPOLARITY	RESERVED			WRITEPROTECT	RESERVED		LIMITEDADDRESS	NANDFORCEPOSTEDWRITE

Bits	Field Name	Description	Type	Reset
31:12	RESERVED	Write 0s for future compatibility. Read returns 0s.	RW	0x00000
11	WAIT3PINPOLARITY	Selects the polarity of input pin WAIT3 0x0: WAIT3 active low 0x1: WAIT3 active high	RW	0x1
10	WAIT2PINPOLARITY	Selects the polarity of input pin WAIT2 0x0: WAIT2 active low 0x1: WAIT2 active high	RW	0x0
9	WAIT1PINPOLARITY	Selects the polarity of input pin WAIT1 0x0: WAIT1 active low 0x1: WAIT1 active high	RW	0x1
8	WAIT0PINPOLARITY	Selects the polarity of input pin WAIT0 0x0: WAIT0 active low 0x1: WAIT0 active high	RW	0x0
7:5	RESERVED	Write 0s for future compatibility. Read returns 0s.	RW	0x0
4	WRITEPROTECT	Controls the WP output pin level 0x0: WP output pin is low 0x1: WP output pin is high	RW	0x0

Bits	Field Name	Description	Type	Reset
3:2	RESERVED	Write 0s for future compatibility. Read returns 0s.	RW	0x0
1	LIMITEDADDRESS	Limited Address device support 0x0: No effect 0x1: A26-A11 are not modified during an external memory access.	RW	0x0
0	NANDFORCEPOSTEDWRITE	Enables the Force Posted Write feature to NAND Cmd/Add/Data location 0x0: Disables Force Posted Write 0x1: Enables Force Posted Write	RW	0x0

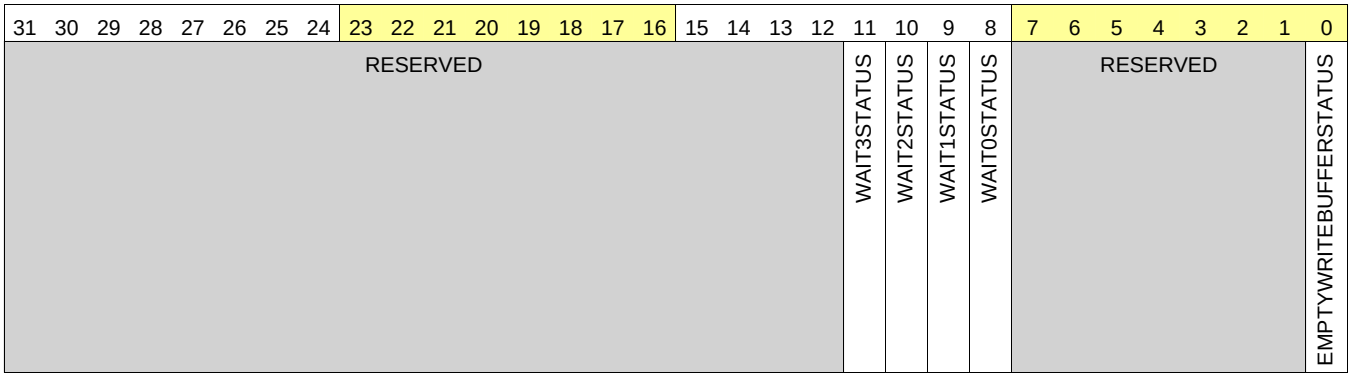
Table 11-46. Register Call Summary for Register GPMC_CONFIG

General-Purpose Memory Controller

- GPMC Environment: [0]
- GPMC Address and Data Bus: [1] [2]
- WAIT Pin Monitoring Control: [3]
- WRITE PROTECT (nWP): [4]
- Asynchronous Access Description: [5] [6]
- NAND Device Basic Programming Model: [7] [8]
- GPMC Register Summary: [9]

Table 11-47. GPMC_STATUS

Address Offset	0x0000 0054		
Physical Address	0x6E00 0054	**Instance**	GPMC
Description	The status register provides global status bits of the GPMC		
Type	RW		



Bits	Field Name	Description	Type	Reset
31:12	RESERVED	Write 0s for future compatibility. Read returns 0s.	RW	0x00000
11	WAIT3STATUS	Is a copy of input pin WAIT3. (Reset value is WAIT3 input pin sampled at IC reset) 0x0: WAIT3 asserted (inactive state) 0x1: WAIT3 de-asserted	R	0x-
10	WAIT2STATUS	Is a copy of input pin WAIT2. (Reset value is WAIT2 input pin sampled at IC reset) 0x0: WAIT2 asserted (inactive state) 0x1: WAIT2 de-asserted	R	0x-
9	WAIT1STATUS	Is a copy of input pin WAIT1. (Reset value is WAIT1 input pin sampled at IC reset) 0x0: WAIT1 asserted (inactive state) 0x1: WAIT1 de-asserted	R	0x-

Bits	Field Name	Description	Type	Reset
8	WAIT0STATUS	Is a copy of input pin WAIT0. (Reset value is WAIT0 input pin sampled at IC reset) 0x0: WAIT0 asserted (inactive state) 0x1: WAIT0 de-asserted	R	0x-
7:1	RESERVED	Write 0s for future compatibility. Reads returns 0	RW	0x00
0	EMPTYWRITEBUFFERSTATUS	Stores the empty status of the write buffer 0x0: Write Buffer is not empty 0x1: Write Buffer is empty	R	0x1

Table 11-48. Register Call Summary for Register GPMC_STATUS

General-Purpose Memory Controller

- NAND Device Basic Programming Model: [0] [1] [2] [3]
- GPMC Register Summary: [4]

Table 11-49. GPMC_CONFIG1_i

Address Offset	0x0000 0060 + (0x0000 0030 * i)	**Index**	i = 0 to 7
Physical Address	0x6E00 0060 + (0x0000 0030 * i)	**Instance**	GPMC
Description	The configuration register 1 sets signal control parameters per chip-select		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
WRAPBURST		READMULTIPLE		READTYPE		WRITEMULTIPLE		WRITETYPE		CLKACTIVATIONTIME				ATTACHEDDEVICEPAGELENGTH		WAITREADMONITORING		WAITWRITEMONITORING		RESERVED		WAITMONITORINGTIME				WAITPINSELECT		RESERVED		DEVICESIZE		DEVICETYPE		MUXADDDATA		RESERVED				TIMEPARAGRANULARITY		RESERVED		GPMCFCLKDIVIDER	

Bits	Field Name	Description	Type	Reset
31	WRAPBURST	Enables the wrapping burst capability. Must be set if the attached device is configured in wrapping burst 0x0: Synchronous wrapping burst not supported 0x1: Synchronous wrapping burst supported	RW	0x0
30	READMULTIPLE	Selects the read single or multiple access 0x0: Single access 0x1: Multiple access (burst if synchronous, page if asynchronous)	RW	0x0
29	READTYPE	Selects the read mode operation 0x0: Read Asynchronous 0x1: Read Synchronous	RW	0x0
28	WRITEMULTIPLE	Selects the write single or multiple access 0x0: Single access 0x1: Multiple access (burst if synchronous, considered as single if asynchronous)	RW	0x0

Bits	Field Name	Description	Type	Reset
27	WRITETYPE	Selects the write mode operation 0x0: Write Asynchronous 0x1: Write Synchronous	RW	0x0
26:25	CLKACTIVATIONTIME	Output GPMC_CLK activation time 0x0: First rising edge of GPMC_CLK at start access time 0x1: First rising edge of GPMC_CLK one GPMC_FCLK cycle after start access time 0x2: First rising edge of GPMC_CLK two GPMC_FCLK cycles after start access time 0x3: Reserved	RW	0x0
24:23	ATTACHEDDEVICEPAGE LENGTH	Specifies the attached device page (burst) length 0x0: 4 Words 0x1: 8 Words 0x2: 16 Words 0x3: Reserved (1 Word = Interface size)	RW	0x0
22	WAITREADMONITORING	Selects the Wait monitoring configuration for Read accesses (Reset value is BOOTWAITEN input pin sampled at IC reset) 0x0: Wait pin is not monitored for read accesses 0x1: Wait pin is monitored for read accesses	RW	0x-
21	WAITWRITEMONITORING	Selects the Wait monitoring configuration for Write accesses 0x0: Wait pin is not monitored for write accesses 0x1: Wait pin is monitored for write accesses	RW	0x0
20	RESERVED	Write 0s for future compatibility. Read returns 0.	RW	0x0
19:18	WAITMONITORINGTIME	Selects input pin Wait monitoring time 0x0: Wait pin is monitored with valid data 0x1: Wait pin is monitored one GPMC_CLK cycle before valid data 0x2: Wait pin is monitored two GPMC_CLK cycle before valid data 0x3: Reserved	RW	0x0
17:16	WAITPINSELECT	Selects the input WAIT pin for this chip-select (Reset value is BOOTWAITSELECT input pin sampled at IC reset for CS0 and 0 for CS1-7) 0x0: Wait input pin is WAIT0 0x1: Wait input pin is WAIT1 0x2: Wait input pin is WAIT2 0x3: Wait input pin is WAIT3	RW	0x-
15:14	RESERVED	Write 0s for future compatibility. Reads returns 0	RW	0x0
13:12	DEVICESIZE	Selects the device size attached (Reset value is BOOTDEVICESIZE input pin sampled at IC reset for CS0 and 0x1 for CS1 to CS7) 0x0: 8 bit 0x1: 16 bit 0x2: Reserved 0x3: Reserved	RW	0x-
11:10	DEVICETYPE	Selects the attached device type 0x0: NOR Flash like, asynchronous and synchronous devices 0x1: Reserved 0x2: NAND Flash like devices, stream mode 0x3: Reserved	RW	0x0

Bits	Field Name	Description	Type	Reset
9	MUXADDDATA	Enables the Address and data multiplexed protocol (Reset value is CS0MUXDEVICE input pin sampled at IC reset for CS0 and 0 for CS1-7) 0x0: Non Multiplexed attached device 0x1: Address and data multiplexed attached device	RW	0x-
8:5	RESERVED	Write 0s for future compatibility. Read returns 0s.	RW	0x0
4	TIMEPARAGRANULARITY	Signals timing latencies scalar factor (Rd/WrCycleTime, Rd/WrAccessTime, PageBurstAccessTime, CSOnTime, CSRd/WrOffTime, ADVOnTime, ADVRd/WrOffTime, OEOnTime, OEOffTime, WEOnTime, WEOffTime, Cycle2CycleDelay, BusTurnAround, TimeOutStartValue) 0x0: x1 latencies 0x1: x2 latencies	RW	0x0
3:2	RESERVED	Write 0s for future compatibility. Read returns 0s.	RW	0x0
1:0	GPMCFCLKDIVIDER	Divides the GPMC_FCLK clock 0x0: GPMC_CLK frequency = GPMC_FCLK frequency 0x1: GPMC_CLK frequency = GPMC_FCLK frequency / 2 0x2: GPMC_CLK frequency = GPMC_FCLK frequency / 3 0x3: GPMC_CLK frequency = GPMC_FCLK frequency /4	RW	0x0

Table 11-50. Register Call Summary for Register GPMC_CONFIG1_i

General-Purpose Memory Controller

- GPMC Environment: [0]
- Clocking, Reset, and Power Management Scheme: [1]
- GPMC Address and Data Bus: [2] [3] [4] [5] [6] [7] [8] [9] [10] [11] [12]
- L3 Interconnect Interface: [13] [14]
- Access Protocol Configuration: [15] [16] [17] [18] [19] [20] [21] [22] [23] [24]
- Timing Setting: [25] [26] [27] [28] [29] [30] [31] [32]
- WAIT Pin Monitoring Control: [33] [34] [35] [36] [37] [38] [39] [40] [41] [42] [43] [44] [45] [46] [47] [48]
- Asynchronous Access Description: [49] [50] [51] [52] [53] [54] [55] [56]
- Synchronous Access: [57] [58] [59] [60] [61] [62] [63]
- pSRAM Basic Programming Model: [64]
- NAND Device Basic Programming Model: [65] [66] [67] [68] [69] [70] [71] [72] [73] [74] [75] [76] [77] [78] [79] [80] [81] [82] [83] [84]
- GPMC Register Summary: [85]

Table 11-51. GPMC_CONFIG2_i

Address Offset	0x0000 0064 + (0x0000 0030 * i)	**Index**	i = 0 to 7
Physical Address	0x6E00 0064 + (0x0000 0030 * i)	**Instance**	GPMC
Description	CS signal timing parameter configuration		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED											CSWROFFTIME					RESERVED			CSRDOFFTIME					CSEXTRADELAY	RESERVED					CSONTIME			

Bits	Field Name	Description	Type	Reset
31:21	RESERVED	Write 0s for future compatibility. Reads returns 0	RW	0x000
20:16	CSWROFFTIME	CS i de-assertion time from start cycle time for write accesses 0x00: 0 GPMC_FCLK cycle 0x01: 1 GPMC_FCLK cycle ... 0x1F: 31 GPMC_FCLK cycles	RW	0x10
15:13	RESERVED	Write 0s for future compatibility. Read returns 0s.	RW	0x0
12:8	CSRDOFFTIME	CS i de-assertion time from start cycle time for read accesses 0x00: 0 GPMC_FCLK cycle 0x01: 1 GPMC_FCLK cycle ... 0x1F: 31 GPMC_FCLK cycles	RW	0x10
7	CSEXTRADELAY	CS i Add Extra Half GPMC_FCLK cycle 0x0: CS i Timing control signal is not delayed 0x1: CS i Timing control signal is delayed of half GPMC_FCLK clock cycle	RW	0x0
6:4	RESERVED	Write 0s for future compatibility. Read returns 0s.	RW	0x0
3:0	CSONTIME	CS i assertion time from start cycle time 0x0: 0 GPMC_FCLK cycle 0x1: 1 GPMC_FCLK cycle ... 0xF: 15 GPMC_FCLK cycles	RW	0x1

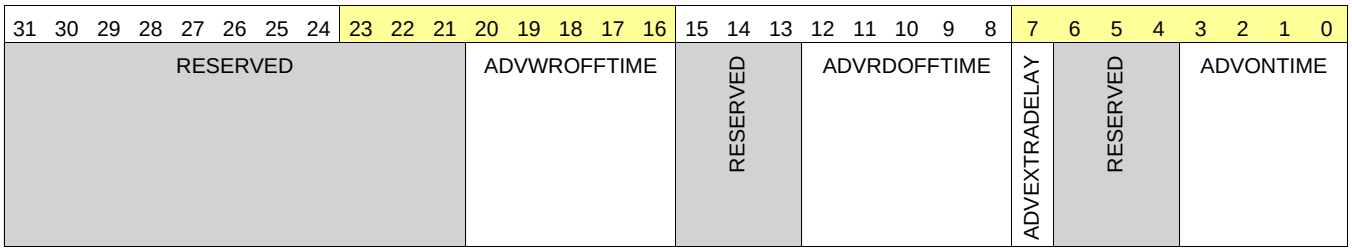
Table 11-52. Register Call Summary for Register GPMC_CONFIG2_i

General-Purpose Memory Controller

- Timing Setting: [0] [1] [2] [3]
- Asynchronous Access Description: [4] [5] [6] [7] [8] [9]
- Synchronous Access: [10] [11] [12] [13] [14] [15]
- GPMC Register Summary: [16]

Table 11-53. GPMC_CONFIG3_i

Address Offset	0x0000 0068 + (0x0000 0030 * i)	**Index**	i = 0 to 7
Physical Address	0x6E00 0068 + (0x0000 0030 * i)	**Instance**	GPMC
Description	nADV signal timing parameter configuration		
Type	RW		



Bits	Field Name	Description	Type	Reset
31:21	RESERVED	Write 0s for future compatibility. Read returns 0s.	RW	0x000
20:16	ADVWROFFTIME	nADV de-assertion time from start cycle time for write accesses 0x00: 0 GPMC_FCLK cycle 0x01: 1 GPMC_FCLK cycle ... 0x1F: 31 GPMC_FCLK cycles	RW	0x02
15:13	RESERVED	Write 0s for future compatibility. Read returns 0s.	RW	0x0

Bits	Field Name	Description	Type	Reset
12:8	ADVRDOFFTIME	nADV de-assertion time from start cycle time for read accesses 0x00: 0 GPMC_FCLK cycle 0x01: 1 GPMC_FCLK cycle ... 0x1F: 31 GPMC_FCLK cycles	RW	0x02
7	ADVEXTRADELAY	nADV Add Extra Half GPMC_FCLK cycle 0x0: nADV Timing control signal is not delayed 0x1: nADV Timing control signal is delayed of half GPMC_FCLK clock cycle	RW	0x0
6:4	RESERVED	Write 0s for future compatibility. Read returns 0s.	RW	0x0
3:0	ADVONTIME	nADV assertion time from start cycle time 0x0: 0 GPMC_FCLK cycle 0x1: 1 GPMC_FCLK cycle ... 0xF: 15 GPMC_FCLK cycles	RW	0x1

Table 11-54. Register Call Summary for Register GPMC_CONFIG3_i

General-Purpose Memory Controller

- Timing Setting: [0] [1] [2] [3]
- Asynchronous Access Description: [4] [5] [6] [7] [8] [9]
- Synchronous Access: [10] [11] [12] [13] [14] [15]
- GPMC Register Summary: [16]

Table 11-55. GPMC_CONFIG4_i

Address Offset	0x0000 006C + (0x0000 0030 * i)	**Index**	i = 0 to 7
Physical Address	0x6E00 006C + (0x0000 0030 * i)	**Instance**	GPMC
Description	nWE and nOE signals timing parameter configuration		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED			WEOFFTIME					WEEXTRADELAY	RESERVED			WEONTIME				RESERVED			OEOFFTIME					OEEXTRADELAY	RESERVED			OEONTIME			

Bits	Field Name	Description	Type	Reset
31:29	RESERVED	Write 0s for future compatibility. Read returns 0s.	RW	0x0
28:24	WEOFFTIME	nWE de-assertion time from start cycle time 0x00: 0 GPMC_FCLK cycle 0x01: 1 GPMC_FCLK cycle ... 0x1F: 31 GPMC_FCLK cycles	RW	0x10
23	WEEXTRADELAY	nWE Add Extra Half GPMC_FCLK cycle 0x0: nWE Timing control signal is not delayed 0x1: nWE Timing control signal is delayed of half GPMC_FCLK clock cycle	RW	0x0
22:20	RESERVED	Write 0s for future compatibility. Read returns 0s.	RW	0x0
19:16	WEONTIME	nWE assertion time from start cycle time 0x0: 0 GPMC_FCLK cycle 0x1: 1 GPMC_FCLK cycle ... 0xF: 15 GPMC_FCLK cycles	RW	0x3
15:13	RESERVED	Write 0s for future compatibility. Read returns 0s.	RW	0x0

Bits	Field Name	Description	Type	Reset
12:8	OEOFFTIME	nOE de-assertion time from start cycle time 0x00: 0 GPMC_FCLK cycle 0x01: 1 GPMC_FCLK cycle ... 0x1F: 31 GPMC_FCLK cycles	RW	0x10
7	OEEXTRADELAY	nOE Add Extra Half GPMC_FCLK cycle 0x0: nOE Timing control signal is not delayed 0x1: nOE Timing control signal is delayed of half GPMC_FCLK clock cycle	RW	0x0
6:4	RESERVED	Write 0s for future compatibility. Read returns 0s.	RW	0x0
3:0	OEONTIME	nOE assertion time from start cycle time 0x0: 0 GPMC_FCLK cycle 0x1: 1 GPMC_FCLK cycle ... 0xF: 15 GPMC_FCLK cycles	RW	0x3

Table 11-56. Register Call Summary for Register GPMC_CONFIG4_i

General-Purpose Memory Controller

- Timing Setting: [0] [1] [2] [3] [4] [5]
- Asynchronous Access Description: [6] [7] [8] [9] [10] [11]
- Synchronous Access: [12] [13] [14] [15] [16] [17]
- NAND Device Basic Programming Model: [18]
- GPMC Register Summary: [19]

Table 11-57. GPMC_CONFIG5_i

Address Offset	0x0000 0070 + (0x0000 0030 * i)	**Index**	i = 0 to 7
Physical Address	0x6E00 0070 + (0x0000 0030 * i)	**Instance**	GPMC
Description	RdAccessTime and CycleTime timing parameters configuration		
Type	RW		

31 30 29 28	27 26 25 24	23 22 21	20 19 18 17 16	15 14 13	12 11 10 9 8	7 6 5	4 3 2 1 0
RESERVED	PAGEBURSTACCESSTIME	RESERVED	RDACCESSTIME	RESERVED	WRCYCLETIME	RESERVED	RDCYCLETIME

Bits	Field Name	Description	Type	Reset
31:28	RESERVED	Write 0s for future compatibility. Read returns 0s.	RW	0x0
27:24	PAGEBURSTACCESSTIME	Delay between successive words in a multiple access 0x0: 0 GPMC_FCLK cycle 0x1: 1 GPMC_FCLK cycle ... 0xF: 15 GPMC_FCLK cycles	RW	0x1
23:21	RESERVED	Write 0s for future compatibility. Read returns 0s.	RW	0x0
20:16	RDACCESSTIME	Delay between start cycle time and first data valid 0x00: 0 GPMC_FCLK cycle 0x01: 1 GPMC_FCLK cycle ... 0x1F: 31 GPMC_FCLK cycles	RW	0x0F
15:13	RESERVED	Write 0s for future compatibility. Reads returns 0	RW	0x0

Bits	Field Name	Description	Type	Reset
12:8	WRCYCLETIME	Total write cycle time 0x00: 0 GPMC_FCLK cycle 0x01: 1 GPMC_FCLK cycle ... 0x1F: 31 GPMC_FCLK cycles	RW	0x11
7:5	RESERVED	Write 0s for future compatibility. Read returns 0s.	RW	0x0
4:0	RDCYCLETIME	Total read cycle time 0x00: 0 GPMC_FCLK cycle 0x01: 1 GPMC_FCLK cycle ... 0x1F: 31 GPMC_FCLK cycles	RW	0x11

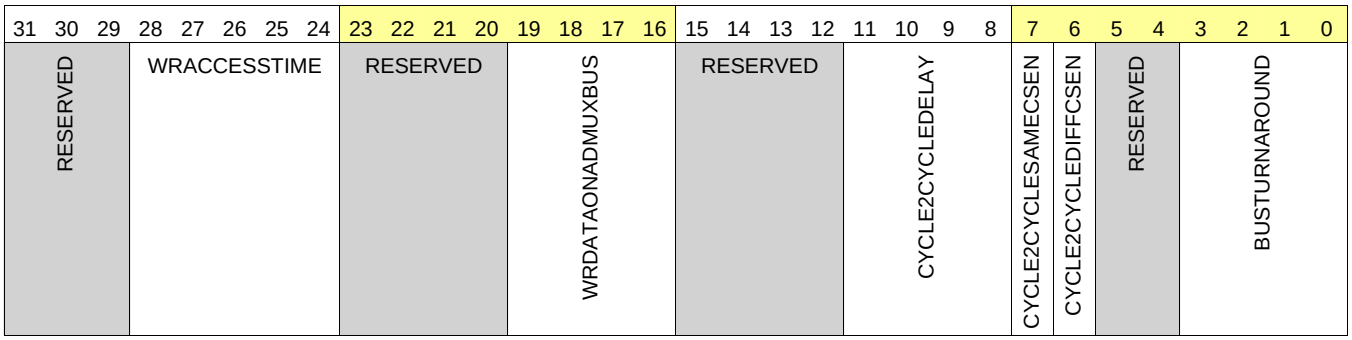
Table 11-58. Register Call Summary for Register GPMC_CONFIG5_i

General-Purpose Memory Controller

- Timing Setting: [0] [1] [2] [3] [4]
- Asynchronous Access Description: [5] [6] [7] [8] [9] [10]
- Synchronous Access: [11] [12] [13] [14] [15] [16]
- GPMC Register Summary: [17]

Table 11-59. GPMC_CONFIG6_i

Address Offset	0x0000 0074 + (0x0000 0030 * i)	**Index**	i = 0 to 7
Physical Address	0x6E00 0074 + (0x0000 0030 * i)	**Instance**	GPMC
Description	WrAccessTime, WrDataOnADmuxBus, Cycle2Cycle and BusTurnAround parameters configuration		
Type	RW		



Bits	Field Name	Description	Type	Reset
31	RESERVED	TI Internal use - Do not modify.	RW	0x1
30:29	RESERVED	Write 0s for future compatibility. Read returns 0s.	RW	0x0
28:24	WRACCESSTIME	Delay from start access time to the GPMC_FCLK rising edge corresponding the the GPMC_CLK rising edge used by the attached memory for the first data capture 0x00: 0 GPMC_FCLK cycle 0x01: 1 GPMC_FCLK cycle ... 0x1F: 31 GPMC_FCLK cycles	RW	0x0F
23:20	RESERVED	Write 0s for future compatibility. Read returns 0s.	RW	0x0
19:16	WRDATAONADMUXBUS	Specifies on which GPMC_FCLK rising edge the first data is driven in the add/data mux bus	RW	0x3
15:12	RESERVED	Write 0s for future compatibility. Read returns 0s.	RW	0x0
11:8	CYCLE2CYCLEDELAY	Chip-select high pulse delay between successive accesses 0x0: 0 GPMC_FCLK cycle 0x1: 1 GPMC_FCLK cycle ... 0xF: 15 GPMC_FCLK cycles	RW	0x0

Bits	Field Name	Description	Type	Reset
7	CYCLE2CYCLESAMECSEN	Add CYCLE2CYCLEDELAY between successive accesses to the same CS (any access type) 0x0: No delay between the two accesses 0x1: Add CYCLE2CYCLEDELAY	RW	0x0
6	CYCLE2CYCLEDIFFCSEN	Add CYCLE2CYCLEDELAY between successive accesses to a different CS (any access type) 0x0: No delay between the two accesses 0x1: Add CYCLE2CYCLEDELAY	RW	0x0
5:4	RESERVED	Write 0s for future compatibility. Reads returns 0	RW	0x0
3:0	BUSTURNAROUND	Bus turn around latency between successive accesses to the same CS (read to write) or to a different CS (read to read and read to write) 0x0: 0 GPMC_FCLK cycle 0x1: 1 GPMC_FCLK cycle ... 0xF: 15 GPMC_FCLK cycles	RW	0x0

Table 11-60. Register Call Summary for Register GPMC_CONFIG6_i

General-Purpose Memory Controller

- Timing Setting: [0] [1] [2] [3]
- WAIT Pin Monitoring Control: [4] [5] [6] [7] [8]
- Asynchronous Access Description: [9] [10] [11]
- Synchronous Access: [12] [13] [14]
- GPMC Register Summary: [15]

Table 11-61. GPMC_CONFIG7_i

Address Offset	0x0000 0078 + (0x0000 0030 * i)	**Index**	i = 0 to 7
Physical Address	0x6E00 0078 + (0x0000 0030 * i)	**Instance**	GPMC
Description	CS address mapping configuration		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																				MASKADDRESS				RESERVED	CSVALID	BASEADDRESS					

Bits	Field Name	Description	Type	Reset
31:12	RESERVED	Write 0s for future compatibility. Read returns 0s.	RW	0x00000
11:8	MASKADDRESS	CS mask address. 0x1000: Chip-select size of 128 Mbytes 0x1100: Chip-select size of 64 Mbytes 0x1110: Chip-select size of 32 Mbytes 0x1111: Chip-select size of 16 Mbytes Other values must be avoided as they create holes in the chip-select address space.	RW	0xF
7	RESERVED	Write 0s for future compatibility. Read returns 0.	RW	0x0
6	CSVALID	CS enable 0x0: CS disabled 0x1: CS enabled	RW	See (1)

(1) Reset value is 0x1 for CS0 and 0x0 for CS1 to CS7

Bits	Field Name	Description	Type	Reset
5:0	BASEADDRESS	CSi base address (16M bytes minimum granularity) bits [5:0] corresponds to A29, A28, A27, A26, A25, and A24. See Figure 11-6	RW	0x00

Table 11-62. Register Call Summary for Register GPMC_CONFIG7_i

General-Purpose Memory Controller

- Chip-Select Base Address and Region Size Configuration: [0] [1] [2]
- NAND Device Basic Programming Model: [3]
- GPMC Register Summary: [4]

Table 11-63. GPMC_NAND_COMMAND_i

Address Offset	0x0000 007C + (0x0000 0030 * i)	**Index**	i = 0 to 7
Physical Address	0x6E00 007C + (0x0000 0030 * i)	**Instance**	GPMC
Description	This register is not a true register, just an address location.		
Type	W		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GPMC_NAND_COMMAND																															

Bits	Field Name	Description	Type	Reset
31:0	GPMC_NAND_COMMAND	This register is not a true register, just an address location.	W	n/a

Table 11-64. Register Call Summary for Register GPMC_NAND_COMMAND_i

General-Purpose Memory Controller

- NAND Device Basic Programming Model: [0] [1] [2] [3] [4] [5] [6]
- GPMC Register Summary: [7]

Table 11-65. GPMC_NAND_ADDRESS_i

Address Offset	0x0000 0080 + (0x0000 0030 * i)	**Index**	i = 0 to 7
Physical Address	0x6E00 0080 + (0x0000 0030 * i)	**Instance**	GPMC
Description	This register is not a true register, just an address location.		
Type	W		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GPMC_NAND_ADDRESS																															

Bits	Field Name	Description	Type	Reset
31:0	GPMC_NAND_ADDRESS	This register is not a true register, just an address location.	W	n/a

Table 11-66. Register Call Summary for Register GPMC_NAND_ADDRESS_i

General-Purpose Memory Controller

- NAND Device Basic Programming Model: [0] [1] [2] [3] [4] [5] [6]
- GPMC Register Summary: [7]

Table 11-67. GPMC_NAND_DATA_i

Address Offset	0x0000 0084 + (0x0000 0030 * i)	**Index**	i = 0 to 7
Physical Address	0x6E00 0084 + (0x0000 0030 * i)	**Instance**	GPMC
Description	This register is not a true register, just an address location.		

Table 11-67. GPMC_NAND_DATA_i (continued)

Type	RW

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GPMC_NAND_DATA																															

Bits	Field Name	Description	Type	Reset
31:0	GPMC_NAND_DATA	This register is not a true register, just an address location.	W	n/a

Table 11-68. Register Call Summary for Register GPMC_NAND_DATA_i

General-Purpose Memory Controller

- NAND Device Basic Programming Model: [0] [1] [2] [3] [4] [5]
- GPMC Register Summary: [6]
- GPMC Register Description: [7] [8]

Table 11-69. GPMC_PREFETCH_CONFIG1

Address Offset	0x0000 01E0		
Physical Address	0x6E00 01E0	Instance	GPMC
Description	Prefetch engine configuration 1		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED	CYCLEOPTIMIZATION			ENABLEOPTIMIZEDACCESS	ENGINECSSELECTOR			PFPWENROUNDROBIN	RESERVED			PFPWWEIGHTEDPRIO				RESERVED	FIFOTHRESHOLD							ENABLEENGINE	RESERVED	WAITPINSELECTOR		SYNCHROMODE	DMAMODE	RESERVED	ACCESSMODE

Bits	Field Name	Description	Type	Reset
31	RESERVED	Write 0s for future compatibility. Read returns 0.	RW	0x0
30:28	CYCLEOPTIMIZATION	Define the number of GPMC_FCLK cycles to be subtracted from RdCycleTime, WrCycleTime, AccessTime, CSRdOffTime, CSWrOffTime, ADVRdOffTime, ADVWrOffTime, OEOffTime, WEOffTime 0x0: 0 GPMC_FCLK cycle 0x1: 1 GPMC_FCLK cycle ... 0x7: 7 GPMC_FCLK cycles	RW	0x0
27	ENABLEOPTIMIZEDACCESS	Enables access cycle optimization 0x0: Access cycle optimization is disabled 0x1: Access cycle optimization is enabled	RW	0x0
26:24	ENGINECSSELECTOR	Selects the CS where Prefetch Postwrite engine is active 0x0 corresponds to CS0 0x1: CS1 ... 0x7: CS7	RW	0x0
23	PFPWENROUNDROBIN	Enables the PFPW RoundRobin arbitration 0x0: Prefetch Postwrite engine round robin arbitration is disabled 0x1: Prefetch Postwrite engine round robin arbitration is enabled	RW	0x0

Bits	Field Name	Description	Type	Reset
22:20	RESERVED	Write 0s for future compatibility. Read returns 0s.	RW	0x0
19:16	PFPWWEIGHTEDPRIO	When an arbitration occurs between a DMA and a PFPW engine access, the DMA is always serviced. If the PFPWEnRoundRobin is enabled, 0x0: the next access is granted to the PFPW engine, 0x1: the two next accesses are granted to the PFPW engine, ..., 0xF: the 16 next accesses are granted to the PFPW engine.	RW	0x0
15	RESERVED	Write 0s for future compatibility. Read returns 0.	RW	0x0
14:8	FIFOTHRESHOLD	Selects the maximum number of bytes read from the FIFO or written to the FIFO by the host on a DMA or interrupt request 0x00: 0 byte 0x01: 1 byte ... 0x40: 64 bytes	RW	0x40
7	ENABLEENGINE	Enables the Prefetch Postwite engine 0x0: Prefetch Postwrite engine is disabled 0x1: Prefetch Postwrite engine is enabled	RW	0x0
6	RESERVED	Write 0s for future compatibility. Read returns 0.	RW	0x0
5:4	WAITPINSELECTOR	Select which wait pin edge detector should start the engine in synchronized mode 0x0: Selects Wait0EdgeDetection 0x1: Selects Wait1EdgeDetection 0x2: Selects Wait2EdgeDetection 0x3: Selects Wait3EdgeDetection	RW	0x0
3	SYNCHROMODE	Selects when the engine starts the access to CS 0x0: Engine starts the access to CS as soon as STARTENGINE is set 0x1: Engine starts the access to CS as soon as STARTENGINE is set AND wait to non wait edge detection on the selected wait pin	RW	0x0
2	DMAMODE	Selects interrupt synchronization or DMA request synchronization 0x0: Interrupt synchronization is enabled. Only interrupt line will be activated on FIFO threshold crossing. 0x1: DMA request synchronization is enabled. A DMA request protocol is used.	RW	0x0
1	RESERVED	Write 0s for future compatibility. Read returns 0.	RW	0x0
0	ACCESSMODE	Selects pre-fetch read or write-posting accesses 0x0: Pre-fetch read mode 0x1: Write-posting mode	RW	0x0

Table 11-70. Register Call Summary for Register GPMC_PREFETCH_CONFIG1

General-Purpose Memory Controller

- NAND Device Basic Programming Model: [0] [1] [2] [3] [4] [5] [6] [7] [8] [9] [10] [11] [12] [13] [14] [15] [16] [17] [18] [19] [20] [21] [22] [23] [24] [25] [26] [27] [28] [29] [30] [31]
- GPMC Register Summary: [32]

Table 11-71. GPMC_PREFETCH_CONFIG2

Address Offset	0x0000 01E4		
Physical Address	0x6E00 01E4	**Instance**	GPMC
Description	Prefetch engine configuration 2		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																TRANSFERCOUNT															

Bits	Field Name	Description	Type	Reset
31:14	RESERVED	Write 0s for future compatibility. Read returns 0s.	RW	0x00000
13:0	TRANSFERCOUNT	Selects the number of bytes to be read or written by the engine to the selected CS 0x0000: 0 byte 0x0001: 1 byte ... 0x2000: 8 Kbytes	RW	0x0000

Table 11-72. Register Call Summary for Register GPMC_PREFETCH_CONFIG2

General-Purpose Memory Controller

- NAND Device Basic Programming Model: [0] [1] [2] [3]
- GPMC Register Summary: [4]

Table 11-73. GPMC_PREFETCH_CONTROL

Address Offset	0x0000 01EC		
Physical Address	0x6E00 01EC	**Instance**	GPMC
Description	Prefetch engine control		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																															STARTENGINE

Bits	Field Name	Description	Type	Reset
31:1	RESERVED	Write 0s for future compatibility. Read returns 0s.	RW	0x00000000
0	STARTENGINE	Resets the FIFO pointer and starts the engine Read 0x0: Engine is stopped Write 0x0 stops the engine Read 0x1: Engine is running Write 0x1 resets the FIFO pointer to 0x0 in prefetch mode and 0x40 in postwrite mode and starts the engine	RW	0x0

Table 11-74. Register Call Summary for Register GPMC_PREFETCH_CONTROL

General-Purpose Memory Controller

- NAND Device Basic Programming Model: [0] [1] [2] [3] [4] [5] [6]
- GPMC Register Summary: [7]

Table 11-75. GPMC_PREFETCH_STATUS

Address Offset	0x0000 01F0		
Physical Address	0x6E00 01F0	**Instance**	GPMC
Description	Prefetch engine status		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED				FIFOPOINTER				RESERVED							FIFOTHRESHOLDSTATUS	RESERVED		COUNTVALUE													

Bits	Field Name	Description	Type	Reset
31	RESERVED	Write 0s for future compatibility. Read returns 0.	RW	0x0
30:24	FIFOPOINTER	Number of available bytes to be read or number of free empty byte places to be written 0x00: 0 byte available to be read or 0 free empty place to be written ... 0x40: 64 bytes available to be read or 64 empty places to be written	R	0x00
23:17	RESERVED	Write 0s for future compatibility. Read returns 0s.	RW	0x00
16	FIFOTHRESHOLDSTATUS	Set when FIFOPointer exceeds FIFOThreshold value 0x0: FIFOPointer smaller or equal to FIFOThreshold. Writing to this bit has no effect 0x1: FIFOPointer greater than FIFOThreshold. Writing to this bit has no effect	R	0x0
15:14	RESERVED	Write 0s for future compatibility. Read returns 0s.	RW	0x0
13:0	COUNTVALUE	Number of remaining bytes to be read or to be written by the engine according to the TransferCount value 0x0000: 0 byte remaining to be read or to be written 0x0001: 1 byte remaining to be read or to be written ... 0x2000: 8 Kbytes remaining to be read or to be written	R	0x0000

Table 11-76. Register Call Summary for Register GPMC_PREFETCH_STATUS

General-Purpose Memory Controller

- NAND Device Basic Programming Model: [0] [1] [2] [3] [4] [5] [6]
- GPMC Register Summary: [7]

Table 11-77. GPMC_ECC_CONFIG

Address Offset	0x0000 01F4		
Physical Address	0x6E00 01F4	**Instance**	GPMC
Description	ECC configuration		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED															ECCALGORITHM	RESERVED			ECCBCHT8	ECCWRAPMODE				ECC16B	ECCTOPSECTOR			ECCCS			ECCENABLE

Bits	Field Name	Description	Type	Reset
31:17	RESERVED	Write 0s for future compatibility. Read returns 0s.	RW	0x0000
16	ECCALGORITHM	ECC algorithm used 0x0: Hamming code 0x1: BCH code	RW	0x0
15:13	RESERVED	Write 0s for future compatibility. Read returns 0s.	RW	0x0
12	ECCBCHT8	Error correction capability used for BCH 0x0: Up to 4 bits error correction (t = 4) 0x1: Up to 8 bits error correction (t = 8)	RW	0x1
11:8	ECCWRAPMODE	Spare area organization definition for the BCH algorithm. See the BCH syndrome/parity calculator module functional specification for more details	RW	0x0
7	ECC16B	Selects an ECC calculated on 16 columns 0x0: ECC calculated on 8 columns 0x1: ECC calculated on 16 columns	RW	0x0
6:4	ECCTOPSECTOR	Number of sectors to process with the BCH algorithm 0x0: 1 sector (512kB page) 0x1: 2 sectors ... 0x3: 4 sectors (2kB page) ... 0x7: 8 sectors (4kB page)	RW	0x3
3:1	ECCCS	Selects the CS where ECC is computed 0x0: Chip-select 0 0x1: Chip-select 1 0x2: Chip-select 2 0x3: Chip-select 3 0x4: Chip-select 4 0x5: Chip-select 5 0x6: Chip-select 6 0x7: Chip-select 7	RW	0x0
0	ECCENABLE	Enables the ECC feature 0x0: ECC disabled 0x1: ECC enabled	RW	0x0

Table 11-78. Register Call Summary for Register GPMC_ECC_CONFIG

General-Purpose Memory Controller

- Error correction code engine (ECC): [0] [1]
- NAND Device Basic Programming Model: [2] [3] [4] [5] [6] [7] [8] [9]
- GPMC Register Summary: [10]
- GPMC Register Description: [11] [12] [13]

Table 11-79. GPMC_ECC_CONTROL

Address Offset	0x0000 01F8		
Physical Address	0x6E00 01F8	**Instance**	GPMC
Description	ECC control		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																							ECCCLEAR	RESERVED				ECCPOINTER			

Bits	Field Name	Description	Type	Reset
31:9	RESERVED	Write 0s for future compatibility. Read returns 0s.	RW	0x000000
8	ECCCLEAR	Clear all ECC result registers Reads returns 0 Write 0x1 to this field clear all ECC result registers Write 0x0 is ignored	RW	0x0
7:4	RESERVED	Write 0s for future compatibility. Read returns 0s.	RW	0x0
3:0	ECCPOINTER	Selects ECC result register (Reads to this field give the dynamic position of the ECC pointer - Writes to this field select the ECC result register where the first ECC computation will be stored); Other enums: writing other values disables the ECC engine (ECCENABLE bit of GPMC_ECC_CONFIG set to 0) 0x0: Writing 0x0 disables the ECC engine (ECCENABLE bit of GPMC_ECC_CONFIG set to 0) 0x1: ECC result register 1 selected 0x2: ECC result register 2 selected 0x3: ECC result register 3 selected 0x4: ECC result register 4 selected 0x5: ECC result register 5 selected 0x6: ECC result register 6 selected 0x7: ECC result register 7 selected 0x8: ECC result register 8 selected 0x9: ECC result register 9 selected	RW	0x0

Table 11-80. Register Call Summary for Register GPMC_ECC_CONTROL

General-Purpose Memory Controller

- NAND Device Basic Programming Model: [0] [1] [2] [3]
- GPMC Register Summary: [4]

Table 11-81. GPMC_ECC_SIZE_CONFIG

Address Offset	0x0000 01FC		
Physical Address	0x6E00 01FC	**Instance**	GPMC
Description	ECC size		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED				ECCSIZE1				RESERVED				ECCSIZE0				RESERVED				ECC9RESULTSIZE				ECC8RESULTSIZE		ECC7RESULTSIZE		ECC6RESULTSIZE		ECC5RESULTSIZE		ECC4RESULTSIZE		ECC3RESULTSIZE		ECC2RESULTSIZE		ECC1RESULTSIZE	

Bits	Field Name	Description	Type	Reset
31:30	RESERVED	Write 0s for future compatibility. Read returns 0s.	RW	0x0
29:22	ECCSIZE1	Defines ECC size 1 0x00: 2 Bytes 0x01: 4 Bytes 0x02: 6 Bytes 0x03: 8 Bytes ... 0xFF: 512 Bytes	RW	0xFF
21:20	RESERVED	Write 0s for future compatibility. Read returns 0s.	RW	0x0
19:12	ECCSIZE0	Defines ECC size 0 0x00: 2 Bytes 0x01: 4 Bytes 0x02: 6 Bytes 0x03: 8 Bytes ... 0xFF: 512 Bytes	RW	0xFF
11:9	RESERVED	Write 0s for future compatibility. Read returns 0s.	RW	0x0
8	ECC9RESULTSIZE	Selects ECC size for ECC 9 result register 0x0: ECCSIZE0 selected 0x1: ECCSIZE1 selected	RW	0x0
7	ECC8RESULTSIZE	Selects ECC size for ECC 8 result register 0x0: ECCSIZE0 selected 0x1: ECCSIZE1 selected	RW	0x0
6	ECC7RESULTSIZE	Selects ECC size for ECC 7 result register 0x0: ECCSIZE0 selected 0x1: ECCSIZE1 selected	RW	0x0
5	ECC6RESULTSIZE	Selects ECC size for ECC 6 result register 0x0: ECCSIZE0 selected 0x1: ECCSIZE1 selected	RW	0x0
4	ECC5RESULTSIZE	Selects ECC size for ECC 5 result register 0x0: ECCSIZE0 selected 0x1: ECCSIZE1 selected	RW	0x0
3	ECC4RESULTSIZE	Selects ECC size for ECC 4 result register 0x0: ECCSIZE0 selected 0x1: ECCSIZE1 selected	RW	0x0
2	ECC3RESULTSIZE	Selects ECC size for ECC 3 result register 0x0: ECCSIZE0 selected 0x1: ECCSIZE1 selected	RW	0x0

Bits	Field Name	Description	Type	Reset
1	ECC2RESULTSIZE	Selects ECC size for ECC 2 result register 0x0: ECCSIZE0 selected 0x1: ECCSIZE1 selected	RW	0x0
0	ECC1RESULTSIZE	Selects ECC size for ECC 1 result register 0x0: ECCSIZE0 selected 0x1: ECCSIZE1 selected	RW	0x0

Table 11-82. Register Call Summary for Register GPMC_ECC_SIZE_CONFIG

General-Purpose Memory Controller

- NAND Device Basic Programming Model: [0] [1] [2] [3]
- GPMC Register Summary: [4]

Table 11-83. GPMC_ECCj_RESULT

Address Offset	0x0000 0200 + (0x0000 0004 * k)	**Index**	j= 1 to 9 and k = 0 to 8
Physical Address	0x6E00 0200 + (0x0000 0004 * k)	**Instance**	GPMC
Description	ECC result register		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED				P2048o	P1024o	P512o	P256o	P128o	P64o	P32o	P16o	P8o	P4o	P2o	P1o	RESERVED				P2048e	P1024e	P512e	P256e	P128e	P64e	P32e	P16e	P8e	P4e	P2e	P1e

Bits	Field Name	Description	Type	Reset
31:28	RESERVED	Write 0s for future compatibility. Read returns 0s.	RW	0x0
27	P2048o	Odd row parity bit 2048, only used for ECC computed on 512 Bytes	R	0x0
26	P1024o	Odd row parity bit 1024	R	0x0
25	P512o	Odd row parity bit 512	R	0x0
24	P256o	Odd row parity bit 256	R	0x0
23	P128o	Odd row parity bit 128	R	0x0
22	P64o	Odd row parity bit 64	R	0x0
21	P32o	Odd row parity bit 32	R	0x0
20	P16o	Odd row parity bit 16	R	0x0
19	P8o	Odd row parity bit 8	R	0x0
18	P4o	Odd Column Parity bit 4	R	0x0
17	P2o	Odd Column Parity bit 2	R	0x0
16	P1o	Odd Column Parity bit 1	R	0x0
15:12	RESERVED	Write 0s for future compatibility. Read returns 0s.	RW	0x0
11	P2048e	Even row parity bit 2048, only used for ECC computed on 512 Bytes	R	0x0
10	P1024e	Even row parity bit 1024	R	0x0
9	P512e	Even row parity bit 512	R	0x0
8	P256e	Even row parity bit 256	R	0x0
7	P128e	Even row parity bit 128	R	0x0
6	P64e	Even row parity bit 64	R	0x0
5	P32e	Even row parity bit 32	R	0x0
4	P16e	Even row parity bit 16	R	0x0
3	P8e	Even row parity bit 8	R	0x0
2	P4e	Even column parity bit 4	R	0x0
1	P2e	Even column parity bit 2	R	0x0

Bits	Field Name	Description	Type	Reset
0	P1e	Even column parity bit 1	R	0x0

Table 11-84. Register Call Summary for Register GPMC_ECCj_RESULT

General-Purpose Memory Controller

- NAND Device Basic Programming Model: [0] [1] [2] [3] [4] [5] [6] [7]
- GPMC Register Summary: [8]

Table 11-85. GPMC_BCH_RESULT0_i

Address Offset	0x0000 0240 + (0x0000 0010 * i)	**Index**	i = 0 to 7
Physical Address	0x6E00 0240 + (0x0000 0010 * i)	**Instance**	GPMC
Description	BCH ECC result (bits 0 to 31)		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BCH_RESULT_0																															

Bits	Field Name	Description	Type	Reset
31:0	BCH_RESULT_0	BCH ECC result (bits 0 to 31)	RW	0x00000000

Table 11-86. Register Call Summary for Register GPMC_BCH_RESULT0_i

General-Purpose Memory Controller

- NAND Device Basic Programming Model: [0]
- GPMC Register Summary: [1]

Table 11-87. GPMC_BCH_RESULT1_i

Address Offset	0x0000 0244 + (0x0000 0010 * i)	**Index**	i = 0 to 7
Physical Address	0x6E00 0244 + (0x0000 0010 * i)	**Instance**	GPMC
Description	BCH ECC result (bits 32 to 63)		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BCH_RESULT_1																															

Bits	Field Name	Description	Type	Reset
31:0	BCH_RESULT_1	BCH ECC result (bits 32 to 63)	RW	0x00000000

Table 11-88. Register Call Summary for Register GPMC_BCH_RESULT1_i

General-Purpose Memory Controller

- NAND Device Basic Programming Model: [0]
- GPMC Register Summary: [1]

Table 11-89. GPMC_BCH_RESULT2_i

Address Offset	0x0000 0248 + (0x0000 0010 * i)	**Index**	i = 0 to 7
Physical Address	0x6E00 0248 + (0x0000 0010 * i)	**Instance**	GPMC
Description	BCH ECC result (bits 64 to 95)		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BCH_RESULT_2																															

Bits	Field Name	Description	Type	Reset
31:0	BCH_RESULT_2	BCH ECC result (bits 64 to 95)	RW	0x00000000

Table 11-90. Register Call Summary for Register GPMC_BCH_RESULT2_i

General-Purpose Memory Controller

- NAND Device Basic Programming Model: [0]
- GPMC Register Summary: [1]

Table 11-91. GPMC_BCH_RESULT3_i

Address Offset	0x0000 024C + (0x0000 0010 * i)	**Index**	i = 0 to 7
Physical Address	0x6E00 024C + (0x0000 0010 * i)	**Instance**	GPMC
Description	BCH ECC result (bits 96 to 103)		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																								BCH_RESULT_3							

Bits	Field Name	Description	Type	Reset
31:8	RESERVED	Write 0s for future compatibility. Read returns 0s.	R	0x000000
7:0	BCH_RESULT_3	BCH ECC result (bits 96 to 103)	RW	0x00

Table 11-92. Register Call Summary for Register GPMC_BCH_RESULT3_i

General-Purpose Memory Controller

- NAND Device Basic Programming Model: [0]
- GPMC Register Summary: [1]

Table 11-93. GPMC_BCH_SWDATA

Address Offset	0x0000 02D0		
Physical Address	0x6E00 02D0	**Instance**	GPMC
Description	This register is used to directly pass data to the BCH ECC calculator without accessing the actual NAND flash interface.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																BCH_DATA															

Bits	Field Name	Description	Type	Reset
31:16	RESERVED	Write 0s for future compatibility. Read returns 0s.	R	0x0000
15:0	BCH_DATA	Data to be included in the BCH calculation. Only bits 0 to 7 are taken into account if the calculator is configured to use 8 bits data (GPMC_ECC_CONFIG[7] ECC16B = 0)	RW	0x0000

Table 11-94. Register Call Summary for Register GPMC_BCH_SWDATA

General-Purpose Memory Controller • GPMC Register Summary: [0]

11.2 SDRAM Controller (SDRC) Subsystem

This section describes the SDRAM controller (SDRC) subsystem of the OMAP34xx multimedia device.

11.2.1 SDRC Subsystem Overview

The SDRC subsystem module provides connectivity between the OMAP™ device and external discrete, stacked or POP-ed SDRAM memory components. The module includes support for low-power or mobile single-data-rate (LPSDR or M-SDR) and low-power double-data-rate SDRAM (LPDDR).

The SDRC subsystem provides a high-performance interface to a variety of fast memory devices. It comprises two submodules:

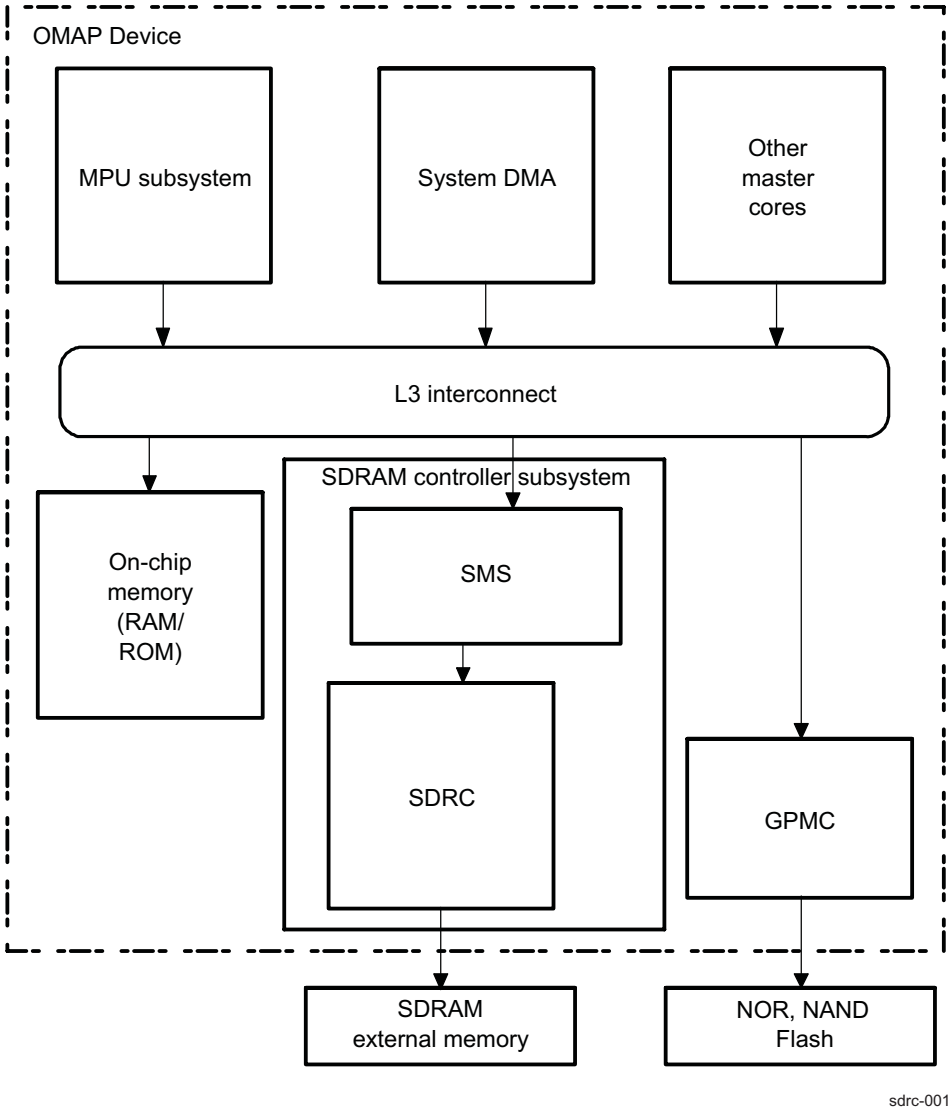
- The SDRAM memory scheduler (SMS), consisting of scheduler and virtual rotated frame-buffer (VRFB) modules
- The SDRC

CAUTION

DDR SDRAM and SDR SDRAM memory types cannot be connected simultaneously to the SDRC memory interface.

Figure 11-41 shows the SDRC subsystem environment.

Figure 11-41. SDRC Subsystem Environment



11.2.1.1 Features

The main features of the SDRC subsystem module are:

- VRFB module
  - Minimizes SDRAM page-miss penalty when accessing graphics buffer in nonnatural raster-scan order
  - Supports rotations of 0, 90, 180, and 270 degrees
  - Transparent to software applications
  - 12 concurrent rotation contexts
- Memory-access scheduler
  - Optimizes latency and bandwidth usage between initiators
  - Per-system initiator group quality-of-service (QoS) control
  - $8 \times 8 \times 64$ request queue FIFO for optimal scheduling
  - Programmable arbitration scheme
  - Focus on real-time memory processes (LCD display, camera interface)
  - Focus on MPU/DSP memory latency

  - Fair arbitration between other system initiators (DMAs, video subsystem, GFX accelerator)
  - Exclusive read-write transaction support
- SDRAM Controller
  - Support for two independent CSs, with their corresponding register sets, and independent page tracking
  - Supports the following memory types:
    - Mobile Single Data Rate SDRAM (M-SDR)
    - Low-Power Double Data Rate SDRAM (LPDDR)
  - Memory device capacity:
    - 16 Mbits, 32 Mbits, 64 Mbits, 128 Mbits, 256 Mbits, 512 Mbits , 1 Gbit, and 2 Gbits device support
  - Memory device organization
    - 2-bank support for 16 Mbits and 32 Mbits
    - 4-bank support for 64 Mbits to 2 Gbits
    - Flexible row/column address multiplexing schemes
    - Bank linear addressing
    - 16- or 32-bit data path to external SDRAM memory
    - 1G byte maximum addressing capability
    - Device driver strength feature for mobile DDR supported
    - New flexible address-muxing scheme lets users choose different bank mapping allocations by configuring the bank and column address decoding ordering.
  - Fully pipelined operation for optimal memory bandwidth usage
  - Burst support
    - Memory burst support
      - System burst for SDR SDRAM: system burst translated into memory burst of 2
      - System burst for mobile DDR SDRAM: system burst translated into memory burst size of 4
      - Read interrupt by read, write interrupt by write
  - CAS latency support 1, 2, 3, 4, 5
  - Fully programmable ac timing parameters (on a per-parameter basis). Parameters are set according to the memory interface clock frequency with respect to the attached memory device timing specifications.
  - Fine tuning of the controlled delay elements when operating with DDR memory
  - Dynamic endianness support
  - 9×64 bit lookahead FIFO in the SDRAM controller with a maximum of four transaction entries
  - Low-power management support
    - Dynamic power-saving features (internal clock gating)
    - Static power-saving features
    - Support for all standard low-power memory features
    - Support for enhanced low-power features (mobile devices)
    - Very low-power controlled-delay technology for optimal performance with DDR memory
    - Can operate with mobile DDR memory at very low clock rates
  - Autorefresh and self-refresh management

Note: The OMAP device does not support regular devices, SDR or DDR, because of electrical incompatibility.

11.2.2 SDRC Subsystem Environment

11.2.2.1 SDRC Subsystem Description

Figure 11-42 shows the SDRC subsystem interfacing with one 16- and one 32-bit SDR external memories. Figure 11-43 shows the SDRC subsystem interfacing with one 16- and one 32-bit DDR memories.

Figure 11-42. SDRC Subsystem Connections to SDR SDRAM

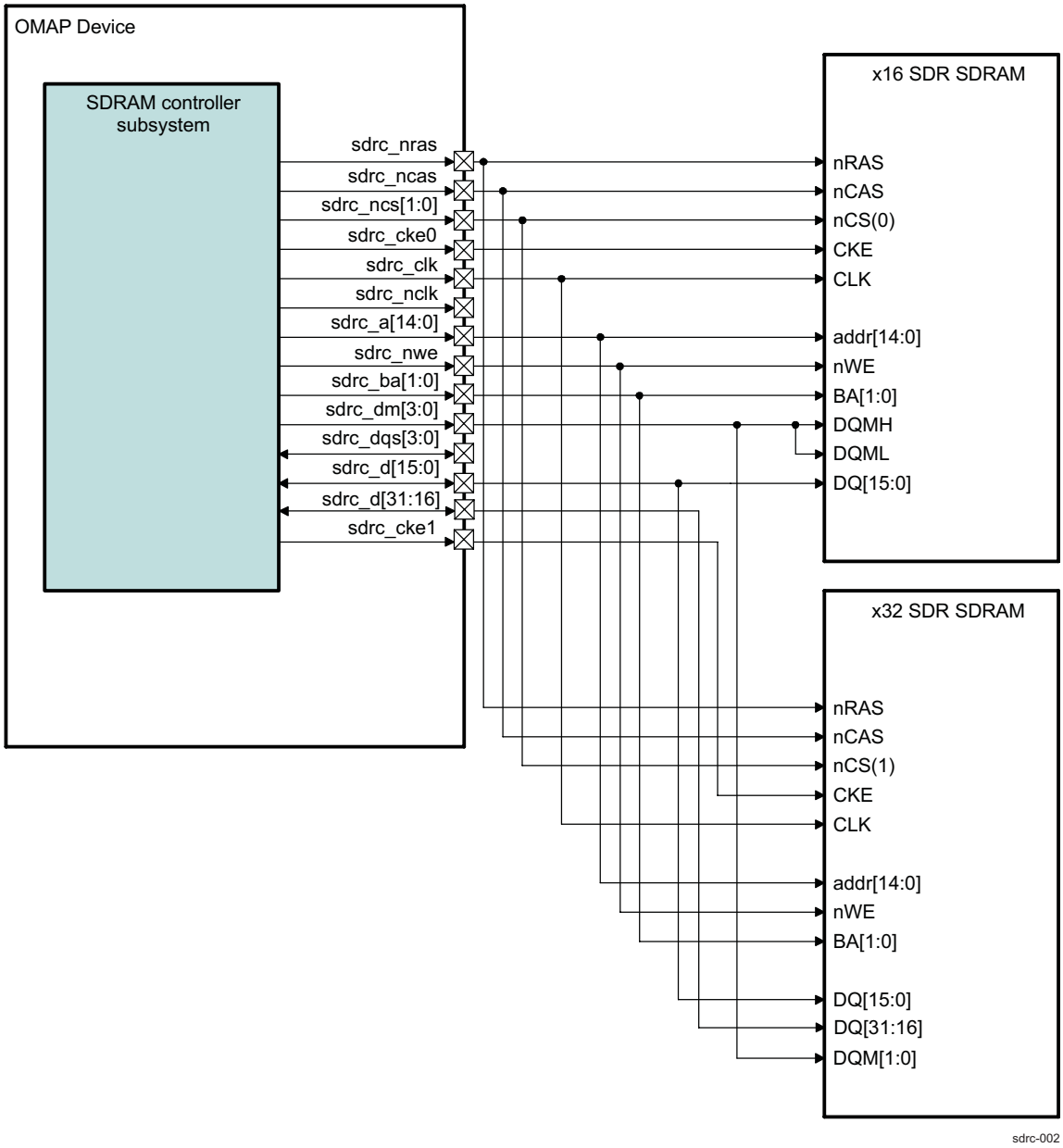
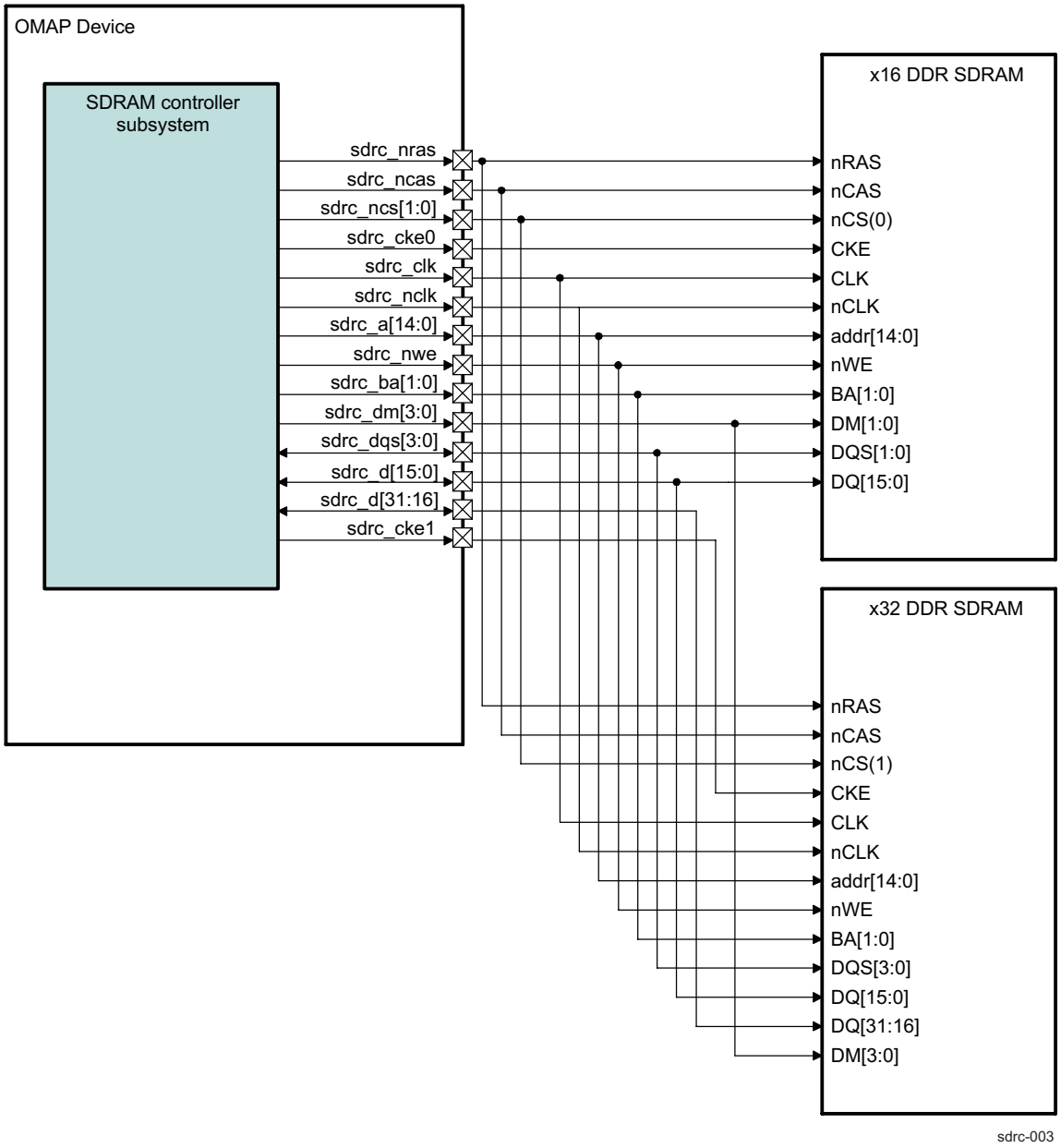


Figure 11-43. SDRC Subsystem Connections to DDR SDRAM



CAUTION

Both memory types (SRD/DDR SDRAM) cannot be connected simultaneously to the SDRC memory interface.

Table 11-95 lists SDRC subsystem I/O pins.

Table 11-95. SDRC Subsystem I/O Description

Pin	Type	Description
sdrc_d[31:0]	I/O	32-bit wide data bus
sdrc_ba[1:0]	O	Bank address 1-0
sdrc_a[14:0]	O	Address bus

Table 11-95. SDRC Subsystem I/O Description (continued)

Pin	Type	Description
sdrc_ncs[1:0]	O	Chip-select 1-0 (active low)
sdrc_clk	O	Clock (1)
sdrc_nclk	O	Clock invert (DDR only)
sdrc_cke0	O	Clock enable (CS0)
sdrc_cke1	O	Clock enable (CS1)
sdrc_nras	O	Row address strobe (active low)
sdrc_ncas	O	Column address strobe (active low)
sdrc_nwe	O	Write enable (active low)
sdrc_dm[3:0]	O	Data mask/OE for SDR read operation
sdrc_dqs[3:0]	I/O	Data output strobe (DDR only)

(1) This output signal is also used as re-timing input.

11.2.2.2 External Interface Configuration

11.2.2.2.1 CS0, CS1 Memory Spaces

The SDRC can be connected independently to two external SDRAM memories.

Mobile SDR SDRAM memory is supported in sizes of 16 Mbits, 32 Mbits, 64 Mbits, 128 Mbits, 256 Mbits, 512 Mbits, 1 Gbit, and 2 Gbits. See Section 11.2.4.4.1 for more information on CS memory spaces.

Note: The OMAP device does not support regular devices, SDR or DDR (see Section 11.2.5.3.5.1, *Chip-Select Configuration*).

11.2.2.2.2 AC Timing Control

The SDRC permits the configuration of many of the ac timing parameters controlling the SDRAM transaction timing.

It is required to adapt the transaction timing parameters to any memory device attached to the SDRC.

A totally independent, complete set of parameters exists for each CS. The ac parameters and their quantification are summarized in Section 11.2.5.3.1, *Chip-Select Configuration*.

11.2.2.2.3 Address Multiplexing

A flexible address scheme has been added to support any new type of address scheme and SDRAM density. This new address-muxing scheme lets users choose a different bank mapping allocation by configuring the order of the bank and row address decoding (column address always remains the same). The SDRC.SDRC_MCFG_p[7:6] BANKALLOCATION field defines the order of the bank and row decoding of the incoming system address. The BANKALLOCATION field can be set on a CS basis. For more details about the flexible address scheme, see Section 11.2.4.4.3, *Address Multiplexing*.

The programming model is compatible with the legacy fixed address scheme and with the new flexible address scheme. This section describes the fixed address multiplexing configurations for both SDRAM components.

The legacy fixed address scheme is selected when the SDRC.SDRC_MCFG_p[19] ADDRMUXLEGACY bit is set to 0 (where p = 0 or 1 for SDRC CS0 or CS1). The fixed address multiplexing configurations are described in this section, for both SDRAM components. An address multiplexing scheme is chosen by programming the SDRC.SDRC_MCFG_p[24:20] ADDRMUX field of the SDRC_MEMCFG registers on a per chip select basis (p = 0 or 1 for CS0 or CS1). Table 11-96 and Table 11-97 show all predefined address multiplexing schemes for SDRAM memory components.

Table 11-96. SDRC Address Multiplexing Scheme Selection vs SDRAM Configurations (x16 Memory Interface)

x16 Memory Interface								
Banks		**Column Address**		**Row Address**		**MUX Scheme**	**Total Size (MBits)**	**Number of Devices**	**Device Organization**
BA0	1	A0-A7	8	A0-A10	11	MUX1	16	1	1M x 16
BA0	1	A0-A7	8	A0-A11	12	MUX2	32	1	2M x 16
BA1, BA0	2	A0-A7	8	A0-A11	12	MUX2	64	1	4M x 16
BA1, BA0	2	A0-A8	9	A0-A11	12	MUX4	128	2 1	8M x 8 8M x 16
BA1, BA0	2	A0-A8	9	A0-A12	13	MUX7	256	1	16M x 16
BA1, BA0	2	A0-A8	9	A0-A13	14	MUX26	512	1	32M x 16
BA1, BA0	2	A0-A9	10	A0-A11	12	MUX6	256	2 1	16M x 8 16M x 16
BA1, BA0	2	A0-A9	10 10	A0-A12	13	MUX10	512	2 1	32M x 8 32M x 16
BA1, BA0	2	A0-A9	10	A0-A13	14	MUX13	1024	1	64M x 16
BA1, BA0	2	A0-A9; A11	11	A0-A12	13	MUX12	1024	2 1	64M x 8 64M x 16

Table 11-97. SDRC Address Multiplexing Scheme Selection vs SDRAM Configurations (x32 Memory Interface)

x32 Memory Interface								
Banks		**Column Address**		**Row Address**		**MUX Scheme**	**Total Size (MBits)**	**Number of Devices**	**Device Organization**
BA0	1	A0-A7	8	A0-A11	12	MUX5	64	2 1	2M x 16 2M x 32
BA1, BA0	2	A0-A7	8	A0-A10	11	MUX3	64	1	2M x 32
BA1, BA0	2	A0-A7	8	A0-A11	12	MUX5	128	1	4M x 32
BA1, BA0	2	A0-A7	8	A0-A12	13	MUX9	256	1	8M x 32
BA1, BA0	2	A0-A7	8	A0-A13	14	MUX27	512	1	16M x 32
BA1, BA0	2	A0-A7	8	A0-A14	15	MUX28	1024	1	32M x 32
BA1, BA0	2	A0-A8	9	A0-A11	12	MUX8	256	4 2 1	8M x 8 8M x 16 8M x 32
BA1, BA0	2	A0-A8	9	A0-A12	13	MUX24	512	1	16M x 32
BA1, BA0	2	A0-A8	9	A0-A13	14	MUX23	1024	1	32M x 32
BA1, BA0	2	A0-A8	9	A0-A14	15	MUX25	2048	1	64M x 32

Table 11-97. SDRC Address Multiplexing Scheme Selection vs SDRAM Configurations (x32 Memory Interface) (continued)

x32 Memory Interface								
Banks		**Column Address**		**Row Address**		**MUX Scheme**	**Total Size (MBits)**	**Number of Devices**	**Device Organization**
BA1, BA0	2	A0-A9	10	A0-A11	12	MUX11	512	4	16M x 8
								2	16M x 16
								1	16M x 32
BA1, BA0	2	A0-A9	10	A0-A12	13	MUX14	1024	4	32M x 8
								2	32M x 16
								1	32M x 32
BA1, BA0	2	A0-A9	10	A0-A13	14	MUX16	2048	2	64M x 16
								1	64M x 32
BA1, BA0	2	A0-A9	10	A0-A14	15	MUX29	4096	1	128M x 32
BA1, BA0	2	A0-A9; A11	11	A0-A12	13	MUX15	2048	4	64M x 8
								2	64M x 16
								1	64M x 32

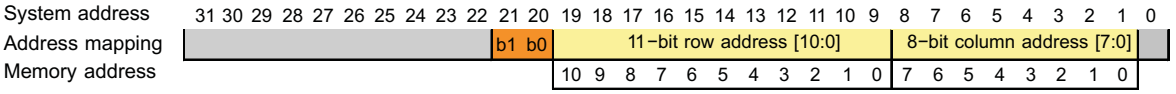
Table 11-96 is valid per CS (in the table, the total size in Mbits corresponds to a single CS). Both chips must use the same address scheme, but can use different address multiplexing configurations.

Figure 11-44 through Figure 11-46 show all the SDRAM MUX schemes, including the mapping of the 32-bit system address of the column and row addresses of the memory device.

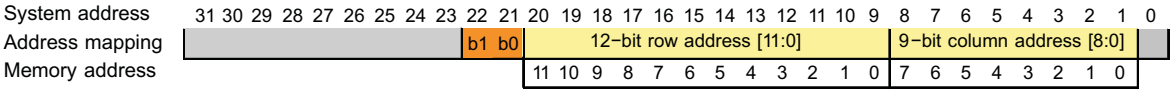
Note: The bank-select bit (BA1) does not exist for 2-bank devices.

Figure 11-44. SDRC SDR/DDR-SDRAM System Address Multiplexing Schemes (1 of 3)

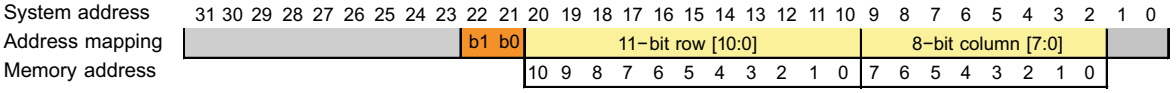
MUX1



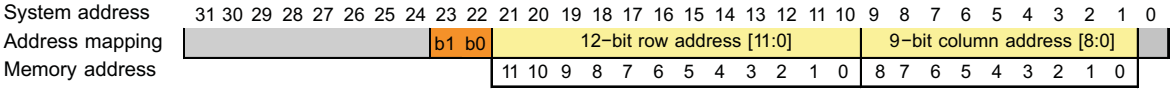
MUX2



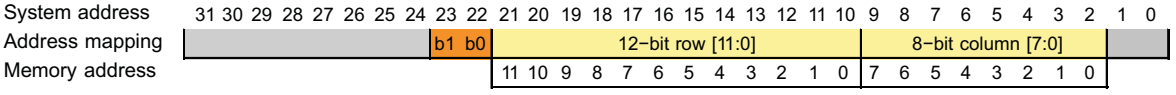
MUX3



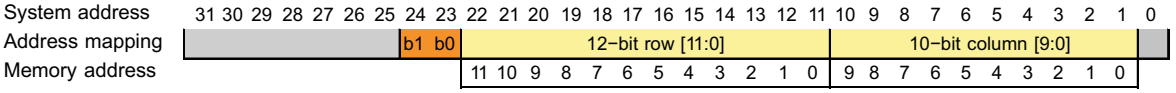
MUX4



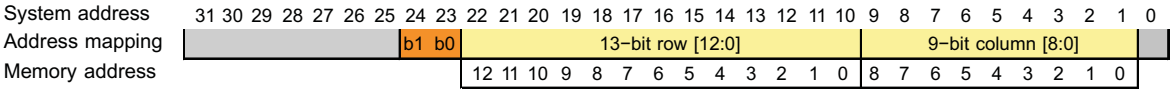
MUX5



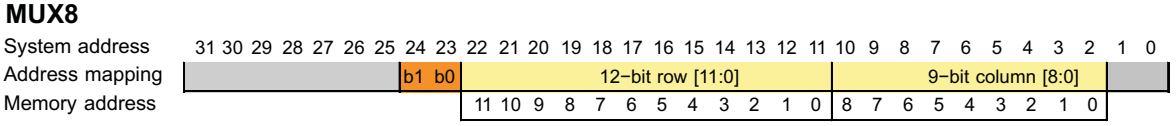
MUX6



MUX7



MUX8



sdrc-004

Figure 11-45. SDRC SDR/DDR-SDRAM System Address Multiplexing Schemes (2 of 3)

MUX9

	31 30 29 28 27 26 25 24	23	22 21 20 19 18 17 16 15 14 13 12 11 10	9 8 7 6 5 4 3 2 1 0
System address				
Address mapping		b1	b0	13−bit row [12:0]	8−bit column [7:0]
Memory address			12 11 10 9 8 7 6 5 4 3 2 1 0	7 6 5 4 3 2 1 0

MUX10

	31 30 29 28 27 26 25 24	23	22 21 20 19 18 17 16 15 14 13 12 11 10	9 8 7 6 5 4 3 2 1 0
System address				
Address mapping		b1	b0	13−bit row [12:0]	10−bit column [9:0]
Memory address			12 11 10 9 8 7 6 5 4 3 2 1 0	9 8 7 6 5 4 3 2 1 0

MUX11

	31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16 15 14 13 12	11 10 9 8 7 6 5 4 3 2 1 0
System address			
Address mapping		b1 b0	12−bit row [11:0]	10−bit column [9:0]
Memory address		11 10 9 8 7 6 5 4 3 2 1 0	9 8 7 6 5 4 3 2 1 0

MUX12

	31 30 29 28 27 26 25 24 23	22 21 20 19 18 17 16 15 14 13 12 11 10	9 8 7 6 5 4 3 2 1 0
System address			
Address mapping		b1	b0	13−bit row [12:0]	11−bit column [10:0]
Memory address		12 11 10 9 8 7 6 5 4 3 2 1 0	10 9 8 7 6 5 4 3 2 1 0

MUX13

	31 30 29 28 27 26 25 24 23	22 21 20 19 18 17 16 15 14 13 12 11 10	9 8 7 6 5 4 3 2 1 0
System address			
Address mapping		b1	b0	14−bit row [13:0]	10−bit column [9:0]
Memory address		13 12 11 10 9 8 7 6 5 4 3 2 1 0	9 8 7 6 5 4 3 2 1 0

MUX14

	31 30 29 28 27 26 25 24 23	22 21 20 19 18 17 16 15 14 13 12 11 10	9 8 7 6 5 4 3 2 1 0
System address			
Address mapping		b1	b0	13−bit row [12:0]	10−bit column [9:0]
Memory address		12 11 10 9 8 7 6 5 4 3 2 1 0	9 8 7 6 5 4 3 2 1 0

MUX15

	31 30 29 28 27 26 25 24 23 22	21 20 19 18 17 16 15 14 13 12 11 10	9 8 7 6 5 4 3 2 1 0
System address			
Address mapping		b1	b0	13−bit row [12:0]	11−bit column [10:0]
Memory address		12 11 10 9 8 7 6 5 4 3 2 1 0	10 9 8 7 6 5 4 3 2 1 0

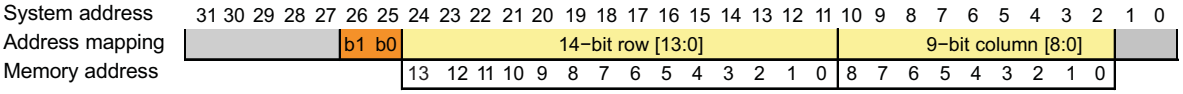
MUX16

	31 30 29 28 27 26 25 24 23 22	21 20 19 18 17 16 15 14 13 12 11 10	9 8 7 6 5 4 3 2 1 0
System address			
Address mapping		b1	b0	14−bit row [13:0]	10−bit column [9:0]
Memory address		13 12 11 10 9 8 7 6 5 4 3 2 1 0	9 8 7 6 5 4 3 2 1 0

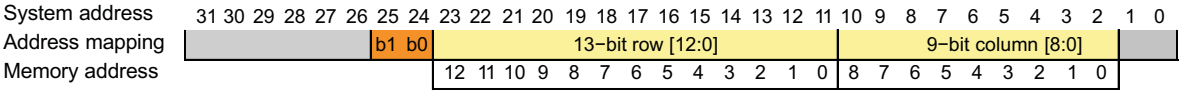
sdrc-005

Figure 11-46. SDRC SDR/DDR-SDRAM System Address Multiplexing Schemes (3 of 3)

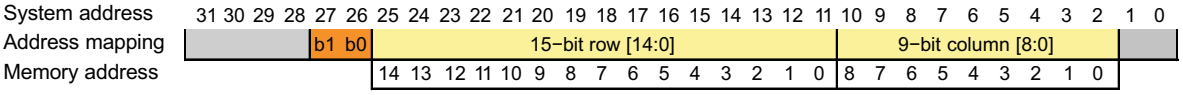
MUX23



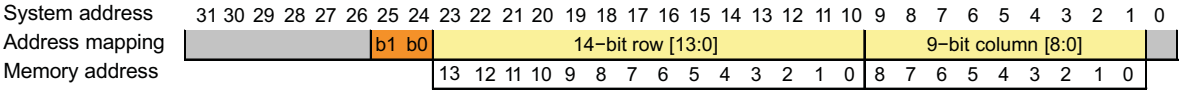
MUX24



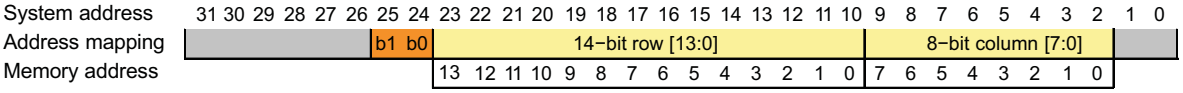
MUX25



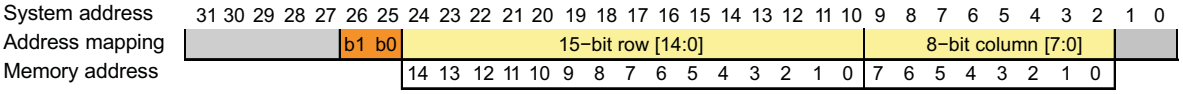
MUX26



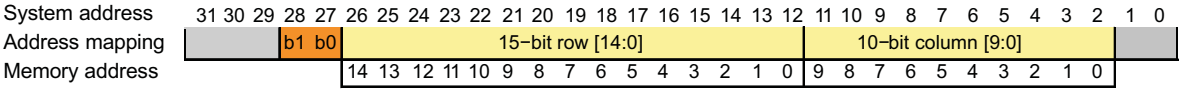
MUX27



MUX28



MUX29

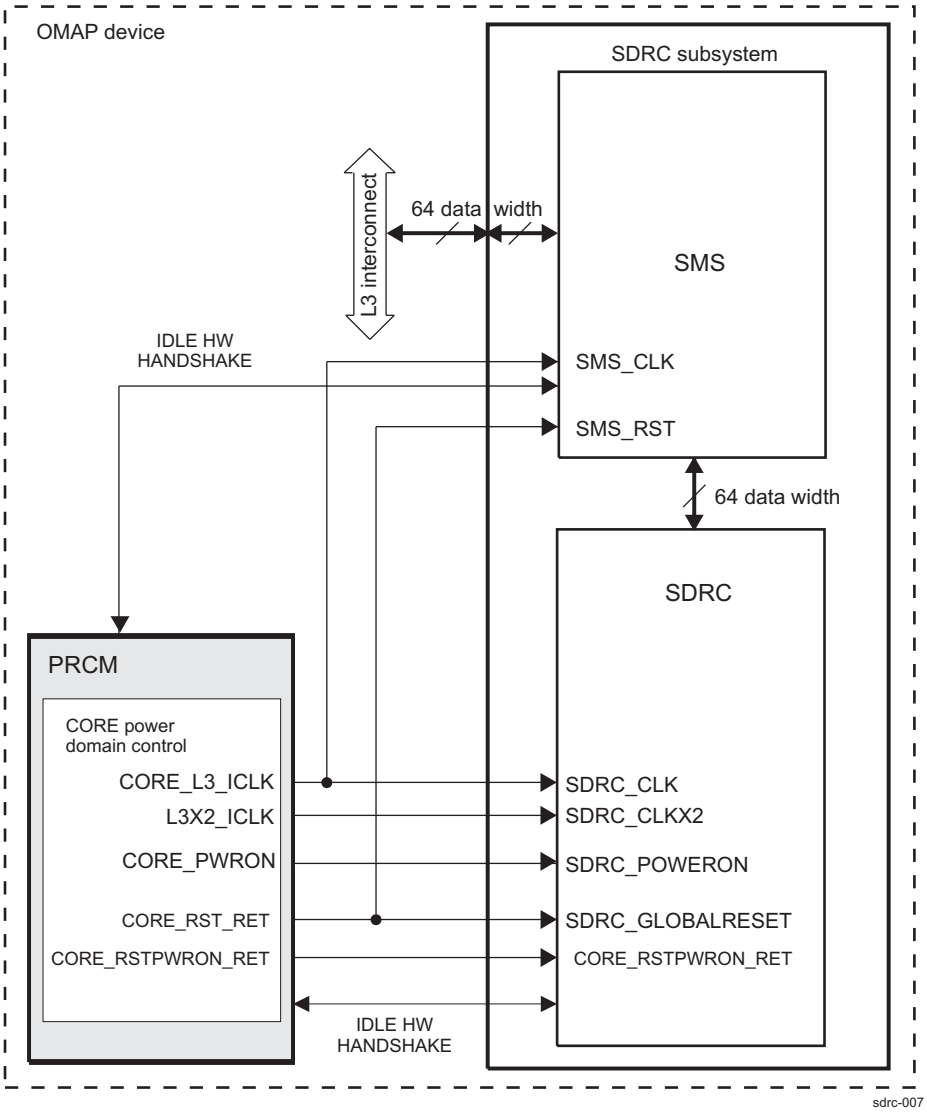


sdrc-006

11.2.3 SDRC Subsystem Integration

Figure 11-47 shows how the SMS and SDRC modules are integrated into the OMAP device and how they interact with the PRCM module.

Figure 11-47. SDRC Integration to the OMAP Device



11.2.3.1 Clocking, Reset, and Power Management Scheme

11.2.3.1.1 Clocking

11.2.3.1.1.1 SMS

The SMS is a single clock domain module. The same clock is used for the interconnect system interface, the SDRC interface, and all internal operations.

SMS_CLK comes internally from the PRCM module and runs at the L3 interconnect frequency. It is also used as a functional clock for the SMS module.

The source of the SMS_CLK is the PRCM CORE_L3_ICLK output: CORE_L3_ICLK belongs to the L3 interconnect clock domain.

11.2.3.1.1.2 SDRC

The SDRC is a single-clock domain module. The same clock is used for the interconnect and the memory interface. The SDRC_CLK clock comes from the PRCM and runs at the L3 interconnect frequency. SDRC_CLK is also used as a functional clock for the SDRC.

The SDRC_CLK clock source is the PRCM CORE_L3_ICLK output: CORE_L3_ICLK belongs to the L3 interconnect clock domain.

An SDRC_CLKX2 clock is provided by the PRCM at a double frequency of the SDRC_CLK clock. This clock can be used in the LPDDR write path, depending on the configuration.

As a power-saving feature, when the SDRC no longer requires the clock domain, the software can disable it at the PRCM level by setting the EN_SDRC bit in the PRCM.CM_ICKLEN1_CORE[1] register.

Note: The domain clock is shut down only if all other modules that receive the clock are capable and ready to accept this idle request and have IdleAck asserted.

For details, see Chapter 4, *Power, Reset, and Clock Management.*

11.2.3.1.2 Hardware Reset

Global reset of the SDRC is done by activating the CORE_RST signal in the CORE_RST domain (see Chapter 4, *Power, Reset, and Clock Management).*

There is one global reset signal, SDRC_GLOBALRESET, which is qualified by the signal SDRC_POWERON. This qualification differentiates whether the signal is a cold reset or a warm reset.

- On a cold reset (that is, the power-on reset, when SDRC_POWERON = 0 and SDRC_GLOBALRESET is applied), all registers and state-machines within the SDRC are asynchronously reset.
- On a warm reset (that is, any other system reset condition under control of the chip top-level power manager, SDRC_POWERON = 1 when SDRC_GLOBALRESET is applied), the SDRC registers and the FSM are not reset, but the external SDRAM memory can be optionally placed in self-refresh mode, depending on the configuration of the SDRC.SDRC_POWER_REG[7] SRFRONRESET bit.

11.2.3.1.3 Software Reset

The SMS and SDRC modules can be reset under software control through the SMS.SMS_SYSCONFIG[1] SOFTRESET and SDRC.SDRC_SYSCONFIG[1] SOFTRESET bits, respectively.

A software reset has the same action as a hardware cold reset; that is, the FSM and all registers are reset immediately and unconditionally.

11.2.3.1.4 Power Management

The SDRC power is supplied by the CORE power domain. For details, see Chapter 4, *Power, Reset, and Clock Management.*

The dynamic voltage and frequency scaling (DVFS) technic is described in the following section while other power-saving features and different idle modes are described in Section 11.2.4.4, *SDRC*.

11.2.3.1.4.1 Dynamic Voltage and Frequency Scaling

If the input clock SDRC_CLK from the PRCM module is changed during operations, the DLL may enter an undefined state and memory accesses may be corrupted. For this reason, it is necessary to manually assert the SDRC_IDLEREQ signal before any clock frequency change. This manual access is done through a register in the chip clock controller. The SDRC then finishes processing all open transactions. If this option is activated in the SDRC_POWER_REG[6] SRFRONIDLEREQ bit, it puts the memory into self-refresh. When done, it unlocks the DLL and puts it into a power-down state, through the SDRC.SDRC_DLLA_CTRL[6:5] DLLMODEONIDLEREQ field. The SDRC then asserts the

SDRC_SIDLEACK signal and the input clock frequency can be changed or stopped, depending on the scenario. After the input clock is stable again, SDRC_IDLEREQ is de-asserted. The DLL then relocks and the SDRC can be accessed normally. Putting the DLL in idle mode during the clock frequency change is required because it cannot automatically relock. It might get into a non-functional state and a new manual DLL configuration phase would then be required, with the risk of corrupting accesses in the mean time.

11.2.4 SDRC Subsystem Functional Description

The SMS optimizes the SDRAM memory usage to provide:

- The QoS level required by each of the initiators in the system
- A VRFB module (also called the 2D rotation engine) that minimizes the SDRAM page-miss penalty when accessing rotated (that is, nonsequentially addressed) lines in a graphic frame buffer

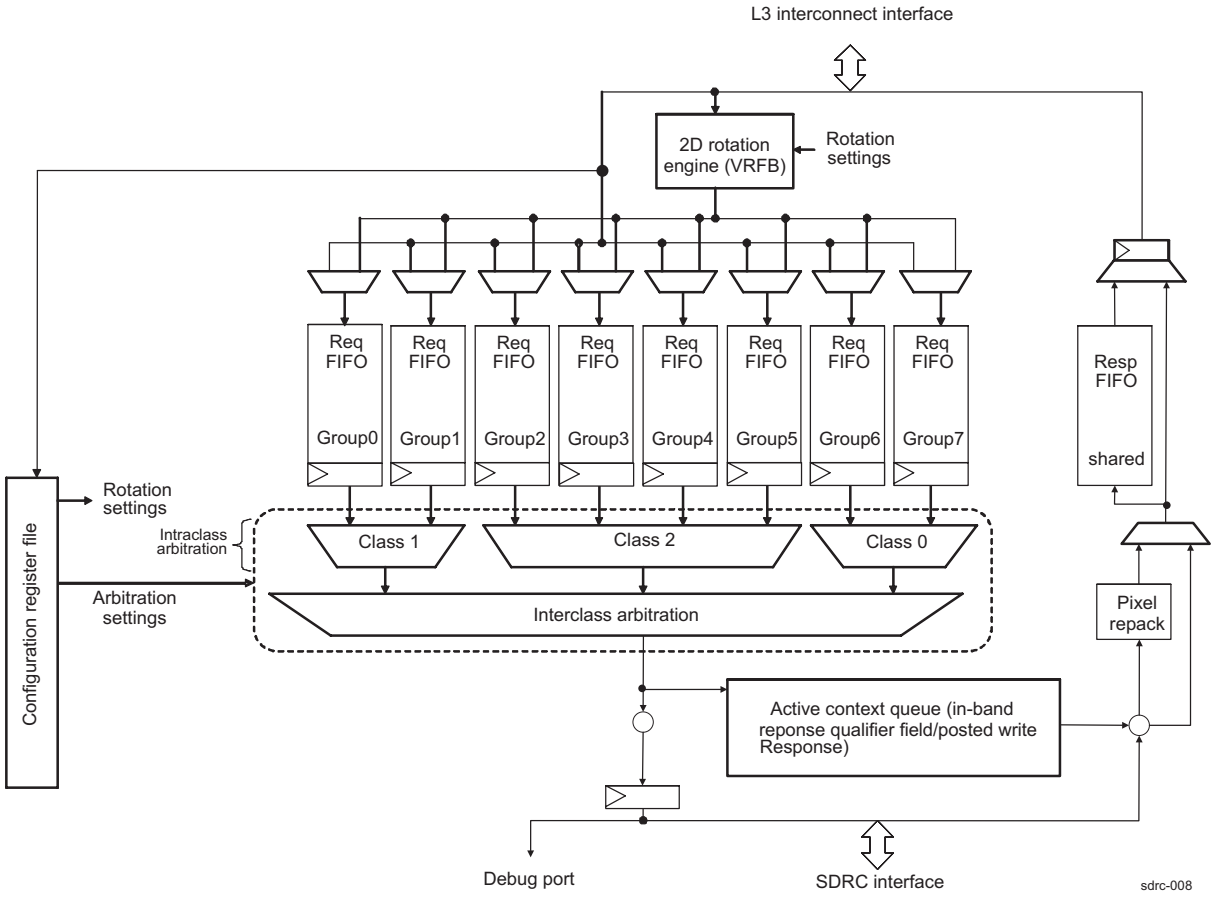
11.2.4.1 SDRAM Memory Scheduler

The SMS module is split into the following subsystems:

- L3 interconnect slave port
- VRFB: Rotation engine (RE)
- Configuration register file
- Request buffers
- Arbitration logic
- SDRC interface: master port
- Debug port
- Response buffer

Figure 11-48 shows the top-level diagram of the SMS.

Figure 11-48. SMS Top-Level Diagram



11.2.4.1.1 Memory Access Scheduling

Note: For a description of the SMS mode of operation and arbitration policy, see Section 11.2.6, *SDRC Use Cases and Tips*.

The SMS includes a set of FIFOs to queue the requests received from the system interconnect or processed by the RE. The FIFO entry contains a complete interconnect request, plus internal qualifiers added by the RE which are required to correctly process the requests modified (or generated) by the RE.

The SMS also includes a response FIFO, which is shared by all response threads. This FIFO provides full support for the response flow-control interconnect extension (handshake protocol).

When a memory transaction request arrives in the memory-access scheduler after being processed by the VRFB module, the request is sent to one of eight FIFO queues.

The allocation to a particular queue is fixed and depends on the source of the request. To prioritize concurrent transactions and optimize memory usage, each FIFO queue (and hence the memory-access request source) is categorized into one of these three arbitration classes:

- Class 0 (highest priority): For bandwidth-sensitive devices with severe real-time constraints. The system fails when the bandwidth requirement is not met. LCD or video display cores, camera capture cores belong to this category.
- Class 1: For latency-sensitive devices where system performance degrades severely when the average memory-access latency increases. These initiators also generally have some significant bandwidth requirements. All CPU cores are class 1 initiators.
- Class 2: For all other devices, possibly with high-bandwidth requirements but without being too latency-sensitive. Associated initiators may also have significant requirements in terms of bandwidth, but if the bandwidth budget requirement cannot be serviced, the system performance degrades, but remains functional (the system does not fail). General-purpose system DMA, multimedia accelerators (graphics, imaging, video) belong to this category.

The arbitration between Class 1 and Class 2 is programmable.

Table 11-98 shows the mapping of FIFO queues and memory-access request sources to arbitration classes.

Table 11-98. Arbitration Class Allocation

Class	FIFO Queue	Source Device
0	6	D2D
	7	Display and Camera subsystems
1	0	MPU subsystem (instruction and data access)
	1	IVA2 subsystem (instruction, data access, and MMU)
2	2	IVA2 subsystem DMA (read and write), sDMA WR
	3	SGX
	4	USB (HS + FS), DAP
	5	sDMA (read)

A 2-level arbitration scheme determines the FIFO queue from which the next memory transaction is granted.

Note: In the register description, a group represents a FIFO queue of requests from the initiator.

11.2.4.1.2 Arbitration Policy

A 2-level arbitration scheme is implemented to grant access to the SDRC. The arbitration uses fully combinatorial logic, and the granted request is clocked before being issued on the interface master port of the SDRC.

- Intra-class arbitration: A first level of arbitration is done in parallel within each class between the different thread groups (request FIFOs).
- Interclass arbitration: A second level of arbitration is done among the three winners of the in-class arbitration.

11.2.4.1.2.1 Arbitration Policy Within a Burst and at a Burst Boundary

Arbitration is performed on the transaction boundary. The transaction can be either a single transaction or a burst transaction.

- Within a burst, if the thread cannot provide the subsequent request of the burst, a mechanism provides a wait for one idle cycle before moving the arbitration grant to the next thread. If only one idle cycle appears on one thread, an arbitration grant must not move (the next request served must be from the same thread). One idle cycle is then inserted in the SDRC request path.
- If the thread still cannot provide the request in the next cycle, the slot can be awarded to another initiator to avoid locking the SDRAM resource, if a thread cannot supply a subsequent request within the burst. In this case, there must not be a second idle cycle insertion in the SDRC request path. A burst with fewer than two idle cycles cannot be interrupted by a higher priority request.
- On burst boundary, the arbitration does not wait one idle cycle before moving the arbitration grant. As soon as the thread cannot request one transaction at a burst boundary, the arbitration grant can move to the next thread.

This is also valid within the ExtendedGrant and NOfServices windows; as soon as the thread cannot request a transaction, the arbitration grant can move to the next thread. For more information, see the following descriptions of the ExtendedGrant and NOfServices features.

11.2.4.1.2.2 Burst-Complete Feature (BURST-COMPLETE Field in SMS_CLASS_ARBITER0, SMS_CLASS_ARBITER1, and SMS_CLASS_ARBITER2)

A burst request can be submitted to the arbitration either as soon as the first request of the burst is received by the SMS or when at least one complete burst is buffered into the FIFO. The behavior is programmable on a per-group basis using the BURST-COMPLETE field of the SMS_CLASS_ARBITER0, SMS_CLASS_ARBITER1, and SMS_CLASS_ARBITER2 registers. A per-FIFO counter tracks the number of complete bursts in a FIFO.

11.2.4.1.2.3 ExtendedGrant Feature (EXTENDEDGRANT Field in SMS_CLASS_ARBITER0, SMS_CLASS_ARBITER1, and SMS_CLASS_ARBITER2)

EXTENDEDGRANT is a programmable control field that allows a group to be granted for N consecutive transactions (single or burst), assuming the group is still requesting service (FIFO is not empty). EXTENDEDGRANT is applicable on a single/burst boundary. This multiple-service grant is intended to take advantage of the high probability of two consecutive transactions within a group accessing consecutive memory addresses (that is, in the same SDRAM page). This mechanism does not apply to consecutive transactions that have been split by the RE. The maximum number of consecutive grants is given in the EXTENDEDGRANT field of the SMS_CLASS_ARBITER0, SMS_CLASS_ARBITER1, and SMS_CLASS_ARBITER2 registers. The allowed range is 1 to 3.

The ExtendedGrant logic is in the internal class arbitration but must be propagated to the interclass arbitration. The flag qualifying a second-service request is provided to the interclass arbiter so the extended grant scheme can be applied at the second level of arbitration.

- Class 0 requests can still override the ExtendedGrant scheme of class 1/class 2. Within an ExtendedGrant window, hand-over to another class/thread (granting another thread) can occur as soon as one idle cycle appears in the thread at burst or single boundary.
- The PWM counter of the interclass arbitration obeys the ExtendedGrant completion before handing priority to the other class.
- When a split transaction from the RE is interleaved within an ExtendedGrant window, completion of the ExtendedGrant window starts with the last ExtendedGrant counter value (not the initial value), because there are two independent counters for ExtendedGrant and NOfServices. The ExtendedGrant counter is reloaded to its programmed value when it reaches 0.

11.2.4.1.2.4 NOfServices Feature (SMS.SMS_CLASS_ROTATIONm[4:0] NOFSERVICES Field)

NOFSERVICES is a programmable control field that allows consecutive transactions (single or burst) coming from the VRFB (with an RE_split qualifier) to be granted consecutively, assuming the group is still requesting service (FIFO is not empty). NOfServices is applicable to the RE single/burst boundary. The maximum number of consecutive grants is given by the NOFSERVICES field of the SMS.SMS_CLASS_ROTATIONm registers. The allowable range is 1 to 31.

The NOfServices logic is in the internal class arbitration, but must be propagated to the interclass arbitration. The flag qualifying a second-service request is provided to the interclass arbiter so the extended grant scheme can be applied at the second level of arbitration.

- Class 0 requests can still override the NOfServices scheme of class 1/class 2. Within a NOfService window, hand-over to another class/thread (granting another thread) can occur as soon as one idle cycle appears in the thread at burst or single boundary. A burst with fewer than two idle cycles cannot be interrupted by a class 0 request.
- The PWM counter of the interclass arbitration obeys the NOfServices completion before handing priority to the other class.
- When a split transaction from the RE is interleaved within an ExtendedGrant window, completion of the NOfservices window starts with the last NOfServices counter value (not the initial value), because there are two independent counters for ExtendedGrant and NOfServices.

If a transaction split by the VRFB follows a nonsplit transaction currently being executed on the SDRC side, an arbitration slot occurs on the nonsplit transaction boundary, regardless of the status of the ExtendedGrant counter. This is because the chances of a nonsplit transaction and a split transaction accessing the same SDRAM page are low. Similar behavior would be observed for a split transaction followed by a nonsplit transaction. The NOfServices counter is reloaded with its programmed value when it reaches 0.

11.2.4.1.3 Internal Class Arbitration

Class 0, class 1, and class 2 internal arbitrations are performed according to the following rules:

- Within a class, a standard least-recently-used (LRU) policy is applied. The LRU thread, if not empty, is granted when an arbitration decision occurs. LRU is applied taking into account the ExtendedGrant/NOfServices counter status. Grant is given to another nonempty LRU thread only if the current thread is serviced for ExtendedGrant/NOfServices times.
- On top of the LRU policy, a high-priority group can be defined through the SMS.SMS_CLASS_ARBITER0[7:6] HIGHPRIOVECTOR field. The high-priority group is unique and programmable. If a high-priority thread in a class, which was previously empty, starts requesting service, the current thread that has been given grant is completely serviced as per ExtendedGrant/NOfServices strategy and then the grant is given to the high-priority thread.

11.2.4.1.3.1 Interclass Arbitration

The interclass arbitration is managed using a time-varying policy driven by the following rules:

- The interclass arbitration is a PWM-like (Pulse Width Modulation) logic that defines two request-based windows:
  - Class 1 has higher priority than class 2 during M requests of class 1, where M is defined by the CLASS1PRIO parameter.
  - Class 2 has higher priority than class 1 during the next N requests of class 2, where N is defined by the CLASS2PRIO parameter. For more information on priority between classes, see Section 11.2.6.2.2.3.
- The PWM counter is decremented each time the class is processing a single 64-bit request in its high-priority window.
- A class 0 request always has the highest priority. The PWM counter is frozen while class 0 requests are being serviced.
- A class 1 transaction can be serviced during the class 2 high-priority window if class 2 is not requesting service (conversely, a class 2 transaction can be serviced during the class 1 high-priority window if class 1 is not requesting service). The PWM counter is frozen when the grant is given to the thread that is not in its high-priority window.

- NOfServices/ExtendedGrant have higher priority than the PWM counter.
- The PWM counter is reloaded with M and N when it reaches 1 and an arbitration decision must be made.
- The priority order is as follows:
  - Current burst service lock (assuming subsequent burst requests available when required)
  - Class 0
  - ExtendedGrant and NOfServices atomicity (assuming subsequent burst requests available when required)
  - Class 1 if PWM priority is to class 1; class 2 if PWM priority is to class 2
  - Class 2 if PWM priority is to class 1; class 1 if PWM priority is to class 2

11.2.4.1.4 Firewalls

Access permissions can be defined in the target memory address space on a per-initiator basis. Initiators are differentiated using the interconnect ConnID extension.

Permissions are allocated to the various initiators on a per-region basis. The memory regions are programmable using a start address and an end address that are defined with 64K-byte granularity. Up to seven distinct regions can be defined; the software must ensure that they do not overlap.

The remaining memory space (total memory space minus the protected areas) is defined as region 0.

Depending on whether the access is a read or a write, and depending on the in-band request qualifiers, a region may be given specific access permissions. When an access is received by the SMS, the access checked against the access attributes.

- The read permission is initiator-based and is controlled using the SMS.SMS_RG_RDPERMi register.
- The write permission is initiator-based and is controlled using the SMS.SMS_RG_WRPERMi register.
- The REQINFO bits taken into account are the incoming MReqInfo attributes: Debug, privilege, and attribute, along with the host parameter decoded in the SMS module (see the SMS.SMS_RG_ATTi[31:0] REQINFO field). For the SMS firewall, the host parameter is set for the MPU initiator and the sDMA initiator. The decoding of the host parameter, based on the MPU ConnID and sDMAConnID generic parameters (defined at design time), is done inside the SMS module.
- Whether the access is accepted (there is one valid bit for each ReqInfo pattern) can be specified for each ReqInfo pattern. ReqInfo permission is controlled using the region attributes register SMS.SMS_RG_ATTi[31:0] REQINFO field.

Table 11-99 lists the ReqInfo parameters ordering.

Table 11-99. ReqInfo Parameters Ordering

Host	Privilege	Reserved for Non-GP Devices	Debug	Type	Req Info	SMS.SMS_RG_ATTi[31:0] REQINFO Field
0: Nonhost 1: Host	0: User 1: Supervisor	Reserved	0: Functional 1: Debug	0: Data Transfer 1: Opcode Fetch		
N/A (1)						0b0...000000000
0	0	0	0	0	0	0b0...000000001
0	0	0	0	0	1	0b0...000000010
0	0	0	1	0	2	0b0...000000100
					...	
0	1	1	1	0	14	0b0...000001...00
0	1	1	1	1	15	0b0...00001...000
1	0	0	0	0	16	0b0...0001...0000
1	0	0	0	1	17	0b0...001...00000
					...	
1	1	1	0	1	29	0b0010...000000
1	1	1	1	0	30	0b0100...000000

(1) Access to the region is not allowed

Table 11-99. ReqInfo Parameters Ordering (continued)

Host	Privilege	Reserved for Non-GP Devices	Debug	Type	Req Info	SMS.SMS_RG_ATTi[31:0] REQINFO Field
0: Nonhost 1: Host	0: User 1: Supervisor	Reserved	0: Functional 1: Debug	0: Data Transfer 1: Opcode Fetch		
1	1	1	1	1	31	0b1000...000000

When all SMS_RG_ATTi[31:0] REQINFO bits are set to 0 the access to the region is not allowed.

Set the REQINFO[0] bit to 1 when NonHost-User-Functional-Data accesses are permitted in this region.

Set the REQINFO[1] bit to 1 when NonHost-User-Debug-Data accesses are permitted in this region.

Set the REQINFO[31] bit to 1 when Host-Supervisor-Debug-Opcode accesses are permitted in this region.

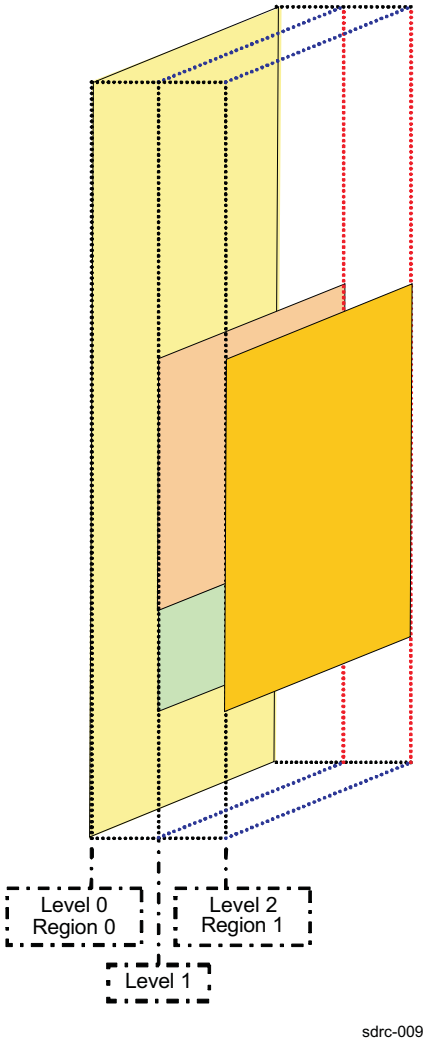
Any bit of the SMS_RG_ATTi[31:0] REQINFO field corresponds to a particular combination of the five attribute bits. When all SMS_RG_ATTi[31:0] REQINFO bits are set to 1 the region is full-access allowed.

The firewall unit performs the following checks to authorize or reject an access to the external memory:

- Compute the Region ID based on the transaction address.
- Generate a violation if overlapping of firewall regions is detected.
- From the Region ID, get the attributes for the region that has been hit.
- Check the transaction REQINFO attributes with respect to the region attributes (debug, privilege, type, and host).
- Reject the transaction if the attributes are not compatible.
- From the Region ID and the transaction qualifiers (MCmd, ConnID), check the initiator permissions for performing the access (read, write).
- Reject the transaction if there is no permission.

The protection regions are organized in priority levels (from Level 0 to Level 2) to prevent problems with overlapping region and corner cases associated with them; the lowest level has the lowest priority (see Figure 11-49).

Figure 11-49. Region Organization



Region 0 (default region containing the whole memory space): Level 0

Region 1 (allows dynamic reprogramming of regions): Level 2

Regions 2-7 (protection regions): Level 1

Region 1 has the highest priority to perform dynamic masking of other already-programmed regions when they are reprogrammed.

Overlapping regions of the same priority level is forbidden and results in a violation when an access to the concerned region occurs. This violation is reported in the error-log register. Priority-level handling is done in the hardware; it does not involve any specific software development.

All transactions are checked, including those the RE has processed.

When detecting a violation on an incoming request, the SMS respects the response ordering within the faulty thread.

A violation flag is raised internally and the MThreadID field is logged. Generation of the interconnect error response is then local to the SMS; the response buffer must be used to manage the potential response collision with regular SDRC responses.

11.2.4.1.5 Rotation Engine

Smartphone applications must often perform image rotation between the orientation of images stored in external memory and the orientation with which they must be displayed. The device offers a hardware mechanism that allows rotation tasks to be implemented efficiently, transparent to software applications, thereby keeping the MPU and DSP CPUs free for other tasks. For a description of the RE mechanism, see Section 11.2.6, *SDRC Use Cases and Tips*.

The SMS includes address processing support for rotated LCD displays (90, 180, and 270 degrees). This function is realized by the VRFB submodule, also called the rotation engine (RE) in this document.

The primary goal of the VRFB is to eliminate the SDRAM page-miss penalty when reading graphics data in nonnatural raster scan order (that is, top to bottom, bottom to top, right to left).

The 2D-rotation module (the VRFB) is included as a black box that intercepts incoming requests when addressed to the virtual frame buffer. If the address of the request targets the VRFB address space, the request is sent to the RE. It processes the address and reinserts the modified request in the SMS request path. The SMS translates the virtual address into physical SDRAM addresses and reinserts a request or multiple requests in the SMS request path to the SDRC.

Note: The use of the word *virtual* does not refer to the usual CPU-related MMU concept; the word is used in a more general context of the address remapping feature, which decouples the system from the actual storage physical organization of the graphics data in the external memory.

The VRFB can be abstracted as a 3-port module:

- Interconnect input port
- Interconnect output port
- Configuration port; all programmable control registers are part of the SMS register file, which is described in Section 11.2.4.1.5.3, *VRFB Configuration Port*.

11.2.4.1.5.1 VRFB Input Port

The VRFB receives a 29-bit address, and two decoding signals, from the SMS module. The 29-bit address is decoded to determine the context and the rotation view of the request.

The VRFB address space is a 768-MB address space split into two non-contiguous virtual address spaces:

- Address space 0: 256 MB in quarter Q1 (start address: 0x7000 0000, end address: 0x7FFF FFFF)
- Address space 1: 512 MB in quarter Q3 (start address: 0xE000 0000, end address: 0xFFFF FFFF)

It can manage up to 12 concurrent rotation contexts. Table 11-100 details the address space of each context.

Table 11-100. VRFB Contexts Virtual Address Spaces vs Rotation Angle

Context Number	0°	90°	180°	270°
0	0x7000 0000	0x7100 0000	0x7200 0000	0x7300 0000
1	0x7400 0000	0x7500 0000	0x7600 0000	0x7700 0000
2	0x7800 0000	0x7900 0000	0x7A00 0000	0x7B00 0000
3	0x7C00 0000	0x7D00 0000	0x7E00 0000	0x7F00 0000
4	0xE000 0000	0xE100 0000	0xE200 0000	0xE300 0000
5	0xE400 0000	0xE500 0000	0xE600 0000	0xE700 0000
6	0xE800 0000	0xE900 0000	0xEA00 0000	0xEB00 0000
7	0xEC00 0000	0xED00 0000	0xEE00 0000	0xEF00 0000
8	0xF000 0000	0xF100 0000	0xF200 0000	0xF300 0000
9	0xF400 0000	0xF500 0000	0xF600 0000	0xF700 0000
10	0xF800 0000	0xF900 0000	0xFA00 0000	0xFB00 0000

Table 11-100. VRFB Contexts Virtual Address Spaces vs Rotation Angle (continued)

Context Number	0°	90°	180°	270°
11	0xFC00 0000	0xFD00 0000	0xFE00 0000	0xFF00 0000

CAUTION

There is no protection in hardware for accesses outside the image resolution. Users can access the full virtual address range for a given context (there is no error reporting). Accessing outside of the image resolution creates aliasing within the image and is also translated as an additional physical memory space requirement on the external memory (some data outside of the image range can be overwritten by mistake). The following formula can be used to calculate the extra memory that will be accessed:

extra_mem_size = (2048-image_width_roundedup) x page_heigth x pixel_size
with image_width_roundedup = ROUNDUP(image_width x pixel_size/page_width) x page_width/pixel_size

11.2.4.1.5.2 VRFB Output Port

The VRFB output port can be a delayed copy of the input port, with the address field transformed. Depending on the rotated view that is active, an incoming transaction can also be split into several internal transactions.

The pipeline delay between the input and the output ports is three clock periods.

11.2.4.1.5.3 VRFB Configuration Port

The configuration port allows 12 concurrent rotation settings.

The VRFB address space is a 768 MB (256 + 512) address space split into two noncontiguous spaces. It can manage up to 12 concurrent rotation settings.

The VRFB configuration port includes all the settings required to control the 12 rotation contexts. This is an input-only port. All settings are provided from the SMS control register file.

For each of the 12 contexts, the buffer physical base address, image height and width, and pixel size can be configured through the following registers (where n is the context number, from 0 to 11):

- SMS.SMS_ROT_PHYSICAL_BAn[30:0] PHYSICALBA field
- SMS.SMS_ROT_SIZEn[26:16] IMAGEHEIGHT and SMS.SMS_ROT_SIZEn[10:0] IMAGEWIDTH fields
- SMS.SMS_ROT_CONTROLn[1:0] PS field

The memory arrangement for the pixels of a frame buffer accessed through the RE is tile-based. A tile is a rectangular array of pixels. The tile size should match the page size of the SDRAM component attached to the controller for optimal performance.

The tile size is defined using the SMS.SMS_ROT_CONTROLn[10:8] page height (PH) and [6:4] page width (PW) fields.

The image height and image width (SMS.SMS_ROT_SIZEn[26:16] IMAGEHEIGHT and [10:0] IMAGEWIDTH fields, expressed in bytes) must be multiples of the tile height and width, respectively. Some padding is required if the image size does not fit this requirement.

For a configuration example, see Section 11.2.5.1.2, *VRFB Context Configuration*.

11.2.4.1.6 Violation Reporting

The SMS firewall can detect violations and report them to the overall system by using the following qualifiers:

- The in-band error response to a non-authorized access
- The out-of-band error signal generation using two out-of-band error signals

Based on the fact that there is no way to prevent the debugger from generating firewall violations during debug, and that users cannot stop checking for functional violations, two out-of-band violation signals are set when:

- A functional violation is detected
- A debug violation is detected

These signals are asserted on each error detection from a read, write, or posted write faulty access: they are deasserted when the software clears the error bit in the SMS_ERR_TYPE register.

11.2.4.2 Module Power Saving

Power-saving is managed through the SMS.SMS_SYSCONFIG register.

The SMS_SYSCONFIG[4:3] SIDLEMODE field defines the power management strategy (force idle mode, no idle mode, or smart idle mode). See Section 11.2.4.3 for more details about the system power management.

By default, the internal interface clock gating strategy is enabled as the SMS_SYSCONFIG[0] AUTOIDLE bit is set to 0x1 after reset. When all FIFO queues are empty and no ongoing transactions remain, the L3 interconnect clock is disabled inside the SMS thus reducing power consumption. The L3 interconnect clock can be disabled after a programmable delay defined in the SMS_POW_CTRL[7:0] IDLEDELAY bit field.

When there is new activity on the interconnect interface, the interconnect clock is restarted without any latency penalty. It is recommended to enable this mode to reduce power consumption.

There is an internal interface clock gating strategy within the SDRC controller. This power-saving feature is always active.

11.2.4.3 System Power Management

The SMS can be configured through the SMS.SMS_SYSCONFIG register to be in one of these idle modes:

- No-idle mode (the SMS.SMS_SYSCONFIG[4:3] SIDLEMODE field is set to 0x1): The module never goes into idle state.
- Force-idle mode (the SMS.SMS_SYSCONFIG[4:3] SIDLEMODE field is set to 0x0): The module goes into idle state immediately after receiving the request from the PRCM.
- Smart-idle mode (the SMS.SMS_SYSCONFIG[4:3] SIDLEMODE field is set to 0x2): SidleAck is asserted once the module has confirmed there are no more outstanding transactions with the SDRC.

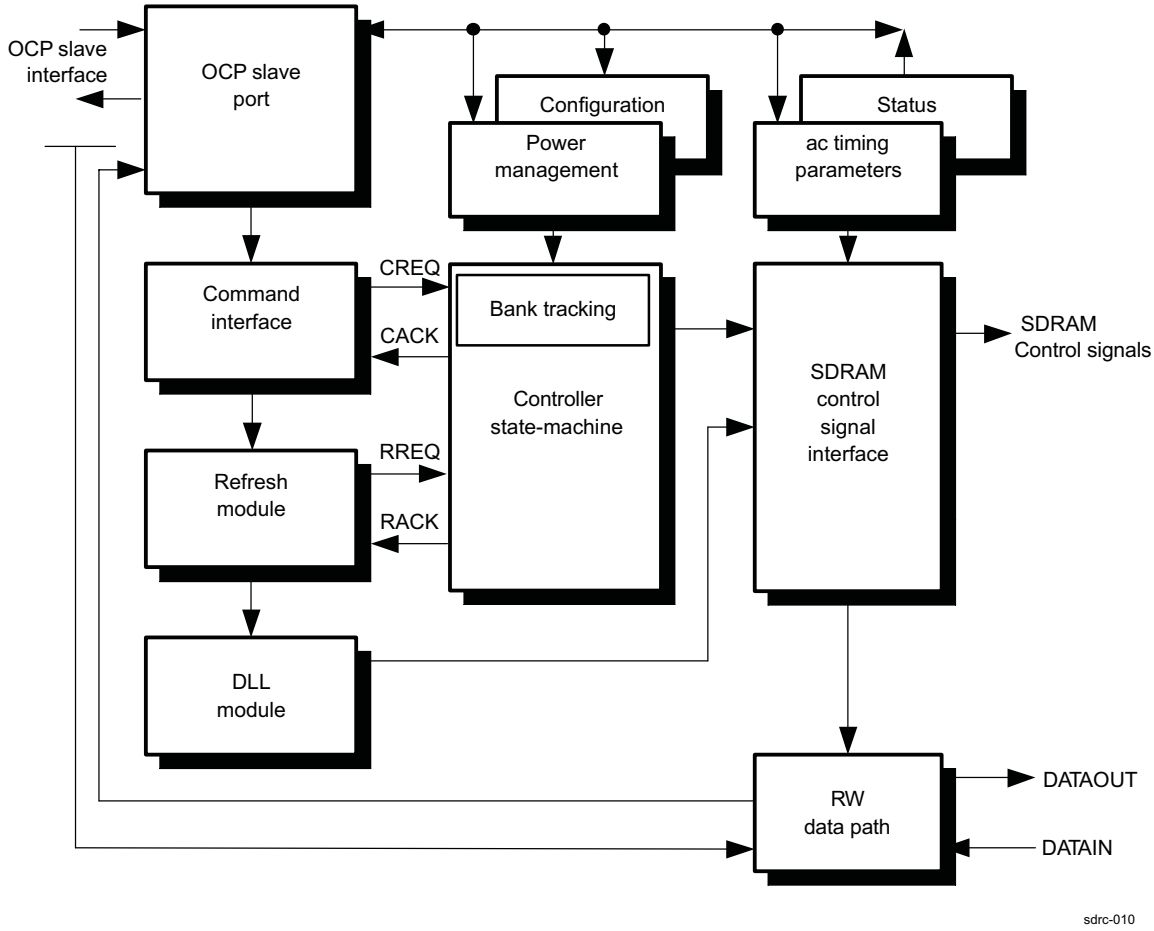
11.2.4.4 SDRC

The SDRC provides two configurable memory areas. Each supports mobile SDR SDRAM and low-power DDR SDRAM from 16 Mbits to 2 Gbits, depending on the memory organization.

Flexible row/column addressing schemes are possible with 2-bank support for 16 Mbits and 32 Mbits memories, and 4-bank support for 64 Mbits, 128 Mbits, 256 Mbits, 512 Mbits, 1 Gbit, and 2 Gbits memories.

Figure 11-50 shows the architecture of the SDRC.

Figure 11-50. SDRC Architecture



11.2.4.4.1 CS0-CS1 Memory Spaces

11.2.4.4.1.1 Chip-Select 0 Start Address

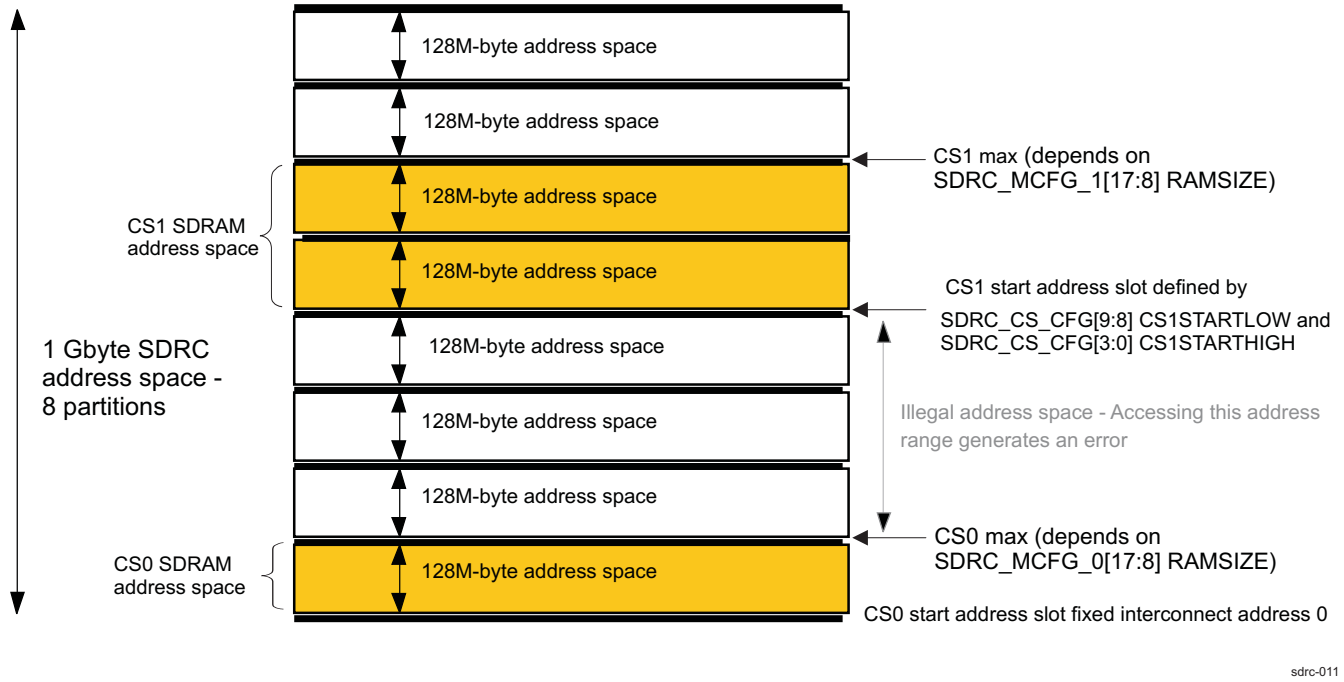
- CS0 always starts at address 0 with respect to the local interconnect address.
- The valid CS0 range is 0–CS0max, where CS0max is defined in the SDRC.SDRC_MCFG_p[17:8] RAMSIZE field where p = 0 (for CS0), and by the number of banks.

11.2.4.4.1.2 Chip-Select 1 Start Address

- CS1 start address is programmable.
- The default base address for CS1 after reset is defined in the register description.
- The SDRC 1G-byte/8G-bit address space is segmented so that 7 possible CS1 start address locations (8 in total minus 1 reserved for CS0) are defined by the SDRC.SDRC_CS_CFG[3:0] CS1STARTHIGH field as shown in Figure 11-51.
- Each 128M-byte address space is also segmented into 32M-byte address spaces defined by the SDRC.SDRC_CS_CFG[9:8] CS1STARTLOW field so that 64 possible CS1 start address locations are defined by the SDRC.SDRC_CS_CFG[3:0] CS1STARTHIGH and SDRC.SDRC_CS_CFG[9:8] CS1STARTLOW fields.
- The valid CS1 range is:

  CS1 start address slot–CS1max, where CS1max is defined by SDRC.SDRC_MCFG_p[17:8] RAMSIZE where p = 1 (for CS1) and SDRC.SDRC_CS_CFG registers.

Figure 11-51. CS0/CS1 Chip-Select Start Address Slots



11.2.4.4.1.3 SDRAM Memory Combinations on CS1 and CS0

The combinations of SDR and DDR memories/SDRAMs are defined as follows:

- SDR and DDR memories can be connected on either CS1 or CS0.
- The only restriction on the coexistence of SDRAMs on CS0 or CS1 is that a combination of SDR on one CS and DDR on the other CS is not allowed.

The SDRAM data bus width on each CS is determined by the SDRC.SDRC_SHARING[11:9] CS0MUXCFG and SDRC.SDRC_SHARING[14:12] CS1MUXCFG fields of the memory-sharing registers.

11.2.4.4.2 Bank Tracking

The main state-machine controls all the accesses to external memories.

The SDRC contains hardware for tracking open pages on a per-bank basis. Up to four open pages are tracked per CS, for a maximum of eight open pages tracked.

To pipeline accesses efficiently, the SDRC includes a request look-ahead FIFO that analyzes interconnect requests with respect to the status of the target banks. A bank status can be any of the following:

- Bank open on another row
- Bank closed
- Bank open on the same row

The SDRC state-machine generates the appropriate sequence of memory commands. All precharge and active commands are hidden as much as possible, to optimize the memory bandwidth usage.

The look-ahead FIFO depth is 9 ×64-bit requests, with a limit of four different transactions. As soon as the look-ahead FIFO stores four complete transactions or when it is full, the SCmdAccept is deasserted and any incoming request is blocked.

Requests sent to the SDRC are treated in order. The four transactions limit in the SDRC look-ahead FIFO ensures that high-priority requests performance is not hampered by a succession of SDRAM page close/page open cycles due to previous lower priority requests already accepted by the SDRC. With this limit, requests not accepted by the SDRC will have to go through the SMS arbitration at a later time. If a high-priority request has arrived in the SMS in the mean time, it will be passed to the SDRC before any lower-priority request, as defined in the regular SMS arbitration scheme.

11.2.4.4.3 Address Multiplexing

A flexible address scheme allows for the support of any new type of SDRAM address multiplexing and density.

The programming model has changed but is still compatible with the legacy fixed address scheme. Both chips, hence both CS must use the same address scheme (fixed or flexible), but can use different address multiplexing configurations.

A dedicated bit controls the address scheme used: the legacy fixed address scheme or the new flexible address scheme:

- The legacy fixed address scheme is selected when the SDRC.SDRC_MCFG_p[19] ADDRMUXLEGACY bit is set to 0 (where p = 0 or 1 for SDRC CS0 or CS1).
- A new flexible address-muxing scheme configuration is selected with the SDRC.SDRC_MCFG_p[19] ADDRMUXLEGACY bit set to 1 (where p = 0 or 1 for SDRC CS0 or CS1).

To use this new flexible address scheme, configure the row address width and the column address with the SDRC.SDRC_MCFG_p[26:24] RASWIDTH and SDRC.SDRC_MCFG_p[22:20] CASWIDTH fields.

For more information about fixed address multiplexing configurations for SDRAM components, see Section 11.2.2.2.3, *Address Multiplexing*.

11.2.4.4.4 Bank Allocation Setting

11.2.4.4.4.1 System Address Decoding

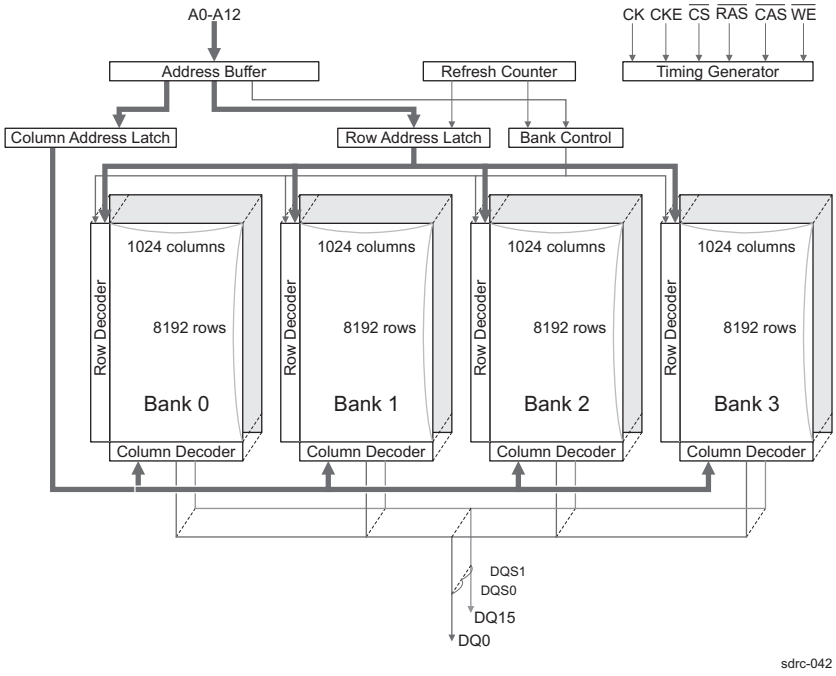
The SDRC has a 64-bit slave interface connected to the L3 main system interconnect. The regular allocation is to see the system address bus as the concatenation bank-row-column. The SDRC controller translates the interconnect request (read or write to memory for instance) into a series of commands and bank, row, column signals. The commands are sent to the external SDRAM through the sdrc_ncs, sdrc_nras, sdrc_ncas and sdrc_nwe signals. The bank, row and column addresses are sent through sdrc_a and sdrc_ba with the associated command. Those words can be defined as follow:

- Bank: the address of one of the four (or two) banks
- Row: the address of a page
- Column: the address of a word in a page

The bank signals are used to select which of the bank is accessed when transferring data from or to the SDRAM. In the first cycle, the SDRAM memory latches the row address (nRAS low): the adequate bank and row are activated. Next cycle, the SDRAM memory latches the column address (nCAS low): the adequate column is activated. The address is hence decoded: memory cells are sensed by sense amplifier and data is read from or written to output buffers.

Figure 11-52 shows a block diagram of a 512 Mbits SDRAM memory (8M x 16 x 4 banks). The data bus DQ[15:0] has a 16-bit width. Rows are addressed through A0-A12 (8192 rows) and columns through A0-A9 (1024 columns).

Figure 11-52. SDRAM Controller Block Diagram



11.2.4.4.4.2 SDRC Controller Commands

Before any read or write command can be issued to a bank in the external memory, a row in that bank must be opened. This is accomplished by the active command, by which the bank and the row are selected. More than one bank (up to four according to the memory used) can be active at one time. Once a row is opened, a read or write command can be issued to that row. The row remains active until a precharge or read/write to another row in the same bank or a refresh to the bank occurs.

Autorefresh command is used during normal operation mode. This command is non-persistent (that is, it must be issued each time a refresh is required). The device requires a refresh of all rows in a periodic interval. This command takes some time (according to the memory used), and during this phase no read or write command can be processed. A precharge-all (that is, a precharge command affecting all banks) is issued before any autorefresh sequence.

An active command to a row of a bank for which another row is already active can be issued only after the previous row has been closed. The precharge command is used to deactivate the open row in a particular bank. The bank is available for a subsequent row access some time after the row precharge command is issued. A minimum time is needed to close and open a new row.

A subsequent active command to another bank can be issued while the first bank is being accessed without closing the row in the first bank. This results in a reduction of row access time in the same bank. In this case, it is not necessary to deactivate (with a precharge command) the row in the other banks. Up to four banks, depending on the memory used, can be activated at the same time. In each bank, one row can be selected at a time.

11.2.4.4.4.3 BANKALLOCATION Parameter

To optimize SDRAM memory accesses in a throughput point of view, the SDRC controller supports various bank-row-column allocation. The bank-row-column allocation choice depends on many parameters, such as the number of initiators in the use case, the memory usage (accesses bandwidth, frequency), and so on.

The SDRC controller not only supports the regular allocation where the system address bus is seen as the concatenation bank-row-column, but it also supports two other types of allocation. The SDRC_MCFG_p[7:6] BANKALLOCATION bit field (where p = 0 or 1 for CS0 or CS1) selects the type of allocation. This feature modifies the bank, row, and column address decoding order:

- BANKALLOCATION = 0x0: Bank-row-column
- BANKALLOCATION = 0x1: Bank1-row-bank0-column
- BANKALLOCATION = 0x2: Row-bank-column

Only the flexible address-muxing scheme allows choosing different bank mapping allocations throughout the BANKALLOCATION parameter (that is, the legacy address-muxing scheme does not).

The flexible address-muxing scheme (SDRC_MCFG_p[19] ADDRMUXLEGACY = 0x1) allows choosing different bank mapping allocations. The SDRC.SDRC_MCFG_p[7:6] BANKALLOCATION bit field (p = 0 or 1 for CS0 or CS1) defines the ordering of the bank, and the column and row decoding of the incoming system address. The BANKALLOCATION bit field can be set on a CS basis.

In the usual allocation, the system address bus is seen as a concatenation of bank-row-column. For instance, consider the SDRAM used in Section 11.2.6.3, *Typical SDRC Connection to an External SDRAM Device*: a 512- Mbit SDRAM (8M x 16 x 4 banks). The system address received from the interconnect into the SDRC is composed of:

- 2 system address bits used as the bank address (going to SDRC controller sdrc_ba[1:0] pins)
- 13 system address bits used as the row address (going to SDRC controller sdrc_a[12:0] pins)
- 10 system address bits used as the column address (going to SDRC controller sdrc_a[12:0] pins)

Setting the BANKALLOCATION bit field to 0x1 or 0x2 changes the order of the system address decoding, as shown in Figure 11-53.

Figure 11-53. Address Multiplexing Scheme According to BANKALLOCATION

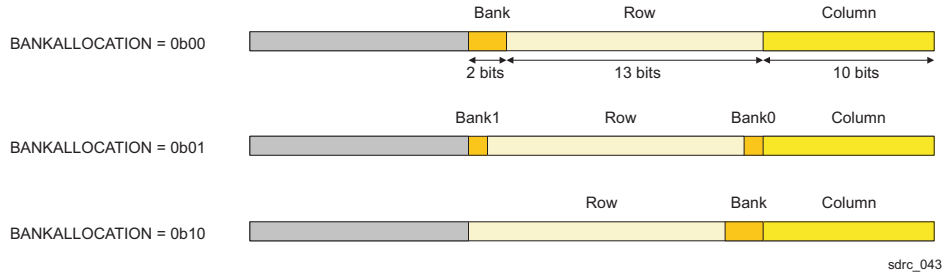
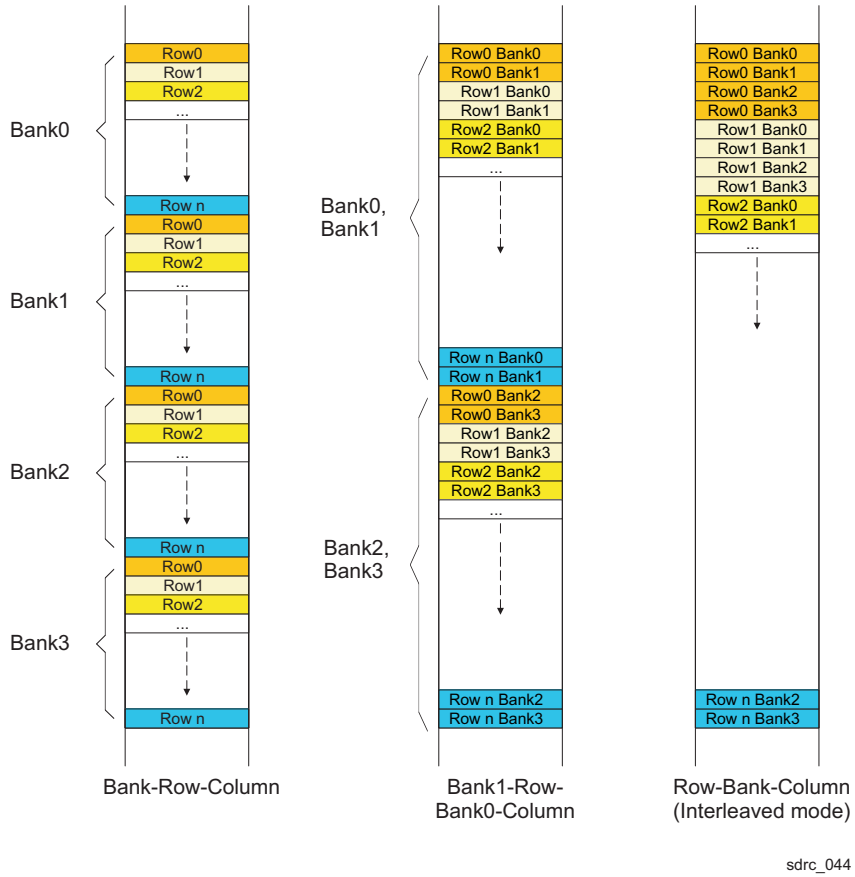


Figure 11-54 compares bank-row-column and row-bank-column memory accesses.

Figure 11-54. Simplified View of Bank-Row-Column vs Row-Bank-Column Bank Allocation



The latency to charge another row in the same bank (that is, to close the opened page and to open another one) is bigger than the latency involved with an access to a given row in another bank. By moving bank select signals ahead of column and after row addresses, bank interleaving is likely to happen more often. More frequent use of the interleaved mode means less overall SDRAM access time and, thus, yields to better overall performance.

In some use cases, several initiators access the same bank (bank0 for instance) in external SDRAM memory at the same time. With this kind of scenario, a lot of penalties are added because of rows opening and closing. The BANKALLOCATION setting improves the latency for reading and writing operations by optimizing memory accesses through the reduction of deactivate sequence use, thereby reducing time penalties.

The bank1-row-bank0-column allocation is a good compromise between the legacy bank allocation (bank-row-column) and the full interleaving bank allocation (row-bank-column). In some use cases, it may be necessary to keep only half of the memory refreshed (on bank0 and bank1 for instance) while the other two banks are unused. In full interleaving bank allocation, the memory must be refreshed through the self-refresh mode, while with bank1-row-bank0-column, the memory can be refreshed through self-refresh mode or through partial array self-refresh mode.

In the following, an example of latency is given based on a 512-Mbit SDRAM memory data sheet. Table 11-101 gives the duration for some AC timing parameters. The frequency used in the example is 133 MHz (that is, tCK = 7.5 ns). The CAS latency = 3.

Table 11-101. Mobile DDR SDRAM AC Timing Parameters

AC Timing Parameter	Description	Duration (ns)
tRFC	Autorefresh cycle time	80 (min)
tRP	Row precharge time	22.5 (or 3 tCK)

Table 11-101. Mobile DDR SDRAM AC Timing Parameters (continued)

AC Timing Parameter	Description	Duration (ns)
tRC	Row cycle time	67.5 (or 9 tCK)
tRAS	Row active time	45 (or 6 tCK)
tRCD	nRAS to nCAS delay time	22.5 (or 3 tCK)
tRRD	Row active to row active delay time	15 (or 2 tCK)

The latency to access another row in the same bank depends on tRP and tRAS timings because a precharge command followed by an active command must be issued. Therefore, 6 tCK are needed to close the current row and open another one.

When accessing another bank, the latency to access another row depends on tRAS timing only because the closing of the current row is not mandatory. In other words only an active command must be issued. Therefore, 3 tCK are needed to open another row in a different bank.

Single initiator use case:

When the initiator wants to access a page in a bank for the first time, it opens only this page. It takes 3 tCK. When the initiator is done with this page, it closes this page and opens another one. It takes 6 tCK.

Two initiators use case:

When one initiator wants to access a page in the same bank, it must necessarily close the current page and open the page it wants to access. It takes 6 tCK.

When one initiator wants to access a page this time in another bank, there can be less page opening and closing. For instance, if the initiator wants to open a page in another bank where no page is open, it takes only 3 tCK, rather than 6 tCK if it was another page in the same bank.

11.2.4.4.5 Data Multiplexing During Write Operations

11.2.4.4.5.1 External Bus Combinations

The SDRC pin allocation scenarios are provided in Table 11-102. These scenarios are defined on a per-CS basis for maximum flexibility. The pin allocation configurations allow implementation of combinations of the 16-/32-bit external interfaces listed in this table. The data multiplexer receives a 64-bit word from the SDRAM command queue and partitions the data into a series of 16-bit or 32-bit accesses. The data multiplexer also steers the data to the appropriate data lane. The data partitioning and data steering are determined by the SDRC.SDRC_SHARING[11:9] CS0MUXCFG and SDRC.SDRC_SHARING[14:12] CS1MUXCFG fields.

Table 11-102. SDRC Data Lane Configurations

Device pins	sdrc_d[31:24]	sdrc_d[23:16]	sdrc_d[15:8]	sdrc_d[7:0]	
	DataLane[31:16]		DataLane[15:0]		
	DQS3	DQS2	DQS1	DQS0	
	DQM3	DQM2	DQM1	DQM0	SDRC.SDRC_SHARING[14:9] CSnMUXCFG field
	D[31:0]				0x0, 0x1
	D[15:0]				0x2, 0x7
			D[15:0]		0x3
					0x4, 0x5, 0x6: Reserved

11.2.4.4.5.2 Endianness-Aware Unpacking

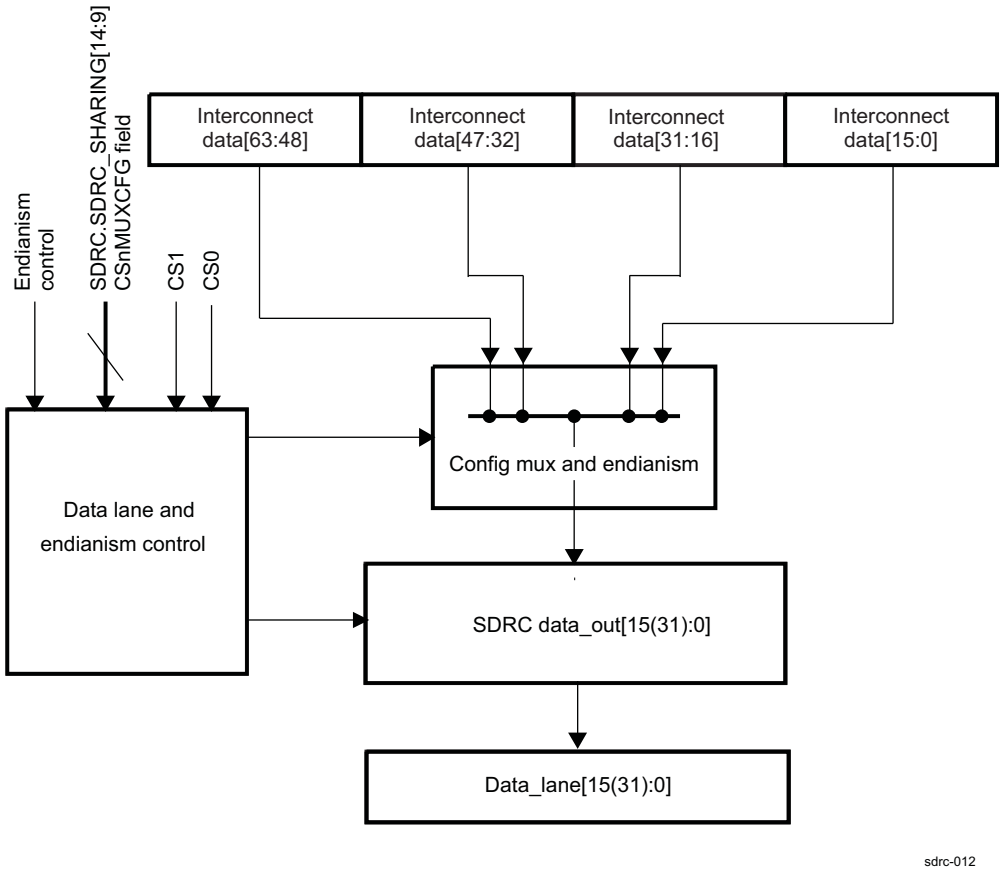
Data transactions can be either big or little endian; the transaction endianness is determined by an in-band interconnect qualifier. Muxing 64-bit data to 16-/32-bit data is performed according to this qualifier.

For a 64-bit interconnect little-endian write transaction on a 16-/32-bit memory, Data[15(31):0] is written at the lowest memory address, and Data[63:48(32)] is written at the highest memory address.

For a 64-bit interconnect big-endian write transaction on a 16-/32-bit memory, Data[15(31):0] is written at the highest memory address, and Data[63:48(32)] is written at the lowest memory address.

Figure 11-55 shows the data multiplexing scheme.

Figure 11-55. Data Multiplexing Scheme



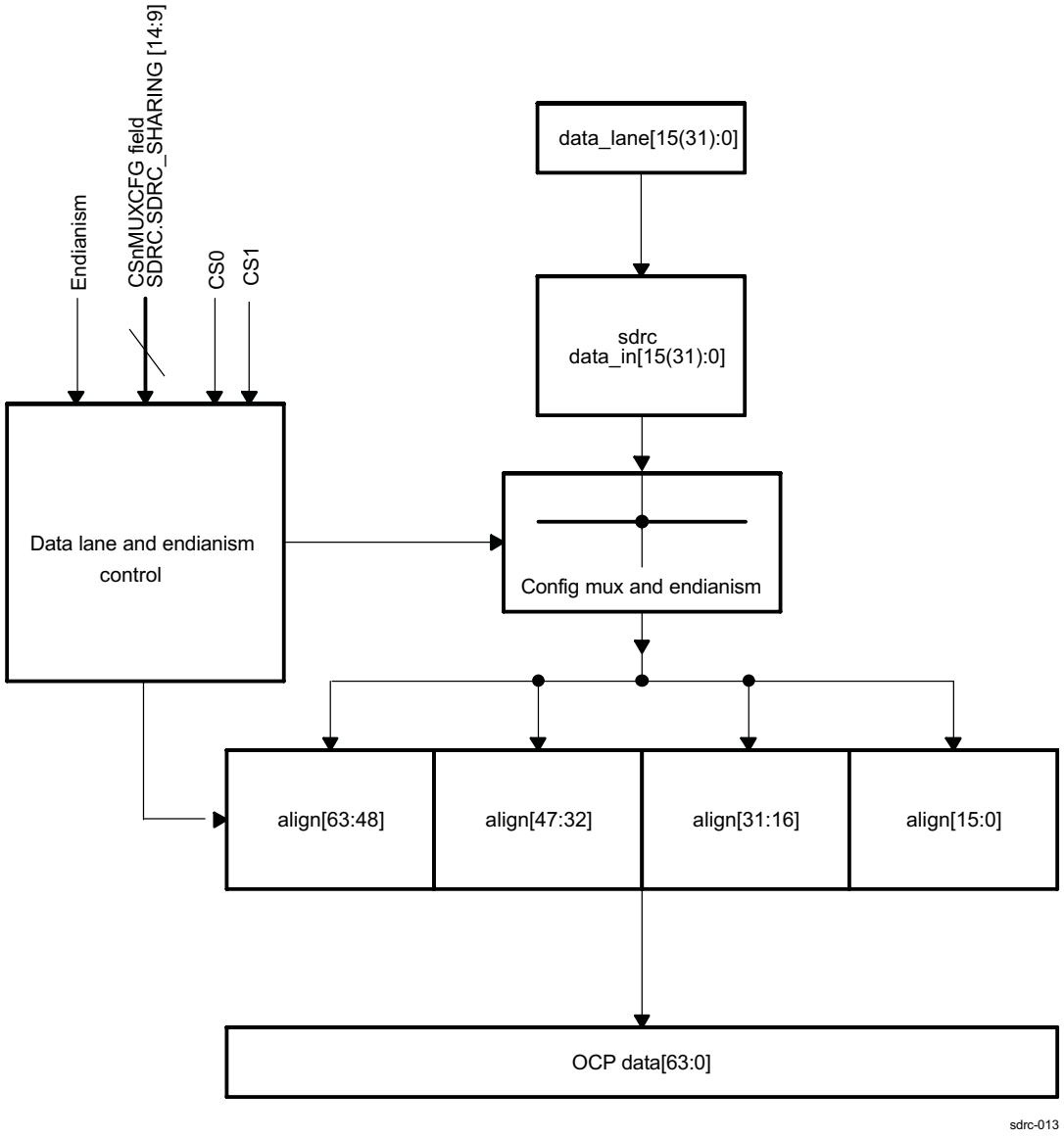
11.2.4.4.6 Data Demultiplexing During Read Operations

11.2.4.4.6.1 External Bus Combinations

The SDRC pin allocation scenarios are shown in Table 11-102. These scenarios are defined on a per-CS basis for maximum flexibility. The pin allocation configurations allow implementation of combinations of 16-/32-bit external interfaces. The data demultiplexer receives 16-/32-bit data from the relevant SDRC data lane, and then performs a data packing function. The packing function formats the data into 64-, 32-, 16-, or 8-bit format. The data demultiplexer also steers the data from the appropriate data lane. The data partitioning and data steering are determined by the SDRC.SDRC_SHARING[11:9] CSOMUXCFG and SDRC.SDRC_SHARING[14:12] CS1MUXCFG bit fields.

Figure 11-56 shows the data demultiplexing scheme.

Figure 11-56. Data Demultiplexing Scheme



11.2.4.4.6.2 Endianness-Aware Packing

Data transactions can be big or little endian; their endianness is determined by an in-band interconnect qualifier. Demuxing 16-/32-bit memory data to 64-bit interconnect data is performed according to this qualifier.

For a 64-bit interconnect little-endian read transaction on a 16-/32-bit memory, Data[15(31):0] is read from the lowest memory address, and Data[63:48(32)] is read from the highest memory address.

For a 64-bit interconnect big-endian read transaction on a 16-/32-bit memory, Data[15(31):0] is read from the highest memory address, and Data[63:48(32)] is read from the lowest memory address.

To preserve data integrity in all situations, that is, regardless of the effective scalar size of the transferred data (byte, Word16, Word32, or DWord64), the endianness specified during the read operation must match that specified during the write operation. If there is no match, the packing and unpacking operations are not consistent. There is no attempt to perform endianness conversion in the SDRC; only endian-aware width conversion is performed.

11.2.4.4.7 Refresh Management

Refresh management can be divided as follows:

- Self-refresh management
- Autorefresh management (programmable refresh period)

11.2.4.4.7.1 Self-Refresh Management

Self-refresh is entered to place an attached SDRAM into an autonomous refresh mode, typically when the OMAP device enters an idle mode and switches off the SDRAM clock.

In self-refresh, the SDRAM supplies the row address generation required to refresh and retain data in the absence of external clocking.

Self-refresh can be entered using any of the following methods:

- Manually, under software control per CS
- Upon a reset event: Automatically on a warm reset event (if the SDRC is programmed to enter self-refresh on warm reset)
- Upon a hardware request: Automatically on an idle request sent by the system power manager (if the SDRC is programmed to enter self-refresh on idle request)
- Automatically after a (programmable) period of inactivity on the interconnect interface (if enabled by setting the SDRC.SDRC_POWER_REG[5:4] CLKCTRL bit field to 0x2)

Self-refresh can be exited as follows:

- Manual software command (exit from self-refresh mode; see Section 11.2.5.3.4, *DLL/CDL Configuration*)
- Automatically after receiving a read or write transaction to access memory

11.2.4.4.7.2 Autorefresh Management

When autorefresh is enabled, a programmable hardware counter within the SDRC generates a periodic event that triggers either a single refresh or a burst of consecutive refresh commands. This is the standard refresh mode used when the system is active, while the running applications regularly access the SDRAM.

An autorefresh command can also be applied using the SDRC.SDRC_MANUAL_p register (see Section 11.2.5.3.4, *DLL/CDL Configuration*). This method can be used to generate a device-specific initialization sequence after power up or after the memory device exits from a low-power mode (self-refresh or deep-power-down).

The autorefresh request period is a user-controlled parameter programmed to meet the specification of the memory devices. Each time an autorefresh request is issued, the SDRC can service the autorefresh using any of the following commands:

- Single autorefresh command
- Burst of four autorefresh commands
- Burst of eight autorefresh commands

When a burst of four or eight is selected, the programmed period value is automatically scaled in hardware by 4 or 8. Consequently, the value to be programmed does not depend on the selected burst length.

11.2.4.4.8 System Power Management

Unlike the SMS, the SDRC can be configured only in smart-idle mode by setting the SDRC.SDRC_SYSCONFIG[4:3] IDLEMODE field to 0x2. Once all asserted output interrupts are acknowledged, the SDRC goes into idle state after it receives the request from the PRCM module.

The SDRC acknowledge is conditioned by the internal activity of the SDRC.

Note: As soon as the SDRC goes out of idle state, stalled accesses can be accepted and processed:

- In SDR mode, the access can be immediately processed (unstalled).
- In DDR fixed delay mode, the accesses is processed (unstalled) after expiration of MODEFIXEDDELAYINITLAT.

11.2.4.4.9 Power-Saving Features

In the SDRC there are three ways to save power and they can be applied simultaneously:

- Page opening/closure policy
- Dynamic low-power mode
- Static low-power mode

11.2.4.4.9.1 Page Opening/Closure Policy

The OMAP device supports only one page policy. The SDRC.SDRC_POWER_REG[0] PAGEPOLICY bit must be set to 1.

The SDRC tracks open pages, if any, and determines whether the current access is to an open or a closed page. If the accessed page is open, the SDRC executes the access immediately. The SDRC performs the following procedure:

1. If the current page is already open on this bank the SDRC automatically issues a *precharge* command to close that bank.
2. Opens the accessed page by issuing an *active* command to that bank
3. Executes the access by issuing a *read* or *write* command

Up to four pages can be open simultaneously with a limit of one page per bank. The pages remain open until one of the following occurs:

- New read or write request to another page in the same bank
- Autorefresh request (a *precharge all* command is issued first)
- Self-refresh entry request (a *precharge all* command is issued first)
- Manual *precharge all* command

11.2.4.4.9.2 Dynamic Low-Power Operating Modes

The dynamic low-power operating modes of the SDRC are designed to:

- Control the external SDRAM clock(s)
- Control the internal clock gating of the SDRC when the interconnect interface is idle
- Control the self-refresh functionality

The external SDRAM is controlled through the SDRC.SDRC_POWER_REG[3] EXTCLKDIS and SDRC.SDRC_POWER_REG[2] PWDENA bits. The EXTCLKDIS bit is used to disable the external clock when no access is ongoing on the memory interface, whereas the PWDENA bit is used to activate the power-down mode of the target memory by pulling the relevant CKE low each time the memory interface is idle.

When the PWDENA bit is enabled but the EXTCLKDIS bit is not enabled, the SDRC still provides a free-running clock to the external memories: clock gating is done internal to the memory component for power savings.

EXTCLKDIS should be modified only when no access is in progress on the SDRAM interface. Software control is required to make sure the interface is idle.

CKE is dynamically controlled based on the current memory command. There is a zero-latency penalty when this mode is enabled.

The SDRC has three modes of automatic internal clock-gating behavior when the interconnect interface is idle, that is. there are no outstanding active transactions in progress. These modes are controlled through the SDRC.SDRC_POWER_REG[5:4] CLKCTRL and SDRC.SDRC_POWER_REG[23:8] AUTOCOUNT fields.

- Mode 0: The autoclock gating feature is disabled. No internal clock gating is performed in the SDRC in response to the detection of an idle state on the interconnect interface.
- Mode 1: A 16-bit counter starts decrementing when an interconnect idle condition is detected. The counter start value is loaded from the AUTOCOUNT 16-bit field. When this counter times out, internal clock gating within the SDRC is enabled.
- If an interconnect active command is received before the counter times out, or if an internal autorefresh request is issued, the procedure is aborted, the counter stops decrementing, and the request is serviced immediately.
- Mode 2: This mode is similar to mode 1, but before the internal clock gating, the SDRC places the SDRAM into self-refresh mode and turns off the external SDRAM clock. This is the lowest power mode.

To achieve maximum power savings, TI recommends the use of the PWDENA-, EXTCLKDIS-, and CLKCTRL-related features.

Table 11-103 explains the different power-saving configurations that can be programmed with the SDRC.SDRC_POWER_REG[3] EXTCLKDIS, SDRC.SDRC_POWER_REG[2] PWDENA, and SDRC.SDRC_POWER_REG[5:4] CLKCTRL bits.

Table 11-103. Dynamic Power Saving Configurations

CLKCTRL	EXTCLKDIS	PWDENA	CKE	External SDRC CLK(1)	SDRAM State	Latency When Exiting Power Mode
0	0	0	Always high	Always on	Keep previous state	
0	0	1	Low when no access	Always on	Power-down	Zero-latency penalty
0	1	0	Always high	Off when no access	Keep previous state	
0	1	1	Low when no access	Off when no access	Power-down	One cycle penalty
1	0	0	Always high	Always on	Keep previous state	
1	0	1	Low when no access	Always on	Power-down	Zero-latency penalty
1	1	0	Always high	Off when no access	Keep previous state	
1	1	1	Low when no access	Off when no access	Power-down	One cycle penalty
2	0	0	Low when no access	Off when no access after AUTOCOUNT expiration	Enter self-refresh after AUTOCOUNT expiration	
2	0	1	Low when no access	Off when no access after AUTOCOUNT expiration	Enter self-refresh after AUTOCOUNT expiration	
2	1	0	Low when no access	Off when no access	Enter self-refresh after AUTOCOUNT expiration	
2	1	1	Low when no access	Off when no access	Enter self-refresh after AUTOCOUNT expiration	

(1) EXTCLK can be set to 1 all the time for power optimization purposes, except when manual commands are sent. In this case, users must set EXTCLKDIS to 0 to ensure that a clock signal is provided to the memory device.

Note: When connected to a DDR memory, the SDRC never gates the clock provided to the DLL components so that the DLL remains locked during these idle modes. This avoids the maximum of 500 clock cycles latency required for relocking the DLL when the DLL clock is switched off.

All settings for the power-saving features are common to the two CSs. When two CSs are used, however, only the accessed CS exits self-refresh or deep-power-down mode.

If the SDRC.SDRC_POWER_REG[6] SRFRONIDLEREQ bit is enabled, the SDRC enters self-refresh mode on a hardware idle request from the PRCM. The memory clock is automatically switched off, after which the SDRC sends an acknowledge back to the PRCM. In this situation, the power manager can switch the SDRC clock off. Therefore, if the SDRC is connected to a DDR memory, and if the DLL is enabled and in TrackingDelay mode, the SDRC waits for the lock status bit of the DLL to be asserted before accessing the memory when the system exits the idle state. The WAKEUPPROC bit of the SDRC_POWER_REG register enables the SDRC to automatically wait for 500 cycles (DLL relocking maximum time) before accessing the memory instead of using the LOCK signal. These 500 wait cycles are obeyed only when the DLL is set in TrackingDelay mode. For SDR mode and DLL ModeFixedDelay mode, the access is processed immediately. This mechanism is independent of the CLKCTRL field.

Note: DLL Behavior Upon a Warm Reset Assertion:

Upon a warm reset event, the DLL is disabled and is no longer locked (the SDRC_DLLA_CTRL[3] ENADLL and SDRC_DLLA_STATUS[2] LOCKSTATUS bits are reset). The input clock runs at low frequency. To prevent the DLL from remaining unlocked, the software must re-enable the DLL by setting the ENADLL bit to 1. Then, after a maximum of 500 cycles latency required to relock it, the DLL is locked (LOCKSTATUS bit = 1). Thus, the DLL is locked on the stable input clock and, therefore, ready to process incoming requests so that the SDRC can again access the memory.

11.2.4.4.9.3 Static Low-Power Operating Modes

The software-driven controls for low-power operation modes include:

- Possibility to put the memory in self-refresh using the manual command register (SDRC.SDRC_MANUAL_p (where p = 0 or 1 for SDRC CS0 or CS1). Each CS can be controlled independently. If both CSs are in self-refresh, the external SDRAM clock can be switched off by setting the SDRC.SDRC_POWER_REG[3] EXTCLKDIS control bit to 1. Self-refresh can be exited automatically if an access is initiated onto the CS. Only DPD must be exited manually.
- Possibility to put the memory in deep-power-down mode, if supported by the SDRAM, using the manual command register. Each CS can be controlled independently. The external SDRAM clock can be switched off if both CSs are either in self-refresh or deep-power-down mode by setting the SDRC.SDRC_POWER_REG[3] EXTCLKDIS control bit to 1. Another manual command must be used for the memory to exit deep-power-down mode. After a memory exits from that mode all data are lost, and a full initialization sequence must be sent to the device, before it can be used.

11.2.4.4.10 SDRC Power-Down Mode

In some applications, the SDRC power domain can be powered down while the external memory is in self-refresh mode.

When the SDRC is powered off, an isolation stage prevents an unwanted exit from self-refresh when the context is restored. SDRC outputs that control the SDRAM are set to maintain self-refresh or deep-power-down (CKE low).

When a reset occurs, the default reset state of the SDRC is power-down enable (PDE).

When exiting the off mode:

- Power is restored to the SDRC.
- The software reconfigures all registers. If the NOMEMORYMRS bit is set, the MR and EMR registers can be set through the SDRC.SDRC_MR_p and SDRC.SDRC_EMR2_p registers (see Section 11.2.4.5, *Mode Registers*).
- Exit from self-refresh mode is achieved through the SDRC.SDRC_MANUAL_p CMDCODE field. Because exit from the self-refresh field is unconditional (that is, the current state of the SDRC state-machine is not considered), ensure that autorefresh is disabled.

Once the context is successfully restored, the software can reinitialize the SDRAM or return the SDRAM to self-refresh. When the device successfully exits self-refresh, autorefresh must be re-enabled.

11.2.4.4.11 Controlled Delay Line

The DLL/CDL module is a hard macro composed of one master delay-locked loop (DLL) and five slave-controlled delay lines (CDL). It is used to generate precise delays suitable for DDR read and write operations. DLL delays are tracked at high frequency on process dispersion, and voltage and temperature variations (PVT) .

A CDL is a component with a clock input signal, a clock output signal, and a delay value input. The output signal is the input signal delayed according to the delay value.

The DLL output is a command that controls the CDLs (plus the controlled voltage) and assures an output signal with 90-degree delay with respect to its input signal. The DLL contains five CDL blocks.

11.2.4.4.11.1 Purpose of the DLL/CDL Module

In DDR applications, the DLL and CDL combination helps provide a data strobe (DQS) with a delay suitable to the main read and write RAM operations that exceed 83 MHz. The DLL functions within a locking range of 83 to 166 Mhz. Below 83 MHz, the DLL must be used in unlocked mode. For lower clock frequencies, set the DLL to bypass mode.

DLL/CDL is used to delay the incoming DQS in case of DDR read, or delay the output DQ (data lines) in case of DDR write (and, hence, to increase the ac timing margin). DQS is used only with DDR memory. There is no need to use the DLL/CDL for the SDR DRAM because data is strobed every clock cycle. The DQS signals are left unconnected for SDR SDRAM memories.

DQS is propagated with the data (thus reducing the impact of the propagation delay) and is used by the receiver to sample the data.

The DLL/CDL combination minimizes the negative effects caused by skews and jitters of clock signals. The delay introduced by the CDL base unit depends on PVT conditions. Moreover, the CDL timing delay is not a linear function of the DLL counter offset. By means of the DLL feedback loop, the delay value is updated in real time and is adjusted according to voltage and temperature variations.

DDR interfaces transmit data on both edges of the DQS bidirectional data strobe. Address and control signals transmit at half the data frequency (that is, at the DDR clock frequency) and latch only on the rising edge of the transmit clock.

The bidirectional data strobe (DQS) is transmitted externally, along with data, for use in data capture at the receiver. sdrc_dqs[3:0] is an SDRC I/O that connects the SDRC with DDR SDRAM DQS pins. See Figure 11-43 for an overview of DDR SDRAM connection with the SDRC controller. DQS is transmitted by the DDR SDRAM during reads and by the SDRC during writes. DQS is edge-aligned with data for reads and center-aligned with data for writes, as shown in Figure 11-57. Another DDR write path can be selected with the WRITEDDRCLKX2DIS bit of the SDRC_DLLA_CTRL register. This path uses a double frequency input clock coming from the PRCM to achieve proper generation of MDDR write data bytes, center-aligned with the corresponding DQS lines. It is recommended to set the WRITEDDRCLKX2DIS bit to 0 to use this path.

Figure 11-57 shows the generic DDR data-write and data-read waveforms.

Figure 11-57. Generic DDR Data-Write and Data-Read Waveforms

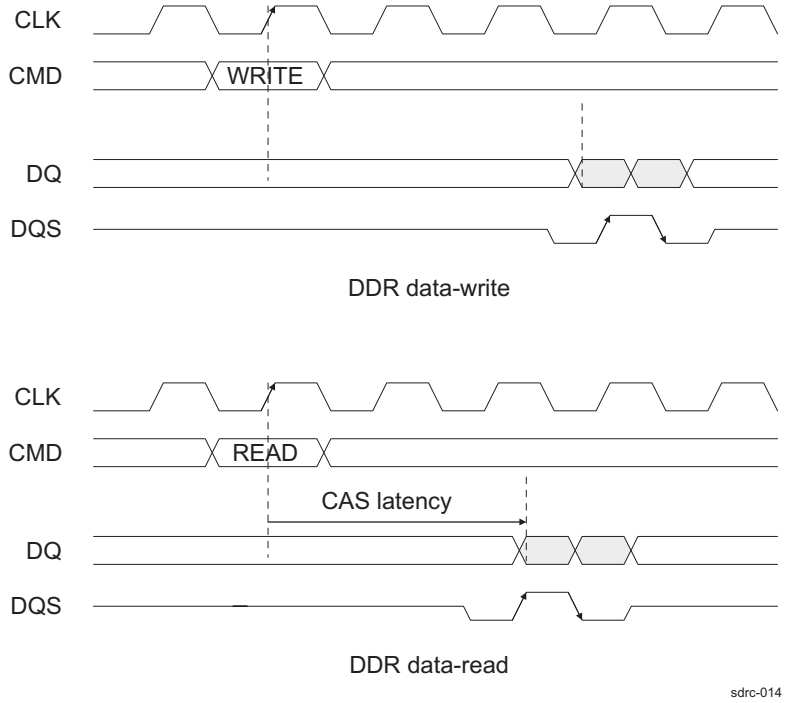
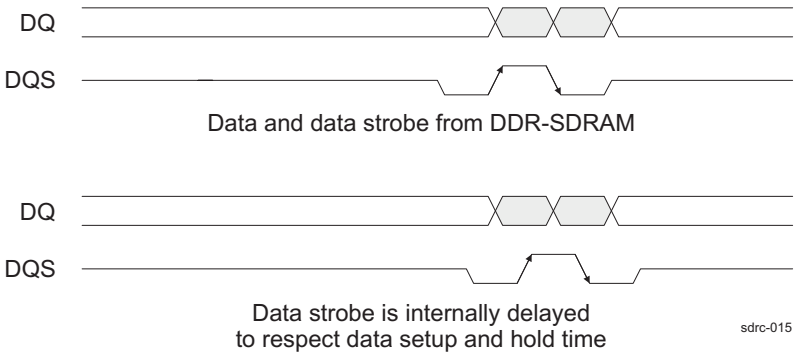


Figure 11-58 shows the DDR SDRAM data and data strobe DQS signals exiting synchronously and in phase during a data read. DQS signals are used to sample incoming data internally and, hence, must be delayed to create data-setup and data-hold time at the synchronization flip-flop inputs, as shown in the bottom waveforms of Figure 11-58. This is the goal of the DLL/CDL module.

Figure 11-58. Required Synchronization DFF Input Signals



11.2.4.4.11.2 DLL/CDL Module Architecture

Figure 11-59 shows how the DLL/CDL interacts with the synchronization flip-flops. See Table 11-102 for more information on device pins and SDRC data-lane configurations regarding DQS.

Figure 11-59. DLL/CDL Architecture

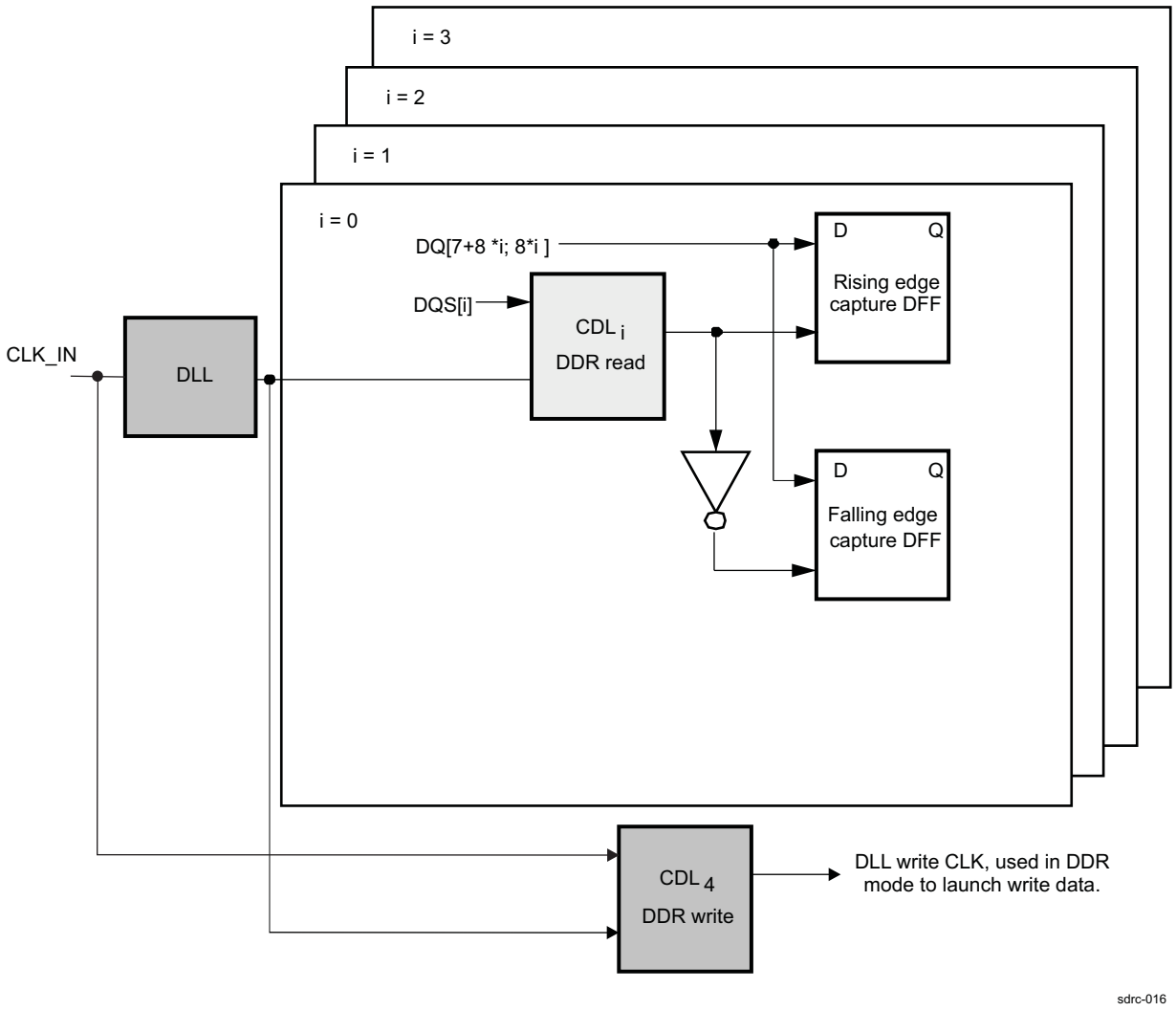
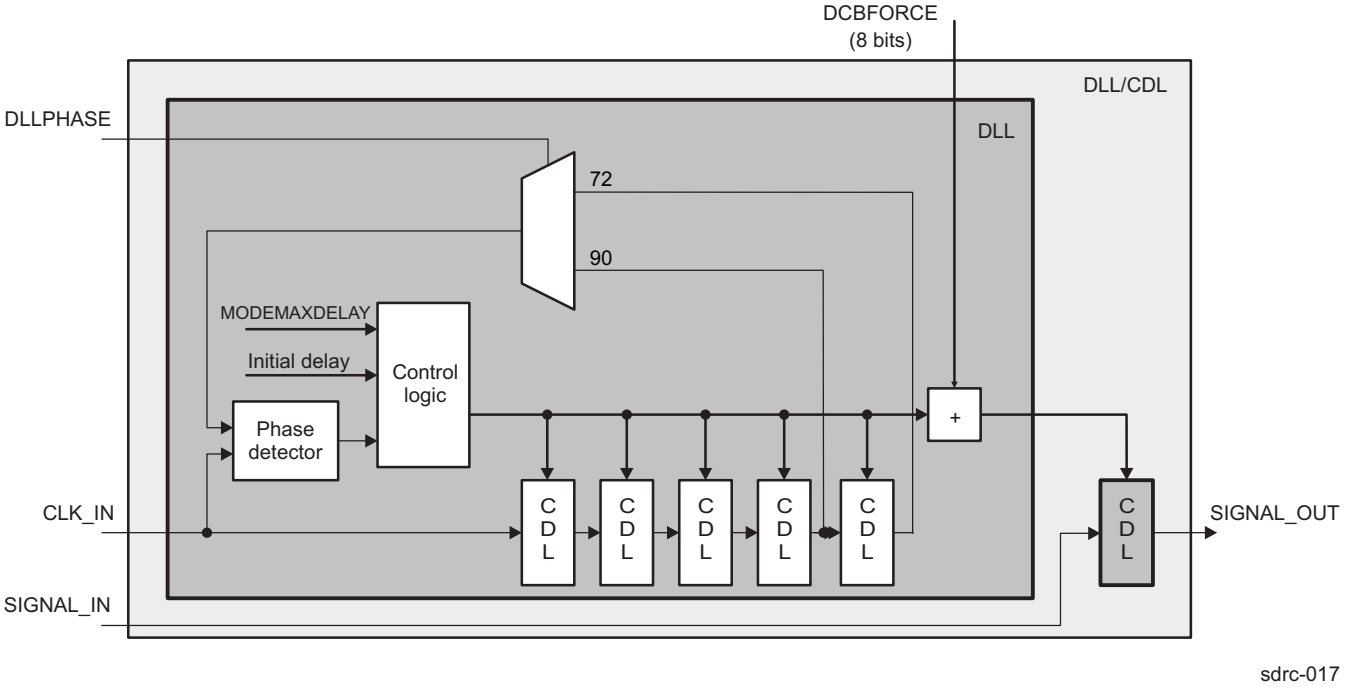


Figure 11-60 shows a simplified block diagram of the DLL/CDL module.

Figure 11-60. Simplified DLL/CDL Block Diagram



The DLL circuit contains five delay elements in series. Therefore, the DLL output code and regulator output voltage determine a delay equivalent to one fifth of the reference input period in a stand-alone DLL/CDL (those CDLs are delay elements integrated into the DLL and are not shown in Figure 11-59). In other words, the DLL/CDL delay is equivalent to either 72 or 90 degrees, that is, to a quarter or a fifth of a DDR clock period.

11.2.4.5 Mode Registers

11.2.4.5.1 Mode Register (MR)

This register is common to all SDR and DDR SDRAMs. It is a 12-bit register and controls the following parameters:

- Write burst mode (SDRC.SDRC_MR_p[9] WBST bit (where p = 0 or 1 for SDRC CS0 or CS1)
- CAS latency (SDRC.SDRC_MR_p[6:4] CASL field)
- Serial/interleaved mode (SDRC.SDRC_MR_p[3] SIL bit)
- Burst length (SDRC.SDRC_MR_p[2:0] BL field)

MR is accessible through SDRC.SDRC_MR_p (where p = 0 or 1 for SDRC CS0 or CS1). Writing to SDRC.SDRC_MR_p initiates an implicit load mode register command qualified by BA1, BA0 = 0, 0.

11.2.4.5.2 Extended Mode Register 2 (EMR2)

This register is specific to low-power SDR and mobile DDR SDRAM devices. It is a 12-bit register and controls the following parameters:

- Temperature-compensated self-refresh (SDRC.SDRC_EMR2_p[4:3] TCSR field)
- Partial array self-refresh (SDRC.SDRC_EMR2_p[2:0] PASR field)

EMR2 is accessible through SDRC.SDRC_EMR2_p (where p stands for CS0 or CS1). Writing to SDRC.SDRC_EMR2_p initiates an implicit load mode register command qualified by BA1, BA0 = 1, 0.

Note: PASR Setting Before Entering Self-Refresh Mode:

Partial array self-refresh (PASR) allows memory accesses to all banks when the attached memory is not in self-refresh. Software must ensure that after a self-refresh mode with PASR field enabled on banks, only accesses to the refreshed parts of the memory are performed. Failure to do so may result in fetching corrupted data.

11.2.5 SDRC Subsystem Basic Programming Model

This section contains programming guides on setting up the SMS and SDRC registers according to the required application.

11.2.5.1 SMS Basic Programming Model

11.2.5.1.1 SMS Firewall Usage

Because region 0 always has level 0 priority, it can be masked by any protected region.

To configure a region:

1. Set the SMS.SMS_RG_ATTi register (where i is the region index from 0 to 7).
2. Set the SMS.SMS_RG_RDPERMi register.
3. Set the SMS.SMS_RG_WRPERMi register.
4. Set the SMS.SMS_RG_STARTj register (except for region 0. j is the region index from 1 to 7).
5. Set the SMS.SMS_RG_ENDj register (except for region 0. j is the region index from 1 to 7).

Region 0 is always active. There are no start and an end address for region 0. As soon as the SMS.SMS_RG_STARTj[30:16] STARTADDRESS and SMS.SMS_RG_ENDj[30:16] ENDADDRESS registers (where j = 1 to 7) are programmed, the firewall is activated. To prevent unexpected violations, ensure that the protection regions do not overlap.

Region 1 ensures the proper dynamic programming of protection for regions 2 through 7. For example, the protected region A is set and currently accessed, but it must be enlarged. To avoid deactivating region A and exposing its content to unwanted leakage, region 1 can be used to mask this whole area during reprogramming. When region A is correctly reprogrammed, region 1 can be deactivated.

When all the required regions are programmed, the locking mechanism allows freezing the configuration, thus ensuring no further reprogramming.

11.2.5.1.2 VRFB Context Configuration

Using the RE requires several initialization programming steps. After an RE context is set up, an application can use it transparently, as if addressing a frame buffer object with a standard raster-based memory arrangement.

1. Define the page configuration. This operation usually depends only on the external memory device. The same page settings are then applied to any newly created rotated frame buffer.

   Consider an SDRAM device with 1024-byte pages. The page can be defined as a 16 ×64 array. The page is not necessarily a square (32 ×32 is also a suitable value). It is recommended that the longest side corresponds to the access direction requiring the maximum bandwidth.

   In terms of register settings, in any context that uses that page size:

   Page (page) width = 2^{pw} bytes (that is, pw = 6)

   Page (page) height = 2^{ph} rows (that is, ph = 4)
2. The application must allocate the appropriate amount of memory in the SDRAM address space, as required by the size of the frame buffer object.
   Example: Create a 400 ×300 frame buffer, 16 bits per pixel
   - Pixel size = 2^{ps} bytes (that is ps = 1)
   - Number of pages per line: 400 ×2/64 = 12.5, rounded up to 13
   - Number of pages per column: 300/16 = 18.75, rounded up to 19

In terms of register settings, the image size parameters correspond to the enlarged image, and are programmed to:

- Image width = w pixels (that is, w = 13 ×64 /2 = 416)
- Image height = h pixels (that is, h = 19 ×16 = 304)

Note: In this example, the values obtained are not integer values, but are rounded up to the closest integer value. This results in a loss of physical memory, generally negligible compared to the total size of the image.

In terms of memory allocation (in the physical memory), this corresponds to a 416 ×304 ×2 = 252,928-byte buffer.

The physical base address of this buffer must be aligned on a page boundary (in that example, a 1024-byte boundary; that is, the 10 LSBs of the base address must be all zeros). This buffer must be allocated as a contiguous memory segment.

All these parameters, once determined, must be loaded in the registers of the chosen context (there are 12 VRFB contexts with 12 independent sets of registers).

Example, context 1:

- Configure the physical base address of the frame buffer. Example: 512M bytes, start of CS0, SMS.SMS_ROT_PHYSICAL_BAn[30:0] PHYSICALBA = 0x20000000 (where n = 1)
- Configure the image height and width:

  Image width (416): SMS.SMS_ROT_SIZEn[10:0] IMAGEWIDTH = 0x1A0 (where n = 1)

  Image height (304): SMS.SMS_ROT_SIZEn[26:16] IMAGEHEIGHT = 0x130
- Configure the control parameters:

  Pixel size (example: 2 bytes, 2^1 bytes): SMS.SMS_ROT_CONTROLn[1:0] PS = 0x1 (where n = 1)

  Page (page) size (example: 1024 bytes = 64 bytes ×16 bytes)

  Page height (example: 16 rows, 2^4 rows): SMS.SMS_ROT_CONTROLn[10:8] PH = 0x4

  Page width (example: 64 bytes, 2^6 bytes): SMS.SMS_ROT_CONTROLn[6:4] PW = 0x6

3. The frame buffer just created is now ready for use by the different system initiators (MPU, sDMA, LCD controller, etc.).

   From the perspective of these modules, the frame buffer object can be accessed at the following VRFB address-space memory locations:

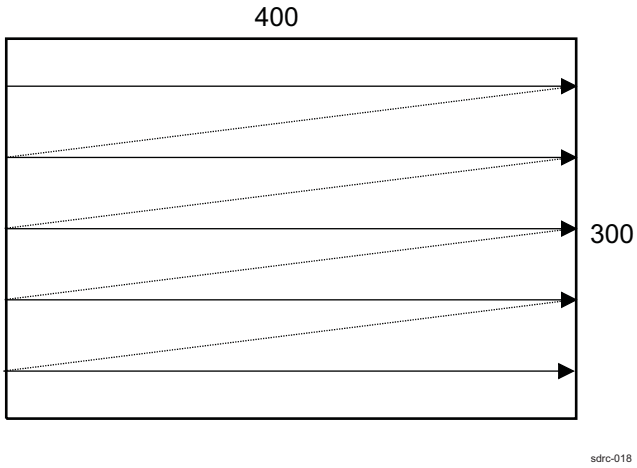
   - Context 1 0-degree view: 0x7400 0000
   - Context 1 90-degree view: 0x7500 0000
   - Context 1 180-degree view: 0x7600 0000
   - Context 1 270-degree view: 0x7700 0000

   The start address of the view corresponds to the logical origin of the image (x = 0, y = 0). The image pitch parameter, commonly defined as the distance between two vertically adjacent pixels, is fixed to 2048 pixels.

   Example of usage by the application:

   In a system using an 400 ×300 LCD panel that has a native landscape orientation, the natural scan order is as shown in Figure 11-61.

Figure 11-61. Natural Scan Order



The display buffer is the one created in the example sequence.

When the application is running and uses the portrait orientation for the display (typically, a PDA-type application):

- The LCD controller accesses the frame buffer using the 0-degree view.
- The processor and other initiators, such as 2D DMA or 3D accelerators, use the 90-degree view.

When the application is running and uses the landscape orientation for the display (typically, a video recorder/player or gaming application):

- The LCD controller still accesses the frame buffer using the 0-degree view.
- The processor and other initiators, such as 2D DMA or 3D accelerators, also use the 0-degree view. See Section 11.2.6.1.1.

11.2.5.1.3 Memory-Access Scheduler Configuration

The memory-access scheduler is configured as follows:

- For each of the three classes, the arbitration parameters are:
  - SMS.SMS_CLASS_ARBITER0 through SMS.SMS_CLASS_ARBITER2
    - One high-priority FIFO queue in the class (HIGHPRIOVECTOR field)
    - Number of consecutive transactions to perform (EXTENDEDGRANT field)
    - Burst transaction submitted for arbitration immediately or after the burst has been buffered (BURST-COMPLETE field)

11.2.5.1.4 Error Logging

All data transfers in the SMS are full handshake. The SMS uses this capability to signal the system when a transaction error is detected.

The SMS captures the address of the faulty access in the SMS.SMS_ERR_ADDR register. The error type is logged in the SMS.SMS_ERR_TYPE register. Once a faulty access is logged and the SMS.SMS_ERR_TYPE[0] ERRORVALID bit is set, the next faulty accesses cannot be logged before clearing the ERRORVALID bit.

In the case of an interconnect transaction, an error response is generated if any of the following occur:

- An incoming request arrives after an idle request from the PRCM.
- An illegal command is received.
- A protection region overlap is detected.
- Protection errors.

It is assumed that the system interconnect on which the SMS is plugged is responsible for signaling the error event to the host MPU based on the interconnect response. The MPU error handler can then consult the error logging registers.

11.2.5.2 SDRC Configuration

11.2.5.2.1 IP Revision

The IP revision code can be read in the SDRC.SDRC_REVISION[7:0] REV field.

11.2.5.2.2 Reset Behavior

The reset behavior of the SDRC can be classified into three subgroups:

- Asynchronous cold-reset (power-on reset) behavior
- Asynchronous warm reset behavior
- Synchronous soft-reset behavior

When the system-wide power-on reset is applied through cold reset, all flops are reset to their default values, and all state-machines are returned to their idle states.

The programming model for data recovery following a warm reset is as follows:

- Program the SDRC.SDRC_POWER_REG register to enable the SDRC.SDRC_POWER_REG[7] SRFRONRESET bit.

A warm reset condition is then issued.

- The SDRC enters self-refresh mode since the SRFRONRESET bit is set.
- The SDRC does not execute global SDRC reset since the reset is not qualified as cold.
- The SDRC state-machine maintains the external memory device in self-refresh.

The first SDRC access to the configuration register must then be:

1. Check the SDRC configuration.
2. Exit self-refresh mode using the manual command register.

A software-controlled reset is also available by using the SDRC.SDRC_SYSCONFIG[1] SOFTRESET bit (set this bit to 1 to activate the reset). The completion of the reset can be determined by reading the SDRC.SDRC_SYSSTATUS[0] RESETDONE bit.

When the SDRC is reset due to the presence of either a soft or cold reset, all SDRC flops are reset.

Note: SDRC Requirement at First Power-Up to Have sdrc_cke Pin High

To comply with the JEDEC standard, sdrc_cke pins values are forced to 1 during the initial memory power-up phase: software must ensure that sdrc_cke pin is released after the initialization phase; it happens only at first power-up (on a cold reset). Thus, at the end of the initial SDRC power-up sequence and before programing the PWDENA field, software must ensure that the sdrc_cke pin is driven by the SDRC module. Then the value of the PWDENA field can be modified. See Section 11.2.5.4.1 for more details on sdrc_cke driving.

11.2.5.3 SDRC Setup

A number of device parameters must be set before executing the initialization sequence.

11.2.5.3.1 Chip-Select Configuration

CS0 always starts at 0x8000 0000 (with respect to the 32-bit interconnect address). There is no restriction on the presence of any SDRAM on CS0 or CS1.

The total address space of the SDRC is 1G byte/8G bits. The total address space is divided into 8 ×128M-byte partitions as shown in Figure 11-51. Each partition is a possible start address for CS1, except for the partition occupied by CS0.

The start address for CS1 is defined by the SDRC.SDRC_CS_CFG[9:8] CS1STARTLOW and SDRC.SDRC_CS_CFG[3:0] CS1STARTHIGH fields.

Space 0, selected by the SDRC using the CS0 (nCS0) output pin, is always at offset 0 from the SDRAM memory space base address. The size of this area is programmable through the SDRC.SDRC_MCFG_p[17:8] RAMSIZE field, where p = 0. Space 1, selected by the SDRC using the CS1 (nCS1) output pin, is at an offset from the SDRAM memory space base address; this offset is programmable in 128M-byte increments. The size of this area is programmable through the SDRC.SDRC_MCFG_p[17:8] RAMSIZE field, where p = 1. The type of device for each area is programmable through the SDRC.SDRC_MCFG_p register (where p = 0 or 1 for SDRC CS0 or CS1).

Note: Ensure that space 0 and space 1 do not overlap each other or extend farther than the maximum 1G-byte SDRC memory space, as explained in Section 2.2, *Global Memory Space Mapping*, of Chapter 2, *Memory Mapping*.

11.2.5.3.2 Memory Configuration

The memory configuration is defined on a per-CS basis through the SDRC.SDRC_MCFG_p register (where p = 0 or 1 for SDRC CS0 or CS1).

Table 11-104 lists the memory configuration.

Table 11-104. Memory Configuration

Bit Field	Comments
ADDRMUXLEGACY	Address multiplexing scheme
RAMSIZE	Defines the physical RAM address space in terms of 2M -byte chunks
B32NOT16	External device data bus width.
DEEPPD	Set this bit if the memory supports deep-power-down mode. (This bit is only a flag for software. It does not affect any SDRC function.)
DDRTYPE	Mobile DDR
RAMTYPE	Single Data Rate or Double Data Rate SDRAM

Note: Exported Register Reset Values and Lock Bit

The reset values of SDRC.SDRC_MCFG_p and SDRC.SDRC_SHARING are exported in the control module. At reset, these registers take the value previously stored in the control module. A new bit is added to each register to provide the capability to lock these three registers into read-only accesses:

- SDRC.SDRC_MCFG_p[30] LOCKSTATUS bit (p = 0 or 1 for CS0 or CS1)
- SDRC.SDRC_SHARING[30] LOCK bit

  The reset value of each lock bit is also imported from the control module.

11.2.5.3.3 SDRAM AC Timing Parameters

The AC parameters described in Table 11-105 can be independently programmed (standard JEDEC LPDDR1 terminology is used here) in clock cycles for each of the two memory areas through registers SDRC.SDRC_ACTIM_CTRLA_p and SDRC.SDRC_ACTIM_CTRLB_p (p = 0 or 1, depending on the CS area).

Table 11-105. Programmable AC Parameters

SDRC AC Parameter	Description	Range (Clock Ticks)
tRFC	AUTO REFRESH to ACTIVE / AUTO REFRESH command period (autoReFresh Cycle time)	0 - 31
tRC	ACTIVE to ACTIVE command period (Row Cycle time)	0 - 31
tRAS	ACTIVE to PRECHARGE command period	0 - 15
tRP	PRECHARGE command period (Row Precharge time)	0 - 7
tRCD	ACTIVE to READ or WRITE delay (Row-to-Column Delay time)	0 - 7
tRRD	ACTIVE bank A to ACTIVE bank B delay	0 - 7
tWR	WRITE Recovery time (also known as tDPL)	0 - 7
tDAL	Auto precharge write recovery + precharge time	0 - 31
tWTR	Internal Write to Read command delay (also known as tCDLR)	1 - 3
tCKE	CKE min pulse width (high and low pulse width)	1 - 7
tXP	Exit Power-Down to next valid command delay	1 - 7
tXSR	Self-Refresh exit to next valid command delay	0 - 255

Example of configuration:

If there is a minimum tRC requirement of 88 ns at 100 MHz, 88/10 = 9 (8.8 is rounded up).

Therefore, nine clock cycles must be set.

The SDRC AC parameters described in Table 11-106 are hard-coded.

Table 11-106. Nonprogrammable AC Parameters

SDRC AC Parameter	Description	Range (Clock Ticks)	Comment
tMRD	MODE REGISTER SET command period	3	
tDQSS	Write command to first DQS latching transition	1 (0.75 - 1.25 CK)	
tRPRE	Read preamble	1 (0.9 - 1.1 CK)	DQS is held low for read when driving on the same cycle as the read command is stopped.

Note: The SDRC uses tXSR only when exiting self-refresh mode, regardless of the first command issued. The SDRC systematically inserts an autorefresh command before it serves the first request.

11.2.5.3.4 DLL/CDL Configuration

The DLL unit is configured by writing to the SDRC.SDRC_DLLA_CTRL register. This register contains the following fields:

- FIXEDDELAY
- MODEFIXEDDELAYINITLAT
- WRITEDDRCLKX2DIS
- DLLMODEONIDLEREQ
- DLLIDLE
- ENADLL
- LOCKDLL
- DLLPHASE

To support low-frequency DDR access, set the DLL in fixed delay mode by setting the SDRC_DLLA_CTRL [2] LOCKDLL bit to 0x1. In this mode, the SDRC_DLLA_CTRL [31:24] FIXEDELAY field allows the programming of the CDL delay. This provides the correct fixed DCB code based on the input frequency. Only the voltage precharge part of the DLL is still used to control the CDL instances in this mode, but the control loop is broken and the voltage control is inactive.

When fixed delay mode is enabled, a counter (programmable through the SDRC_DLLA_CTRL [23:16]MODEFIXEDDELAYINITLAT field) based on the input frequency allows the SDRC to stall all incoming requests. Once this counter expires, the SDRC starts accessing the memory. The DLL characterization shows that this feature is not useful and that this value shall be set to 0x0, leading actually to a null delay.

WRITEDDRCLKX2DIS controls which clock is used for the write data path. The WRITEDDRCLKX2DIS bit ensures the proper MDDR write data bytes by using a double frequency input clock coming from the PRCM module.

The DLL can be put in idle mode using the SDRC_DLLA_CTRL[4] DLLIDLE bit. When in idle mode, the DLL lock is lost. The precharge voltage is kept stable to enable faster relock when going back to a functional state. If set, this bit overrides the ENADLL bit. Thus, avoid completely powering down the DLL by keeping the precharge voltage and cutting off the analog loop. Exiting from this idle mode saves some clock cycles compared to exiting from power-down mode. Refer to the note (DLL behavior upon a warm reset assertion) in Section 11.2.4.4.9.2, *Dynamic Low-Power Operating Modes*). Furthermore, the SDRC_DLLA_CTRL[6:5] DLLMODEONIDLEREQ field defines the modes (power-down/DLLIDLE/No action) of the DLL that are automatically entered upon an Idle_req assertion. The DLL mode (TrackedDelay or ModeFixedDelay mode) is updated only when Idle_req/Idle_ack handshake protocol occurs, warm reset occurs, PWRDN is enabled, or DLLIDLE mode enabled.
Power-down mode is asserted upon warm reset assertion. If power-down and DLLIDLE are asserted simultaneously, power-down overwrites DLLIDLE mode inside the DLLCDL cell.

To modify the SDRC input clock when the DLL is locked and active, the DLL must first enter either its idle mode or its power-down mode. The SDRC input frequency can be changed by using any of the following scenarios:

- Internal signals handshaking protocol with PRCM module
- Warm reset event
- DLLIDLE mode
- Power-down mode

The ENADLL bit controls the PWRDN mode of the DLL module.

The LOCKDLL bit sets the DLL in TrackedDelay (lock) or ModeFixedDelay (unlock) mode. ModeFixedDelay mode is supported up to 83 MHz.

The DLLPHASE control bit is used to set up the nominal delay tracked by the DLL. See Section 11.2.4.4.11, *Controlled Delay Line*, for more information on the CDL/DLL module.

DLLMODEONIDLEREQ is of no importance to the Idle_ack generation. In case DLLMODEONIDLEREQ = 2 and Idle_req occurs, Idle_ack can be generated and SDRC/core can go to idle; however, recovery from this state may be incorrectly performed and the DLL cannot relock. If this happens, the only solution to recover is to disable and then re-enable the DLL. DLLMODEONIDLEREQ = 2 is not a valid configuration under Idle_req assertion and must not be used. DLLMODEONIDLEREQ can be set 0 or 1, depending on whether the priority is shorter wake-up latency or lower power consumption.

11.2.5.3.5 Mode Register Programming and Modes of Operation

The SDRC contains a group of registers known as the memory mode registers. These registers define the operational modes of the target SDRAM.

- MR: Mode register used to define operational parameters common to SDR and DDR SDRAMs.
- EMR1: Extended mode used to define operational parameters exclusive to DDR SDRAM (irrelevant to the SDRC since regular DDRs are not supported).
- EMR2: Extended mode used to define operational parameters exclusive to mobile SDRAM.

11.2.5.3.5.1 Mode Register (MR)

The 12-bit SDRC.SDRC_MR_p register (p = 0 or 1 for CS0 or CS1) is common to all SDR and DDR SDRAMs and controls the following parameters:

- Write burst mode
- CAS latency
- Serial/interleaved mode
- Burst length

When programming the SDRC.SDRC_MR_p bit fields (where p = 0 or 1 for SDRC CS0 or CS1), consider the following:

- CAS latencies of 1, 2, 3, 4, and 5 are supported (CASL).
- Only serial mode (not interleaved mode) is supported (SIL = 0x0).
- A burst length of 2 is supported for SDR SDRAM (BL = 0x2).
- A burst length of 4 is supported for DDR SDRAM (BL = 0x4).
- Burst lengths of 1 (BL = 0x0), 8 (BL =0x3), and full page (BL = 0x7) are not supported.

Writing to SDRC.SDRC_MR_p initiates an implicit Load Mode register command qualified by BA1,BA0 = 0,0 except if the NOMEMORYMRS bit is set.

11.2.5.3.5.2 Extended Mode Register 2 (EMR2)

The SDRC.SDRC_EMR2_p register (p = 0 or 1 for CS0 or CS1) is specific to mobile SDRAM devices. It is a 12-bit register that controls the following standard parameters:

- Partial Array Self-Refresh (PASR)
- Temperature Compensated Self-Refresh (TCSR)
- Driver Strength (DS)

The SDRC.SDRC_EMR2_p[2:0] PASR field programs the partial array self-refresh feature. The low power SDR SDRAM granularity is much finer than the granularity available in a mobile DDR SDRAM. The SDRC.SDRC_EMR2_p[4:3] TCSR field programs the temperature compensated self-refresh feature. The TCSR granularity available in a low power SDR is much finer than the granularity available in a mobile DDR.

Writing to SDRC.SDRC_EMR2_p initiates an implicit load mode register command qualified by BA1, BA0 = 1,0 except if SDRC_SYSCONFIG[8] NOMEMORYMRS is set.

11.2.5.3.6 Autorefresh Management

The SDRAM refresh configuration register group controls refresh management in normal operation. This group contains two SDRC.SDRC_RFR_CTRL_p registers that are defined on a per-CS basis and contain the following bit fields:

- SDRC.SDRC_RFR_CTRL_p[1:0] ARE (where p = SDRC CS value 0 or 1)
- SDRC.SDRC_RFR_CTRL_p[23:8] ARCV (where p = SDRC CS value 0 or 1)

These bit fields can enable and disable autorefresh. Autorefresh bursts of 1, 4, and 8 are programmed using these fields. The autorefresh burst starts when the 16-bit autorefresh counter decrements to 0. The ARCV field loads the autorefresh counter with a 16-bit autorefresh value. The ARCV value is calculated using the following formula:

Refresh value = (refresh interval / clock period / number of rows) - margin

Note: Memory refresh interval in time unit. Margin is 50 (cycles).

The margin considers the possibility of an ongoing access when the counter expires, thus delaying the effective refresh sequence.

The value to be programmed is independent of the burst-refresh configuration: if a burst-refresh is configured, the value is automatically scaled in hardware to the burst-refresh size.

Autorefresh is enabled by programming the SDRC.SDRC_MANUAL_p[3:0] CMDCODE field to 0x2 (where p = 0 or 1 for SDRC CS0 or CS1).

11.2.5.3.7 Page Closure Strategy

The page closure strategy is defined on a per-bank basis by setting the SDRC.SDRC_POWER_REG[0] PAGEPOLICY bit. SDRC defines one type of page closure strategy:

1. High power/high bandwidth: Bandwidth consumption is critical.

The SDRC tracks open pages. The SDRC determines whether the current access is an open page or a closed page. The SDRC does the following:

1. If the current page is already open on this bank the SDRC automatically issues a *precharge* command to close that bank.
2. Opens the accessed page
3. Executes the access

In the worst case, four pages (one page per bank) may be opened simultaneously.

The PAGEPOLICY bit must be set to 1 to enable this mode.

11.2.5.4 Manual Software Commands

The manual commands register SDRC.SDRC_MANUAL_p (where p = 0 or 1 for SDRC CS0 or CS1) is used to implement software-driven commands:

- NOP command
- Precharge All command
- Autorefresh command
- Enter deep-power-down command
- Exit deep-power-down command
- Enter self-refresh command
- Exit self-refresh command
- Set CKE high command
- Set CKE low command

  These commands are described in the following section:
- NOP (CMDCODE: 0x0)

  When the SDRC.SDRC_MANUAL_p[3:0] CMDCODE field is programmed with 0x0, the SDRC issues a NOP/inhibit command. The following table lists the status of the SDRC memory port signals.

  NOP does not initiate any new operation, but it is needed to complete operations that require more than a single clock cycle-like autorefresh.

  Inhibit is also a NOP. nCS high disables the command decoder so that nRAS, nCAS, nWE, and all the address inputs are ignored.

Command	nCS	nRAS	nCAS	nWE
Inhibit	H	X	X	X
NOP	0	H	H	H

- Precharge all (CMDCODE: 0x1)

  When the SDRC.SDRC_MANUAL_p[3:0] CMDCODE field is programmed with 0x1, the SDRC issues a precharge all command. The following table lists the status of the SDRC memory port signals.

  During the command, address A10 remains high, and bank information (BA0 and BA1) is don't care.

  All banks can be precharged at the same time by using the precharge all command.

  At the end of tRP, after performing precharge on all of the banks, they enter the idle state.

Command	A10	nCS	nRAS	nCAS	nWE
Precharge all	H	L	L	H	L

- Autorefresh (CMDCODE: 0x2)

  When the SDRC.SDRC_MANUAL_p[3:0] CMDCODE field is programmed with 0x2, the SDRC issues an autorefresh command. The following table lists the status of the SDRC memory port signals.

  In addition to the signal status mentioned below, the CKE signal is at logic high for autorefresh.

  The autorefresh command can only be asserted when all banks are in idle state and the device is not in power-down mode (CKE is high in the previous cycle).

  The autorefresh command must be followed by NOPs until the autorefresh operation completes.

  All banks are in the idle state at the end of the autorefresh operation.

Command	nCS	nRAS	nCAS	nWE
Autorefresh	L	L	L	H

- Enter deep-power-down (CMDCODE: 0x3)

  When the SDRC.SDRC_MANUAL_p[3:0] CMDCODE field is programmed with 0x3, the SDRC executes an enter DPDM command. This command is used for low-power devices (that is, LPSDR or MDDR devices that support deep-power-down mode). The following table shows the status of the SDRC memory port signals.

  The device enters deep-power-down mode by having nCS and nWE held at logic low with nRAS and nCAS high at the rising edge of the clock, while CKE is low.

Command	nCS	nRAS	nCAS	nWE	CKE
Enter DPDM	L	H	H	L	L

- Exit deep-power-down command (CMDCODE: 0x4)

  When the SDRC.SDRC_MANUAL_p[3:0] CMDCODE field is programmed with 0x4, the SDRC executes an exit deep-power-down command. The device exits deep-power-down mode when the inhibit command is sampled with CKE held at logic high.

- Enter self-refresh (CMDCODE: 0x5)

  When the SDRC.SDRC_MANUAL_p[3:0] CMDCODE field is programmed with 0x5, the SDRC puts the memory into self-refresh. The signal status is the same as defined in autorefresh, and CKE is held at logic low during the self-refresh entry command.

  The self-refresh mode is entered from the all-banks-idle state by asserting nCS, nRAS, nCAS, and CKE low, and nWE high.

  Once the self-refresh mode is entered, only the CKE state being low matters; all other inputs, including the clock, are ignored and remain in self-refresh mode.

- Exit self-refresh (CMDCODE: 0x6)

  When the SDRC.SDRC_MANUAL_p[3:0] CMDCODE field is programmed with 0x6, the SDRC executes the self-refresh exit command and, after meeting the tXSR timing parameter, executes one autorefresh command to adhere to the memory protocol. It is an exit self-refresh command when CKE is detected high with a NOP command.

  Self-refresh is exited by restarting the external clock and then asserting CKE high. This must be followed by NOPs for a minimum time of tXSR before the SDRAM reaches idle state to begin normal operation.

- Set the CKE signal high (CMDCODE: 0x7)

When the SDRC.SDRC_MANUAL_p[3:0] CMDCODE field is programmed with 0x7, CKE is set to high. An example of where this command is used is during DDR memory initialization.

- Set the CKE signal low (CMDCODE: 0x8)

  When the SDRC.SDRC_MANUAL_p[3:0] CMDCODE field is programmed with 0x8, CKE is set to low. An example of where this command can be used is to put memory in power-down mode.

11.2.5.4.1 Low-Power SDR/Mobile DDR Initialization Sequence

The initialization sequence is executed after the following occur:

1. Power is applied.
2. The clock is stable.
3. The power-on reset sequence is executed.

The initialization sequence is executed by programming the following manual command registers, in the SDRC.SDRC_MANUAL_p[3:0] CMDCODE bit field, on a per-CS basis:

1. Set CMDCODE to 0x0 (NOP command), for a minimum delay of 200 µs. Another way to stabilize the connected device internal circuits is to deactivate the corresponding CS for the same amount of time.

CAUTION

Programmers must ensure that the next command (precharge-all command) is sent after a minimum delay of 200 µs. Other timings, such as tRP timing between the precharge-all command and autorefresh command, are handled automatically by the SDRC controller depending on the timing values programmed in the appropriate registers.

2. Set CMDCODE to 0x1 (precharge all command) to precharge all banks.
3. Set CMDCODE to 0x2 (autorefresh command). The autorefresh command is automatically generated after a time of tRP, hence SDRC_ACTIM_CTRLA_p[17:15] TRP (where p = 0 or 1, for CS0 or CS1) must be programmed as stated in the external memory datasheet.
4. A second autorefresh command must be programmed. Set CMDCODE to 0x2 another time.
5. Then configure the mode register SDRC_MR_p (p = 0 or 1) with BA0 and BA1 set to 0.

Because the mode register powers up in an unknown state, it must be loaded before applying any operational command.

When MR is reprogrammed, a suitable time must elapse before an active command is issued. The hardware ensures this by fixing tMRD to two clock cycles. The SDRC can handle CKE as force on CKE is released through the control module.

After VDD initialization sdrc_cke0 and sdrc_cke1 signals are forced outside the SDRC subsystem by the control module. When the external SDRAM device is correctly initialized the control module must release the force on these sdrc_cke signals by clearing the corresponding MUXMODE bit fields in CONTROL.CONTROL_PADCONF_SAD2D_SBUSFLAG[18:16] and CONTROL.CONTROL_PADCONF_SDRC_CKE1[2:0] bit fields for sdrc_cke0 and sdrc_cke1 respectively. See Section 11.2.5.2.2 for more information on reset behavior.

11.2.5.4.2 Read/Write Access

The commands required for a normal read/write access are automatically generated as a function of the following:

- The read/write command
- The address bus. A10 defines precharge all
- The relevant SDRC_MR_p / SDRC_EMR2_p registers

11.2.5.4.3 Memory Power Management

11.2.5.4.3.1 Clock Enable Management

The SDRC supports two autonomous clock enable pins CKE0 and CKE1. Each CS (CS0 or CS1) has its own CKE signal. Power management of the CS0 memory is controlled by CKE0. Power management of the CS1 memory is controlled by CKE1. This allows the SDRAM memories associated with CS0 or CS1 to be independently placed in power-down, deep-power-down (DPD) or self-refresh (SR) mode. This is achieved using the manual control register SDRC.SDRC_MANUAL_p[3:0] CMDCODE field which is defined on a per chip select basis. Entry and exit of these low-power modes is achieved using the relevant CMDCODE.

Once the memory associated with a CS has been powered down in deep-power-down mode it cannot be powered up by dynamic power management features. It can only exit the low-power state through the programming of the relevant CMDCODE. The SDRC can automatically exit other low-power modes.

11.2.5.4.3.2 Clock-Controlled Memory Power Management

The SDRC power-management register (SDRC.SDRC_POWER_REG) contains the EXTCLKDIS field, which controls suspension of the external clock on a per-CS basis. Writing 1 to this field in the relevant register freezes the clock with a latency dependent on the SDRC.SDRC_POWER_REG register programming. Subsequently, writing 0 to this field in the relevant register enables the clock with a latency dependent on the SDRC.SDRC_POWER_REG register programming.

11.2.5.4.3.3 Manual Power-Down Mode Power Management

For automatic power-down mode refer to Section 11.2.4.4.9.2. The programming model for manual power-down mode is different for an SDR and a DDR.

SDR Mechanism/Power-Down Mode Entry

1. Ensure that no accesses are pending or active.
2. Program the CMDCODE field of the relevant manual command register to 0001.
   This ensures that all banks are idle by executing a precharge all command.
3. Wait two clock cycles.
4. Program the CMDCODE field of the relevant manual command register to 0000.
   This executes a NOP command.
5. Program the CMDCODE field of the relevant manual command register to 1000.
   This sets CKE low.
6. Maintain the clock at a stable value.

SDR Mechanism/Power-Down Mode Exit

1. Provide clock.
2. Program the CMDCODE field of the relevant manual command register to 0111.
   This sets the relevant CKE high.
3. Program the CMDCODE field of the relevant manual command register to 0000.
   This executes a NOP command.
4. Program the CMDCODE field of the relevant manual command register to 0001.
   This ensures that all banks are idle by executing a precharge.

The length of time the SDRC spends in power-down mode must not exceed the refresh period; otherwise, data becomes corrupted.

For a DDR, the sequence is as follows:

DDR Mechanism/Power-Down Mode Entry

1. Ensure that no access is currently pending or active.
2. To reduce power consumption (not mandatory): disable DLL by setting the ENADLL field of the relevant SDRC.SDRC_DLLA_CTRL register to 0x0.
3. Program the CMDCODE field of the relevant manual command register to 0001.

This ensures that all banks are idle by executing a precharge all command.

4. Program the CMDCODE field of the relevant manual command register to 0000.
   This executes a NOP command.
5. Program the CMDCODE field of the relevant manual command register to 1000.
   This sets the relevant CKE low.
6. Maintain the clock at a stable value.

DDR Mechanism/Power-Down Mode Exit

1. Provide clock.
2. Program the CMDCODE field of the relevant manual command register to 0111.
   This sets the relevant CKE high.
3. Program the CMDCODE field of the relevant manual command register to 0000.
   This executes a NOP command.
4. Program the CMDCODE field of the relevant manual command register to 0001.
   This ensures that all banks are idle by executing a precharge all command.
5. Enable DLL by setting the ENADLL field of the relevant SDRC.SDRC_DLLA_CTRL register to 0x1 .

The amount of time the SDRC spends in power-down mode must not exceed the refresh period; otherwise, data becomes corrupted.

11.2.5.4.3.4 Deep-Power-Down Mode Power Management

When in deep-power-down mode the power distribution to the entire memory array is cut. The programming model for deep-power-down mode is as follows:

Deep-power-down Mode Entry

- Precharge all banks (CMDCODE: 0x1). This ensures that all banks are idle.
- Enter deep-power-down mode (CMDCODE: 0x3).

Deep-power-down Mode Exit

- Exit deep-power-down mode (CMDCODE: 0x4).

The MR and EMR values are retained upon exiting deep-power-down mode.

Note: Because power-pown entry/exit sequences depend on memory devices, see the memory specification for the complete sequence.

11.2.5.4.3.5 Manual Self-Refresh Mode Power Management

The programming model for entering and exiting self-refresh mode is as follows:

Self-Refresh Entry

- Precharge all banks (CMDCODE: 0x1).
- NOP (CMDCODE: 0x0)
- Enter self-refresh mode (CMDCODE: 0x5).

There is no need for the software to disable the autorefresh in the SDRC.SDRC_RFR_CTRL_p register before entering self-refresh. The autorefresh counter is reset in hardware so that an autorefresh cycle is automatically generated immediately after a self-refresh exit, and before any other command.

Self-Refresh Exit

- Exit self-refresh mode (CMDCODE: 0x6).
- If needed, reconfigure SDRC registers as required.
- Enable autorefresh by programming:
  - The relevant SDRC.SDRC_RFR_CTRL_p[23:8] ARCV field
  - The relevant SDRC.SDRC_RFR_CTRL_p[1:0] ARE field to the desired refresh burst

11.2.5.5 Error Management

All data transfers in the SDRC operate a system of full handshaking. A valid read or write request that is presented to the SDRC by the L3 interconnect sequencer results in the SDRC acknowledging the transfer by raising the SCmdAccept flag. Failure to do this within a defined temporal window constitutes an error. Errors can arise from the following sources:

- Any transaction while the memory is in deep-power-down mode
- An illegal initiator access. The address of the last illegal access is captured in the SDRC.SDRC_ERR_ADDR register

If an error occurs, the software error handler performs the following actions:

- Interrogates the ERRORVALID field of the SDRC.SDRC_ERR_TYPE register to verify the presence of an error
- Interrogates the ERRORDPD field of the SDRC.SDRC_ERR_TYPE register to determine whether a transaction error resulting from the device being in deep-power-down mode is present
- Interrogates the ERRORCONNID field of the SDRC.SDRC_ERR_TYPE register to determine whether a transaction error resulting from an illegal access by an interconnect initiator is present
- Interrogates the ERRORADD field of the SDRC.SDRC_ERR_TYPE register to determine whether a transaction error resulting from an illegal address is present:
  - Interconnect access to an address outside the memory space (0x0)
  - Interconnect access to an address outside the register space (0x1)
- Writes 0 to the ERRORVALID field of the SDRC.SDRC_ERR_TYPE register to clear the active error status
- Executes error recovery from software

11.2.6 SDRC Use Cases and Tips

11.2.6.1 How to Program the VRFB

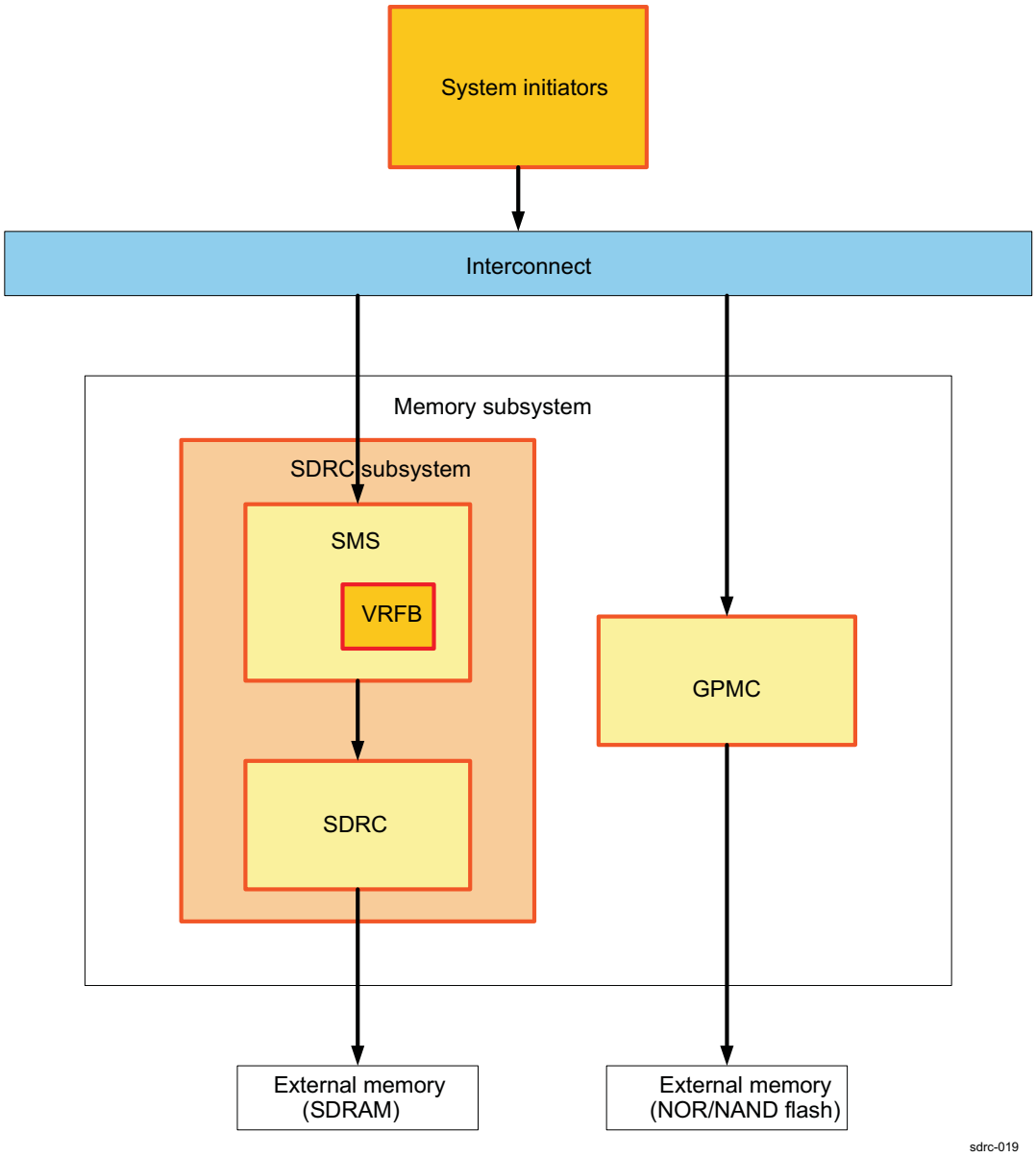
11.2.6.1.1 VRFB Rotation Mechanism

An inherent limitation of SDRAM technology is high-memory latency caused by page-miss penalties incurred when downloading to a memory cache. For example, switching from one page to another in external memory can cause a page-miss, indicating that the page accessed for the current pixel is different from that for the previous pixel.

A DMA engine is used to rotate pictures in external DRAM, but this rotation method increases the number of page misses.

The efficient way to rotate image data in external SDRAM is to use the VRFB module, which is an RE embedded in the SMS of the OMAP device, as shown in Figure 11-62. It is configured in the SMS registers.

Figure 11-62. SDRC Subsystem Overview



Accessing images stored in external SDRAM in a non-natural order requires an address transaction: the virtual address of an image is translated into a physical address in the corresponding buffer. Address translation causes an image to be rotated in 0-, 90-, 180-, or 270-degree views. With multiple views of the image, the RE can change addresses and issue multiple requests to the SDRC so that a maximum of consecutive accesses is performed, thus decreasing the number of page-miss penalties. The RE cannot reorder requests to the SDRAM.

A VRFB context defines the configuration used to access a picture in external SDRAM. For each VRFB context, a set of registers in the SMS describes:

- The size of the page (height and width)
- The picture parameters before rotation (width, height, and pixel format)
- The rotation angle (0-, 90-, 180-, or 270-degree)
- The physical base address in external memory

Any initiator in the OMAP device can access up to 12 images simultaneously, each image having an independent context. These 12 contexts is considered virtually addressed image buffers. Each context is assigned to a physical image buffer.

The virtual address space of each context is subdivided into four separate address regions pointing to the same physical image buffer but corresponding to the four possible rotations: 0/90/180/270.

After the VRFB context is configured, all data accesses to an address space are automatically translated when accessing SDRAM through the RE.

11.2.6.1.2 Setting a VRFB Context

Note: YUV Format

The YUV standard shows a color space with three elements:

- Y for luminance
- U for chrominance
- V for chrominance

As shown in Figure 11-63, pixel format (not pixel size) used in the YUV standard is spread onto a 32-bit data structure:

- This 32-bit data structure represents a packet of two pixels: P0 and P1.
- Each element (Y0, Y1, U, and V) is coded in 8 bits.
- P0 = Y0; U, V, and P1 = Y1, U, V.

There are 24 bits of information per pixel (Y, U, and V); however, because the chrominance elements U and V are common to both P0 and P1 pixels, only 16 bits are used to store a pixel in YUV format. The YUV standard uses a 32-bit data structure to represent 2 pixels; the pixel format uses 32 bits (4 bytes).

Thus, when defining YUV image parameters, the image width must be set to one-half the number of pixels per row and the pixel format must be set to 4 bytes, because the YUV pixel data is spread onto a 32-bit word representing 2 pixels.

Figure 11-63 shows the pixel representation of the YUV format.

Figure 11-63. YUV Format: Pixel Representation

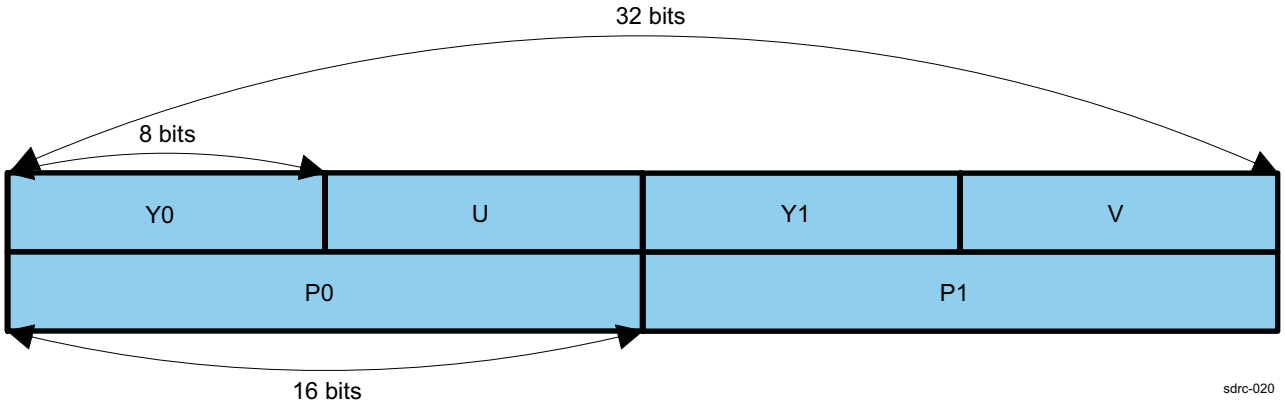


Figure 11-64 shows a generic way to configure a VRFB context to perform a rotation view on pixel data in external DRAM.

Figure 11-64. VRFB Context Configuration

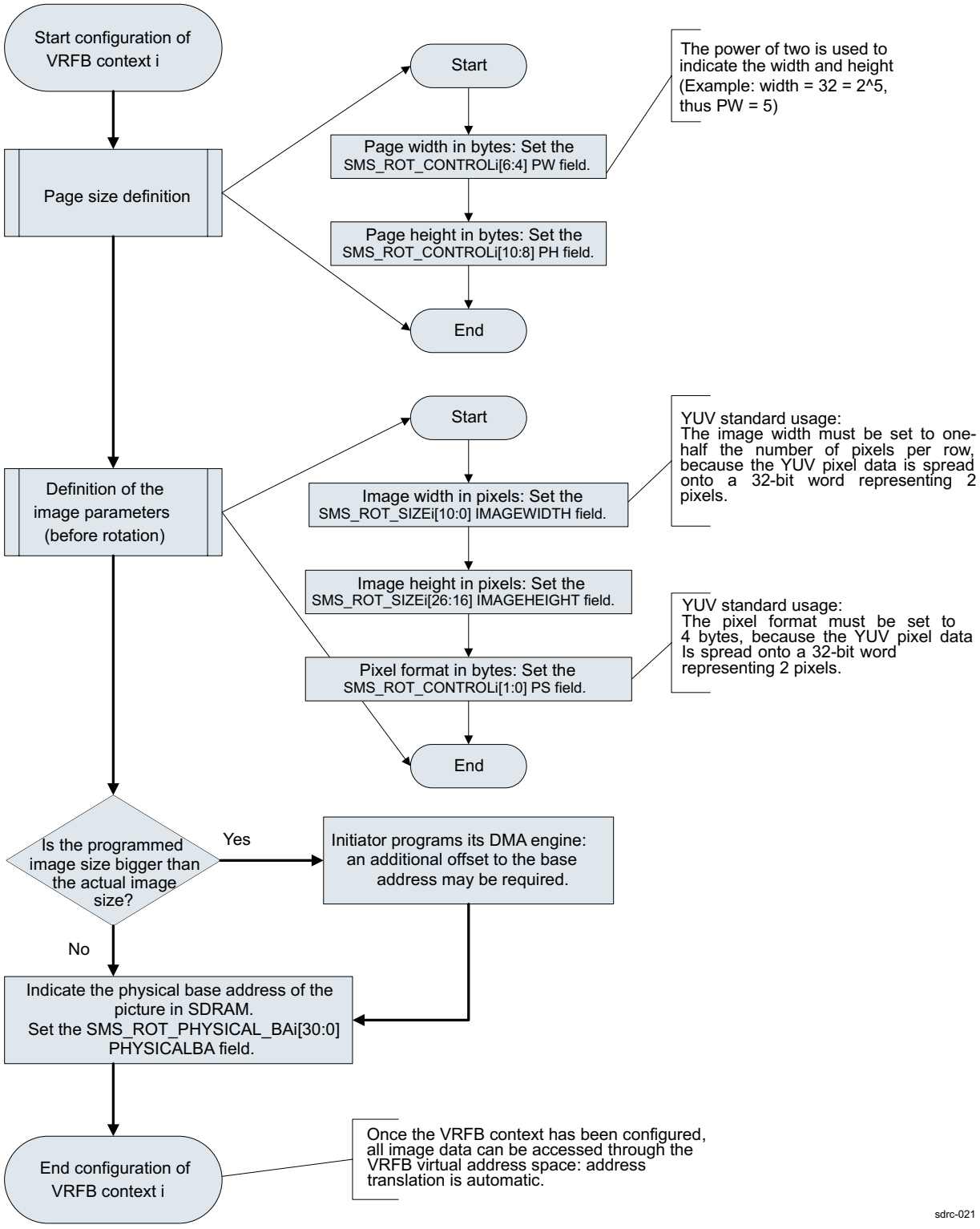


Figure 11-65 is an example of VRFB context configuration using both RGB and YUV image formats. This example represents the configuration of a 1024-byte page size (32 ×32 byte page) and a 240 ×240 image size using VRFB context 1.

Figure 11-65. Example of VRFB Context 1 Configuration

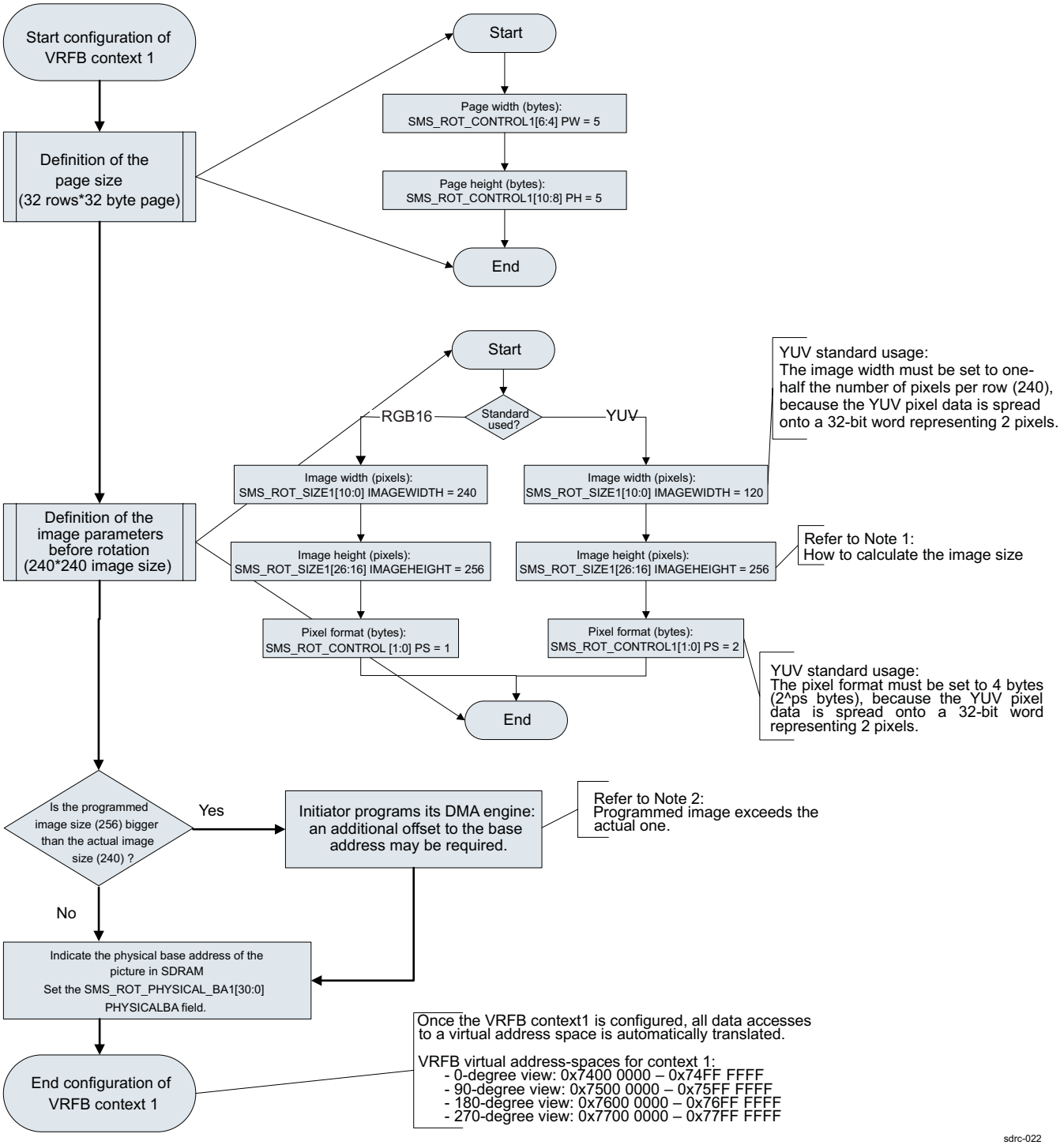


Table 11-107 lists guidelines for calculating image size.

Table 11-107. Calculating Image Size[1][2]

Steps		RGB16 Format (16-Bit Pixel Format)		YUV Format (32-Bit Pixel Format)	
		IMAGEWIDTH	IMAGEHEIGHT	IMAGEWIDTH	IMAGEHEIGHT
1	Calculate the required number of pages per line and per column.	240 pixels/32 bytes ×2 bytes per pixel = 15 pages per line	240/32 rows = 7.5, rounded to 8 pages per column	120 pixels/32 bytes ×4 bytes per pixel format = 15	240/32 rows = 7.5, rounded to 8
2	Using the number of pages calculated in Step 1, calculate the image size in pixels.	15 ×32 bytes/2 bytes per pixel = 240	8 ×32 rows = 256	I5 ×32 bytes/4 bytes per pixel format = 120	256

(1) Working on a page basis, the image size must be a multiple of the page size.
(2) Depending on the page dimensions and the image size, the programmed image size (256) is larger than the actual image size (240).

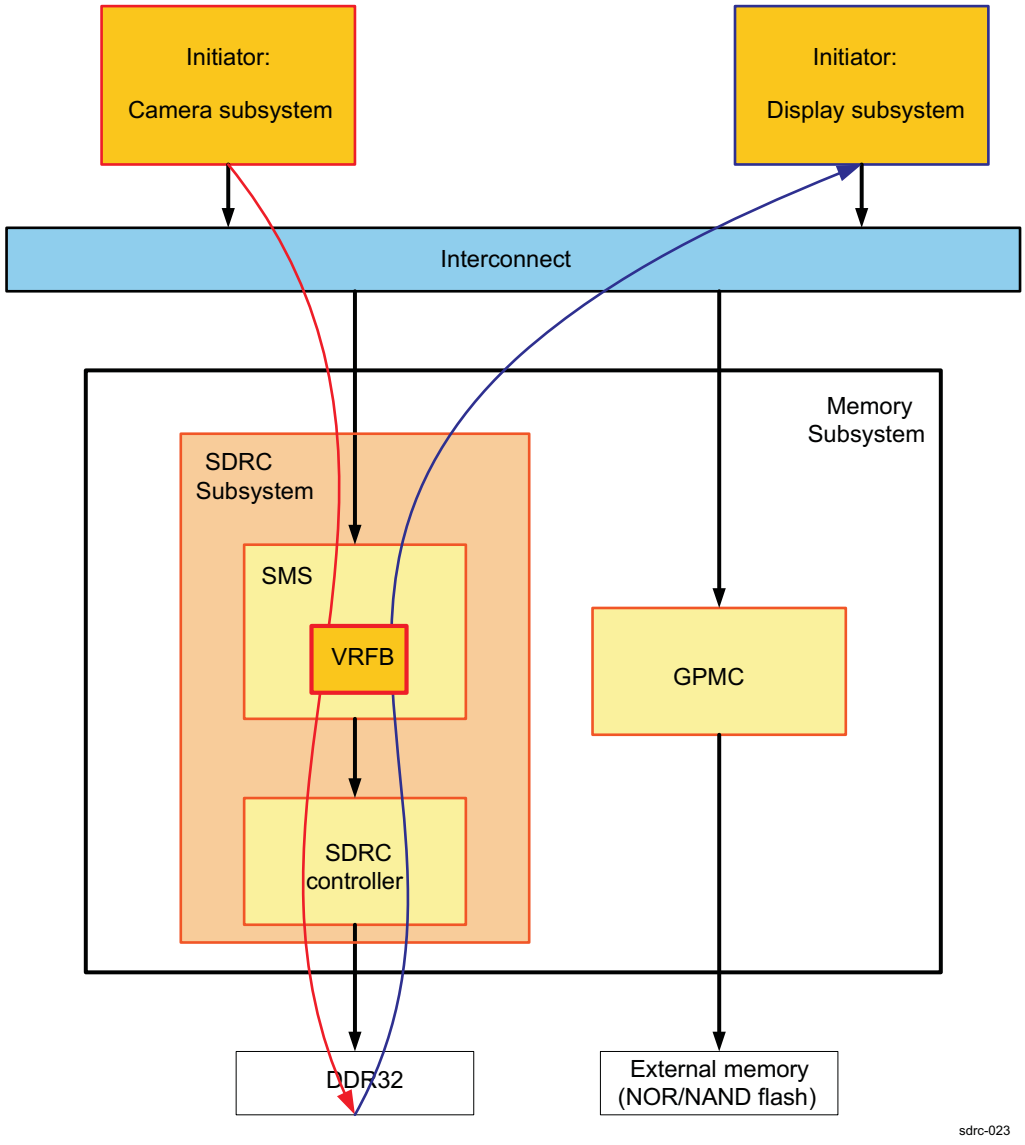
To use VRFB rotation, each initiator must configure its DMA engine correctly; an additional offset to the base address may be required. For details, see Chapter 15, *Display Subsystem*.

11.2.6.1.3 Applicative Use Case and Tips

Use Case Scenario: Display a Rotated QVGA Image

Figure 11-66 shows the use case for displaying a rotated QVGA image. The camera captures an image with a particular rotation and stores the pixel data in external memory. Displaying the image on the LCD requires a different orientation.

Figure 11-66. Display a Rotated QVGA Image



The following components interact while displaying the rotated image:

- SDRC subsystem
- Camera subsystem
- Display subsystem
- External DDR
- VRFB

The following conditions exist for this application:

- QVGA image size is 320 ×240 (width ×height).
- Pixel format is YUV4:2:2 (YUV2) in little endian.
- The rotation view is 90 degrees.
- The VRFB context is VRFB context 1.
- The camera module writes QVGA images to external DRAM at 15 fps.
- The display subsystem reads QVGA images from external DDR32 at 60 fps and displays them on the LCD screen.
- Read and write initiators use burst mode:

  - The camera uses 8 × 64-bit burst size.
  - The display uses 8 × 32-bit burst size.
  - Double-indexing mode is selected.
- The physical address of the buffer in external memory is 0x8030 0000.
- The memory allocation for the buffer is 320 × 240- × 16-bit (or 160 × 240 × 32-bit).
- The camera subsystem uses its dedicated DMA channel 0.
- The display subsystem uses its video channel 1.

Configuring the VRFB Through the SMS Registers

The size of the page supported by the DDR32 memory is 1K byte, which is arranged as 32 × 32 bytes page (width ×height).

Configuring the page size:

- SMS_ROT_CONTROLn[6:4] PW = 5 (where n = 1)
- SMS_ROT_CONTROLn[10:8] PH = 5

Configuring the image parameters:

- SMS_ROT_SIZEn[10:0] IMAGEWIDTH = 160 (where n = 1)
- SMS_ROT_SIZEn[26:16] IMAGEHEIGHT = 256
- SMS_ROT_CONTROLn[1:0] PS = 2

Physical base address and rotation angle:

- SMS_ROT_PHYSICAL_BAn[30:0] PHYSICALBA = 0x8030 0000 (where n = 1).

The image data is accessed at the following virtual address range for 90-degree rotation: 0x7500 0000 - 0x75FF FFFF.

CAUTION

Image rotation using the YUV2 image format causes the stream to become untidy. The display must be specifically configured to read and reorganize the stream to conform to the YUV2 standard.

Tips for Configuring Successful Rotation

The following guidelines ensure optimal image rotation:

- Page arrangement:

  Usually, the recommendation is to have a *square* page. If this is not possible, the longest page side should correspond to the access direction that requires the maximum bandwidth; set the longest page side to optimize the page break.

  Using a 1024-byte page size, a 32 ×32-byte page arrangement is used as an example. With a 2K-byte page organized as a 32 ×64 byte page, depending on the read or write operation, set the longest page size to optimize the page break. If 0 is written and the 270-degree view is read, page height is greater than page width (PH > PW). Set PH to 64 bytes (PH = 6).
- Virtual address memory arrangement:

  When accessing image data through virtual addresses, the maximum line size supported by the VRFB is 2,048 pixels.

  In the memory buffer, the distance between two vertically adjacent pixels is fixed at 2,048 multiplied by the pixel format in bytes. This means that when reading or writing image data through virtual addresses, there must be an offset of: (2048 - IMAGEWIDTH) × PS bytes at the end of every line.
- Base address alignment:

  For optimization, the base address is aligned on the page size. For instance, a 1K-byte page size organized as a 32 ×32-byte page is aligned on 0x400 (to be adjusted on the base address).
- To improve performance on 90° rotation consider two things:
  - Because a read access can appear more critical than a write access, a posted write may be appropriate.

– When performing burst accesses with 90° or 270° rotation views, the burst is split on the memory side, adding latency.
  For a burst access with 90° rotation, split the data of the burst on the write side (not the read side):
  - Write in 270° and read in 0°.

11.2.6.2 SMS Mode of Operation

11.2.6.2.1 SDRAM Memory Scheduler and Arbitration Policy

The SDRAM memory scheduler improves access to external memory by:

- Optimizing SDRAM bandwidth
- Prioritizing requests to external memory

This mechanism relies on a complex arbitration policy that employs specific terminology:

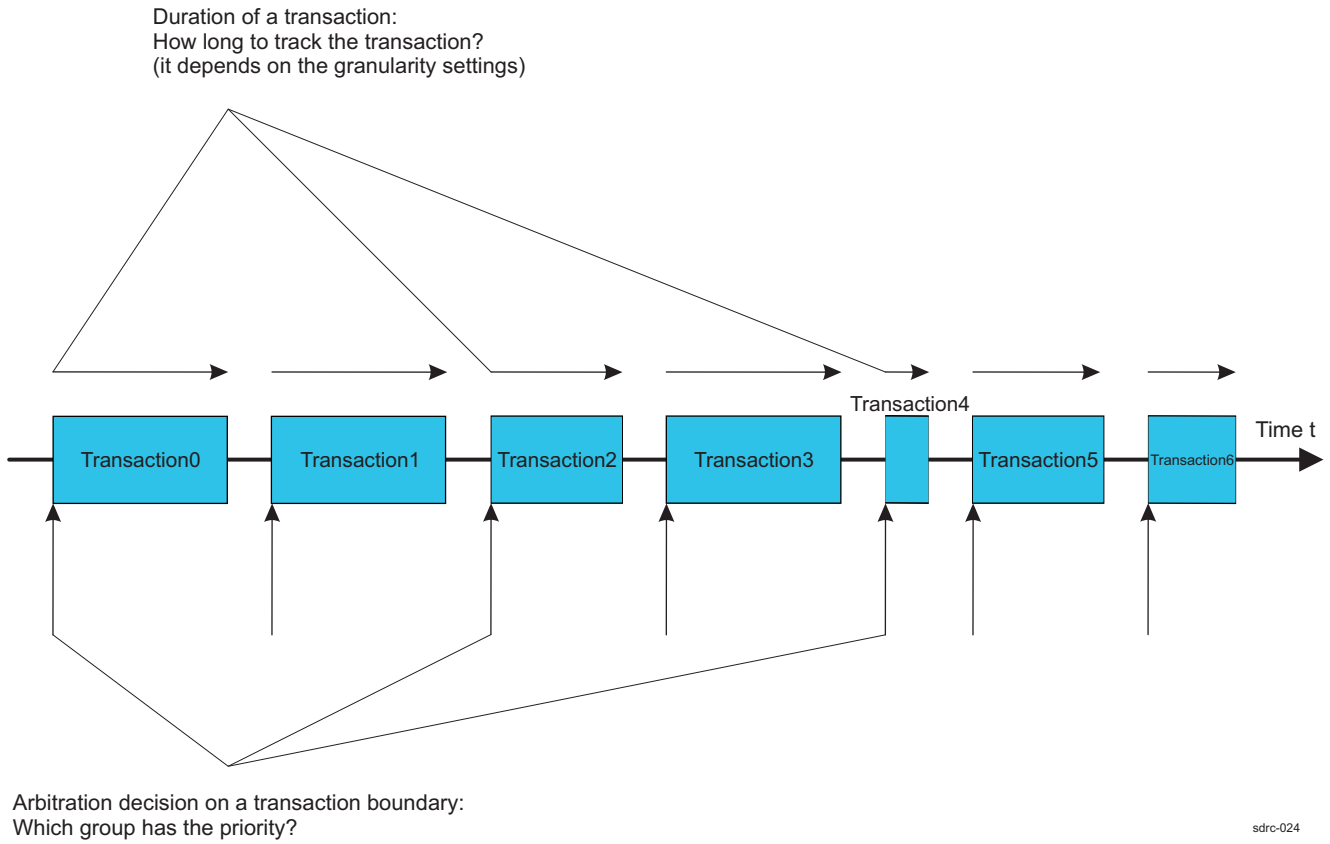
- Group: a FIFO queue of requests from initiators
- Class: A collection of groups
- Transaction: A full burst request
- Arbitration grant: Authorization of a service requested by an initiator

The arbitration policy operates on two interdependent mechanisms:

- The arbitration decision, which establishes priority for processing requests. The question: Which request is processed next? occurs on a transaction boundary.
- Arbitration granularity, which determines the length of an arbitration grant. The question: How long to keep the grant? defines the boundary of the arbitration decision point in time.

Figure 11-67 shows the link between these two concepts. On a transaction boundary, the questions: What request is serviced next, and for how long? merge the two mechanisms. The mechanisms for specifying the granularity of requests also influence the boundary of a transaction.

Figure 11-67. Arbitration Granularity Versus Arbitration Decision



11.2.6.2.2 Arbitration Decision

The SMS module controls arbitration. Requests from initiators for access to the external SDRAM are collected into several independent FIFO queues, and each queue is assigned to a class. Priority is then assigned to groups and classes; this defines the next request to be serviced.

11.2.6.2.2.1 Burst-Complete Mechanism

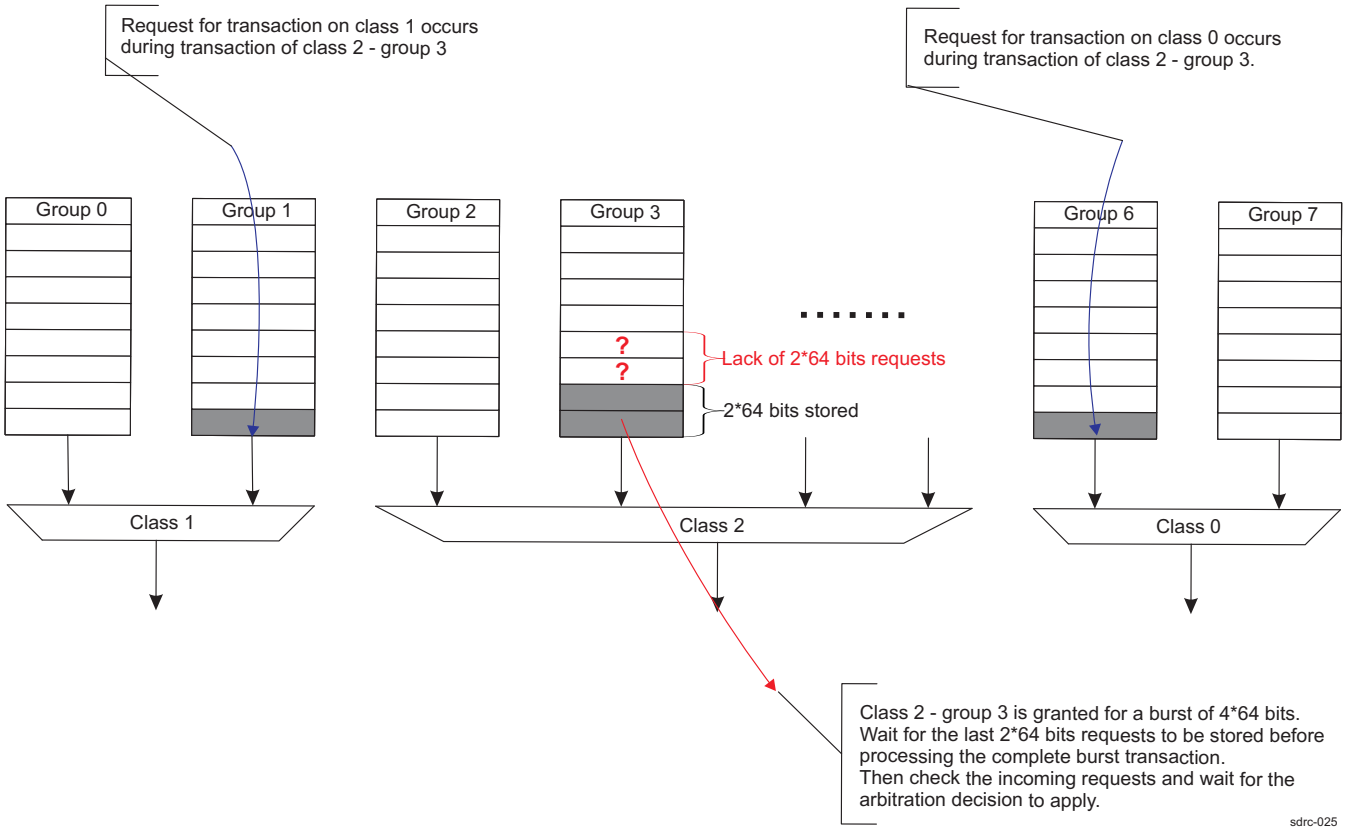
The burst-complete mechanism applies granularity to the arbitration scheme. Access cannot be granted to a group within a class until a complete burst has been stored in the FIFO.

Example:
There is no ongoing transaction on Class 0. The initiator requests a 4 ×64-bit burst on Group 3 of Class 2 (SMS_CLASS_ARBITER2[27] BURST-COMPLETE = 0x1). Only 2 ×64-bit requests are stored in the FIFO.

The mode of operation is:

- Wait for the last 2 ×64-bit request to be stored in the FIFO.
- Arbitration is requested once all requests of a burst have been received.
- After the 4 ×64-bit burst is complete, the transaction occurs.
- The arbitration choice mechanism resumes:
  - Were there any incoming requests during the last transaction?
  - What is the next request to be serviced/granted?

Figure 11-68. BURST-COMPLETE On Class 2-Group 3



11.2.6.2.2.2 Priority Between Groups

The SMS.SMS_CLASS_ARBITERi[7:6] HIGHPRIOVECTOR field defines the priority between groups in a class.

11.2.6.2.2.3 Priority Between Classes

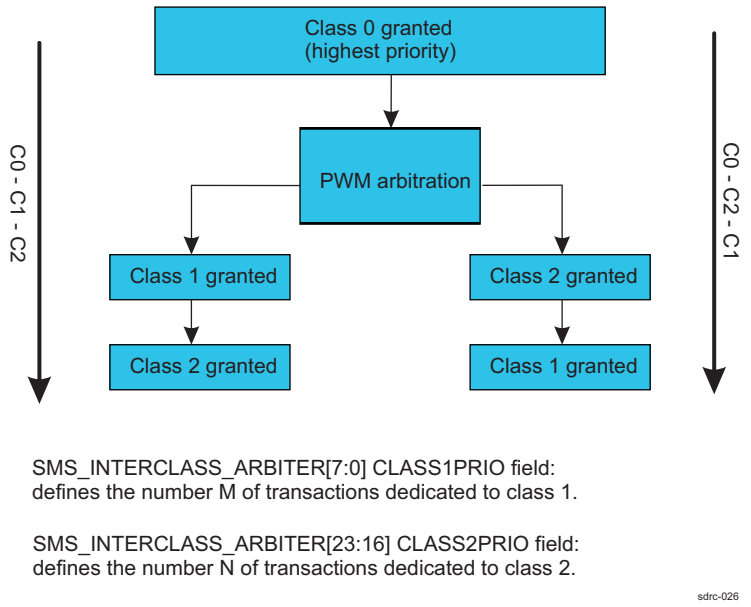
Class 0 always has the highest priority. Then, a weight is programmed within the arbitration of Class 1 and Class 2:

- The SMS.SMS_INTERCLASS_ARBITER[23:16] CLASS1PRIO field specifies the number (M) of transactions dedicated to Class 1.
- The SMS.SMS_INTERCLASS_ARBITER[7:0] CLASS2PRIO field specifies the number (N) of transactions dedicated to Class 2.

M and N parameters are used to set a PWM arbitration. For instance, two schemes appear: at time t, Class 1 is serviced for M cycles, and is directly followed by service to Class 2 for N cycles. If Class 2 is serviced first for N cycles, then Class 1 is granted for M cycles. Arbitration is given more importance; Class 1/2 is favored over Class 2/1. As shown in Figure 11-69, two arbitration schemes can appear:

- Class 0 is serviced first, followed by Class 1, and then Class 2 (C0 - C1 - C2).
- Class 0 is serviced first, followed by Class 2, and then Class 1 (C0 - C2 - C1).

Figure 11-69. Priority Between Classes



11.2.6.2.3 Arbitration Granularity

Arbitration is the mechanism that give more or less importance to incoming requests depending on the initiator origin. Granularity influences the boundary of a transaction because it imposes the following questions:

- How many requests to service?
- How long to track the current transaction?
- When to make the next arbitration decision, or when does the next transaction boundary occur?

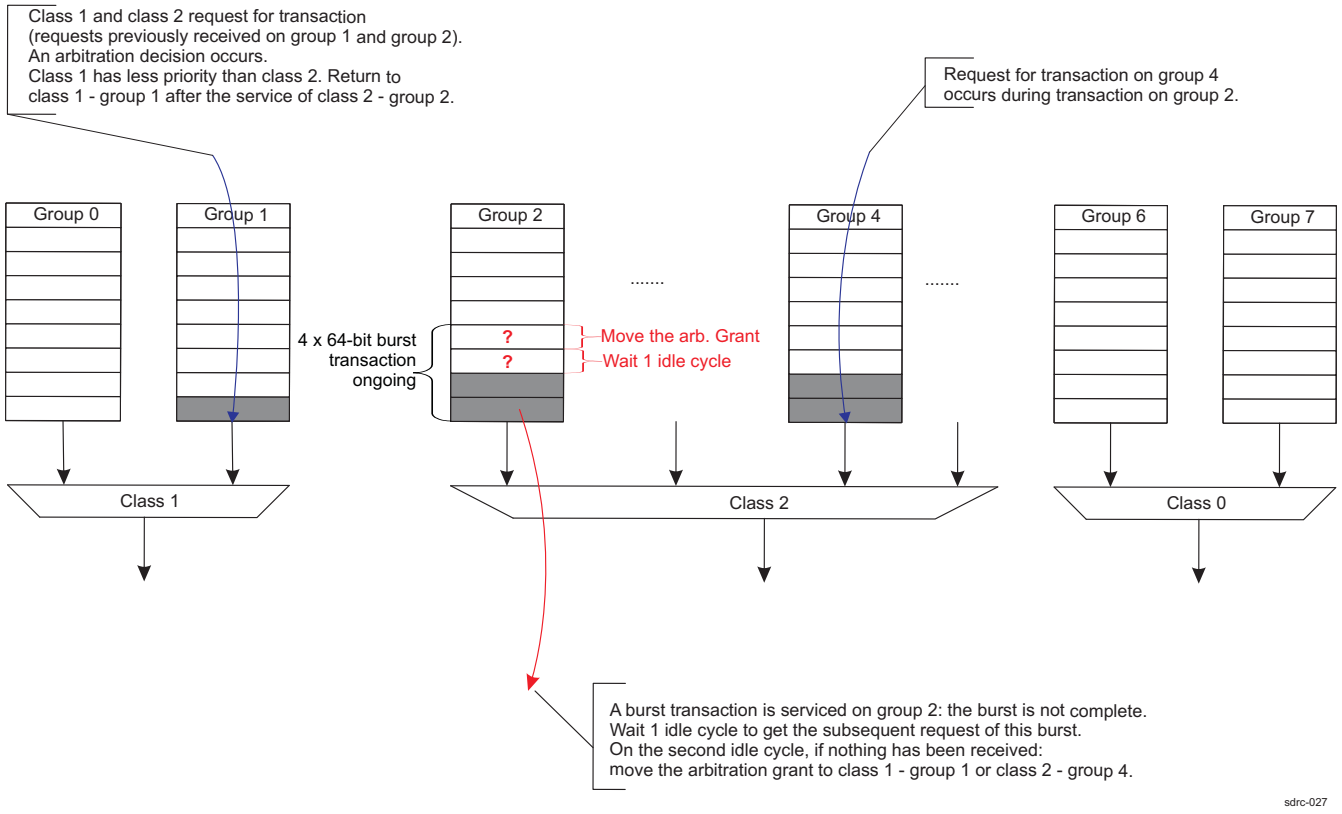
11.2.6.2.3.1 Idle Cycle

The idle gives more granularity to the arbitration; it lets the arbiter wait one idle cycle before moving the arbitration grant to another thread.

Within a burst (see Figure 11-70 for an example):

1. A thread requests a burst transaction. According to the thread ID, the last request was serviced on Class 2 - Group 2.
2. The thread cannot provide the subsequent request of the burst. The module waits for one idle cycle without moving the arbitration grant, to receive the subsequent request (from the same thread).
3. On the second idle cycle, if the subsequent request has not been received, the arbitration grant is moved so that a request from another thread is serviced.

Figure 11-70. Idle Cycle Mechanism Within A Burst



Two bursts must be serviced at the burst boundary. One idle cycle after servicing these two bursts, the arbitration grant is moved.

11.2.6.2.3.2 Extended-Grant Mechanism

The extended-grant mechanism gives additional granularity to the arbitration. An extended grant defines the number of consecutive transactions (from 1 to 3) a group is granted.

Example:

Requests were already available on Class 1-Group 0 and Class 2-Group 3.

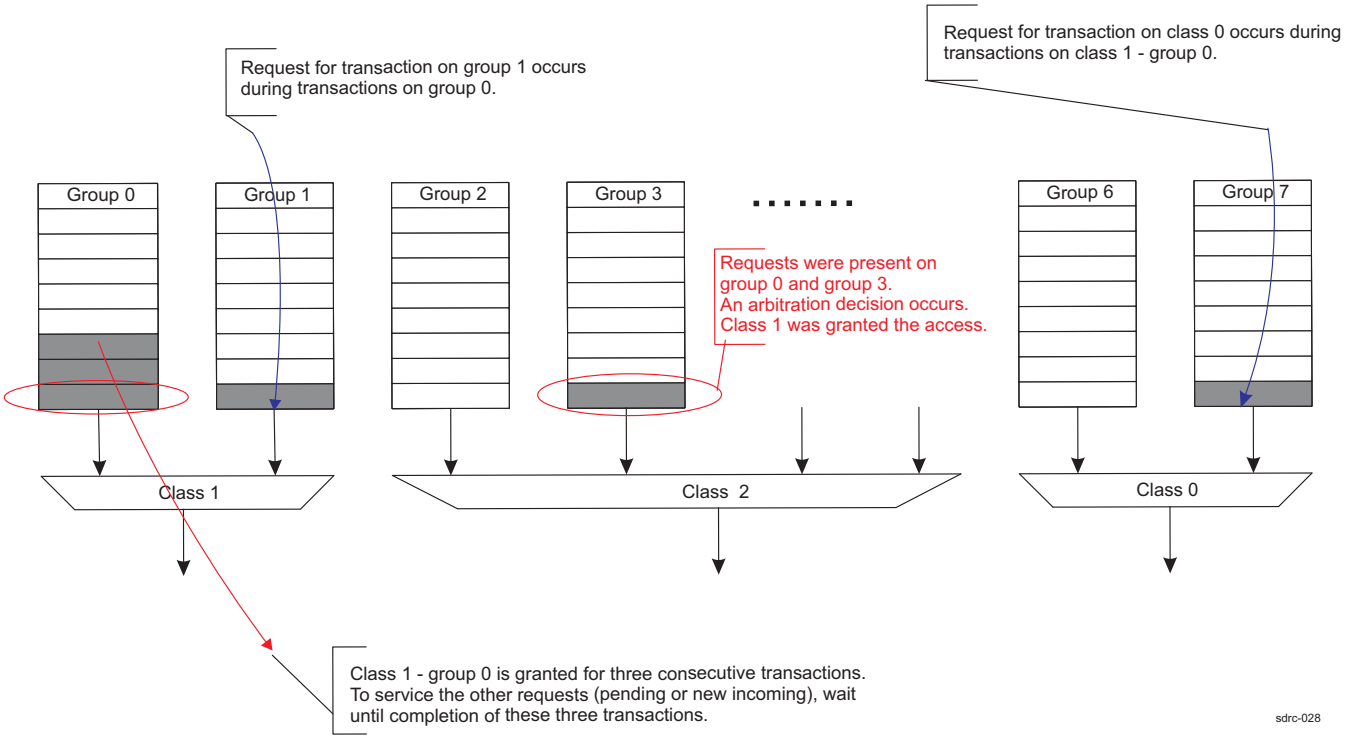
An arbitration decision occurs: grant access to Class 1-Group 0.

A request is now being serviced on Class 1-Group 0 with SMS_CLASS_ARBITER1[9:8] EXTENDEDGRANT = 0x3 (three consecutive transactions are granted to Group 0). The mode of operation is:

- Process the request for three cycles.
- If another request comes from another class (even Class 0) during the processing, it is ignored; the arbiter does not treat incoming requests.
- After three consecutive transactions, return to the arbitration decision:
  - Were there any incoming requests during the last transaction?
  - What is the next request to be serviced?

Figure 11-71 shows this mechanism on Class 1-Group 0:

Figure 11-71. Example of EXTENDEDGRANT Mechanism



11.2.6.2.3.3 Number of Service Mechanism.

The SMS_CLASS_ROTATIONm[4:0] NOFSERVICES field defines the number of consecutive transactions (from 1 to 31) the VRFB is granted. For more information on LRU policy and Number of Service, refer to Section 11.2.4.1.3.

11.2.6.2.4 How these Mechanisms Interact

As described in the previous sections, the arbitration within the SMS module is based on several mechanisms giving even more importance to the arbitration. Wen we need to grant the access on a transaction boundary, we recall that the question we have to answer is: What will be the next request to be serviced, and for how long?
Thus, the two important concepts regarding the SMS mode of operation are:

- The arbitration decision: we choose the next request to be serviced and so we grant it the access to the external memory.
- The arbitration granularity: depending on the mechanism used, we define the boundary of the transaction: how long do we keep the grant?

Figure 11-72 describes the decision happening between classes.
Figure 11-73 explains the priority between groups within Class 1.
So far, the arbitration decision has taken place at class boundaries and within a class containing two groups. Figure 11-74 recalls the generic way to service and therefore prioritize the requests within a class. Finally, Figure 11-75 recalls mechanisms used to define the boundary of the transaction being serviced (how long do we keep the grant?).

Figure 11-72. Arbitration Between Classes

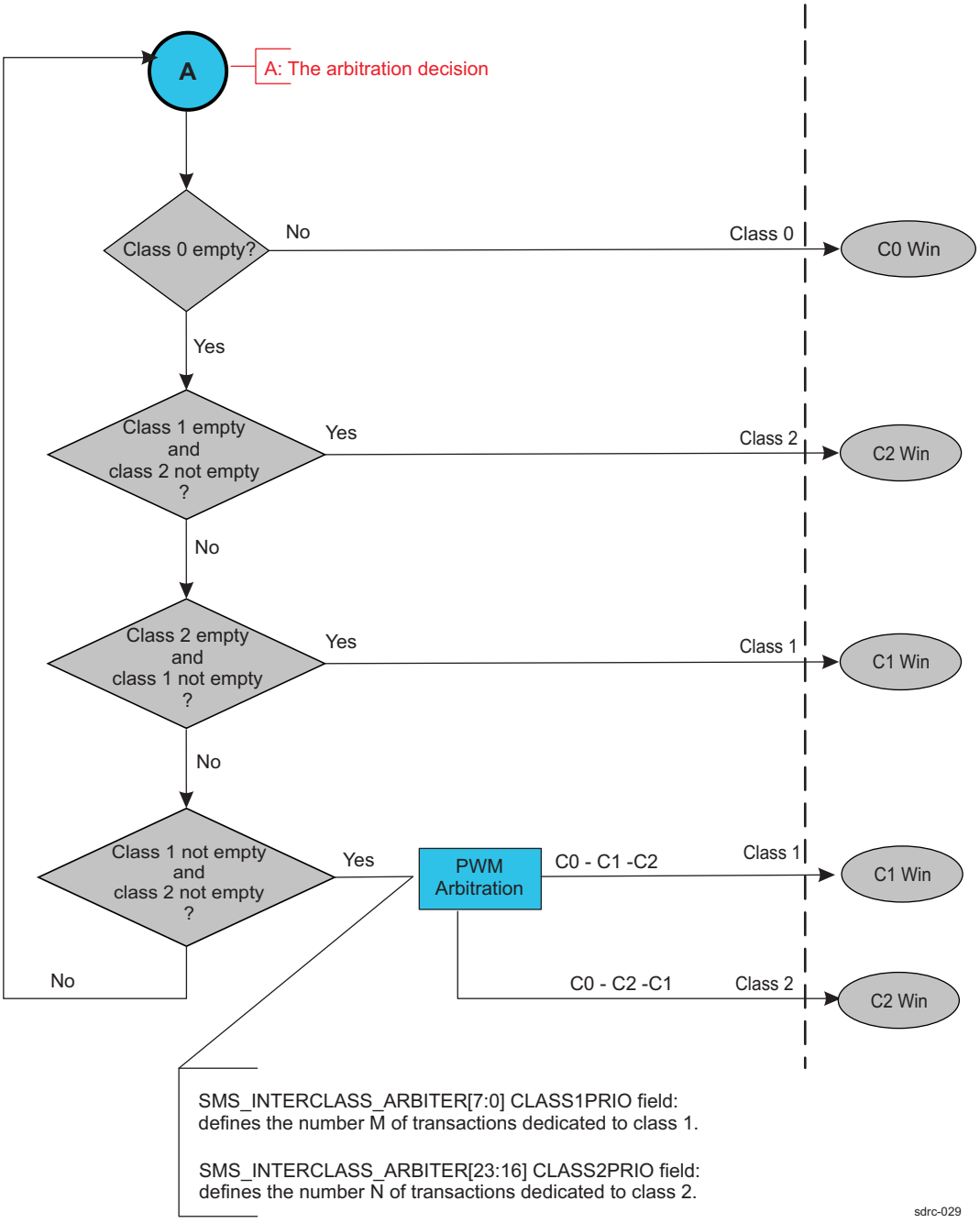


Figure 11-73. Arbitration Within a Class

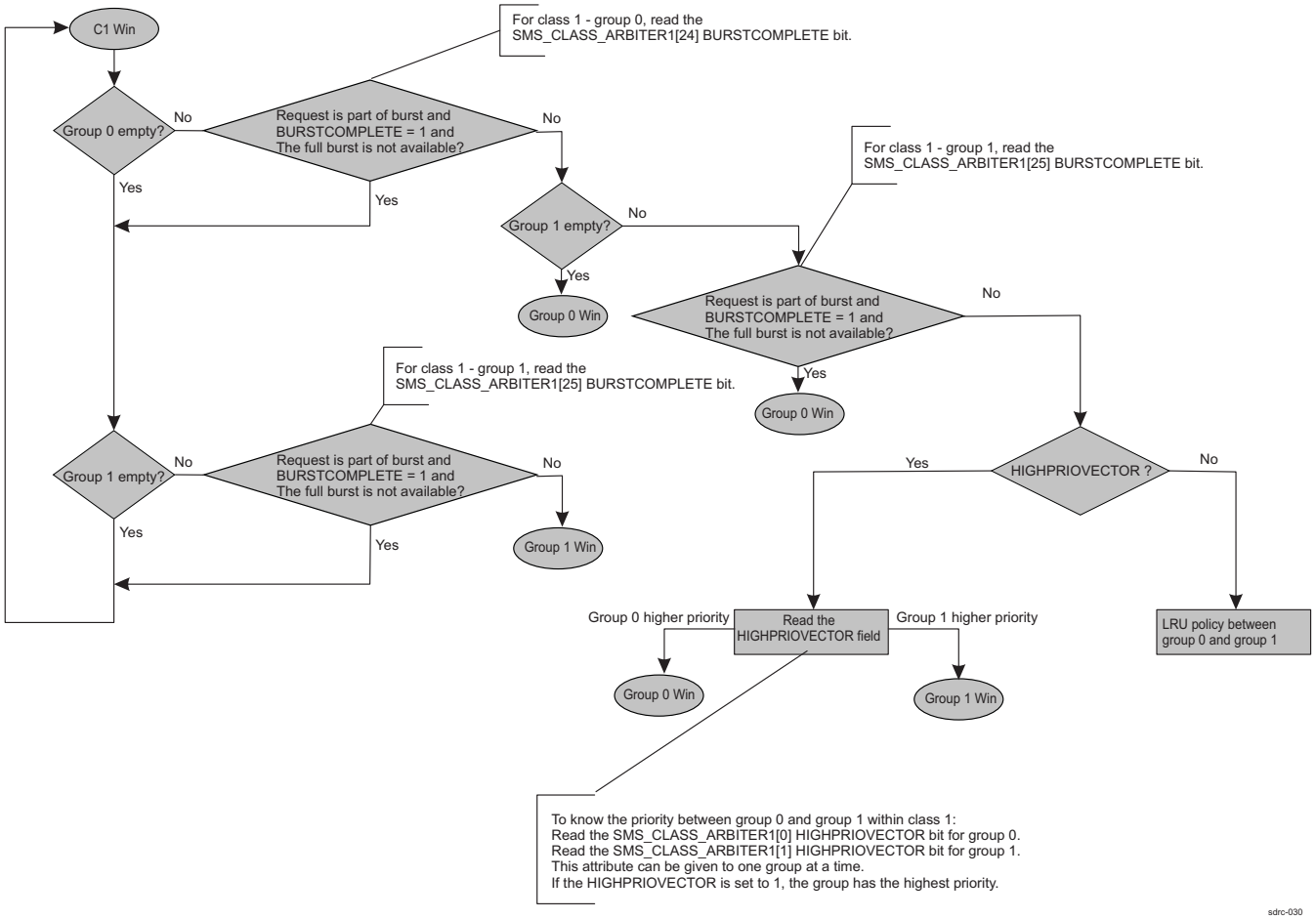


Figure 11-74. Generic Arbitration Decision

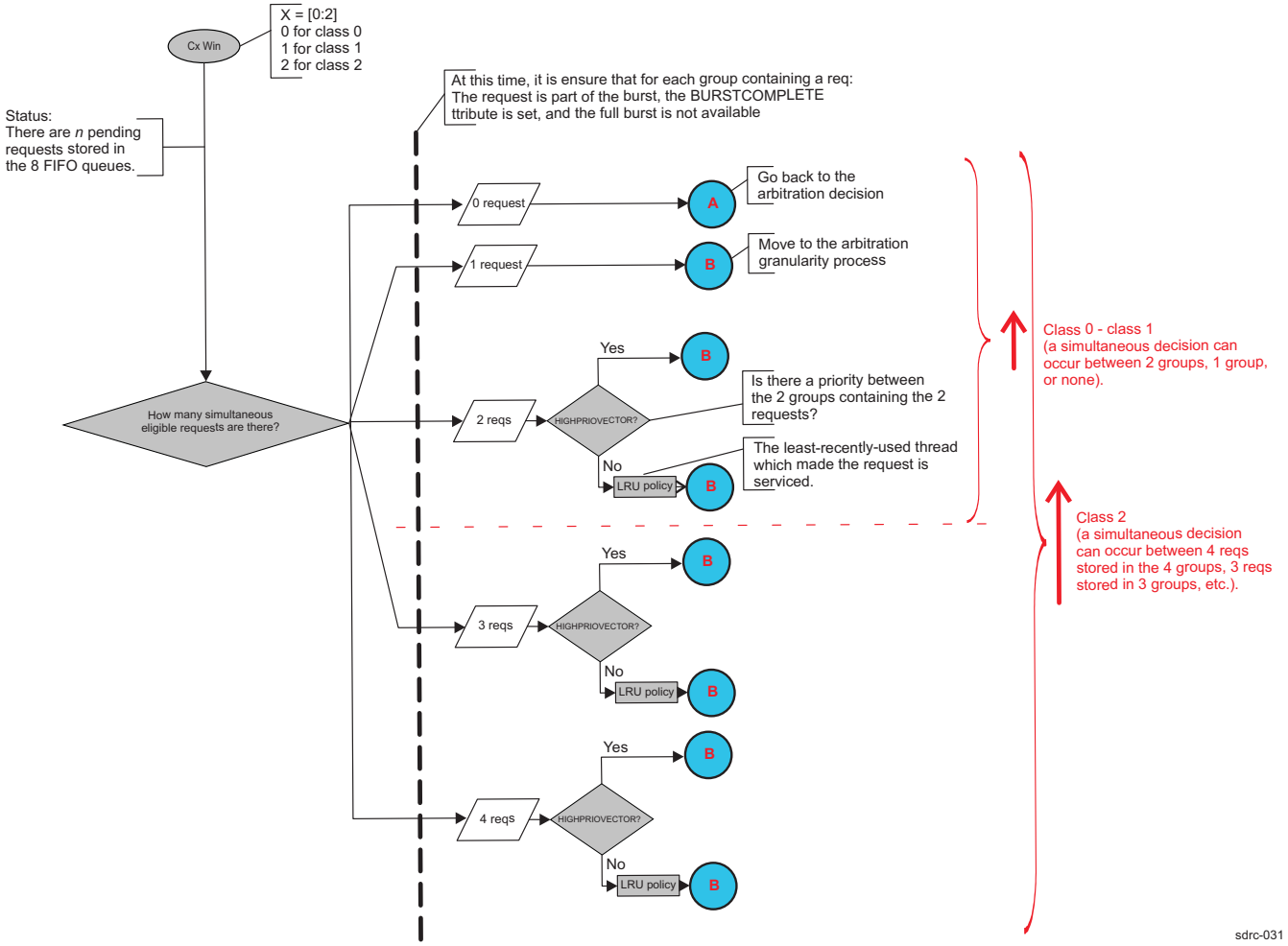
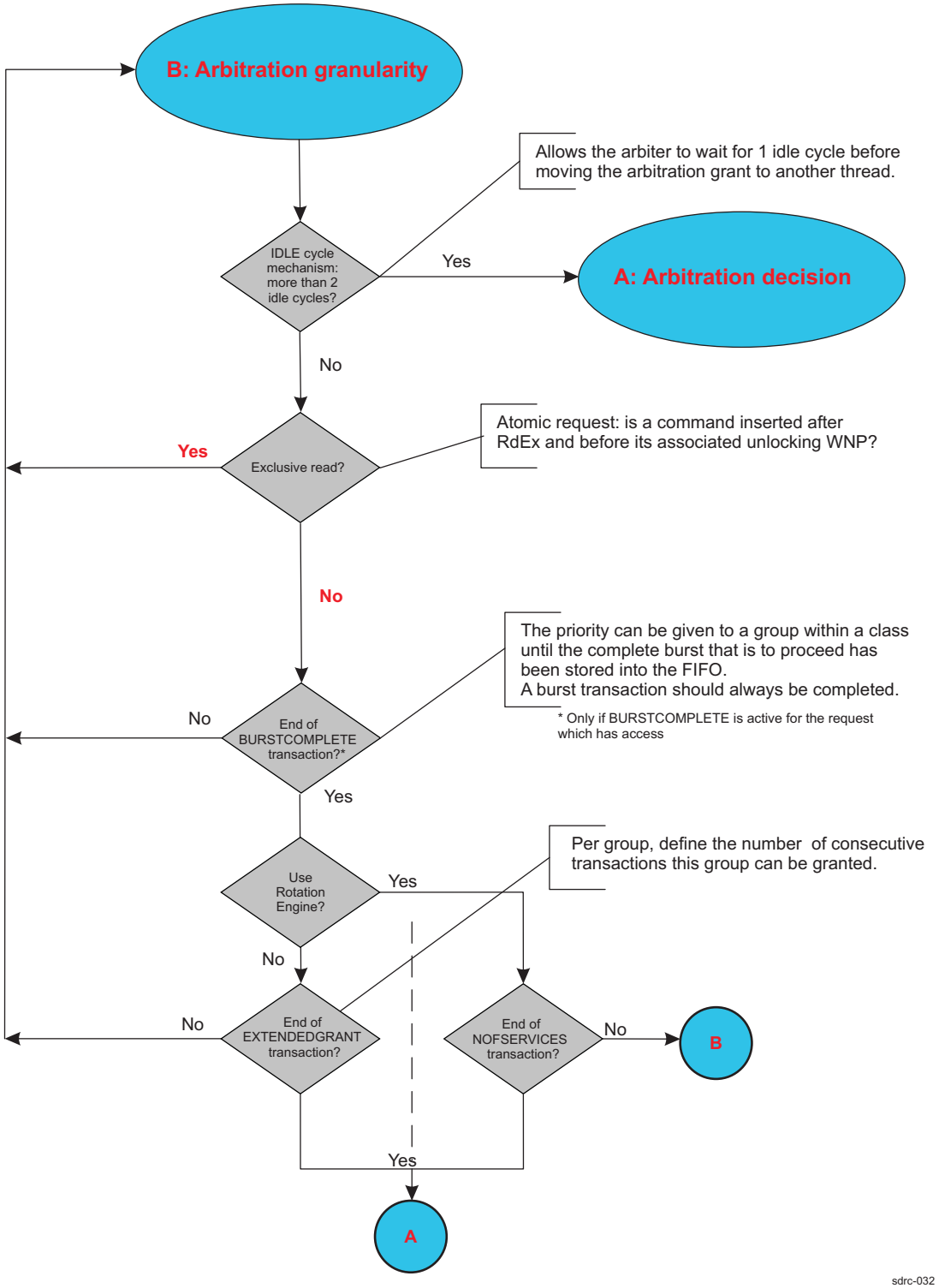


Figure 11-75. Arbitration Granularity



11.2.6.3 Typical SDRC connection to an External SDRAM Device

The SDRAM controller (SDRC) transfers data to and from standard SDRAM based on user request. It interfaces between an external SDRAM memory subsystem and the SDRAM Memory Scheduler that optimizes memory access for each application.

This section shows a typical connection between the SDRC module and an external SDRAM. The use case explains important parameters and their required configurations.

11.2.6.3.1 External Memory Attached to the SDRC Module

The SDRC module supports the following external memory types:

- Mobile single data rate (SDR)-SDRAM devices
- Low-power DDR SDRAM device

This section describes how to calculate the SDRC timing parameters and provides a typical setup of those parameters according to the access performed.

A 512 Mb mobile DDR SDRAM memory with the following characteristics has been chosen:

- Type: Mobile DDR SDRAM
- Size: 512 Mbits (8M ×16 ×4 banks)
- Data bus: 16-bit width
- Speed: 133-MHz clock frequency, 7.5-ns clock period

11.2.6.3.2 DDR-SDRAM Memory General Facts

This section details the interconnection of the mobile DDR SDRAM with the SDRC interface; it then gives a basic SDRAM workflow and introduces the difference between the autorefresh and self-refresh concepts.

11.2.6.3.2.1 Mobile DDR Interconnection With SDRC

Figure 11-76 shows a simplified interconnection diagram of the mobile DDR SDRAM with the SDRC interface.

Figure 11-76. Mobile DDR SDRAM to SDRC Interface

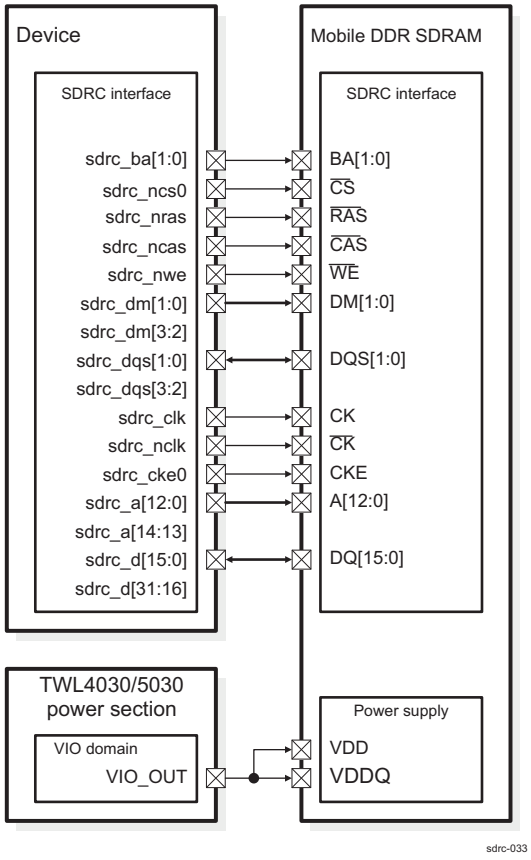


Table 11-108 lists the I/Os of the SDRC module from an SDRC point of view.

Table 11-108. SDRC Signals Description

Signal Name	I/O[(1)]	Description
sdrc_clk, sdrc_nclk	O	Address and control input signal sample clock
sdrc_cke0	O	Clock-enable signal
sdrc_ncas, sdrc_nras, sdrc_nwe	O	Column address strobe, row address strobe, and write enable. Defines the command being entered
sdrc_ba0, sdrc_ba1	O	Bank address outputs
sdrc_dm[3:0]	O	Data mask. sdrc_dm[0] corresponds to the data on sdrc_d[7:0]; sdrc_dm[1] corresponds to the data on sdrc_d[15:8]. sdrc_dm[3:2] unconnected.
sdrc_ncs0	O	Chip-select for external SDRAM selection (only one enabled here)
sdrc_a[12:0]	O	Address bus sdrc_a[14:13] unconnected.
sdrc_d[15:0]	I/O	Data bus sdrc_d[31:16] unconnected.
sdrc_dqs0, sdrc_dqs1	I/O	DDR data strobe. Output with read data, input with write data sdrc_dqs[3:2] unconnected.

(1) I = Input; O = Output

11.2.6.3.2.2 SDRAM Operating Flow

This section describes the basic DDR-SDRAM workflow. For more information about the SDRAM operation, see the *Low Power Double Data Rate (LPDDR) SDRAM* specification (JEDEC committee JC42.3) or other appropriate mobile DDR SDRAM documentation.

Figure 11-77 shows a simplified operating diagram of the power-up sequence required to correctly power up an SDRAM.

Figure 11-78 shows a simplified diagram of the normal operating sequence.

Figure 11-77. SDRAM Power-Up Sequence

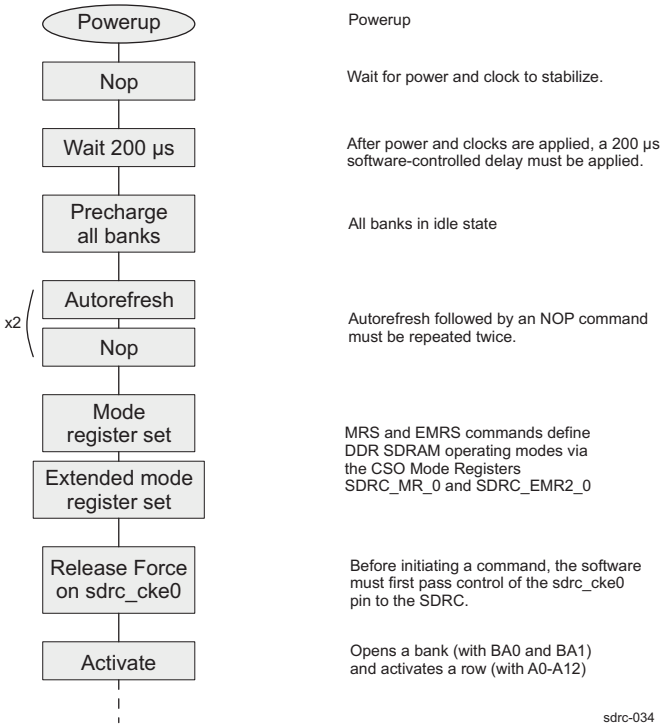
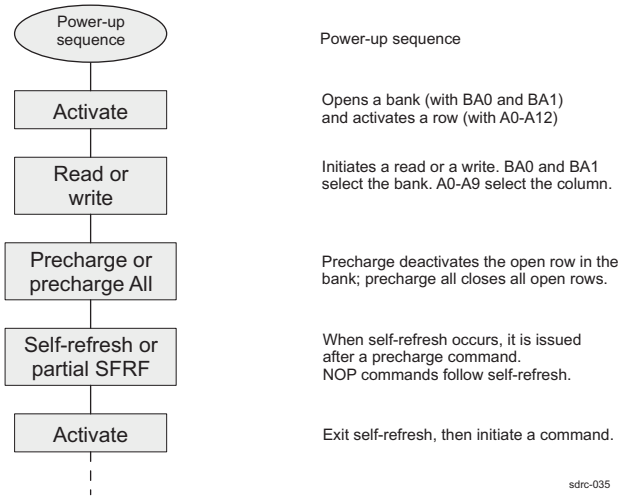


Figure 11-78. Normal Operating Sequence



11.2.6.3.2.3 Autorefresh vs Self-Refresh

What are the differences between an autorefresh and a self-refresh?

DDR SDRAM devices require a refresh of all rows in 64-ms interval as specified in the JEDEC LPDDR SDRAM standard. The autorefresh period may depend on the technology. Each refresh is generated either by an autorefresh command from the SDRC in normal mode, or by an internal counter in the SDRAM in self-refresh mode.

The autorefresh command must be generated each time a refresh is required. The average refresh interval period (tREFI), also called autorefresh duty cycle, is given by the division of 64 ms into the number of rows (8192 for the selected mobile DDR SDRAM). An internal refresh controller in the SDRAM takes care or the refreshing process and control, and the autorefresh command must be issued regularly by the SDRAM controller.

The self-refresh command is used when the system is powered down and enables data retention even if the clock is stopped.

For more information on refresh modes, see Section 11.2.4.4.7, *Refresh Management*.

11.2.6.3.3 SDRC Typical Setup

When a given external memory device is selected, the SDRC must be configured according to the memory parameters.

11.2.6.3.3.1 Memory Type General Configuration

Table 11-109 summarizes the mobile DDR SDRAM address configuration.

Table 11-109. Mobile DDR SDRAM Address Configuration

Item	Description
RAM size	512 Mbits
Number of banks	4
Bank address pins	BA0, BA1
Row addresses	A0–A12
Column addresses	A0–A9
tREFI (µs)	7.8

The following parameters must be configured in register SDRC.SDRC_MCFG_p, where p = 0 (configuration register for the SDRAM connected to the sdrc_ncs0) as follow:

- SDRC_MCFG_p[19] ADDRMUXLEGACY = 0x1 selects the flexible address mux scheme.
- SDRC_MCFG_p[1:0] RAMTYPE = 0x1 selects DDR-SDRAM rather than SDR-SDRAM.
- SDRC_MCFG_p[2] DDRTYPE = 0x0 is the default setting that sets the DDR memory type as mobile DDR.
- SDRC_MCFG_p[17:8] RAMSIZE = 0x020 is the RAM address space size expressed in 2M-byte chunks. In this example, the DDR SDRAM size is 512M-bits or 64M-bytes. These are 32 2M-byte chunks, that is, 20 2M-byte chunks expressed in hexadecimal format.
- SDRC_MCFG_p[4] B32NOT16 = 0x0. The external SDRAM bus width is 16-bit.
- SDRC_MCFG_p[26:24] RASWIDTH = 0x2 because RAS width is 13 bits.
- SDRC_MCFG_p[22:20] CASWIDTH = 0x5 because CAS width is 10 bits.
- SDRC_MCFG_p[3] DEEPPD = 0x1. The memory device supports deep-power-down mode.
- Other bit fields as LOCKSTATUS and BANKALLOCATION may be set up as required.

11.2.6.3.3.2 Mode Register Configuration

The SDRAM mode of operation is defined through the mode register as follows (p = 0 for CS0):

- SDRC_MR_p[6:4] CASL = 0x3. The CAS latency, or read period, is 3 clock cycles long.
- SDRC_MR_p[2:0] BL = 0x2. The memory burst length must be set at 4 for DDR SDRAM.
- SDRC_MR_p[3] SIL = 0x0. The serial mode is always used.
- SDRC_MR_p[9] WBST = 0x0. Write burst equals read burst.

To configure the DLL/CDL module as second clock write path, set up the SDRC_DLLA_CTRL register as follows:

- SDRC_DLLA_CTRL[7] WRITEDDRCLKX2DIS = 0x1 so that the DLL/CDL write path can be used as mobile DDR write path.
- SDRC_DLLA_CTRL[3] ENADLL = 0x1 enables the DLL.
- SDRC_DLLA_CTRL[2] LOCKDLL must be set to 0 for high-speed operations.
- SDRC_DLLA_CTRL[1] DLLPHASE = 0x0 for read accesses and 0x1 for write accesses. This corresponds to a CDL delay of a quarter of a clock period or a fifth of a clock period, respectively.

The recommended configuration is to use the data path using double frequency logic rather than the DLL/CDL write path. This is obtained by setting WRITEDDRCLKX2DIS to 0x0.

11.2.6.3.3.3 SDRAM Programmable AC Timing Parameters Configuration

Some parameters are hard-coded in the SDRC. Apart from those parameters, users can program a list of programmable AC timings using the SDRC_ACTIM_CTRLA_p and SDRC_ACTIM_CTRLB_p registers, where p = 0 or 1, depending on the SDRAM chip (in the section p = 0, because only one chip is connected to the SDRC interface). The setup values are expressed in clock cycles.

Table 11-110 summarizes the most important AC timing parameters to be configured. F = 133 MHz and tCK = 7.5 ns for the clock signal.

Table 11-110. Mobile DDR SDRAM AC Timings Parameters

AC Timing Parameter	Description	Duration (ns)
tRC	Row cycle time	67.5 (or 9 tCK)
tRAS	Row active time	45 (or 6 tCK)
tRP	Row precharge time	22.5 (or 3 tCK)
tRCD	nRAS to nCAS delay time	22.5 (or 3 tCK)
tRFC	Autorefresh cycle time	80 (min)
tRRD	Row active to row active delay time	15 (or 2 tCK)
tDPL	Data-in to precharge command time or tWR (write recovery time)	15 (or 2 tCK)

Table 11-110. Mobile DDR SDRAM AC Timings Parameters (continued)

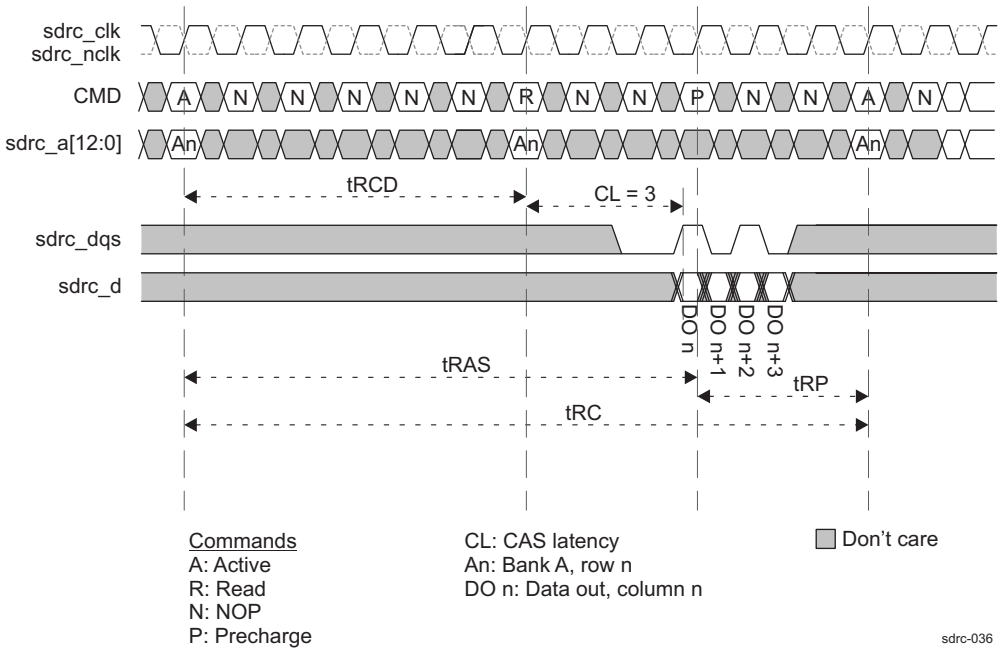
AC Timing Parameter	Description	Duration (ns)
tWTR	Write-to-read time (also called tCDLR). Last data in to read command	1 tCK
tCKE	CKE minimum pulse width time (high and low pulse width)	2 tCK
tDAL	Last data in to active delay time	2 tCK + tRP
tXSR	Exit self-refresh to active command time	120
tXP	Exit power-down to active command time	25

Before calculating the timing parameters on the SDRC side, lets define some terms used to describe the timing between the controller and the memory device:

- nRAS to nCAS delay time is the row-activating phase, or minimum time between an ACTIVE and a READ or WRITE command.
- CAS latency is the time between a READ or a WRITE command and the data driven on DQ lines. With CL = 3, the first data element is valid at (2 × tCK + tAC) after the clock at which the READ command is registered. With CL = 2, the first data element is valid at (tCK + tAC) after the clock at which the READ command is registered. tAC is the DQ output window relative to the clock, and is the long term component of DQ skew.
- Row active time is the minimum bank activate to PRECHARGE time. A PRECHARGE cannot start until tRAS has elapsed.
- Row precharge time: Following the PRECHARGE command, a subsequent command to the same bank cannot be issued until tRP is met. After tRP, an ACTIVE command to the same bank can be initiated.
- Row cycle time is the minimum time between the activation of a first row and the activation of another row in the same bank. It is also called ACTIVE to ACTIVE command period.
- Last data in to active delay time is the minimum time between the last data in and the next ACTIVE command.
- Autorefresh cycle time: Refreshing starts with an AUTO REFRESH command and ends when tRFC is met.

Figure 11-79 shows a simplified timing diagram of a burst read of SDRAM.

Figure 11-79. SDRAM Burst-Read Timing Diagram



The SDRC_ACTIM_CTRLA_p register (p = 0) must be programmed with the AC timing parameters listed in Table 11-111. Values are expressed in number of tCK periods.

Table 11-111. Calculation of the SDRC.SDRC_ACTIM_CTRLA_p Timing Parameters

DDR-SDRAM Parameter Name	Number of Clock Cycles (F = 133 MHz, tCK = 7.5 ns)	SDRC_ACTIM_CTRLA_p Bit Field Value
tRC	9	TRC = 0x09
tRAS	6	TRAS = 0x6
tRP	3	TRP = 0x3
tRCD	3	TRCD = 0x3
tRFC	80 / 7.5 (minimum)	TRFC = 0x0B
tRRD	2	TRRD = 0x2
tDPL	2	TDPL = 0x2
tDAL	2×1 + 4 (in clock cycles)	TDAL = 0x6

The SDRC_ACTIM_CTRLB_p (p = 0) register must be programmed with the AC Timing Parameters listed in Table 11-112. Values are expressed in number of tCK.

Table 11-112. Calculation of the SDRC.SDRC_ACTIM_CTRLB_p (p = 0) Timing Parameters

DDR-SDRAM Parameter Name	Number of Clock Cycles (F = 133 MHz, tCK = 7.5 ns)	SDRC_ACTIM_CTRLB_p Bit Field Value (p = 0)
tCDLR	1	TWTR [17:16] = 0x1
tCKE	2	TCKE [14:12] = 0x2
tXP	25 ns(1)	TXP [10:8] = 0x4
tXSR	120 / 7.5	TXSR [7:0] = 0x10

(1) JEDEC LPDDR SDRAM standard states 25 ns with a minimum of 1 clock period.

11.2.6.3.3.4 Autorefresh Management Configuration

In normal operation mode, the autorefresh is managed by the SDRAM refresh configuration register SDRC_RFR_CTRL_p (where p = 0). The autorefresh counter is uploaded with the result of (tREFI / tCK) - 50.

50 (cycles) is a margin in case a command is already in progress on the SDRAM interface, thus requiring delay before initiating the autorefresh command. See Section 11.2.5.3.6, *Autorefresh Management*, for more details.

Table 11-113 details the required configuration.

Table 11-113. Calculation of the SDRC.SDRC_RFR_CTRL_p (p = 0) Timing Parameter

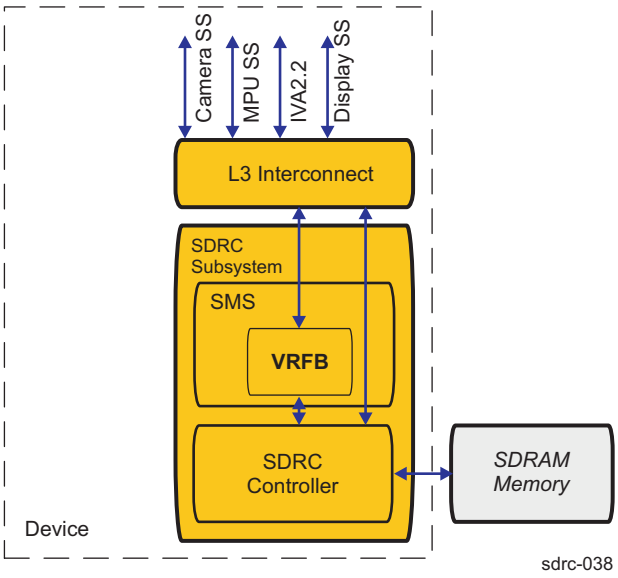
DDR-SDRAM Parameter Name	Contents Formula	SDRC_RFR_CTRL_p Bit Field Value (p = 0)
Autorefresh counter value	(7.8 µs / 7.5 ns) – 50	ARCV [23:8] = 0x03DE
Autorefresh enable	Load Autorefresh counter with 8 for maximum Autorefresh burst length	ARE [1:0] = 0x3

11.2.6.4 Camcorder Use Case: How to Configure the VRFB

11.2.6.4.1 Overview

This section discusses the configuration of the VRFB for rotating the video on an external VGA FlatLink™ 3G. The VRFB receives a video from the camera subsystem as a sequence of images. These images are then saved into SDRAM through the SDRC. Figure 11-80 is an overview of the VRFB in the camcorder use case.

Figure 11-80. SDRC Camcorder Use Case Overview



After the VRFB configuration, the rotation process is transparent to the MPU operations. Four buffers are required in SDRAM to ensure stabilization, a smooth encoding process, and a tearing-free preview on the LCD display. Each of the four buffers uses a concurrent context. For each context, the following parameters must be configured:

- Page and pixel size
- Picture parameter (image height and width)
- Buffer physical base address

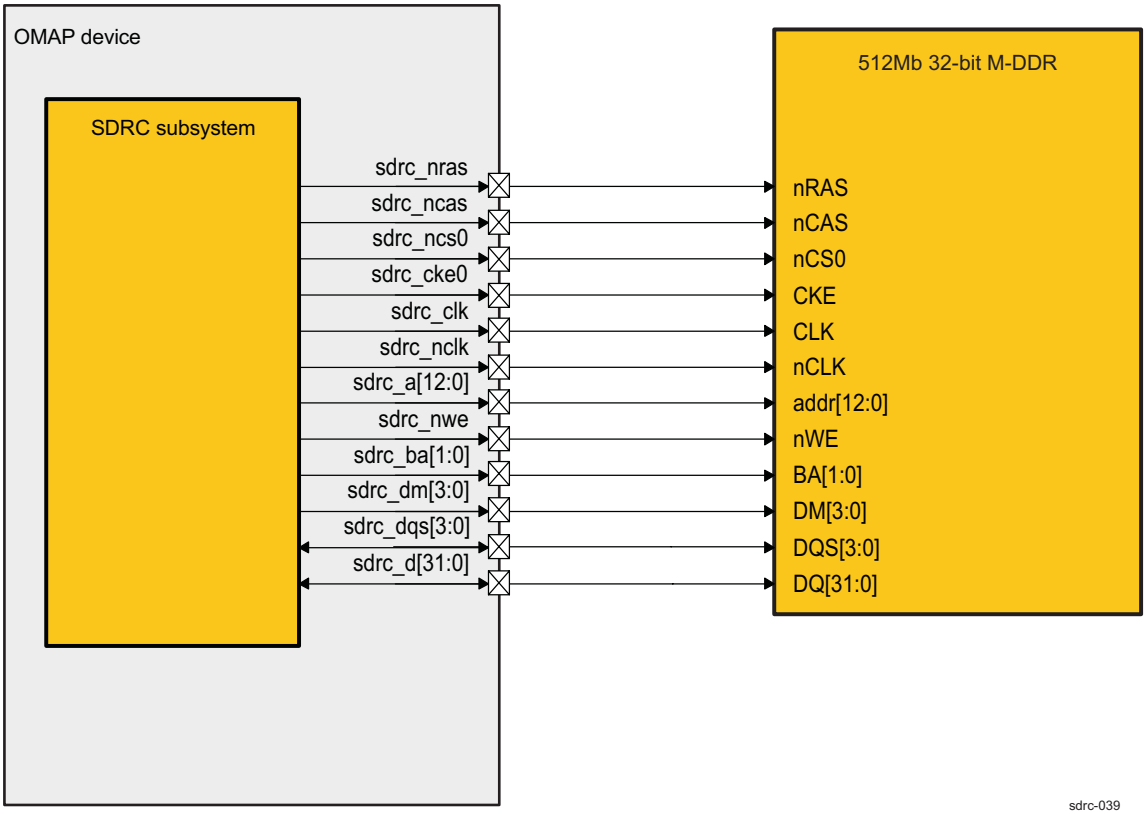
11.2.6.4.2 Environment

In this use case, the SDRC controller is connected to an external 512-Mb 32-bit Mobile-DDR SDRAM 166-MHz device (16M × 32).

The SDRC_CLK clock source fed to the SDRAM is the PRCM CORE_L3_ICLK output.

The SDRAM device is connected to the SDRC subsystem (see Figure 11-81). For more information on the SDRC environment, see Section 11.2.2, *SDRC Subsystem Environment*.

Figure 11-81. SDRC Camcorder Use Case Environment



11.2.6.4.3 Data Path

In this use case, the VRFB manages the rotation of the video stream, which is a sequence of images in YUV 4:2:2 format received from the camera subsystem.

- Write actual image received from the camera subsystem into SDRAM through the VRFB in the first view (0°).
- Display the image in SDRAM on the external display through the display subsystem in any of the four available VRFB views (0°, 90°, 180°, or 270°).

See Figure 11-66 for the VRFB data path of this use case.

The row increment in the SDRAM is programmed in the display subsystem registers and depends on the VRFB rotation angle.

11.2.6.4.4 Programming Flow

A main function automatically configures the VRFB contexts. This function is called four times to configure each of the four contexts. For each context, a set of parameters is given to the function:

- The actual image width and height (from the camera subsystem)
- The physical address (of the given context)
- The rotation of the context

The main function initializes the VFRB in three steps:

1. Virtual frame buffer configuration. This is done with a page size calculation function, as described in Section 11.2.6.4.4.1.
2. Image size configuration. This is done with a picture size calculation function, as described in Section 11.2.6.4.4.2.
3. Physical base address configuration, as described in Section 11.2.6.4.4.3.

11.2.6.4.4.1 Page Size Calculation Function

The page size is configured through three parameters: page width (PW), page height (PH), and pixel size (PS). The three page size parameters are configured into the SMS_ROT_CONTROLn registers, where n is the context number (n = 0 to 3):

- Page width: SMS_ROT_CONTROLn[6:4] PW. The PW parameter defines the page width according to the value of 2^{PW} bytes.
- Page height: SMS_ROT_CONTROLn[10:8] PH. The PH parameter defines the page height according to the value of 2^{PH} lines.
- Pixel format: SMS_ROT_CONTROLn[1:0] PS. The PS parameter defines the pixel size according to the value of 2^{PS} bytes. In this use case, the pixel size is a constant linked to the YUV 4:2:2 format. For more details about YUV 4:2:2 format, see Section 11.2.6.1.2, *Setting a VRFB Context*.

The recommendation is to have a squared page arrangement. The page can be defined as a 32-byte ×32-byte array (1024-byte page) or 16-byte ×16-byte array (256-byte page). This arrangement is pixel-based, which means the pixel size must be part of the calculation. Other settings are possible, but from a performance point of view it is recommended to have the page size consistent with the SDRAM page.

In this use case, however, the page size calculation function defines the page width and the page height. The pixel size is not part of the calculation. Hence, instead of having a 64 ×32 page (because a pixel takes 2 bytes) the function calculates a 32 ×32 page. This gives a page size of 1K-byte, and memory area usage is optimized.

First, the function checks whether the width of the image received from the camera is a multiple of 32 bytes. If the width is not a multiple of 32 bytes, the function checks whether the image is a multiple of 16 bytes. If the image from the camera is neither a multiple of 32 bytes nor 16 bytes, the function sets the page width to 32 bytes as the default. In this use case, the image received from the camera has a width of 736 pixels.

1. Check that 736 is a multiple of 32: 736 / 32 = 23. It is. As a consequence, the page width is 32 (bytes).
2. Configure a page width of 32 with the 2^{PW} bytes formula: SMS_ROT_CONTROLn[6:4] PW = 0x5.

   Then, the function checks whether the height of the image received from the camera is a multiple of 32 (rows). If the height is not a multiple of 32 (rows), the function checks whether the image is a multiple of 16 rows. If the image from the camera is neither a multiple of 32 rows nor 16 rows, the function sets the page height to 32 rows as the default. In this use case, the image received from the camera has a height of 560 lines.
3. Check that 560 is a multiple of 32: 560 / 32 = 17.5. It is not.
4. Check that 560 is a multiple of 16: 560 / 16 = 35. It is. As a consequence, the page height is 16 (lines).
5. Configure a page height of 16 with the 2^{PH} lines formula: SMS_ROT_CONTROLn[10:8] PH= 0x4.

   The pixel format of this use case is YUV 4:2:2. Two pixels are stored on 4 bytes.
6. Configure a pixel size of 4 bytes with the 2^{PS} bytes formula: SMS_ROT_CONTROLn[1:0] PS = 0x2.

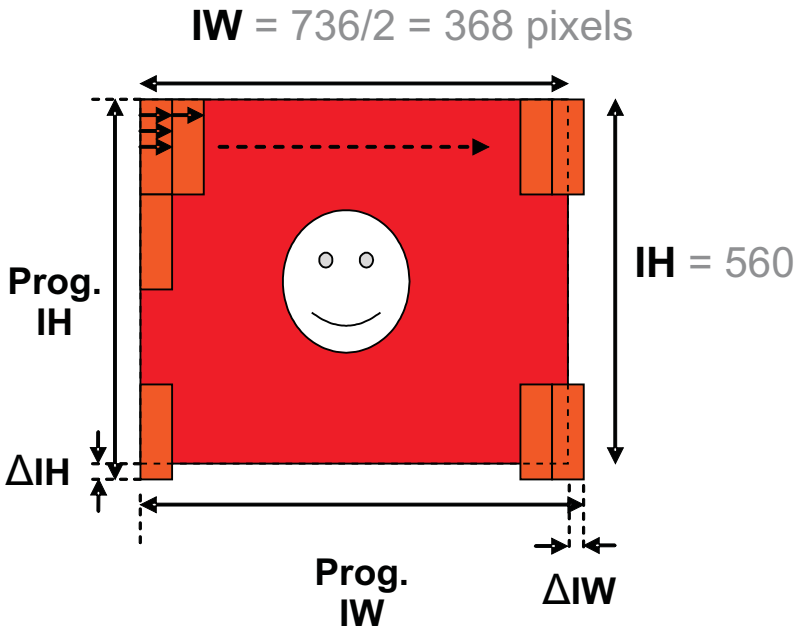
11.2.6.4.4.2 Picture Size Calculation Function

The picture size is configured through two parameters: image width (IMAGEWIDTH) and image height (IMAGEHEIGHT). Two sizes must be distinguished:

- Actual image width and actual image height . This is the size of the image received by the camera subsystem.
- Programmed image width and programmed image height. This is the size of the image programmed in the VRFB.

Figure 11-82 shows the actual image width (IW) and image height (IH) versus the programmed image width (IW + Δ_{IW}) and programmed image height (IH + Δ_{IH}).

Figure 11-82. VRFB Actual Image Size vs Programmed Image Size



The programmed image size parameters are configured into the SMS_ROT_SIZEn registers, where n is the context number (n = 0 to 3):

- Image width: SMS_ROT_SIZEn[10:0] IMAGEWIDTH. This parameter is expressed in the number of pixels.
- Image height: SMS_ROT_SIZEn[26:16] IMAGEHEIGHT. This parameter is expressed in the number of pixels.

The picture size calculation function determines the required number of pages per line and per column using the page size calculated previously. That is a page width of 8 pixels (32 bytes and 4-byte pixel) and a page height of 16 pixels.

1. Determine the required number of pages per line with the formula:
   [(actual_image_width / 2) *pixels* / page_width *bytes*] ×pixel_size *bytes/pixel*
   This gives a required number of pages per line of [(736 / 2) / 32] ×4 = 46.
   If the number of pages per line is not an integer, it must be rounded up; for instance, if the result was 46.25 the number of pages required per line would be 47. Figure 11-82 shows this point.
2. Determine the required number of pages per column with the formula:
   (actual_image_height *pixels*) / page_height *lines*
   This gives a required number of pages per column of (560) / 16 = 35.
   If the number of pages per column is not an integer, it must be rounded up.

Finally, calculate the programmed image size in pixels using the number of pages calculated above:

1. Calculate the programmed image width with the formula:
   (rounded_up_number_of_pages_per_line ×page_width *bytes*) / pixel_size *bytes/pixel*
   This gives a programmed image width of (46 ×32) / 4 = 368.
   The function sets SMS_ROT_SIZEn[10:0] IMAGEWIDTH = 0x170.
2. Calculate the programmed image height with the formula:
   (rounded_up_number_of_pages_per_column ×page_height *lines*)
   This gives a programmed image height of (35 ×16) = 560.
   The function sets SMS_ROT_SIZEn[26:16] IMAGEHEIGHT = 0x230.

Because the actual image is a multiple of the page size (both width and height, as explained in Section 11.2.6.4.4.1), the actual image size and the programmed size are the same. In other words, ΔIW and ΔIH are null in this use case.

11.2.6.4.4.3 Physical Base Address Configuration

The physical base address is the address of the picture in the external SDRAM. In this use case, the address space of CS0 is 512 Mbits, ranging from 0x8000 0000 to 0x83FF FFFF because the Mobile DDR SDRAM used in example is a 512 Mbits memory device.

The physical address of the image is a parameter entered by the user. It is directly written into the SMS_ROT_PHYSICAL_BAn[30:0] PHYSICALBA bit field (where n = 0 to 11, for the 12 contexts).

11.2.6.4.4.4 VRFB Use Case Summarizing Register Values

Table 11-114 summarizes all the VRFB registers to be configured with the required values.

Table 11-114. VRFB Use Case Summarizing Register Print

Register Name	Context Number n	Physical Address	Value	Value Description
SMS_ROT_CONTROLn	0	0x6C00 0180	0x0000 0452	The page size is 8-pixel wide and 16-pixel high. This is a width of 32 bytes and a height of 16 lines. The pixel format is YUV 4:2:2 (2 pixels take 4 bytes).
	1	0x6C00 0190		
	2	0x6C00 01A0		
	3	0x6C00 01B0		
SMS_ROT_SIZEn	0	0x6C00 0184	0x0230 0170	Actual image size is 736 ×560. This gives 368 ×560 with the YUV 4:2:2 pixel format
	1	0x6C00 0194		
	2	0x6C00 01A4		
	3	0x6C00 01B4		
SMS_ROT_PHYSICAL_BAn	0	0x6C00 0188	0x8090 0000	Example of physical base address
	1	0x6C00 0198	0x8100 0000	
	2	0x6C00 01A8	0x8190 0000	
	3	0x6C00 01B8	0x8200 0000	

11.2.6.5 Understanding SDRAM Subsystem Address Spaces

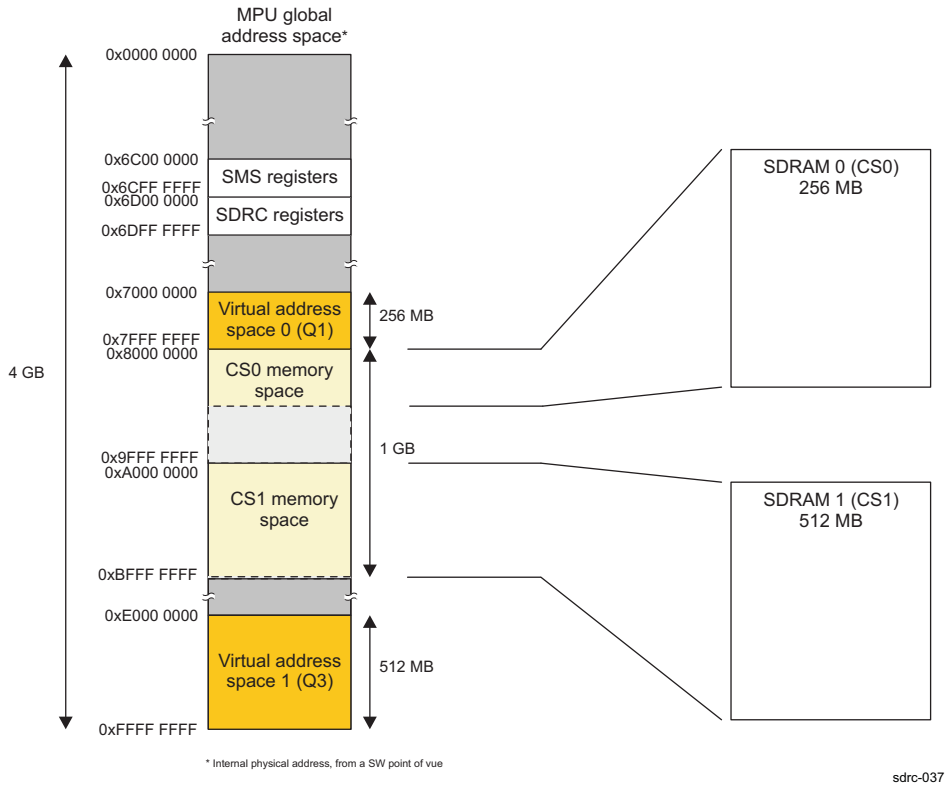
11.2.6.5.1 Physical vs Virtual Address Spaces

One thing that the SMS does is to translate virtual addresses into physical SDRAM addresses in case of rotation only; that is, when accessing virtual address space 0 (quarter 1) and virtual address space 1 (quarter 3) as shown in Figure 11-83. The SMS then reinserts a request, or multiple requests depending on the SMS parameters, in the SMS request path to the SDRC controller (through the VRFB or not).

The SDRAM subsytem global memory space mapping reaches 1.768G-bytes:

- 1G-byte of CS memory space (CS0 and CS1 memory spaces): the SDRC controller automatically accesses the two external memory devices through direct accesses (addresses are simply translated).
- 768M-bytes of virtual address space (address space 0 and address space 1): the SDRC controller automatically accesses the two external memory devices trough re-organized access (requests are modified accordingly to the context number and rotation angle before address translation). See section Section 11.2.6.5.1.2 for more information on VRFB contexts and rotation angles.

Figure 11-83. SDRC Address Space in MPU Global Address Space



Configuration register space is detailed in Table 11-115.

Table 11-115. SDRC and SMS Configuration Register Space

Module	Start Address	End Address	Total Space
SMS	0x6C00 0000	0x6CFF FFFF	16 MB
SDRC	0x6D00 0000	0x6DFF FFFF	16 MB

11.2.6.5.1.1 Physical Address Space

The physical address space of the SDRC is 1G byte (maximum addressing capability).

The SDRC has a memory device capacity of 16M bits to 2G bits. The smallest granularity of supported memory device is 16M bits/2M bytes.

11.2.6.5.1.2 Virtual Address Space

The SDRC-SMS virtual memory space is a memory space used to access a subset of the SDRC memory space through the rotation engine (VRFB). The virtual address space size is 768M bytes split into two parts: the first 256M-byte part is in the second quarter (Q1) of the memory; the second 512M-byte part is in the fourth quarter (Q3) of the memory. See Chapter 2, *Memory Mapping*, for more information on global memory mapping.

The VRFB is a rotation engine that:

- Supports rotations of 0, 90, 180, and 270 degrees
- Can handle up to 12 concurrent rotation contexts

The VRFB has therefore 48 different contexts in the virtual address space. Table 11-116 gives the VRFB address-space memory locations at which the frame buffer object can be accessed. The table summarizes the virtual addresses of all 48 available image buffer from a global memory space (top level) point of view.

Table 11-116. VRFB Contexts vs Rotation Angle

Context Number	0°	90°	180°	270°
0	0x7000 0000	0x7100 0000	0x7200 0000	0x7300 0000
1	0x7400 0000	0x7500 0000	0x7600 0000	0x7700 0000
2	0x7800 0000	0x7900 0000	0x7A00 0000	0x7B00 0000
3	0x7C00 0000	0x7D00 0000	0x7E00 0000	0x7F00 0000
4	0xE000 0000	0xE100 0000	0xE200 0000	0xE300 0000
5	0xE400 0000	0xE500 0000	0xE600 0000	0xE700 0000
6	0xE800 0000	0xE900 0000	0xEA00 0000	0xEB00 0000
7	0xEC00 0000	0xED00 0000	0xEE00 0000	0xEF00 0000
8	0xF000 0000	0xF100 0000	0xF200 0000	0xF300 0000
9	0xF400 0000	0xF500 0000	0xF600 0000	0xF700 0000
10	0xF800 0000	0xF900 0000	0xFA00 0000	0xFB00 0000
11	0xFC00 0000	0xFD00 0000	0xFE00 0000	0xFF00 0000

The physical address of a page is calculated with the formula:

Physical address = Physical base address + Base address of page

where *Physical base address* is defined through the SMS_ROT_PHYSICAL_BAn[30:0] PHYSICALBA field (buffer physical base address on which the rotation occurs), and *Base address of page* is the address obtained in the function of the context number and rotation angle, as given in Table 11-116.

11.2.6.5.2 CS Memory Spaces

Two SDRC chip-selects (sdrc_ncs0 and sdrc_ncs1) are available to access two external SDRAM memories.

11.2.6.5.2.1 SDRAM Capacity Calculation

This section aims to explain how to calculate an SDRAM device capacity.

From the following SDRAM characteristics:

- Four banks: BA0–BA1
- Row addresses: A0–A12
- Column addresses: A0–A9
- 16-bit data bus to external memory

First calculate the number of addressable locations:

- Number of address lines: 13 (A0-A12)
- Number of banks address lines: 2 (BA0-BA1)
- Maximum number of rows = 13 (number of address lines used for row decoding)
- Maximum number of columns = 10 (number of address lines used for column decoding)

Total locations in a bank = $(2^{13}) \times (2^{10})$

Then calculate the total locations in the device:

Total locations in the device = (Number of banks) $\times$(Total locations in a bank) = $(2^{2}) \times (2^{13} \times 2^{10}) = 2^{25}$

Finally calculate the SDRAM device capacity:

Total device capacity = (Total locations in the device) $\times$(Device organization) = $(2^{25}) \times (16) = 2^{29}$ bits

Hence, the device has a maximal capacity of 512M bits (or 64M bytes).

In the same way, the SDRC maximal capacity can be calculated as follow:

- Number of address lines: 15 (A0-A14)

- Maximum number of rows = 15 (limited by the number of address lines, hence do not program SDRC_MCFG_p[26:24] RASWIDTH above 15)
- Maximum number of columns = 12 (limited by SDRC_MCFG_p[22:20] CASWIDTH)
- Data bus width to external device: 32-bit

SDRC total capacity = ($2^{15} \times 2^{12}$) $\times$32 = 4G bits (or 512M bytes)

Hence, the SDRC has a maximum addressing capability of 8G bits/1G byte (as seen in Section 11.2.6.5.1.1) and can manage SDRAM devices with capacity of up to 4G bits/512M bytes (see the following Caution).

CAUTION

Because the SDRC controller is aligned on the JEDEC LPDDR1 SDRAM standard, it is guaranteed that SDRAM with up to 2-Gbit capacity are supported. At the time of writing, 4-Gbit low-power DDR SDRAM is not yet in production, and depending on its design (that is, the number of banks, page size, and other characteristics) this SDRAM may or may not be supported by the SDRC controller.

11.2.6.5.2.2 CS Size

Each chip-select has its programmable size. The CS size is expressed in SDRC_MCFG_p[17:8] RAMSIZE (p = 0 or 1 for CS0 or CS1) as a number of 2M-byte chunks.

For instance, when connecting a 256M-bit SDRAM memory (see the mux scheme MUX7 in Table 11-96) to the SDRC controller, RAMSIZE must be set with the value 0x010 (32M bytes = 2MB $\times$16).

11.2.6.5.2.3 CS Start and End Addresses

See Figure 11-84, *CS0/CS1 Chip-Select Start Address Slots,* for a graphical representation of CS starts addresses and CS size.

CS0 start address is fixed at :

- 0x0000 0000 from an SDRC point of view
- 0x8000 0000 from a global memory space point of view.

Therefore, when connecting a 256M bit/32M-byte SDRAM memory (16M $\times$ 8 such as in MUX6, Table 11-96), the CS0 end address is 0x0200 0000 (SDRC point of view).

CS1 start address is programmable and its default value is :

- 0x2000 0000 from an SDRC address point of view
- 0xA000 0000 from a global memory space point of view.

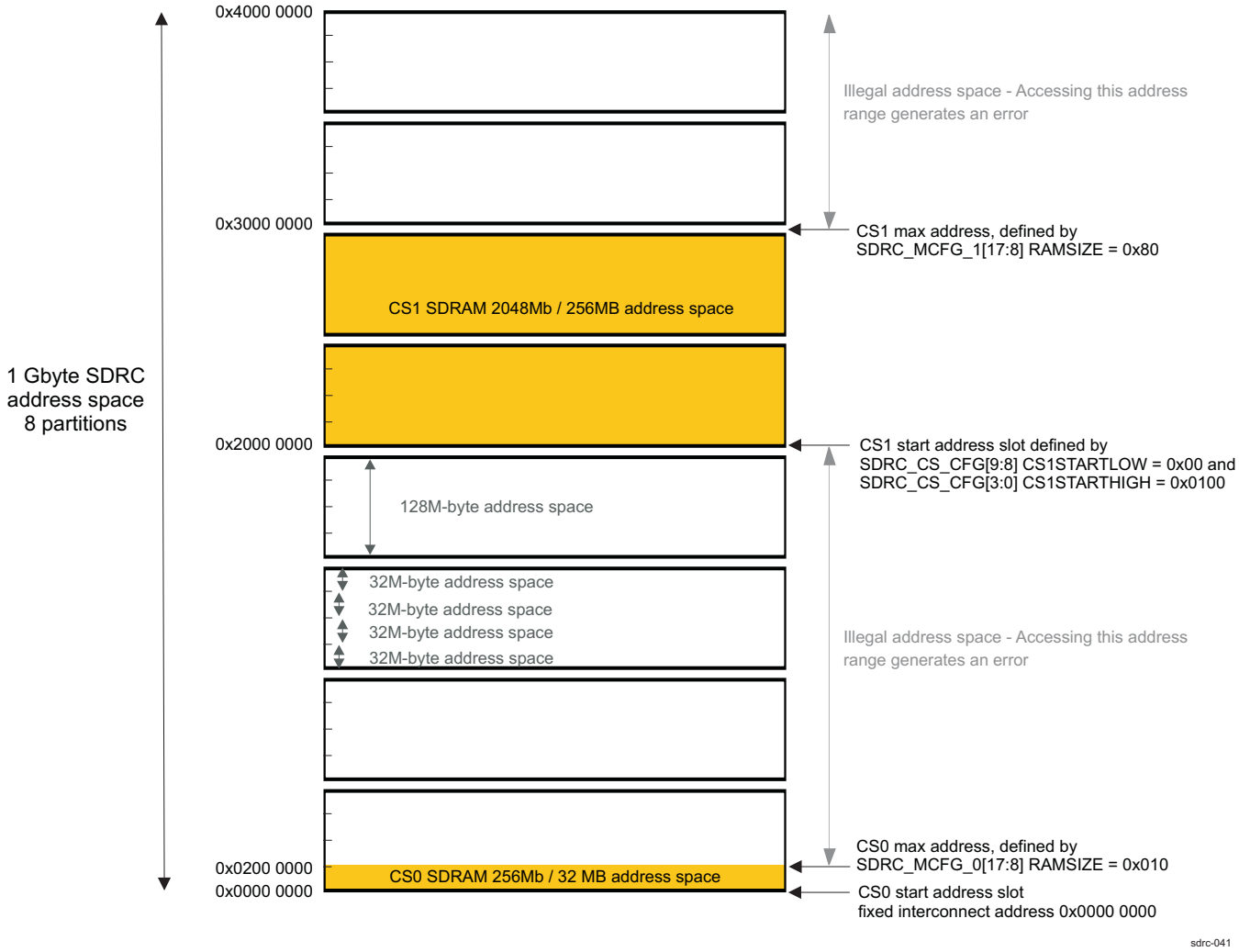
The SDRC 1G-byte address space is segmented into eight 128M-byte address spaces, which are themselves segmented into four 32M-byte address spaces so that 32 possible CS1 start address locations are defined, as shown in Figure 11-84. The first 32M-byte address space of the first 128M-byte address space is reserved for CS0 (address 0x0000 0000). Any other address can be used as the CS1 start address. To define a CS1 start address, set the SDRC_CS_CFG[9:8] CS1STARTLOW and SDRC_CS_CFG[3:0] CS1STARTHIGH fields as follows:

- SDRC_CS_CFG[3:0] CS1STARTHIGH corresponds to one of the eight 128M-byte address spaces (total address space of SDRC is 1G-byte): 0x0000 for the first partition up to 0x0111 for the eights partition. The SDRC_CS_CFG[3] must always be set to 0.
- SDRC_CS_CFG[9:8] CS1STARTLOW corresponds, for a given 128M-byte address space, to one of the four 32M-byte address spaces: 0x00 for the first, up to 0x11 for the fourth.

Therefore, for a start address of 0x2000 0000 (SDRC point of view) when connecting a 2048-Mbit SDRAM memory (64M $\times$32 such as in MUX25, Table 11-97), the SDRC_CS_CFG[9:8] CS1STARTLOW bit field must be set to 0x00 (the first 32M-byte address space, as shown in Figure 11-84), and the SDRC_CS_CFG[3:0] CS1STARTHIGH bit field must be set to 0x0100 (fourth 128M-byte address space). The start address must be aligned on the memory size.

The CS1 end address (0x3000 0000) can be deduced from the CS1 size (that is, from the RAMSIZE parameter), which takes the value 0x80 when connecting a 2048-Mbit SDRAM memory.

Figure 11-84. CS Start and End Address Configuration Example



11.2.6.6 How to Choose a Suitable SDRAM

This section describes how to select a suitable SDRAM device to interface with the SDRC controller. Basically, if an SDRAM device is aligned with the JEDEC LPDDR1 SDRAM standard (JEDEC committee JC42.3), the SDRAM device is likely to be compatible with the SDRC controller. To ensure that an SDRAM device is fully compatible, see the following sections.

Section 11.2.6.6.1 discusses the SDRAM device features to consider. Section 11.2.6.6.2 discusses the SDRC controller features to consider. As an example, Section 11.2.6.6.3 lists chosen SDRAM parameters versus the supported characteristics of the SDRC controller.

11.2.6.6.1 SDRAM Device Parameters

Several settings must be considered when selecting an SDRAM device to interface with the SDRC controller:

- SDRAM type
- Operating voltage

- Maximum operating frequency
- Maximum memory size
- Memory organization:
  - Number of banks
  - Data bus width
- Burst length
- Page size
- Column address strobe (CAS) latency
- Refresh rate
- AC timing parameters and especially the access time parameter tAC

These parameters are defined in the SDRAM device datasheet and must meet the SDRC controller specifications.

11.2.6.6.2 SDRC Controller Characteristics

Keeping the SDRAM device settings in mind, the SDRC controller supports:

- Supported device type: Up to two mobile single data rate (M-SDR) SDRAMs, or up to two low-power double data rate (LPDDR) SDRAMs.
- Operating voltage: VDD = VDDQ = 1.7 V to 1.9 V; VSS = VSSQ = 0.0 V (and LVCMOS 1.8 V I/Os)
- Temperature range: -25°C to 85°C ambient
- Maximum supported operating Frequency: 166 MHz (condition: single device optimized board layout).
- Maximum supported memory size: 128M bytes per external SDRAM bank (256M- and 512M-byte SDRAM may be supported, depending on their implementation)
- Minimum supported memory size: 2M bytes
- Maximum SDRC addressing capability: 1G byte
- Memory organization:
  - Number of internal SDRAM banks: two (for 2 and 4 MB memory device only) or four banks (other memory device only)
  - Data path to external SDRAM memory: 16- or 32-bit
- Burst Length: Burst of 2 (M-SDR) and burst of 4 (LPDDR)
- Page size: Programmable value (up to 16K bytes)
- Column address strobe latency (CL): 1 to 5 system clock cycles
- Refresh intervals: Programmable value
- Major operation is 3-3-3 (CL-tRCD-tRP) conditions.
  - CL = 3
  - tRCD = 3 clocks cycle time
  - tRP = 3 clocks cycle time
  - tRRD = 2 clocks cycle time
  - tRAS = 7 clocks cycle time
  - tRC = 10 clocks cycle time

11.2.6.6.3 SDRAM Device Compatibility Verification

In this example, a mobile DDR SDRAM memory with the following characteristics is verified for compatibilty:

- Type: Mobile DDR SDRAM
- Size: 512 Mbits (16M × 32 in 4 banks)

Table 11-117 lists the SDRAM versus the SDRC controller characteristics.

Table 11-117. SDRAM vs SDRC Controller Characteristics

SDRAM Parameter	SDRC Controller	512-Mbit MDDR SDRAM (16M × 32)	Compatibility
Device type	Mobile SDR / low-power DDR	Low-power DDR	ü
Operating voltage	VDD = VDDQ = 1.7V to 1.9V; VSS = VSSQ = 0.0V (and LVCMOS 1.8V I/Os)	0.8xVDDQ ≥ High level ≤ VDDQ + 0.3V -0.3V ≤ Low level ≤ 0.2 × VDDQ 1.8V ≤ I/O power ≤ 1.95V	ü
Max operating frequency	166 MHz	166 MHz (DDR333)	ü
Memory size	Min: 16 Mbits, max: 1024M bits (2048 and 4096M bits may be supported)	512M bits	ü
Number of banks	2 (16 and 32 Mb) or 4 (other)	4	ü
Data path SDRC/SDRAM	16- and 32-bit	32-bit	ü
Burst length	Burst of 2 (M-SDR) and Burst of 4 (LPDDR)	2, 4, 8	ü
Page size	Up to 16K bytes	2K bytes	ü
CAS latency	1 to 5 clock cycles	3	ü
Refresh interval	64ms / number of rows	7.8 µs	ü

11.2.7 SMS Register Manual

11.2.7.1 SMS Instance Summary

Table 11-118 describes the SMS module instance.

Table 11-118. SMS Instance Summary

Module Name	Base Address	Size
SMS	0x6C00 0000	64K bytes

11.2.7.2 SMS Register Summary

Table 11-119 summarizes the SMS register mapping summary.

Table 11-119. SMS Register Summary

Register Name	Type	Register Width (Bits)	Address Offset	Physical Address
SMS_REVISION	R	32	0x0000 0000	0x6C00 0000
SMS_SYSCONFIG	RW	32	0x0000 0010	0x6C00 0010
SMS_SYSSTATUS	R	32	0x0000 0014	0x6C00 0014
SMS_RG_ATTi (1)	RW	32	0x0000 0048 + (0x0000 0020 * i)	0x6C00 0048 + (0x0000 0020 * i)
SMS_RG_RDPERMi (1)	RW	32	0x0000 0050 + (0x0000 0020 * i)	0x6C00 0050 + (0x0000 0020 * i)
SMS_RG_WRPERMi (1)	RW	32	0x0000 0058 + (0x0000 0020 * i)	0x6C00 0058 + (0x0000 0020 * i)
SMS_RG_STARTj (2) where k = j - 1 (3)	RW	32	0x0000 0060 + (0x0000 0020 * k)	0x6C00 0060 + (0x0000 0020 * k)
SMS_RG_ENDj (2) where k = j - 1 (3)	RW	32	0x0000 0064 + (0x0000 0020 * k)	0x6C00 0064 + (0x0000 0020 * k)
SMS_CLASS_ARBITER0	RW	32	0x0000 0150	0x6C00 0150
SMS_CLASS_ARBITER1	RW	32	0x0000 0154	0x6C00 0154
SMS_CLASS_ARBITER2	RW	32	0x0000 0158	0x6C00 0158
SMS_INTERCLASS_ARBITER	RW	32	0x0000 0160	0x6C00 0160
SMS_CLASS_ROTATIONm (4)	RW	32	0x0000 0164 + (0x0000 0004 * m)	0x6C00 0164 + (0x0000 0004 * m)
SMS_ERR_ADDR	R	32	0x0000 0170	0x6C00 0170
SMS_ERR_TYPE	RW	32	0x0000 0174	0x6C00 0174
SMS_POW_CTRL	RW	32	0x0000 0178	0x6C00 0178
SMS_ROT_CONTROLn (5)	RW	32	0x0000 0180 + (0x0000 0010 * n)	0x6C00 0180 + (0x0000 0010 * n)
SMS_ROT_SIZEn (5)	RW	32	0x0000 0184 + (0x0000 0010 * n)	0x6C00 0184 + (0x0000 0010 * n)
SMS_ROT_PHYSICAL_BAn (5)	RW	32	0x0000 0188 + (0x0000 0010 * n)	0x6C00 0188 + (0x0000 0010 * n)

(1) i = 0 to 7
(2) j = 1 to 7
(3) k = 0 to 6
(4) m = 0 to 2
(5) n = 0 to 11

11.2.7.3 SMS Register Description

This section provides a description of SMS registers.

Table 11-120. SMS_REVISION

Address Offset	0x0000 0000		
Physical Address	0x6C00 0000	**Instance**	SMS
Description	This register contains the IP revision code. IP revision code is defined at design time		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																								REV							

Bits	Field Name	Description	Type	Reset
31:8	RESERVED	Read returns 0.	R	0x000000
7:0	REV	IP revision code [7:4] Major revision [3:0] Minor revision Examples: 0x10 for 1.0, 0x21 for 2.1	R	See (1)

(1) TI internal data

Table 11-121. Register Call Summary for Register SMS_REVISION

SDRAM Controller (SDRC) Subsystem

- SMS Register Summary: [0]

Table 11-122. SMS_SYSCONFIG

Address Offset	0x0000 0010		
Physical Address	0x6C00 0010	**Instance**	SMS
Description	This register controls the various parameters of the Interconnect.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																							RESERVED	RESERVED			SIDLEMODE		RESERVED	SOFTRESET	AUTOIDLE

Bits	Field Name	Description	Type	Reset
31:9	RESERVED	Write 0s for future compatibility. Read returns 0s.	RW	0x000000
8	RESERVED	Write 0s for future compatibility. Read returns 0s.	RW	0x0
7:5	RESERVED	Write 0s for future compatibility. Read returns 0s.	RW	0x0
4:3	SIDLEMODE	Power management Req/Ack Control 0x0: Force Idle - An idle request is acknowledged unconditionally 0x1: No Idle - An idle request is never acknowledged. 0x2: Smart Idle - Acknowledgment to an idle request is based on the internal activity of the module 0x3: Reserved - Do not use.	RW	0x0
2	RESERVED	Write 0s for future compatibility. Read returns 0.	RW	0x0
1	SOFTRESET	Software reset 0x0: Normal mode (no reset applied) 0x1: Software reset is activated	RW	0x0
0	AUTOIDLE	Internal interface clock gating strategy 0x0: Interface clock is free-running 0x1: Automatic interface clock gating strategy is applied, based on the interconnect activity	RW	0x1

Table 11-123. Register Call Summary for Register SMS_SYSCONFIG

SDRAM Controller (SDRC) Subsystem

- Clocking, Reset, and Power Management Scheme: [0]
- Module Power Saving: [1] [2] [3]
- System Power Management: [4] [5] [6] [7]
- SMS Register Summary: [8]

Table 11-124. SMS_SYSSTATUS

Address Offset	0x0000 0014		
Physical Address	0x6C00 0014	**Instance**	SMS
Description	This register provides module status, excluding interrupt status info.		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																								RESERVED							RESETDONE

Bits	Field Name	Description	Type	Reset
31:8	RESERVED	Reserved for module-specific status information. Read returns 0s.	R	0x000000
7:1	RESERVED	Reserved for interconnect socket status information. Read returns 0s.	R	0x00
0	RESETDONE	Internal reset monitoring 0x0: Internal module reset is ongoing. 0x1: Reset complete. The module is ready to be used.	R	0x-

Table 11-125. Register Call Summary for Register SMS_SYSSTATUS

SDRAM Controller (SDRC) Subsystem

- SMS Register Summary: [0]

Table 11-126. SMS_RG_ATTi

Address Offset	0x0000 0048 + (0x0000 0020 * i)	**Index**	i = 0 to 7
Physical Address	0x6C00 0048 + (0x0000 0020 * i)	**Instance**	SMS
Description	Request information permission		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
REQINFO																															

Bits	Field Name	Description	Type	Reset
31:0	REQINFO	Request information permission The REQINFO field is a bit vector of permissions, one per MReqInfo encoding: NonHost/Host - User/Supervisor - Functional/Debug - Data Transfer/Opcode Fetch (1)	RW	0x--------

(1) See Table 11-99 for more information on REQINFO values

Table 11-127. Register Call Summary for Register SMS_RG_ATTi

SDRAM Controller (SDRC) Subsystem

- SDRAM Memory Scheduler: [0] [1] [2] [3] [4] [5]
- SMS Basic Programming Model: [6]
- SMS Register Summary: [7]

Table 11-128. SMS_RG_RDPERMi

Address Offset	0x0000 0050 + (0x0000 0020 * i)	**Index**	i = 0 to 7
Physical Address	0x6C00 0050 + (0x0000 0020 * i)	**Instance**	SMS
Description	This register provides the list of all initiators that have permission for reading from that memory region.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																CONNIDVECTOR															

Bits	Field Name	Description	Type	Reset
31:16	RESERVED	Write 0s for future compatibility. Read returns 0s.	RW	0x0000
15:0	CONNIDVECTOR	One bit per initiator group. Bit 0 set to 1 means that initiator whose ConnID = 0 has read permission to the protected region i.	RW	0x---- (1)

(1) Reset value exported from control module for region 0 and region 1; reset value equal to 0x0000 for other

Table 11-129. Register Call Summary for Register SMS_RG_RDPERMi

SDRAM Controller (SDRC) Subsystem

- SDRAM Memory Scheduler: [0]
- SMS Basic Programming Model: [1]
- SMS Register Summary: [2]

Table 11-130. SMS_RG_WRPERMi

Address Offset	0x0000 0058 + (0x0000 0020 * i)	**Index**	i = 0 to 7
Physical Address	0x6C00 0058 + (0x0000 0020 * i)	**Instance**	SMS
Description	This register provides the list of all initiators that have permission for writing to that memory region.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																CONNIDVECTOR															

Bits	Field Name	Description	Type	Reset
31:16	RESERVED	Write 0s for future compatibility. Read returns 0s.	RW	0x0000
15:0	CONNIDVECTOR	One bit per initiator group. Bit 0 set to 1 means that initiator whose ConnID = 0 has write permission to the protected region i.	RW	0x---- (1)

(1) Reset value exported from control module for region 0 and region 1; reset value equal to 0x0000 for other

Table 11-131. Register Call Summary for Register SMS_RG_WRPERMi

SDRAM Controller (SDRC) Subsystem

- SDRAM Memory Scheduler: [0]
- SMS Basic Programming Model: [1]
- SMS Register Summary: [2]

Table 11-132. SMS_RG_STARTj

Address Offset	0x0000 0060 + (0x0000 0020 * (k))	**Index**	j = 1 to 7 and k = 0 to 8
Physical Address	0x6C00 0060 + (0x0000 0020 * (k))	**Instance**	SMS
Description	This register provides the region #j start address (lowest address inside the region), with a 64-KB granularity.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								STARTADDRESS								RESERVED															

Bits	Field Name	Description	Type	Reset
31	RESERVED	Write 0s for future compatibility. Read returns 0.	RW	0x0
30:16	STARTADDRESS	Region #j start address (included in the region) Aligned on 64-KB boundary. [15:0] must be written with 0s. No STARTADDRESS parameter for region 0.	RW	0x----
15:0	RESERVED	Write 0s for future compatibility. Read returns 0.	RW	0x0000

Table 11-133. Register Call Summary for Register SMS_RG_STARTj

SDRAM Controller (SDRC) Subsystem

- SMS Basic Programming Model: [0] [1]
- SMS Register Summary: [2]

Table 11-134. SMS_RG_ENDj

Address Offset	0x0000 0064 + (0x0000 0020 * (K))	**Index**	j = 1 to 7 and k = 0 to 8
Physical Address	0x6C00 0064 + (0x0000 0020 * (K))	**Instance**	SMS
Description	This register provides the region #j end address (lowest address outside the region), with a 64-KB granularity.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								ENDADDRESS								RESERVED															

Bits	Field Name	Description	Type	Reset
31	RESERVED	Write 0s for future compatibility. Read returns 0s.	RW	0x0
30:16	ENDADDRESS	Region #j end address (not included in the region) Aligned on 64-KB boundary. [15:0] must be written with 0s. No ENDADDRESS parameter for region 0.	RW	0x----
15:0	RESERVED	Write 0s for future compatibility. Read returns 0s.	RW	0x0000

Table 11-135. Register Call Summary for Register SMS_RG_ENDj

SDRAM Controller (SDRC) Subsystem

- SMS Basic Programming Model: [0] [1]
- SMS Register Summary: [2]

Table 11-136. SMS_CLASS_ARBITER0

Address Offset	0x0000 0150		
Physical Address	0x6C00 0150	**Instance**	SMS
Description	This register controls the arbitration parameters between the class 0 request groups.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BURST-COMPLETE		RESERVED						EXTENDEDGRANT				RESERVED												HIGHPRIOVECTOR		RESERVED					

Bits	Field Name	Description	Type	Reset
31:30	BURST-COMPLETE	Delayed service until burst request complete BurstComplete[k], k= 6 to 7 (BURST-COMPLETE[30] is for group number 6, BURST-COMPLETE[31] is for group number 7) 0x0: Group #k request to arbiter issued as soon as the first burst request is available 0x1: Group #k request to arbiter delayed until a complete burst transaction is buffered	RW	0x0
29:24	RESERVED	Write 0s for future compatibility. Read returns 0s.	RW	0x00
23:20	EXTENDEDGRANT	Extended grant service inside a class Vector specifying the number of consecutive services a group is granted. 2 bits per group ExtendedGrant[2*k+1,2*k], k = 6 to 7 (EXTENDEDGRANT[21:20] is for group number 6, EXTENDEDGRANT[23:22] is for group number 7) 0x1: 1 service for group #k when granted 0x2: 2 services for group #k when granted 0x3: 3 services for group #k when granted	RW	0x5
19:8	RESERVED	Write 0s for future compatibility. Read returns 0s.	RW	0x000
7:6	HIGHPRIOVECTOR	High-priority attribute inside a class Vector allocating a higher priority to one of the class members. A single group may be given this attribute at a time. HighPrioVector[k], k= 6 to 7 (HIGHPRIOVECTOR[6] is for group number 6, HIGHPRIOVECTOR[7] is for group number 7) 0x0: Group #k has standard priority (LRU based). 0x1: Group #k has the highest priority over all other class members.	RW	0x0
5:0	RESERVED	Write 0s for future compatibility. Read returns 0s.	RW	0x00

Table 11-137. Register Call Summary for Register SMS_CLASS_ARBITER0

SDRAM Controller (SDRC) Subsystem

- SDRAM Memory Scheduler: [0] [1] [2]
- SMS Basic Programming Model: [3]
- SMS Register Summary: [4]

Table 11-138. SMS_CLASS_ARBITER1

Address Offset	0x0000 0154		
Physical Address	0x6C00 0154	**Instance**	SMS
Description	This register controls the arbitration parameters between the class 1 request groups.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED						BURST-COMPLETE		RESERVED												EXTENDEDGRANT				RESERVED						HIGHPRIOVECTOR	

Bits	Field Name	Description	Type	Reset
31:26	RESERVED	Write 0s for future compatibility. Read returns 0s.	RW	0x00
25:24	BURST-COMPLETE	Delayed service until burst request complete BurstComplete[k], k= 0 to 1 (BURST-COMPLETE[24] is for group number 0, BURST-COMPLETE[25] is for group number 1) 0x0: Group #k request to arbiter issued as soon as the first burst request is available 0x1: Group #k request to arbiter delayed until a complete burst transaction is buffered	RW	0x0
23:12	RESERVED	Write 0s for future compatibility. Read returns 0s.	RW	0x000
11:8	EXTENDEDGRANT	Extended grant service inside a class Vector specifying the number of consecutive services a group is granted. 2 bits per group ExtendedGrant[2*k+1,2*k], k = 0 to 1 (EXTENDEDGRANT[9:8] is for group number 0, EXTENDEDGRANT[11:10] is for group number 1) 0x1: 1 service for group #k when granted 0x2: 2 services for group #k when granted 0x3: 3 services for group #k when granted	RW	0x5
7:2	RESERVED	Write 0s for future compatibility. Read returns 0s.	RW	0x00
1:0	HIGHPRIOVECTOR	High-priority attribute inside a class Vector allocating a higher priority to one of the class members. A single group may be given this attribute at a time. HighPrioVector[k], k= 0 to 1 (HIGHPRIOVECTOR[1] is for group number 1, HIGHPRIOVECTOR[0] is for group number 0) 0x0: Group #k has standard priority (LRU based). 0x1: Group #k has the highest priority over all other class members.	RW	0x0

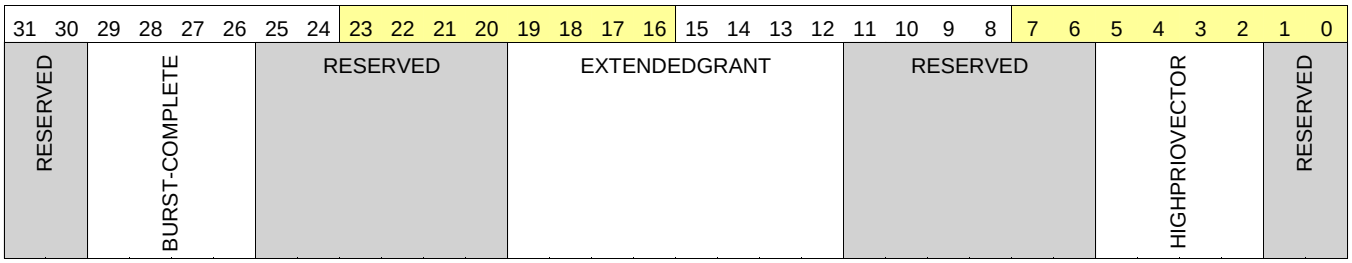
Table 11-139. Register Call Summary for Register SMS_CLASS_ARBITER1

SDRAM Controller (SDRC) Subsystem

- SDRAM Memory Scheduler: [0] [1]
- SMS Mode of Operation: [2]
- SMS Register Summary: [3]

Table 11-140. SMS_CLASS_ARBITER2

Address Offset	0x0000 0158		
Physical Address	0x6C00 0158	**Instance**	SMS
Description	This register controls the arbitration parameters between the class 2 request groups.		
Type	RW		



Bits	Field Name	Description	Type	Reset
31:30	RESERVED	Write 0s for future compatibility. Read returns 0s.	RW	0x0
29:26	BURST-COMPLETE	Delayed service until burst request complete BurstComplete[k], k= 2 to 5 (BURST-COMPLETE[29] is for group number 5, BURST-COMPLETE[28] is for group number 4, BURST-COMPLETE[27] is for group number 3, BURST-COMPLETE[26] is for group number 2) 0x0: Group #k request to arbiter issued as soon as the first burst request is available 0x1: Group #k request to arbiter delayed until a complete burst transaction is buffered	RW	0x0
25:20	RESERVED	Write 0s for future compatibility. Read returns 0s.	RW	0x00
19:12	EXTENDEDGRANT	Vector specifying the number of consecutive services a group is granted. 2 bits per group ExtendedGrant[2*k+1,2*k], k = 2 to 5 (EXTENDEDGRANT[19:18] is for group number 5, EXTENDEDGRANT[17:16] is for group number 4, EXTENDEDGRANT[15:14] is for group number 3, EXTENDEDGRANT[13:12] is for group number 2) 0x1: 1 service for group #k when granted 0x2: 2 services for group #k when granted 0x3: 3 services for group #k when granted	RW	0x55
11:6	RESERVED	Write 0s for future compatibility. Read returns 0s.	RW	0x00
5:2	HIGHPRIOVECTOR	Vector allocating a higher priority to one of the class members. A single group may be given this attribute at a time. HighPrioVector[k], k= 2 to 5 (HIGHPRIOVECTOR[5] is for group number 5, HIGHPRIOVECTOR[4] is for group number 4, HIGHPRIOVECTOR[3] is for group number 3, HIGHPRIOVECTOR[2] is for group number 2) 0x0: Group #k has standard priority (LRU based). 0x1: Group #k has the highest priority over all other class members.	RW	0x0
1:0	RESERVED	Write 0s for future compatibility. Read returns 0s.	RW	0x0

Table 11-141. Register Call Summary for Register SMS_CLASS_ARBITER2

SDRAM Controller (SDRC) Subsystem

- SDRAM Memory Scheduler: [0] [1]
- SMS Basic Programming Model: [2]
- SMS Mode of Operation: [3]
- SMS Register Summary: [4]

Table 11-142. SMS_INTERCLASS_ARBITER

Address Offset	0x0000 0160		
Physical Address	0x6C00 0160	**Instance**	SMS
Description	This register controls the PWM counter that defines the priority alternation between class 1 and class 2.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								CLASS2PRIO								RESERVED								CLASS1PRIO							

Bits	Field Name	Description	Type	Reset
31:24	RESERVED	Write 0s for future compatibility. Read returns 0s.	RW	0x00
23:16	CLASS2PRIO	Class 2 high-priority window width (clock cycle count). Do not set to 0x00.	RW	0x40
15:8	RESERVED	Write 0s for future compatibility. Read returns 0s.	RW	0x00
7:0	CLASS1PRIO	Class 1 high-priority window width (clock cycle count). Do not set to 0x00.	RW	0x40

Table 11-143. Register Call Summary for Register SMS_INTERCLASS_ARBITER

SDRAM Controller (SDRC) Subsystem

- SMS Mode of Operation: [0] [1]
- SMS Register Summary: [2]

Table 11-144. SMS_CLASS_ROTATIONm

Address Offset	0x0000 0164 + (0x0000 0004 * m)	**Index**	m = 0 to 2
Physical Address	0x6C00 0164 + (0x0000 0004 * m)	**Instance**	SMS
Description	This register controls the number of consecutive services that is allocated to a thread whose transactions have been split by the rotation engine.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																											NOFSERVICES				

Bits	Field Name	Description	Type	Reset
31:5	RESERVED	Write 0s for future compatibility. Read returns 0s.	RW	0x0000000
4:0	NOFSERVICES	Number of RE split transactions serviced consecutively when the thread gets granted by the arbitration logic.	RW	0x01

Table 11-145. Register Call Summary for Register SMS_CLASS_ROTATIONm

SDRAM Controller (SDRC) Subsystem

- SDRAM Memory Scheduler: [0]
- SMS Mode of Operation: [1]
- SMS Register Summary: [2]

Table 11-146. SMS_ERR_ADDR

Address Offset	0x0000 0170		
Physical Address	0x6C00 0170	**Instance**	SMS
Description	This register captures the address of an access that has generated an error.		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ERRORADDRESS																															

Bits	Field Name	Description	Type	Reset
31:0	ERRORADDRESS	Access address that has generated an error (bit 31 is always 0)	R	0x00000000

Table 11-147. Register Call Summary for Register SMS_ERR_ADDR

SDRAM Controller (SDRC) Subsystem

- SMS Basic Programming Model: [0]
- SMS Register Summary: [1]

Table 11-148. SMS_ERR_TYPE

Address Offset	0x0000 0174		
Physical Address	0x6C00 0174	**Instance**	SMS
Description	This register provides additional information about the access that has generated the error.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED					ERRORREGIONID			RESERVED	ERRORMCMD			ERRORCONNID				RESERVED					UNEXPECTEDADD	UNEXPECTEDREQ	ILLEGALCMD	RESERVED				ERRORSECOVERLAP	RESERVED	RESERVED	ERRORVALID

Bits	Field Name	Description	Type	Reset
31:27	RESERVED	Write 0s for future compatibility. Read returns 0s.	RW	0x00
26:24	ERRORREGIONID	ID of the region that has been illegally accessed 0x0: Region 0 0x1: Region 1 ... 0x7: Region 7 0x8 to 0xF: reserved	R	0x0
23	RESERVED	Write 0s for future compatibility. Read returns 0s.	RW	0x0
22:20	ERRORMCMD	Interconnect command that caused the error	R	0x0
19:16	ERRORCONNID	Identifies the illegal access initiator interconnect ConnID of the illegal access initiator. Refer to the top level documentation of the device using the SMS module	R	0x0
15:11	RESERVED	Write 0s for future compatibility. Read returns 0s.	RW	0x00

Bits	Field Name	Description	Type	Reset
10	UNEXPECTEDADD	Request targeting non-defined rotation contexts (such as contexts 12, 13, 14, or 15) or non-defined L3 Interconnect request signals used for context number decoding. Read 0x0: No unexpected request received Write 0x0: No effect Read 0x1: A request has been received on the interconnect with address decoding targeting rotation contexts 12, 13, 14, or 15, or a signal used for context number decoding was undefined. Write 0x1: Clear UNEXPECTEDADD bit.	RW	0x0
9	UNEXPECTEDREQ	Unexpected request received during SMS idle state Read 0x0: No unexpected request received Write 0x0: No effect Read 0x1: A request has been received on the interconnect after the SMS was put in idle mode by the system power manager. Write 0x1: Clear the UNEXPECTEDREQ bit field.	RW	0x0
8	ILLEGALCMD	Illegal command on the L3 interface Read 0x0: No illegal command received Write 0x0: No effect Read 0x1: Illegal command has been received. Write 0x1: Clear ILLEGALCMD bit field.	RW	0x0
7:4	RESERVED	Write 0s for future compatibility. Read returns 0s.	RW	0x0
3	ERRORSECOVERLAP	Protection region overlapping error Read 0x0: No overlap violation detected Write 0x0: No effect Read 0x1: A protection region overlap violation has been detected. Write 0x1: Clear ERRORSECOVERLAP bit field.	RW	0x0
2	RESERVED	Reserved for non-GP devices.	RW	0x0
1	RESERVED	Reserved for non-GP devices.	RW	0x0
0	ERRORVALID	Error validity status - Must be explicitly cleared with a write transaction Read 0x0: No effect Write 0x0: All error fields no longer valid Read 0x1: Error detected and logged in the other error fields Write 0x1: Clear ERRORVALID bit field	RW	0x0

Table 11-149. Register Call Summary for Register SMS_ERR_TYPE

SDRAM Controller (SDRC) Subsystem

- SDRAM Memory Scheduler: [0]
- SMS Basic Programming Model: [1] [2]
- SMS Register Summary: [3]

Table 11-150. SMS_POW_CTRL

Address Offset	0x0000 0178		
Physical Address	0x6C00 0178	**Instance**	SMS
Description	This register controls the SMS power management in conjunction with the regular interconnect socket registers.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																								IDLEDELAY							

Bits	Field Name	Description	Type	Reset
31:8	RESERVED	Write 0s for future compatibility. Read returns 0s.	RW	0x000000
7:0	IDLEDELAY	Delay (expressed in L3 clock cycle units) before autoidle, that is, before disabling the SMS functional clock when no more traffic in the SMS module.	RW	0x80

Table 11-151. Register Call Summary for Register SMS_POW_CTRL

SDRAM Controller (SDRC) Subsystem

- Module Power Saving: [0]
- SMS Register Summary: [1]

Table 11-152. SMS_ROT_CONTROLn

Address Offset	0x0000 0180 + (0x0000 0010 * n)	**Index**	n = 0 to 11
Physical Address	0x6C00 0180 + (0x0000 0010 * n)	**Instance**	SMS
Description	This register configures the virtual rotated frame buffer module for context #n.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																					PH			RESERVED	PW			RESERVED		PS	

Bits	Field Name	Description	Type	Reset
31:11	RESERVED	Write 0s for future compatibility. Read returns 0s.	RW	0x000000
10:8	PH	Exponent based 2 value, 2^{ph} indicates the page height in bytes for context #n.	RW	0x0
7	RESERVED	Write 0s for future compatibility. Read returns 0s.	RW	0x0
6:4	PW	Exponent based 2 value, 2^{pw} indicates the page width in bytes for context #n.	RW	0x0
3:2	RESERVED	Write 0s for future compatibility. Read returns 0s.	RW	0x0
1:0	PS	Exponent based 2 value, 2^{ps} indicates the pixel size in bytes for context #n. A value of 3 is invalid.	RW	0x0

Table 11-153. Register Call Summary for Register SMS_ROT_CONTROLn

SDRAM Controller (SDRC) Subsystem

- SDRAM Memory Scheduler: [0] [1]
- SMS Basic Programming Model: [2] [3] [4]
- How to Program the VRFB: [5] [6] [7]
- Camcorder Use Case: How to Configure the VRFB: [8] [9] [10] [11] [12] [13] [14] [15]
- SMS Register Summary: [16]

Table 11-154. SMS_ROT_SIZEn

Address Offset	0x0000 0184 + (0x0000 0010 * n)	**Index**	n = 0 to 11
Physical Address	0x6C00 0184 + (0x0000 0010 * n)	**Instance**	SMS
Description	This register configures the bank organization for context #n.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								IMAGEHEIGHT								RESERVED								IMAGEWIDTH							

Bits	Field Name	Description	Type	Reset
31:27	RESERVED	Write 0s for future compatibility. Read returns 0s.	RW	0x00
26:16	IMAGEHEIGHT	Image height in pixels for context #n	RW	0x000
15:11	RESERVED	Write 0s for future compatibility. Read returns 0s.	RW	0x00
10:0	IMAGEWIDTH	Image width in pixels for context #n	RW	0x000

Table 11-155. Register Call Summary for Register SMS_ROT_SIZEn

SDRAM Controller (SDRC) Subsystem

- SDRAM Memory Scheduler: [0] [1] [2]
- SMS Basic Programming Model: [3] [4]
- How to Program the VRFB: [5] [6]
- Camcorder Use Case: How to Configure the VRFB: [7] [8] [9] [10] [11] [12]
- SMS Register Summary: [13]

Table 11-156. SMS_ROT_PHYSICAL_BAn

Address Offset	0x0000 0188 + (0x0000 0010 * n)	**Index**	n = 0 to 11
Physical Address	0x6C00 0188 + (0x0000 0010 * n)	**Instance**	SMS
Description	This register allows to configure the physical base address for context #n.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED	PHYSICALBA																														

Bits	Field Name	Description	Type	Reset
31	RESERVED	Write 0s for future compatibility. Read returns 0s.	RW	0x0
30:0	PHYSICALBA	Physical base address of the frame buffer for context #n in SDRAM	RW	0x00000000

Table 11-157. Register Call Summary for Register SMS_ROT_PHYSICAL_BAn

SDRAM Controller (SDRC) Subsystem

- SDRAM Memory Scheduler: [0]
- SMS Basic Programming Model: [1]
- How to Program the VRFB: [2]
- Camcorder Use Case: How to Configure the VRFB: [3] [4]
- Understanding SDRAM Subsystem Address Spaces: [5]
- SMS Register Summary: [6]

11.2.8 SDRC Register Manual

11.2.8.1 SDRC Instance Summary

Table 11-158 gives some details on the SDRC module instance.

Table 11-158. SDRC Instance Summary

Module Name	Base Address	Size
SDRC	0x6D00 0000	64K bytes

11.2.8.2 SDRC Register Summary

Table 11-159 summarizes the SDRC register mapping.

Table 11-159. SDRC Register Summary

Register Name	Type	Register Width (Bits)	Address Offset	Physical Address
SDRC_REVISION	R	32	0x0000 0000	0x6D00 0000
SDRC_SYSCONFIG	RW	32	0x0000 0010	0x6D00 0010
SDRC_SYSSTATUS	R	32	0x0000 0014	0x6D00 0014
SDRC_CS_CFG	RW	32	0x0000 0040	0x6D00 0040
SDRC_SHARING	RW	32	0x0000 0044	0x6D00 0044
SDRC_ERR_ADDR	R	32	0x0000 0048	0x6D00 0048
SDRC_ERR_TYPE	RW	32	0x0000 004C	0x6D00 004C
SDRC_DLLA_CTRL	RW	32	0x0000 0060	0x6D00 0060
SDRC_DLLA_STATUS	R	32	0x0000 0064	0x6D00 0064
SDRC_POWER_REG	RW	32	0x0000 0070	0x6D00 0070
SDRC_MCFG_p (1)	RW	32	0x0000 0080 + (0x0000 0030 * p)	0x6D00 0080 + (0x0000 0030 * p)
SDRC_MR_p (1)	RW	32	0x0000 0084 + (0x0000 0030 * p)	0x6D00 0084 + (0x0000 0030 * p)
SDRC_EMR2_p (1)	RW	32	0x0000 008C + (0x0000 0030 * p)	0x6D00 008C + (0x0000 0030 * p)
SDRC_ACTIM_CTRLA_p (1)	RW	32	0x0000 009C + (0x0000 0028 * p)	0x6D00 009C + (0x0000 0028 * p)
SDRC_ACTIM_CTRLB_p (1)	RW	32	0x0000 00A0 + (0x0000 0028 * p)	0x6D00 00A0 + (0x0000 0028 * p)
SDRC_RFR_CTRL_p (1)	RW	32	0x0000 00A4 + (0x0000 0030 * p)	0x6D00 00A4 + (0x0000 0030 * p)
SDRC_MANUAL_p (1)	RW	32	0x0000 00A8 + (0x0000 0030 * p)	0x6D00 00A8 + (0x0000 0030 * p)

(1) p = 0 to 1.

11.2.8.3 SDRC Register Description

This section provides a description of SDRC registers.

Table 11-160. SDRC_REVISION

Address Offset	0x0000 0000		
Physical Address	0x6D00 0000	**Instance**	SDRC
Description	This register contains the IP revision code. This code is specified at design time.		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																								REV							

Bits	Field Name	Description	Type	Reset
31:8	RESERVED	Read returns 0.	R	0x000000
7:0	REV	IP revision code [7:4]: Major revision [3:0]: Minor revision Examples: 0x10 for 1.0, 0x21 for 2.1	R	See (1)

(1) TI internal data

Table 11-161. Register Call Summary for Register SDRC_REVISION

SDRAM Controller (SDRC) Subsystem

- SDRC Configuration: [0]
- SDRC Register Summary: [1]

Table 11-162. SDRC_SYSCONFIG

Address Offset	0x0000 0010		
Physical Address	0x6D00 0010	**Instance**	SDRC
Description	This register controls the various parameters of the interconnect.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																							NOMEMORYMRS	RESERVED			IDLEMODE		RESERVED	SOFTRESET	RESERVED

Bits	Field Name	Description	Type	Reset
31:9	RESERVED	Write 0s for future compatibility Read returns 0.	RW	0x000000
8	NOMEMORYMRS	No external memory MRS command 0x0: When set to 0, the SDRC internal SDRC_MR_p and SDRC_EMR2_p registers (both CS) are written and MR, EMR2 commands are performed to the corresponding registers of the external SDRAM. 0x1: When set to 1, only SDRC internal SDRC_MR_p and SDRC_EMR2_p registers (both CS) are written, no MR or EMR2 commands are performed to SDRAM.	RW	0x0
7:5	RESERVED	Write 0s for future compatibility. Read returns 0.	RW	0x0

Bits	Field Name	Description	Type	Reset
4:3	IDLEMODE	Power management Req/Ack control 0x0: Reserved - Do not use. 0x1: Reserved - Do not use. 0x2: Smart Idle - Acknowledgment to an idle request is based on the internal activity of the module. Issued when the SDRC enters self-refresh. 0x3: Reserved - Do not use.	RW	0x2
2	RESERVED	Write 0s for future compatibility. Read returns 0	RW	0x0
1	SOFTRESET	Software reset 0x0: Normal mode (no reset applied) 0x1: Software reset activated	RW	0x0
0	RESERVED	Write 0s for future compatibility. Read returns 0.	RW	0x0

Table 11-163. Register Call Summary for Register SDRC_SYSCONFIG

SDRAM Controller (SDRC) Subsystem

- Clocking, Reset, and Power Management Scheme: [0]
- SDRC: [1]
- SDRC Configuration: [2]
- SDRC Setup: [3]
- SDRC Register Summary: [4]

Table 11-164. SDRC_SYSSTATUS

Address Offset	0x0000 0014		
Physical Address	0x6D00 0014	**Instance**	SDRC
Description	This register provides module status, excluding interrupt status info.		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																								RESERVED							RESETDONE

Bits	Field Name	Description	Type	Reset
31:8	RESERVED	Reserved for module-specific status information Read returns 0.	R	0x000000
7:1	RESERVED	Reserved for interconnect socket status information Read returns 0.	R	0x00
0	RESETDONE	Internal reset monitoring 0x0: Internal module reset is ongoing 0x1: Reset completed - The module is ready to be used	R	0x-

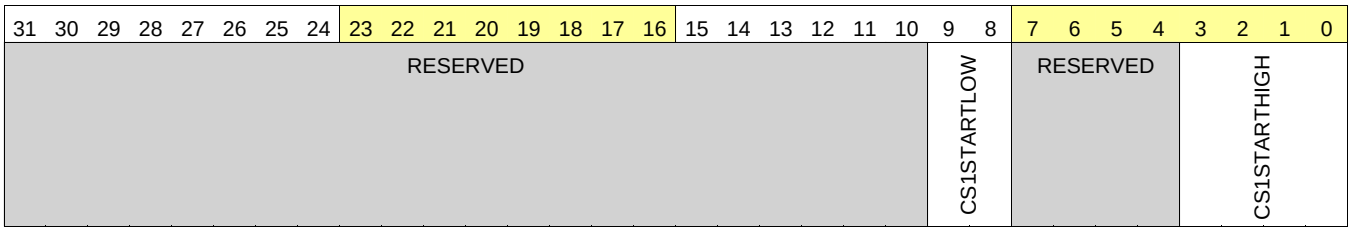
Table 11-165. Register Call Summary for Register SDRC_SYSSTATUS

SDRAM Controller (SDRC) Subsystem

- SDRC Configuration: [0]
- SDRC Register Summary: [1]

Table 11-166. SDRC_CS_CFG

Address Offset	0x0000 0040		
Physical Address	0x6D00 0040	**Instance**	SDRC
Description	This register configures the start address of CS1 address space. Must be aligned on a boundary that is a multiple of the size of the attached memory, or of the next power of two if the memory size is not a power of two.		
Type	RW		



Bits	Field Name	Description	Type	Reset
31:10	RESERVED	Write 0s for future compatibility. Read returns 0.	RW	0x000000
9:8	CS1STARTLOW	CS1 address space start address (lower add bits a1:a0) / 32MB unit	RW	0x0
7:4	RESERVED	Write 0s for future compatibility. Read returns 0.	RW	0x0
3:0	CS1STARTHIGH	CS1 address space start address (upper add bits a5:a4:a3:a2) / 128MB unit	RW	0x4

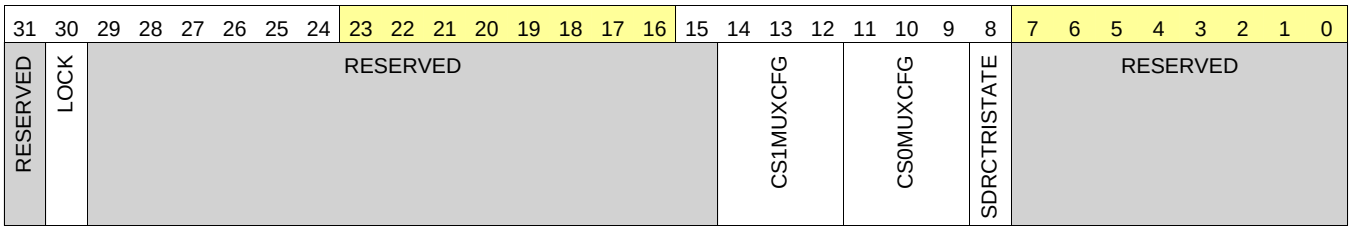
Table 11-167. Register Call Summary for Register SDRC_CS_CFG

SDRAM Controller (SDRC) Subsystem

- SDRC: [0] [1] [2] [3] [4]
- SDRC Setup: [5] [6]
- Understanding SDRAM Subsystem Address Spaces: [7] [8] [9] [10] [11] [12] [13]
- SDRC Register Summary: [14]

Table 11-168. SDRC_SHARING

Address Offset	0x0000 0044		
Physical Address	0x6D00 0044	**Instance**	SDRC
Description	This register specifies the SDRC attached memory size and position on the SDRC IOs.		
Type	RW		



Bits	Field Name	Description	Type	Reset
31	RESERVED	Write 0s for future compatibility. Read returns 0.	RW	0x0
30	LOCK	Read-only access lock bit 0x0: This register is fully writable. 0x1: When this bit is set, the register can not be unset until next reset of the module.	RW	See (1)

(1) Reset value is copied from the system control module. See the note in Section 11.2.5.3.2 and Section 7.4.9, *SDRC Registers*, in Chapter 7, *System Control Module*.

Bits	Field Name	Description	Type	Reset
29:15	RESERVED	Write 0s for future compatibility. Read returns 0s.	RW	See (1)
14:12	CS1MUXCFG	Identifies the SDRC pins used by CS1 0x0: 32-bit SDRAM on Datalane[31:0] 0x1: 32-bit SDRAM on Datalane[31:0] 0x2: 16-bit SDRAM on Datalane[31:16] 0x3: 16-bit SDRAM on Datalane[16:0] 0x4: Reserved 0x5: Reserved 0x6: Reserved 0x7: 16-bit SDRAM on Datalane[31:16]	RW	See (1)
11:9	CS0MUXCFG	Identifies the SDRC pins used by CS0 0x0: 32-bit SDRAM on Datalane[31:0] 0x1: 32-bit SDRAM on Datalane[31:0] 0x2: 16-bit SDRAM on Datalane[31:16] 0x3: 16-bit SDRAM on Datalane[16:0] 0x4: Reserved 0x5: Reserved 0x6: Reserved 0x7: 16-bit SDRAM on Datalane[31:16]	RW	See (1)
8	SDRCTRISTATE	Static 3-state command for the SDRC I/O pads 0x0: All SDRC interface pins are set to hi-Z. 0x1: Normal mode: SDRC drives the I/O pads based on the memory traffic.	RW	See (1)
7:0	RESERVED	Write 0s for future compatibility. Read returns 0s.	RW	See (1)

Table 11-169. Register Call Summary for Register SDRC_SHARING

SDRAM Controller (SDRC) Subsystem

- SDRC: [0] [1] [2] [3] [4] [5] [6]
- SDRC Setup: [7] [8]
- SDRC Register Summary: [9]

Table 11-170. SDRC_ERR_ADDR

Address Offset	0x0000 0048		
Physical Address	0x6D00 0048	**Instance**	SDRC
Description	This register captures the address of the last illegal access received on the interconnect.		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ERRORADDRESS																															

Bits	Field Name	Description	Type	Reset
31:0	ERRORADDRESS	Address of illegal access (Bit 31 is always 0.)	R	0x00000000

Table 11-171. Register Call Summary for Register SDRC_ERR_ADDR

SDRAM Controller (SDRC) Subsystem

- Error Management: [0]
- SDRC Register Summary: [1]

Table 11-172. SDRC_ERR_TYPE

Address Offset	0x0000 004C		
Physical Address	0x6D00 004C	**Instance**	SDRC
Description	This register provides additional information about the last illegal access.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																				ERRORCONNID				RESERVED	ERRORMCMD			ERRORADD		ERRORDPD	ERRORVALID

Bits	Field Name	Description	Type	Reset
31:12	RESERVED	Write 0s for future compatibility. Read returns 0.	RW	0x00000
11:8	ERRORCONNID	Identifies the interconnect ConnID of the illegal access initiator: Refer to the top level documentation of the device using the SDRC module.	RW	0x0
7	RESERVED	Write 0 for future compatibility. Read returns 0.	RW	0x0
6:4	ERRORMCMD	System command of the transaction that caused the error (3-bit field)	RW	0x0
3:2	ERRORADD	Flag that indicates access is to an illegal address Read 0x0: The system request was to an address outside the memory space. Write 0x0: Clear ErrorAdd bit field. Read 0x1: The system request was to an address outside the register space. Write 0x1: No effect Read 0x2: No Err Add. Not an address error Write 0x2: No effect Read 0x3: No Err Add. Not an address error Write 0x3: No effect	RW	0x2
1	ERRORDPD	Transaction error while the memory is in deep-power-down mode 0x0: The memory was not in deep-power-down mode when the error occurred. 0x0: No effect 0x1: The error is due to an unexpected access while the memory was in deep-power-down mode. 0x1: Clear the ErrorDPD bit field.	RW	0x0
0	ERRORVALID	Error validity status - Must be explicitly cleared with a write 0 transaction. 0x0: All error fields no longer valid 0x0: Clear ErrorValid bit field 0x1: Error detected and logged in the other error fields 0x1: No effect	RW	0x0

Table 11-173. Register Call Summary for Register SDRC_ERR_TYPE

SDRAM Controller (SDRC) Subsystem

- Error Management: [0] [1] [2] [3] [4]
- SDRC Register Summary: [5]

Table 11-174. SDRC_DLLA_CTRL

Address Offset	0x0000 0060		
Physical Address	0x6D00 0060	**Instance**	SDRC
Description	This register controls the SDRC DLL A resource used for fine timing tuning on a double-data-rate interface.		
Type	RW		

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10 9 8	7	6	5	4	3	2	1	0
FIXEDDELAY	MODEFIXEDDELAYINITLAT	RESERVED	WRITEDDRCLKX2DIS	DLLMODEONIDLEREQ		DLLIDLE	ENADLL	LOCKDLL	DLLPHASE	RESERVED

Bits	Field Name	Description	Type	Reset
31:24	FIXEDDELAY	Phase offset value in ModeFixedDelay mode. Maximum frequency supported in this mode is 83 MHz. FIXEDDELAY steps are defined after DLL cell characterization.	RW	0x00
23:16	MODEFIXEDDELAYINITLAT	Initial latency before first request to be processed in ModeFixedDelay mode[(1)] 0x0: no initial latency before first access 0x1: 2 clock cycles before first access 0x2: 4 clock cycles before first access 0x3: 6 clock cycles before first access 0x4 : 8 clock cycles before first access ... 0xFF: 510 clock cycles before first access	RW	0x00
15:8	RESERVED	Write 0s for future compatibility. Read returns 0.	RW	0x00
7	WRITEDDRCLKX2DIS	Disable MDDR write path using the double frequency input clock. 0x0: MDDR write path using the double frequency clock logic 0x1: MDDR write path using the DLL/CDL write path logic	RW	0x0
6:5	DLLMODEONIDLEREQ	Selects the DLL mode upon hardware idle request 0x0: DLL in Power-down mode upon hardware idle request 0x1: DLL in DLL idle mode upon hardware idle request 0x2: No action upon hardware idle request. Input clock frequency must not be changed. 0x3: Reserved for future use (no action upon hardware idle request).	RW	0x0
4	DLLIDLE	Enables the DLL Idle mode 0x0: DLL Idle mode disabled 0x1: DLL Idle mode enabled	RW	0x0
3	ENADLL	Enables DLL 0x0: DLL disabled 0x1: DLL enabled	RW	0x0

(1) The DLL characterization shows that this feature is not useful and that this value must be left at 0x0.

Bits	Field Name	Description	Type	Reset
2	LOCKDLL	Selects the DLL functionality between the TrackingDelay mode (previously called lock mode) or the ModeFixedDelay mode (previously called unlock mode). The DLL mode is updated in the DLL cell after: - DLLIDLE mode - DLL power-down mode 0x0: LOCKDLL at 0 puts the DLL in TrackingDelay mode (tracking counter started). 0x1: LOCKDLL at 1 puts the DLL in ModeFixedDelay mode. The fixed delay defined in FIXEDDELAY bit field is used to delay DQS lines for read accesses.	RW	0x0
1	DLLPHASE	Nominal digitally-controlled delay when DLL is enabled 0x0: 72 degrees (20% of the clock period) 0x1: 90 degrees (25% of the clock cycle)	RW	0x0
0	RESERVED	Write 0s for future compatibility. Reads return zero.	RW	0x0

Table 11-175. Register Call Summary for Register SDRC_DLLA_CTRL

SDRAM Controller (SDRC) Subsystem

- Clocking, Reset, and Power Management Scheme: [0]
- SDRC: [1] [2]
- SDRC Setup: [3] [4] [5] [6] [7] [8]
- Manual Software Commands: [9] [10]
- Typical SDRC connection to an External SDRAM Device: [11] [12] [13] [14] [15]
- SDRC Register Summary: [16]

Note:

Table 11-176. SDRC_DLLA_STATUS

Address Offset	0x0000 0064		
Physical Address	0x6D00 0064	**Instance**	SDRC
Description	This register reflects the current status of the DLL A.		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																													LOCKSTATUS	RESERVED	RESERVED

Bits	Field Name	Description	Type	Reset
31:3	RESERVED	Reads return zeros.	R	0x00000000
2	LOCKSTATUS	DLL lock status 0x0: The DLL is not locked. 0x1: The DLL is locked.	R	0x0
1	RESERVED	Reads return zero.	R	0x0
0	RESERVED	Reads return zero.	R	0x0

Table 11-177. Register Call Summary for Register SDRC_DLLA_STATUS

SDRAM Controller (SDRC) Subsystem

- SDRC: [0]
- SDRC Register Summary: [1]

Table 11-178. SDRC_POWER_REG

Address Offset	0x0000 0070		
Physical Address	0x6D00 0070	**Instance**	SDRC
Description	This SDRC power-management register defines the global power-management policy (shared by CS0/CS1).		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED					WAKEUPPROC	RESERVED		AUTOCOUNT																SRFRONRESET	SRFRONIDLEREQ	CLKCTRL		EXTCLKDIS	PWDENA	RESERVED	PAGEPOLICY

Bits	Field Name	Description	Type	Reset
31:27	RESERVED	Write 0s for future compatibility. Read returns 0.	RW	0x00
26	WAKEUPPROC	Select if after a SDRC wakeup in DDR mode (in DLL tracking-delay mode), the first request is stalled during 500 cycles latency or until the lock signal from the DLL/CDL analog cell is asserted. 0x0: SDRC allows DDR access after 500 L3 clock cycles. 0x1: SDRC allows DDR access as soon as DLL LOCKSTATUS bit is 1.	RW	0x0
25:24	RESERVED	Write 0s for future compatibility. Read returns 0.	RW	0x0
23:8	AUTOCOUNT	16-bit programmable count value used for delayed automatic clock gating and self-refresh entry, assuming CLKCTRL field is not 0	RW	0x0000
7	SRFRONRESET	Enter self refresh when a warm reset is applied: 0x0: Feature disabled 0x1: Feature enabled	RW	0x1
6	SRFRONIDLEREQ	Enter self refresh when on hardware idle request: 0x0: Feature disabled 0x1: Feature enabled	RW	0x0
5:4	CLKCTRL	Clock control feature defines clock gating and self refresh: 0x0: No auto clk feature turned on 0x1: Enable internal clock gating on timeout of Auto_cnt 0x2: Enable self-refresh on timeout of Auto_cnt 0x3: Reserved	RW	0x0
3	EXTCLKDIS	Disable the clock provided to the external memories: 0x0: Enable clock 0x1: Disable clock- Logical 0 is applied.	RW	0x0
2	PWDENA	Activate the power-down mode of the target memory through CKE pin. 0x0: Power-down mode feature disabled 0x1: Power-down mode feature enabled	RW	0x1
1	RESERVED	Write 0 for future compatibility. Read returns 0.	RW	0x0

Bits	Field Name	Description	Type	Reset
0	PAGEPOLICY	Page/segment closure policy with respect to power versus bandwidth trade-off - Must be set to 1 0x0: Reserved - must not be used 0x1: High-power/high bandwidth mode (HPHB)	RW	0x1

Table 11-179. Register Call Summary for Register SDRC_POWER_REG

SDRAM Controller (SDRC) Subsystem

- Clocking, Reset, and Power Management Scheme: [0] [1]
- SDRC: [2] [3] [4] [5] [6] [7] [8] [9] [10] [11] [12] [13] [14]
- SDRC Configuration: [15] [16]
- SDRC Setup: [17]
- Manual Software Commands: [18] [19] [20]
- SDRC Register Summary: [21]

Table 11-180. SDRC_MCFG_p

Address Offset	0x0000 0080 + (0x0000 0030 * p)	**Index**	p = 0 to 1
Physical Address	0x6D00 0080 + (0x0000 0030 * p)	**Instance**	SDRC
Description	This register provides the memory configuration register.		
Type	RW		

Register Description for ADDRMUXLEGACY = 0x1

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED	LOCKSTATUS	RESERVED			RASWIDTH			RESERVED		CASWIDTH		ADDRMUXLEGACY	RESERVED			RAMSIZE								BANKALLOCATION		RESERVED	B32NOT16	DEEPPD	DDRTYPE		RAMTYPE

Bits	Field Name	Description	Type	Reset
31	RESERVED	Write 0s for future compatibility. Read returns 0.	RW	0x0
30	LOCKSTATUS	Read-only access lock bit 0x0: This register is fully writable 0x1: When this bit is set, the register can not be unset until next reset of the module.	RW	See (1)
29:27	RESERVED	Write 0s for future compatibility. Read returns 0.	RW	See (1)
26:24	RASWIDTH	RAS address width 0x0: RAS width = 11 bits 0x1: RAS width = 12 bits 0x2: RAS width = 13 bits 0x3: RAS width = 14 bits 0x4: RAS width = 15 bits 0x5: RAS width = 16 bits - Must not be used 0x6: RAS width = 17 bits - Must not be used 0x7: RAS width = 18 bits - Must not be used	RW	See (1)
23	RESERVED	Write 0s for future compatibility. Read returns 0.	RW	See (1)

(1) Reset value is copied from the system control module. See the note in Section 11.2.5.3.2 and Section 7.4.9, *SDRC Registers*, in Chapter 7, *System Control Module*.

Bits	Field Name	Description	Type	Reset
22:20	CASWIDTH	CAS address width 0x0: CAS width = 5 bits 0x1: CAS width = 6 bits 0x2: CAS width = 7 bits 0x3: CAS width = 8 bits 0x4: CAS width = 9 bits 0x5: CAS width = 10 bits 0x6: CAS width = 11 bits 0x7: CAS width = 12 bits	RW	See (1)
19	ADDRMUXLEGACY	Selects the fixed address-muxing scheme or the flexible address-muxing scheme 0x0: Fixed address mux scheme 0x1: Flexible address mux scheme	RW	See (1)
18	RESERVED	Write 0s for future compatibility. Read returns 0.	RW	See (1)
17:8	RAMSIZE	RAM address space size number of 2-MB chunks	RW	See (1)
7:6	BANKALLOCATION	SDRAM banks mapping. Selects the position of the bank address, the row address, and the column address in the system address. 0x0: Bank-row-column 0x1: Bank1-row-bank0-column 0x2: Row-bank-column 0x3: Reserved	RW	See (1)
5	RESERVED	Write 0s for future compatibility. Read returns 0.	RW	See (1)
4	B32NOT16	External SDRAM bus width 0x0: External SDRAM device is x16 bit. 0x1: External SDRAM device is x32 bit.	RW	See (1)
3	DEEPPD	Indicates if the memory supports deep-power-down mode 0x0: No deep-power-down mode support 0x1: The memory supports deep-power-down mode	RW	See (1)
2	DDRTYPE	DDR memory type (assuming RAMTYPE = 01) 0x0: Mobile DDR 0x1: Reserved for future use	RW	See (1)
1:0	RAMTYPE	Memory type 0x0: SDR-SDRAM (single data rate) 0x1: DDR-SDRAM (double data rate) 0x2: Reserved 0x3: Reserved	RW	See (1)

Table 11-181. Register Call Summary for Register SDRC_MCFG_p

SDRAM Controller (SDRC) Subsystem

- External Interface Configuration: [0] [1] [2]
- SDRC: [3] [4] [5] [6] [7] [8] [9] [10] [11]
- SDRC Setup: [13] [14] [15] [16] [17] [18]
- Typical SDRC connection to an External SDRAM Device: [19] [20] [21] [22] [23] [24] [25] [26] [27]
- Understanding SDRAM Subsystem Address Spaces: [28] [29] [30]
- SDRC Register Summary: [31]

Register Description for ADDRMUXLEGACY = 0x0

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED	LOCKSTATUS	RESERVED						ADDRMUX				ADDRMUXLEGACY	RESERVED	RAMSIZE										BANKALLOCATION		RESERVED	B32NOT16	DEEPPD	DDRTYPE	RAMTYPE	

Bits	Field Name	Description	Type	Reset
31	RESERVED	Write 0s for future compatibility. Read returns 0.	RW	0x0
30	LOCKSTATUS	Read-only access lock bit 0x0: This register is fully writable. 0x1: When this bit is set, the register can not be unset until next reset of the module.	RW	0x-
29:25	RESERVED	Write 0s for future compatibility. Read returns 0.	RW	0x-
24:20	ADDRMUX	Address multiplexing scheme - See address-muxing paragraph for more details. 0x0: Address mux scheme 1 0x1: Address mux scheme 2 0x2: Address mux scheme 3 0x3: Address mux scheme 4 0x4: Address mux scheme 5 0x5: Address mux scheme 6 0x6: Address mux scheme 7 0x7: Address mux scheme 8 0x8: Address mux scheme 9 0x9: Address mux scheme 10 0xA: Address mux scheme 11 0xB: Address mux scheme 12 0xC: Address mux scheme 13 0xD: Address mux scheme 14 0xE: Address mux scheme 15 0xF: Address mux scheme 16 0x10: Reserved 0x11: Reserved 0x12: Reserved 0x13: Reserved 0x14: Reserved 0x15: Reserved 0x16: Address mux scheme 23 0x17: Address mux scheme 24 0x18: Address mux scheme 25 0x19: Address mux scheme 26 0x1A: Address mux scheme 27 0x1B: Address mux scheme 28 0x1C: Address mux scheme 29	RW	0x-

Bits	Field Name	Description	Type	Reset
19	ADDRMUXLEGACY	Selects the fixed address-muxing scheme or the flexible address-muxing scheme. 0x0: Fixed address mux scheme 0x1: Flexible address mux scheme	RW	0x-
18	RESERVED	Write 0s for future compatibility. Read returns 0.	RW	0x-
17:8	RAMSIZE	RAM address space size (number of 2-MB chunks)	RW	0x-
7:6	BANKALLOCATION	SDRAM banks mapping. Selects the position of the bank address, the row address, and the column address in the system address. 0x0: Bank-row-column 0x1: Bank1-row-bank0-column 0x2: Row-bank-column 0x3: Reserved	RW	0x-
5	RESERVED	Write 0s for future compatibility. Read returns 0.	RW	0x-
4	B32NOT16	External SDRAM bus width 0x0: External SDRAM device is ×16 bit. 0x1: External SDRAM device is ×32 bit.	RW	0x-
3	DEEPPD	Indicates if the memory supports deep-power-down mode. 0x0: No deep-power-down mode support 0x1: The memory supports deep-power-down mode.	RW	0x-
2	DDRTYPE	DDR memory type (assuming RAMTYPE = 01) 0x0: Mobile DDR 0x1: Reserved for future use	RW	0x-
1:0	RAMTYPE	Memory type 0x0: SDR-SDRAM (single data rate) 0x1: DDR-SDRAM (double data rate) 0x2: Reserved 0x3: Reserved	RW	0x-

Table 11-182. SDRC_MR_p

Address Offset	0x0000 0084 + (0x0000 0030 * p)	**Index**	p = 0 to 1
Physical Address	0x6D00 0084 + (0x0000 0030 * p)	**Instance**	SDRC
Description	This 12-bit register corresponds to the JEDEC SDRAM MR register, with the standard bit fields. All 12 bits are loaded into memory for future extension support. The SDRC keeps an internal copy register used internally; that is, returned when a read access is performed at that address. Load into memory on interconnect write access using MRS command with BA1,BA0 = 0,0.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																				ZERO_1		WBST		ZERO_0				CASL				SIL		BL			

Bits	Field Name	Description	Type	Reset
31:12	RESERVED	Write 0s for future compatibility. Read returns 0s.	RW	0x00000
11:10	ZERO_1	Write 0s, as required by memory specifications. Read returns 0.	RW	0x0
9	WBST	Write burst support must be zero. 0x0: Write burst equals read burst 0x1: Write burst disable (single write access only)	RW	0x0
8:7	ZERO_0	Write 0s, as required by memory specifications. Read returns 0s.	RW	0x0

Bits	Field Name	Description	Type	Reset
6:4	CASL	CAS latency as defined by clock periods 0x1: CAS latency = 1 0x2: CAS latency = 2 0x3: CAS latency = 3 0x4: CAS latency = 4 0x5: CAS latency = 5	RW	0x2
3	SIL	Serial or interleaved mode: must be zero 0x0: Serial mode (always used) 0x1: interleaved mode (never used)	RW	0x0
2:0	BL	Memory burst length 0x0: Burst length = 1 - Not supported 0x1: Burst length = 2 - SDR memory only 0x2: Burst length = 4 - DDR memory only 0x3: Burst length = 8 - Not supported 0x4: Reserved 0x5: Reserved 0x6: Reserved 0x7: Full page - Not supported	RW	0x4

Table 11-183. Register Call Summary for Register SDRC_MR_p

SDRAM Controller (SDRC) Subsystem

- SDRC: [0]
- Mode Registers: [1] [2] [3] [4] [5] [6]
- SDRC Setup: [7] [8] [9]
- Manual Software Commands: [10] [11]
- Typical SDRC connection to an External SDRAM Device: [12] [13] [14] [15]
- SDRC Register Summary: [16]
- SDRC Register Description: [17] [18]

Table 11-184. SDRC_EMR2_p

Address Offset	0x0000 008C + (0x0000 0030 * p)	**Index**	p = 0 to 1
Physical Address	0x6D00 008C + (0x0000 0030 * p)	**Instance**	SDRC
Description	This 12-bit register corresponds to the low-power EMR register, as defined in the mobile DDR JEDEC Standard. All 12 bits are loaded into the memory, thus assuring future extension support. The SDRC keeps an internal copy register used internally; that is, returned when a read access is performed at that address. Load into memory on interconnect write access using MRS command with BA1,BA0 = 1,0.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																				ZERO				DS			TCSR			PASR	

Bits	Field Name	Description	Type	Reset
31:12	RESERVED	Write 0s for future compatibility. Read returns 0s.	RW	0x00000
11:7	ZERO	Write 0s, as required by memory specifications. Read returns 0	RW	0x00
6:5	DS	Driver strength 0x0: Full strength driver 0x1: Weak strength driver 0x2: Reserved 0x3: Reserved	RW	0x0

Bits	Field Name	Description	Type	Reset
4:3	TCSR	Temperature-compensated self-refresh 0x0: 70 degrees maximum temperature 0x1: 45 degrees maximum temperature 0x2: 15 degrees maximum temperature 0x3: 85 degrees maximum temperature	RW	0x0
2:0	PASR	Partial array self-refresh 0x0: All banks. 0x1: 1/2 array 0x2: 1/4 array 0x3: Reserved 0x4: Reserved 0x5: 1/8 array 0x6: 1/16 array 0x7: Reserved	RW	0x0

Table 11-185. Register Call Summary for Register SDRC_EMR2_p

SDRAM Controller (SDRC) Subsystem

- SDRC: [0]
- Mode Registers: [1] [2] [3] [4]
- SDRC Setup: [5] [6] [7] [8]
- Manual Software Commands: [9]
- SDRC Register Summary: [10]
- SDRC Register Description: [11] [12]

Table 11-186. SDRC_ACTIM_CTRLA_p

Address Offset	0x0000 009C + (0x0000 0028 * p)	**Index**	p = 0 to 1
Physical Address	0x6D00 009C + (0x0000 0028 * p)	**Instance**	SDRC
Description	The ac timing control register A sets the ac parameter values in clock cycle units to best match the memory characteristics.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TRFC								TRC								TRAS								TRP								TRCD								TRRD								TDPL								RESERVED								TDAL							

Bits	Field Name	Description	Type	Reset
31:27	TRFC	Autorefresh to active	RW	0x00
26:22	TRC	Row cycle time	RW	0x00
21:18	TRAS	Row active time	RW	0x0
17:15	TRP	Row precharge time	RW	0x0
14:12	TRCD	Row to column delay time	RW	0x0
11:9	TRRD	Active to active command period	RW	0x0
8:6	TDPL	Data-in to precharge command (write recovery time tWR)	RW	0x0
5	RESERVED	Write 0s for future compatibility. Read returns 0.	RW	0x0
4:0	TDAL	Data-in to active command	RW	0x00

Table 11-187. Register Call Summary for Register SDRC_ACTIM_CTRLA_p

SDRAM Controller (SDRC) Subsystem

- SDRC Setup: [0]
- Manual Software Commands: [1]
- Typical SDRC connection to an External SDRAM Device: [2] [3] [4]
- SDRC Register Summary: [5]

Table 11-188. SDRC_ACTIM_CTRLB_p

Address Offset	0x0000 00A0 + (0x0000 0028 * p)	**Index**	p = 0 to 1
Physical Address	0x6D00 00A0 + (0x0000 0028 * p)	**Instance**	SDRC
Description	The ac timing control register B sets the ac parameter values in clock cycle unit, to best match the memory characteristics		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED														TWTR		RESERVED	TCKE			RESERVED	TXP			TXSR							

Bits	Field Name	Description	Type	Reset
31:18	RESERVED	Write 0s for future compatibility Reads return zeros.	RW	0x00000
17:16	TWTR	Internal write to read command delay. 0x0: 1 minimum clock cycle before next command 0x1: 1 minimum clock cycle 0x2: 2 minimum clock cycles 0x3: 3 minimum clock cycles	RW	0x0
15	RESERVED	Write 0s for future compatibility Reads return zeros.	RW	0
14:12	TCKE	CKE minimum pulse width (high and low) 0x0: 1 minimum clock cycle 0x1: 1 minimum clock cycle 0x2: 2 minimum clock cycles ... 0x7: 7 minimum clock cycles	RW	0x0
11	RESERVED	Write 0s for future compatibility Reads return zeros.	RW	0
10:8	TXP	Exit power-down to next valid command delay. 0x0: 1 minimum clock cycle before next command 0x1: 1 minimum clock cycle 0x2: 2 minimum clock cycles ... 0x7: 7 minimum clock cycles	RW	0x0
7:0	TXSR	Self-refresh exit to active period	RW	0x00

Table 11-189. Register Call Summary for Register SDRC_ACTIM_CTRLB_p

SDRAM Controller (SDRC) Subsystem

- SDRC Setup: [0]
- Typical SDRC connection to an External SDRAM Device: [1] [2] [3]
- SDRC Register Summary: [4]

Table 11-190. SDRC_RFR_CTRL_p

Address Offset	0x0000 00A4 + (0x0000 0030 * p)	**Index**	p = 0 to 1
Physical Address	0x6D00 00A4 + (0x0000 0030 * p)	**Instance**	SDRC
Description	SDRAM memory autorefresh control		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								ARCV																RESERVED						ARE	

Bits	Field Name	Description	Type	Reset
31:24	RESERVED	Write 0s for future compatibility. Read returns 0.	RW	0x00
23:8	ARCV	Autorefresh counter value to set the refresh period. The autorefresh counter is uploaded with the result of: (tREFI / tCK) - 50	RW	0x0000
7:2	RESERVED	Write 0s for future compatibility. Read returns 0.	RW	0x00
1:0	ARE	Autorefresh enable 0x0: Autorefresh is disabled 0x1: Counter is loaded with ARCV: 1 autorefresh command when autorefresh counter reaches 0. 0x2: Counter is loaded with 4 ×ARCV: Burst of 4 autorefresh commands when autorefresh counter reaches 0. 0x3: Counter is loaded with 8 ×ARCV: Burst of 8 autorefresh commands when autorefresh counter reaches 0.	RW	0x0

Table 11-191. Register Call Summary for Register SDRC_RFR_CTRL_p

SDRAM Controller (SDRC) Subsystem

- SDRC Setup: [0] [1] [2]
- Manual Software Commands: [3] [4] [5]
- Typical SDRC connection to an External SDRAM Device: [6] [7]
- SDRC Register Summary: [8]

Table 11-192. SDRC_MANUAL_p

Address Offset	0x0000 00A8 + (0x0000 0030 * p)	**Index**	p = 0 to 1
Physical Address	0x6D00 00A8 + (0x0000 0030 * p)	**Instance**	SDRC
Description	This register allows to send specific commands to the external memory devices under software control. Any write to this register generates the appropriate sequence based on the command code.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CMDPARAM																RESERVED												CMDCODE			

Bits	Field Name	Description	Type	Reset
31:16	CMDPARAM	Manual command parameter, if any.	RW	0x0000
15:4	RESERVED	Write 0s for future compatibility. Read returns 0s.	RW	0x000
3:0	CMDCODE	Memory command opcode; other values reserved for future implementations. 0x0: NOP command - no parameter 0x1: Precharge all command - no parameter 0x2: Autorefresh command - no parameter 0x3: Enter deep-power-down - no parameter 0x4: Exit deep-power-down - no parameter 0x5: Enter self-refresh - no parameter 0x6: Exit self-refresh - no parameter 0x7: Set CKE signal high - no parameter 0x8: Set CKE low - no parameter 0x9: Reserved 0xA: Reserved 0xB: Reserved 0xC: Reserved	RW	0x0

Table 11-193. Register Call Summary for Register SDRC_MANUAL_p

SDRAM Controller (SDRC) Subsystem

- SDRC: [0] [1] [2]
- SDRC Setup: [3]
- Manual Software Commands: [4] [5] [6] [7] [8] [9] [10] [11] [12] [13] [14] [15]
- SDRC Register Summary: [16]

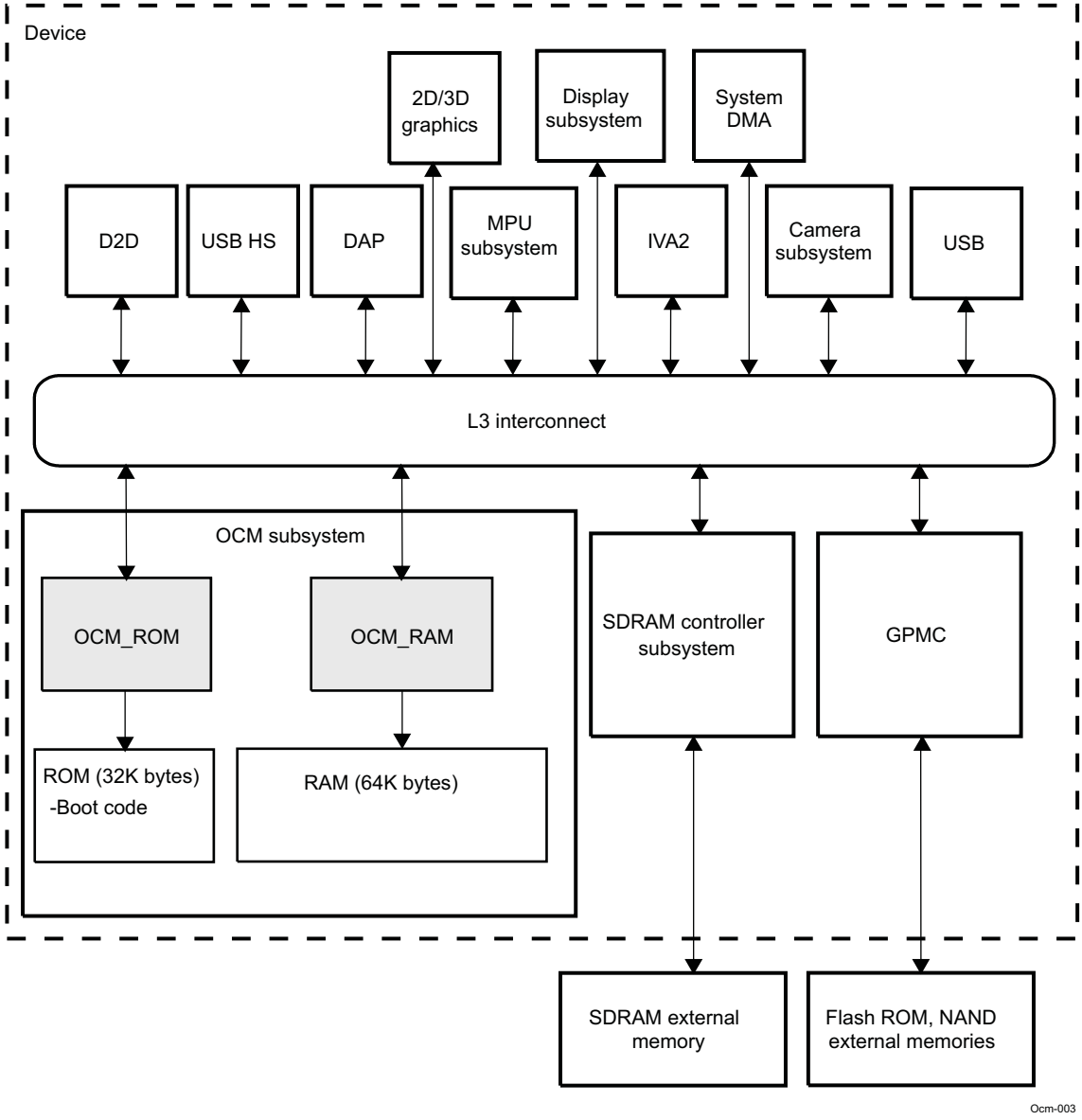
11.3 On-Chip Memory Subsystem

11.3.1 OCM Subsystem Overview

The on-chip memory subsystem consists of two separate on-chip memory controllers, one connected to an on-chip ROM (OCM_ROM) and the other connected to an on-chip RAM (OCM_RAM). Each memory controller has its own dedicated interface to the L3 interconnect.

Figure 11-85 is an overview of the OCM subsystem.

Figure 11-85. OCM Subsystem Overview



Multiple L3 initiators (such as remote devices) have access to the RAM through 2D/3D graphics, the MPU subsystem, sDMA, the camera subsystem, the display subsystem, IVA2, and USB.

ROM is used for direct boot code and boot from external NAND flash.

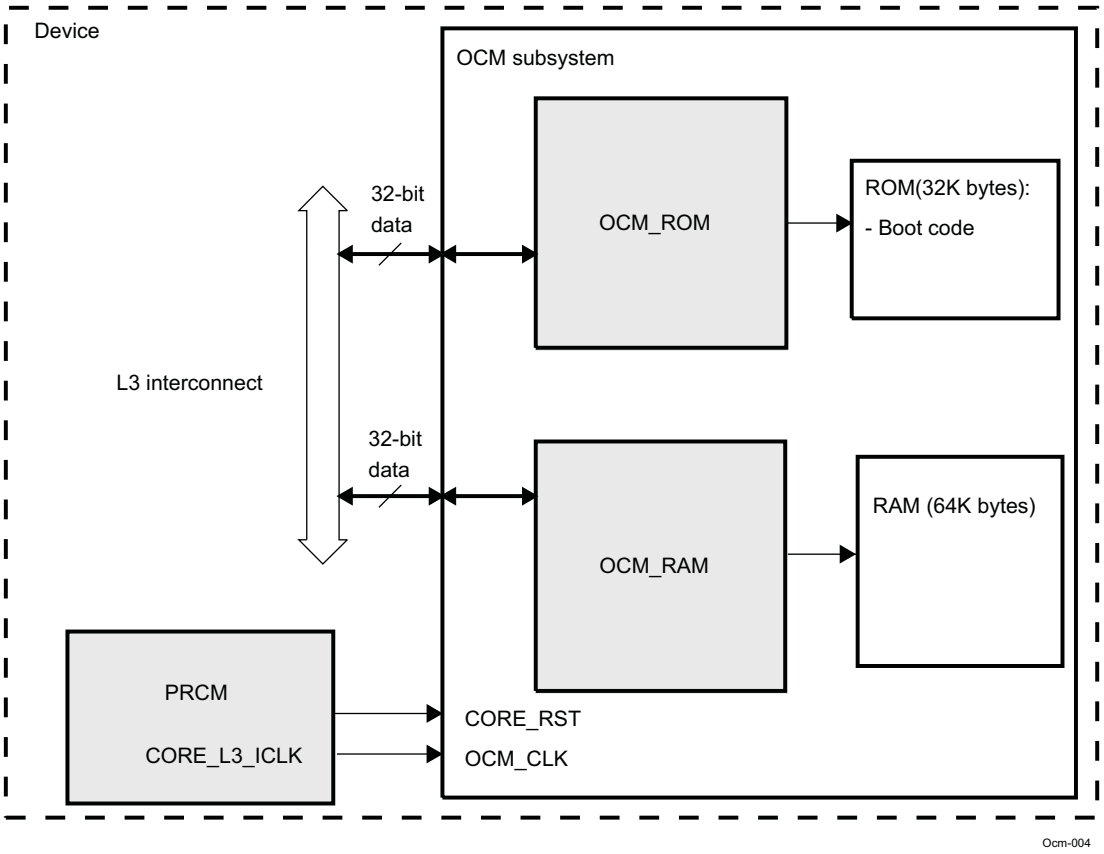
11.3.2 OCM Subsystem Integration

11.3.2.1 Description

The OCM_ROM and OCM_RAM allow transactions between the system initiators and the multiple memories, through the L3 interconnect.

Figure 11-86 shows the integration of the OCM subsystem to the OMAP processor.

Figure 11-86. OCM Subsystem Integration to the Device



11.3.2.2 Clocking, Reset, and Power-Management Scheme

11.3.2.2.1 Clocking

The on-chip boot ROM and RAM are clocked only when they are accessed.

The interface clock OCM_CLK comes from the PRCM module and runs at the L3 interconnect frequency. The OCM_CLK source is PRCM CORE_L3_ICLK output. This clock is also used as the functional clock for the OCM module.

For the OCM subsystem, no register enables the gating of OCM_CLK.

However, OCM does support the handshaking protocol with the PRCM module. It is not programmable by software. The following signals support the handshaking protocol for ROM and RAM devices:

- OCMROM_IDLEREQ/OCMROM_SIDLEACK (for ROM devices)
- OCMRAM_IDLEREQ/OCMRAM_SIDLEACK (for RAM devices)

OCM_CLK is gated if all modules (including the OCM) belonging to the L3 CLK domain send a SIdleAck back to the PRCM module after reception of the MIdleReq request.

For details, see Chapter 4, *Power, Reset, and Clock Management.*

When the memory is not accessed by the system, the module performs automatic clock gating. Because the clock to the memory is dynamically gated, there is no extra latency when the clock must be switched on after an idle state.

11.3.2.2.2 Hardware Reset

Global reset of the module is performed by activation of the CORE_RST in the core reset domain (see Chapter 4, *Power, Reset, and Clock Management.*).

11.3.2.2.3 Power Domain

OCM power is supplied by the CORE power domain (see Chapter 4, *Power, Reset, and Clock Management*).

11.3.3 OCM Subsystem Functional Description

11.3.3.1 OCM_ROM

The embedded ROM is used primarily for booting, flashing, and context restoring.

The device-embedded ROM (total 32K bytes) has the following characteristics:

- The OCM_ROM is always accessible, and contains the boot area.
- The OCM_ROM supports single and burst access transactions.
- The OCM_ROM operates at full interconnect clock frequency.
- The COM_ROM needs three cycles for initial access and one cycle per subsequent access.

The memory space of the embedded ROM starts at 0x4001 4000 and ends at 0x4001 BFFF.

11.3.3.2 OCM_RAM

By default, only 2K bytes are available after reset; however, the configuration can then be changed to adapt to booting/flashing, normal boot, or to any application requirement.

The device-embedded RAM has the following characteristics:

- Operates at full L3 interconnect clock frequency
- Fully pipelined, one 32-bit access per cycle
- Restricted access support, based on:
  - A region-based partitioning (see the L3 firewall description)
  - The module owner of the access, with respect to its read and write permission to that region
  - The transaction attributes of the access, with respect to the region permission properties:
    - User/supervisor
    - Code/data access

The OCM_RAM can be partitioned using the L3 firewall and used as RAM for a video frame buffer or to any application.

The RAM memory space starts at 0x4020 0000 and ends at 0x4020 FFFF.

Camera Subsystem

This chapter describes the camera subsystem in the OMAP34xx multimedia device with the Image Signal Processor (ISP) feature.

Topic **Page**

12.1 Camera ISP Overview **1308**
12.2 Camera ISP Environment **1313**
12.3 Camera ISP Integration **1359**
12.4 Camera ISP Functional Description **1370**
12.5 Camera ISP Basic Programming Model **1441**
12.6 Camera ISP Register Manual **1495**

12.1 Camera ISP Overview

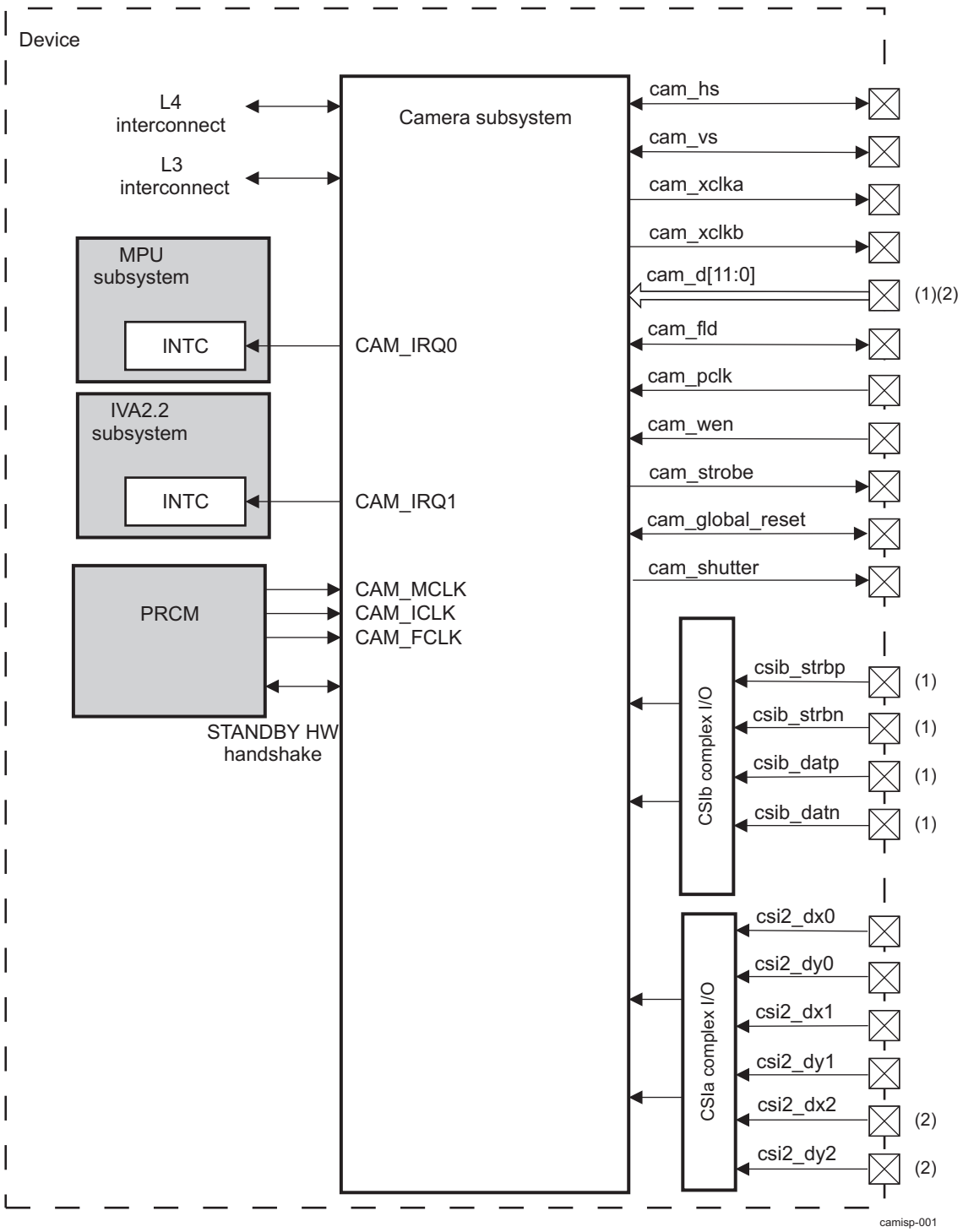
The Camera ISP is a key component for imaging and video applications such as video preview, video record, and still-image capture with or without digital zooming.

The Camera ISP provides the system interface and the processing capability to connect RAW image-sensor modules to the OMAP34xx device.

The Camera ISP supports one parallel interface (CPI) and two serial interfaces: MIPI® CSI1 and one MIPI CSI2 interface, that can be active simultaneously. However, only one of these interfaces can use the video-processing hardware, when it is not used by the parallel interface. Because of a pin-muxing limitation, the parallel interface and CSI1 (CSIb) cannot be active at the same time.

Figure 12-1 shows the Camera ISP.

Figure 12-1. Camera ISP Highlight



(1) There is no mux control bit. If the CSIb interface is enabled, the cam_d6 to cam_d9 pins are used by the CSIb interface.

(2) These signals are muxed. They are controlled by the CONTROL_PADCONF_CAM_FLD and CONTROL_PADCONF_CAM_D1 registers.

12.1.1 Camera ISP Features

The Camera ISP can support the following features:

- **Image sensor:**
  - Interface with various image sensors:
    - R, G, B primary colors
    - Ye, Cy, Mg, G complementary colors
  - Support for electronic rolling shutter (ERS) and global-release reset shutters
- **CSI1 serial interface:** The CSI1 receiver is compatible with the MIPI CSI1 specification. It supports the following features:
  - Image from sensor
    - Transfer of pixels and data received by the CSI1 DSI_PHY RX to system memory or to the video pipeline
    - Unidirectional data link
    - 1D and 2D addressing mode
    - False synchronization code protection
    - Ping-pong mechanism for double-buffering
    - Support of RGB, RAW, YUV, and JPEG formats
  - Image read from memory
    - RAW formats supported
- **MIPI CSI2 serial interface:** The Camera ISP supports one MIPI CSI2 serial interface (CSIa) with 2 data lanes. MIPI CSI2 enables data transfer at up to 1.6G bps. It is based on the MIPI CSI2 Specification 1.0.
  - Transfer pixels and data received by the CSI2 DSI_PHY RX to the system memory or to the image pipeline
  - Uses unidirectional data link
  - Supports two data-configurable links, in addition to the clock signaling.
  - Maximum data rate of up to 800M bps per data lane
  - Data merger for two-data-lane configuration
  - Error detection and correction by the protocol engine
  - DMA engine integrated with dedicated FIFO
  - 1-D and 2-D addressing mode (rotation is not supported by the 2D mode)
  - Ping-pong mechanism for double buffering
  - Burst support
  - JPEG support for unknown length transfer
  - RGB, RAW, and YUV formats supported
  - Storage in progressive mode for interlaced stream (using line numbering)
  - Conversion of the RGB formats
- **Parallel interface:** The Camera parallel interface (CPI) supports two modes:
  - **SYNC mode:** In this mode, the image-sensor module provides horizontal and vertical synchronization signals to the parallel interface, along with the pixel clock. This mode works with 8-, 10-, 11-, and 12-bit data (above 10-bit RAW data, the processing pipe cannot be used; data must be transferred to memory). SYNC mode supports progressive and interlaced image-sensor modules.
  - **ITU mode:** In this mode, the image-sensor module provides an ITU-R BT 656-compatible data stream. The horizontal and vertical synchronization signals are not provided to the interface. Instead, the data stream embeds start-of-active (SAV) and end-of-active video (EAV) synchronization code. This mode works in 8- and 10-bit configurations.

Notes:

- Up to 8-bit data at 130 MHz can be transferred to memory.
- Up to 10-bit data at 75 MHz can be processed by the image pipeline or transferred to memory.
- Up to 12-bit data at 75 MHz can be transferred to memory as is, or after processing inside the CCDC. It can also be internally converted to 10-bit data for full processing.

The parallel interface can be active simultaneously with one serial interface (only CSIa). However, the serial interface cannot use the video-processing hardware when the parallel interface is active. The parallel interface can use the video-processing hardware.

Note: CPI and CSIb cannot be active at the same time, due to a pin-muxing limitation.

- **Video processing:** The video-processing hardware removes the need for expensive camera modules to perform processing functions. The hardware pipeline contains two parts: front end and back end:
  - **Video processing front end (VPFE):** Performs signal-processing operations on RAW image input data. The output data can go directly to memory for software processing, or to the video-processing back end for further processing. The video-processing front end is supported by the CCDC module. Signal-processing operations include:
    - Optical clamping
    - Optical black clamp
    - Black-level compensation
    - Look-up table (LUT) based faulty pixel correction
    - 2D lens-shading compensation
    - Data formatter
    - Output formatter

    Note: Up to 12-bit data at 83 MHz can be transferred from the video port to the ISP submodules. The video processing ISP can treat 1 pixel every two interconnect clock cycles.

  - **Video processing back end (VPBE):** Performs signal-processing operations on RAW image input data. Outputs YCbCr 4:2:2 data.
    - **Preview module:** Signal-processing operations include:
      - A-law decompression: transforms non-linear 8-bit data to 10-bit linear data. The CCDC module can perform A-law compression
      - Noise reduction and faulty pixel correction
        - Dark frame capture and subtraction
        - Horizontal median filter
        - Programmable filter: 3x3 kernel of the same color
        - Couplet faulty pixel correction
      - Digital gain
      - White balance
      - Programmable color filter array (CFA) interpolation: 5x5 kernel
      - Black adjustment
      - Programmable color correction (RGB to RGB)
      - Programmable gamma correction: 1024 entries for each color
      - Programmable color conversion (RGB to YCbCr 4:4:4)
      - Color subsampling (YCbCr 4:4:4 to YCbCr 4:2:2)
      - Luminance enhancement (non-linear), chrominance suppression and offset

      The preview module can also work from memory to memory.
    - **Resizer module:** Performs on-the-fly upsampling (up to x4) and downsampling (down to x0.25) of YCbCr 4:2:2 data by applying high-quality horizontal and vertical filters. The horizontal and vertical resizer ratios are independent. Applicable ratios are 256/N, with N ranging from 64 to

1024. This feature enables digital zooming (upsampling) and video preview (downsampling). The resizer module can also work from memory to memory. Higher or lower ratios can be obtained by combining on-the-fly resizing followed by memory-to-memory resizing.

- **Statistic collection modules (SCM):** The host CPU uses statistics to adjust various parameters for processing image data.
  - **3A metrics:** Collects on-the-fly RAW image data metrics, which are required to perform the control loops for auto white balance (AWB), auto exposure (AE), and autofocus (AF). The MPU subsystem typically uses data metrics to adjust various parameters for processing image data.
  - **Histogram:** Performs on-the-fly pixel binning of RAW image, based on color value ranges and regions. Supports up to 4 regions and up to 256 bins per color. The MPU subsystem typically uses the histogram with 3A metrics to adjust various parameters for processing image data.

    The histogram module can also work from memory to memory.
- **Central-resource shared buffer logic (SBL):** Buffers and schedules memory accesses requested by Camera ISP modules
- **Circular buffer:** Prevents storage of full image frames in memory when data must be postprocessed and/or preprocessed by software
- **Memory management unit (MMU):** Manages virtual-to-physical address translation for external addresses and solves the memory-fragmentation issue. Enables the camera driver to dynamically allocate and deallocate memory; the MMU handles memory fragmentation.
- **Clock generator:** Generates two independent clocks that can be used by two external image sensors
- **Timing control:**
  - Generation of two clocks that can be used by the external image sensors
  - Generation of signals for strobe flash, mechanical shutter, and global reset. Support for red-eye removal.
- **Open core protocol (OCP) compliant:**
  - One 64-bit master interface connected to L3
  - One 32-bit slave interface connected to L4

12.2 Camera ISP Environment

12.2.1 Camera ISP Functions

Table 12-1 describes the Camera ISP functions and the corresponding application fields.

Table 12-1. Camera ISP Functions

Function	Description
Parallel interface in generic configuration (SYNC mode)	The Camera ISP supports up to 12 bits (Most image-processing pipeline operations support only up to 10 bits; when it is more than 10 bits, some limited capabilities apply to CCDC only). The Camera ISP can interface with RAW interlaced or progressive image sensors using RGB or complementary color mosaic filters.
Parallel interface in ITU-R BT.656 configuration (ITU mode)	The Camera ISP can extract the synchronization signal start of active video and end of active video from the ITU-R BT.656 bit stream. 8-bit and 10-bit modes are supported.
CSIb serial interface configuration (serial mode) (CSI1)	The Camera ISP supports one CSI1 (CSI1b).
CSIa serial interface configuration (serial mode) (MIPI CSI2)	The Camera ISP supports one MIPI CSI2 serial interface (CSIa).

Note: The CSIa (CSI2 receiver) and CSIb (CSI1 receiver) can be active simultaneously. The CSIa or CSIb interface data can go through the video-processing hardware. The CSI interface not using the video-processing hardware goes directly to memory.

The parallel interface is limited to 10 bits when used simultaneously with CSIa (CSI2 receiver).

The parallel interface cannot be used simultaneously with CSIb (CSI1 receiver).

12.2.2 Camera ISP Signal Descriptions

Table 12-2. I/O Description

Signal Name	I/O[1]	Description	Parallel SYNC Mode	Parallel ITU Mode	Serial Mode CSI1	Serial Mode CSI2
cam_hs	I/O	Line trigger input/output signal	+			
cam_vs	I/O	Frame trigger input/output signal	+			
cam_fld	I/O	Field identification input/output signal	+			
cam_pclk	I	Parallel interface pixel clock	+	+		
cam_d[11:0]	I	Parallel mode: input data bits 0 to 11	+	+ [9:0] when bridge is not used; [7:0] when bridge is used		
cam_wen	I	External write-enable signal	+			
cam_strobe	O	Flash strobe control signal	+	+	+	
cam_shutter	O	Mechanical shutter control signal	+	+	+	
cam_global_ reset	I/O	Global reset release shutter signal	+	+	+	

(1) I = Input, O = Output, PWR = Power

Table 12-2. I/O Description (continued)

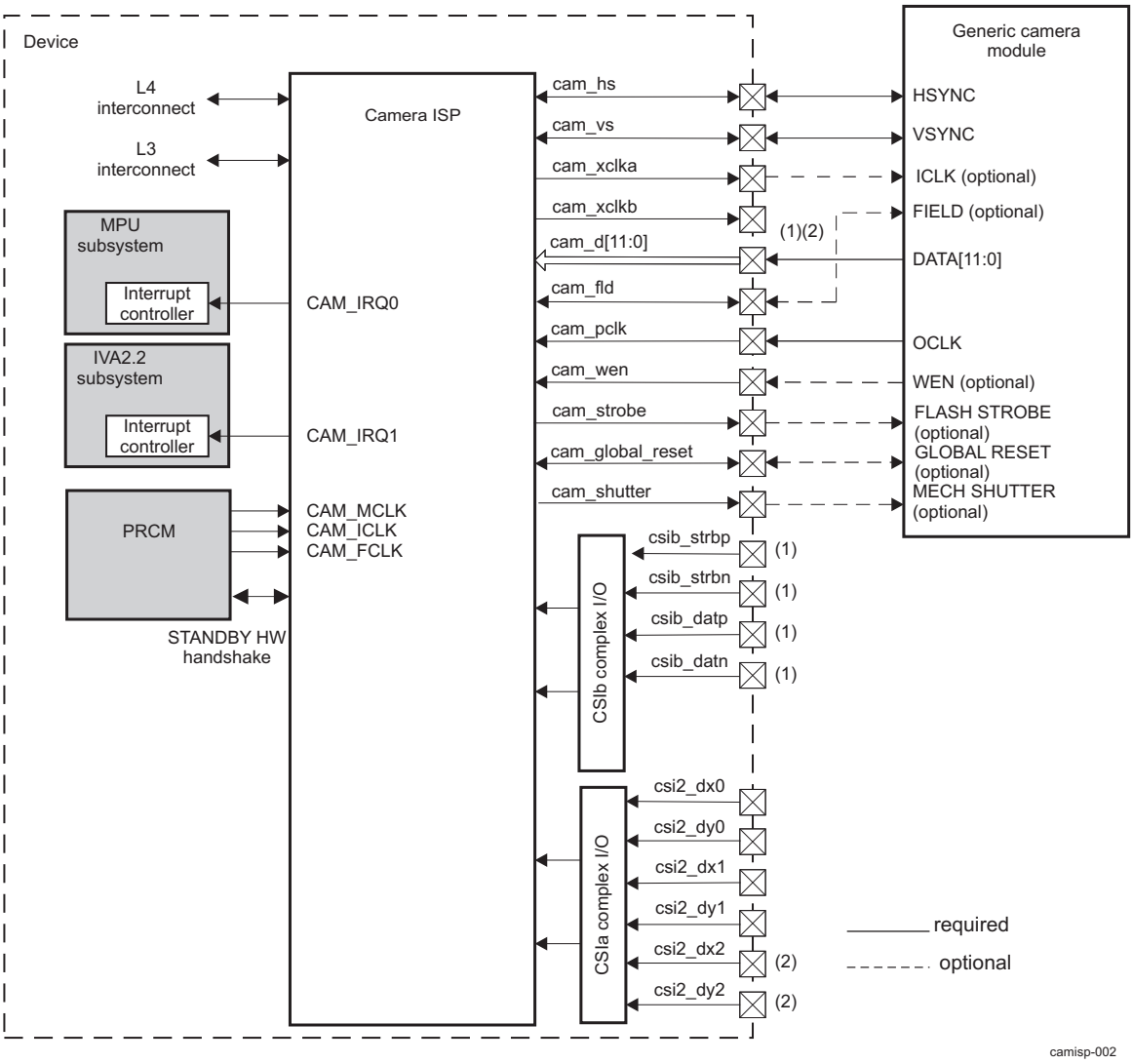
Signal Name	I/O[1]	Description	Parallel SYNC Mode	Parallel ITU Mode	Serial Mode CSI1	Serial Mode CSI2
csi2_dx0	I	Serial CSI2 mode: Differential clock positive input				+
csi2_dy0	I	Serial CSI2 mode: Differential clock negative input				+
csi2_dx1	I	Serial CSI2 mode: Differential data lane 1 positive input				+
csi2_dy1	I	Serial CSI2 mode: Differential data lane 1 negative input				+
csi2_dx2	I	Serial CSI2 mode: Differential data lane 2 positive input				+
csi2_dy2	I	Serial CSI2 mode: Differential data lane 2 negative input				+
csib_strbp	I	Serial CSI1 mode: Differential clock positive input			+	
csib_strbn	I	Serial CSI1 mode: Differential clock negative input			+	
csib_datp	I	Serial CSI1 mode: Differential data positive input			+	
csib_datn	I	Serial CSI1 mode: Differential data negative input			+	
cam_xclka	O	External clock for the image-sensor module	+	+	+	+
cam_xclkb	O	External clock for the image-sensor module	+	+	+	+

12.2.3 Camera ISP Modes

- **Parallel interface in generic configuration**

  Figure 12-2 shows a block diagram of the parallel interface in generic configuration.

Figure 12-2. Parallel Interface in Generic Configuration



(1) There is no mux control bit. If the CSIb interface is enabled, the cam_d6 to cam_d9 pins are used by the CSIb interface.

(2) These signals are muxed. They are controlled by the CONTROL_PADCONF_CAM_FLD and CONTROL_PADCONF_CAM_D1 registers.

CAUTION

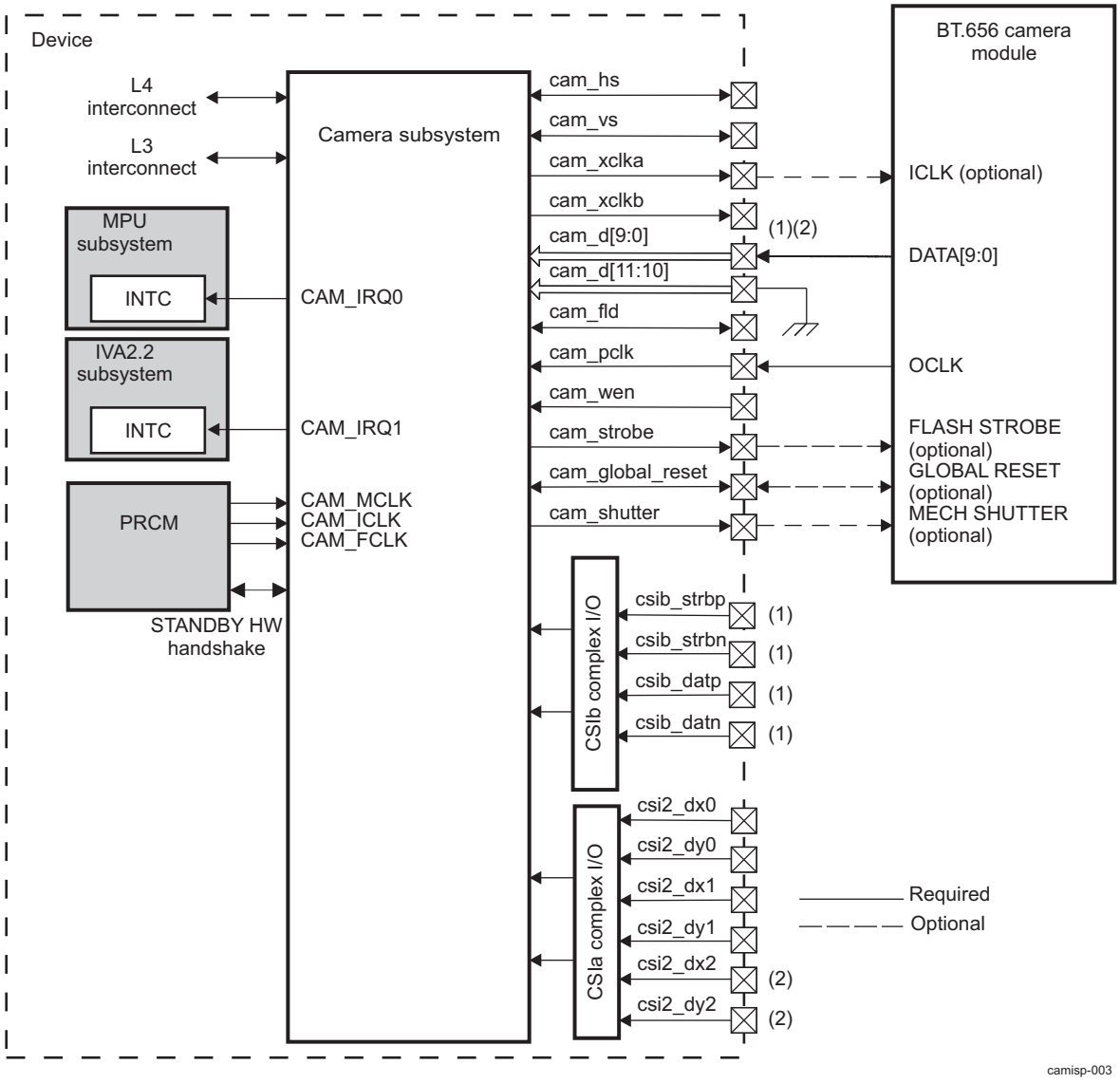
The parallel interface in generic configuration mode works with 8-, 10-, 11-, and 12-bit sensors. 8 to 12 bits refers to data-lane count, not pixel coding. For example, when a video decoder is used, it sends 16 bits per pixel. The data is transmitted through 8 data lanes at x2 pixel clock. The bridge can be used to reassemble the 16-bit pixels.

Note: For information about using the data-lane shifter for different configurations, see Table 12-27.

Note: Due to a pin-muxing limitation, CPI and CSIb cannot be active at the same time.

- **Parallel interface in ITU-R BT.656 configuration**
  Figure 12-3 is a block diagram of the parallel interface in ITU-R BT.656 configuration.

Figure 12-3. Parallel Interface in ITU-R BT.656 Configuration



(1) There is no mux control bit. If the CSIb interface is enabled, the cam_d6 to cam_d9 pins are used by the CSIb interface.

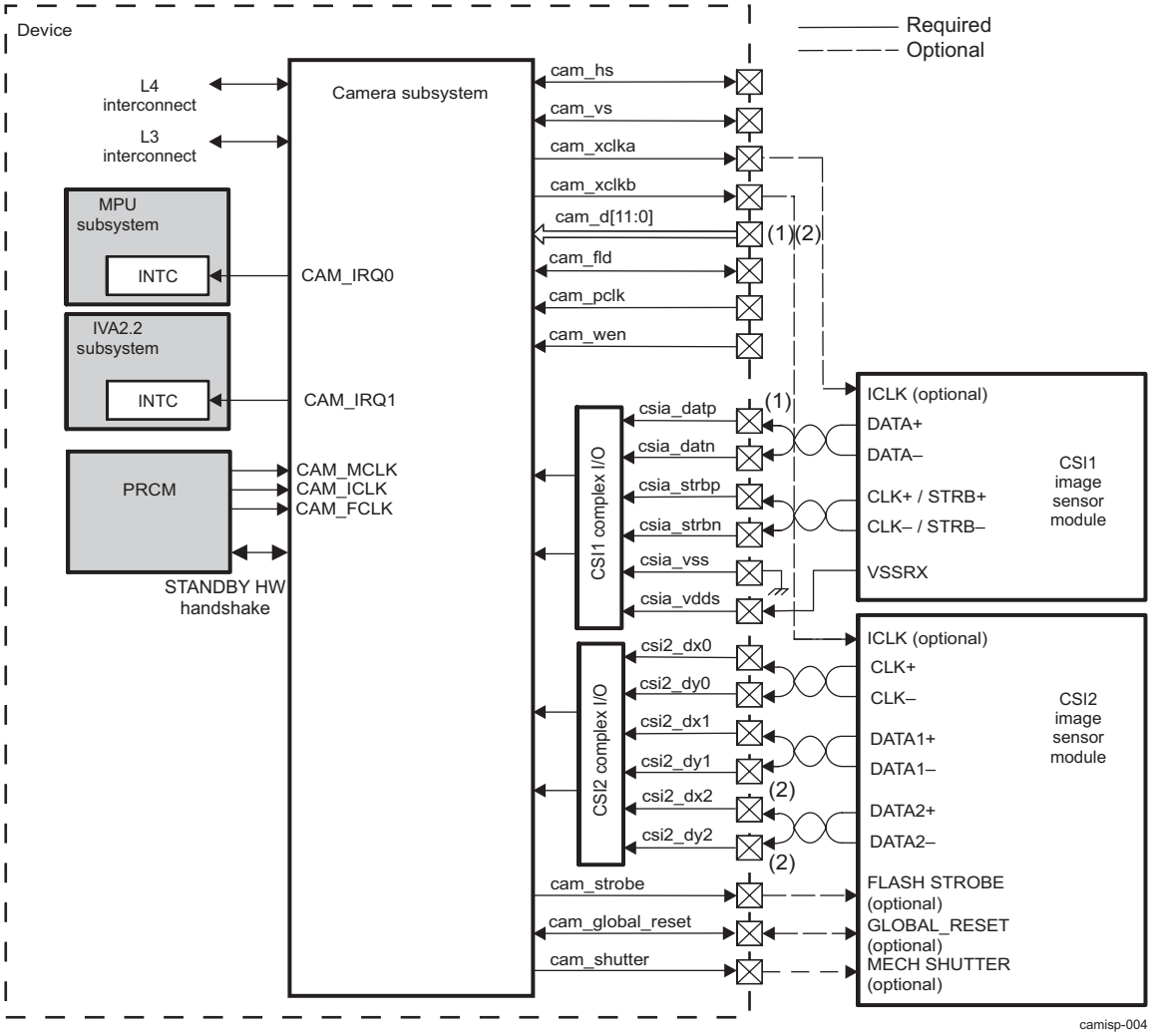
(2) These signals are muxed. They are controlled by the CONTROL_PADCONF_CAM_FLD and CONTROL_PADCONF_CAM_D1 registers.

Note: For information about using the data-lane shifter for different configurations, see Table 12-27.

Note: A pin-muxing limitation prevents CPI and CSIb from being active at the same time.

- **Serial interface configuration**
  Figure 12-4 shows a block diagram of the serial interface configuration.

Figure 12-4. CSI2, CSI1 Serial Interface Configuration



(1) There is no mux control bit. If the CSIb interface is enabled, the cam_d6 to cam_d9 pins are used by the CSIb interface.

(2) These signals are muxed. They are controlled by the CONTROL_PADCONF_CAM_FLD and CONTROL_PADCONF_CAM_D1 registers.

Note: Due to a pin-muxing limitation, CPI and CSIb cannot be active at the same time.

cam_strobe, cam_global_reset, and cam_shutter can be connected to the CSI1 camera module instead of CSI2 camera module.

12.2.4 Camera ISP Protocols and Data Formats

12.2.4.1 Parallel Generic Configuration Protocol and Data Format (8, 10, 11, 12 Bits)

The SYNC mode implements a generic parallel interface with the image sensor. The SYNC mode supports 8 to 14-bit-wide data signals.

In this configuration, no assumptions are made on the data format of pixels, but the dynamic range is limited to 8-, 10-, 12- or 14-bit (data can be pure luminance for black and white sensor, RGB444, Bayer RGB, etc.). The pixel data is presented on cam_d, where one pixel is sampled for every cam_pclk rising edge (or falling edge, depending on the configuration of cam_pclk polarity). For more information, see Section 12.4.

Additional pixel times between rows represent blanking periods. Active pixels are identified by a combination of two additional timing signals: horizontal synchronization (cam_hs) and vertical synchronization (cam_vs). During the image-sensor readout, these signals define when a row of valid data begins and ends, and when a frame starts and ends.

Note: For correct operation, the clock cam_pclk must run during blanking periods (cam_hs and cam_vs inactive). cam_pclk must start before sending cam_d and start cam_vs and cam_hs.

Figure 12-5 and Figure 12-6 show the frame and data timing, respectively, based on synchronization signals in the parallel No BT configuration.

Figure 12-5. Synchronization Signals and Frame Timing in SYNC Mode

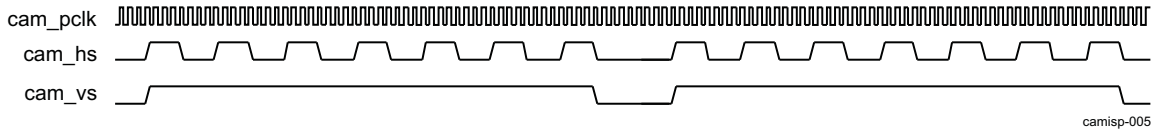
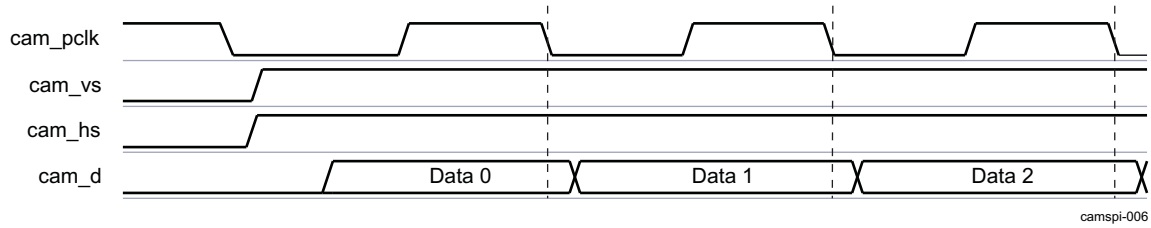


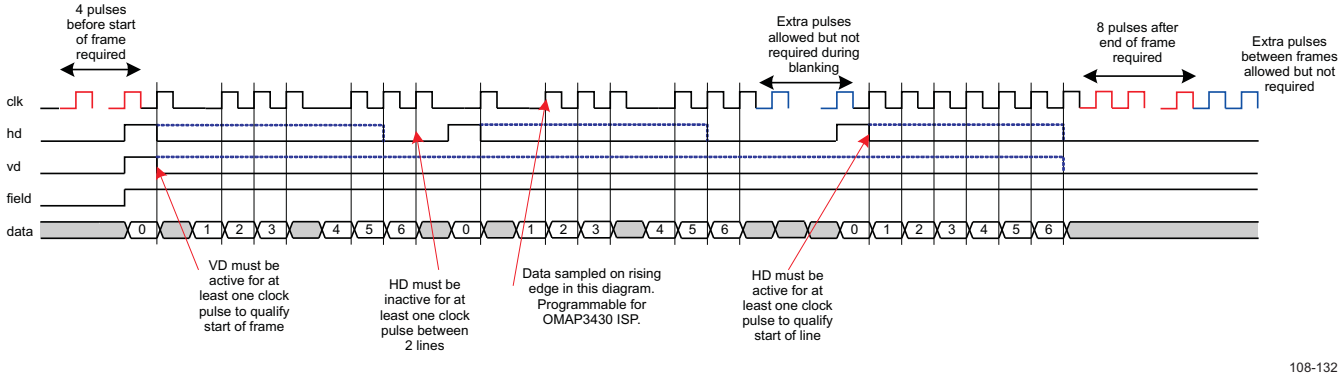
Figure 12-6. Synchronization Signals and Data Timing in SYNC Mode



Note: The pixel clock can be gated to qualify valid pixels. It can also be gated during blanking periods to reduce power consumption. However, at least 4 clock pulses are required before sending active image data and synchronization information; 8 clock pulses are required after the end of active video. Extra-clock pulses are allowed but not required during the line blanking periods.

Figure 12-7 shows the timing diagram of the SYNC move clock gating.

Figure 12-7. SYNC Mode Clock Gating



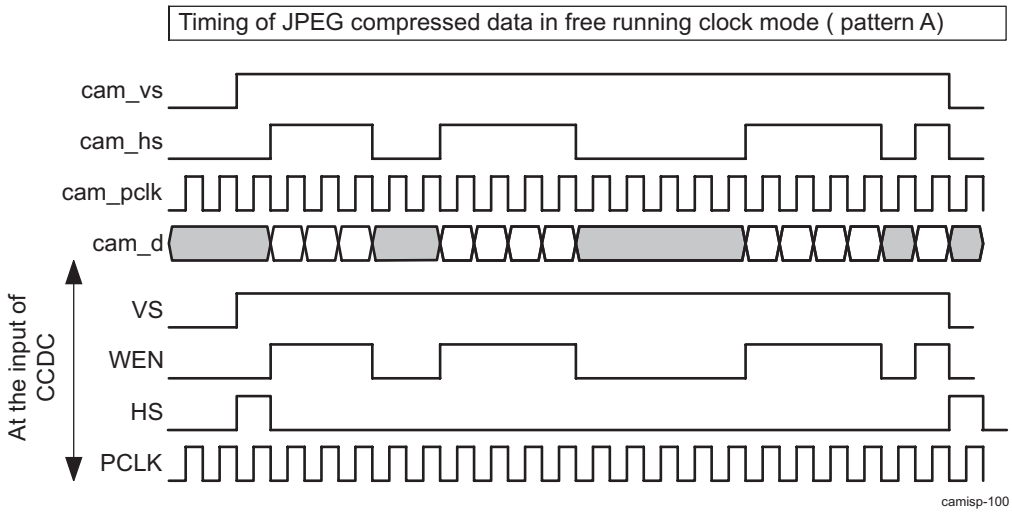
12.2.4.2 Parallel Generic Configuration: JPEG Sensor Connection on the Parallel Interface

Some camera modules integrate an image-signal processor (ISP) and a JPEG encoder. The CCDC can interface with these camera modules and transfer the received JPEG stream to memory.

Only pattern A mode is supported. To use this mode, set the ISP_CTRL [30] JPEG_FLUSH bit.

Figure 12-8 shows timing diagrams for an A pattern.

Figure 12-8. JPEG Stream Timing Diagrams



CAUTION

The bridge cannot be used for JPEG sensor connections.

12.2.4.3 ITU-R BT.656 Protocol and Data Format (8, 10 Bits)

CAUTION

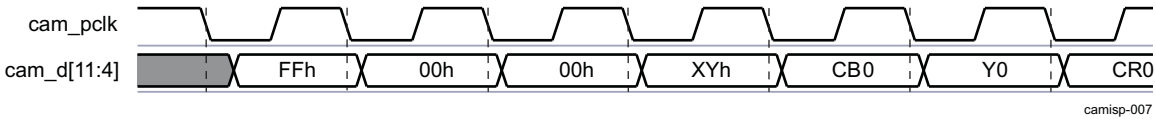
The ITU-R BT.656 mode cannot be used when the bridge is enabled.

The Camera ISP interface supports data in ITU-R BT.656 format.

The ITU-R BT.656 standard specifies a method of transferring YUV422 data over an 8- or 10-bit video interface.

Figure 12-9 shows the data timing diagram with embedded synchronization signal.

Figure 12-9. Data Timing With Embedded Synchronization Signals (8-Bit Case)



In BT.656, the data words (8- or 10-bit) in which the eight most-significant bits (MSBs) are all set to 1, or all set to 0 are reserved. Only 254 of the possible 256 8-bit word values, and 1016 of the possible 1024 10-bit word values represent signal values.

The data is multiplexed in the following order: Cb0 Y0 Cr0 Y1 Cb2 Y2 Cr2 Y3, etc., where the byte sequence Cb2n Y2n Cr2n refers to interleaved luminance and chroma samples and the following byte Y2n + 1 corresponds to the next luminance sample.

The BT.656 protocol uses unique timing reference signals embedded in the video stream. The synchronization signals cam_hs and cam_vs are not needed. This reduces the number of wires required for a BT.656 video interface.

There are two timing reference codes: The start of active video (SAV) reference code precedes each video data block, and the end of active video (EAV) follows each video block. Each timing reference signal consists of a 4-byte sequence in the following hexadecimal format: **FF 00 00 XY**. The first 3 bytes are a fixed preamble (See the ITU-R BT.656 specification). The fourth byte (XY) contains information defining field identification (F), blanking (V), and SAV/EAV information (H), and 4 parity bits calculated as a function of F, V, and H (see the ITU-R BT.656 specification).

Table 12-3 lists the video timing reference codes for SAV and EAV.

Table 12-3. Video Timing Reference Codes for SAV and EAV

Data Bit Number	First Word (FF)	Second Word (00)	Third Word (00)	Fourth Word (XY)
9 (MSB)	1	0	0	1
8	1	0	0	F
7	1	0	0	V
6	1	0	0	H
5	1	0	0	P3
4	1	0	0	P2
3	1	0	0	P1
2	1	0	0	P0
1	1	0	0	0
0	1	0	0	0

Table 12-4 contains a description of the F, V, and H signals.

Table 12-4. F, V, H Signal Descriptions

Signal	Value	Command
F	0	Field 1
	1	Field 2
V	0	0
	1	Vertical blank
H	0	SAV
	1	EAV

The resulting Hamming distance between any two code words is four, allowing two error detections and one error correction. To enable or disable the error-correcting capability, configure the CCDC_REC656IF [1] ECCFVH bit.

Note: The 2-bit errors are detected, but not flagged or corrected. Errors of more than 2 bits are not corrected or flagged.

Table 12-5 lists the F, V, and H protection (error-correction) bits.

Table 12-5. F, V, H Protection (Error-Correction) Bits

F	V	H	P3	P2	P1	P0
0	0	0	0	0	0	0
0	0	1	1	1	0	1
0	1	0	1	0	1	1
0	1	1	0	1	1	0
1	0	0	0	1	1	1
1	0	1	1	0	1	0
1	1	0	1	1	0	0
1	1	1	0	0	0	1

When operating in CCIR-656 mode, data is stored in SDRAM according to the format shown in Table 12-6 when CCDC_SYN_MODE [11] PACK8 is enabled.

Table 12-6. BT.656 Mode Data Format in SDRAM

8 bit x 4	Pixel3 (Y1/Cr0)	Pixel2 (Cr0/Y1)	Pixel1 (Y0/Cb0)	Pixel0 (Cb0/Y0)
	Bit 31			Bit 0

Note: The CCDC outputs the XY code in the SAV and EAV into memory. To eliminate this, users must set the SPH register field to +1. In addition, the NPH register field must be set to accurately represent the number of active pixels.

12.2.4.4 Camera Serial Interface (CSI1) Protocol and Data Formats

This section describes CSI1 protocol and data formats. Table 12-7 describes the I/O for serial interface CSI1.

Table 12-7. I/O Description for Serial Interface CSI1

Signal Name		I/O(1)	Description	Value at Reset
csib_datp	CAM_S_DATA	I	Serial data input	N/A
csib_datn				
csib_strbp	CAM_S_CLK	I	Serial clock input	N/A
csib_strbn				

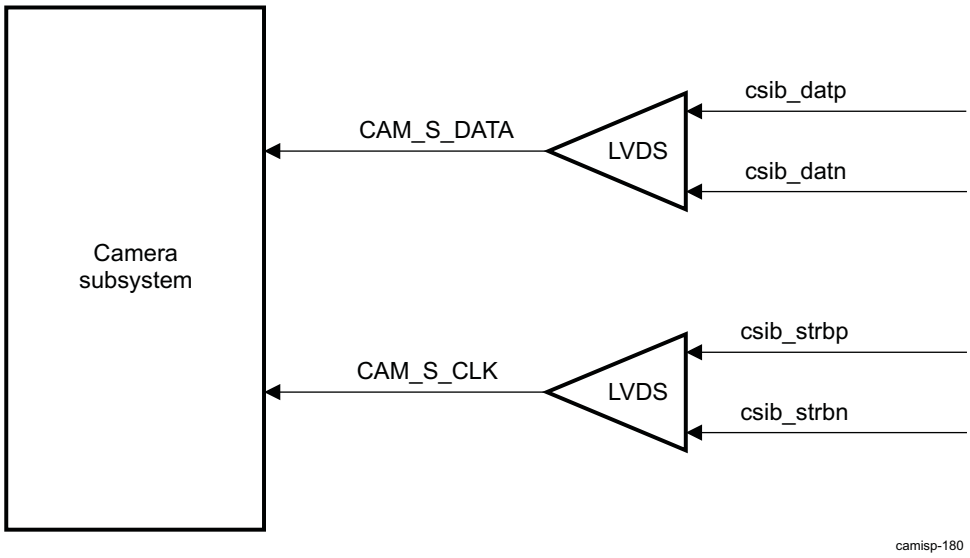
(1) I = Input, O= Output

From the device point of view, the CSI1 interface comprises four differential inputs representing two effective signals: the serial data and the clock.

From the Camera ISP side, the CSI1 interface comprises two serial inputs (CAM_S_DATA and CAM_S_CLK) that are the result of low-voltage differential signal (LVDS) converters, as shown in Figure 12-10.

Figure 12-10 shows all possible interfaces of the Camera ISP in the CSI1 serial configuration.

Figure 12-10. CSI1 Receiver Configuration Interface Signals



Note: The complex I/O between the CSI1 receiver and an external camera can be configured to transfer data in serial or in parallel mode. The scheme in Figure 12-10 represents the serial setting: CSI1_CTRL [2] IO_OUT_SEL = 0x0. This bit must be set to 0x1 to transfer data at the output of the complex I/O in parallel. See Section 12.4.2.2.2, *Complex I/O*, for details.

The CSI1 receiver is a serial interface to an image sensor. Signals from the camera are a serial input data (CAM_S_DATA) and a clock signal (CAM_S_CLK).

Each frame is clearly identified by four unique 32-bit synchronization codes: frame start, frame end, line start, and line end, which are embedded in the serial bitstream. The logical channel identification number is also encoded in the synchronization codes.

- Frame-start code (FSC): Identifies the start of a new frame
- Line-start code (LSC): Identifies the start of a new line and is received for every line, except the first line, that starts with an FSC
- Line-end code (LEC): Identifies the end of a line and is received for every line, except the last line, that ends with an FEC
- Frame-end code (FEC): Identifies the end of the last line and the end of the current frame

Table 12-8 summarizes the 32-bit synchronization code values.

Table 12-8. Synchronization Codes

Synchronization Code	Value	Comments
LSC	0xFF00:0000	Aligned on a 32-bit boundary Not used in case of JPEG8 data
LEC	0xFF00:0001	Aligned on a 32-bit boundary Not used in case of JPEG8 data
FSC	0xFF00:0002	Aligned on a 32-bit boundary
FEC	0xFF00:0003	Aligned on a 32-bit boundary

The CSI1 receiver interface has several image-data operating modes, summarized in Table 12-9.

Table 12-9. CSI1 Image Data Operating Modes and Alignment Constraints

CSI1_CTRL 1[7:3] Format	CSI1 Data Format	OCP Bits per Pixel (bpp) (when sending data to memory, N/A when sending to VP)	Width Constraint: Must Be a Multiple of n Pixels	Storage Increase Versus Packed	2D Mode Availability[(1)]	Comments
0x0	YUV422 big endian	16	8	N/A	Yes	
0x1	YUV422 little endian	16	8	N/A	Yes	
0x2	YUV420	12	32	N/A	No	
0x3	YUV4:2:2 + VP	16	2	N/A	N/A	
0x3	RAW8 + VP	N/A, data are sent to VP	4	N/A	N/A	
0x4	RGB444 + EXP16	16	8	N/A	Yes	
0x5	RGB565	16	8	N/A	Yes	
0x6	RGB888	24	16	N/A	No	
0x7	RGB888 + EXP32	32	4	N/A	Yes	
0x8	RAW6 + EXP8	8	16	33%	No	
0x9	RAW6 + DPCM10 + EXP16	16	8	167%	No	DPCM decompression
0xA	RAW6 + DPCM10 + VP	N/A, data are sent to VP	16	N/A	N/A	DPCM decompression
0xB	RAW10 -> RAW6 DPCM	6	64	40%	No	DPCM compression
0xC	RAW7 + EXP8	8	16	14%	No	
0xD	RAW7 + DPCM10 + EXP16	16	8	129%	No	DPCM decompression
0xE	RAW7 + DPCM10 + VP	N/A, data are sent to VP	32	N/A	N/A	DPCM decompression
0xF	RAW10 -> RAW6 DPCM + EXP8	8	16	20%	No	DPCM compression
0x10	RAW6	6	64	N/A	No	
0x10	RAW7	7	128	N/A	No	
0x10	RAW8	8	16	N/A	No	
0x11	RAW8 + DPCM10 + EXP16	16	8	100%	No	DPCM decompression
0x12	RAW8 + DPCM10 + VP	N/A, data are sent to VP	4	N/A	N/A	DPCM decompression
0x13	RAW10 -> RAW7 DPCM	7	128	30%	No	DPCM compression
0x14	RAW10	10	64	N/A	No	
0x15	RAW10 + EXP16	16	8	60%	No	
0x16	RAW10 + VP	N/A, data are sent to VP	16	N/A	N/A	
0x17	RAW10 -> RAW7 DPCM + EXP8	8	16	20%	No	DPCM compression
0x18	RAW12	12	32	N/A	No	
0x19	RAW12 + EXP16	16	8	33%	No	

(1) If 2D mode is available; there are no supplementary constraints on data width. 2D mode is not applicable when sending to video port (VP).

Table 12-9. CSI1 Image Data Operating Modes and Alignment Constraints (continued)

CSI1_CTRL 1[7:3] Format	CSI1 Data Format	OCP Bits per Pixel (bpp) (when sending data to memory, N/A when sending to VP)	Width Constraint: Must Be a Multiple of n Pixels	Storage Increase Versus Packed	2D Mode Availability[(1)]	Comments
0x1A	RAW12 + VP	N/A, data are sent to VP	8	N/A	N/A	
0x1B	RAW10 -> RAW8 DPCM	8	16	20%	No	DPCM decompression
0x1C	JPEG, 8-bit data	N/A	N/A	N/A	No	
0x1D	JPEG, 8-bit data + FSP	N/A	N/A	N/A	No	
0x1E	RAW10 -> RAW8	8	16	20%	No	Data right shift

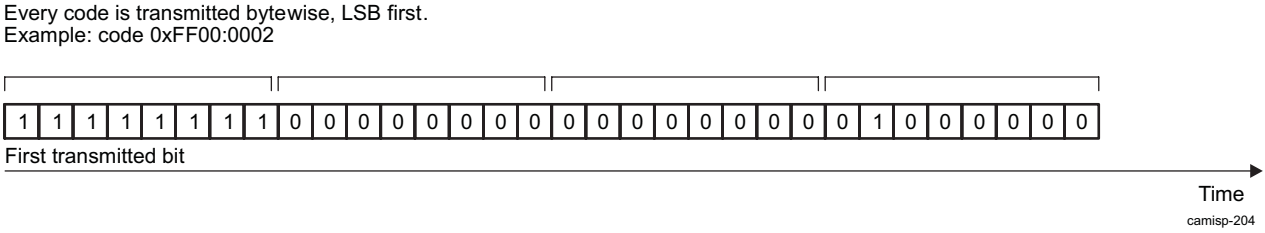
Notes:

- EXP8 = Data expansion to 8 bits, padding with zeros
- EXP16 = Data expansion to 16 bits, padding with alpha or zeros
  CSI1_CTRL1 [15:8] ALPHA can be used to set an alpha value.
  For RGB444 + EXP16:
  - data_out[31:28] = ALPHA[3:0]
  - data_out[15:12] = ALPHA[3:0]
- EXP32 = Data expansion to 32 bits, padding with alpha
  CSI1_CTRL1 [15:8] ALPHA can be used to set an alpha value.
  For RGB888 + EXP32: data_out [31:24] = ALPHA[7:0]
- FSP = False synchronization code protection decoding. Applies only to JPEG8 data format.
- VP = Output to the video-preprocessing hardware is enabled. The programmer must ensure that only one logical channel is enabled to the video preprocessing hardware. The behavior of the hardware is unpredictable if several logical channels to the video preprocessing hardware are enabled simultaneously.

The preamble 0xFF0000 is fixed by construction and must not be modified. However, the CSI1 receiver programming model allows the synchronization code identifier to be overwritten: bits 0 to 3.

Every code is transmitted bytewise, least-significant bit (LSB) first. For example, the code 0xFF00:0002 transmitted from the image sensor corresponds to the following bitstream: 11111111 - 00000000 - 00000000 - 01000000. Every default code starts with a set of eight 1s and sixteen 0s that are never received in pixel data. This means that content having eight 1s and sixteen 0s is not allowed. Figure 12-11 shows an example of 0xFF00 0002 transmission.

Figure 12-11. Example of 0xFF00 0002 Transmission



12.2.4.4.1 Pixel Data Format

This section summarizes how the CSI1 pixel data formats are transmitted over the serial interface and how the pixels are reconstructed, stored in memory, or passed to the video port.

The CSI1 receiver can cope with all data formats if the data line length sent through the CS1 physical protocol is a multiple of 32 bits. This condition is required for the CSI1 receiver to work correctly.

However, some data formats impose stronger line-length constraints to correctly finish pixel reconstruction at the end of the lines. If the additional constraints are not respected:

- Only the last reconstructed pixels in every line are erroneous. The missing bits are replaced with 0s to perform pixel reconstruction.
- The FW_IRQ interrupt is triggered.

12.2.4.4.1.1 YUV CSI1 Operating Modes

The YUV422 data format can be stored to memory in little- or big-endian format. The line length sent through the CSI1 physical protocol must be a multiple of 32 bits.

YUV422 data format can also be sent to the video port.

YUV422 + VP is used to output RAW8 data to the video port: YUV422 + VP is equivalent to RAW8 + VP. Figure 12-12 and Figure 12-13 show big-endian and little-endian YUV422 format, respectively.

Figure 12-12. YUV422 Big Endian

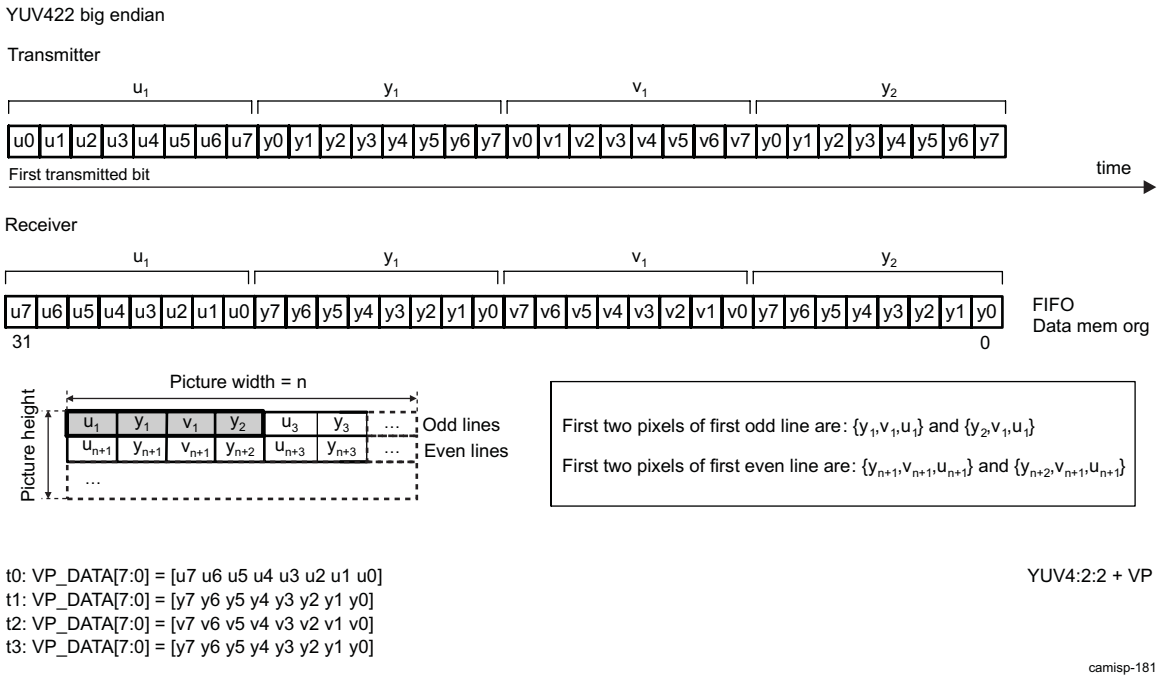
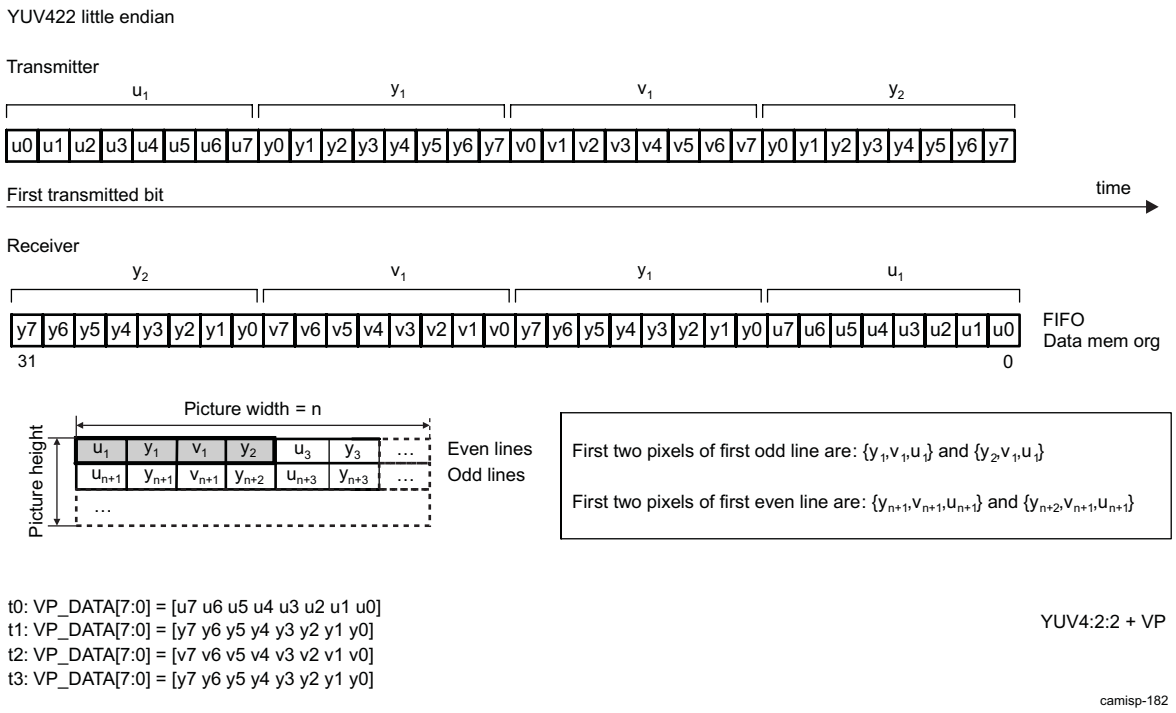


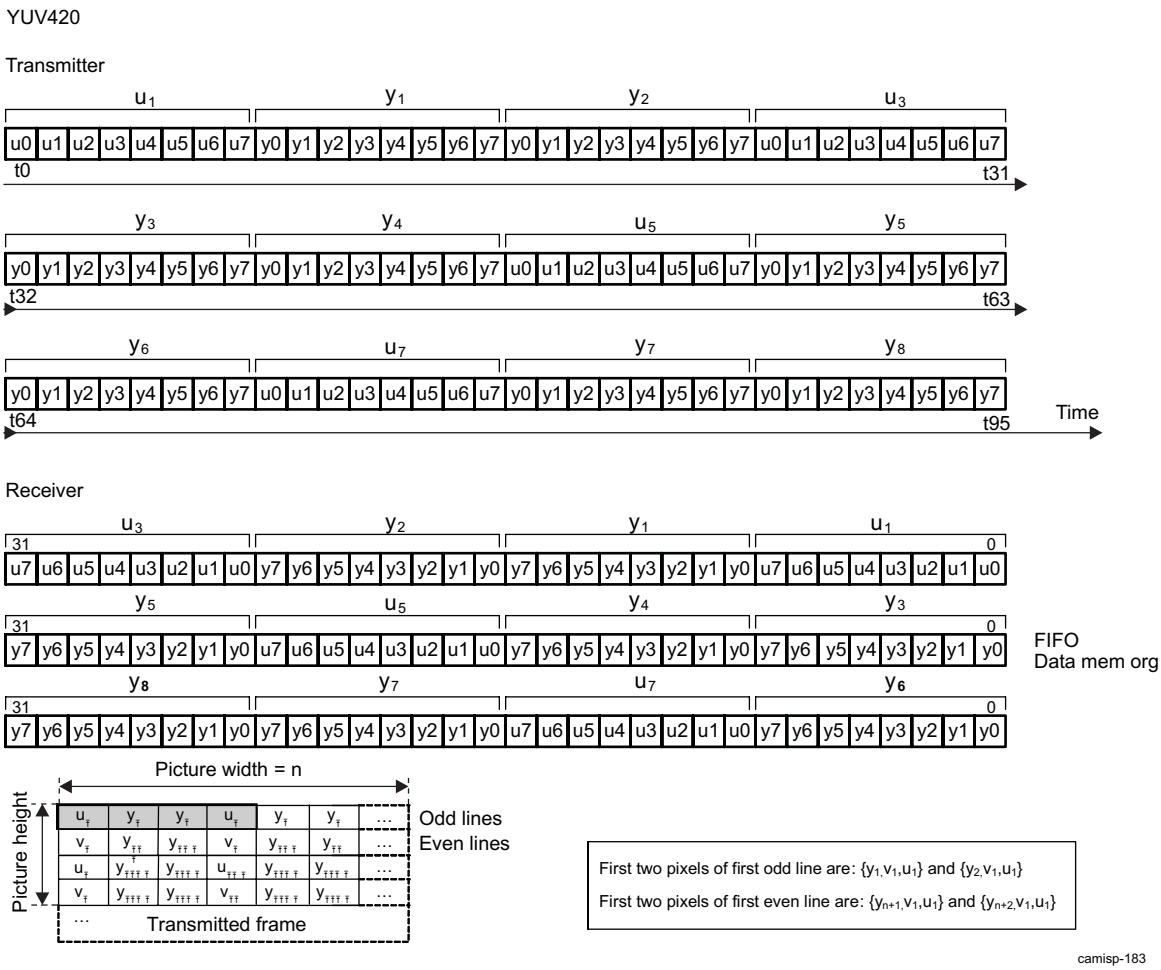
Figure 12-13. YUV422 Little Endian



The line length sent through the CSI1 physical protocol is a multiple of 32 bits. Furthermore, the line length is a multiple of 3 x 32 bits and the number of lines is even to correctly finish the pixel reconstruction.

The line structure is different for odd and even lines. Odd lines transport the U component, while even lines contain the V component. This is shown in Figure 12-14.

Figure 12-14. YUV420



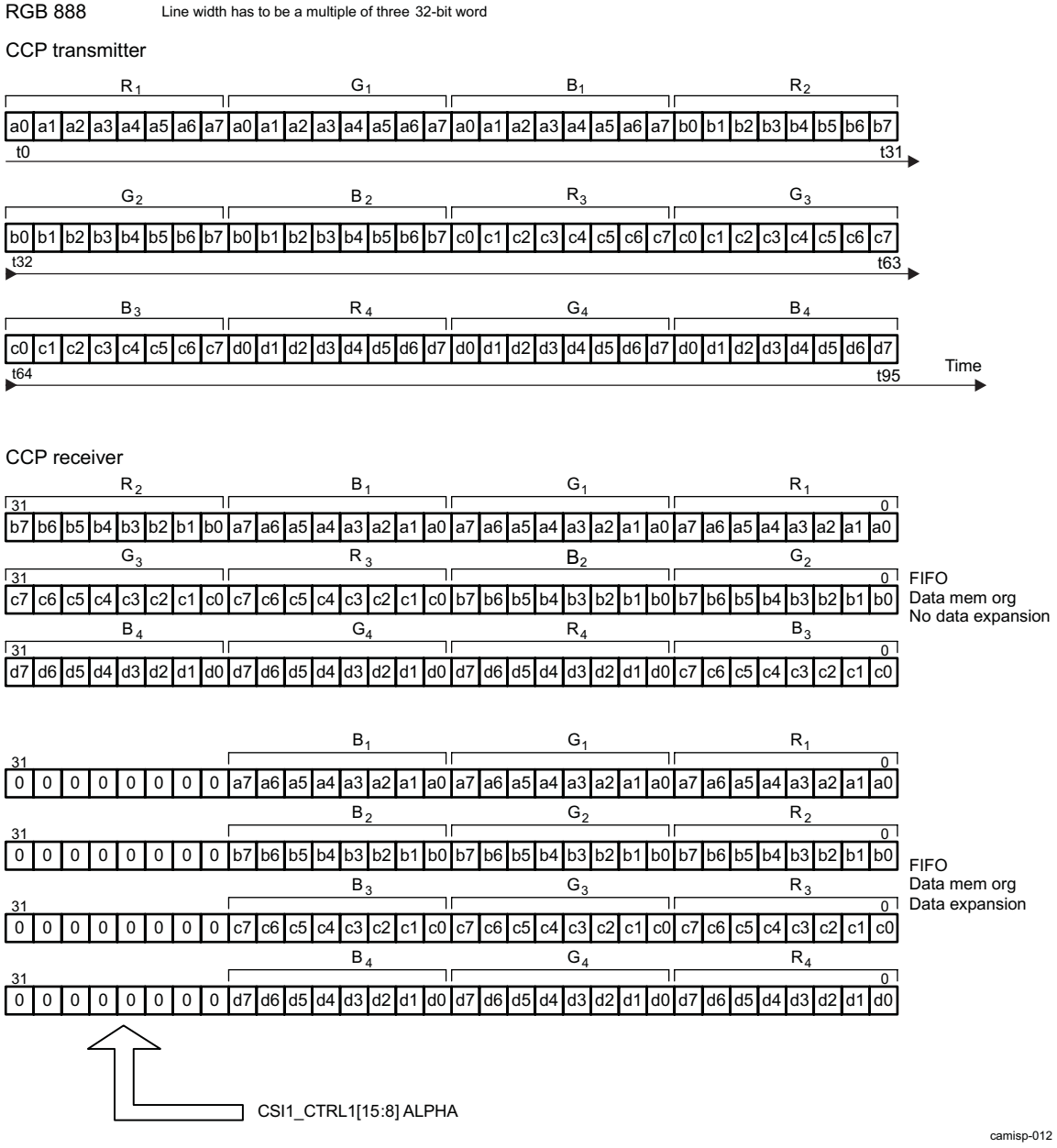
12.2.4.4.1.2 RGB CSI1 Operating Modes

RGB888 data format can be output to memory in two formats: with no data expansion and with data expansion.

If data expansion is used, the value of the 8 upper bits is programmable and can be set with an alpha value for computer graphics applications. The line length sent through the CSI1 physical protocol is a multiple of 32 bits. Furthermore, the line length is a multiple of 3 x 32 bits to correctly finish pixel reconstruction.

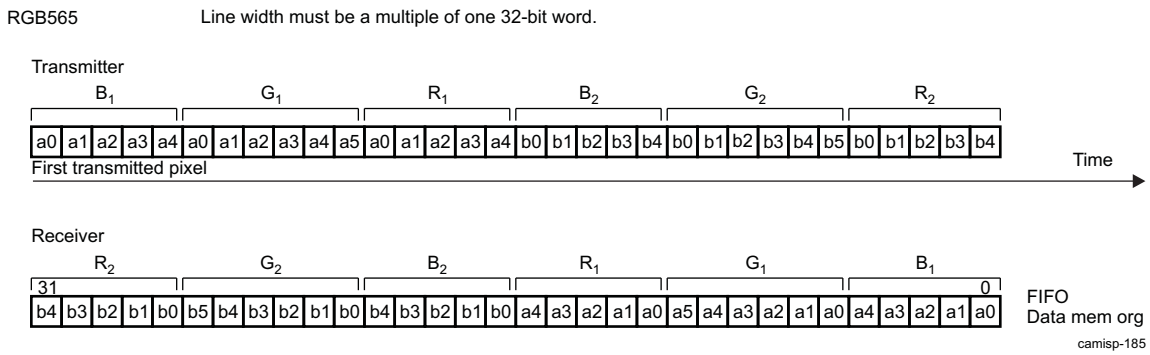
Figure 12-15 shows an example of RGB888 format.

Figure 12-15. RGB888



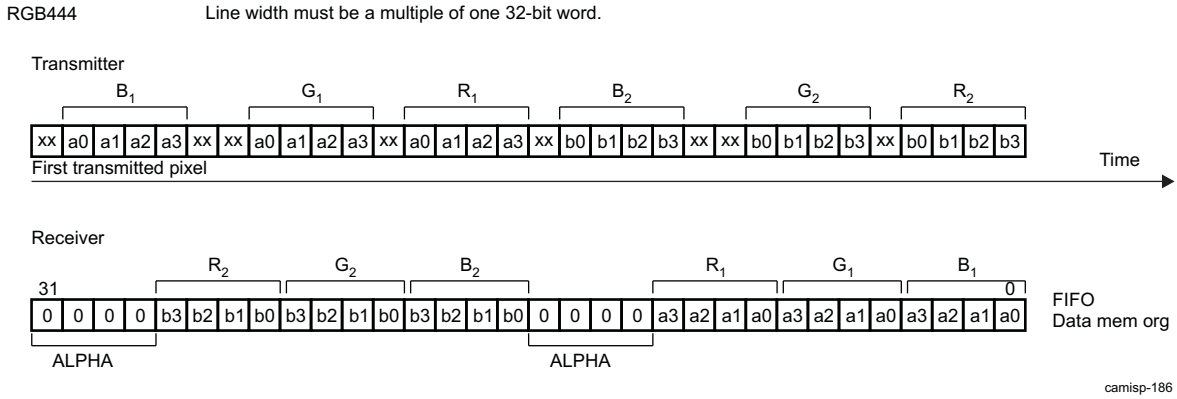
For RGB565, the line length sent through the CSI1 physical layer is a multiple of 32 bits (see Figure 12-16).

Figure 12-16. RGB565



RGB444 data format is output to memory with data expansion. If data expansion is used, the value of the 4 upper bits is programmable and can be set with an alpha value for computer graphics applications. The line length sent through the CSI1 physical layer is a multiple of 32 bits (see Figure 12-17).

Figure 12-17. RGB444



12.2.4.4.1.3 RAW Bayer RGB CSI1 Operating Modes

12.2.4.4.1.3.1 RAW8

RAW8 data format can be output to memory in two formats: with no data expansion and with data expansion.

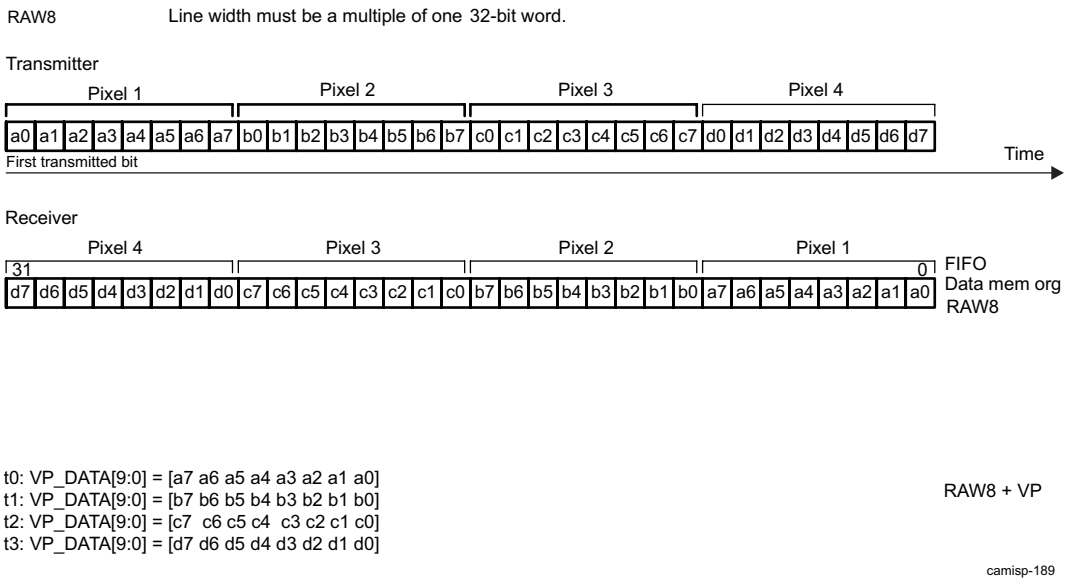
The line length sent through the CSI1 physical layer is a multiple of 32 bits.

Figure 12-18 shows RAW8 format.

Notes:

- Use YUV422 + VP to output RAW8 data to the video port: YUV422 + VP is equivalent to RAW8 + VP.

Figure 12-18. RAW8



12.2.4.4.1.3.2 RAW10

RAW10 data format can be output to memory in two formats: with no data expansion and with data expansion.

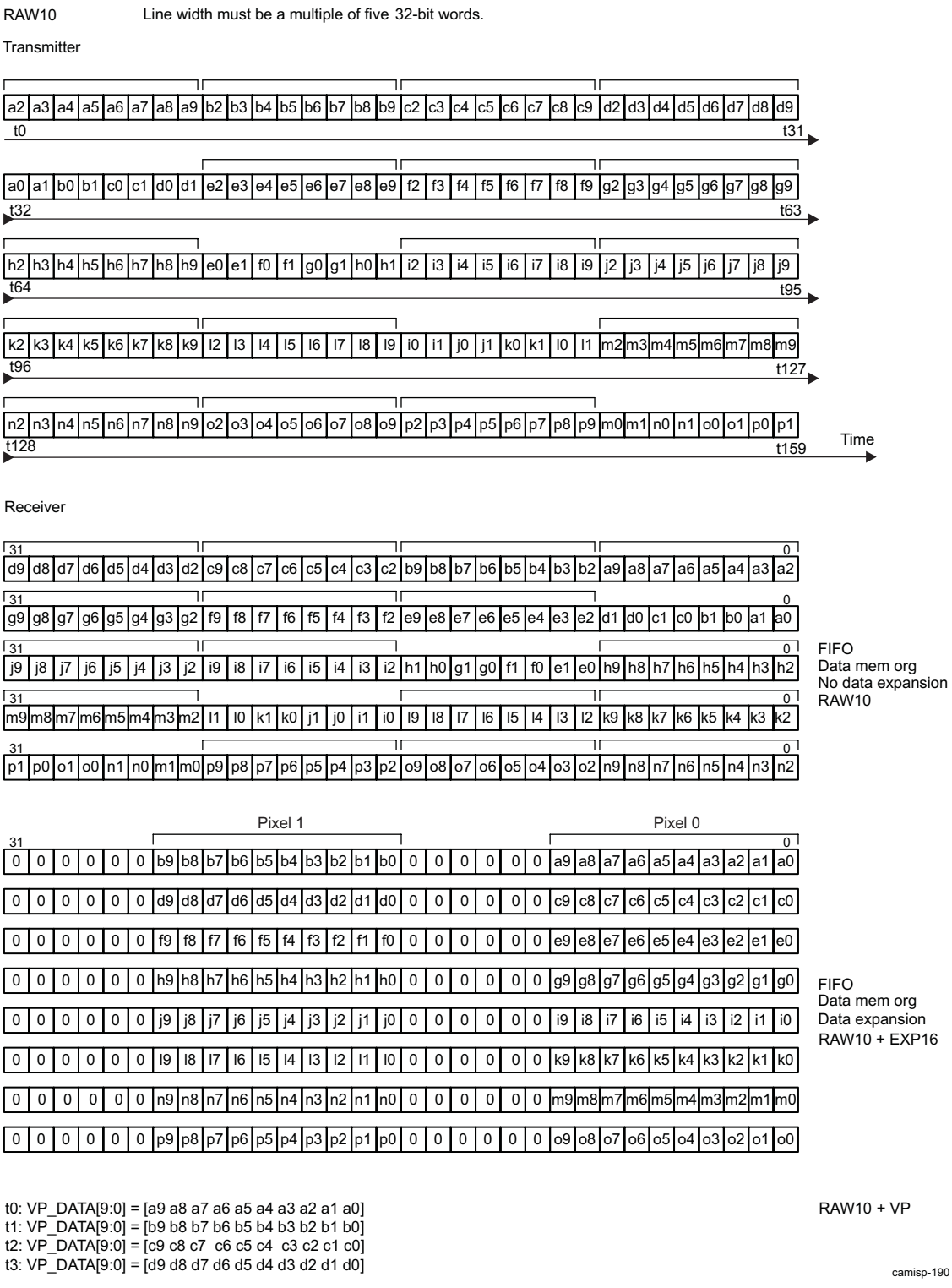
If data expansion is used, the 10-bit data are padded with 0s on a 16-bit word.

The line length sent through the CSI1 physical layer is a multiple of 32 bits. Furthermore, the line length is a multiple of 5 x 32 bits to correctly finish pixel reconstruction (the lowest common multiple of 32 and 10 is 320: 10 x 32 bits).

RAW10 data format can be sent to the video port.

Figure 12-19 shows RAW10 format.

Figure 12-19. RAW10



12.2.4.4.1.3.3 RAW12

RAW12 data format can be output to memory in two formats: with no data expansion and with data expansion.

If data expansion is used, the 12-bit data are padded with 0s on a 16-bit word.

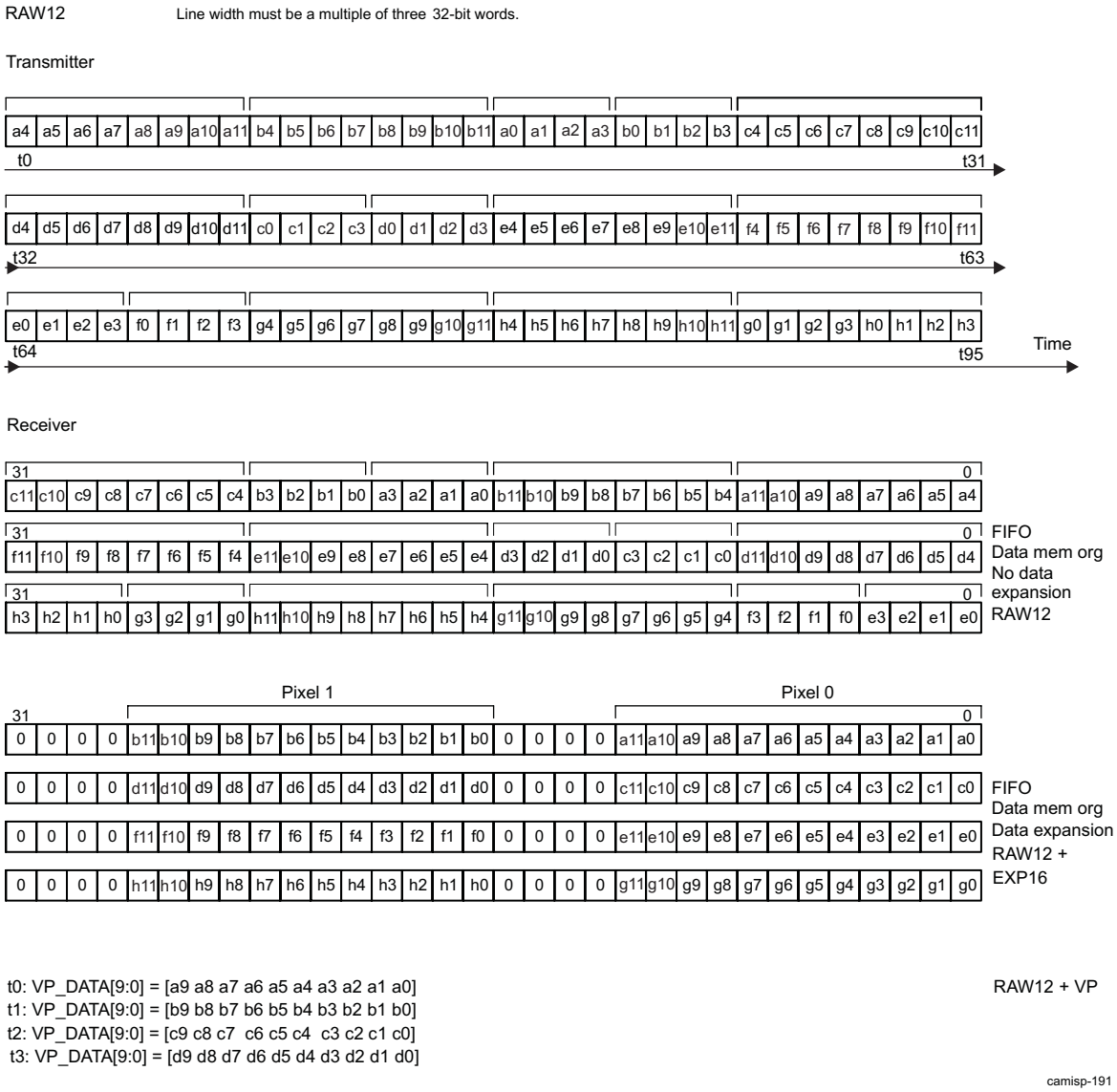
The line length sent through the CSI1 physical layer is a multiple of 32 bits. Furthermore, the line length is a multiple of 3 x 32 bits to correctly finish pixel reconstruction (the lowest common multiple of 32 and 12 is 96: 3 x 32 bits).

RAW12 data format can be sent to the video port.

Figure 12-20 shows RAW12 format.

Note: The hardware pipe is 10-bit only. Typically, the data lane shifter must be used to perform 10-bit-only processing.

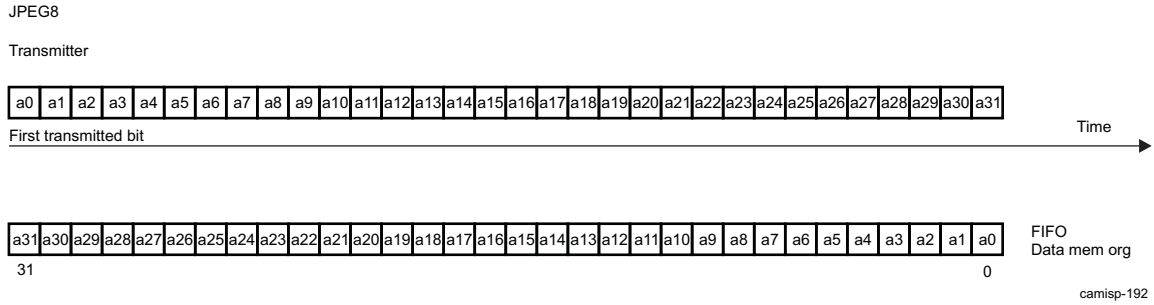
Figure 12-20. RAW12



12.2.4.4.1.4 JPEG8 CSI1 Operating Modes

The line length sent through the CSI1 physical layer is a multiple of 32 bits. The false synchronization protection (FSP) code insertion on the transmitter side automatically ensures this line length. It is impossible to know in advance the size of a compressed stream. Figure 12-21 shows JPEG8 and JPEG8 FSP format.

Figure 12-21. JPEG8 and JPEG8 FSP



In JPEG8 mode, the JPEG encoder on the sensor side must avoid generating data equal to the synchronization code and must deliver a synchronization-code-free bitstream.

In JPEG8 FSP mode, the bitstream is a standard JPEG8 bitstream in which byte-stuffing is used to differentiate synchronization code from original data with the same value. Thus, the JPEG encoder can generate a standard bitstream, and a postprocessing stage (FSP encoder) is used to highlight natural bitstream data equal to synchronization codes, so that the other end can discard the false synchronization codes.

FSP decoding is performed for byte-aligned data and occurs after the frame-start synchronization code is received. If the byte n = 0x0, the FSP decoder looks for bytes 1 and 2. If the bytes are equal to an illegal combination (synchronization codes), the byte 0xA5 that comes after (byte n + 1) is removed from the bitstream.

The 0xA5 padding bytes at the end of the frame are also removed by FSP decoding so that padding does not generate additional data in the FIFO. FSP decoding is then transparent to the software.

Normally, the module detects the illegal combination and then the 0xA5. When the 0xA5 is corrupted or replaced by another code, the value is automatically removed from the bitstream and an interrupt is generated to signal that the code is corrupt and therefore the received data are suspicious. If the line is too noisy for FSP decoding, it is better to configure the subsystem in JPEG8 and use software postprocessing.

12.2.4.5 Camera Serial Interface (CSI2) Protocol and Data Format

The CSI2 supports MIPI CSI2. This section describes MIPI CSI2 protocol and data formats.

12.2.4.5.1 Physical Layer

Table 12-10 lists the CSI2 receiver I/O.

Table 12-10. I/O Description for CSI2 Serial Interface

Signal Name		I/O[(1)]	Description	Value at Reset
csi2_dx0	line 1	I	Serial data/clock input	N/A
csi2_dy0				
csi2_dx1	line 2	I	Serial data/clock input	N/A
csi2_dy1				
csi2_dx2	line 3	I	Serial data/clock input	N/A
csi2_dy2				

(1) I = Input

Note: Each serial line (line 1, line 2, and line 3) can be used as clock lane or data lane. The MIPI CSI2 protocol requires one clock line (others are data lane or unused lane).

From the device point of view, the CSI2 interface comprises six differential input signals representing three lanes: the serial clock and one or two serial data.

Lanes support four operating modes:

- HS mode: High-speed transmit mode
- LP mode: Low-power transmit mode
- ULPM: Ultra-low power mode used between two transmissions
- Off mode: Lane is off.

12.2.4.5.2 Lane Merger

The layer consists of lane merger logic to merge the incoming serial stream into a byte stream. The bits are sent with the LSB first. The order of the lanes at the CSI2 receiver core depends on the lane configuration.

The number of lanes and their configuration can be changed only in ULPM or when all data lanes are in off mode.

The lane merger can merge two lanes into a single byte stream.

In case of a single lane, the lane merger is not used to merge byte streams.

Figure 12-22 and Figure 12-23 show the byte position into each serial link for 1 and 2 data lane configurations. The byte stream always starts from lane 1. It finishes on one of the lanes, depending on the number of bytes to receive and the number of lanes.

Figure 12-22. Two Data-Lane Merger Configuration

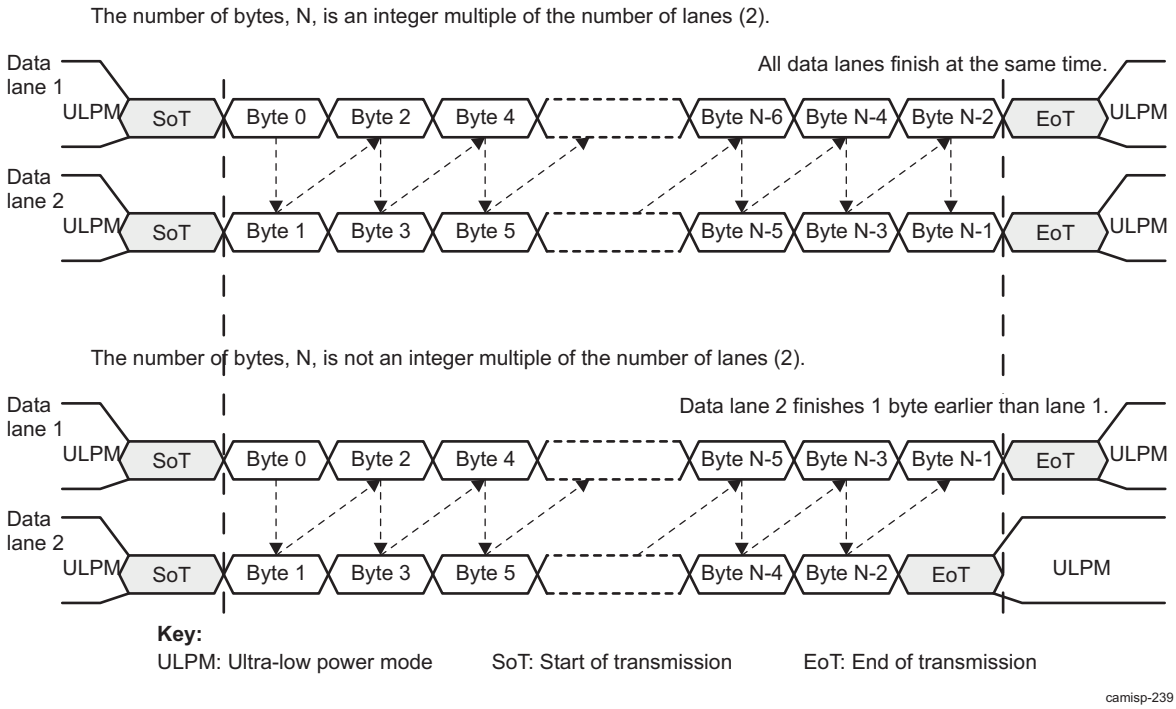
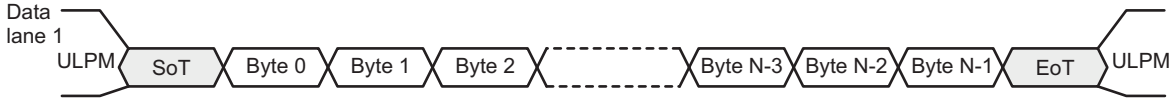


Figure 12-23. One Data-Lane Configuration



Key:
ULPM: Ultra-low power mode SoT: Start of transmission EoT: End of transmission

camisp-240

12.2.4.5.3 Protocol Layer

The low-level protocol (LLP) is a byte-oriented protocol from the lane merger layer. It supports short and long packet formats.

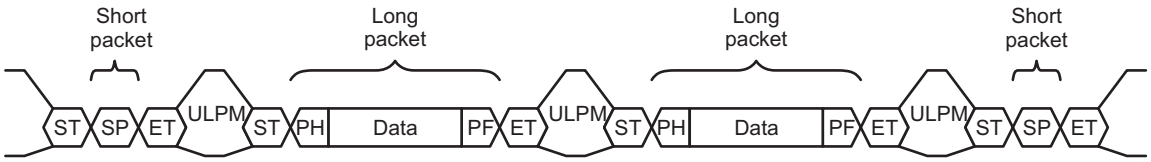
The CSI2 protocol layer defines how image-sensor data is transported onto the physical layer.

The feature set of the protocol layer implemented by the CSI2 receiver is:

- Transport of arbitrary data (payload-independent)
- 8-bit word size
- Support for up to four interleaved virtual channels on the same link
- Special packets for frame-start, frame-end, line-start, and line-end information
- Descriptor for the type, pixel depth, and format of application-specific payload data
- Error-correction code (ECC) for 1-bit error correction or 2-bit error detection in the header
- 16-bit checksum code for payload error detection

Figure 12-24 shows the CSI2 protocol layer with short and long packets.

Figure 12-24. CSI2 Protocol Layer With Short and Long Packets



Key:

ST: Start of transmission	ET: End of transmission
PH: Packet header	PF: Packet footer
ULPM: Ultra-low power mode	SP: Short packet

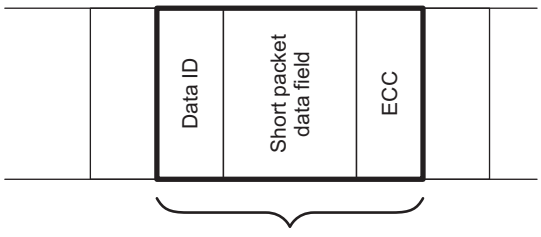
camisp-241

Two packets are always separated from each other with a sequence of a ULPM, an ET, and an ST.

12.2.4.5.3.1 Short Packet

A short packet is identified by data types 0x00 to 0x0F. A short packet can be used for frame or line synchronization or for generic data. Figure 12-25 shows the structure of a short packet.

Figure 12-25. Short Packet Structure



camisp-242

For frame-synchronization data types, the short packet data field is the frame number. For line-synchronization data types, the short packet data field is the line number. For generic short packet data types, the content of the short packet data field is user-defined.

The 16-bit frame number, when used, is always nonzero to distinguish it from the use case where the frame number is inoperative and remains set to 0. The behavior of the 16-bit frame number is one of the following:

- The frame number is always 0. The frame number is inoperative.
- The frame number increments by 1 for every FS packet with the same virtual channel and is periodically reset to 1 (1, 2, 1, 2, 1, 2, 1, 2 or 1, 2, 3, 4, 1, 2, 3, 4).

For line-start code (LSC) and line-end code (LEC) synchronization packets, the short packet data field contains a 16-bit line number. This line number is the same for the LS and LE packets corresponding to a given line. Line numbers are logical line numbers and do not necessarily equal physical line numbers. The 16-bit line number, when used, is always nonzero to distinguish it from the case where the line number is inoperative and remains set to 0.

The behavior of the 16-bit line number is one of the following:

- The line number is always 0. The line number is inoperative.
- The line number increments by one for every LS packet within the same virtual channel and the same data type. The line number is periodically reset to 1 for the first LS packet after an FS packet. The intended usage is for progressive scan (non-interlaced) video data streams. The line number must be a nonzero value.
- The line number increments by the same arbitrary step value greater than one for every LS packet within the same virtual channel and the same data type. The line number is periodically reset to a nonzero arbitrary start value for the first LS packet after an FS packet. The arbitrary start value can be different between successive frames. The intended usage is for interlaced video data streams.

The ECC byte allows single-bit errors to be corrected and 2-bit errors to be detected in the short packet.

Short packets apply to all contexts using the same virtual channel ID (there are up to eight contexts to support eight dedicated configurations of virtual channel ID and data types). The data type associated with the context is not used to distinguish which context is used when receiving short packets.

12.2.4.5.3.2 Long Packet

A long packet is identified by data types 0x10 to 0x37. A long packet consists of three elements: a 32-bit packet header (PH), an application-specific data payload with a variable number of 8-bit data words, and a 16-bit packet footer (PF). The packet header is further composed of three elements: an 8-bit data identifier, a 16-bit word count field, and an 8-bit ECC. The packet footer has one element, a 16-bit checksum. Figure 12-26 and Table 12-11 show the structure of a long packet.

Figure 12-26. Long Packet Structure

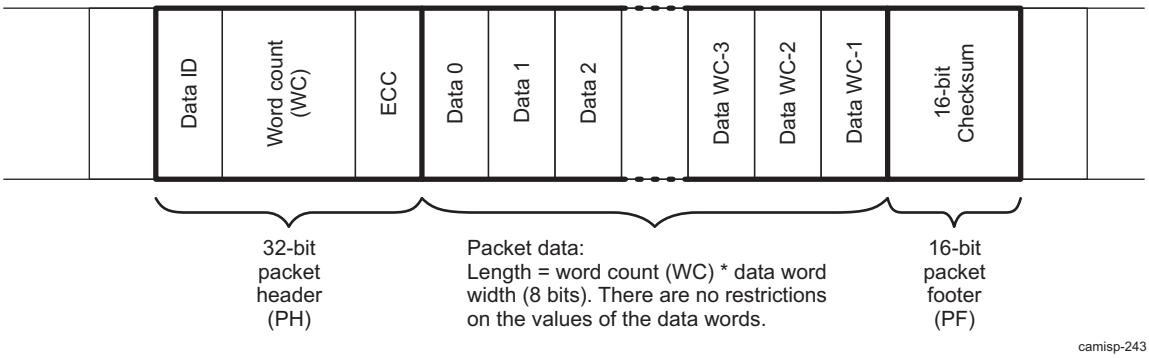


Table 12-11. Long Packet Structure Description

Packet Part	Field Name	Size (Bit)	Description
Header	Data ID	8	Contains the virtual channel identifier and the data-type information
	Word count	16	Number of data words in the packet data. A word is 8 bits.
	ECC	8	ECC for data ID and WC field. Allows 1-bit error recovery and 2-bit error detection.
Data	Data	WC * 8	Application-specific payload (WC words of 8 bits)
Footer	Checksum	16	16-bit cyclic redundancy check (CRC) for packet data

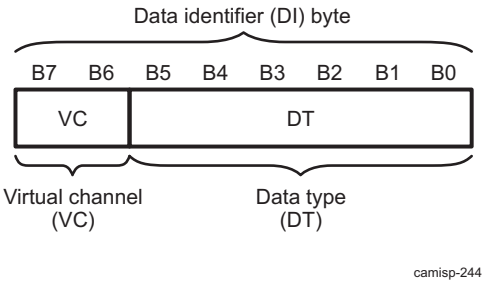
There are no restrictions on packet data size, but each data format can impose additional restrictions on the length of the payload data (for example, a multiple of 4 bytes).

12.2.4.5.3.3 Data Identifier

The data identifier byte contains the virtual channel identifier (VC) value and the data-type (DT) value, as shown in Figure 12-27. The VC is in the 2 MSBs of the data identifier byte. The DT value is in the 6 LSBs of the data identifier byte.

Figure 12-27 shows the data identifier structure.

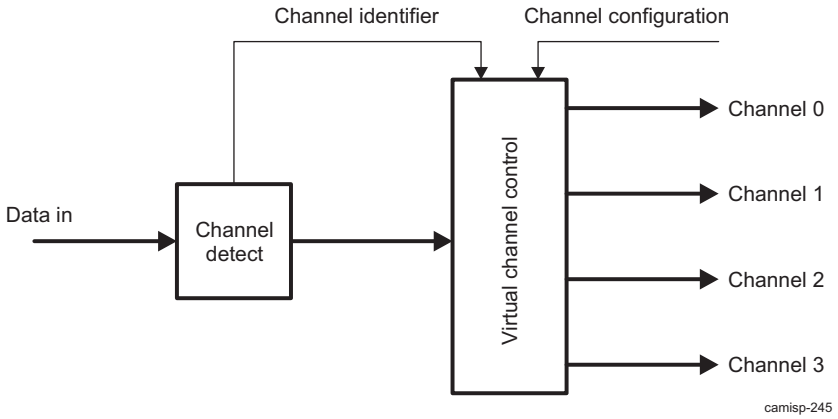
Figure 12-27. Data Identifier Structure



Virtual Channel

The CSI2 protocol layer transports virtual channels. Virtual channels are built up of frames. A frame can comprise embedded data and image-sensor data. Two contexts are used to send the two types of data separately. Each frame is identified by unique mandatory synchronization codes: frame start and frame end. The following synchronization codes are optional for the transmitter: line start and line end. A set of registers is associated with each context defined by the virtual channel ID and the data type. Figure 12-28 shows a virtual channel.

Figure 12-28. Virtual Channel



Pixel Formats

Image-sensor data can have multiple data types. Table 12-12 summarizes the pixel formats supported by the CSI2 receiver interface.

Table 12-12. CSI2 Pixel Format Modes

Mode	Description
YUV420 8-bit	YUV4:2:0 image data
YUV420 8-bit + VP	YUV4:2:0 image data
YUV420 10-bit	YUV4:2:0 image data
YUV420 8-bit legacy	YUV4:2:0 image data
YUV420 8-bit + CSPS	YUV4:2:0 image data
YUV420 10-bit + CSPS	YUV4:2:0 image data
YUV422 8-bit	YUV4:2:2 image data
YUV422 10-bit	YUV4:2:2 image data
RGB565	RGB565 image data
RGB888	RGB888 image data
RGB888 + EXP32	RGB888 image data
RGB666 + EXP32_24	RGB666 image data
RGB666 + EXP32	RGB666 image data
RGB444 + EXP16	RGB444 image data
RGB555 + EXP16	RGB555 image data
RAW6 + EXP8	RAW Bayer, 6-bit image data
RAW7 + EXP8	RAW Bayer, 7-bit image data
RAW8	RAW Bayer, 8-bit image data
RAW8 + VP	RAW Bayer, 8-bit image data
RAW10	RAW Bayer, 10-bit image data
RAW10 + EXP16	RAW Bayer, 10-bit image data
RAW10 + VP	RAW Bayer, 10-bit image data
RAW12	RAW Bayer, 12-bit image data
RAW12 + EXP16	RAW Bayer, 12-bit image data
RAW12 + VP	RAW Bayer, 12-bit image data
RAW14	RAW Bayer, 14-bit image data
RAW14 + EXP16	RAW Bayer, 14-bit image data
RAW14 + VP	RAW Bayer, 14-bit image data
JPEG, 8-bit data	JPEG8

For more information on how the data formats are transmitted and how the data are stored in memory, see Section 12.2.4.5.4, *CSI2 Operating Modes*.

12.2.4.5.3.4 Synchronization Codes

Data reception from the image-sensor module uses four synchronization codes embedded in the serial bit-stream:

- FSC: Identifies the start of a new frame
- LSC: Identifies the start of a new line; received for every line
- LEC: Identifies the end of a line; received for every line
- FEC: Identifies the end of the last line and the end of the current frame

Table 12-13 summarizes the synchronization code values.

Table 12-13. Synchronization Codes

Synchronization Code	Value	Comments
FSC	0x0	Mandatory
FEC	0x1	Mandatory
LSC	0x2	Optional
LEC	0x3	Optional
Reserved	0x4 to 0x7	Not used

12.2.4.5.3.5 Generic Short Packet Codes

When the synchronization code value is between 0x8 and 0xF, the short packet is called a generic short packet. Short packets are not processed by the camera interface hardware. A generic short packet is stored in a register without the ECC and an interrupt can be generated. Therefore, generic short packets must be handled by software.

12.2.4.5.3.6 Generic Long Packet Codes

The code value 0x10 indicates null packets, which can be received at any time. They are discarded by the protocol engine.

The code value 0x11 indicates blanking packets, which can be received at any time. They are discarded by the protocol engine.

The code value 0x12 indicates embedded 8-bit non-image data typically used for JPEG.

Code values from 0x13 to 0x17 are reserved.

12.2.4.5.3.7 Frame Structure

Each frame consists of short packets to indicate start of frame and end of frame. Optional short packets for start of line and end of line can be sent by the image sensor.

Some information before and after the picture data can be sent as start-of-frame (SOF) and end-of-frame (EOF) information by the image sensor to the memory through the L3 port.

For each frame, the pixel data (arbitrary data or user-defined byte data) are valid only after an SOF short packet. If the data are invalid, they are discarded by the protocol engine.

A frame comprises embedded data and image-sensor data. Figure 12-29 shows where the embedded data and image sensor data are in the frame. The following definitions apply:

- Zero or more SOF status lines (SOF lines) can be embedded at the beginning of a CSI2 frame.
- The image data comprises pixels of the same or different data formats.
- Zero or more EOF status lines (EOF lines) can be embedded at the end of a CSI2 frame.
- The SOF lines, pixel data, and EOF lines do not overlap.

The CSI2 receiver does not use the information in the status lines. However, it extracts it and stores it in memory for software use.

Because the data types are different, the CSI2 receiver uses a different context for embedded data and image-sensor data.

Embedded data is supported as a context by the CSI2 receiver; therefore, there is no specific hardware support for embedded data.

Figure 12-29. General Frame Structure (Informative)

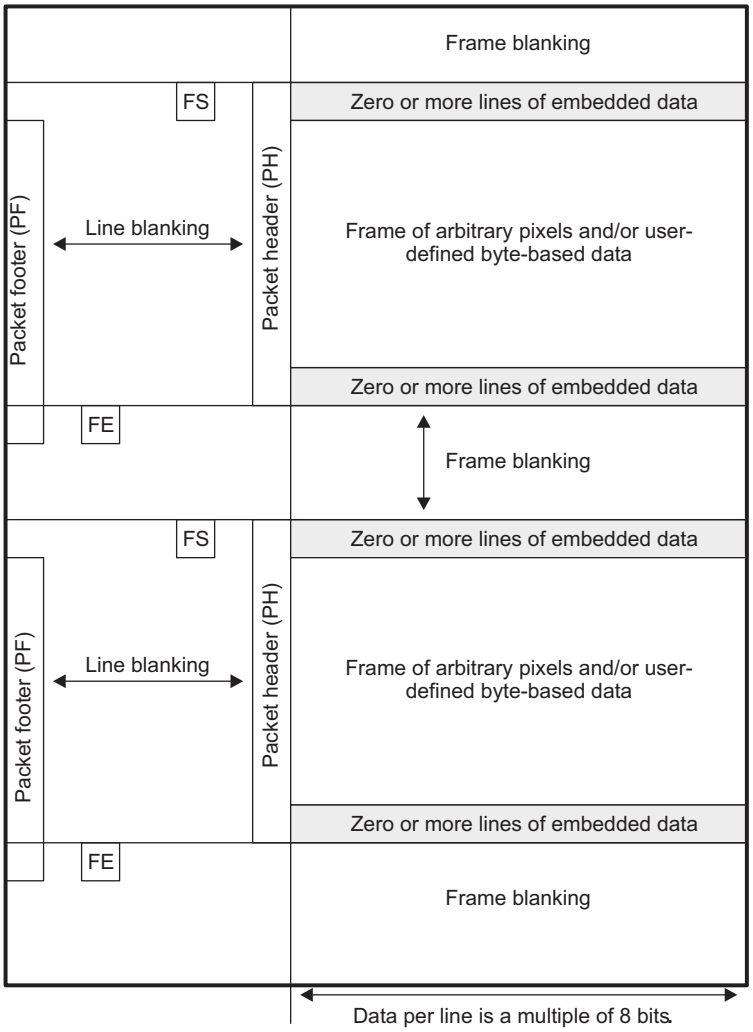
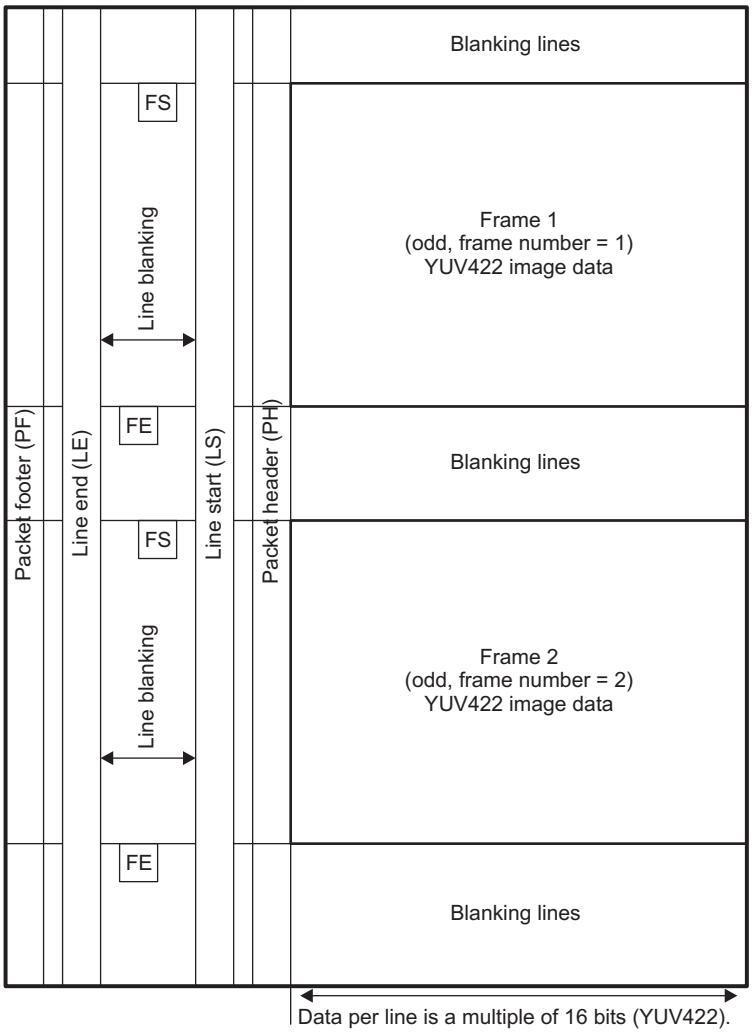


Figure 12-30 shows the frame structure of a YUV422 interlaced video frame without embedded data.

Figure 12-30. Digital Interlaced Video Frame (Informative)



The period between the LEC and the new LSC is the line blanking period. The time between the FEC and the new FSC is the frame blanking period. The receiver works with the line blanking period set to 0.

12.2.4.5.4 CSI2 Operating Modes

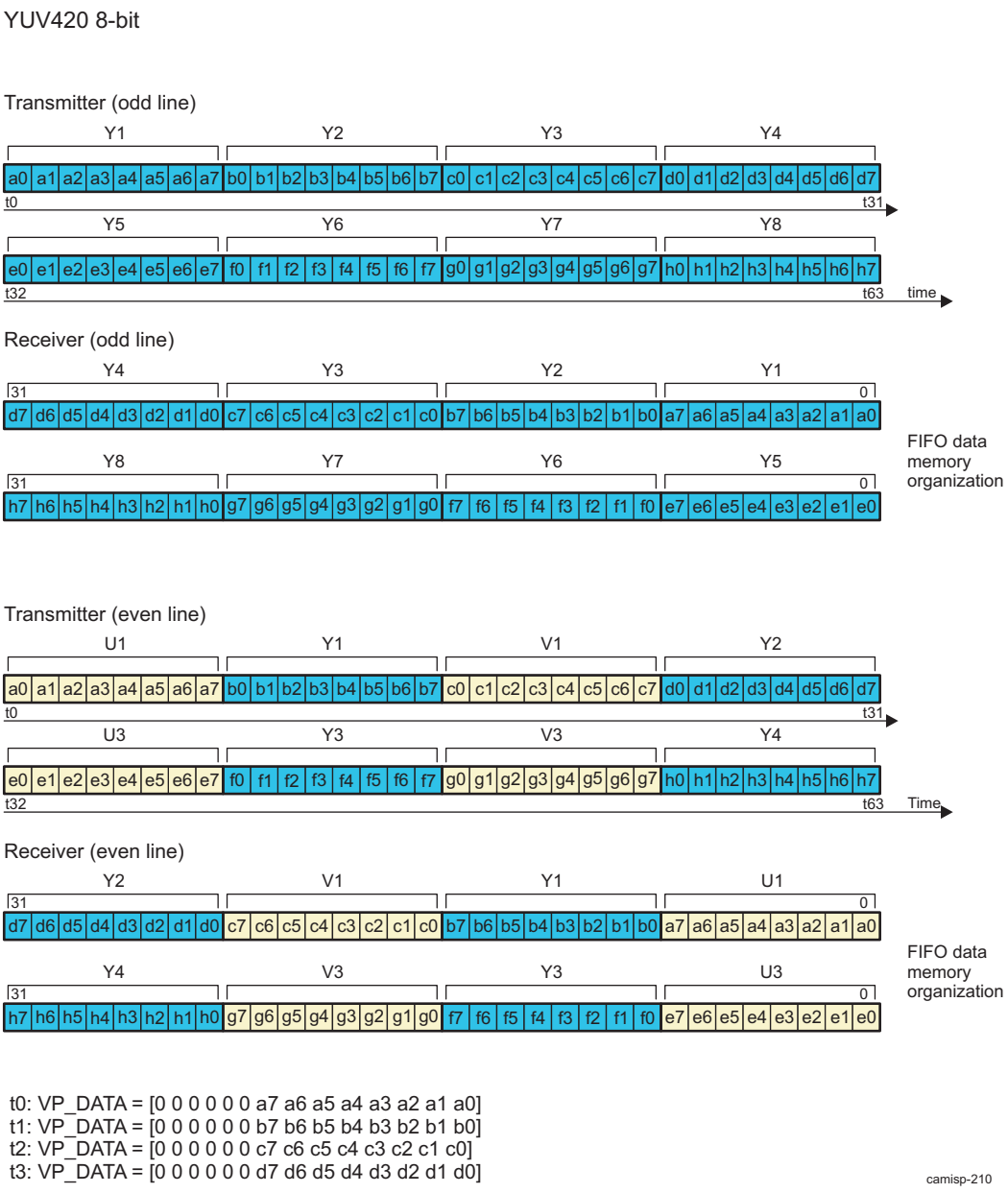
12.2.4.5.4.1 YUV Operating Modes

12.2.4.5.4.1.1 YUV420 8-Bit

YUV420 8-bit data can be stored to memory in little-endian format. The line length sent through the CSI2 physical protocol is a multiple of 16 bits for odd lines and 32 bits for even lines.

For correct pixel reconstruction, the line length must be a multiple of 3*32 bits and the number of lines must be even. Figure 12-31 shows the storage format for YUV420 8-bit data.

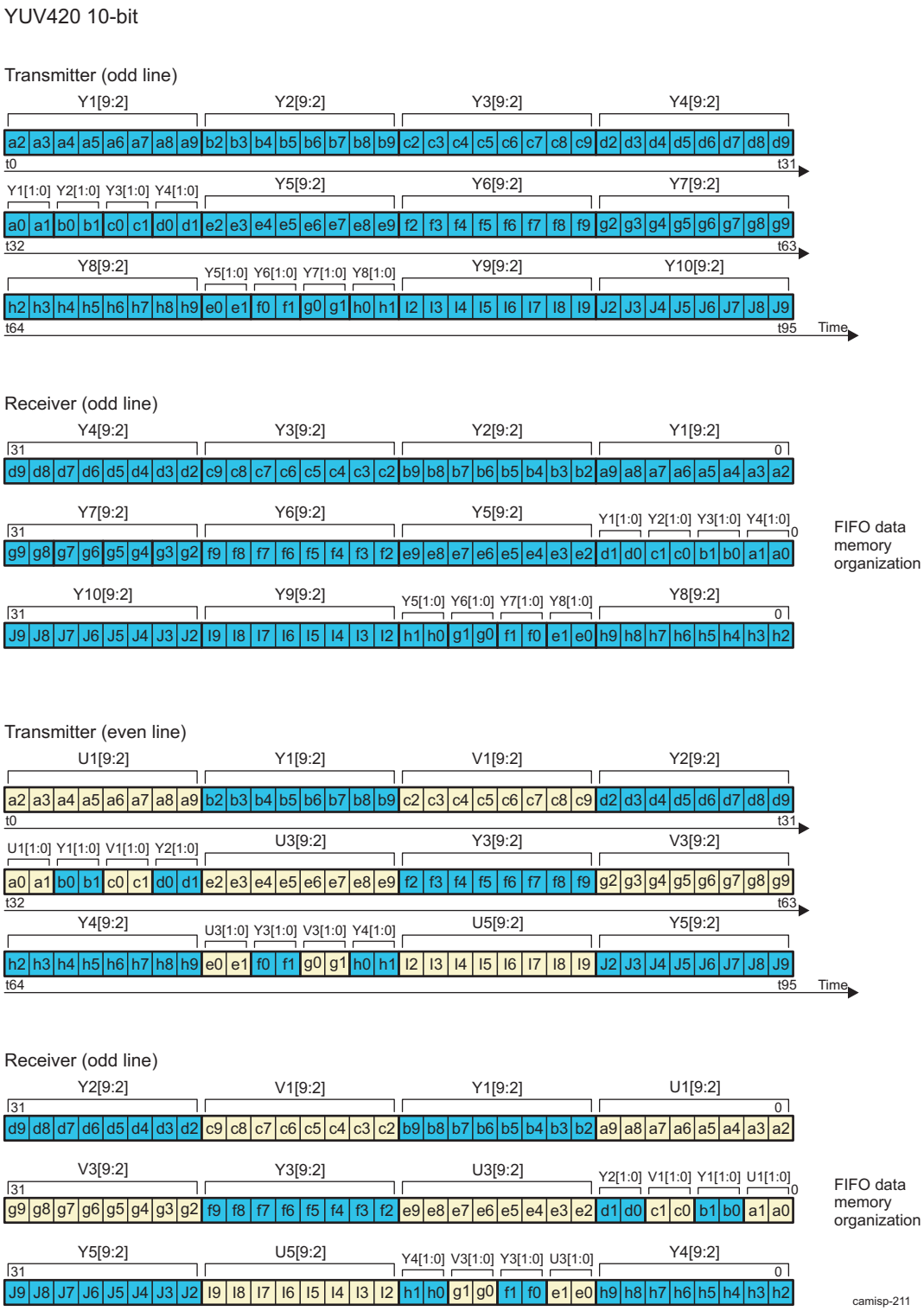
Figure 12-31. YUV420 8-Bit



12.2.4.5.4.1.2 YUV420 10-Bit

YUV420 10-bit data can be stored to memory in little-endian format. The line length sent through the CSI2 physical protocol is a multiple of 40 bits for odd lines and 80 bits for even lines. Figure 12-32 shows the storage format for YUV420 10-bit data.

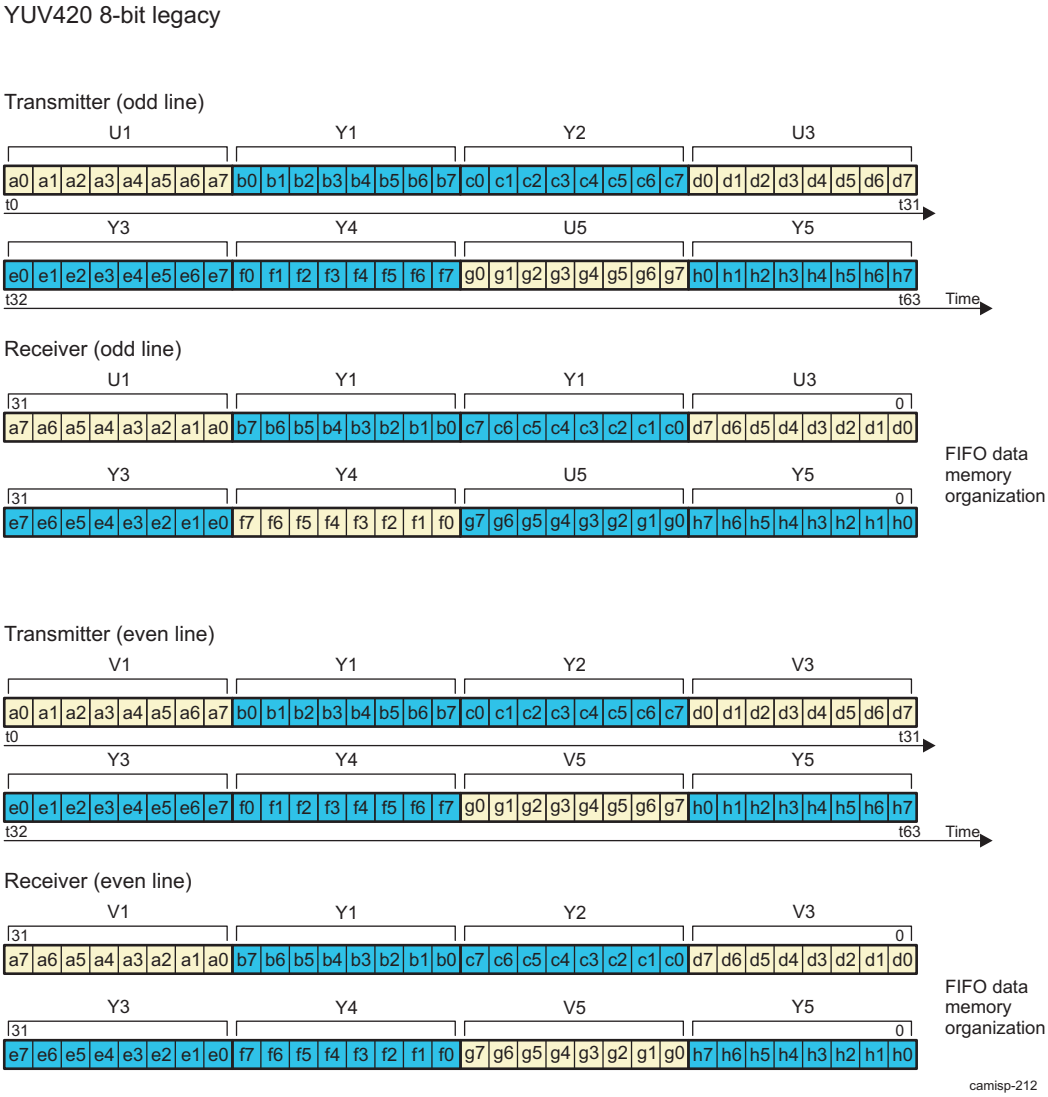
Figure 12-32. YUV420 10-Bit



12.2.4.5.4.1.3 YUV420 8-Bit Legacy

YUV420 8-bit legacy data can be stored to memory in big-endian format. The line length sent through the CSI2 physical protocol is a multiple of 4 bytes. Figure 12-33 shows the storage format for YUV420 8-bit legacy data.

Figure 12-33. YUV420 8-Bit Legacy

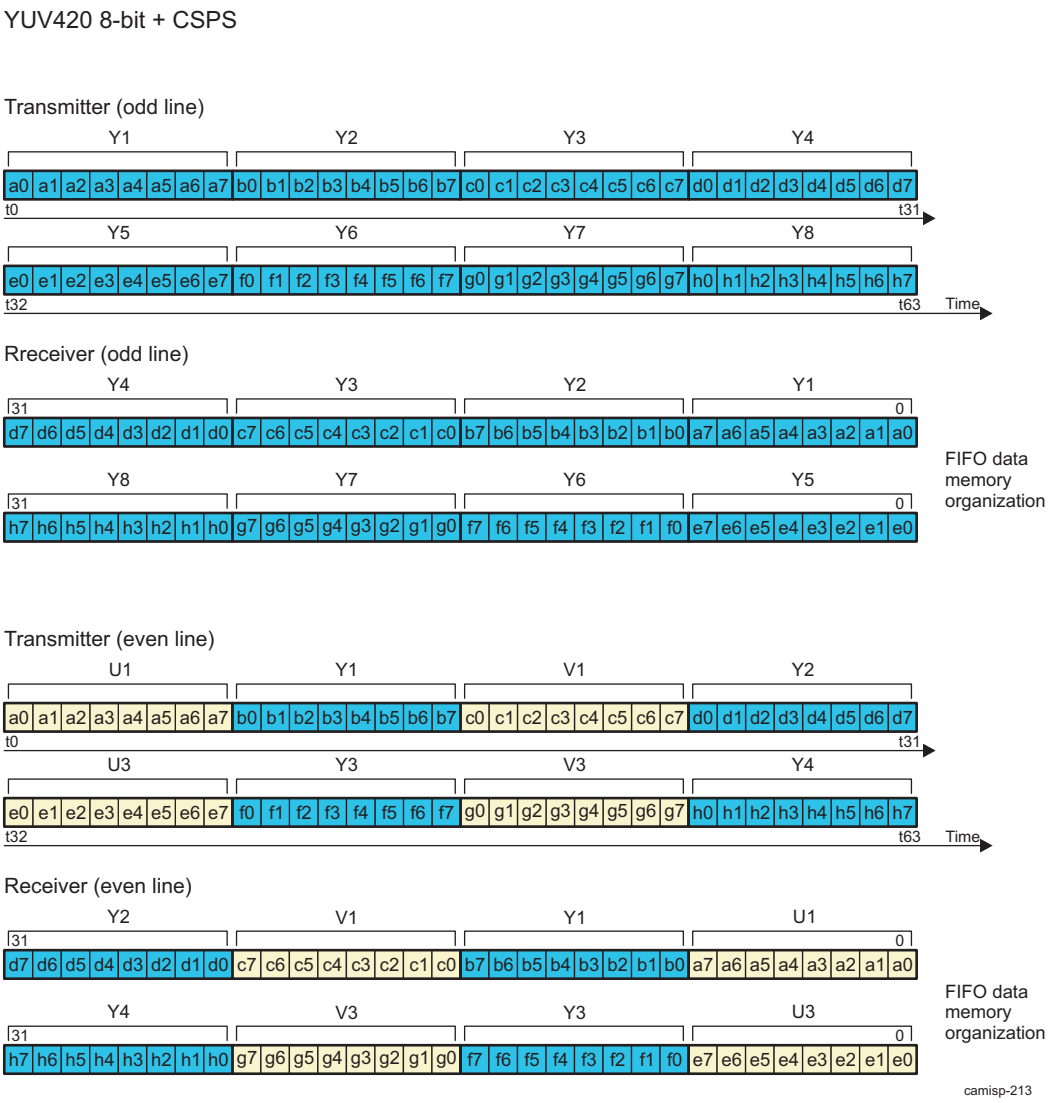


12.2.4.5.4.1.4 YUV420 8-Bit + CSPS

YUV420 8-bit CSPS data can be stored to memory in little-endian format. The line length sent through the CSI2 physical protocol is a multiple of 16 bits for odd lines and 32 bits for even lines.

For correct pixel reconstruction, the line length must be a multiple of 3*32 bits and the number of lines must be even. Figure 12-34 shows the storage format for YUV420 8-bit + CSPS data.

Figure 12-34. YUV420 8-Bit + CSPS



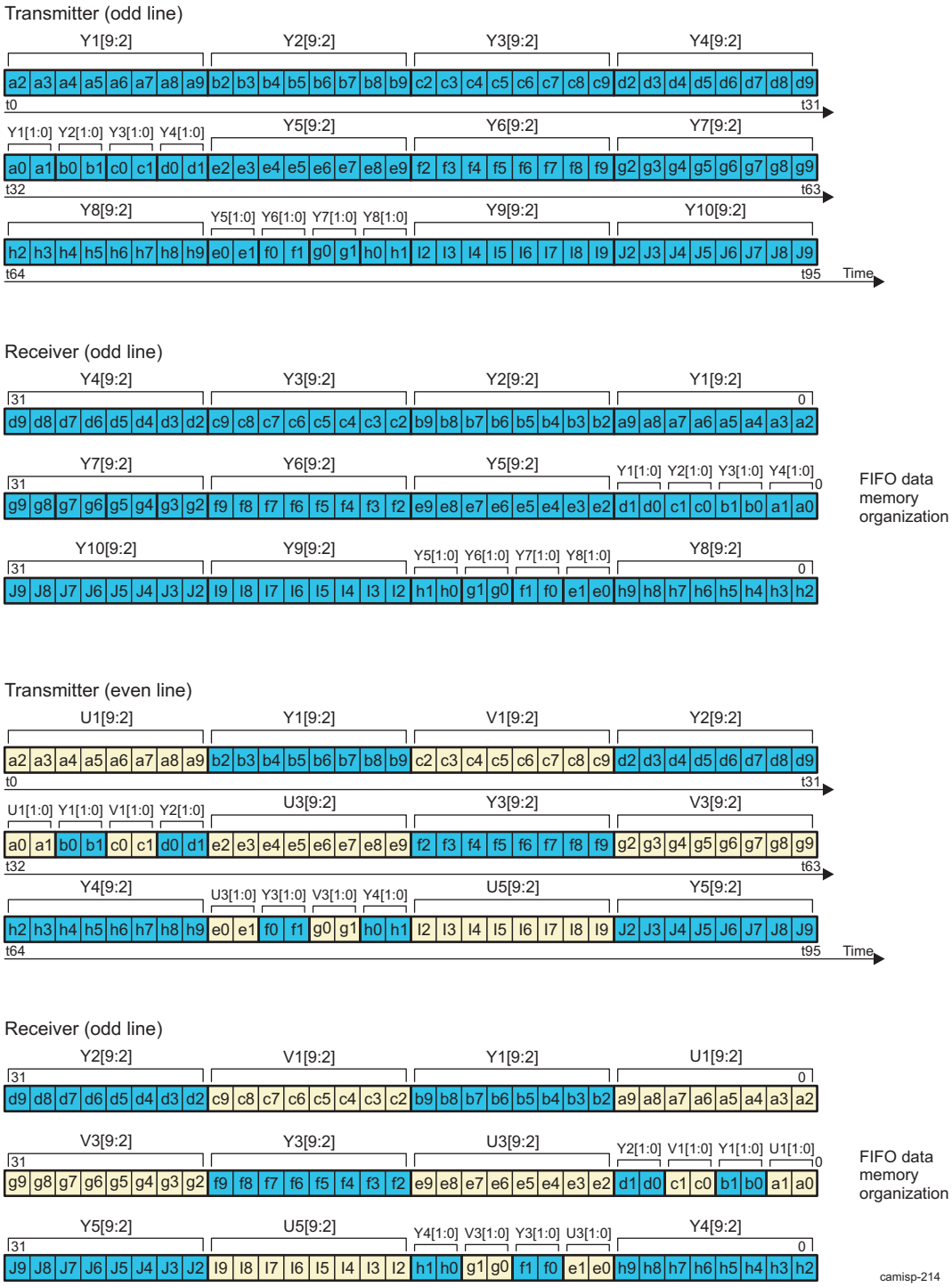
12.2.4.5.4.1.5 YUV420 10-Bit + CSPS

YUV420 10-bit CSPS data can be stored to memory in little-endian format. The line length sent through the CSI2 physical protocol is a multiple of 40 bits for odd lines and 80 bits for even lines.

For correct pixel reconstruction, the line length must be a multiple of 3*32 bits and the number of lines must be even. Figure 12-35 shows the storage format for YUV420 10-bit + CSPS data.

Figure 12-35. YUV420 10-Bit + CSPS

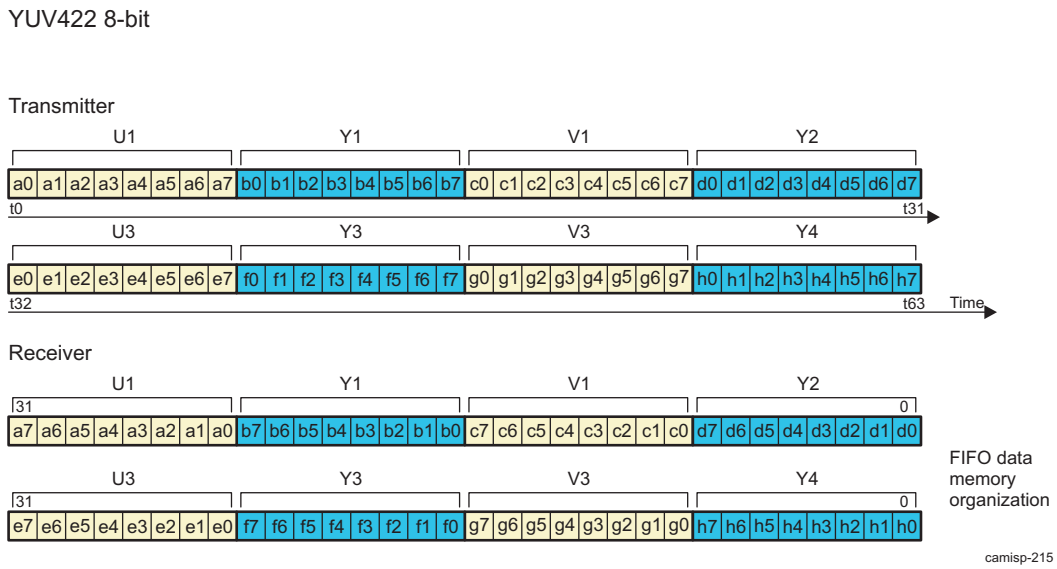
YUV420 10-bit + CSPS



12.2.4.5.4.1.6 YUV422 8-Bit

YUV422 data can be stored to memory in big-endian format. The line length sent through the CSI2 physical protocol is a multiple of 32 bits. Figure 12-36 shows the storage format for YUV422 8-bit data.

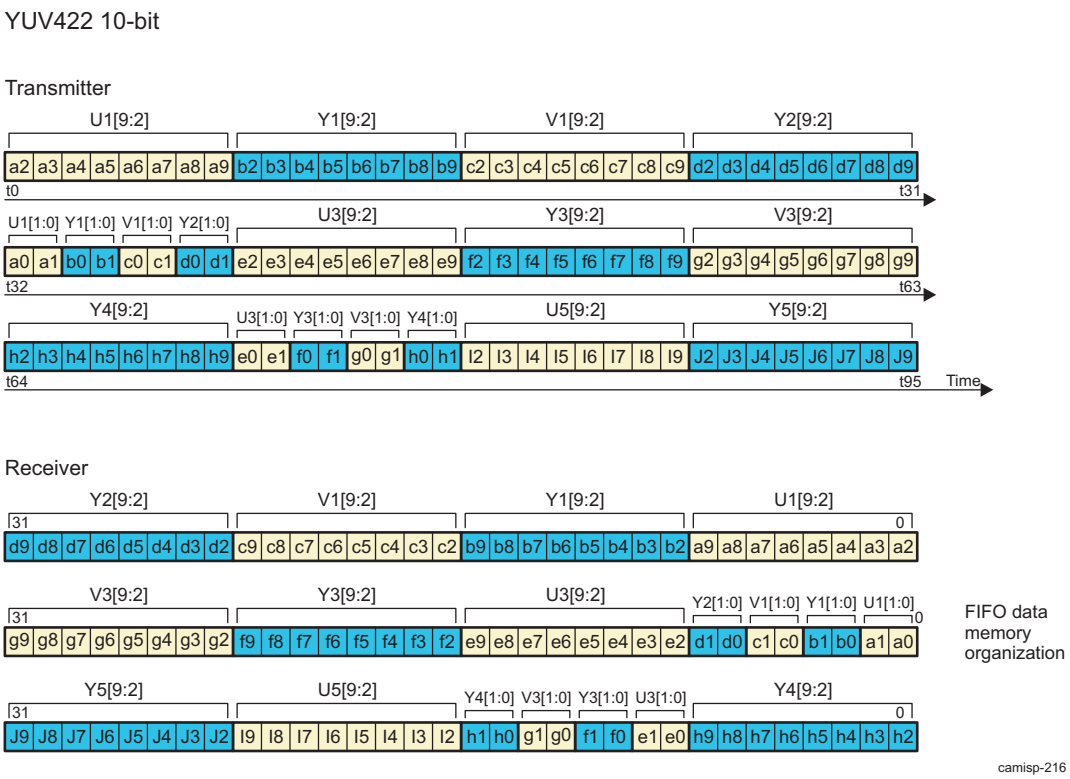
Figure 12-36. YUV422 8-Bit



12.2.4.5.4.1.7 YUV422 10-Bit

YUV422 data can be stored to memory in little-endian format. The line length sent through the CSI2 physical protocol is a multiple of 40 bits. Figure 12-37 shows the storage format for YUV422 10-bit data.

Figure 12-37. YUV422 10-Bit

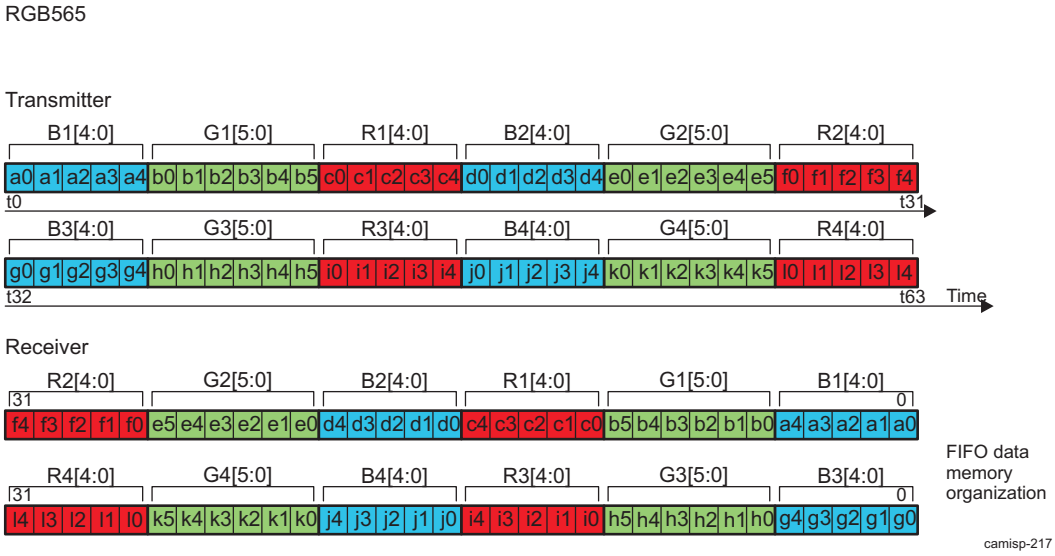


12.2.4.5.4.2 RGB Operating Modes

12.2.4.5.4.2.1 RGB565

RGB565 data is output to memory without data expansion. The line length sent through the CSI2 physical layer is always a multiple of 16 bits. Figure 12-38 shows the storage format for RGB565 data.

Figure 12-38. RGB565

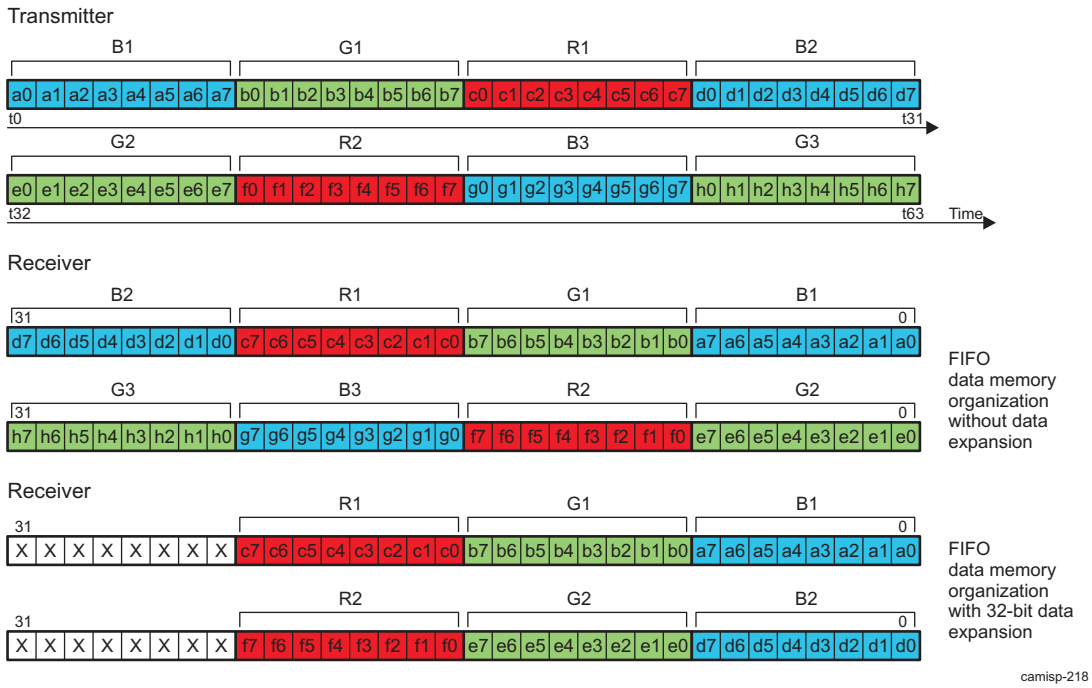


12.2.4.5.4.2.2 RGB888

RGB888 data can be output to memory in two formats: with or without data expansion. If data expansion is used, the value of the 8 upper bits is programmable and can be set with an alpha value for computer graphics applications (the CSI2_CTx_CTRL3 [29:16] ALPHA bit field). Figure 12-39 shows the storage format for RGB888 data.

Figure 12-39. RGB888

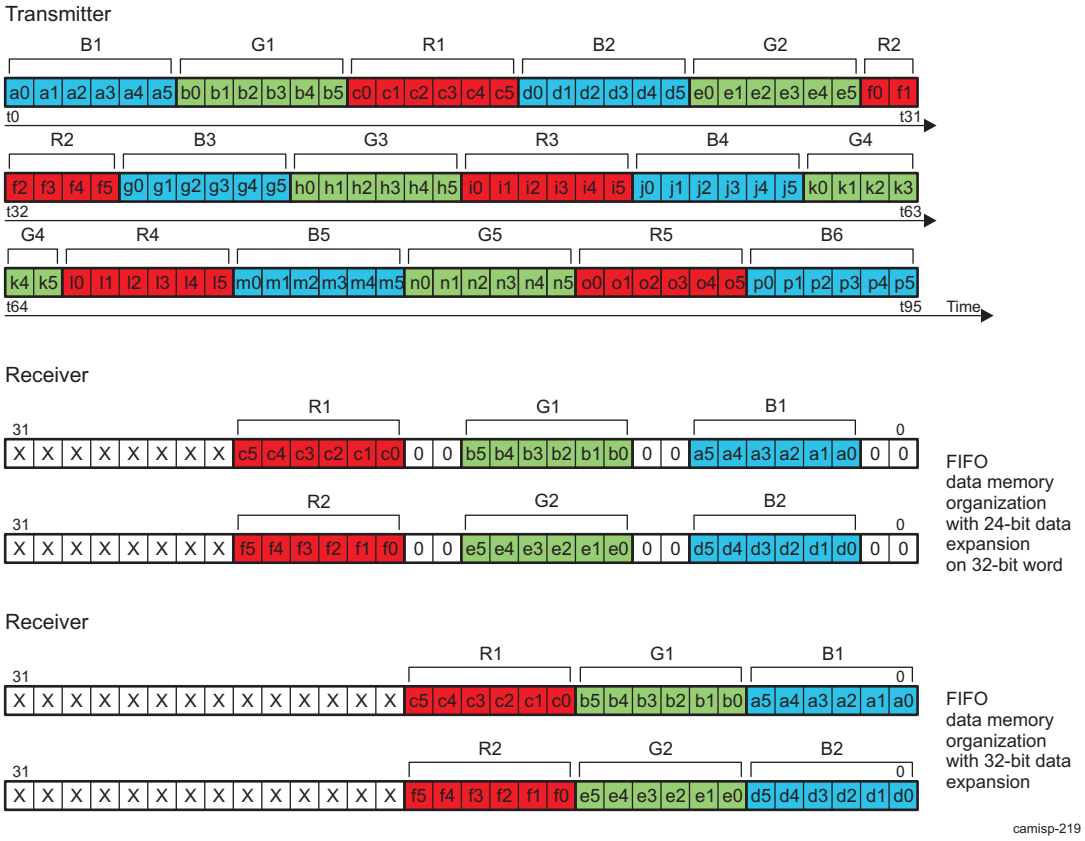
RGB888



12.2.4.5.4.2.3 RGB666

RGB666 data is always output to memory with data expansion. The value of the 14 upper bits is programmable and can be set with an alpha value for computer graphics applications (the CSI2_CTx_CTRL3 [29:16] ALPHA bit field). The line length sent through the CSI2 physical protocol is a multiple of 8 bits. Furthermore, the line length is a multiple of 9x8 bits to correctly finish the pixel reconstruction. Figure 12-40 shows the storage format for RGB666 data.

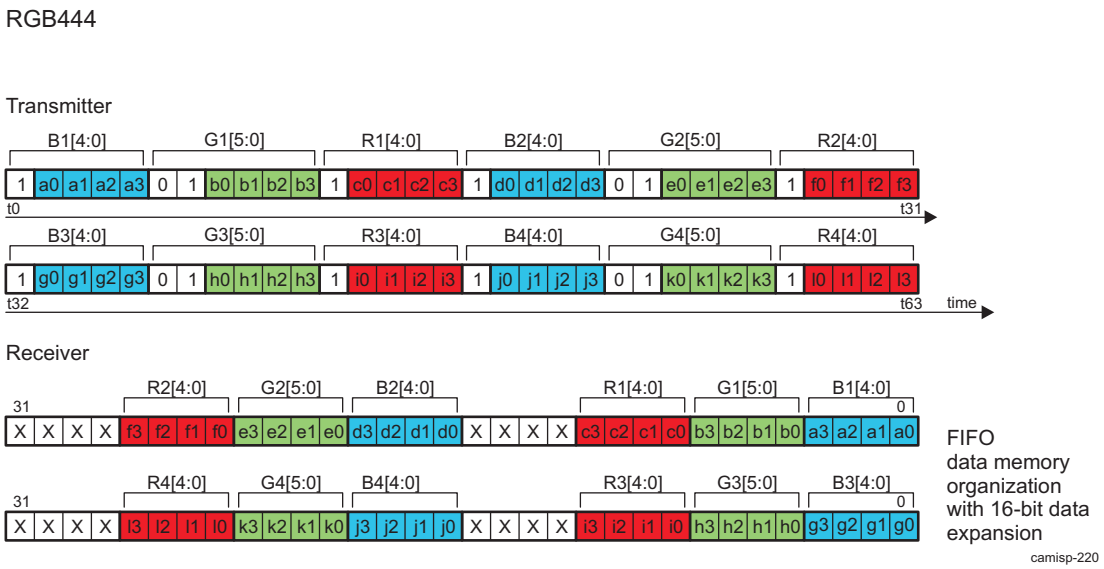
Figure 12-40. RGB666



12.2.4.5.4.2.4 RGB444

RGB444 data is output to memory with data expansion. When data expansion is used, the value of the 4 upper bits is programmable and can be set with an alpha value for computer graphics applications (the CSI2_CTx_CTRL3[29:16] ALPHA bit field). Figure 12-41 shows the storage format for RGB444 data.

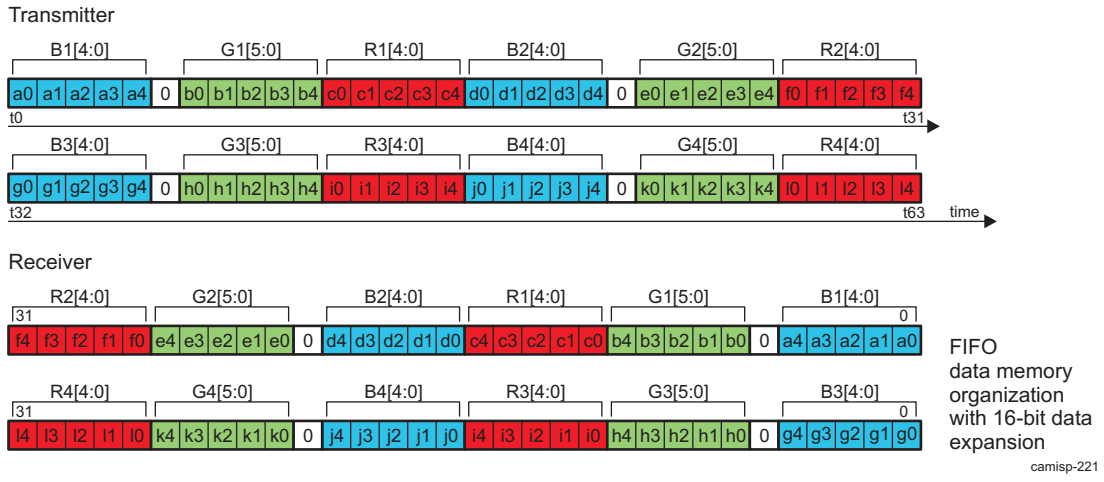
Figure 12-41. RGB444



12.2.4.5.4.2.5 RGB555

RGB555 data is output to memory with data expansion. Figure 12-42 shows the storage format for RGB555 data.

Figure 12-42. RGB555

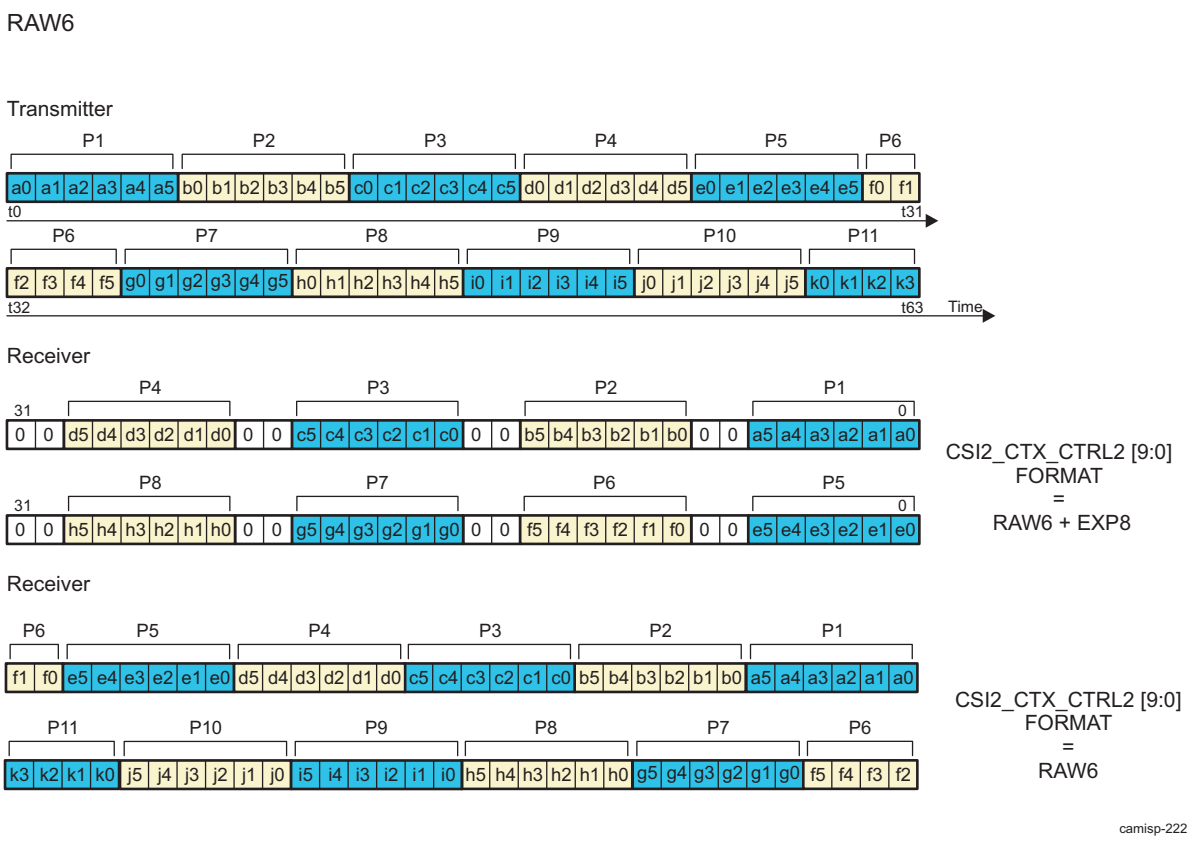


12.2.4.5.4.3 RAW Bayer RGB Operating Modes

12.2.4.5.4.3.1 RAW6

RAW6 data can be output only with data expansion. The line length sent through the CSI2 physical layer is a multiple of 8 bits. Furthermore, the line length is a multiple of 3x8 bits to correctly complete the pixel reconstruction (the lowest common multiple of 8 and 6 is 24, so 3x8 bits). Figure 12-43 shows the storage format for RAW6 data.

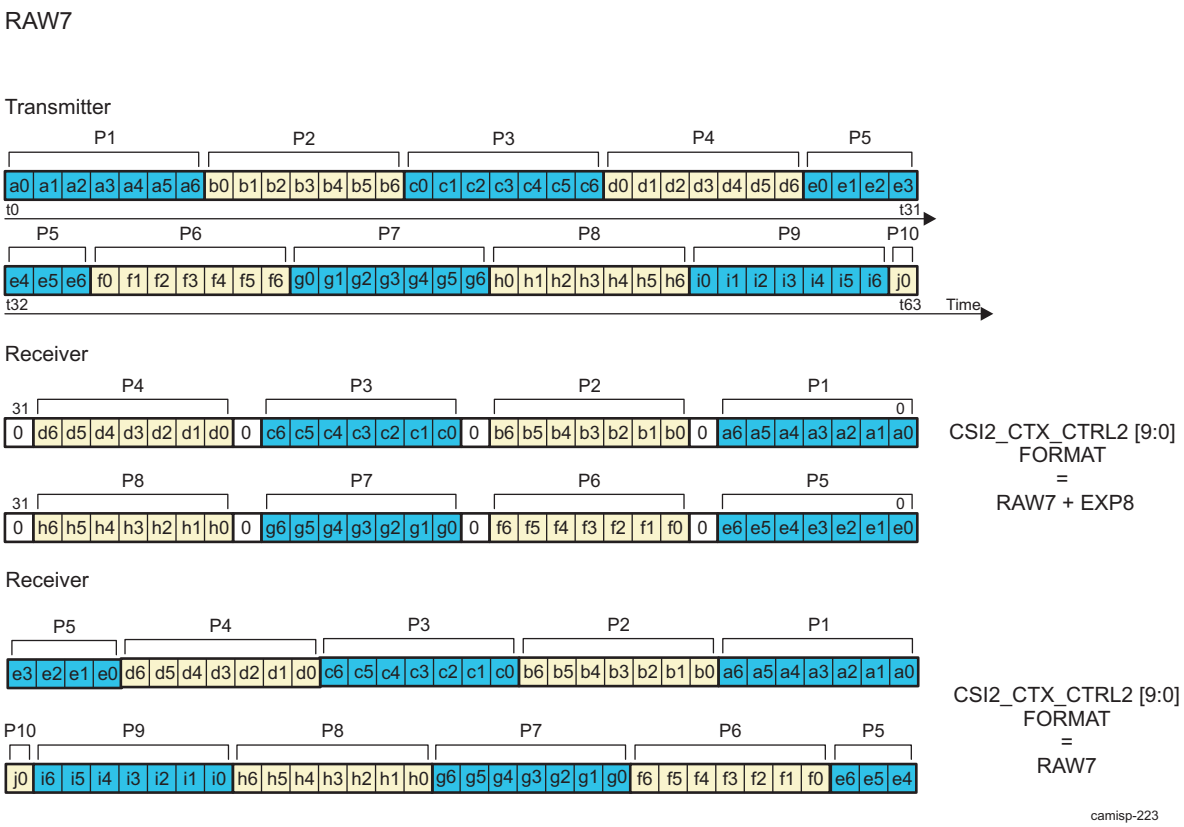
Figure 12-43. RAW6



12.2.4.5.4.3.2 RAW7

RAW7 data can be output only with data expansion. The line length sent through the CSI2 physical layer is a multiple of 8 bits. Furthermore, the line length is a multiple of 7x8 bits to correctly complete the pixel reconstruction (the lowest common multiple of 8 and 7 is 56, so 7x8 bits). Figure 12-44 shows the storage format for RAW7 data.

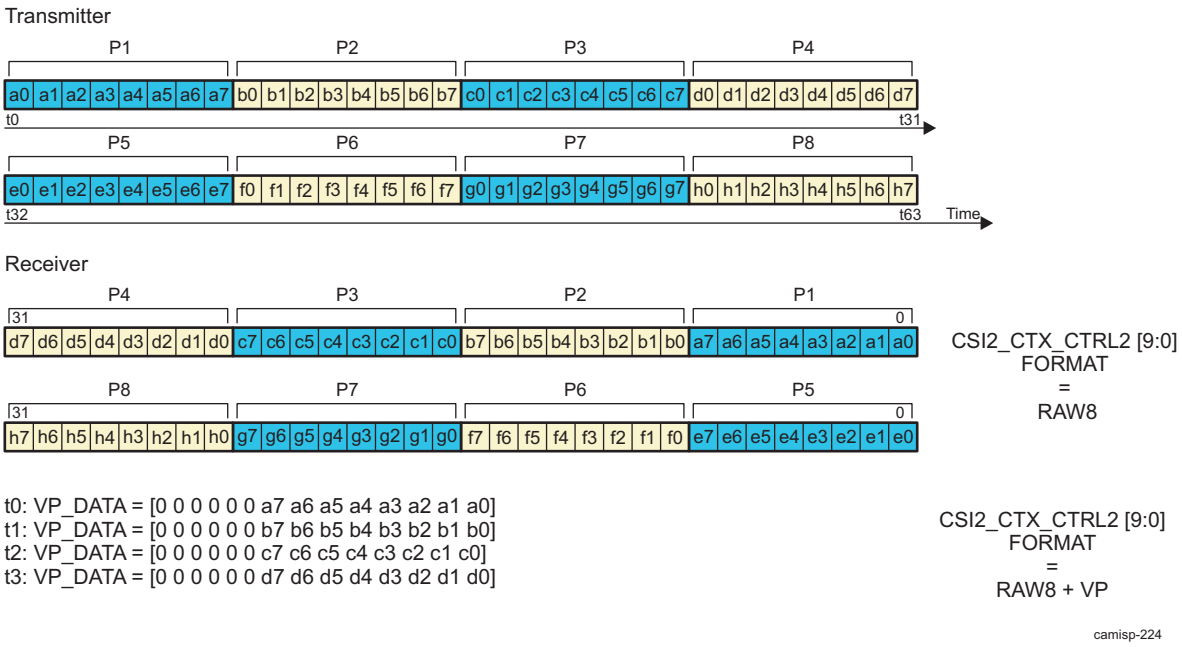
Figure 12-44. RAW7



12.2.4.5.4.3.3 RAW8

RAW8 data can be output only without data expansion. The line length sent through the CSI2 physical layer is always a multiple of 8 bits. Figure 12-45 shows the storage format for RAW8 data.

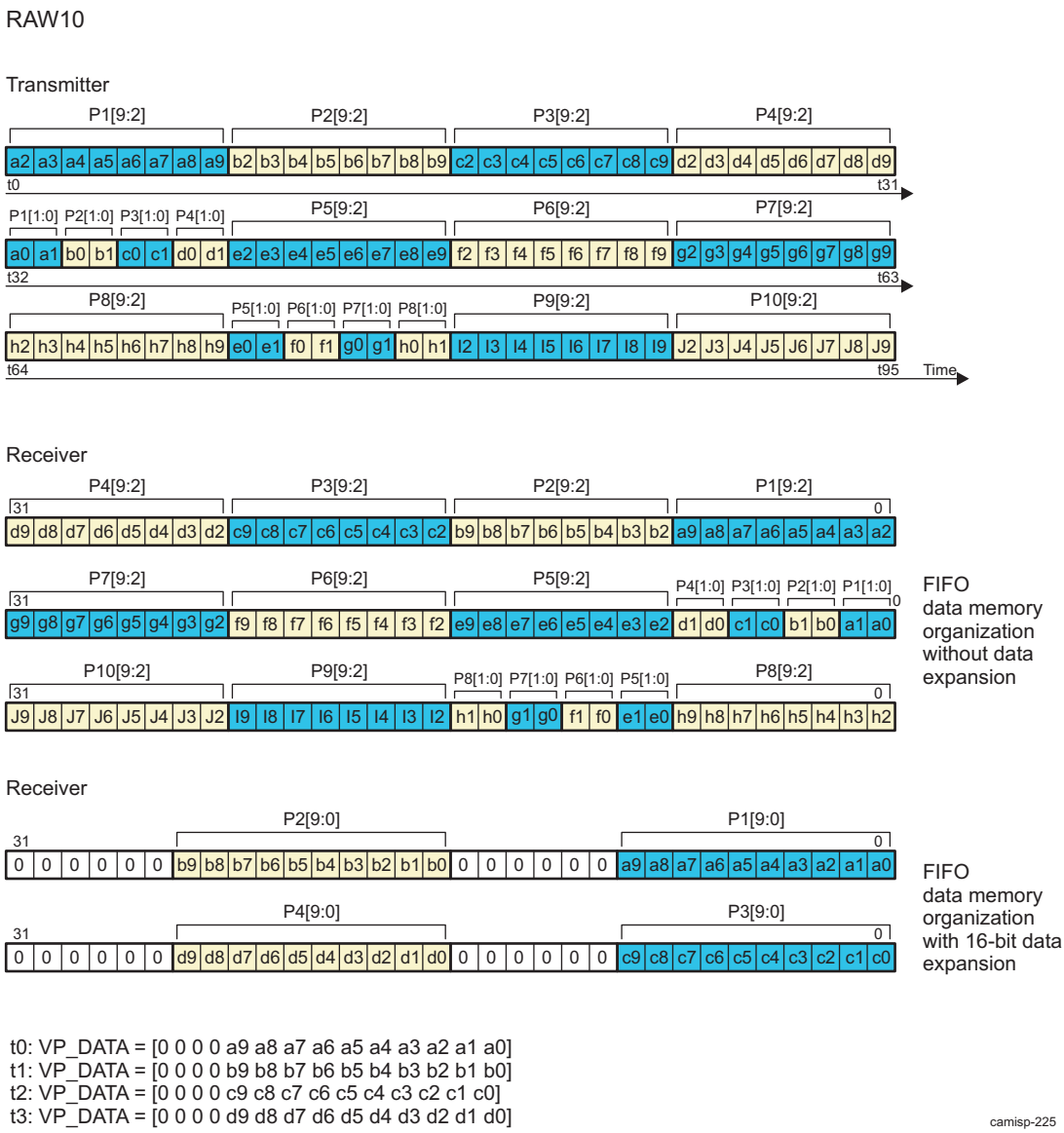
Figure 12-45. RAW8



12.2.4.5.4.3.4 RAW10

RAW10 data can be output memory in two formats: with or without data expansion. It can be sent to video port too. If data expansion is used, the 10-bit data are padded with 0s on a 16-bit word. The line length sent through the CSI2 physical layer is a multiple of 8 bits. Furthermore, the line length is a multiple of 5x8 bits to correctly complete the pixel reconstruction (the lowest common multiple of 8 and 10 is 40, so 5x8 bits). Figure 12-46 shows the storage format for RAW10 data.

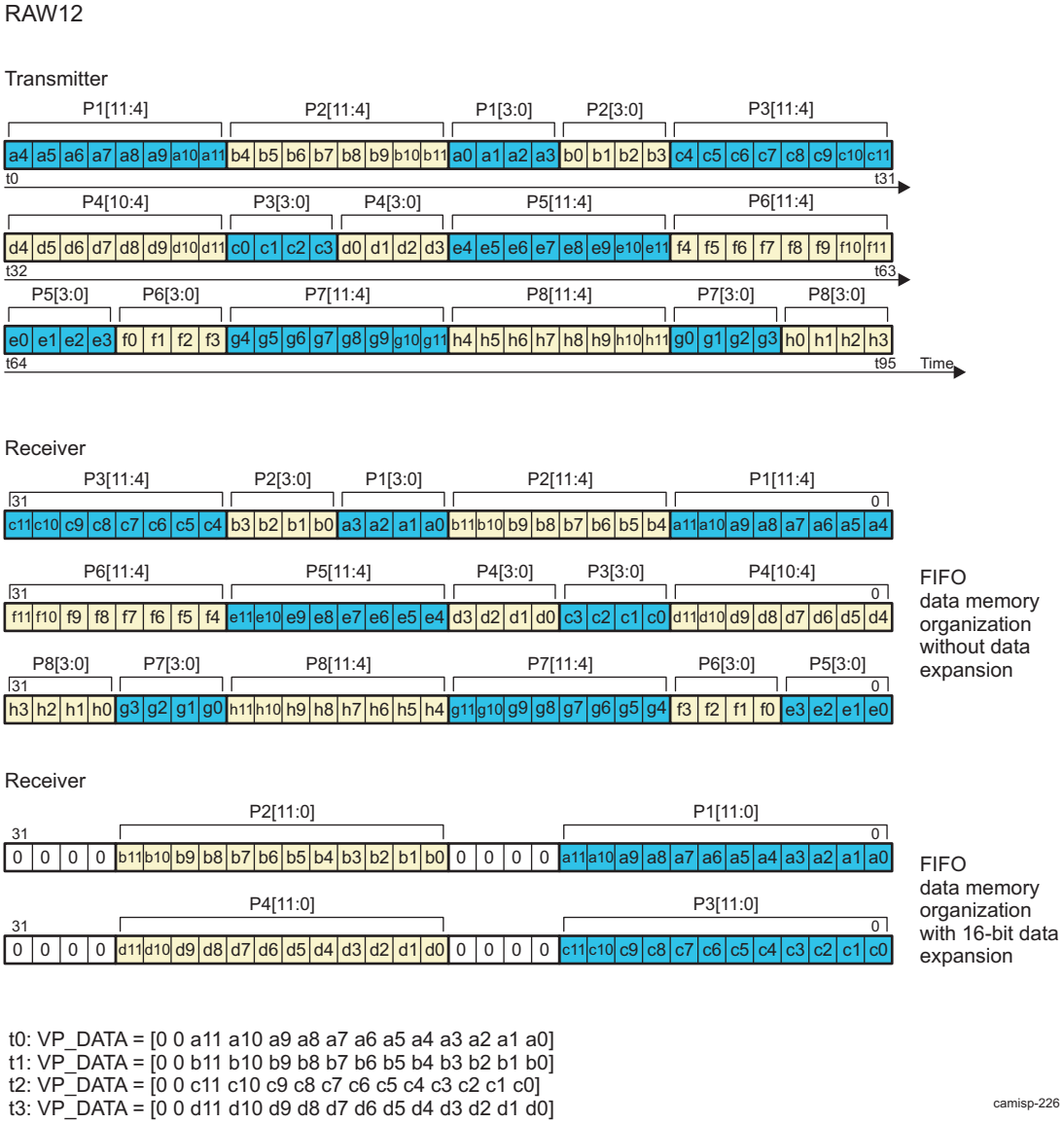
Figure 12-46. RAW10



12.2.4.5.4.3.5 RAW12

RAW12 data can be output to memory in two formats: with or without data expansion. It can be sent to video port too. If data expansion is used, the 12-bit data are padded with 0s on a 16-bit word. The line length sent through the CSI2 physical layer is a multiple of 8 bits. Furthermore, the line length is a multiple of 3x8 bits to correctly complete the pixel reconstruction (the lowest common multiple of 8 and 12 is 24, so 3x8 bits). Figure 12-47 shows the storage format for RAW12 data.

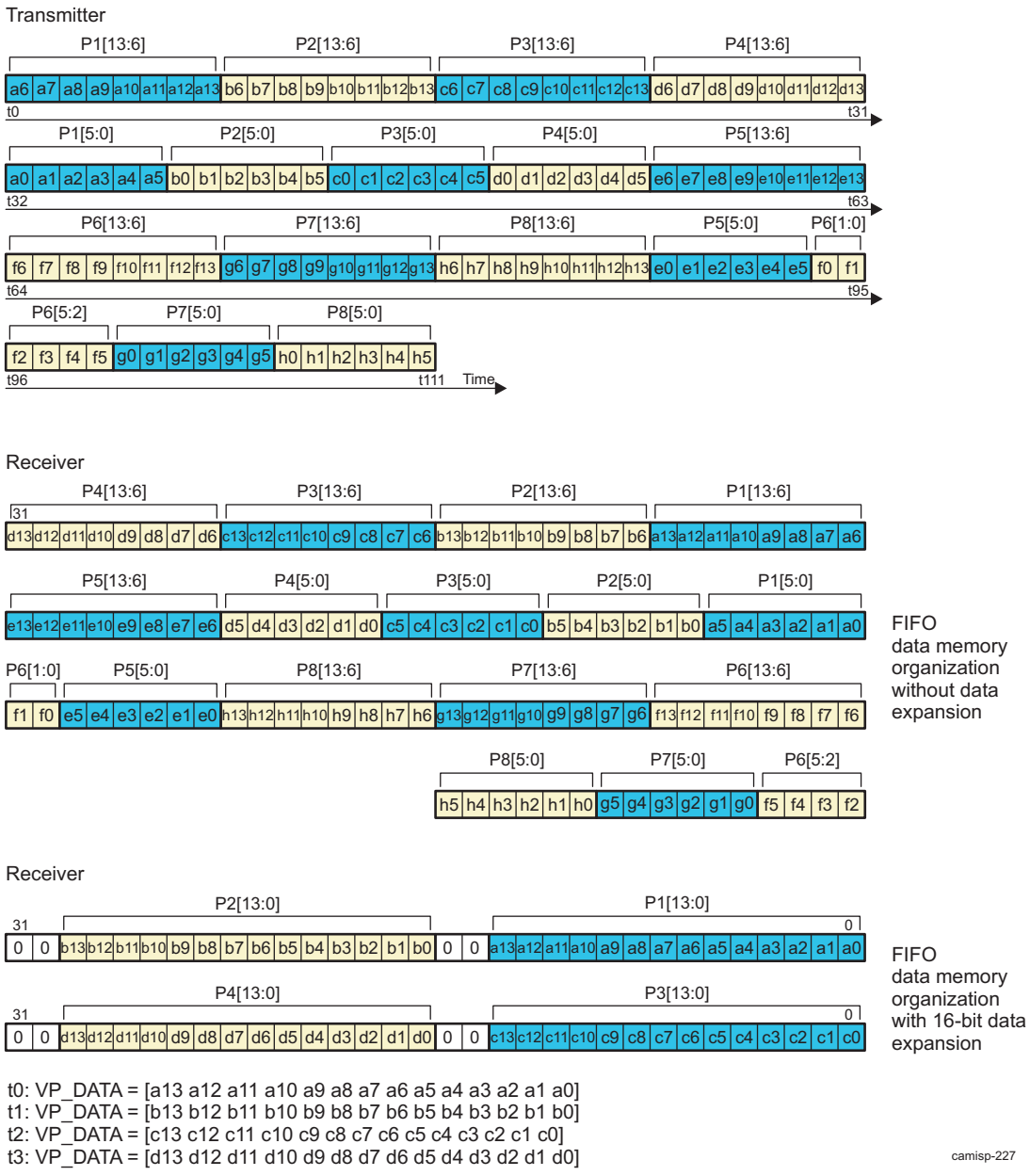
Figure 12-47. RAW12



12.2.4.5.4.3.6 RAW14

RAW14 data can be output to memory in two formats: with or without data expansion. It can be sent to video port too. If data expansion is used, the 14-bit data are padded with 0s on a 16-bit word. The line length sent through the CSI2 physical layer is a multiple of 8 bits. Furthermore, the line length is a multiple of 7x8 bits to correctly complete the pixel reconstruction (the lowest common multiple of 8 and 14 is 56, so 7x8 bits). Figure 12-48 shows the storage format for RAW14 data.

Figure 12-48. RAW14



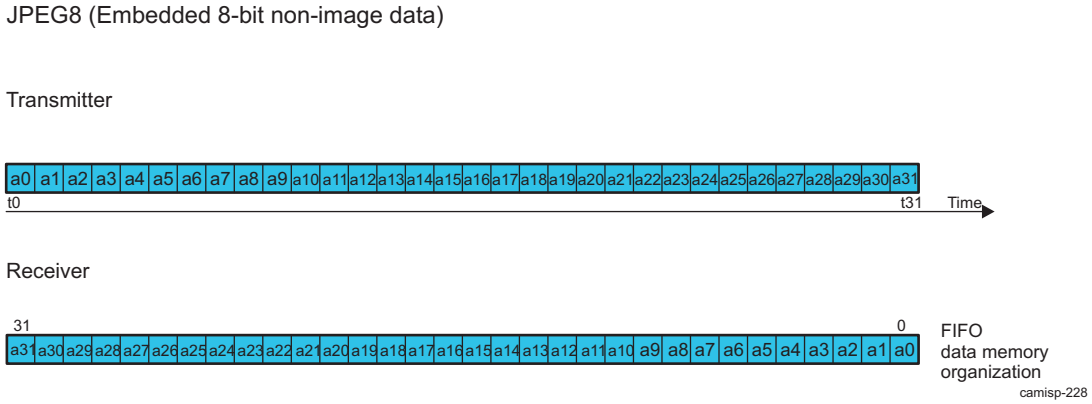
t0: VP_DATA = [a13 a12 a11 a10 a9 a8 a7 a6 a5 a4 a3 a2 a1 a0]
t1: VP_DATA = [b13 b12 b11 b10 b9 b8 b7 b6 b5 b4 b3 b2 b1 b0]
t2: VP_DATA = [c13 c12 c11 c10 c9 c8 c7 c6 c5 c4 c3 c2 c1 c0]
t3: VP_DATA = [d13 d12 d11 d10 d9 d8 d7 d6 d5 d4 d3 d2 d1 d0]

camisp-227

12.2.4.5.4.4 JPEG8 Operating Modes

The size of a compressed stream can be known in advance. Figure 12-49 shows the format for storing JPEG8 data.

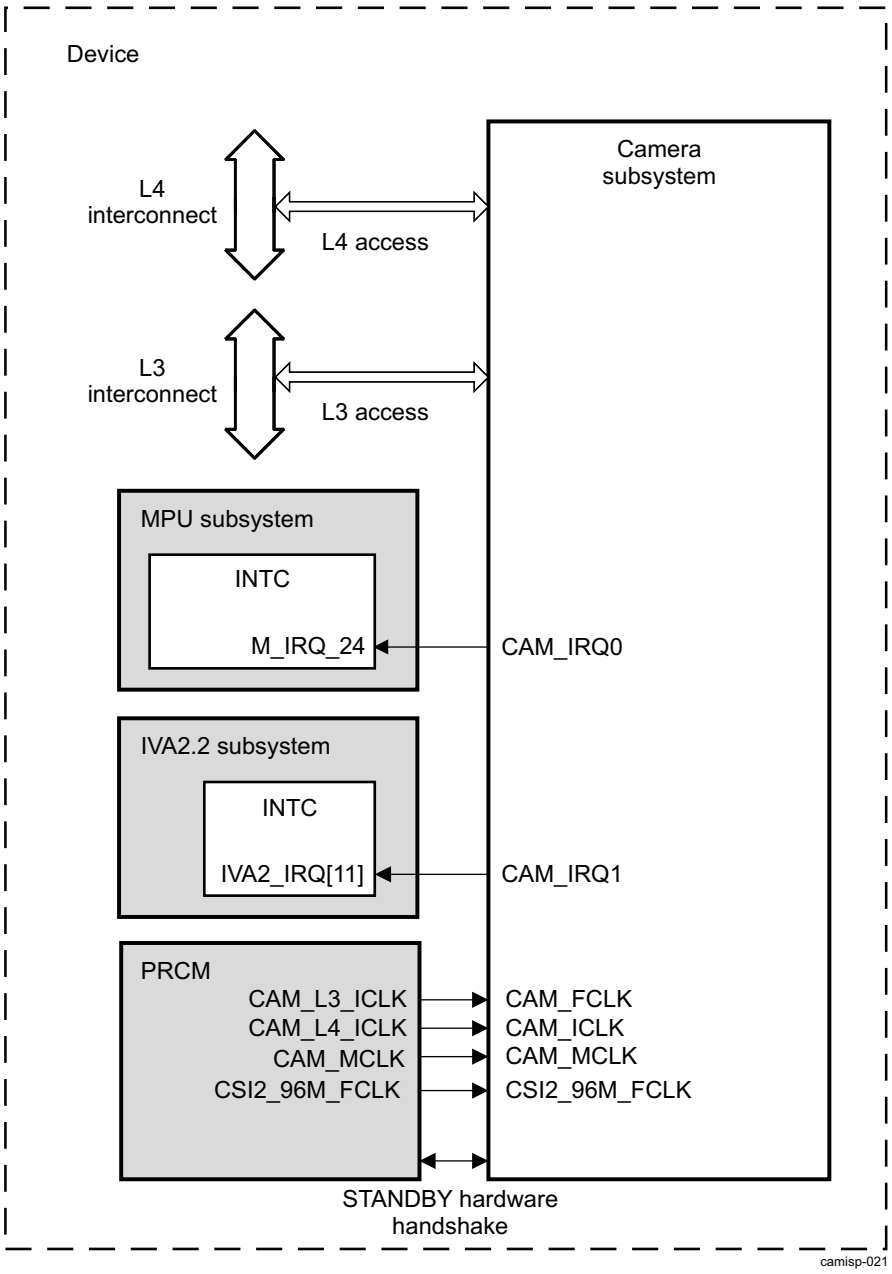
Figure 12-49. JPEG8



12.3 Camera ISP Integration

Figure 12-50 shows the Camera ISP integration.

Figure 12-50. Camera ISP Integration



12.3.1 Clocking, Reset, and Power-Management Scheme

12.3.1.1 Clocks

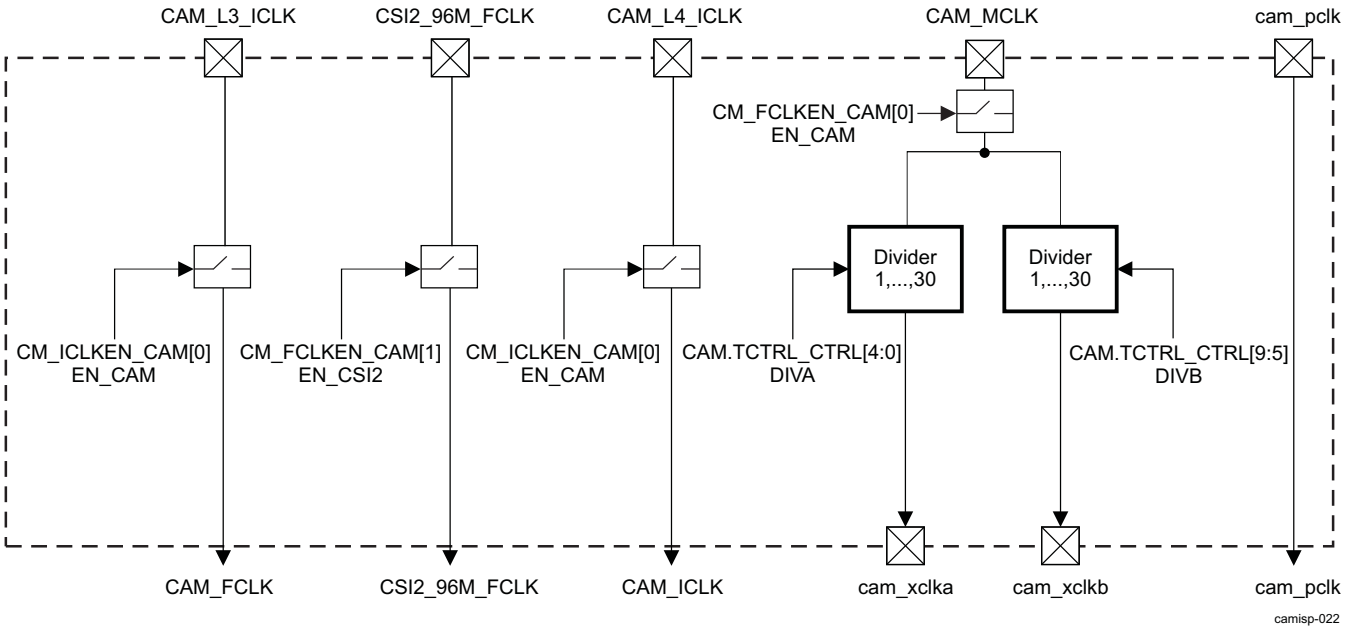
There are four clock domains in the Camera ISP:

- Functional clock domain
- Interface clock domain
- Serial sensor clock domain
- Parallel sensor clock domain

12.3.1.1.1 Clock Tree

Figure 12-51 shows the clock tree for the Camera ISP module.

Figure 12-51. Clock Tree



12.3.1.1.2 Clock Descriptions

Table 12-14 describes the Camera ISP clocks.

Table 12-14. Clock Descriptions

Signal Name	I/O	Description
CAM_FCLK	Input	Functional clock (L3 interconnect clock domain) Functional clock domain.
CAM_ICLK	Input	Interface clock (L4 interconnect clock domain) Interface clock domain.
CAM_MCLK(1)	Input	Internal clock from PRCM at 216 MHz. The CAM_MCLK is only used by the clock generator to generate cam_xclka and cam_xclkb.
cam_xclka	Output	External clock for the image-sensor module. For serial or parallel sensor.
cam_xclkb	Output	External clock for the image-sensor module. For serial or parallel sensor.
cam_pclk	Input	Parallel mode: pixel clock for parallel input data. The data on the parallel interface are presented on cam_d, one pixel for every cam_pclk rising or falling edge. Parallel sensor clock domain.
CSI2_96M_FCLK	Input	CSI2 functional clock driven by CM_FCLKEN_CAM[1] EN_CSI2.

(1) The CAM_MCLK frequency can be less than, or equal to 216 MHz.

12.3.1.1.3 Clock Configuration

- CSI2_96M_FCLK is the CSI2 functional clock running at 96 MHz and enabled when CM_FCLKEN_CAM[1] EN_CSI2 = 1
- CAM_FCLK control and settings

  The CAM_FCLK clock runs at the device L3 clock frequency.

The CAM_FCLK source is the PRCM CAM_L3_ICLK output.

When the Camera ISP no longer needs CAM_FCLK, the software can disable it at PRCM level by setting the PRCM.CM_ICLK_CAM [0] EN_CAM bit to 0x0. Note that the clock is not effectively shut down until the Camera ISP module has reached the IDLE state (that is, it does not generate anymore traffic on the device interconnect and is internally idled). For information, note that an automatic HW handshake protocol takes place between the Camera ISP and the PRCM to prevent CAM_FCLK from being cut while the module is processing or transferring data.

An autoidle mode can be activated for this clock (PRCM.CM_AUTOIDLE_CAM [0] AUTO_CAM bit set to 1). This allows global management of the CAM_FCLK clock along with the CAM power domain at PRCM level. For more information, see Chapter 4, *Power, Reset, and Clock Management*.

- CAM_ICLK control and settings:

  CAM_ICLK is the interface clock. It runs at device L4 interconnect clock speed and triggers access to the Camera ISP L4 interface. Its source is the PRCM CAM_L4_ICLK output.

  When the Camera ISP no longer needs CAM_ICLK, the software can disable it at PRCM level by setting to 0x0 the PRCM.CM_ICLK_CAM [0] EN_CAM bit. Note that the clock is not effectively shut down until the Camera ISP module has reached the IDLE state (that is, It does not generate anymore traffic on the device interconnect and is internally idled). As information, note an automatic HW handshake protocol takes place between the Camera ISP and the PRCM to prevent CAM_ICLK from being cut while the module is processing or transferring data.

  An autoidle mode can be activated for this clock (PRCM.CM_AUTOIDLE_CAM [0] AUTO_CAM bit set to 1). This allows global management of the CAM_ICLK clock along with the CAM power domain at PRCM level. For more information, see Chapter 4, *Power, Reset, and Clock Management*.

- CAM_MCLK control and settings:

  The source for CAM_MCLK is the PRCM CAM_MCLK output. It is generated trough a peripheral DPLL and its frequency can be adjusted using the PRCM.CM_CLKSEL_CAM CLKSEL_CAM bit field. When the module no longer needs it, it can be disabled at PRCM level by setting the PRCM.CM_FCLKEN_CAM EN_CAM bit to 0x0. Note that the PRCM register bit setting has a direct effect on the clock; in other terms, there is no HW handshake protocol to ensure whether the clock can be cut regarding the Camera ISP activity: as soon as the bit is set to 0x0, CAM_MCLK is shut down.

  – cam_xclka and cam_xclkb control and settings:

    cam_xclka and cam_xclkb are generated inside the Camera ISP module from CAM_MCLK clock input. Table 12-15 and Table 12-16 give the settings of the related bit fields.

Table 12-15. cam_xclka Configuration

ref_clk	CAM.TCTRL_CTRL[4:0] DIVA field	cam_xclka
CAM_MCLK	0x0	Stable low level. Divider disabled.
	0x1	Stable high level. Divider disabled.
	0x2	CAM_MCLK/2
	...	...
	0x1F	CAM_MCLK

Table 12-16. cam_xclkb Configuration

ref_clk	CAM.TCTRL_CTRL[9:5] DIVB field	cam_xclkb
CAM_MCLK	0x0	Stable low level. Divider disabled.
	0x1	Stable high level. Divider disabled.
	0x2	CAM_MCLK/2
	...	...
	0x1F	CAM_MCLK

12.3.1.2 Power Management

Power consumption can be reduced at two different levels:

- A local power-management optimization
- A system power management

12.3.1.2.1 Local Power Management

To optimize power consumption, a local autoidle feature is implemented on the CAM_ICLK interface clock at Camera ISP module level. By enabling this autoidle feature, CAM_ICLK is automatically gated at the module boundary as soon as it is not required and restarted without any latency when needed again. This allows power saving when the module is not involved in any transfer from/to the device interconnect. This autoidle feature is enabled by setting:

- ISP_SYSCONFIG [0] AUTO_IDLE bit to 1
- CSI1_SYSCONFIG [0] AUTO_IDLE bit to 1
- CSI2_SYSCONFIG [0] AUTO_IDLE bit to 1
- MMU_SYSCONFIG [0] AUTOIDLE bit to 1
- ISP_CTRL [21] SBL_AUTOIDLE bit to 1

The decision to gate the interconnect clock is based on interface activity, regardless of hardware handshake protocols.

After a reset, this mode is enabled, by default.

Note: It is recommended that this mode be enabled to reduce power consumption.

12.3.1.2.2 System Power Management

As part of the system power management scheme, the Camera ISP module interacts with the PRCM through an automatic standby hardware protocol that allows dynamic power savings at CAMERA power domain level.

Being an initiator on the L3 interconnect, the Camera ISP module alerts the PRCM as soon as it is ready to switch to a lower power state. Prior to any power state change to the CAMERA power domain, the PRCM first shuts off the camera clocks, depending on the Camera ISP behavior and the clock settings at PRCM level. Namely, as soon as the Camera ISP is ready to go to STANDBY, it asserts an automatic HW STANDBY request to the PRCM, which shuts down the clocks if they have been previously disabled by software and then potentially change the CAMERA power domain state. Refer to Chapter 4, *Power, Reset, and Clock Management* for further details.

When it goes to standby, the Camera ISP module alerts the PRCM differently, depending on the ISP_SYSCONFIG [13:12] MIDLE_MODE bit field settings. The entry conditions for the Camera ISP to go into standby mode are:

- The module has finished any current transaction and does not generate any more traffic on the interconnect.
- The module is idle (on-going transactions are finished).

The MIDLE_MODE bit field allows the following settings for the Camera ISP module:

- Force standby

  The Camera ISP is set to force standby when MIDLE_MODE is set to 0x0. In this mode, the Camera ISP module asserts its HW standby request to the PRCM as soon as it is disabled. Namely, the Camera ISP is disabled when the following bit fields are set to 0x0:

  – ISP_CTRL [13] RSZ_CLK_EN
  – ISP_CTRL [12] PRV_CLK_EN
  – ISP_CTRL [11] HIST_CLK_EN
  – ISP_CTRL [10] H3A_CLK_EN

  - ISP_CTRL [8] CCDC_CLK_EN

    Moreover, CSIa must also be disabled by setting the related CSI2_CTRL[0] IF_EN bits to 0x0 and CSIb must also be disabled by setting the related CSI1_CTRL [0] IF_EN bits to 0x0.

- No standby

  The Camera ISP is set to no standby when MIDLE_MODE is set to 0x1. In this mode, the Camera ISP module never asserts its standby request to the PRCM. Note that this also prevents the CAMERA power domain from going to a lower power state. Refer to Chapter 4, *Power, Reset, and Clock Management*, for further details.

- Smart standby

  The Camera ISP module is set to smart standby when MIDLE_MODE is set to 0x2. In this mode, the Camera ISP module asserts its HW standby request to the PRCM according with its internal activity (namely, when there is no more activity on the Camera ISP master interface, that is, when there is no more data in the central resource buffer).

Note: CSI1 must also be configured to smart standby mode when the Camera ISP is set to smart standby (CSI1_SYSCONFIG [13:12] MSTANDBY_MODE set to 0x2).

CSI2 must also be configured to smart standby mode when the Camera ISP is set to smart standby (CSI2_SYSCONFIG[13:12] MSTANDBY_MODE set to 0x2).

A standby request asserted to the PRCM does not necessarily lead to CAM_FCLK and CAM_ICLK being cut. The PRCM must also be set correctly for that purpose.

The clocks are cut, allowing a CAMERA power state change if:

- PRCM.CM_ICLKEN_CAM[0] EN_CAM = 0 and PRCM.CM_FCLKEN_CAM[0] EN_CAM = 0
- PRCM.CM_FCLKEN_CAM[0] EN_CAM = 0, (PRCM.CM_ICLKEN_CAM[0] EN_CAM =1, PRCM.CM_AUTOIDLE_CAM[0] = 1, and a domain transition is required at PRCM level)

  See Chapter 4, *Power, Reset, and Clock Management*, for further details.

12.3.1.3 Power Domain

The Camera ISP belongs to the CAMERA power domain. For more information about the CAMERA power domain, see Chapter 4, *Power, Reset, and Clock Management*.

12.3.1.4 Resets

12.3.1.4.1 Hardware Reset

Global reset of the Camera ISP module is accomplished by activation of the CAM_RST signal in the CAMERA domain (for more information, see Chapter 4, *Power, Reset, and Clock Management*).

12.3.1.4.2 Software Reset

A global software reset can be accomplished by setting the Camera ISP ISP_SYSCONFIG [1] SOFT_RESET bit to 1. Setting this bit enables active software reset functionality equivalent to hardware reset for the entire module, including CSI1 and CSI2 receivers.

Note: The CSI1 receiver accepts a general software reset, propagated through the hierarchy. This reset can be performed to initialize the module and has the same effect as a hardware reset. The CSI1 receiver can be reset by writing CSI1_SYSCONFIG [1] SOFT_RESET bit to 1. The software can monitor the CSI1_SYSSTATUS [0] RESET_DONE status bit to wait for the completion of the reset procedure.
If after 5 reads, CSI1_SYSSTATUS [0] RESET_DONE still returns 0, it can be assumed that an error occurred during the reset stage.

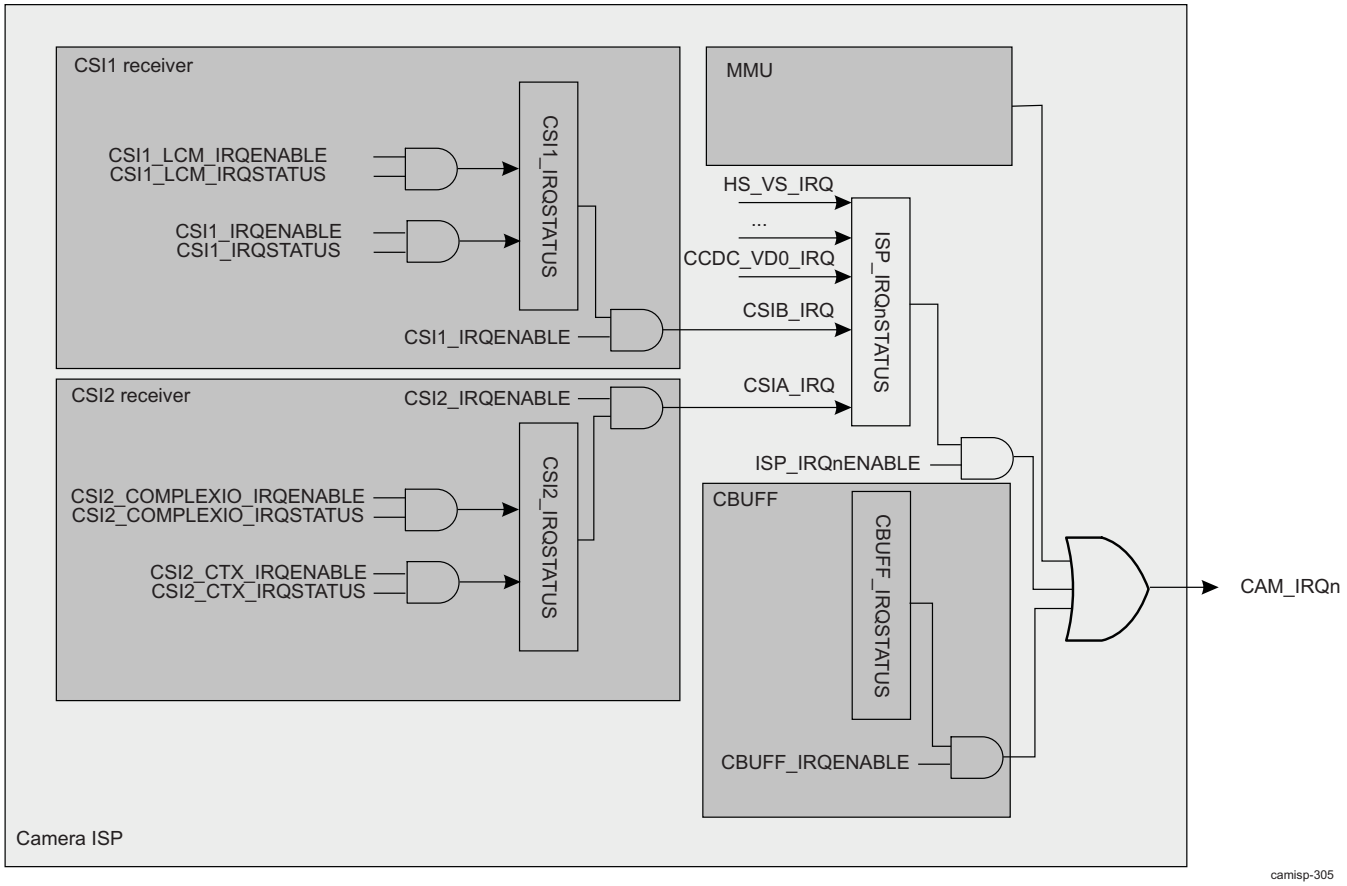
The CSI2 receiver accepts a general software reset, propagated through the hierarchy. This reset can be performed to initialize the module and has the same effect as a hardware reset. The CSI2 receiver can be reset by writing CSI2_SYSCONFIG [1] SOFT_RESET bit to 1. The software can monitor the CSI2_SYSSTATUS [0] RESET_DONE status bit to wait for the completion of the reset procedure.
If after 5 reads, CSI2_SYSSTATUS [0] RESET_DONE still returns 0, it can be assumed that an error occurred during the reset stage.

12.3.2 Hardware Requests

12.3.2.1 Interrupt Requests

Figure 12-52 shows the interrupt generation tree of the camera subsystem.

Figure 12-52. Interrupt Generation Tree



The Camera ISP can generate two interrupts:

- CAM_IRQ0 is an interrupt to the MPU subsystem interrupt controller. It is mapped on M_IRQ_24.
- CAM_IRQ1 is an interrupt to the IVA2.2 subsystem interrupt controller. It is mapped on IVA2_IRQ[11].

Table 12-17 summarizes events that cause interrupts.

Table 12-17. Camera ISP Interrupts

Event	Mask	Description
ISP_IRQ0STATUS [31] HS_VS_IRQ	ISP_IRQ0ENABLE [31] HS_VS_IRQ	HS or VS synchronization event: triggered if a rising or falling edge is detected on the HS or VS signal. The rising or falling edge and the HS or VS signal selection are chosen with the ISP_CTRL [15:14] SYNC_DETECT bit field.
ISP_IRQ0STATUS [28] MMU_ERR_IRQ	ISP_IRQ0ENABLE [28] MMU_ERR_IRQ	MMU error
ISP_IRQ0STATUS [25] OVF_IRQ	ISP_IRQ0ENABLE [25] OVF_IRQ	Central-resource SBL overflow: triggered when one of the buffers in the central-resource SBL overflows
ISP_IRQ0STATUS [24] RSZ_DONE_IRQ	ISP_IRQ0ENABLE [24] RSZ_DONE_IRQ	RESIZER module. Resizer processing-done event: Triggered at the end of the frame when processing is complete for the current frame. It applies to one-shot and continuous modes.
ISP_IRQ0STATUS [21] CBUFF_IRQ	ISP_IRQ0ENABLE [24] CBUFF_IRQ	CBUFF module event. See CBUFF_IRQSTATUS to know which interrupt it is.
ISP_IRQ0STATUS [20] PRV_DONE_IRQ	ISP_IRQ0ENABLE [20] PRV_DONE_IRQ	PREVIEW module: Processing-done event: triggered at the end of the frame when processing is complete for the current frame
ISP_IRQ0STATUS [19] CCDC_LSC_PREFETCH_ERROR	ISP_IRQ0ENABLE [19] PRV_DONE_IRQ	CCDC module. The prefetch error indicates when the gain table was read too slowly from memory. When this event is pending, the module goes into transparent mode (output = input). Normal operation can be resumed at the start of the next frame after: 1) Clearing this event 2) Disabling the LSC module 3) Enabling it
ISP_IRQ0STATUS [18] CCDC_LSC_PREFETCH_COMPLETED	ISP_IRQ0ENABLE [18] CCDC_LSC_PREFETCH_COMPLETED	CCDC module. Indicates the current state of the prefetch buffer. Can be used to start sending the data once the buffer is full to minimize the risk of an underflow. This event is triggered when the buffer contains 3 full paxel rows. It can be used to minimize buffer underflow risks.
ISP_IRQ0STATUS [17] CCDC_LSC_DONE	ISP_IRQ0ENABLE [17] CCDC_LSC_DONE	CCDC module. The event is triggered when the internal state of LSC toggles from BUSY to IDLE. This happens when the LSC module has completed processing the current frame.
ISP_IRQ0STATUS [16] HIST_DONE_IRQ	ISP_IRQ0ENABLE [16] HIST_DONE_IRQ	HIST module. Processing-done event: triggered at the end of the frame when processing is complete for the current frame
ISP_IRQ0STATUS [13] H3A_AWB_DONE_IRQ	ISP_IRQ0ENABLE [13] H3A_AWB_DONE_IRQ	H3A module. Auto exposure and auto white balance processing done event: Triggered at the end of the frame when processing is complete for the current frame
ISP_IRQ0STATUS [12] H3A_AF_DONE_IRQ	ISP_IRQ0ENABLE [12] H3A_AF_DONE_IRQ	H3A module. Autofocus processing-done event: triggered at the end of the frame when processing is complete for the current frame
ISP_IRQ0STATUS [11] CCDC_ERR_IRQ	ISP_IRQ0ENABLE [11] CCDC_ERR_IRQ	CCDC module. Faulty-pixel correction error: Faulty-pixel correction memory underflow. Triggered to signal an error in the faulty-pixel correction logic. The hardware did not have time to read the faulty-pixel LUT from external memory in time.
ISP_IRQ0STATUS [10] CCDC_VD2_IRQ	ISP_IRQ0ENABLE [10] CCDC_VD2_IRQ	CCDC module. Programmable event 2: triggered by the falling edge on the cam_wen signal. This event is not programmable.

Table 12-17. Camera ISP Interrupts (continued)

Event	Mask	Description
ISP_IRQ0STATUS [9] CCDC_VD1_IRQ	**ISP_IRQ0ENABLE [9] CCDC_VD1_IRQ**	CCDC module. Programmable event 1: triggered after a programmable number of horizontal lines is received after a VS pulse
ISP_IRQ0STATUS [8] CCDC_VD0_IRQ	**ISP_IRQ0ENABLE [8] CCDC_VD0_IRQ**	CCDC module. Programmable event 0: triggered after a programmable number of horizontal lines is received after a VS pulse
ISP_IRQ0STATUS [4] CSIB_IRQ	**ISP_IRQ0ENABLE [4] CSIB_IRQ**	CSIb RECEIVER module event. See CSI1_IRQSTATUS to know which interrupt it is.
ISP_IRQ0STATUS [3] CSIB_LCM_IRQ	**ISP_IRQ0ENABLE [3] CSIB_LCM_IRQ**	CSIb RECEIVER module event on memory channel.
ISP_IRQ0STATUS [0] CSIA_IRQ	**ISP_IRQ0ENABLE [0] CSIA_IRQ**	CSIa RECEIVER module event. See CSI2_IRQSTATUS to know which interrupt it is.

Note: n is equal to 0 or 1.

I is equal to 01 or 23.

Table 12-18 summarizes the CBUFF interrupts.

Table 12-18. CBUFF Interrupt Details

Event	Mask	Description
CBUFF_IRQSTATUS [5] IRQ_CBUFF1_OVR	**CBUFF_IRQENABLE [5] IRQ_CBUFF1_OVR**	Buffer overflow event: The generation of this event depends on the CBUFFx_CTRL.ALLOW_NW_EQ_CR flag and the circular buffer mode (read or write). This event indicates a bandwidth mismatch between data producer and data consumer. When it occurs, CBUFFx does NOT go into error state. However, the data in the physical buffer is very likely to be corrupted.

Table 12-18. CBUFF Interrupt Details (continued)

Event	Mask	Description
CBUFF_IRQSTATUS [4] IRQ_CBUFF1_INVALID	CBUFF_IRQENABLE [4] IRQ_CBUFF1_INVALID	Invalid access: • Camera controller writes the virtual space of CBUFFx in read mode. • Camera controller reads the virtual space of CBUFFx in write mode. • HW writes the virtual space of circular buffer x outside the CW or NW window in write mode. • HW reads the virtual space of circular buffer x outside the CW or NW window in read mode. • NW full • CPU write the DONE bit when physical buffers are not ready for the CPU. This event indicates a wrong configuration of the circular buffer, the Camera ISP, or bogus software. When it occurs, CBUFFx goes into an error state. In this state all accesses to the virtual space of CBUFFx are cancelled: they are not forwarded to the physical space. The purpose is to prevent corruption of the physical memory. The error state can be left by disabling the CBUFFx and reenabling it. Before doing so, the SW must ensure that there are no more outstanding requests to the virtual space of CBUFFx.
CBUFF_IRQSTATUS [3] IRQ_CBUFF1_READY	CBUFF_IRQENABLE [3] IRQ_CBUFF1_READY	The CPUW1 physical buffer is ready to be accessed by the CPU.
CBUFF_IRQSTATUS [2] IRQ_CBUFF0_OVR	CBUFF_IRQENABLE [2] IRQ_CBUFF0_OVR	Buffer overflow event. See description of IRQ_CBUFF1_OVR event.
CBUFF_IRQSTATUS [1] IRQ_CBUFF0_INVALID	CBUFF_IRQENABLE [1] IRQ_CBUFF0_INVALID	Invalid access. See description of IRQ_CBUFF1_INVALID event.
CBUFF_IRQSTATUS [0] IRQ_CBUFF0_READY	CBUFF_IRQENABLE [0] IRQ_CBUFF0_READY	The CPUW1 physical buffer is ready to be accessed by the CPU.

Table 12-19 describes the CSI1 receiver interrupts.

Table 12-19. CSI1 Receiver Interrupt Details

Event	Mask	Description
CSI1_IRQSTATUS [11] FS_IRQ	CSI1_IRQENABLE [11] FS_IRQ	Frame-start synchronization code detection
CSI1_IRQSTATUS [10] LE_IRQ	CSI1_IRQENABLE [10] LE_IRQ	Line-end sync code detection
CSI1_IRQSTATUS [9] LS_IRQ	CSI1_IRQENABLE [9] LS_IRQ	Line-start sync code detection
CSI1_IRQSTATUS [8] FE_IRQ	CSI1_IRQENABLE [8] FE_IRQ	Frame-end sync code detection
CSI1_IRQSTATUS [7] COUNT_IRQ	CSI1_IRQENABLE [7] COUNT_IRQ	Frame counter reached
CSI1_IRQSTATUS [5] FIFO_OVF_IRQ	CSI1_IRQENABLE [5] FIFO_OVF_IRQ	FIFO overflow error
CSI1_IRQSTATUS [3] FSP_IRQ	CSI1_IRQENABLE [3] FSP_IRQ	FSP error
CSI1_IRQSTATUS [2] FW_IRQ	CSI1_IRQENABLE [2] FW_IRQ	Frame-width error
CSI1_IRQSTATUS [1] FSC_IRQ	CSI1_IRQENABLE [1] FSC_IRQ	False sync code error
CSI1_IRQSTATUS [0] SSC_IRQ	CSI1_IRQENABLE [0] SSC_IRQ	Shifted sync code error: This interrupt cannot be triggered if the complex I/O is set in parallel mode (CSI1_CTRL [2] IO_OUT_SEL = 1).
CSI1_LCM_IRQSTATUS [1] LCM_OCPERROR	CSI1_LCM_IRQENABLE [1] LCM_OCPERROR	An OCP error occurred on the master read port.

Table 12-19. CSI1 Receiver Interrupt Details (continued)

Event	Mask	Description
CSI1_LCM_IRQSTATUS [0] LCM_EOF	**CSI1_LCM_IRQENABLE [0] LCM_OEF**	Memory read channel– End of frame: This interrupt is triggered when a frame has been completely read from memory.

Table 12-20 shows CSI2 receiver event generation through the CSI2 interrupt status and interrupt enable registers.

Table 12-20. CSI2 Receiver Event Generation (CSI2_IRQSTATUS and CSI2_IRQENABLE Registers)

Bit	Event	Clear	Description
0	CONTEXT0	No	At least one interrupt event enabled from Context 0 occurred (see Table 12-22).
1	CONTEXT1	No	At least one interrupt event enabled from Context 1 occurred (see Table 12-22).
2	CONTEXT2	No	At least one interrupt event enabled from Context 2 occurred (see Table 12-22).
3	CONTEXT3	No	At least one interrupt event enabled from Context 3 occurred (see Table 12-22).
4	CONTEXT4	No	At least one interrupt event enabled from Context 4 occurred (see Table 12-22).
5	CONTEXT5	No	At least one interrupt event enabled from Context 5 occurred (see Table 12-22).
6	CONTEXT6	No	At least one interrupt event enabled from Context 6 occurred (see Table 12-22).
7	CONTEXT7	No	At least one interrupt event enabled from Context 7 occurred (see Table 12-22).
8	FIFO_OVF_IRQ	Yes	FIFO overflow error: This interrupt is triggered on detection of a FIFO overflow. An overflow can occur if there is a mismatch between the data input and output rates. A reset of the module is required to restart correctly.
9	COMPLEXIO_ERR_IRQ	No	Error signaling from complex I/O: This interrupt is triggered when any error is received from the complex I/O (events are defined in CSI2_COMPLEXIO1_IRQSTATUS [see Table 12-21]).
11	ECC_NO_CORRECTION_IRQ	Yes	ECC was not used to correct the header because the error is larger than 1 bit (short and long packets).
12	ECC_CORRECTION_IRQ	Yes	ECC was used to correct a 1-bit error (short packet only).
13	SHORT_PACKET_IRQ	Yes	Short packet reception (other than sync events: line start, line end, frame start, and frame end; only data types between 0x8 and 0xF are considered).
14	OCP_ERR_IRQ	Yes	OCP error

Table 12-21 shows CSI2 receiver event generation through the CSI2 complex I/O interrupt status and interrupt enable registers.

Table 12-21. CSI2 Receiver Event Generation (CSI2_COMPLEXIO_IRQSTATUS and CSI2_COMPLEXIO_IRQENABLE Registers)

Bit	Event	Clear	Description
0	ERRSOTHS1	Yes	Start of transmission error for lane 1
1	ERRSOTHS2	Yes	Start of transmission error for lane 2
2	ERRSOTHS3	Yes	Start of transmission error for lane 3
5	ERRSOTSYNCHS1	Yes	Start of transmission sync error for lane 1
6	ERRSOTSYNCHS2	Yes	Start of transmission sync error for lane 2
7	ERRSOTSYNCHS3	Yes	Start of transmission sync error for lane 3
10	ERRESC1	Yes	Escape entry error for lane 1

Table 12-21. CSI2 Receiver Event Generation (CSI2_COMPLEXIO_IRQSTATUS and CSI2_COMPLEXIO_IRQENABLE Registers) (continued)

Bit	Event	Clear	Description
11	ERRESC2	Yes	Escape entry error for lane 2
12	ERRESC3	Yes	Escape entry error for lane 3
15	ERRCONTROL1	Yes	Control error for lane 1
16	ERRCONTROL2	Yes	Control error for lane 2
17	ERRCONTROL3	Yes	Control error for lane 3
20	STATEULPM1	Yes	Lane 1 in ULPM
21	STATEULPM2	Yes	Lane 2 in ULPM
22	STATEULPM3	Yes	Lane 3 in ULPM
25	STATEALLULPMENTER	Yes	All active lanes are entering in ULPM.
26	STATEALLULPMEXIT	Yes	At least one active lane exited the ULPM.

As the CSI2 receiver supports eight contexts, the CSI2_CTx_IRQSTATUS and CSI2_CTx_IRQENABLE registers are present eight times (one time per context).

The events are generated only for the enabled context(s). Table 12-22 describes the CS12 receiver event generation through the CTX interrupt status and interrupt enable registers.

Table 12-22. CSI2 Receiver Event Generation (CSI2_CTx_IRQSTATUS and CSI2_CTx_IRQENABLE Registers)

Bit	Event	Clear	Description
0	FS_IRQ	Yes	Frame start: This interrupt is triggered on the detection of a frame-start synchronization code into the CSI2 data stream.
1	FE_IRQ	Yes	Frame end: This interrupt is triggered on the detection of a frame-end synchronization code into the CSI2 data stream.
2	LS_IRQ	Yes	Line start: This interrupt is triggered on the detection of a line-start synchronization code into the CSI2 data stream.
3	LE_IRQ	Yes	Line end: This interrupt is triggered on the detection of a line-end synchronization code into the CSI2 data stream.
5	CS_IRQ	Yes	CS error: This interrupt is triggered upon detection of a mismatch between the transmitter and receiver checksums (payload).
6	FRAME_NUMBER_IRQ	Yes	Frame counter reached: This interrupt is triggered when the frame counter reaches its programmable target value.
7	LINE_NUMBER_IRQ	Yes	Line number reached: The programmable line number is received. The modulo feature can be selected (CSI2_CTx_CTRL1.LINE_MODULO). When selected, the interrupt is generated for each line number multiple of the programmed line number (CSI2_CTx_CTRL3.LINE_NUMBER); otherwise, the interrupt is generated only for the line number.
8	ECC_CORRECTION_IRQ	Yes	ECC was used to correct a 1-bit error (long packets only).

12.4 Camera ISP Functional Description

12.4.1 Block Diagram

Figure 12-53 is the Camera ISP top-level block diagram. The Camera ISP supports two simultaneous pixel flows, but only one of them can use the video-processing hardware at a time:

- Camera parallel interface (CPI) + CSIa (MIPI CSI2): CPI interface data go through the video-processing hardware. CSIa interface data go directly to memory.
- CSIa (MIPI CSI2) + CSIb (MIPI CSI1): CSIa or CSIb interface data go through the video-processing hardware. The CSI interface not using the video-processing hardware goes directly to memory.

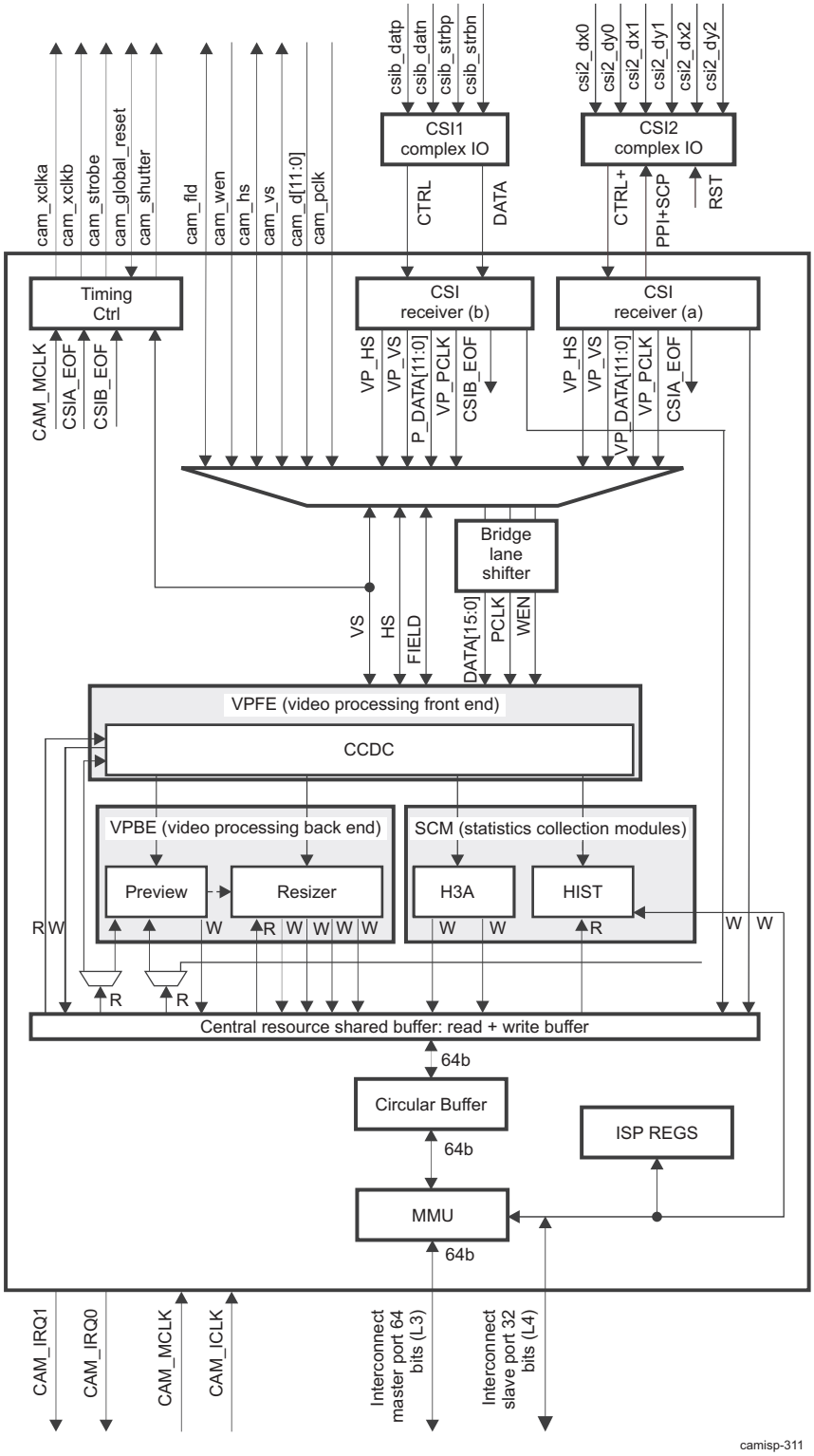
Note: Due to a pin-muxing limitation, CPI and CSIb cannot be active at the same time.

CPI + CSIa: the data width is limited to 10 bits on the parallel interface.

The Camera ISP master port is connected to the L3 interconnect, and the slave port is connected to the L4 interconnect (from the Camera ISP point of view, commands are output from the Camera ISP to the L3 and data are input/output. From the Camera ISP point of view, commands are input from the L4 to the Camera ISP and data are input/output).

Figure 12-53 shows the Camera ISP block diagram.

Figure 12-53. Camera ISP Block Diagram



The Camera ISP module comprises the following blocks:

- Timing control: Includes a timing generator and a control-signal generator:
  - The timing generator allows the generation of two clocks that can be used by the external image sensors.

  - The control-signal generator allows the generation of signals for strobe flash, mechanical shutter, and global reset.
- Two CSI receivers (one MIPI CSI1 and one MIPI CSI2): The CSI receiver interfaces with a serial camera through the complex I/O and transfers received data to memory or to the video pipeline.
- Bridge-lane shifter:
  - The data-lane shifter gives flexibility to the parallel camera connection and permits dynamic reduction of pixel data.
  - The bridge allows higher transfer rates when data is captured from the parallel interface and sent to memory.
- Video-processing front end (VPFE): Comprises the CCDC. This module provides the Camera ISP with a powerful and flexible front end interface. It directly affects the input image data:
  - The CCDC provides an interface to image sensors and digital video sources and processes image data.
- Video-processing back end (VPBE): Comprises preview and resizer modules:
  - The preview module is a parameterized hardwired image-processing block whose image-processing functions can be customized for each sensor type to realize good image quality and video frame rates for digital still camera preview displays and video-recording modes.
  - The resizer module provides a means of sizing the input image data to the desired display or video-encoding resolution. Zoom is limited to 4x in both vertical and horizontal directions for each pass. After one (on-the-fly) upscaling pass, the image can be sent to memory and then resent through the resizer.
- Statistics-collection modules (SCM): H3A and histogram modules that provide statistics on the incoming images to help designers of camera systems:
  - The hardware 3A module supports the control loops for AF, AWB, and AE by collecting metrics about RAW image data from the CCDC.
  - The histogram module bins input color pixels, depending on the amplitude, and provide statistics required to implement various 3A (AE/AF/AWB) algorithms and tune the final image/video output. The histogram module can operate on RAW image data from CCDC or memory.
- Central-resource shared buffer logic (CRSBL): Buffers and schedules memory accesses requested by the camera modules
- Circular buffer: Avoids storage of full image frames in the memory when the data must be post and/or preprocessed by software.
- MMU: Performs virtual-to-physical address translation between its interconnect slave and interconnect master access ports

12.4.1.1 Possible Data Paths Inside the Camera ISP

Data paths inside the Camera ISP hardware depend on the image format sourced by the sensor (RAW RGB, YUV4:2:2, JPEG,...).

Table 12-23 lists the modules used for the different data types. The formats described in the columns are the formats at the inputs of the CCDC (after the bridge-lane shifter). It is the internal parallel format.

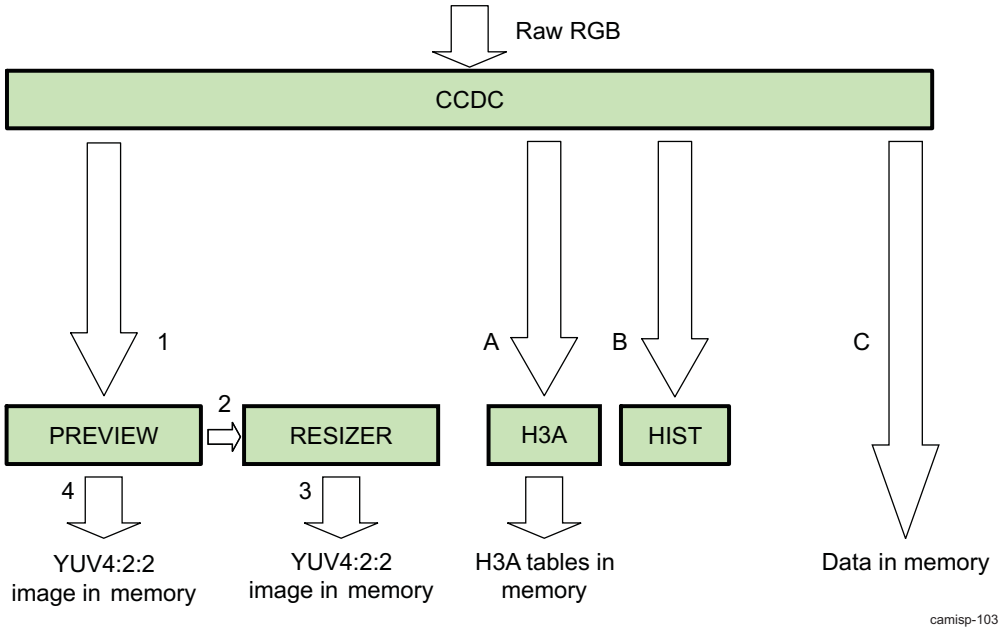
Table 12-23. Allowed Data Flows for Hardware at the Input of the CCDC Module

Module	6/7 Bits RAW Format	8/10 Bits RAW Format	11/12/14 Bits RAW Format	YUV4:2:2 Format	Other Formats (JPEG,RGB4:4:4,...)
CCDC	+	+	+	+	+ All formats not treated by hardware
Preview	+	+			
Resizer	+	+		+	
H3A	+ Except AF	+ Except AF for RAW8			
Histogram	+	+			

12.4.1.1.1 RGB RAW Data

Figure 12-54 shows the data path of images in RAW format.

Figure 12-54. Camera ISP/Data Path/RAW RGB Images



RAW data are processed through the CCDC module and are directly pipelined to the preview engine. Another way is to output directly from the CCDC to memory (C) In the preview block; the format is converted from RAW data to YUV4:2:2 (1). The data can be output to memory (4) or pipelined to the resizer (2). The rescaled YUV4:2:2 image is finally stored in memory (3).

In parallel, processed data in the CCDC are used by the H3A module (A), which writes tables of statistics in memory, and by the HIST module (B). The results of the HIST modules are stored in status registers in the HIST module.

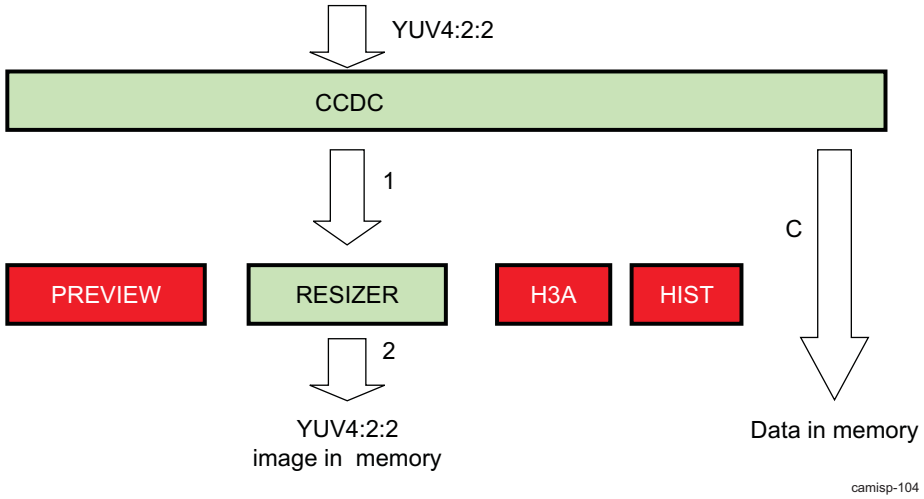
Notes:

- This data path is correct for RAW10 data.
- The data path for RAW8 data is similar to that for RAW10, except that the autofocus module in the H3A does not support RAW8.
- RAW11 data must be sent directly to memory (C).
- RAW12 data must be sent to memory (C) or pixel dynamic must be reduced RAW8 or RAW10 by the bridge-lane shifter module, before the CCDC.
- RAW14 data must be sent to memory (C) or pixel dynamic must be reduced to RAW8 or RAW10 by the bridge-lane shifter module, before the CCDC.

12.4.1.1.2 YUV4:2:2 Data

Figure 12-55 shows the data path of images in YUV4:2:2 format.

Figure 12-55. Camera ISP/Data Path/YUV4:2:2 Images



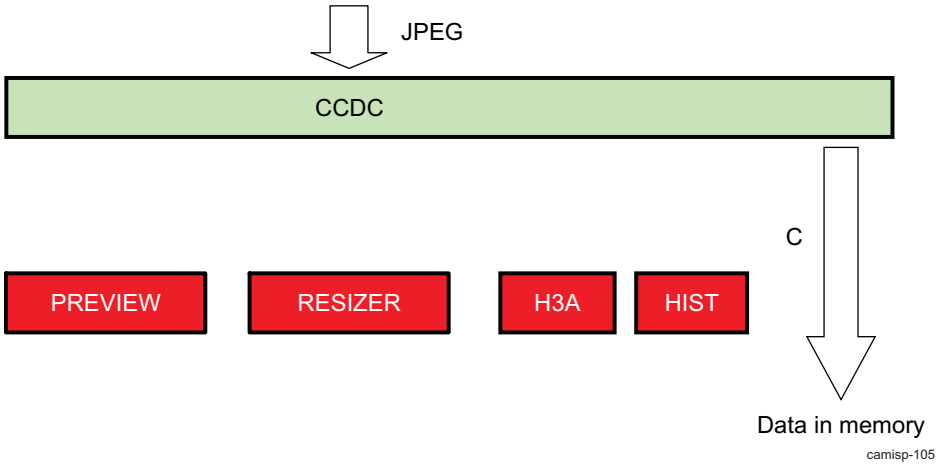
When the sensor-output format is YUV4:2:2, the CCDC block is directly pipelined to the resizer (1) or output directly to the memory (C). The rescaled YUV4:2:2 images are finally stored in memory (2).

The modules in red in Figure 12-55 are not used when the format is YUV4:2:2.

12.4.1.1.3 JPEG Data

Figure 12-56 shows the data path of images in JPEG format.

Figure 12-56. Camera ISP/Data Path/JPEG Images



When the sensor output format is JPEG, the CCDC block is directly output to the memory (C).

The modules in red in Figure 12-56 are not used when the format is JPEG.

12.4.2 CSI1 Receiver

12.4.2.1 CSI1 Receiver Features

The CSI1 receiver features are listed below:

- Image from sensor
  - Transfer of pixels and data received by the CSI1 DSI_PHY RX to system memory or to the video pipeline
  - Unidirectional data link
  - 1D and 2D addressing mode
  - False synchronization code protection
  - Ping-pong mechanism for double-buffering
  - Support of RGB, RAW, YUV, and JPEG formats
- Image read from memory
  - RAW formats supported

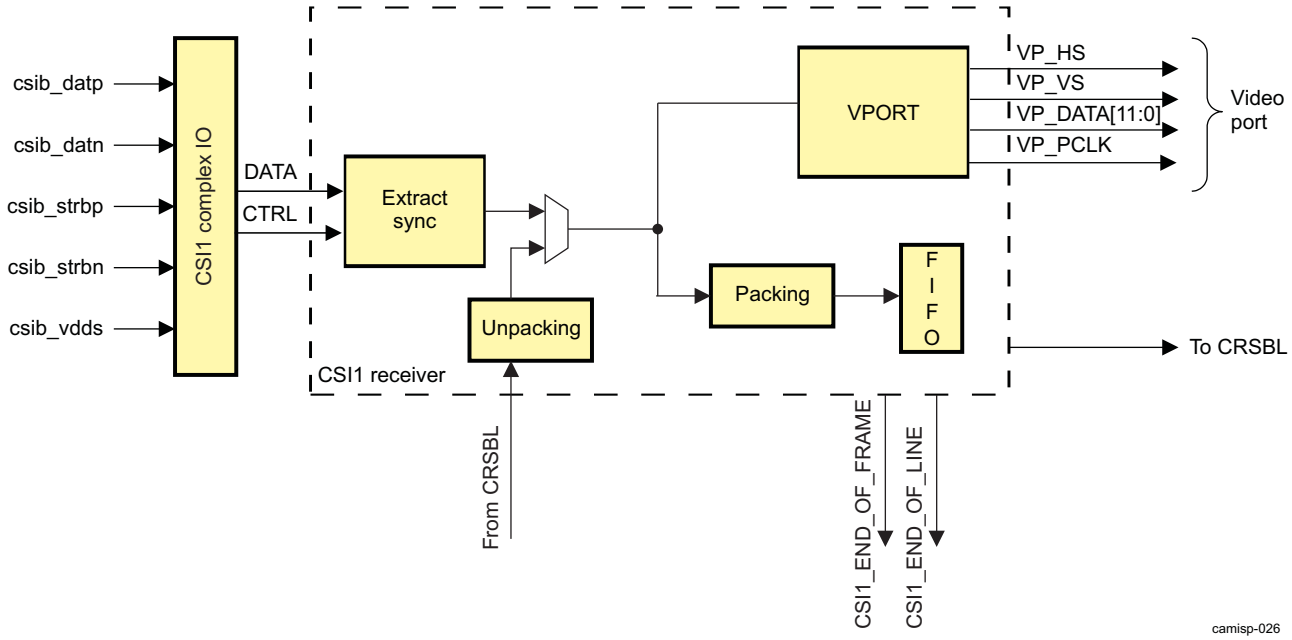
12.4.2.2 CSI1 Receiver Functional Description

12.4.2.2.1 Overview

Figure 12-57 is the CSI1 receiver top-level block diagram. The CSI1 receiver takes serial data from a CSI1-compatible image sensor, converts it to parallel data, extracts the logical channels, detects and extracts the synchronization codes, reformats the data, and outputs it through the video-port interface or the interconnect interface.

The CSI1 receiver video-port interface is connected to the video-preprocessing hardware.

Figure 12-57. CSI1 Receiver Block Diagram



12.4.2.2.2 Complex I/O

The CSI1 complex I/O is controlled by two other SCM registers: CONTROL.CONTROL_CSIRXFE and CONTROL.CONTROL_CSI:

- CONTROL.CONTROL_CSIRXFE:
  - The CSIb interface can be reset by setting the CONTROL.CONTROL_CSIRXFE[13] CSIB_RESET bit to 0.
  - The CSIb differential transceiver is activated through the CONTROL.CONTROL_CSIRXFE[12]

CSIB_PWRDNZ bit.
  - The clock polarity can be changed through the CONTROL.CONTROL_CSIRXFE[7] CSIB_INV bit.

  A 5-bit resistor trim is implemented in the CSI complex I/O. The trim is done for the termination resistors between the csib_datp/csib_datn and csib_strbp/csib_strbn pins. The same trim-bits are used to trim both resistors. This resistor value is set by eFuse, but can be overridden through the SCM:
- CONTROL.CONTROL_CSI:
  - The resistor trimming value set by eFuse can be read through the CONTROL.CONTROL_CSI[25:21] CSI_B_TRIM_FUSE bit field for CSIb.
  - The CONTROL.CONTROL_CSI[31] CSI_B_MUX_CTRL bit permits to override CSIb receiver trimming eFuse.
  - The CONTROL.CONTROL_CSI[20:16] CSI_B_TRIM_REG bit field contains the value to replace eFuse one for CSIb.
  - The CONTROL.CONTROL_CSIRXFE[8] CSIB_RESENABLE bit must be set to 1 to enable the CSIb internal resistor.

12.4.2.2.3 Physical Layer

The CSI1 serial interface is a unidirectional differential serial interface. The physical layer of the CSI1 is based on SubLVDS signaling.

Data-clock signaling consists of two parallel signals: data and clock:

- The data signal carries bit-serial data. The CSI1 transmitter writes the data on each falling edge of the clock. The data is usually transmitted bytewise, LSB first.
- The data clock signal carries the clock signal. The CSI1 transmitter writes the data on each falling edge of the clock. The CSI1 receiver reads the data on the rising edge of the clock.

12.4.2.2.4 Protocol Layer

The CSI1 protocol layer defines how image-sensor data are transported to the physical layer.

The CSI1 protocol layer transports logical channels, which are composed of frames. A frame comprises embedded data and image-sensor data. Each frame is clearly identified by unique synchronization codes: frame start, frame end, line start, and line end. Image-sensor data can have multiple data types (select the data type in the CSI1_CTRL1 [7:3] FORMAT bit field). For details about data types, synchronization code, and FSP encoding/decoding, see Section 12.2.4.4.

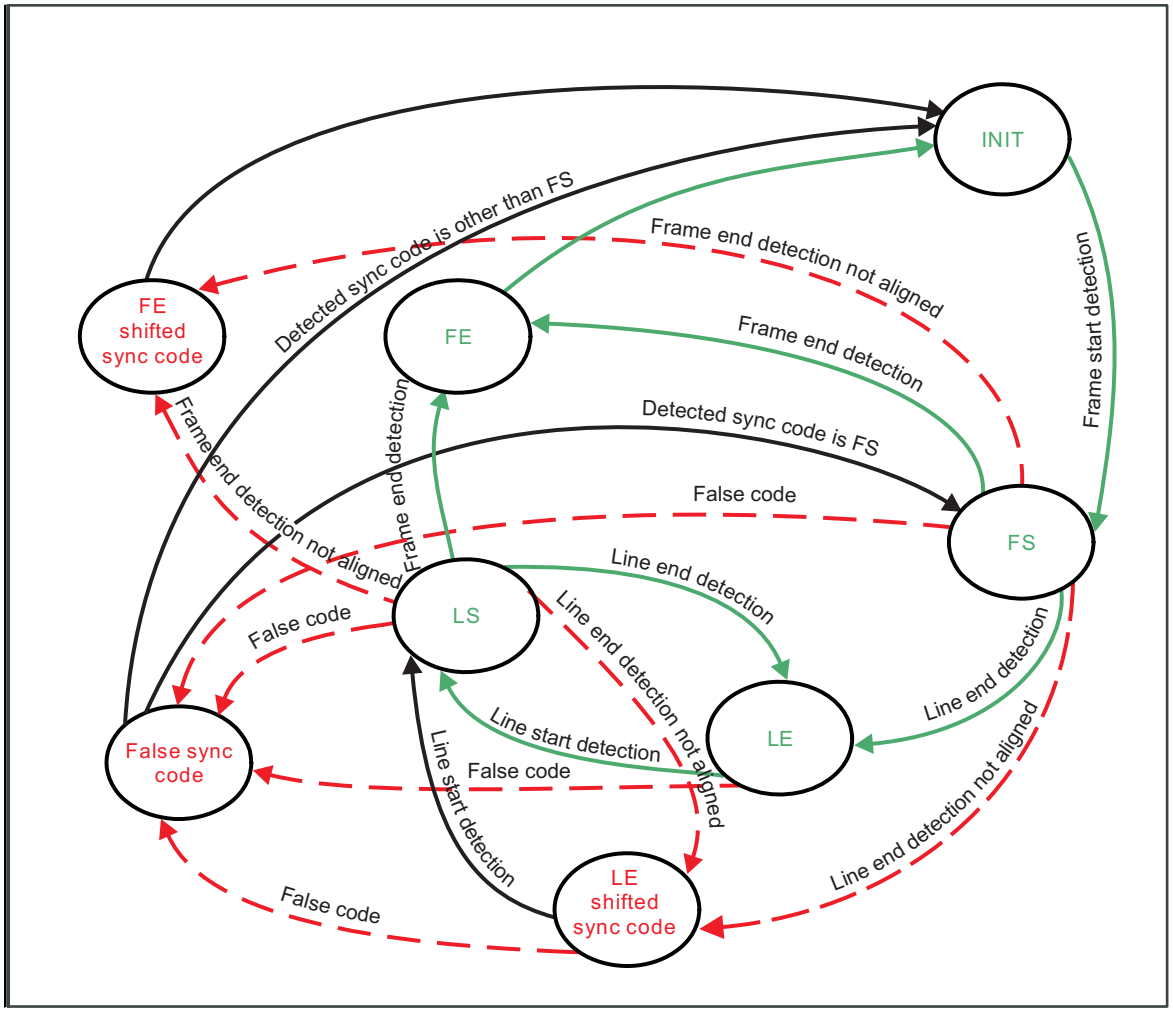
12.4.2.2.4.1 Synchronization Finite State-Machine

Figure 12-58 shows the CSI1 receiver FSM. The CSI1 receiver synchronization operates bitwise.

The expected synchronization code order is FSC, LEC, LSC, and LEC...etc., LSC, LEC, LSC, FEC for all data types except JPEG8. For JPEG8, the expected synchronization code order is FSC, FEC.

If the synchronization code order is not respected, the state-machine goes to FalseSyncCode state. Because line length is always a multiple of 32 bits, the LEC and FEC codes are always aligned on a 32-bit boundary. If LEC or FEC is not aligned on a 32-bit boundary, the state-machine goes to LEShiftedSyncCode or FEShiftedSyncCode state. Figure 12-58 shows the synchronization state-machine.

Figure 12-58. Synchronization State-Machine



In case of synchronization code errors, the CSI1 receiver can reinitialize itself. No software intervention is required for most synchronization-code errors:

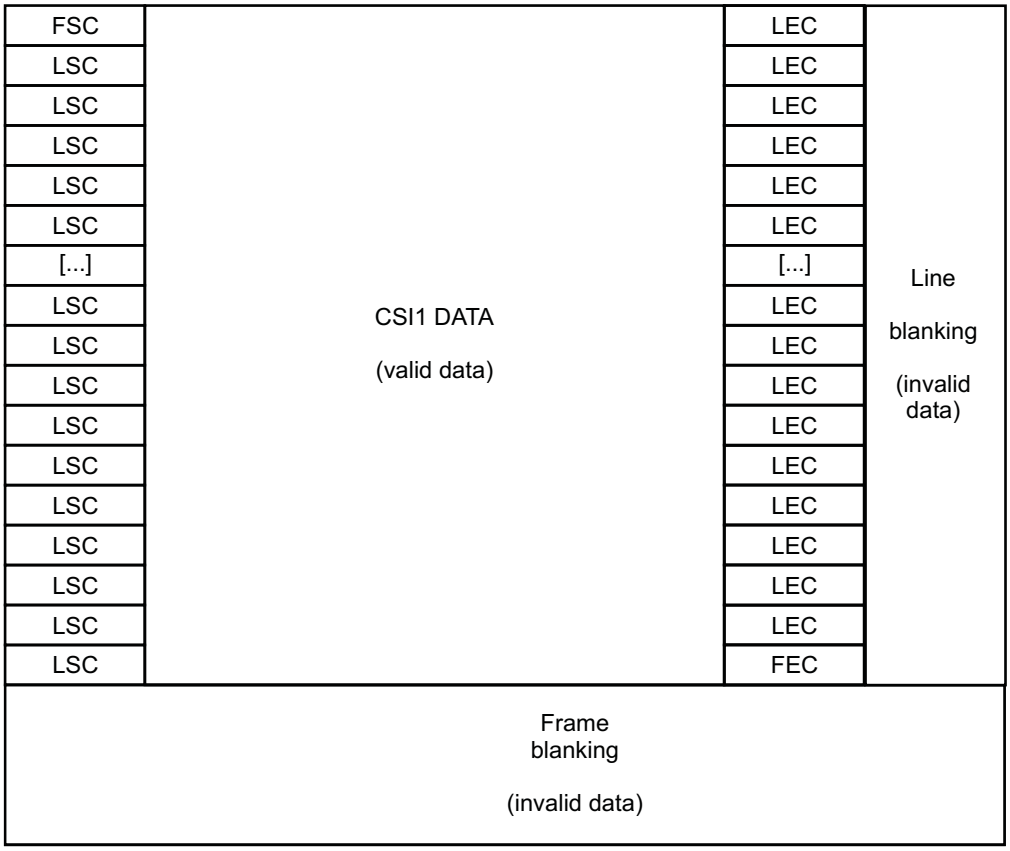
- Line-end shifted code: The receiver either removes the additional bits or adds dummy bits. The next LS synchronization code resynchronizes the state-machine to normal behavior. A shifted line-end (LE) synchronization code triggers an LE_IRQ event.
- Frame-end shifted code: The receiver either removes the additional bits or adds dummy bits. The next FS synchronization code resynchronizes the state-machine to normal behavior. A shifted frame-end (FE) synchronization code triggers an FE_IRQ event.
- False synchronization code: The current frame is lost. The receiver stops acquiring data, flushes its internal FIFO, and asserts the FSC_IRQ event. The next FS synchronization code resynchronizes the state-machine to normal behavior.

12.4.2.2.4.1.1 Frames

Structure

Figure 12-59 shows the generic (non-JPEG) description of a CSI1 frame with synchronization codes. Each CSI1 frame line is composed of a finite number of 32-bit words.

Figure 12-59. CSI1 Frame Structure: Non-JPEG Data Format

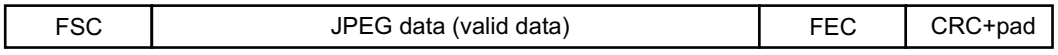


camisp-028

The period between LEC and new is called the line-blanking period. The time between FEC and new FSC is called the frame-blanking period. The CSI1 receiver works with line-blanking periods set to 0 or longer.

Figure 12-60 shows CSI1 frame structure for the JPEG8 case. The JPEG stream is composed of a finite number of 32-bit words. LSC and LEC synchronization codes are never used when the CSI1 interface transports a JPEG bitstream; only FSC and FEC synchronization codes are used.

Figure 12-60. Frame Structure: JPEG8 Data Format



camisp-196

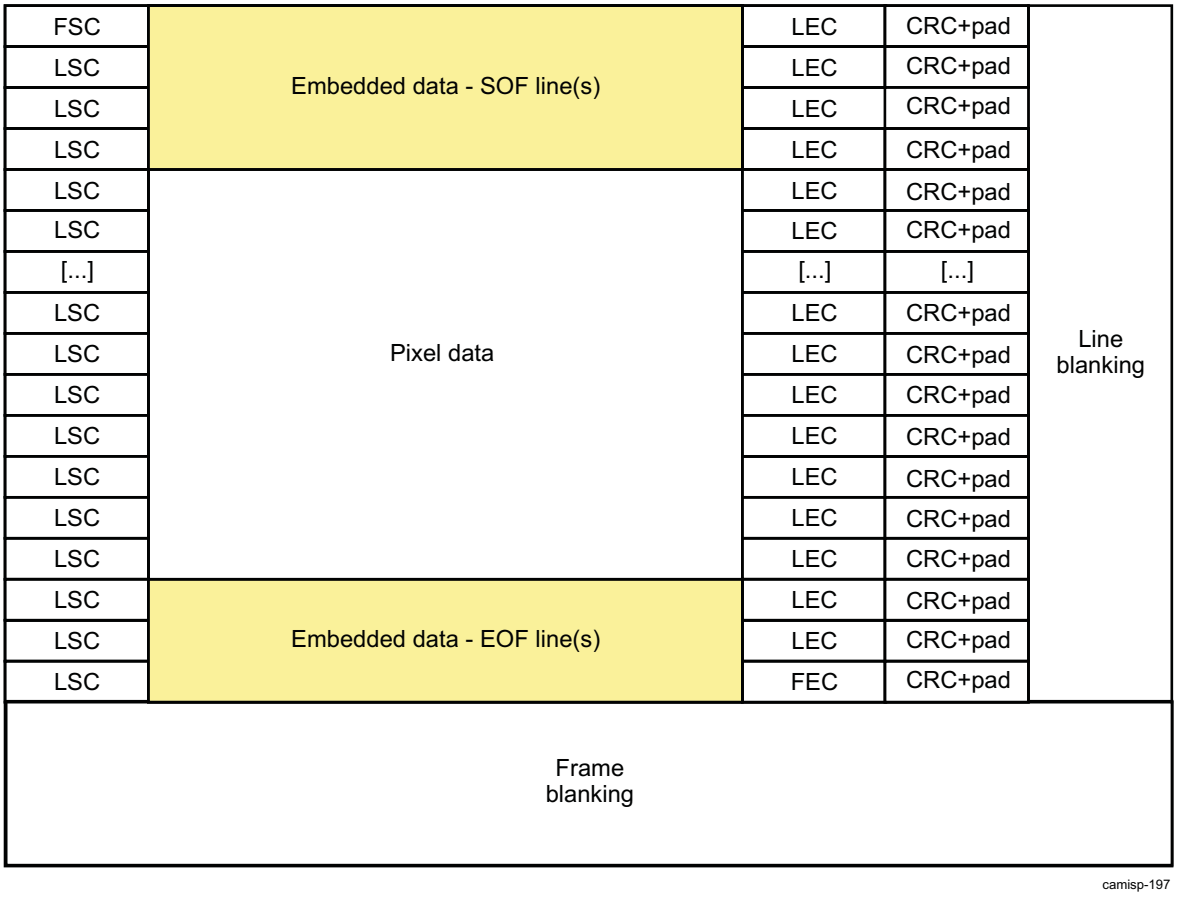
Data

A frame comprises embedded data and image-sensor data. Figure 12-61 shows the location of embedded data and image-sensor data in the frame. The following definitions apply:

- 0 or more start-of-frame (SOF) status lines can be embedded at the beginning of a CSI1 frame.
- Image data comprises pixels of the same data formats. It may contain visible or nonvisible pixels.
- 0 or more end-of-frame (EOF) status line(s) can be embedded at the end of a CSI1 frame.
- SOF lines, pixel data, and EOF lines do not overlap.

The CSI1 receiver does not use the information contained in the status lines. However, it extracts it and stores it in memory for use by the software. Figure 12-61 shows the CSI1 data structure.

Figure 12-61. CSI1 Data Structure



Embedded information (SOF and EOF lines):

- Embedded information is never compressed .
- Embedded information covers full lines.
- Embedded information is not encoded in the same data format as pixel data. The CSI1 receiver extracts embedded information, but does not modify the data format.
- False-synchronization-code protection is implemented; the receiver ensures that the embedded data contains no synchronization codes.

Pixel data:

- Pixel data can be compressed.
- Pixel data comprises pixels of the same data format.
- False-synchronization-code protection is implemented; the receiver ensures that the pixel data contains no synchronization codes.

12.4.2.2.5 Memory Read Channel

The memory channel can perform the following operations:

- Reads data from memory. It is unpacked if necessary.
- Can send the data to the video pipeline
- Can send the data back to memory. It can be packed before it is sent to memory.

It cannot receive its input data directly from the sensor, and the logical channels are disabled when the memory channel is enabled.

12.4.2.2.5.1 Video Port

The video port always receives unpacked data. It can be enabled using the CSI1_LCM_CTRL [2] DST_PORT register. Its clock can be selected with the CSI1_CTRL [9:8] VP_OUT_CTRL register.

The data format used by the video port is defined by the CSI1_LCM_CTRL [26:24] DST_FORMAT register.

12.4.2.2.5.2 Encode, Pack, and Store Data

This stage is used only when data are sent to memory. Memory destination is selected using the CSI1_LCM_CTRL [2] DST_PORT register. The output data format is defined by the CSI1_LCM_CTRL [26:24] DST_FORMAT, and CSI1_LCM_CTRL [31] DST_PACK registers.

The destination address and offset for the output data of the memory channel are set by the CSI1_LCM_DST_ADDR and CSI1_LCM_DST_OFST registers.

12.4.3 CSI2 Receiver

12.4.3.1 CSI2 Receiver Features

The CSI2 receiver module is a master on the L3 interconnect for storing data in memory and a slave on the L4 interconnect for register access.

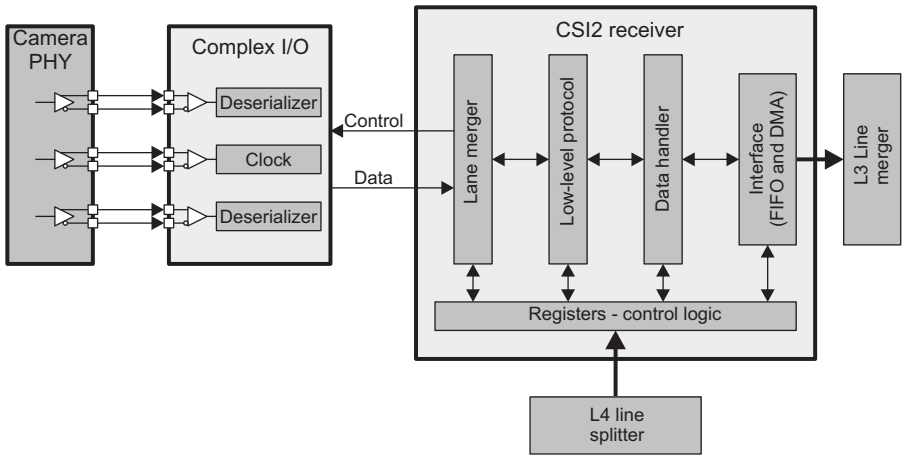
The main features of the CSI2 receiver module are:

- Transfer pixels and data received by the CSI2 complex I/O to the system memory
- Unidirectional data link
- Minimum of one and maximum of two configurable data links in addition to clock signaling
- Maximum data rate of 800M bps per data pair
- Data merger for 2-data lane configuration
- Error detection and correction by the protocol engine
- DMA engine integrated with dedicated FIFO
- 1D and 2D addressing modes
- Up to eight contexts to support eight dedicated configurations of virtual channel ID and data types
- Ping-pong mechanism for double-buffering
- JPEG support for unknown length transfer (no extraction of the thumbnail)
- All primary and secondary MIPI-defined formats are supported (RGB, RAW, and YUV)
- Storage in progressive mode for interlaced stream (using line numbering)
- Configuration of the complex I/O
- Fully configurable interface of the complex I/O: position of the clock and data and order of +/- differential signals for each pair

12.4.3.2 CSI2 Receiver Block Diagram

Figure 12-62 is the block diagram of the CSI2 receiver connected to the complex I/O.

Figure 12-62. CSI2 Receiver Block Diagram



12.4.3.3 Physical Layer Lane Configuration

The CSI2 serial interface is a unidirectional differential serial interface with data/clock for the physical layer. The CSI2 PHY is based on the MIPI DSI_PHY Specification version 0.89.

The maximum CSI2 receiver data transfer capacity is 800M bps per data lane.

Data-clock signaling consists of four to six signals: one or two data lanes and one clock lane:

- The data signal carries the bit-serial data. The CSI2 transmitter in the image sensor sends the data in-quadrature with the DDR clock in HS mode; otherwise, the clock is extracted from the received data in LS mode. Data is transmitted byte-wise, LSB first. The CSI2 complex I/O receives the data and sends the byte stream to the CSI2 receiver.
- The clock signal carries the DDR clock signal.

Each physical lane can be a data or a clock lane. The clock/data lane must be configured before transmission to indicate the byte order while merging the received bytes into a byte stream.

Lanes are configured through the CSI2_COMPLEXIO_CFG1 register. The CLOCK_POSITION and CLOCK_POL fields configure which lane transmits the clock and define its polarity. Four fields (DATA1_POSITION, DATA1_POL, DATA2_POSITION, and DATA2_POL) configure the data lanes and their polarity. The DATA2_POSITION field can be set to 0; in this case, only the data lane defined in the DATA1_POSITION field is used, and data is transmitted on only one clock lane and one data lane.

The CSI2_COMPLEXIO_CFG1 register also contains a bit field affecting complex I/O power management.

12.4.3.4 ECC and Checksum Generation

The CSI2 receiver includes an ECC in the packet header and a checksum in the packet footer for long-packet transmission. These two fields can be used to detect and/or correct errors in the received packet.

12.4.3.4.1 ECC

To detect and correct transmission errors of the header of short and long packets, an 8-bit ECC is included in the header of packets (short and long packet).

The ECC concerns all the fields for a short packet (data ID and short-packet data field) and the packet header for long packet (data ID and word count). The ECC can only correct one error. Additional errors cannot be repaired, but they are flagged.

The CSI2 receiver ECC is compared against the CSI2 transmitter ECC embedded in the bit stream. If the ECC does not match, an interrupt is triggered to the host CPU.

For both long and short packets, the correction is always done if there is only one error per packet.

An ECC error with or without correction can be reported at two levels, depending on the type of packet. Table 12-24 describes the field where events are logged. Logging cannot be disabled, but users can set the corresponding bit in the CSI2_IRQENABLE and CSI2_CTx_IRQENABLE registers to prevent event generation at a higher level.

Table 12-24. ECC Event Logging

	Short Packet	Long Packet
With correction	Global CSI2_IRQSTATUS[12] ECC_CORRECTION_IRQ	Context CSI2_CTx_IRQSTATUS[8] ECC_CORRECTION_IRQ
Without correction	Global CSI2_IRQSTATUS[11] ECC_NO_CORRECTION_IRQ	Global CSI2_IRQSTATUS[11] ECC_NO_CORRECTION_IRQ

The ECC check can be disabled (short and long packet) by writing 0 in the CSI2_CTRL[2] ECC_EN bit field. Writing 1 enables the ECC check.

12.4.3.4.2 Checksum

To detect errors in transmission of the payload of long packets, a 16-bit CRC checksum is computed on the payload of the long packets in the transmitter. This CRC is stored in the packet footer. A CRC is also computed in the CSI2 receiver. If the checksums do not match, an event is triggered to the host CPU.

CRC errors are logged in the CS_IRQ field of the corresponding context register, CSI2_CTx_IRQSTATUS. Logging cannot be disabled, but users can set the corresponding bit in the CSI2_CTx_IRQENABLE register to prevent event generation at a higher level.

The CRC check can be disabled for a specific context by writing 0 in the CSI2_CTx_CTRL1[5] CS_EN bit. Writing 1 enables the CRC check.

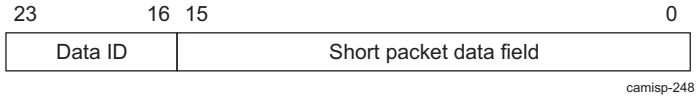
12.4.3.5 Short Packet

There are two types of short packets in the CSI2 receiver:

- Synchronization short packet: Used by the protocol engine to synchronize frame and line (data ID from 0x0 to 0x7)
- Generic short packet: User-dependent; not treated by the protocol engine (data ID from 0x8 to 0xF)

When a generic short packet is received by the CSI2 receiver, the ECC check is performed if it is enabled (see Section 12.4.3.4.1, *ECC*). Then, the short packet is written in the CSI2_SHORT_PACKET[23:0] SHORT_PACKET bit field. The ECC field is deleted from the short packet. Figure 12-63 shows the SHORT_PACKET field format.

Figure 12-63. SHORT_PACKET Field Format



An event is logged when a short packet is stored in the SHORT_PACKET_IRQ bit in the CSI2_IRQSTATUS[13] register. Logging cannot be disabled, but users can set the corresponding bit in the CSI2_IRQENABLE register to prevent event generation at a higher level.

The application reads the CSI2_SHORT_PACKET register before the next short packet with a code between 0x8 and 0xF. There is a single register for capturing the generic short packet, because no data type in it is associated with context.

12.4.3.6 Virtual Channel and Context

The CSI2 protocol layer transports virtual channels. The purpose of virtual channels is to separate different data flows interleaved in the same data stream. Each virtual channel is identified by a unique channel identification number in the packet header. This channel identification number is encoded in the 2-bit code.

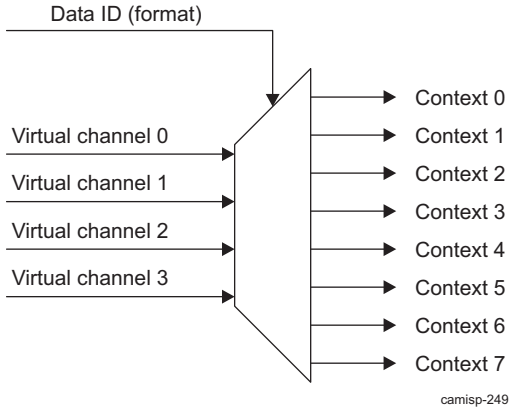
The CSI2 receiver monitors the channel identifier number and demultiplexes the interleaved data streams. The CSI2 receiver supports up to four concurrent virtual channels.

The CSI2 receiver supports eight contexts to control the four possible virtual channels and the different data transmitted through them. A context is linked to a specific data type transported by a given virtual channel. The following two bit fields permit configuration of a context:

- CSI2_CTx_CTRL2[12:11] VIRTUAL_ID: Configures the virtual ID linked to the current context
- CSI2_CTx_CTRL2[9:0] FORMAT: Configures the data format linked to the current context

Figure 12-64 shows the relationships between virtual channels and contexts.

Figure 12-64. Virtual Channel to Context



Each context consists of eight registers: six registers to control the corresponding context and two to log and enable events from the context. All registers in a context can be modified at any time; however, modifications apply only from the start of the following frame.

A context can be enabled independently by writing 1 in the CSI2_CTx_CTRL1[0] CTX_EN bit field; writing 0 disables the corresponding context.

When acquiring frames on a context, users can write the number of frames to capture in the CSI2_CTx_CTRL1[15:8] COUNT bit field. Acceptable values are 0:255; 0 stands for infinite capture (no count). After each frame acquired, the count value is decremented by 1. When the count value reaches 0, the CSI2_CTx_IRQSTATUS[6] FRAME_NUMBER_IRQ event is set and the CTX_EN bit is set to 0. To write a value in the COUNT bit field, the CSI2_CTx_CTRL1[4] COUNT_UNLOCK bit must be set to 1. If the COUNT_UNLOCK value is 0, a write in the COUNT bit field has no effect.

The CSI2_CTx_CTRL3[15:0] LINE_NUMBER bit field configures the generation of the CSI2_CTx_IRQSTATUS[7] LINE_NUMBER_IRQ event. The CSI2_CTx_CTRL1[1] LINE_MODULO bit configures how the LINE_NUMBER event is generated:

- 0: The event is generated one time by frame.
- 1: The event is generated modulo LINE_NUMBER (the event can be generated more than once in a frame).

During a frame capture, the CSI2_CTx_CTRL2[31:16] FRAME_NUMBER bit field shows the number that identifies the frame received.

12.4.3.7 DMA Engine

The CSI2 receiver integrates its own DMA engine with dedicated FIFO.

Global DMA configuration (single access, non-streaming and posted writes) is common to the eight channels and is defined in the CSI2_CTRL register. Configuration of the ping-pong address and the offset between lines is specific for a given context; therefore, each context has its own DMA configuration registers.

The DMA engine supports the following requests:

- Single write

When an element (the size depends on the data type) is present in the FIFO, the DMA engine initiates a single write.

All single requests sent to the interconnect are back-to-back requests with no idle, if the FIFO has enough data to supply the DMA.

The DMA starts to write in memory using the CSI2_CTx_DAT_PING_ADDR[31:5] ADDR bit field for the first frame to be transferred and then uses the CSI2_CTx_DAT_PONG_ADDR[31:5] ADDR bit field and the ping address alternately. So, the first frame uses the ping address, the second frame uses the pong address, the third frame uses the ping address, and so on.

The CSI2_CTx_CTRL1[3] PING_PONG status bit indicates whether the ping address (CSI2_CTx_DAT_PING_ADDR) or the pong address (CSI2_CTx_DAT_PONG_ADDR) was used to store the pixel data of the last frame. After reset or after a 0-to-1 edge transition in the CSI2_CTRL[0] IF_EN register, the pixel data is written in the ping buffer and CSI2_CTx_CTRL1[3] PING_PONG = PONG. When the number of FECs received equals the value programmed in the SI2_CTX_CTRL1[23:16] FEC_NUMBER bit field, the pixel data are written in the pong buffer and CSI2_CTx_CTRL1[3] PING_PONG = PING. CSI2_CTx_CTRL1[3] PING_PONG toggles after the CSI2_CTx_CTRL1[23:16] FEC_NUMBER FEC sync code with the virtual channel ID defined is received in the CSI2_CTx_CTRL2[12:11] VIRTUAL_ID bit field.

The CSI2_CTx_CTRL1[23:16] FEC_NUMBER bit field must be set as follows:

- In progressive mode, set to 1.
- In interlaced mode, set to the number of interlaced frames to recreate a progressive image in the PING_PONG buffer.

12.4.3.7.1 Progressive Frame to Progressive Storage

After each line, a new start line address is computed, depending on the value of the CSI2_CTX_DAT_OFST[15:5] OFST bit field:

- If OFST = 0, the new line starts immediately after the last pixel (data are written contiguously in memory).
- Otherwise, the OFST value sets the offset between the first pixel of the previous line and the first pixel of the current line in memory.

For the ping frame:

```
@Line0 = CSI2_CTX_DAT_PING_ADDR @Line1 = @Line0 + CSI2_CTX_DAT_OFST

@Line2 = @Line1 + CSI2_CTX_DAT_OFST
```

For the pong frame:

```
@Line0 = CSI2_CTX_DAT_PONG_ADDR @Line1 = @Line0 + CSI2_CTX_DAT_OFST

@Line2 = @Line1 + CSI2_CTX_DAT_OFST
```

12.4.3.7.2 Interlaced Frame to Progressive Storage

The mode is functional only when the line numbers are transmitted. It is automatically enabled without setting.

For the ping frame:

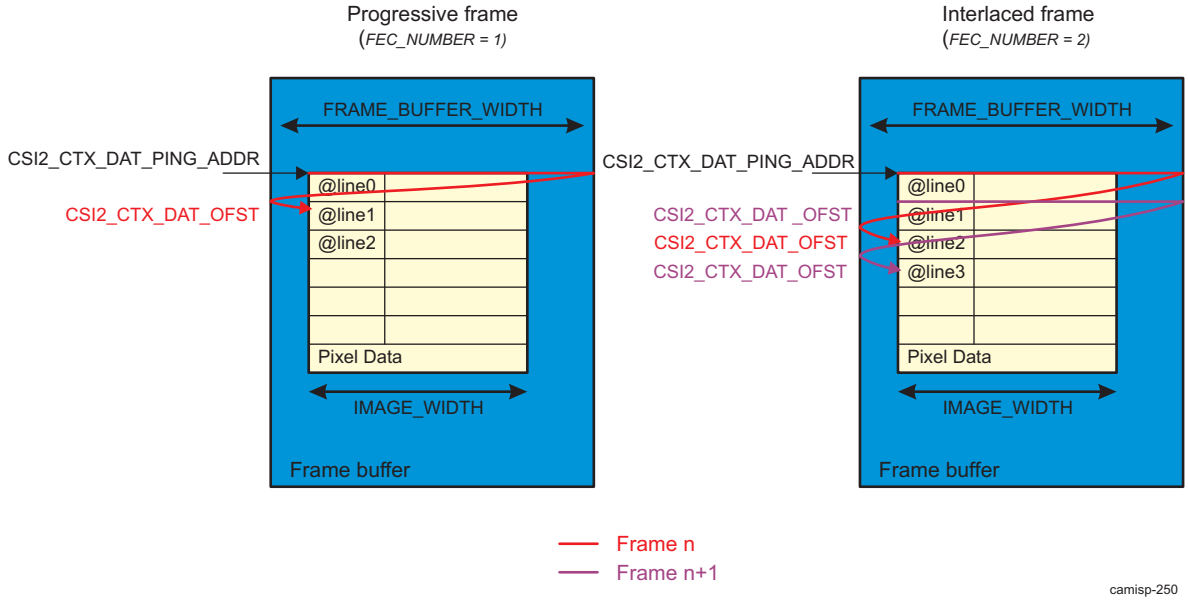
```
@LineX = CSI2_CTX_DAT_PING_ADDR + CSI2_CTX_DAT_OFST * Line_Number
```

For the pong frame:

```
@LineX = CSI2_CTX_DAT_PONG_ADDR + CSI2_CTX_DAT_OFST * Line_Number
```

Figure 12-65 shows how data are stored in memory regarding DMA configuration.

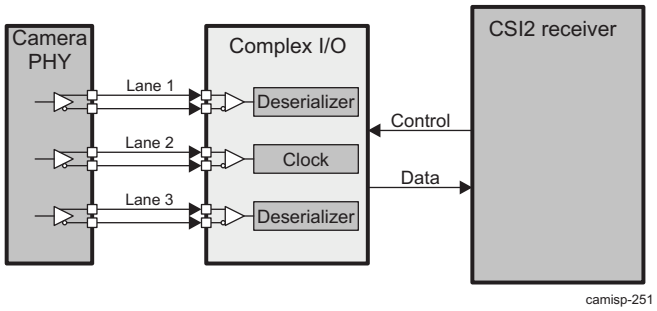
Figure 12-65. Pixel Data Destination Setting in Progressive and Interlaced Mode



12.4.3.8 Complex I/O

The complex I/O is the interface between the CSI2 transmitter (camera sensor) and the CSI2 receiver. This module transforms the bit stream divided into one or two serial data lanes into a bit stream compatible with the CSI2 receiver and one clock lane. Figure 12-66 shows the CSI2 receiver.

Figure 12-66. Complex I/O Overview



The CSI2_COMPLEXIO1_IRQSTATUS register logs complex I/O events of the following types:

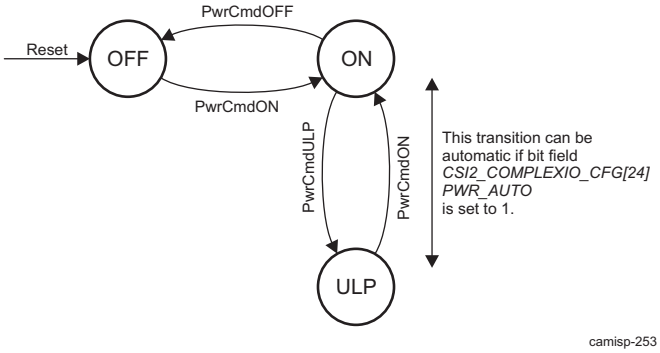
- Line power state change (all lanes in ULPM, at least one lane exits ULPM, etc.)
- Error on one lane

The CSI2 receiver embeds two registers to configure/read some complex I/O parameters:

- The CSI2PHY_CFG0 register detects clock miss with respect to the MIPI specification and control timing.
- The CSI2PHY_CFG1 register reports completion of reset on the different parts of the module and configures timing parameters.

The complex I/O has three power modes: on, off, and ULP (ultra-low power). These modes can reflect the ON or ULP power state of the three differential lines if the CSI2_COMPLEXIO_CFG1[24] PWR_AUTO bit is set to 1. If the PWR_AUTO bit is at reset value (0), the complex I/O power state is controlled by the CSI2_COMPLEXIO_CFG1[28:27] PWR_CMD bit field, which directly defines the power state. Figure 12-67 shows the complex I/O power finite state machine (FSM).

Figure 12-67. Complex I/O Power FSM



Another register, CSI2_TIMING, is used to control the power state of the complex I/O module with regards to the differential line state. This register is used to control the mode of the complex I/O (RxMode or NoRxMode) and the delay between all the differential lines on STOP state and complex I/O on NoRxMode. The CSI2_TIMING[15] FORCE_RX_MODE_IO1 bit field sets the complex I/O in RxMode or in NoRxMode (stopped mode). The FORCE_RX_MODE_IO1 bit is automatically reset to 0 by hardware when the counter ends and the FSM returns to NoRxMode. Three bit fields (CSI2_TIMING[14] STOP_STATE_X16_IO1, CSI2_TIMING[13] STOP_STATE_X4_IO1, and CSI2_TIMING[12:0] STOP_STATE_COUNTER_IO1) configure the delay between line stop mode and complex I/O stop mode. The delay represents the number of functional clock (CAM_FCLK) cycles and can be calculated as follows:

Total delay in CAM_FCLK cycle = CSI2_TIMING.STOP_STATE_COUNTER_IO1 x (1 + CSI2_TIMING.STOP_STATE_X16_IO1 x15) x (1 + CSI2_TIMING.STOP_STATE_X4_IO1 x 3).

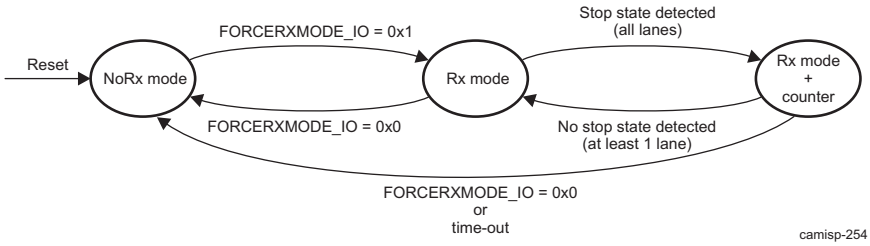
Table 12-25 lists the possible values of the delay, in terms of the CAM_FCLK cycles, depending on the values of the STOP_STATE_X16_IO1 and STOP_STATE_X4_IO1 bits.

Table 12-25. Possible Time-Out Value for RxMode Counter

STOP_STATE_X16_IO1	STOP_STATE_X4_IO1	Possible Delay Value (In Functional Clock Cycles)
0x0	0x0	8191 (with step of 1)
0x0	0x1	32764 (with step of 4)
0x1	0x0	131056 (with step of 16)
0x1	0x1	524224 (with step of 64)

The state-machine controlling the RxMode is presented in Figure 12-68.

Figure 12-68. RxMode and StopState FSM



12.4.4 Timing Control

12.4.4.1 Timing-Control Features

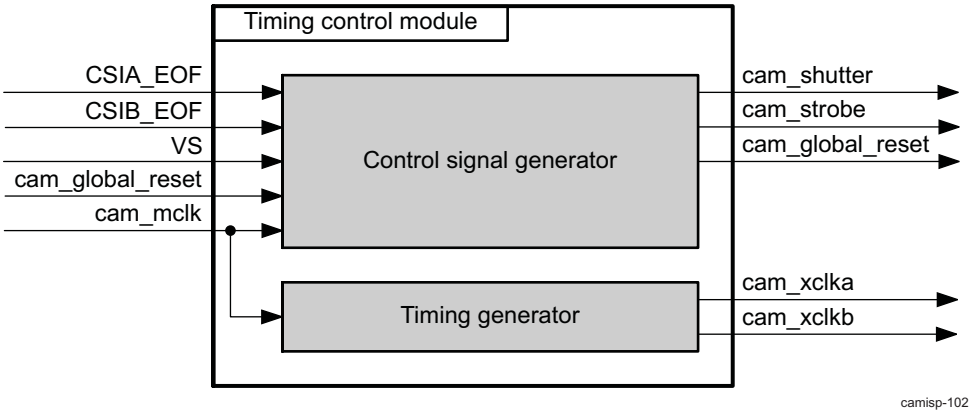
The timing-control module provides two clocks (cam_xclka and cam_xclkb) that can be used by external camera modules. It also generates the control signals (cam_strobe and cam_shutter) for the flash prestrobe, flash strobe, and mechanical shutters.

The timing-control module includes a timing generator and a control-signal generator.

12.4.4.2 Timing Control

Figure 12-69 shows a block diagram of the timing-control module.

Figure 12-69. Timing-Control Module



12.4.4.2.1 Timing Generator

The timing generator generates the cam_xclka and cam_xclkb clocks based on the cam_mclk, which is equal to 216 MHz. The CAM_MCLK frequency can be less than, or equal to 216 MHz. The cam_mclk is used only by the clock generator; the cam_xclka and cam_xclkb clocks are not used internally by the Camera ISP. The clock divider is programmable.

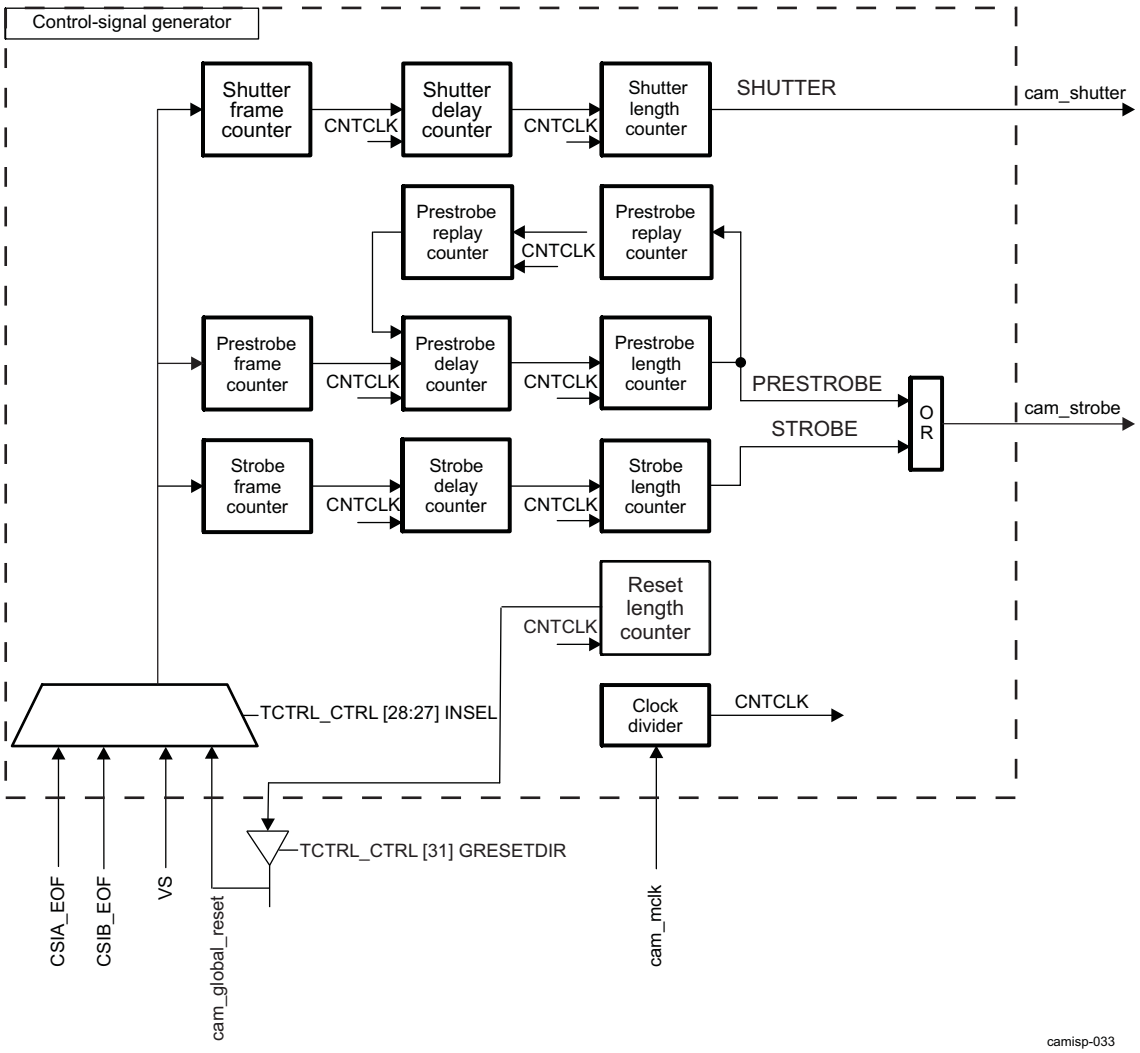
The possible frequencies of cam_xclka and cam_xclkb and their respective configurations are described in Section 12.3.1.1.3, *Clock Configuration*.

Table 12-26 summarizes the possible frequencies as a function of the divisor values.

12.4.4.2.2 Control-Signal Generator

The control-signal generator generates the prestrobe, strobe, and shutter signals: cam_strobe and cam_shutter. Figure 12-70 shows the principle of control-signal generation.

Figure 12-70. Control-Signal Generation



The control-signal generator gathers precise timings for the cam_strobe and cam_shutter signals, to assert and deassert the signals at known times. The timing-control-signal generator can be synchronized either on the vertical synchronization signal coming from the CSIa (VP_VS from CSIa), CSIb (VP_VS from CSIb), or PARALLEL interface (cam_vs), or on an externally-generated cam_global_reset signal.

A multiplexer controls which of the CSIa, CSIb, or PARALLEL interface drives control-signal generation. This multiplexer can also select the externally-generated cam_global_reset signal as the trigger event. The TCTRL_CTRL [31] GRESETDIR register defines the direction of the cam_global_reset signal.

- The external generated cam_global_reset is used as a trigger when TCTRL_CTRL [31] GRESETDIR = 0 and TCTRL_CTRL[27:28] INSEL = 3.
- The internally generated cam_global_reset is used as a trigger when TCTRL_CTRL [31] GRESETDIR = 1 and TCTRL_CTRL[27:28] INSEL = 3.
- If the PARALLEL interface is selected, control-signal generation works for both ITU and SYNC modes and on both output ports: video port and shared-buffer-logic (SBL) port.
- If the CSIa or CSIb interface is selected, control-signal generation works on both output ports: video port and interconnect port.

The cam_global_reset signal can also be generated internally by the control-signal generator under software control. In this case, the prestrobe and shutter signals are synchronized on the internally generated cam_global_reset signal. The multiplexer controls whether control-signal generation must be triggered by the internal or external cam_global_reset signal.

The prestrobe-, strobe-, and shutter-control signals can be individually enabled at any time. These signals must not be disabled by software.

The clock divider generates the CNTCLK clock based on the cam_mclk clock. The clock divider is programmable. Table 12-26 summarizes the possible frequencies as a function of the divisor values.

Table 12-26. Control-Signal Generator: CNTCLK Frequencies

Divisor Value TCTRL_CTRL [18:10] DIVC	CNTCLK Clock
0 (default)	Clock gated. No clock.
1	216 MHz, free-running.
2	108 MHz
3	72 MHz
4	54 MHz
...	...
510	0.424 MHz
511	0.423 MHz

There are three counters per control signal, for a total of nine counters. Each counter is programmable.

- The frame counter is decreased each time a full new frame is received, based on the EOF events from the CCDC or CSI receiver modules.
  - A new frame is detected in the CSI RECEIVER modules on detection of a frame-start code (FSC) followed by a frame-end code (FEC).
  - A new frame is detected in the CCDC module by using the falling edge of the vertical synchronization signal at the input of the CCDC module.

    Note: The rising edge of the vertical synchronization signal and the vertical synchronization polarity settings inside the CCDC cannot be used. The modules have no effect on this detection.

  - The frame counter determines how many whole frames must be ignored before the delay counter is triggered. The frame counters can be set to 0 to bypass them.
- The delay counter determines the control-signal activation delay. The counter is decreased at every CNTCLK clock cycle. When the counter reaches 0, the control signal is asserted. If the delay counter is set to 0, the control signal is asserted immediately.
- The activation-length counter determines the control-signal assertion length. The counter is decreased at every CNTCLK clock cycle. When the counter reaches 0, the signal is deasserted and the control-signal enable bit is disabled. If the activation length is set to 0, the control signal is not asserted and the control-signal enable bit is disabled.

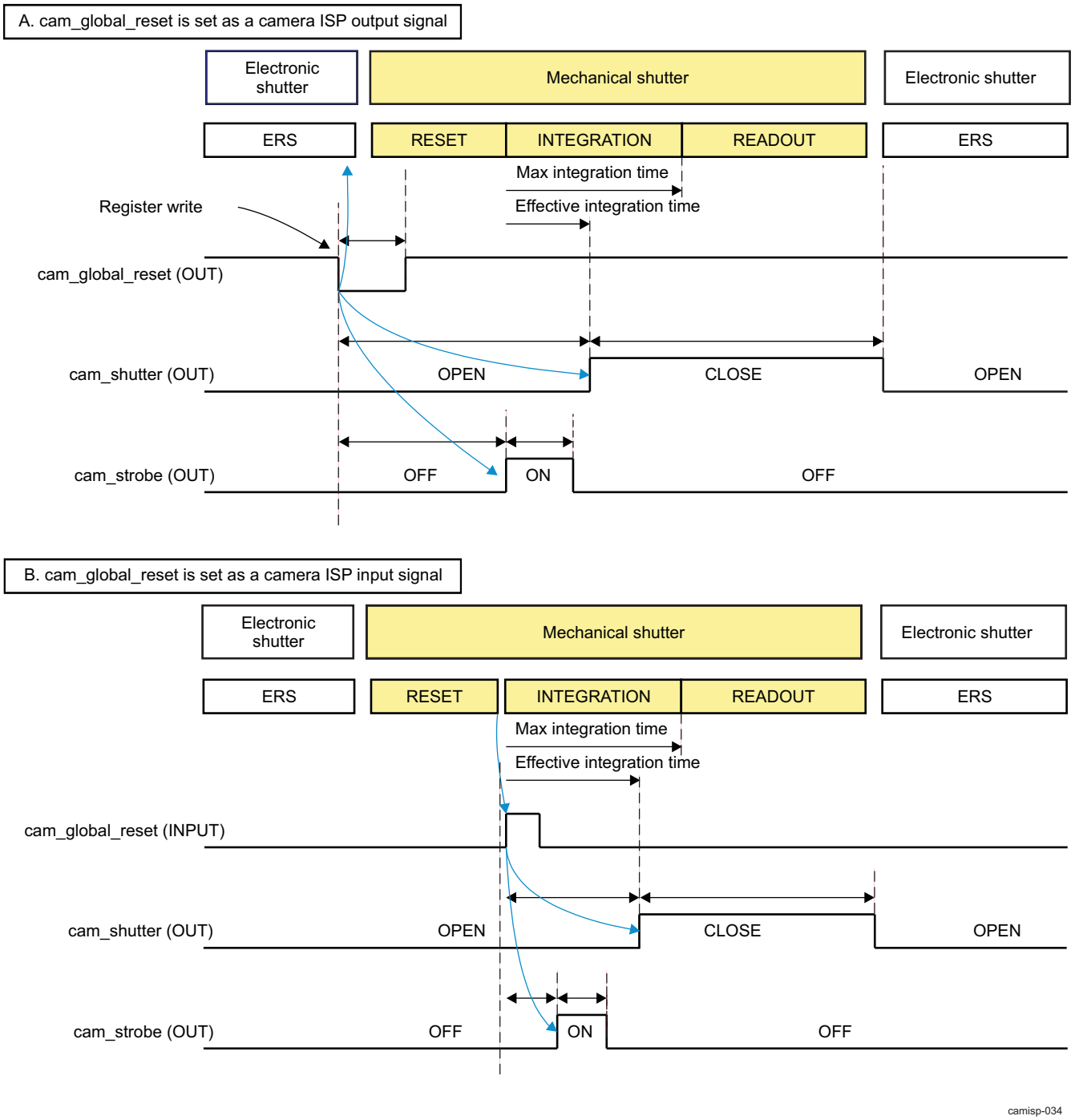
The polarity of the following signals can be individually selected:

- TCTRL_CTRL [26] STRBPSTRBPOL for the prestrobe and strobe signals
- TCTRL_CTRL [24] SHUTPOL for the shutter signal
- TCTRL_CTRL [30] GRESETPOL for the cam_global_reset signal

The software can trigger the generation of the cam_global_reset signal to the camera module. The signal-activation length is programmable. The counter is decreased at every CNTCLK clock cycle. When the counter reaches 0, the signal is deasserted and the global reset enable bit is disabled (TCTRL_CTRL [29] GRESETEN bit). If the activation length is set to 0, the control signal is not asserted and the control-signal enable bit is disabled. The polarity of the cam_global_reset signal can be selected (TCTRL_CTRL [30] GRESETPOL bit).

Figure 12-71 shows the use of the cam_global_reset signal set as an input or output signal. cam_global_reset is asynchronous, edge-sensitive, and asserted for at least one interconnect clock cycle

Figure 12-71. Use of cam_global_reset With Global Reset Release Camera Modules



There are two types of shutter mechanisms: mechanical and electronic. A mechanical shutter is used only for high-resolution sensors. The three control signals (cam_global_reset, cam_shutter, and cam_strobe) are useful with a mechanical shutter. High frame rates can be achieved only with an electronic shutter. When an electronic shutter is used, none of the three control signals is used.

Mechanical shutter mechanism:

- Reset: All pixels of the sensor are reset to their black value. When the sensor has a global reset feature, the mechanical shutter can be open during reset.

- Integration: The light received by the sensor is transformed into electrical charges that are stored inside pixels. At the end of the integration time, the shutter must be closed. Exposure time is defined by the time between reset release and shutter close.
- Readout: The charges accumulated in pixels are converted to digital values that are sent to the camera receiver.

Electronic rolling shutter mechanism (ERS):

- Each line of the sensor is reset separately and read after a fixed amount of time. Expose time is defined by the time between reset and read.

12.4.5 Bridge-Lane Shifter

The bridge-lane shifter module contains a data-lane shifter and an optional bridge:

- The data-lane shifter routes data sent to physical pins to the proper inputs of the CCDC module. It is controlled by the ISP_CTRL [7:6] SHIFT register. Table 12-27 describes the different configurations.

Table 12-27. Data-Lane Shifter

Sensor	Connected to	Data Lane Shifter 0	Data Lane Shifter 1	Data Lane Shifter 2	Data Lane Shifter 3	Note
8 bits	[7:0]	8 bits	6 bits	4 bits	2 bits	
	[9:2]	10 bits	8 bits	6 bits	4 bits	
	[11:4]	12 bits	10 bits	8 bits	6 bits	
	[13:6]	14 bits	12 bits	10 bits	8 bits	CSI2 only
10 bits	[9:0]	10 bits	8 bits	6 bits	4 bits	
	[11:2]	12 bits	10 bits	8 bits	6 bits	
	[13:4]	14 bits	12 bits	10 bits	8 bits	CSI2 only
12 bits	[11:0]	12 bits	10 bits	8 bits	6 bits	
	[13:2]	14 bits	12 bits	10 bits	8 bits	CSI2 only
14 bits	[13:0]	14 bits	12 bits	10 bits	8 bits	CSI2 only

Blue shading means that the data is too wide for the image pipeline: use IVA2.2.

Green shading means that precision is increased for intermediate results.

Orange shading means that precision is reduced.

Unshaded cells imply normal precision.

- An optional bridge allows the packing of bytes into 16-bit words. When it is used, the maximum data rate allowed is increased. The placement of 8-bit data inside 16-bit words is configurable through the ISP_CTRL [3:2] PAR_BRIDGE register. This mode can be useful to transfer an YCbCr data stream or compressed stream to memory at very high speed:
  - A minimum line-blanking period of 2 pixels must be obeyed when the bridge is enabled.
  - Some CCDC modules cannot work with the bridge. For details, see Table 12-28.

12.4.6 Video-Processing Front End

The video-processing front end (VPFE) comprises the CCDC and the lens-shading compensation.

12.4.6.1 CCDC

12.4.6.1.1 CCDC Features

The CCDC is responsible for accepting RAW (unprocessed) image/video data from a sensor. It can also accept YUV video data in numerous formats. For RAW inputs, the CCDC output requires additional image processing to transform the input image to a final processed image. This processing can be performed either on-the-fly in the preview engine, or in software on the IVA2.2 subsystem. In parallel, RAW data input to the CCDC can be used for computing statistics (H3A, histogram) to eventually control image/video tuning parameters. The CCDC module supports the following features:

- **Image sensors:** Supports most image sensors (resolutions up to 4096 x 4096).
- **CCDC interface:** The Camera ISP module interface comes either from the external parallel interface or from the output of the CSI receiver (through the video-port interface). It is a 16-bit interface. To raise this maximum 16-bit interface, the bridge data-lane shifter before the CCDC module allows the packing of 8-bit data into 16 bits (not supported with ITU mode):
  - Data up to 8-bit at 130 MHz can be transferred to memory.
  - Data up to 10-bit at 75 MHz can be processed by the image pipeline or transferred to memory.
  - Data up to 12-bit at 75 MHz can be transferred to memory or internally converted into 10-bit data to be processed by the image pipeline.
  - Supports two synchronization modes:
    - **SYNC mode:** In this mode, the cam_hs and cam_vs signals use dedicated wires. Synchronization signals are provided by either the sensor or the Camera ISP. This mode works with 8-, 10-, 11-, and 12-bit data. It supports both progressive and interlaced image-sensor modules.
    - **ITU mode:** In this mode, the image-sensor module provides an ITU-R BT 656-compatible data stream. Horizontal and vertical synchronization signals are not provided to the interface. Instead, the data stream embeds SAV and EAV synchronization code. This mode works in 8- and 10-bit configurations.
- **RAW data processing:** Output data can go directly to memory for software processing, or to the PREVIEW module for further processing. The operations include:
  - Optical clamp
  - Black-level compensation
  - Faulty-pixel correction
  - 2D map based lens-shading compensation
  - Data formatter
  - Output formatter
- **YUV data processing:** The output data can go directly to memory for software processing or to the resizer module for further processing. The operations include:
  - DC subtract
  - Culling
  - Output formatter
- Memory ports: The CCDC module can access memory as a master through the CRSBL, so it does not require the assistance of the system DMA.

12.4.6.1.2 CCDC Block Diagram

Figure 12-72 shows the top-level block diagram of the CCDC module.

Figure 12-72. CCDC Block Diagram

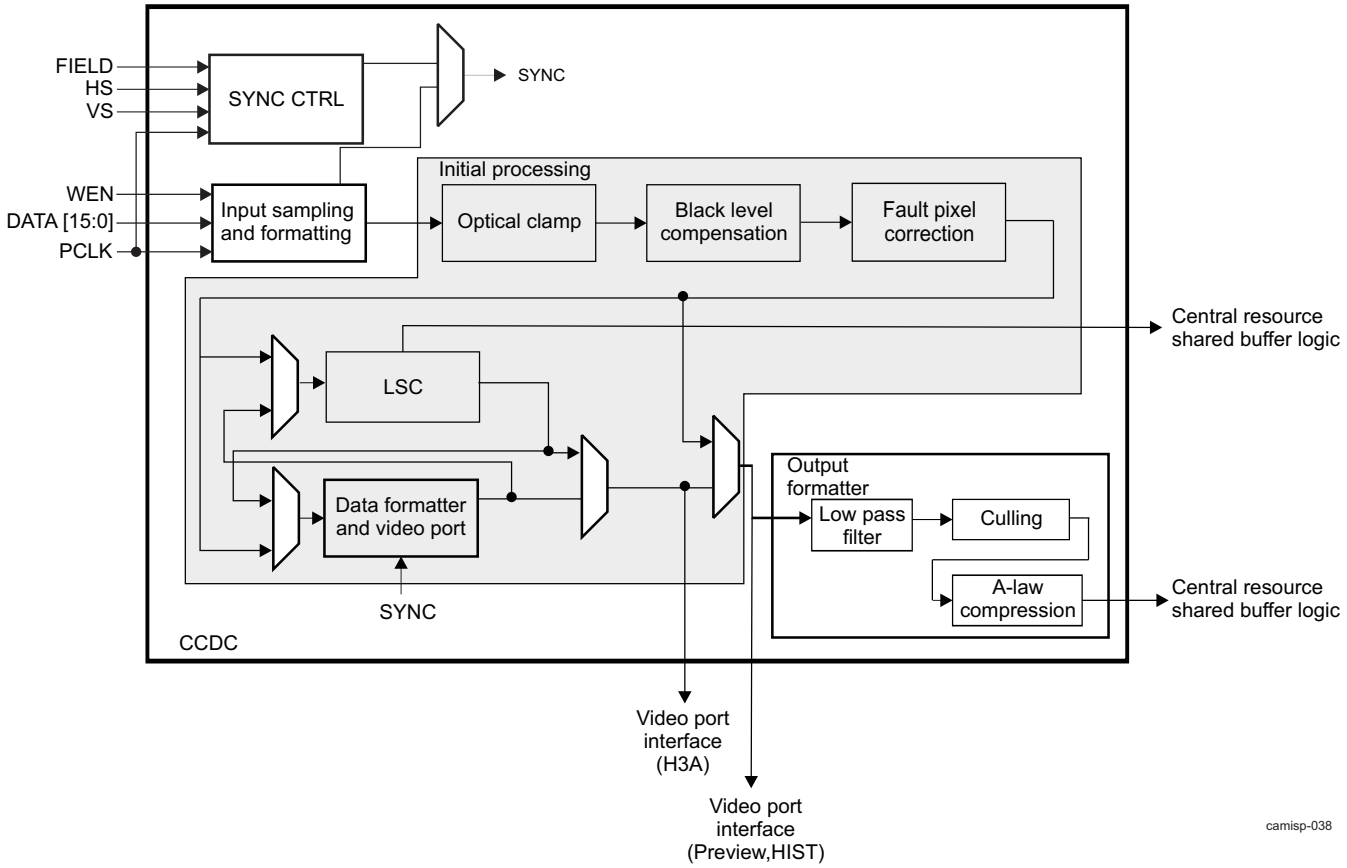


Table 12-28 summarizes allowed data flows through the CCDC.

Table 12-28. Allowed Data Flows Through the CCDC

Module		6/7 Bits RAW Format, SYNC Mode	8/10 Bits RAW Format, SYNC Mode	11/12 Bits RAW Format, SYNC Mode	8/10 Bits YUV4:2:2 Format, BT656 Mode	8/10 Bits YUV4:2:2 Format, SYNC Mode <75 MHz	8/10 Bits YUV4:2:2 Format, SYNC Mode <130 MHz	Other Formats (JPEG...), SYNC Mode
Bridge data-lane shifter	Bridge						+	
CCDC-SYNC CTRL	SYNC CTRL	+	+	+		+	+	+
CCDC-Input sampling and formatting	BT656 decoder				+			
CCDC-Initial processing	DC subtract	+	+	+	+	+	+	
	Optical black clamp	+	+	+				
	Black-level comp	+	+	+				

Table 12-28. Allowed Data Flows Through the CCDC (continued)

Module		6/7 Bits RAW Format, SYNC Mode	8/10 Bits RAW Format, SYNC Mode	11/12 Bits RAW Format, SYNC Mode	8/10 Bits YUV4:2:2 Format, BT656 Mode	8/10 Bits YUV4:2:2 Format, SYNC Mode <75 MHz	8/10 Bits YUV4:2:2 Format, SYNC Mode <130 MHz	Other Formats (JPEG...), SYNC Mode
	Faulty-pixel correction	+	+	+				
CCDC-Data formatter	Data formatter	+	+					
	Shading compensator		+					
	Preview, H3A, HIST data path	+	+					
CCDC-Output formatter	Pixel selection	+	+	+	+	+	+	
	Low-pass filter	+	+	+				
	Culling	+	+	+	+	+	+	
	A-Law	+	+	+				
	Resizer data path				+	+	+	
	Memory data path	+	+	+	+	+	+	+

Note: Data from CSIa or CSIb receivers corresponds to SYNC columns.

The data flow through the module differs, depending on whether the input is RAW data or YUV data. It also depends on the application scenario.

For RAW8/10 data (CCDC_SYN_MODE [13:12] INPMODE = 0 && CCDC_REC656IF [0] REC656ON = 0), the following functions apply:

- Video-preview and video-capture data flows can pass through the optical-clamp, black-level compensation, faulty-pixel correction, lens-shading compensator, and data-reformatter submodules. Output is transmitted to the HISTH3Amodules for further processing.
- JPEG still-image-capture data flow passes through the optical clamp, black-level compensation, faulty-pixel correction, lens-shading compensator, and data-reformatter submodules. Then the data flow is sent to memory through the SBL, Circular buffer, and MMU to be read by the external JPEG CODEC.
- RAW still-image-capture data flow typically passes through the optical clamp, black-level compensation, faulty-pixel correction, lens-shading compensator and output-formatter submodules.

For YUV data (CCDC_SYN_MODE [13:12] INPMODE = 1 or 2 && CCDC_REC656IF [0] REC656ON = 1), the following functions apply:

- Data can be written to memory directly through the central-resource SBL module or sent to the resizer module for upscaling or downscaling.

12.4.6.1.3 CCDC Functional Operations

12.4.6.1.3.1 SYNC CTRL Module

The SYNC CTRL module receives the pixel-clock signal from the image sensor (PCLK). The module can be slave or master of the horizontal and vertical synchronization signals (HS and VS) and of the field-identification signal (FIELD).

The HS, VS, and FLD signals can be set as inputs or outputs. The polarity of the HS, VS, and FLD signals can be set as positive or negative. If the HS, VS, and FLD signals are output, the signal length can be set.

12.4.6.1.3.2 Input Sampling and Formatting

- Data is latched by the pixel clock.
- Pixel-clock polarity can be either rising- or falling-edge. This is set through ISP_CTRL [4] PAR_CLK_POL.
- Data can be interpreted as normal or inverted (CCDC_SYN_MODE[6] DATAPOL).
- For RAW data:
  - Data is clipped to the number of LSBs specified in the CCDC_SYN_MODE [10:8] DATSIZ field. This also sets the maximum data size allowed in subsequent clipping/limiting operations and is the output data alignment if data is written to memory.
- For YUV data (BT656 or SYNC mode):
  - The MSB of the chroma signal can also be inverted (CCDC_CFG [13] MSBINVI). It adds 128 to chrominance signals, to be compatible with several sensors.
- BT656 decoder: Separates data and synchronization signals. This module outputs HS, VS, and FIELD signals.

12.4.6.1.3.3 CCDC Initial Processing

Optical Clamp

The optical black clamping function provides a means of averaging the optically black pixels and subtracting that value from each input pixel as a first step. The goal is to remove an offset caused by the sensor technology.

The averaging circuit takes an average of masked (black) pixel values from the image sensor, averaging pixels at the start (CCDC_CLAMP [24:10] OBST) of each line (CCDC_CLAMP [30:28] OBSLEN) and for the number of indicated lines (CCDC_CLAMP [27:25] OBSLN), plus an optional gain adjustment (CCDC_CLAMP [4:0] OBGAIN), and subtracting this value from the image data at the succeeding line. Users can control the position of the black pixels, the number of pixels (1, 2, 4, 8, or 16) in each line averaged, and the averaged number of lines (1, 2, 4, 8, or 16).

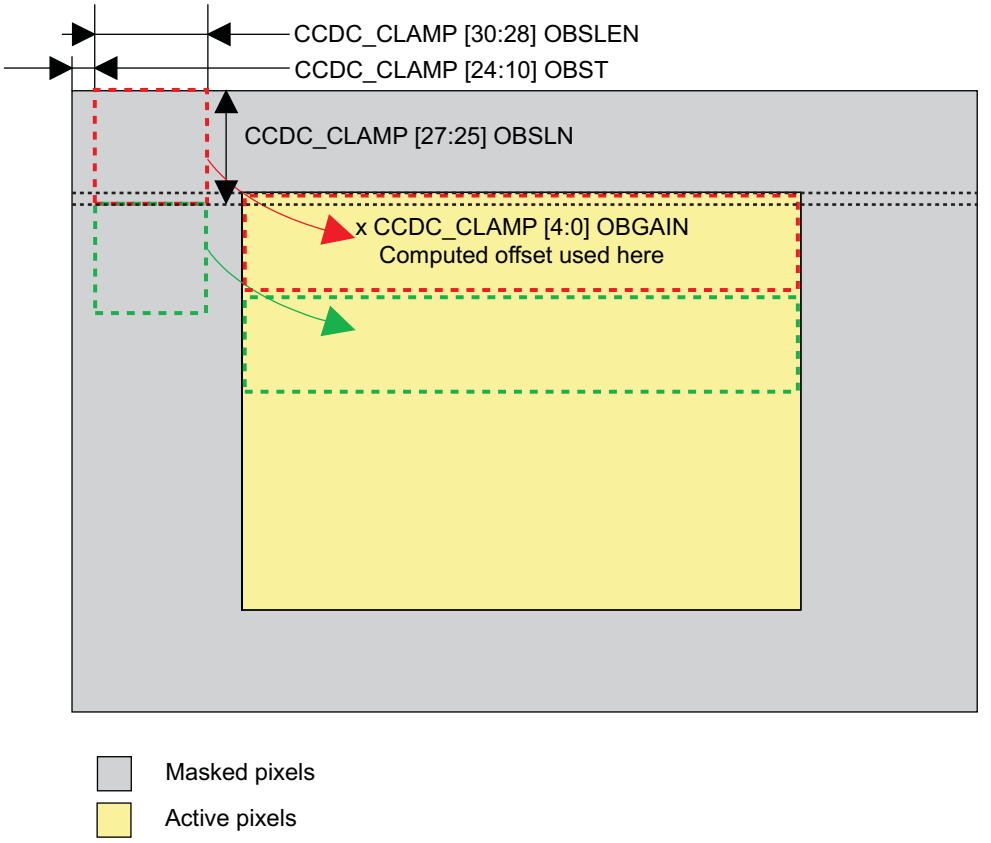
Alternately, users can disable black clamp averaging (CCDC_CLAMP [31] CLAMPEN) and select a constant black value for subtraction (CCDC_DCSUB [13:0] DCSUB), instead of using the calculated average value.

Note: For YUV data, this operation subtracts a fixed value (CCDC_DCSUB [13:0] DCSUB) from the luminance sample. To disable this operation, set the subtraction value to zero.

This function does not clip negative results to 0 for YUV 8 bit input or REC656 input modes (CCDC_SYN_MODE [13:12] INPMOD == 2 || CCDC_REC656IF [0] REC656ON == 1).

Figure 12-73 shows an optical clamp representation.

Figure 12-73. Optical Clamp Representation



Black-Level Compensation

After the optical clamp, black-level compensation is applied to the data. In this operation, a fixed value, depending on the color (R/Ye, Gr/Cy, Gb/G, and B/Mg), can be subtracted from the data. The offset (CCDC_BLKCMP register, fields R_YE, GR_CY, GB_G, B_MG) applied to each data sample is selected according to the pixel position (0/1/2/3) and the color (0/1/2/3) specified for each pixel position (CCDC_COLPTN). The color pattern definition is flexible to accommodate different sensor types (such as Bayer CFA sampling, VGA Movie Mode).

Faulty-Pixel Correction

Faulty-pixel correction in the CCDC module requires the camera driver to have information about the image-sensor faulty-pixel number and positions. This method leads to the best image quality. However, if the position of the faulty pixels is unavailable to the camera driver, it can apply another faulty-pixel correction algorithm in the preview module. This algorithm leads to lower-quality images.

The CCDC module implements an optional (CCDC_FPC [15] FPCEN) faulty-pixel correction operation using a look-up table stored in external memory, which contains information about the horizontal and vertical positions of the pixels to be corrected, as well as the type of operation to be performed on the pixels. The CCDC_FPC_ADDR register specifies the starting address in memory for the faulty-pixel correction table.

Note: The memory address must be 64-byte-aligned (6 LSB are ignored).

Note: For YUV data, the faulty-pixel correction operation is not applicable and must be disabled/bypassed (CCDC_FPC [15] FPCEN).

12.4.6.1.3.4 Data Formatter, Lens-Shading Compensation, and Video-Port Interface

Note: For YUV data, the data formatter, lens-shading compensation, and video-port interface must be bypassed (CCDC_FMTCFG [15] VPEN = 0x0 and CCDC_SYN_MODE [18] VP2SDR = 0x0).

The data-formatter module can be enabled to rearrange data after the faulty-pixel correction operation. It is intended to be used:

- When output data lines from the CCDC include pixels from multiple resolution lines. Internally, these sensors combine pixels from multiple horizontal lines into one output pseudoline.
- To smooth bandwidth. This helps reduce peak bandwidth when image cropping is used with the resizer.

The reformatter module performs the decomposition before the remainder of the normal CCDC processing stages.

The lens-shading compensation (LSC) module can be placed at two different locations:

- Before the data reformatter. It can only be used for Bayer sensors. The H3A video port gets the shading corrected image.
- After the data reformatter. This setup is required for non-Bayer sensors. Histogram and preview modules get the shading corrected image; however, H3A receives a non-corrected image.

Video-port/data-formatter output can also be saved to memory (instead of the RAW data). When CCDC_SYN_MODE [18] VP2SDR is set to 1, video-port data is sent to the output formatter. In addition, the CCDC_SYN_MODE [17] WEN bit must be enabled to store the output to memory.

The data-formatter and video-port interfaces are only 10 bits wide; therefore, the input data must be adjusted as it enters these modules. For flexibility, the bits to be retained can be selected by CCDC_FMTCFG [14:12] VPIN.

The reformatter decomposes each input line into multiple output lines with new, internally generated HS/VS signals. The reformatter sends it to memory or to other Camera ISP modules. These new HS/VS signals, rather than the original sensor HS/VS signals, then gate the downstream processing.

Conversion Area Select Parameters

When the data formatter is enabled, HS/VS signals are still generated as output (CCDC_SYN_MODE [16] VDHDEN = 0x1). The settings for these output signals are in the following fields:

- CCDC_HD_VD_WID[27:16] HDW
- CCDC_HD_VD_WID[11:0] VDW
- CCDC_PIX_LINES[31:16] PPLN
- CCDC_PIX_LINES[15:0] HLPRF

Note: These four registers are not used when HS/VS signals are input signals (CCDC_SYN_MODE [16] VDHDEN = 0x0).

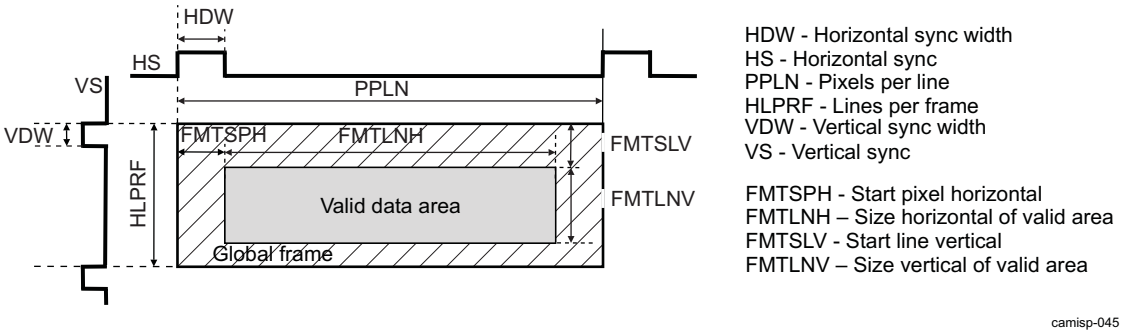
Note: The settings reflect those for the sensor readout frame, not the resultant reformatted frame.

Registers CCDC_FMT_HORZ and CCDC_FMT_VERT control the interpretation of the input data frame when the data formatter is enabled.

Registers CCDC_HORZ_INFO, CCDC_VERT_START, and CCDC_VERT_LINES control the interpretation of the input data frame in normal mode (when the data formatter is not enabled).

Figure 12-74 shows the data formatter conversion area selection.

Figure 12-74. Data Formatter Conversion Area Selection



Line Decomposition

When enabled, the reformatter can be configured to operate in line-alternating mode or program mode (CCD_FMTCFG [1] LNALT).

When line-alternating mode is enabled, the reformatter swaps even and odd lines. Basically, the 0^{th} input line is output as the first line, the first input line is output as the 0^{th} line, and so on. If this option is set, the start and number of lines for the formatter (CCDC_VERT_START and CCDC_VERT_LINES) must be even.

In program mode, or normal reformatter mode, the goal is to convert a single line of movie mode data to 1, 2, 3, or 4 lines of Bayer data. Each incoming line is decomposed according to the data-formatter settings and buffered into an internal line memory. The readout of the resultant multiple reformatted lines occurs as the next input line is being read from the sensor (and reformatted) to ensure that all output lines are fully constructed. The reformatter is subject to the restrictions listed in Table 12-29.

Table 12-29. Reformatter Output Limitations

Number of Output Lines/Input Line	Max Pixels/Output Line
1	4 * 1376
2	2 * 1376
3	1 * 1376
4	1 * 1376

The reformatter derives its flexibility from supporting up to 8 different addresses and a program that can contain up to 16 entries each for the odd and even lines. Each of the 8 addresses supports either autoincrement or autodecrement. This capability is the key for supporting a multitude of readout patterns. The following examples show the programmability of the reformatter. Decomposition is controlled by defining the following parameters:

- CCDC_FMTCFG [3:2] LNUM
  Number of lines into which each input line is to be decomposed
- CCDC_FMTCFG [11:8] PLEN_EVEN
  Number of program entries on even line
- CCDC_FMTCFG [7:4] PLEN_ODD
  Number of program entries on odd line
- CCDC_FMT_ADDR_i (i = 0:7)
  Output line in which the original pixel is to be placed, and initial address on the output line
- CCDC_PRGEVEN0 or CCDC_PRGEVEN1
  Program to be run on the even input lines
- CCDC_PRGODD0 or CCDC_PRGODD1
  Program to be run on the odd input lines

The CCDC_FMT_ADDR_i registers define 8 address pointers (ADDR0 to ADDR7) that define which output line the input pixel belongs to, and the starting address (position) on that output line. Users can use up to 8 address pointers. The address pointer can be incremented or decremented, depending on the input pixel pattern. The address value computed should always follow the reformatter rules as formerly stated and should never be negative.

Lens-Shading Compensation

- *Overview*

  The purpose of the LSC function is lens-shading correction by multiplying an image with a gain factor 2-D map, pixel by pixel. The image must be in Bayer CFA format having a 2x2 color pattern. The gain factor map is stored in external memory downsampled, and is accessed and upsampled by the LSC module before being applied to the pixel data.

  The LSC function is useful for lens-shading compensation and for scene- and image-dependent lighting adjustment. Downsampled gain map reduces memory requirements for storage and bandwidth requirements for access of the gain maps in external memory.

- *Features supported*
  - The memory stored gain map is MxN downsampled, M being the horizontal sampling factor, N being the vertical sampling factor, M and N being {4, 8, 16, 32, 64} independently and N M
  - 8-bit entries in the gain map (in U8Q8, U8Q7, U8Q6, and U8Q5 format with optional base of 1.0)
  - Up to 10-bit unsigned image data input/output
  - Up to 4096 x 3072 image dimension

- *Functional description*

  The lens-shading correction module multiplies an image by a gain map that is stored downsampled in memory. The gain map can be dependent on aperture, lens-shading characteristics, and other photographic settings. It is generally used to correct the dim corner effect, but can also be used to implement adaptive lighting adjustment.

  Separate horizontal and vertical sampling factors allow the lens-shading correction to work with the normal 1:1 aspect ratio image pixels, as well as tall-and-skinny pixels typical in the sensors preview mode image data.

12.4.6.1.3.5 Output Formatter

The output formatter starts with a framing selection to limit the processing area by setting CCDC_HORZ_INFO, CCDC_VERT_START, and CCDC_VERT_LINES registers. This framing selection is applied in addition to the framing applied at the beginning and at the end of the data formatter operation if the video-port path to memory is selected (CCDC_SYN_MODE [18] VP2SDR).

The option to send the CCDC output to the resizer module (CCDC_SYN_MODE [19] SDR2RSZ) should not be used when in RAW data mode, because the resizer operates only on YUV format data. Use the preview module when resizing is desired in RAW data mode.

Low-Pass Filter (LPF)

An optional horizontal low-pass antialiasing filter can be applied (CCDC_SYN_MODE [14] LPF) after reframing. The low-pass filter consists of a simple 3-tap (1/4, /12, and 1/4) filter. Two pixels on the left and two pixels on the right of each line are cropped if the filter is enabled. Use of the LPF is intended for bandwidth reduction if culling is enabled.

Note: For YUV data, the LPF must be disabled (CCDC_SYN_MODE [14] LPF = 0x0).

Culling

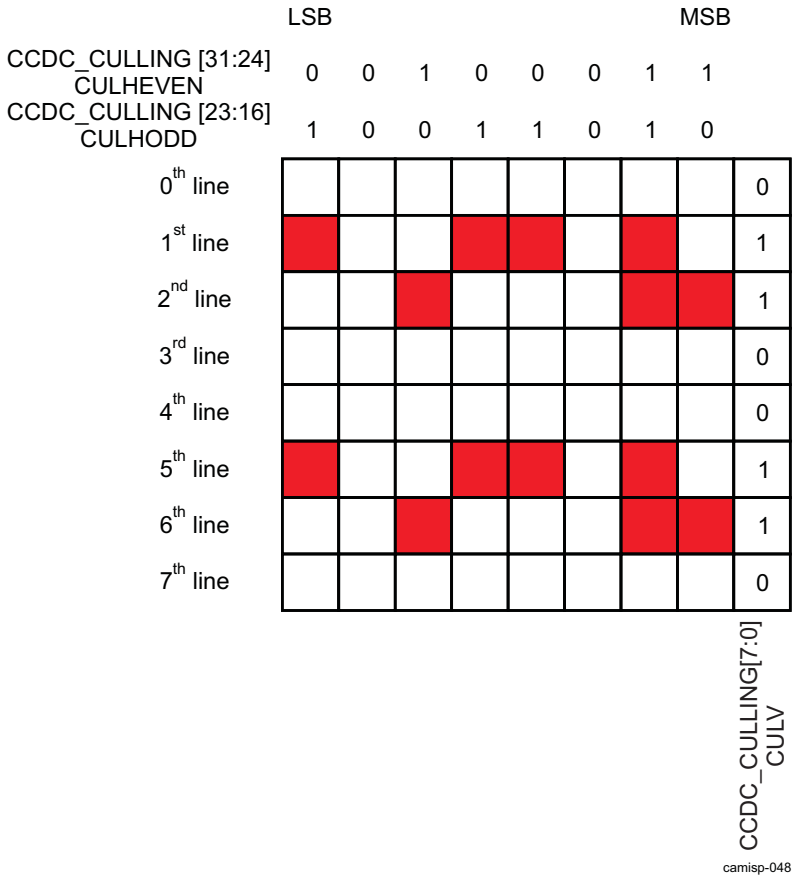
An optional culling operation can be enabled (CCDC_CULLING register). This operation allows selected pixel data to be culled (deleted) from a line (CCDC_CULLING [31:24] CULHEVN, CCDC_CULLING [23:16] CULHODD - 8-bit repeating mask, one per field) and selected lines to be culled from a frame (CCDC_CULLING [7:0] CULV).

Figure 12-75 is an example of how register values apply the decimation pattern to the data. The red pixels are saved to memory and the white pixels are discarded. In this example, CCDC_CULLING = 0x239A0066:

- CCDC_CULLING [31:24] CULHEVN = 0x23
- CCDC_CULLING [23:16] CULHODD = 0x9A
- CCDC_CULLING [7:0] CULV = 0x66

Note: Culling can be used with YUV data, but care must be taken to preserve the YUV422 output format.

Figure 12-75. CCDC/Culling: Example for Decimation Pattern



A-Law Compression

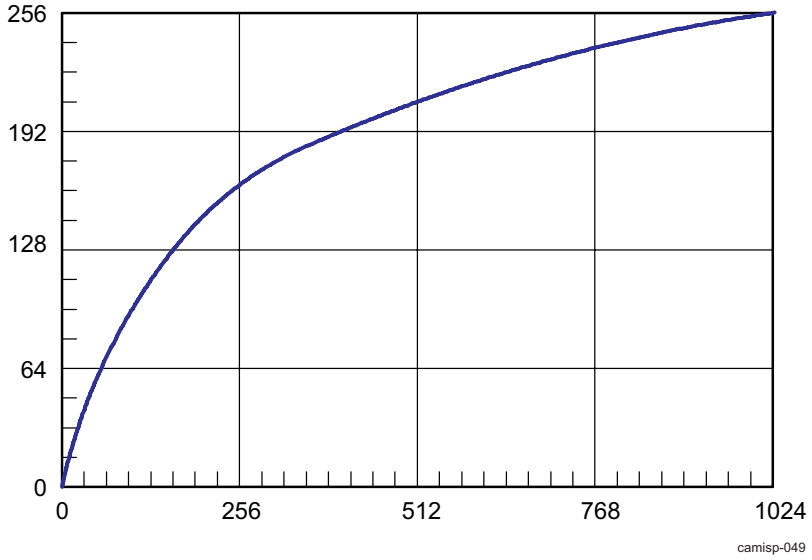
An optional 10-to-8-bit A-Law compression using a fixed A-Law table can be applied (CCDC_ALAW [3] CCDTBL) as the final processing stage. Using this causes data width to be reduced to 8 bits and allows packing to 8 bits/pixel when saving to memory. Because data resolution can be greater than 10 bits at this stage, the 10 bits for input to the A-Law operation must be selected (CCDC_ALAW [2:0] GWDI).

The preview module has an inverse A-Law table (A-Law decompression) option so that this nonlinear operation can be reversed if this saved data is to be read back in for further processing.

Note: A-Law compression should not be used (CCDC_ALAW [3] CCDTBL = 0) with YUV data.

Figure 12-76 show the A-Law table.

Figure 12-76. A-Law Table



Line-Output Control

Note: Line-output control can be used with YUV data.

The CCDC final stage is line-output control, which controls how the input sensor lines are written to memory. The value of CCDC_SDR_ADDR [31:0] ADDR defines the starting address where the frame should be written in memory. The value of CCDC_HSIZE_OFF [15:0] LNOFST defines the distance between two lines for each output line to memory. The starting address and line-offset values should be aligned to 32-byte boundaries; that is, either 16 or 32 pixels, depending on the CCDC_SYN_MODE [11] PACK8 setting. Register CCDC_SDOFST can be used to define additional offsets, depending on the field ID and even/odd line numbers. This provides a means to deinterlace an interlaced 2-field input and to invert an input image vertically. See Table 12-30.

Table 12-30. CCDC_SDOFST Description

Register	Description
CCDC_SDOFST [14] FIINV	Invert interpretation of the field ID signal
CCDC_SDOFST [13:12] FOFST	Offset, in lines, of field = 1
CCDC_SDOFST [11:9] LOFTS0	Offset, in lines, between even lines on even fields (field 0)
CCDC_SDOFST [8:6] LOFTS1	Offset, in lines, between odd lines on even fields (field 0)
CCDC_SDOFST [5:3] LOFTS2	Offset, in lines, between even lines on odd fields (field 1)
CCDC_SDOFST [2:0] LOFTS3	Offset, in lines, between odd lines on odd fields (field 1)

Line Output Control: 2D Addressing Example

The CCDC memory port can be configured to write a subimage into a bigger image buffer. This feature is useful for overlaying a video preview flow with the rest of the display. To use this feature, the CCDC_SDR_ADDR [31:0] ADDR register points to the first pixel to write and the CCDC_SDOFST [13:12] FOFST is set to the size of one line of the destination buffer.

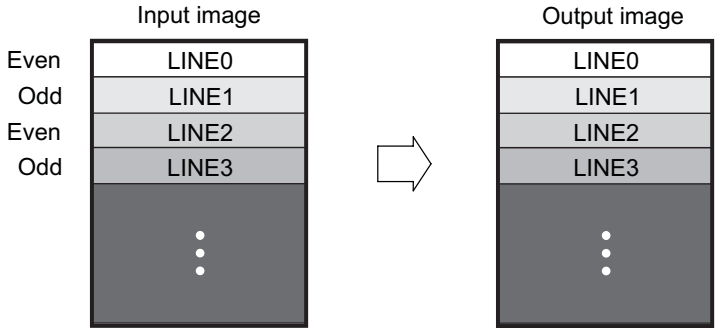
Note:

- This mode can be used only when the source and destination buffers use the same image format.
- 2D addressing is also supported by other Camera ISP modules.

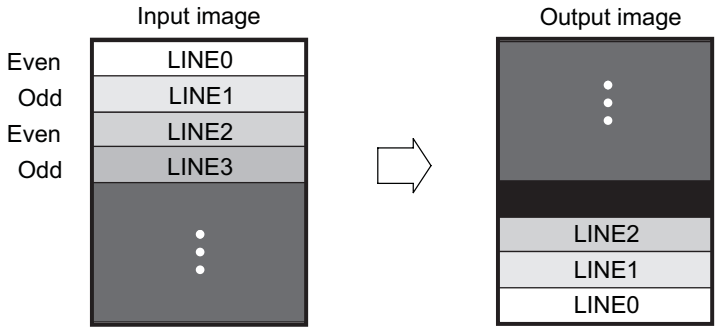
Line-Output Control: Examples

Figure 12-77 shows an example of sample formats of input and output images.

Figure 12-77. CCDC/Line-Output Control: Sample Formats of Input and Output Images




```
CCDC_SDOFST[14] FIINV = 0x0        ;Non inverse
CCDC_SDOFST[13:12] FOFST = 0x0     ;+1 line; first line, even field
CCDC_SDOFST[11:9] LOFST0 = 0x1     ;+2 lines; even lines, even fields
CCDC_SDOFST[8:6] LOFST1 = 0x1      ;+2 lines; odd lines, even fields
CCDC_SDOFST[5:3] LOFST2 = 0x1      ;+2 lines; even lines, odd fields
CCDC_SDOFST[2:0] LOFST3 = 0x1      ;+2 lines; odd lines, odd fields
```




```
CCDC_SDOFST[14] FIINV = 0x1        ;inverse
CCDC_SDOFST[13:12] FOFST = 0x0     ;+1 line; first line, even field
CCDC_SDOFST[11:9] LOFST0 = 0x5     ;–2 lines; even lines, even fields
CCDC_SDOFST[8:6] LOFST1 = 0x5      ;–2 lines; odd lines, even fields
CCDC_SDOFST[5:3] LOFST2 = 0x5      ;–2 lines; even lines, odd fields
CCDC_SDOFST[2:0] LOFST3 = 0x5      ;–2 lines; odd lines, odd fields
```

camisp-051

Output Format

The data bits comprising each pixel are stored in the lower bits of a 16-bit memory word and the unused bits are zero-filled. The memory data format is shown in Table 12-31. The format is determined by the CCDC_SYN_MODE [10:8] DATSIZ field.

If 8-bit data is input, or if A-Law compression is applied, the data can be packed through the CCDC_SYN_MODE [11] PACK8 setting so that a pixel occupies only 8 bits.

Data is output to memory only when enabled through the CCDC_SYN_MODE [17] WEN setting. The pixels are ordered in little-endian format.

Table 12-31. Memory Output Format for RAW Data

	Upper word		Lower word	
	MSB(31)	LSB(16)	MSB(15)	LSB(0)
16 bit	Pixel 1		Pixel 0	
15 bit	0	Pixel 1	0	Pixel 0
14 bit	0	Pixel 1	0	Pixel 0
13 bit	0	Pixel 1	0	Pixel 0
12 bit	0	Pixel 1	0	Pixel 0
11 bit	0	Pixel 1	0	Pixel 0
10 bit	0	Pixel 1	0	Pixel 0
9 bit	0	Pixel 1	0	Pixel 0
8 bit	0	Pixel 1	0	Pixel 0
8-bit packed	Pixel 3	Pixel 2	Pixel 1	Pixel 0

Note: YUV data is stored in memory in packed YUV422 mode, using two pixels per 32 bits, as shown in Table 12-32.

Table 12-32. Memory Output Format for YUV Data

Memory Address	Upper Word		Lower Word	
	MSB(31)	LSB(16)	MSB(15)	LSB(0)
N	Y1	Cr0	Y0	Cb0
N + 1	Y3	Cr1	Y2	Cb1
N + 2	Y5	Cr2	Y4	Cb2

12.4.6.1.4 DMA

The CCDC module is a master. It has its own address generator and sends addresses and data to the central-resource shared-buffer logic. The central-resource shared-buffer logic arbitrates the data requests and generates the bursts to memory.

12.4.6.1.5 Memories

The CCDC module has one memory:

- REFORMATTER BUFFER: 3 1376 40 bits
- LSC PREFETCH BUFFER: 1536 32 bits

12.4.7 Video-Processing Back End

The video-processing back end (VPBE) comprises the preview and resizer modules.

12.4.7.1 Preview Engine Features

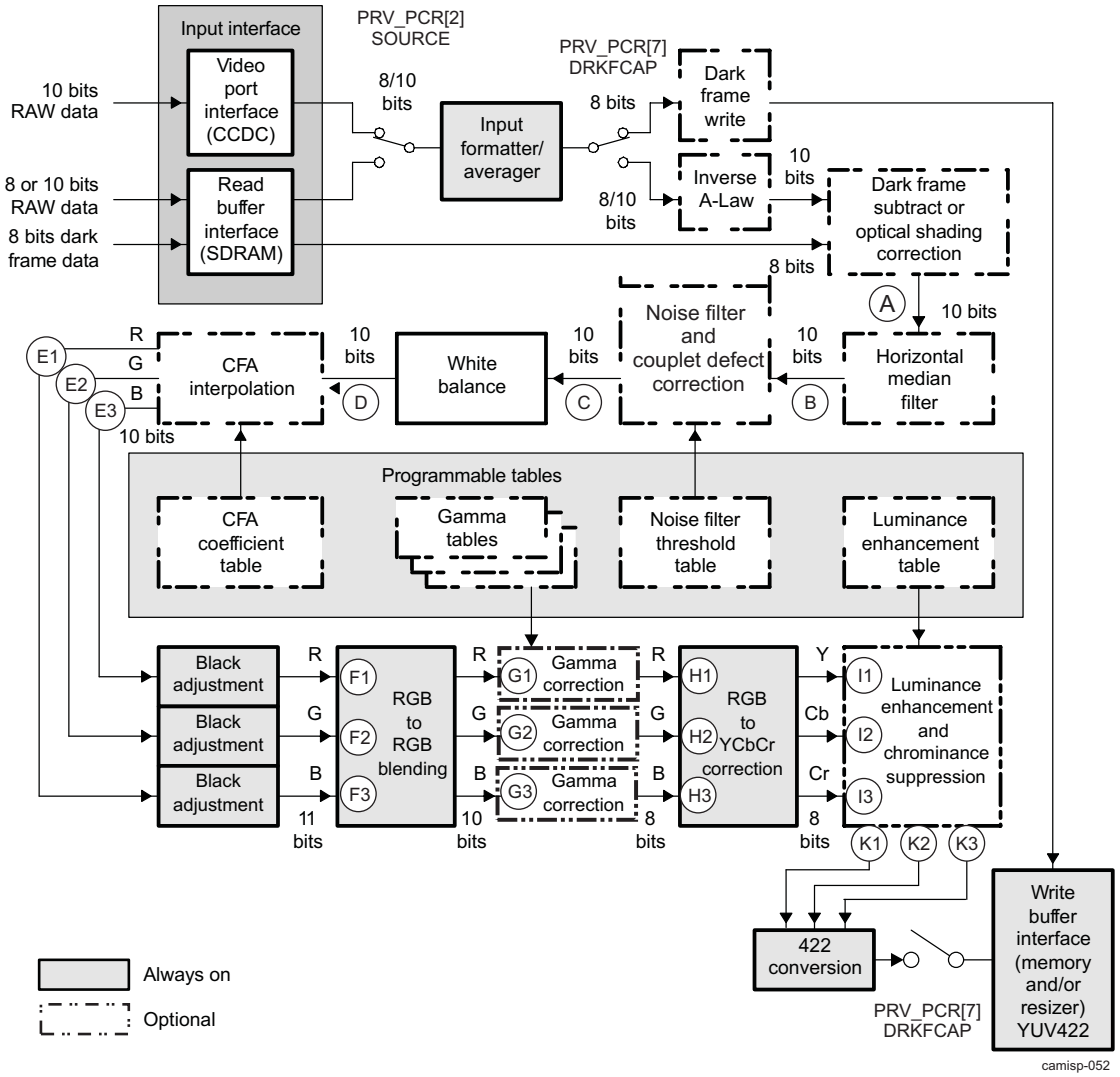
The preview module is responsible for transforming data from a RAW image sensor into YUV422 data that is amenable to still-image encoding, video encoding, or display. It supports the following features:

- Flexible input: Module input data can come from the RAW image sensor (10 bits) or from memory (8 or 10 bits).
- Flexible input formats: Bayer RGB color filter array, complementary-color filter array, Super CCD Honeycom _**UnicodeEncodeError**_ char was 174 sensors
- Horizontal averaging by a factor of 2, 4, or 8 in the horizontal direction. The preview module can output a maximum of only 3312 pixels horizontally due to fixed memory-line sizes.
- A-Law decompression: Transforms nonlinear 8-bit data to 10-bit linear data. The CCDC module can perform A-Law compression.
- Noise reduction and faulty-pixel correction:
  - Dark-frame capture and subtraction
  - Horizontal median filter
  - Programmable noise filter: 33 kernel of same color pixels
  - Couplet faulty-pixel correction
- Digital gain and white balance
- Programmable CFA interpolation that operates on a 55 grid
- Programmable RGB-to-RGB blending matrix: 9 coefficients for the 33 matrix
- Programmable gamma correction: 1024 entries for each color held in local memory
- Programmable RGB-to-YUV color conversion: 9 coefficients for the 33 matrix
- Luminance enhancement (nonlinear), chrominance suppression and offset

12.4.7.1.1 Preview Block Diagram

Figure 12-78 shows the preview engine block diagram.

Figure 12-78. Preview Engine Block Diagram



12.4.7.1.2 Input Interface

The preview engine receives RAW image/video data from the video port interface through the CCDC or from the read buffer interface through the memory (PRV_PCR [2] SOURCE). When the source of input data is the CCDC, the input data is always 10 bits wide. When the source of input data is memory (read buffer interface), the data can be 8 or 10 bits wide. Use the PRV_PCR[4] WIDTH field to set the input data width. The 8 bits can be linear or nonlinear (A-Law compressed).

In addition, the preview engine can fetch a dark frame from memory, with each pixel 8 bits wide.

The frame-input size is configured using the PRV_HORZ_INFO and PRV_VERT_INFO registers.

If the input source is the CCDC, the input height set in the preview engine must be less than or equal to the output height of the video-port output of the CCDC. The input width must be at least 4 pixels smaller than the CCDC output width (SPH >= 2; EPH >= 2 pixels before the last pixel sent from the CCDC).

The input memory address (PRV_RSDR_ADDR) and line offset (PRV_RADR_OFFSET) registers should be aligned on 32-byte boundaries when the input source is set to memory. The dark-frame input address (PRV_DSDR_ADDR) and line offset (PRV_DRKF_OFFSET) should also be aligned on 32-byte boundaries when the dark-frame subtract function is enabled.

When the input source is memory, the preview engine always operates in one-shot mode. After enabling the preview engine and processing a frame, the enable bit is turned off and it is up to firmware to reenable it to process the next frame from memory.

When the input source is the CCDC, the preview engine can be configured to operate in one-shot mode or continuous mode (PRV_PCR [3] ONESHOT).

The SBL image data read port is shared between the preview and CSI1 receiver module. When the image is read from memory, the read port must be affected to the preview receiver module by writing 0 into the ISP_CTRL[27] SBL_SHARED_RPORTA. Programmers must ensure that the CSI1 module does not use this port before switching to the preview module.

12.4.7.1.3 Input Formatter/Averager

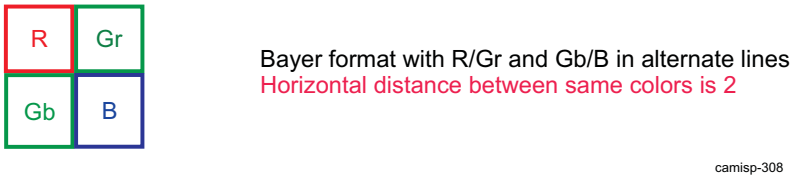
Preview-engine output is limited to 3312 pixels per horizontal line due to line memory-width restrictions in the noise filter and CFA interpolation blocks. To support sensors that output more than 3312 pixels per line, an averager is incorporated to downsample by factors of 1 (no averaging), 2, 4, or 8 in the horizontal direction (PRV_AVE [1:0] COUNT). The horizontal distance between two consecutive pixels of the same color to be averaged is selectable from among 1, 2, 3, or 4 for both even (PRV_AVE [3:2] EVENDIST) and odd (PRV_AVE [5:4] ODDDIST) lines. This must be configured to match the input pattern type.

For example, a Bayer pattern has a horizontal distance of two pixels of the same color for both even and odd lines.

Valid output of the input formatter/averager is either 8 or 10 bits wide.

Figure 12-79 shows the horizontal distances for different patterns.

Figure 12-79. Horizontal Distances for Different Patterns



12.4.7.1.4 Dark-Frame Write

The preview engine is capable of capturing and saving a dark frame to memory instead of performing conventional processing steps (PRV_PCR [7] DRKFCAP).

This dark frame can later be subtracted from the RAW image data to eliminate the repeatable baseline noise level in the frame.

Each input pixel is written as an 8-bit value; if the input pixel value is greater than 255, it is saturated to 255. If a dark pixel is greater than 255, it is more likely to be faulty, and can be corrected by the faulty-pixel correction module in the CCDC. If properly corrected, the value must be less than 255 when it reaches the preview engine.

The PRV_WSDR_ADDR and PRV_WADD_OFFSET registers must be used to indicate the output address and line offset, respectively, of the dark-frame output in memory.

12.4.7.1.5 Inverse A-Law

To save memory capacity and bandwidth, the CCDC includes an option to apply 10-bit to 8-bit A-Law compression and to pack the sensor data to 1 byte per pixel. To process this data correctly, the inverse A-Law block is provided to decompress the 8-bit nonlinear data back to 10-bit linear data if enabled (PRV_PCR [5] INVALAW). Even if the inverse A-Law block is not enabled, but the input is still 8 bits (PRV_PCR [4] WIDTH), the data is-left shifted by 2 to make it 10-bit data. If the input is 10 bits wide, no operation is performed on the data.

Note: When A-Law compression in CCDC is enabled during dark-frame write, it must be enabled when the stored dark frame is used.

12.4.7.1.6 Dark-Frame Subtract or Shading Compensation

The preview engine can fetch a dark frame containing 8-bit values from memory (8 bits in input are converted internally to 10 bits by adding two zeros to the left) and subtracting it, pixel by pixel, from the incoming input frame (PRV_PCR [6] DRKFEN). This function removes pattern noise in the sensor. The output of the dark-frame subtract operation is 10 bits wide (U10Q0).

There must be adequate memory bandwidth if this feature is enabled. If the data fetched from memory arrives late, the PRV_PCR [31] DRK_FAIL status bit is set to indicate a fail.

Instead of performing the dark-frame subtract, the preview engine can perform lens-shading compensation (if PRV_PCR [21] SCOMP_EN is set along with PRV_PCR [6] DRKFEN). In this case, the 8-bit unsigned value fetched from memory is multiplied by the incoming pixel, and the result is right-shifted by the number of bits specified by the PRV_PCR [24:22] SCOMP_SFT parameter (0-7 bits).

The SBL data read port is shared between the CCDC and preview module. The read port must be affected to the preview module by writing 0 into the ISP_CTRL [28] SBL_SHARED_RPORTB. Programmers must ensure that the CCDC module does not use this port before switching to the preview module.

12.4.7.1.7 Horizontal Median Filter

The preview engine contains a horizontal median filter that can help reduce temperature-induced noise effects. The input and output of the horizontal median filter are 10 bits wide (U10Q0).

Note: Line-width reduction: If the horizontal median filter is enabled, the preview engine reduces the length of the output line of this stage by 4 pixels (2 starting pixels -left edge and 2 ending pixels -right edge). For example, if the input size is 656x490 pixels, the output is 652x490 pixels. There is no truncation of input data line if this block is disabled.

12.4.7.1.8 Noise Filter and Faulty Pixel Correction

The noise filter and couplet defect correction (CDC) operate on the same 33 matrix of same color pixels. Both functions can be enabled separately using the PRV_PCR [27] DCOREN and PRV_PCR [9] NFEN bits.

The faulty pixel correction function replaces the central pixel (x0) with one of the neighbors (x1-x8) in the following cases:

- When it has been identified as faulty.
- When the feature is enabled.

The noise filter modifies the central pixel when it is enabled.

Note: Some details of these features are not available in public domain

12.4.7.1.9 White Balance

The white-balance module has a digital gain adjuster and a white-balance adjuster. In the digital gain adjuster (PRV_WB_DGAIN [9:0] DGAIN), RAW data is multiplied by a fixed-value gain, regardless of the color of the pixel to be processed.

In the white-balance gain adjuster (PRV_WBGAIN), RAW data is multiplied by a selected gain corresponding to the color of the processed pixel.

Note: Some details of this feature are not available in public domain

12.4.7.1.10 CFA Interpolation

The CFA function is responsible for providing full RGB data for each pixel. Depending on sensor type and configuration, interpolation and/or rephasing are required.

The CFA function can be enabled (PRV_PCR [10] CFAEN) and configured to different interpolation modes (PRV_PCR [14:11] CFAFMT).

Note: Some details of this feature are not available in the public domain.

12.4.7.1.10.1 CFA for Bayer Modes

In Bayer modes, the CFA interpolation block is responsible for populating the missing color pixels at a given location, resulting in a 3-color RGB pixel. It does this by interpolating data from neighboring pixels of the same color.

Note: Some details of this feature are not available in the public domain.

12.4.7.1.10.2 CFA Options

If the CFA interpolation block is disabled, the same input pixel is broadcast to all three output pixels.

Note: Image-size reduction: If CFA interpolation is enabled, the preview engine reduces the output of this stage. Two pixels/line in the left, right, top, and bottom edges are truncated in the Bayer/conventional and other modes.

If the CFA format is 2 downsampling in both horizontal and vertical directions, only 2 lines at the top and bottom of the image are truncated. The two left- and right-most pixels are processed.

12.4.7.1.11 Black Adjustment

The CFA interpolation output is three pixels (red, blue, and green values) and this is fed as input to the black-adjustment module, which performs the following calculation for an adjustment of each color level:

data_out = data_in + bl_offset

The bl_offset values for each color are programmable in the PRV_BLKADJOFF register and coded in S8Q0 format (-128..+127). A simple one addition and clip operation are processed in this module. The output data_out [10..0] is signed.

Note: Some details of this feature are not available in the public domain.

12.4.7.1.12 RGB Blending

The RGB2RGB blending module has a general 33 square matrix and redefines the RGB data from the CFA interpolation module, which can be used as a function of a color correction. This is programmable (PRV_RGB_MAT1, PRV_RGB_MAT2, PRV_RGB_MAT3, PRV_RGB_MAT4, PRV_RGB_MAT5, PRV_RGB_OFF1, & PRV_RGB_OFF2) so that the color spectrum of the sensor can be adjusted to the human color spectrum. The input is signed 11 bits and the output is unsigned 10 bits. The following calculation is performed in this module:

$$\begin{bmatrix} R_{out} \\ G_{out} \\ B_{out} \end{bmatrix} = \begin{bmatrix} MTX_RR & MTX_GR & MTX_BR \\ MTX_RG & MTX_GG & MTX_BG \\ MTX_RB & MTX_GB & MTX_BB \end{bmatrix} \begin{bmatrix} R_{in} \\ G_{in} \\ B_{in} \end{bmatrix} + \begin{bmatrix} MTX_OFFR \\ MTX_OFFG \\ MTX_OFFB \end{bmatrix}$$

camisp-E094

Gains are in S12Q8 format, and offsets are in S10Q0 format.

$$\begin{bmatrix} R_{out} \\ G_{out} \\ B_{out} \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} R_{in} \\ G_{in} \\ B_{in} \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}$$

camisp-E161

That leads to:

$$MTX_RR = MTX_GG = MTX_BB = 256$$
$$others = 0$$

camisp-E162

12.4.7.1.13 Gamma Correction

Gamma correction is performed on each of the R, G, and B pixels separately by indexing programmable gamma look-up tables. Each table has 1024 8-bit entries. The input data value is used to index into the table and the table content is the output.

The gamma table can be bypassed (PRV_PCR [26] GAMMA_BYPASS). In this case, the output of the gamma correction is the 8 MSB of the 10-bit input. The gamma table can be written only while the preview engine is disabled.

12.4.7.1.14 RGB to YCbCr Conversion, Luminance Enhancement, Chrominance Suppression, Contrast and Brightness, and 4:2:2 Downsampling and Output Clipping

12.4.7.1.14.1 RGB-to-YCbCr Conversion

The RGB-to-YCbCr conversion module has a 33 square matrix and converts the RGB color space of the image data into the YCbCr color space. In this module, the following calculation is performed using the contents of the PRV_CSC0, PRV_CSC1, PRV_CSC2, and PRV_CSC_OFFSET registers:

$$\begin{bmatrix} Y \\ Cb \\ Cr \end{bmatrix} = \begin{bmatrix} CSCRY & CSCGY & CSCBY \\ CSCRCB & CSCGCB & CSCBCB \\ CSCRCR & CSCGCR & CSCBCR \end{bmatrix} \begin{bmatrix} R_{in} \\ G_{in} \\ B_{in} \end{bmatrix} + \begin{bmatrix} YOFST \\ OFSTCB \\ OFSTCR \end{bmatrix}$$

camisp-E095

Gains are in S10Q8 format, and offsets are in S8Q0 format for chroma and U10Q0 for luma.

Note: Program the following values after reset for correct color conversion:

- PRV_CSC2 [19:10] CSCGCR = 0x39E
- PRV_CSC2 [9:0] CSCRCR = 0x080

12.4.7.1.14.2 Nonlinear Luminance Enhancement

Nonlinear luminance enhancement functions as an edge enhancer (crossed in the horizontal direction). It can be enabled or disabled using the PRV_PCR [15] YNENHEN parameter. If it is enabled, a look-up table with 127 20-bit entries must be programmed. Each entry contains a 10-bit signed offset value in the MSBs, and a 10-bit signed slope in the LSBs.

Note: Some details of this feature are not available in the public domain.

12.4.7.1.14.2.1 Chrominance Suppression

Occasionally, in very bright portions of an image, only one or two of the color channels are saturated, while the remaining channel(s) are not. This can lead to a false-color effect. One common example is the appearance of pink where white should be. Chrominance suppression can be used to correct this issue.

Chrominance suppression can be enabled or disabled using the PRV_PCR [16] SUPEN parameter.

Note: Some details of this feature are not available in the public domain.

12.4.7.1.14.2.2 Contrast and Brightness

The luminance component can be adjusted for contrast (scaling/multiplication) and brightness (offset/addition). Contrast is set in the PRV_CNT_BRT [15:8] CNT field (U8Q4 precision), and brightness is set in the PRV_CNT_BRT [7:0] BRT field (U8Q0 precision).

12.4.7.1.14.2.3 Downsampling and Output Clipping

The 4:2:2 conversion module converts image data to YCbCr-4:2:2 format by averaging every other Cb and Cr component in the horizontal direction. Before outputting the data, the preview engine performs clipping on the YCC components separately. The minimum and maximum threshold values for the Y and C values are specified using the PRV_SETUP_YC register. If no clipping is required, the register must be set to its reset values of 0xFF for the maximum Y and C values, and 0 for the minimum Y and C values.

12.4.7.1.15 Write-Buffer Interface

The output of the preview engine may be passed directly to the resizer (PRV_PCR [19] RSZPORT) and/or written to memory (PRV_PCR [20] SDRPORT).

If the output is written to memory, the write address (PRV_WSDR_ADDR) and line offset (PRV_WADD_OFFSET) must be on 32-byte boundaries. The output format of the YCC data is programmable by setting the PRV_PCR [18:17] YCPOS parameter.

The final width and height of the output vary depending on which processing functions are enabled. Table 12-33 indicates how many edge pixels/lines are truncated by enabling certain modules in the preview engine. These values must be subtracted from the input height and width after the averager to determine the size of preview-engine output.

Table 12-33. Image Cropping by Preview Functions

Image Cropping by Preview Functions		
Function	**Pix/Line**	**Lines**
Horizontal median filter	4	0
Noise filter	4	4
CFA (Bayer, Super CCD Honeycom or RGBE)	4	4
Color suppression OR luminance enhancement	2	0
Maximum total	14	8

Note: Different CFA modes are mutually exclusive.

12.4.7.2 Resizer

12.4.7.2.1 Features

The resizer module enables image upsampling and downsampling, see Table 12-34. The following features are supported:

- Flexible input sources:
  - Preview module: Enables on-the-fly processing
  - Memory: Enables differed processing
- Flexible input format:
  - YUV422 packed data (16 bits)
  - Color-separate data (8 bits). The data must be contiguous in memory. The input source for the data must be the memory: not available for on-the-fly processing.
  - Same output format as input
- Upsampling: Up to 4. Enables digital zoom:
  - General polyphase filter:
    - Ratio of 1 to 4: 4 taps (horizontal and vertical) and 8 phases
- Downsampling: Down to 0.25:
  - General polyphase filter:
    - Ratio of 0.25 to 0.5: 7 taps (horizontal and vertical) and 4 phases
    - Ratio of 0.5 to 1: 4 taps (horizontal and vertical) and 8 phases
- Constraints:
  - The following input width (IW) and output width (OW) constraints must be obeyed due to limited on-chip memory resources.

Table 12-34. Resizer Use Constraints

Resizer Use Constraints			Horizontal Resizer Ratio	
			0.25 to 0.5	0.5 to 4
			7 taps	4 taps
Vertical resizer ratio	0.25 to 0.5	7 taps	OW<=1650	OW<=1650
	0.5 to 4	4 taps	OW<=3312	OW<=3312

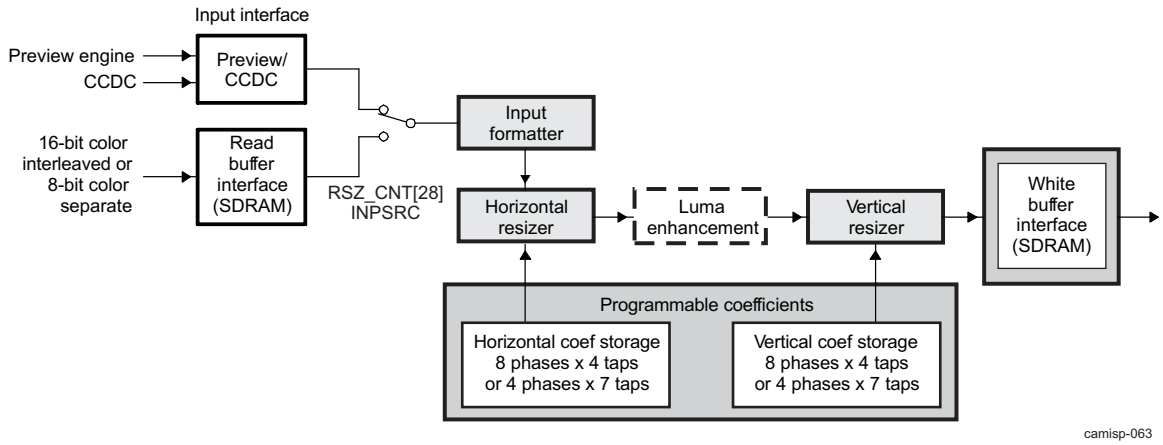
  - The horizontal resizer output rate must not exceed half the functional clock; that is, at 166-MHz functional clock, the horizontal resizer output rate must not exceed 83M pixels/s. This limitation applies only for the on-the-fly processing input source.
- Flexible resizing ratios: Independent resizing factors for the horizontal and vertical directions. The applicable ratio is 256/N, with N ranging from 64 to 1024.
- Programmable luminance enhancer
- Continuous and one-shot operation

12.4.7.2.2 Block Diagram

The resizer module performs either upsampling (digital zoom) or downsampling on image/video data within a range of 0.25 to 4 resizing. The input source can be sent to either the preview engine/CCDC or memory, and the output is sent to memory.

The resizer module performs horizontal resizing, then vertical resizing, independently. Between them there is an optional edge-enhancement feature. This process is shown in Figure 12-80.

Figure 12-80. Resizer Process



12.4.7.2.3 Input and Output Interfaces

The input source can be sent to either the preview engine/CCDC or memory (RSZ_CNT [28] INPSRC). The input width (RSZ_IN_SIZE [12:0] HORZ) must be at least 32 pixels.

12.4.7.2.3.1 Preview Engine/CCDC Input Mode

In the preview engine/CCDC input mode, internal hardware synchronization signals define input frames. The horizontal starting byte (RSZ_IN_START [12:0] HORZ_ST) and vertical starting line (RSZ_IN_START [28:16] VERT_ST) define a starting pixel with respect to the upper-left corner of an input image (signaled through horizontal and vertical synchronization signals). The input width and height in the RSZ_IN_SIZE register specify the exact input range (relative to the starting pixel) necessary to generate an output frame of specified width/height.

Note: Care must be taken to ensure that the input sizes specified by the RSZ_IN_START and RSZ_IN_SIZE registers are less than or equal to the output from the preview engine or CCDC; otherwise, incorrect hardware operation may occur.

RSZ_SDR_INADD and RSZ_SDR_INOFF must be programmed to be 0x0 in this mode.

Also, the output ports of the CCDC (CCDC_SYN_MODE [19] SDR2RSZ) and preview engine (PRV_PCR [19] RSZPORT) to the resizer must be configured so that only one of them is enabled. If both are enabled, the CCDC gains control of this interface.

If input is from the CCDC, the output of the CCDC must be in YUV422 format (the resizer does not support resizing RAW data from the CCDC).

12.4.7.2.3.2 Memory-Input Mode

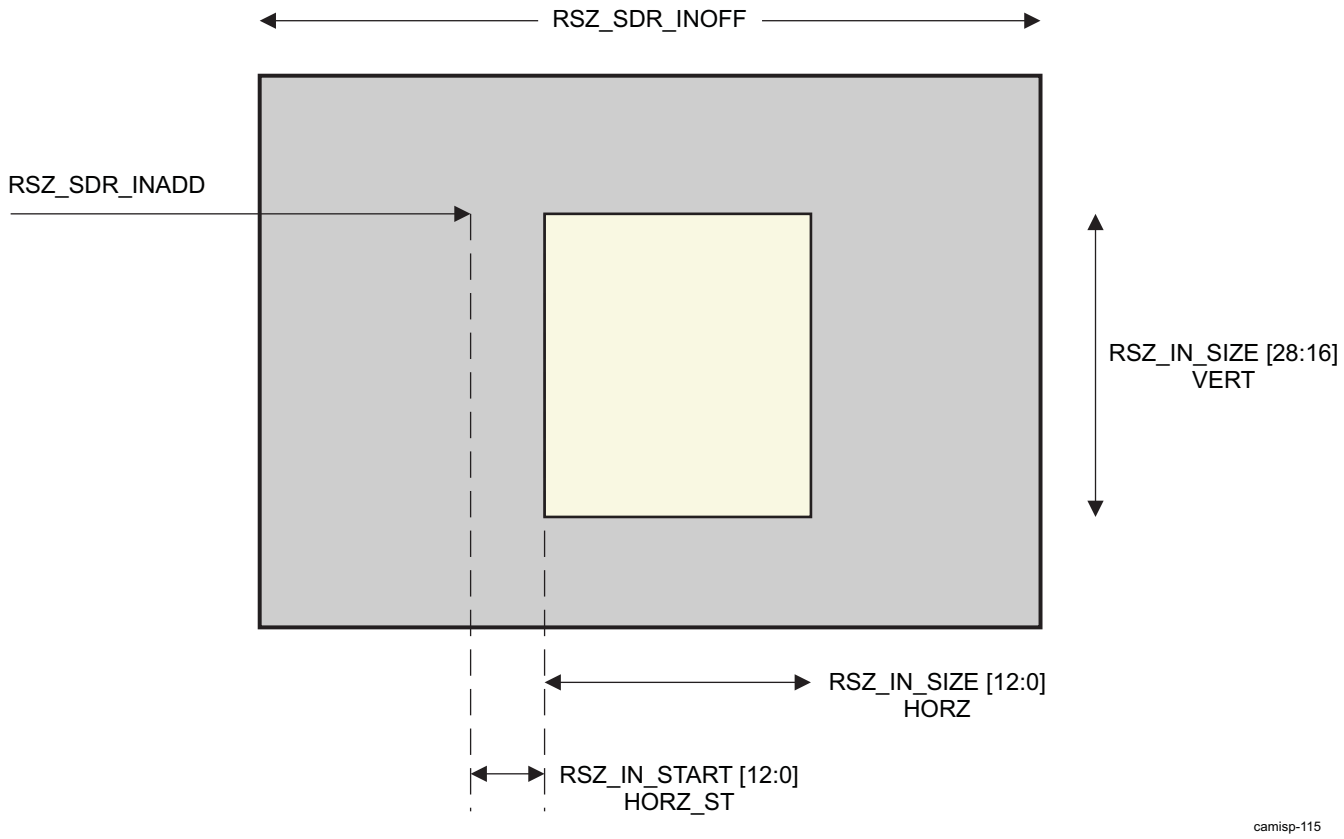
In memory-input mode, the memory address in RSZ_SDR_INADD points to the 32-byte-aligned memory address where the starting pixel resides.

The horizontal starting pixel (RSZ_IN_START [12:0] HORZ_ST) defines a starting pixel within that 32-byte alignment; HORZ_ST is constrained to 0...15 for the 422 format, and 0...31 for the color-separated format.

The vertical starting pixel (RSZ_IN_START [28:16] VERT_ST) must be zero in memory-input mode. The RSZ_SDR_INOFF register specifies the address offset between rows of input data. The input width and height in the RSZ_IN_SIZE register specify the exact input range (relative to the starting pixel) necessary to generate an output frame of specified width/height.

Figure 12-81 shows the resizer in memory-input mode.

Figure 12-81. Resizer in Memory-Input Mode



12.4.7.2.3.3 Output Interface

In both input modes, the RSZ_OUT_SIZE register specifies output width/height, the RSZ_SDR_OUTADD register specifies output starting pixel (upper-left corner) memory address, and the RSZ_SDR_OUTOFF register specifies memory address offset between the beginning of output rows. The resizer output always goes to memory.

Note: RSZ_SDR_INADD, RSZ_SDR_OUTADD, RSZ_SDR_INOFF, and RSZ_SDR_OUTOFF must be 32-byte-aligned; the lower 5 bits of the byte address are assumed to be zero.

Output-width constraints: The output width (RSZ_OUT_SIZE [11:0] HORZ) must be at least 16 pixels, and be even (so that the same number of Cb and Cr components is outputted). Due to the vertical memory size constraint, the output width (RSZ_OUT_SIZE [11:0] HORZ) cannot be greater than:

- 3312 pixels if the vertical resizing ratio is between 1/2 and 4 ((RSZ_CNT [19:10] VRSZ + 1) = 512)
- 1650 pixels wide if the vertical resizing ratio is between 1/2 and 1/4 ((RSZ_CNT [19:10] VRSZ + 1) > 512)

12.4.7.2.4 Horizontal and Vertical Resizing

In the rest of this section:

- HRSZ is used for RSZ_CNT [9:0] HRSZ + 1.
- VRSZ is used for RSZ_CNT [19:10] VRSZ + 1.

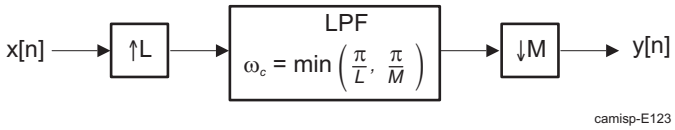
The resizer module can upsample or downsample image data with independent resizing factors in the horizontal and vertical directions. The HRSZ and VRSZ parameters can range from 64 to 1024 to give a resampling range from 4 to 0.25 (256/HRSZ or 256/VRSZ).

The resizer module uses the same resampling algorithm for the horizontal and vertical directions.

The resizing/resampling algorithm uses a programmable polyphase sample rate converter (resampler). The polyphase filter coefficients are programmable so that any user-specified filter can be implemented.

Figure 12-82 shows a general sample-rate converter where the resampling rate is equal to L/M.

Figure 12-82. Typical Sample-Rate Converter



L phases are used in a typical polyphase implementation. The resizer module, however, fixes the number of phases to 8 for a resizing range of 1/2 ~ 4 (RSZ = 64 ~ 512), or 4 for a resizing range of 1/4 ~1/2 (RSZ = 513 ~ 1024). In this way, the upsampling value (L) is fixed to either 8 or 4, and the downsampling value (M) is based on RSZ. To achieve a resizing ratio of 256/RSZ, the downsampling value (M) is equal to (PxRSZ)/256, where P is the number of phases. Figure 12-83 shows the resizer functionality.

Figure 12-83. Resizer Functionality

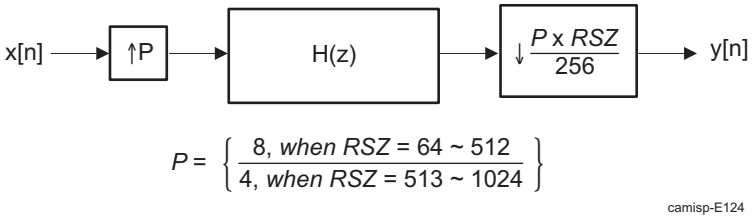
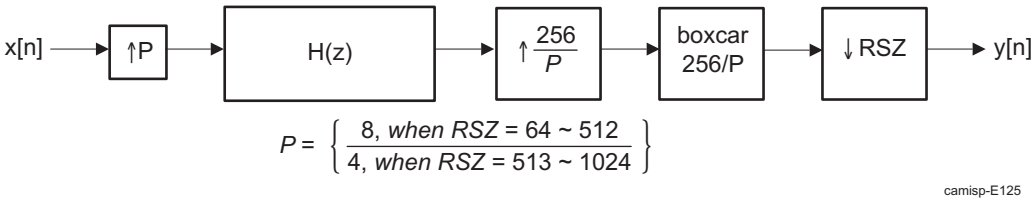


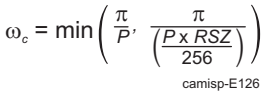
Figure 12-84 shows a model of the resolution of the noninteger downsampling ratio. The interpolated output from the filter is upsampled and replicated 256/P times before it is downsampled by the RSZ factor.

Figure 12-84. Resizer Approximation Scheme



This implementation means that a resizing ratio with 256/RSZ times the input size can be obtained. However, each output pixel is rounded to the nearest interpolated output of 1/P input-pixel precision. The polyphase filter coefficients are programmable so that any user-specified filter can be implemented. It is recommended that coefficient sets be chosen to implement a sample rate converter where a low-pass filter is used, with a cutoff frequency as shown in Figure 12-85.

Figure 12-85. Cutoff Frequency for Low-Pass Filter



If polyphase resampling is used, all upsampling factors can share the same set of coefficients. However, a different coefficient set is required when changing between 8-phase and 4-phase modes, and with different downsampling factors.

32 programmable coefficients are available for the horizontal direction (registers RSZ_HFILT10 to

RSZ_HFILT3130) and another 32 programmable coefficients are available for the vertical direction (registers RSZ_VFILT10 to RSZ_VFILT3130). The 32 programmable coefficients are arranged as either 4 taps and 8 phases for the resizing range of 1/2 ~ 4 (RSZ = 64 ~ 512), or 7 taps and 4 phases for a resizing range of 1/4 ~1/2 (RSZ = 513 ~ 1024)(RSZ is either HRSZ or VRSZ). Table 12-35 shows the arrangement of the 32 filter coefficients. Each tap is arranged in S10Q8 format.

Table 12-35. Arrangement of the Filter Coefficients

Filter Coefficient	0.5x to 4x		0.24x to ~0.5x	
	Phase	Tap	Phase	Tap
0	0	0	0	0
1		1		1
2		2		2
3		3		3
4	1	0		4
5		1		5
6		2		6
7		3		Not used
8	2	0	1	0
9		1		1
10		2		2
11		3		3
12	3	0		4
13		1		5
14		2		6
15		3		Not used
16	4	0		
17		1		
18		2		
19		3		
20	5	0		
21		1		
22		2		
23		3		Not used
24	6	0	3	0
25		1		1
26		2		2
27		3		3
28	7	0		4
29		1		5
30		2		6
31		3		Not used

The indexing scheme of coefficients is oriented for dot-product (or inner product), rather than for impulse response. In other words, the first data point contributing to a particular output is multiplied by the coefficient associated with tap 0, and the last data point is multiplied by the coefficient associated with tap 3 or tap 6 (depending on whether it is 4-tap or 7-tap mode). The normal raster-scan order is used where the upper-left corner gets the (0, 0) coordinate. Pixel 0 is the left-most column of pixels for horizontal resizing, and the top-most row of pixels for vertical resizing. Figure 12-86 shows an example of the alignment of input pixels to tap coefficients using a simple 1:1 resize case (4-tap mode). In this example, only one phase output is necessary.

Figure 12-86 shows the alignment of input pixels to tap coefficients.

Figure 12-86. Alignment of Input Pixels to Tap Coefficients

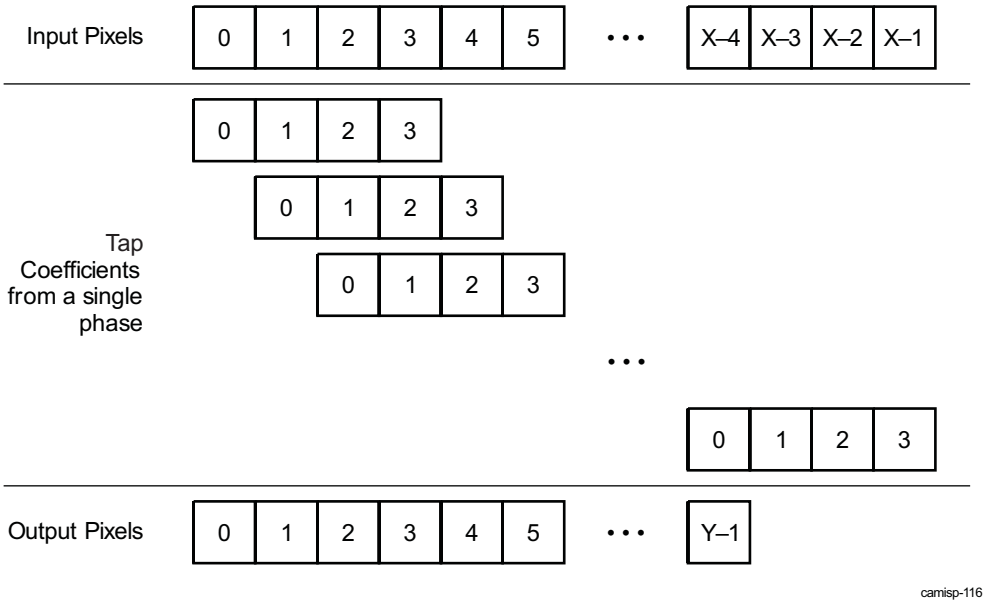


Figure 12-86 also shows how the first output is computed when all taps are aligned with input pixels. To compute the last several pixels in each line/column, the filter requires more input pixels than the following equation calculates:

input size = output size*RSZ/256

In the former example, where the input size is X and the output size is Y, three extra input pixels are required to generate the correct number of output pixels:

X = Y*256/256 + 3 (to account for the extra pixels required for filtering)

The input size calculation depends on the starting phase and rounding issues in the algorithm. Table 12-36 lists the input size calculations derived from the algorithm description in Section 12.4.7.2.5. The input width and height parameters must be programmed strictly according to these equations; otherwise, incorrect hardware operation may occur.

Table 12-36. Input Size Calculations

	8-Phase, 4-Tap Mode	4-Phase, 7-Tap Mode
RSZ_IN_SIZE[12:0] HORZ	(32*sph + (ow - 1)*hrsz + 16) >> 8 + 7	(64*sph + (ow - 1)*hrsz + 32) >> 8 + 7
RSZ_IN_SIZE[28:16] VERT	(32*spv + (oh - 1)*vrsz + 16) >> 8 + 4	(64*spv + (oh - 1)*vrsz + 32) >> 8 + 7

Where:

sph = Start phase horizontal (RSZ_CNT [22:20] HSTPH)

spv = Start phase vertical (RSZ_CNT [25:23] VSTPH)

ow = Output width (RSZ_OUT_SIZE [11:0] HORZ + extra)

oh = Output height (RSZ_OUT_SIZE [27:16] VERT)

hrsz = Horizontal resize value (RSZ_CNT [9:0] HRSZ + 1)

vrsz = Vertical resize value (RSZ_CNT [19:10] VRSZ + 1)

extra = 0 when RSZ_YENH [17:16] ALGO = 0 (edge enhancement disabled)
extra = 4 when RSZ_YENH [17:16] ALGO != 0 (edge enhancement enabled)

The horizontal and vertical starting phases can be programmed in the RSZ_CNT [22:20] HSTPH and RSZ_CNT [25:23] VSTPH fields, respectively. The chrominance data can be resized using bilinear interpolation or the same algorithm as the luminance data (RSZ_CNT [29] CBILIN). For more information about how these fields are used in the algorithm, see Section 12.4.7.2.5, *Resampling Algorithm*.

12.4.7.2.5 Resampling Algorithm

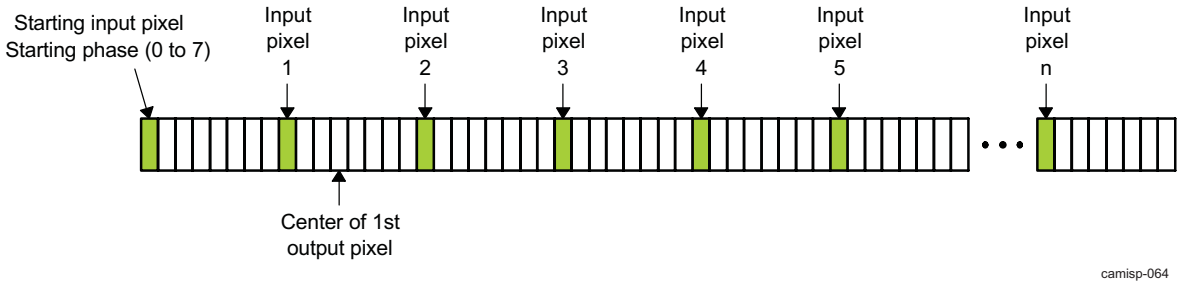
The resizer module uses the same resampling algorithm for the horizontal and vertical directions. For the rest of this section, the horizontal direction is used in describing the resampling algorithm. The algorithm is described first for the 4-tap/8-phase mode, then for the 7-tap/4-phase mode.

Section 12.4.7.2.5.1 and Section 12.4.7.2.5.2 explain the generic algorithm without detailing the differences between color components. Section 12.4.7.2.5.3 describes how interleaved chroma are processed when input is YUV422.

12.4.7.2.5.1 4-Tap/8-Phase Mode

In the 4-tap/8-phase mode, the coefficients for each of the 8 phases may be set to interpolate 8 intermediate pixels between input pixels. For each output pixel calculation, a fine-input pointer with 1/256 input-pixel precision is incremented by the RSZ_CNT [9:0] HRSZ +1 value. A coarse-input pointer with 1/8 input-pixel precision (corresponds to one of the 8 phases) is calculated by rounding the fine-input pointer to the nearest 1/8 pixel. The output pixel is calculated by the dot product of the coefficients of the phase filter (selected by the coarse-input pointer) and the appropriate four input pixels. Figure 12-87 shows a pseudo-code description of the resizer algorithm in the 4-tap/8-phase mode.

Figure 12-87. Pseudo-Code Description of the Resizer Algorithm in the 4-Tap/8-Phase Mode



- The starting input pixel location (in whole pixels) (see Section 12.4.7.2.3, *Input and Output Interfaces*) and the starting phase (in 1/8 pixel) (RSZ_CNT [22:20] HSTPH) are programmed through the resizer registers.
- A fine-input pointer is maintained in 1/256-pixel precision.
- A coarse-input pointer and a pixel-input pointer are computed for each output, based on the fine-input pointer.
- The coarse-input pointer is in 1/8 pixel precision. The pixel-input pointer is in whole-pixel precision.
- Initially, fine-input pointer = 256* starting input pixel + 32* starting phase - 256. The fine-input pointer defines the starting 1/8 pixel location covered by the filter waveform.
- For each output pixel:

```
Coarse-input pointer =                  /* Rounded to the nearest phase */
(fine-input pointer + 16) >> 5

Pixel-input pointer =                   /* Rounded up to a whole pixel, when already on an
(coarse-input pointer >> 3) + 1            integer pixel, go to next one to simplify coefficient
                                           organization */

Coefficient phase =                     /* 3 LSBs = phase */
(coarse-input pointer & 7)

Output = dot product of the 4 coefficients and the 4 inputs starting with pixel-input pointer

Clip output to 8-bit unsigned for luma, 8-bit signed for chroma
```

fine-input pointer = fine-input pointer
+ (RSZ_CNT [9:0] HRSZ +1)

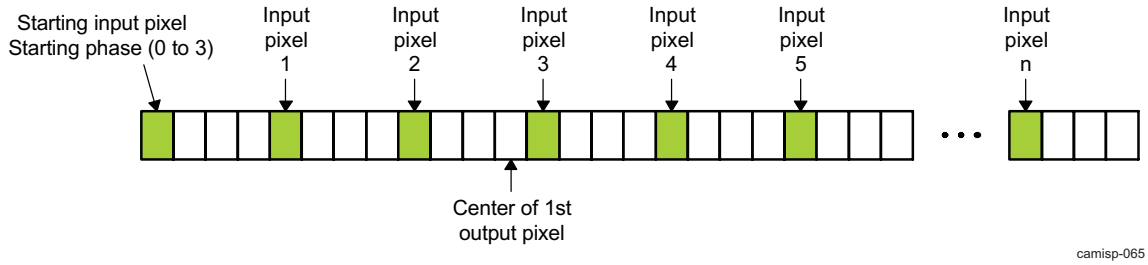
/* distance between outputs = 1/resize_factor = (RSZ_CNT [9:0] HRSZ + 1) /256 = (RSZ_CNT [9:0] HRSZ + 1) in 1/256 precision */

- Same algorithm in the horizontal and vertical directions, except with separate initial pixel/phase values and separate RSZ values.

12.4.7.2.5.2 7-Tap/4-Phase Mode

In the 7-tap/4-phase mode, the coefficients for each of the 4 phases may be set to interpolate 4 intermediate pixels between input pixels. For each output pixel calculation, a fine-input pointer with 1/56 input-pixel precision is incremented by the RSZ_CNT [9:0] HRSZ + 1 value. A coarse-input pointer with 1/2 input-pixel precision (corresponds to one of the 4 phases) is calculated by rounding the fine-input pointer to the nearest 1/2 pixel. The output pixel is calculated by the dot product of the coefficients of the phase filter (selected by the coarse-input pointer) and the appropriate 7 input pixels. Figure 12-88 shows a pseudo-code description of the resizer algorithm in the 7-tap/4-phase mode.

Figure 12-88. Pseudo-Code Description of the Resizer Algorithm in the 7-Tap/4-Phase Mode



- The starting input-pixel location (in whole pixels) (see Section 12.4.7.2.3, *Input and Output Interfaces*) and the starting phase (in 1/2 pixel) (RSZ_CNT [22:20] HSTPH) are programmed through the resizer registers.
- A fine-input pointer is maintained in 1/256 pixel precision.
- A coarse-input pointer and a pixel-input pointer are computed for each output based on the fine-input pointer.
- The coarse-input pointer is in 1/2 pixel precision. The pixel-input pointer is in whole-pixel precision.
- Initially, fine-input pointer = 256 * starting input pixel + 64 * starting phase - 256. The fine-input pointer defines the starting 1/2 pixel location covered by the filter waveform.
- For each output pixel:

Coarse-input pointer = (fine-input pointer + 32) >> 6 /* Round to the nearest phase */

Pixel-input pointer = (coarse-input pointer >> 2) + 1 /* Round up to a whole pixel; when already on an integer pixel, go to next one to simplify coefficient organization */

Coefficient phase = (coarse-input pointer & 3) /* 2 LSBs = phase */

Output = Dot product of the 7 coefficients and the 7 inputs starting with the pixel-input pointer

Clip output to 8-bit unsigned for luma, 8-bit signed for chroma

/* It is acceptable to require the 8th coefficients to be filled with zeros by firmware so that 8 coefficients and 8 inputs, the last input being *don't care* value, are multiply-added */

Fine-input pointer = Fine input pointer + (RSZ_CNT [9:0] HRSZ + 1)

/* Distance between outputs = 1/resize_factor = (RSZ_CNT [9:0] HRSZ + 1)/256 = (RSZ_CNT [9:0] HRSZ + 1) in 1/256 precision */

- Same algorithm in both the horizontal and vertical directions, but with separate initial pixel/phase values and separate RSZ values.

Note: The pixel-input pointer (pip) in the algorithm description, points to pixels, not bytes or shorts in the memory. The fine-input pointer (fip) points to a 1/256 resolution sub-pixel position. The coarse-input point (cip) points to a 1/8 or 1/4 resolution sub-pixel location, depending on the number of phases.

12.4.7.2.5.3 Horizontal Resizing With Interleaved Chroma

Chroma inputs, Cb and Cr, are 8-bit unsigned values that represent 128-biased 8-bit signed values (the signed chroma are called U and V instead of Cb and Cr). During the resizing computation, the chroma values have the 128 bias subtracted to convert to the 8-bit signed format. After vertical resizing, the 128 bias is added back to convert back to 8-bit unsigned format.

Chroma components, which are 2:1 horizontally downsampled with respect to luma, have two methods of horizontal resizing processing: filtering with luma, and bilinear interpolation.

The horizontal resizing with interleaved chroma option can be selected in the RSZ_CNT [29] CBILIN field independently of the RSZ_CNT [22:20] HSTPH parameters. However, filtering with luma is intended only for downsampling, and bilinear interpolation is intended only for upsampling.

For horizontal resizing of Y/Cb/Cr in a combined filtering flow, the algorithm is modified as shown in the following algorithm descriptions:

Filter Chroma With Luma (4-Tap/8-Phase Mode):

```
For (i=0; i<output_width; i++) { /* output width depends of input width and resizing factors*/
        Coarse-input pointer = (fine-input pointer + 16) >> 5     /* Round to nearest phase */
        Pixel-input pointer = (coarse-input pointer >> 3) + 1     /* Round up to a whole pixel */
        Coefficient phase = pixel-input pointer & 7               /* 3 LSB = phase */

        if (i & 1 == 0) { /* even output pixel, generate YCbCr */
            Yout = dot product of the 4 coefficients and the 4 Y inputs starting with pixel-input pointer
            Cbout = dot product of the 4 coefficients and the 4 upsampled Cb inputs starting with
            pixel-input pointer
            Crout = dot product of the 4 coefficients and the 4 upsampled Cr inputs starting with
            pixel-input pointer
            Clip outputs to 8-bit unsigned for luma, 8-bit
            signed for chroma
        }
        Else {/* odd output pixel, generate Y only */
            Yout = dot product of the 4 coefficients and the 4 Y inputs starting with pixel-input pointer
            Clip output to 8-bit unsigned

        }
        Fine-input pointer = fine-input pointer + (RSZ_CNT [9:0] HRSZ + 1)
}
        }
```

Filter Chroma With Luma (7-Tap/4-Phase Mode):

```
For (i=0; i<output_width; i++) { /* output width depends of input width and resizing factors*/
        Coarse-input pointer = (fine-input pointer + 32) >> 5     /* Round to nearest phase */
        Pixel-input pointer = (coarse-input pointer >> 2) + 1     /* Round up to a whole pixel */
        Coefficient phase = pixel-input pointer & 3               /* 2 LSB = phase */
        if (i & 1 == 0) { /* Even output pixel, generate YCbCr
        */
            Yout = dot product of the 7 coefficients and the 7 Y inputs starting with pixel-input pointer
            Cbout = dot product of the 7 coefficients and the 7 upsampled Cb inputs starting with
            pixel-input pointer
            Crout = dot product of the 7 coefficients and the 7 upsampled Cr inputs starting with
            pixel-input pointer
            Clip outputs to 8-bit unsigned for luma, 8-bit
            signed for chroma
        }
        Else {/* Odd output pixel, generate Y only */
            Yout = dot product of the 7 coefficients and the 7 Y inputs starting with pixel-input pointer
            Clip output to 8-bit unsigned
        }
        Fine-input pointer = fine-input pointer + (RSZ_CNT [9:0] HRSZ + 1)
}
   }
```

Note: The chroma input values are internally replicated to realize 1:2 upsampling to line up with luma input values. Only required chroma outputs are computed; they correspond to even luma outputs.

Bilinear Interpolation (4-Tap or 7-Tap):

For the bilinear interpolation flow of chroma horizontal resizing, the algorithm is adapted as follows. For the bilinear interpolation option, it is not necessary to replicate chroma samples.

```
For (i=0; i<output_width; i++) {
    if (i & 1 == 0) { /* even output pixel, generate YCbCr */
        Coarse-input pointer = ... /*Calculation issued from 4 taps or 7
        taps*/
        Pixel-input pointer = ... /*Calculation issued from 4 taps or 7 taps*/
        Yout = dot product of ... /*Calculation issued from 4 taps or 7
        taps*/
        C_fine_input_pointer = fine_input_pointer + 128*ntaps     /* Points to center of filter
                                                                  kernel */
        Cidx = C_fine_input_pointer >> 9                          /* Truncate to even pixel
                                                                  grid to find left value */
        Cbin[0] = Cb[Cidx]
```

```
            Cbin[1] = Cb[Cidx + 1]
            Crin[0] = Cr[Cidx]
            Crin[1] = Cr[Cidx + 1]
            frac = C_fine_input_pointer & 511                    /* 9-bit fraction */
            Cbout = ((512 - frac) * Cbin[0] + frac * Cbin[1] + 256) >> 9
            Crout = ((512 - frac) * Crin[0] + frac * Crin[1] + 256) >> 9
            Clip outputs to 8-bit unsigned for luma, 8-bit signed for chroma
            Fine-input pointer = fine-input pointer + (RSZ_CNT [9:0] HRSZ + 1)
}
            Else {/* odd output pixel, generate Y only */
            . . .
            }
    }
```

In the former algorithm, fixed-point arithmetic is used. The variable *frac* is an unsigned integer representing the fraction f. Thus, 1- f becomes 512 - frac. After the sum of products, 256, representing 0.5 real-numbers, is added to the sum, and then the sum is right-shifted by 9 bits to get back to the integer chroma representation.

In both algorithm options, the chroma outputs computed are interleaved with luma values to generate the YCbYCr output format (or the alternate format specified in RSZ_CNT [26] YCPOS).

In the vertical resizing stage, the two chroma planes are processed interleaved as one separate image. Because there is no resolution issue vertically, and no horizontal dependency in vertical resizing, the vertical scheme is consistent with conventional processing, and is not analyzed here.

12.4.7.2.5.4 Algorithm Functionality

Table 12-37 is an example of 1:2.56 (hrsz = 100) horizontal resizing that illustrates the address calculation and chroma processing in 4:2:2 format (4-tap 8-phase mode). The starting pixel and phase are assumed to be zero.

Table 12-37. Processing Example for 1:2:56 Horizontal Resize

Output	Y0	Cb0	Cr0	Y1	Y2	Cb2	Cr2	Y3	Y4	Cb4	Cr4	Y5
fip (+= hrsz)	-256			-156	-56			44	144			244
cip (= (fip+16)>>5)	-8			-5	-2			1	5			8
pip (= (cip>>3) + 1)	0			0	0			1	1			2
coef ph (= cip & 7)	0			3	6			1	5			0
Inputs needed (chroma filtered like luma)	Y0	Cb0	Cr0	Y0	Y0	Cb0	Cr0	Y1	Y1	Cb0	Cr0	Y2
	Y1	Cb2	Cr0	Y1	Y1	Cb0	Cr0	Y2	Y2	Cb2	Cr2	Y3
	Y2	Cb2	Cr2	Y2	Y2	Cb2	Cr2	Y3	Y3	Cb2	Cr2	Y4
	Y3	Cb2	Cr2	Y3	Y3	Cb2	Cr2	Y4	Y4	Cb4	Cr4	Y5
Cfip (= fip+512)		256				488				720		
Cidx (= Cfip >> 9)		0				0				1		
Inputs needed for chroma bilinear interpolation		Cb0	Cr0			Cb0	Cr0			Cb2	Cr2	
		Cb2	Cr2			Cb2	Cr2			Cb4	Cr4	

Note the distinction between using {Cb0, Cb0, Cb2, Cb2} and {Cb0, Cb2, Cb2, Cb4} as input to the filter. The 4 filter taps are applied in order, so with the different chroma component repetition, the result is different (even when the coefficient phase is the same).

The Cidx of the chroma bilinear interpolation flow points to the chroma sample in linear array order, so Cidx = 1 means we Cb2 and Cb4 are being grabbed.

12.4.7.2.6 Luma Edge Enhancement

Edge enhancement can be applied to the horizontally resized luminance component before the output of the horizontal stage is sent to the line memories and the vertical stage. The RSZ_YENH [17:16] ALGO parameter can be set to disable edge enhancement, or to select a 3-tap or a 5-tap horizontal high-pass filter (HPF) for luminance enhancement. The edge enhancement algorithm is as follows:

If edge enhancement is selected, the two left-most and two right-most pixels in each line are not outputted to the line memories and the vertical stage. The RSZ_OUT_SIZE [11:0] HORZ register is the final output width, up to 1280 pixels when vertical 4-tap mode is used, and up to 640 pixels when vertical 7-tap mode is used. When edge enhancement is enabled, the horizontal resizer output width used to calculate the required input width must be RSZ_OUT_SIZE [11:0] HORZ + 4.

Note: Some details of this feature are not available in the public domain,

RSZ_YENH [7:0] CORE is in U8Q0. RSZ_YENH [11:8] SLOP is in U4Q4. RSZ_YENH [15:12] GAIN is in U4Q4.

12.4.8 Statistics Collection Modules

The statistics-collection modules (SCMs) are the H3A and histogram modules that provide statistics on the incoming images to help designers of camera systems.

12.4.8.1 Statistics Collection: H3A

12.4.8.1.1 Features

The H3A module supports control loops for autofocus, auto white balance, and auto exposure by collecting metrics about the imaging/video data. The metrics are used to adjust parameters for processing the imaging/video data. There are two main blocks in the H3A module:

- Autofocus engine (AF):
  The AF submodule extracts and filters the red, green, and blue data from input image data and provides the accumulation or peaks of the data in a specified region. The specified region is a two-dimensional block of data referred to as a paxel. The AF engine supports the following features:
  - Peak mode in a paxel (a paxel is defined as a two dimensional block of pixels): Accumulate the maximum focus value of each line in a paxel.
  - Accumulation of the maximum focus value of each line in a paxel
  - Accumulation mode
  - Accumulation/sum mode (instead of peak mode): Accumulation of focus value in a paxel
  - Up to 36 paxels/windows in the horizontal direction and up to 128 paxels/windows in the vertical direction
  - Programmable width and height for the paxel/window
  - Programmable red, green, and blue position within a 22 matrix
  - Separate horizontal start for paxel and filtering
  - Programmable vertical line increments within a paxel
  - Parallel IIR filters configured in a dual-biquad configuration with individual coefficients (2 filters with 11 coefficients each)
- Auto exposure and auto white balance engine (AE/AWB)
  The AE/AWB engine accumulates values and checks for saturated values in a subsampling of the

video data. In the case of the AE/AWB, the two-dimensional block of data is referred to as a window. Thus, other than having different names, paxels and windows are essentially the same. However, the numbers, dimensions, and starting positions of AF paxels and AE/AWB windows are programmable separately. AE/AWB supports the following features:

- Accumulation of clipped pixels along with all nonsaturated pixels
- Up to 36 horizontal windows/paxel and up to 128 vertical windows/paxel
- Separate vertical start coordination and height for a black row of paxels different from the remaining color paxels
- Programmable horizontal and vertical sampling points in a window

Note: Some details of this feature are not available in the public domain.

12.4.8.1.2 Autofocus Engine

The autofocus engine works by extracting each red, green, and blue pixel from the video stream and subtracting a fixed offset from the pixel value (128 when A-Law is enabled or 512 when A-Law is disabled). The offset value is then passed through two IIR filters, and the absolute values of the filter outputs are the focus values (FV). For each paxel, the pixel values and the two focus value outputs are accumulated for each color and sent to memory. The following sections describe this process in more detail.

Only RAW10 data is supported by the autofocus function. In some cases, RAW8 or RAW12 data can be converted to RAW10 using the data-lane shifter.

12.4.8.1.3 AE/AWB Engine

The AE/AWB engine starts by dividing the frames into windows and further subsampling each window into 2x2 blocks. Then, for each subsampled 2x2 block, each pixel is accumulated. Also, each pixel is compared to a limit set in a register. If any pixels in a 2x2 block are greater than or equal to the limit, the block is not counted in the unsaturated block counter. Pixels greater than the limit are replaced by the limit, and the value of the pixel is accumulated.

12.4.8.2 Statistics Collection: Histogram

12.4.8.2.1 Features

The histogram accepts RAW image/video data from either the video-port interface of the CCDC or from memory, performs a color-separate gain on each pixel (white/channel balance), and bins the pixels according to the amplitude, color, and region specified through the CCDC register settings. It can support 4 colors (Bayer), and up to 4 regions, simultaneously. Figure 12-89 shows the processing of the histogram module.

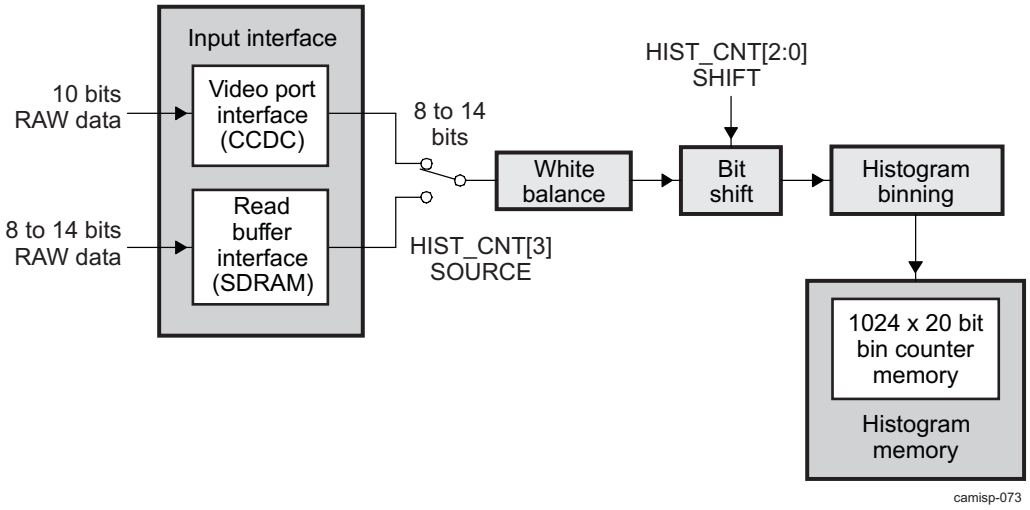
The histogram module is typically used with 3A metrics by the host processor to adjust various parameters for processing image data. The following features are supported:

- Flexible input: Input data can come from the RAW image sensor (through the CCDC module) or from memory.
- Color-separate gain: A digital gain per color component can be applied before histogram computation.
- Histogram computation: The module performs pixel binning of the incoming RAW image data. Each bin collects the number of pixels with values in a range. If no saturation occurs, the sum of the values in every bin is equal to the total number of pixels in the input image. The histogram computation can occur over one frame or be accumulated over multiple frames. The computation occurs on rectangular regions. A histogram is computed for each color component in the image:
  - The RAW image data dynamic can be up to 10 bits (pixel values in the range 0 to 1023).
  - Range: Each bin covers a pixel range. Each bin can cover a maximum of 128 pixel values.
  - Programmable regions: There can be up to four regions. The region positions and horizontal and vertical sizes are programmable. When regions overlap, pixels from the overlapped area are

accumulated into the highest-priority region only; region0 has the highest priority and region3 the lowest priority.
  - Programmable number of bins: There can be 32, 64, 128, or 256 bins per color and per region. The number of bins depends on the number of regions, because histogram memory size is fixed.
  - Color components: A histogram is computed for each color component in the RAW image. There are usually 4 color components in Bayer image sensors.
- Saturation: If the pixel count exceeds 2^{20}-1, the pixel count is saturated.
- Memory clear: The histogram memory is cleared automatically when it is read.
- Output: The histogram result is stored in a local RAM read by a system initiator, typically the system DMA, to be written to memory.

12.4.8.2.2 Block Diagram

Figure 12-89. Histogram Process



12.4.8.2.3 Input Interface

The histogram receives RAW image/video data from the video port interface through the CCDC (which is interfaced to an external sensor) or from the read buffer interface through memory (HIST_CNT [3] SOURCE).

The input data is 10 bits wide if the source is the video-port interface. When the input source is from memory, data-bit width can range from 8 to 14 bits. If the input data is 8 bits packed (memory contains two 8-bit pixels for every 16 bits), the HIST_CNT [8] DATSIZ bit must be set. If memory contains one pixel for every 16 bits, the DATSIZ bit must be cleared.

Likewise, if memory contains one pixel for every 16 bits, the DATSIZ bit must be cleared. The input memory address (HIST_RADD) and line-offset (HIST_RADD_OFF) registers are used to specify the location of the input frame in memory. Both of these registers should be aligned on 32-byte boundaries.

The frame-input width and height are configured using the HIST_H_V_INFO [29:16] HSIZE and HIST_H_V_INFO [13:0] VSIZE register fields, respectively.

The histogram module supports 4-color Bayer color patterns. The HIST_CNT [6] CFA field is used to select the color pattern of the input data (Bayer).

12.4.8.2.4 White Balance

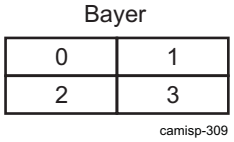
A white-balance gain can be separately applied to each color channel by programming the fields in the HIST_WB_GAIN register. Table 12-38 indicates which pixel index in the color pattern corresponds to each field in the WB_GAIN register.

Table 12-38. White Balance Field-to-Pattern Assignments

HIST_WB_GAIN fields	Bayer
WG00	0
WG01	1
WG02	2
WG03	3

Note: There are four different physical locations, with one color per location for Bayer. See Figure 12-90.

Figure 12-90. Color Pattern Index



Each gain constant is 8 bits wide with 5 bits of decimal precision (U8Q5).

12.4.8.2.5 Histogram Binning

The histogram bins the input data by amplitude, color, and region (see Figure 12-91). Each bin is a counter, counting the number of pixels of a color in the range associated with the bin. The number of bins can be programmed to 32, 64, 128, or 256 bins in the HIST_CNT [5:4] BINS field. However, due to limited histogram memory size (1024 words), the number of bins (times 4 colors) limits the number of regions that can be active, as shown in Table 12-39.

Table 12-39. Regions and Bins for Histogram

Number of Bins	Number of Regions Allowed
256	1
128	2
64	4
32	4

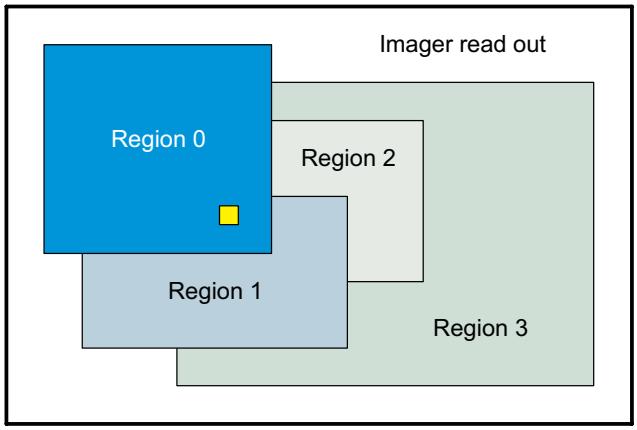
As indicated by Table 12-39, up to four overlapping regions can be designated within the frame. Each region is defined by the horizontal starting (HIST_Rn_HORZ [29:16] HSTART) and ending (HIST_Rn_HORZ [13:0] HEND) pixels, and the vertical starting (HIST_Rn_VERT [29:16] VSTART) and ending (HIST_Rn_VERT [13:0] VEND) lines (where n is the region number 0...3).

Note: If the starting and ending pixels/lines are the same, the region size is treated as zero, and there is no binning for such a region.

12.4.8.2.5.1 Region Priority

Up to four regions can be active at any time, but a pixel is binned into only one region. The priority is Region 0 > Region 1 > Region 2 > Region 3. For example, the yellow pixel in Figure 12-91 is binned only for Region 0, although it is present in all four regions.

Figure 12-91. Region Priority



12.4.9 Central-Resource Shared Buffer Logic

The central-resource shared buffer logic (SBL) receives data read and write requests from the CCDC, preview, H3A, histogram, resizer, CSIa, and CSIb modules. It arbitrates between requestors and constructs bursts to transfer data to and from memory.

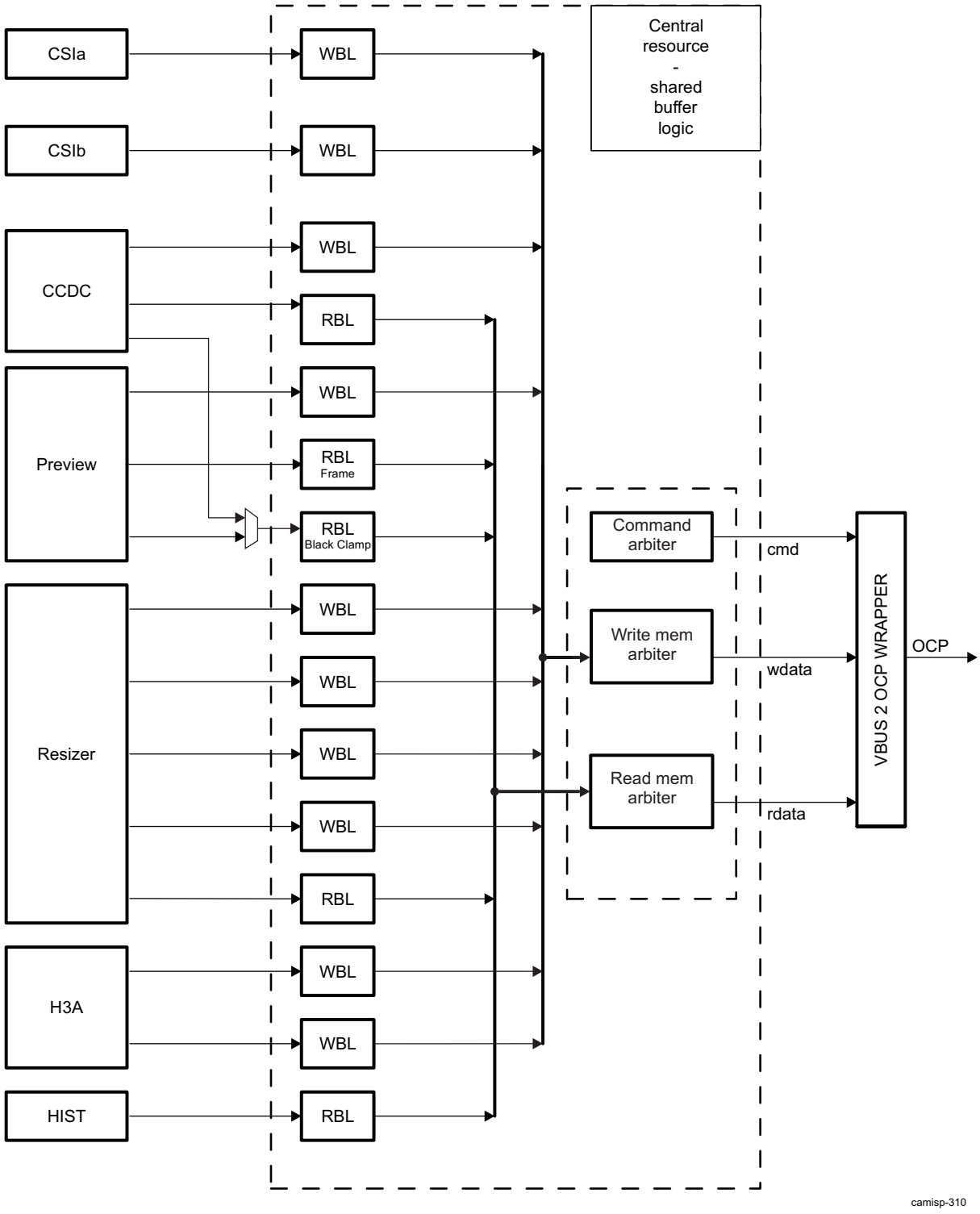
The central-resource SBL performs the following functions:

- Interface to the CCDC module:
  - Collects output data from the CCDC in the write buffer (1 port)
  - Transfers faulty-pixel table data to the CCDC from the read buffer (2 ports)
  - Transfer lens-shading compensation to the CCDC engine from the read buffer. This port is shared with PREVIEW module dark frame subtract port.
- Interface to the CSI receiver modules:
  - Collects output from CSIa in the write buffer (1 port)
  - Collects output from CSIb in the write buffer (1 port)
  - Transfer input data to the CSIb from the read buffer. This port is shared with the PREVIEW module input data read port.
- Interface to the preview module:
  - Collects output data from the preview engine in the write buffer (1 port)
  - Transfer input data to the PREVIEW engine from the read buffer (1 ports). This port is shared with CSIb data read port.
  - Transfer dark frame subtract data to the PREVIEW engine from the read buffer . This port is shared with CCDC lens-shading compensation read port.
- Interface to the H3A module:
  - Collects output data from the H3A in the write buffer (2 ports)
- Interface to the histogram module:
  - Transfers input data to the histogram from the read buffer (1 port)
- Interface to the resizer module:
  - Collects output data from the resizer in the write buffer (4 ports)
  - Transfers input data to the resizer from the read buffer (1 port)
- Requests arbitration between the different initiators. Based on fixed priorities.
- Performs throttle memory read requests for preview, resizer, and histogram to limit bandwidth in memory-to-memory operations

12.4.9.1 Block Diagram

Figure 12-92 shows the central-resource SBL. It comprises WBL and RBL blocks, read and write buffers, and arbitration logic. The VBUSM data-width to the memory is 64 bits.

Figure 12-92. Central-Resource SBL Block Diagram



12.4.9.2 Functional Operations

12.4.9.2.1 Parameters

Table 12-40 summarizes the central-resource SBL parameters. Those parameters are fixed at design time and cannot be changed by users. The functional operations of the central-resource SBL are based on a fixed data size of 256 bytes called a data unit (DU). P0 has the highest priority and P14 has the lowest priority.

Table 12-40. Central Resource SBL Fixed Parameters

Port	Port Direction	Port Priority	Buffer Size Bytes	Description
CSIa	WRITE	P3	1024 = 4DUs	CSIa output port
CSIb	WRITE	P2	1024 = 4DUs	CSIb output port
CCDC	WRITE	P4	1024 = 4DUs	CCDC output port
	READ	P1	512 = 2DUs	CCDC fault pixel correction input port
PREVIEW	WRITE	P9	1024 = 4DUs	PREVIEW output port
	READ	P13	1024 = 4DUs	PREVIEW input port CSI1 data input port
	READ	P0	1024 = 4DUs	PREVIEW dark-frame input port CCDC lens-shading compensation input port
RESIZER	WRITE	P5	1024 = 4DUs	RESIZER output line 1 port
	WRITE	P6	1024 = 4DUs	RESIZER output line 2 port
	WRITE	P7	1024 = 4DUs	RESIZER output line 3 port
	WRITE	P8	1024 = 4DUs	RESIZER output line 4 port
	READ	P12	1024 = 4DUs	RESIZER input port
H3A	WRITE	P10	512 = 2DUs	H3A output - AF port
	WRITE	P11	512 = 2DUs	H3A output - AE/AWB port
HIST	READ	P14	512 = 2DUs	HIST input port

12.4.9.2.2 Write-Buffer Logic (WBL) and Write Buffer

The central-resource SBL uses multiple WBL blocks to interface between the modules' write ports and the write-buffer memory.

- One WBL is instantiated for each module write port.
- Each WBL collects the module's write port output data and transfers the data to the write-buffer memory. Arbitration occurs between the WBL blocks to access the write-buffer memory.
- Each WBL is responsible for tracking all corresponding DUs in the write-buffer memory. There can be 2 or 4 DUs in the write buffer associated with a WBL.
- A WBL generates a command to memory when:
  - The write data crosses a DU boundary. At this point, the module starts filling a new DU. Also, the WBL generates a command to transfer the previous DU to memory.
  - An end-of-frame occurs. The DU (even if not full) is transferred to memory, and a command is issued.
  - An end-of-line occurs. The DU (even if not full) is transferred to the memory, and a command is issued.

12.4.9.2.3 Read Buffer Logic (RBL) and Read Buffer

The central-resource SBL uses multiple RBL blocks to interface between the modules' read ports and the read-buffer memory.

- One RBL is instantiated for each module read port.

- Each RBL is responsible for accepting input data from the read-buffer memory and for sending the input data to the read port of the corresponding module.
- Each RBL is responsible for tracking all corresponding DUs in the read-buffer memory. There can be 2 or 4 DUs in the read buffer associated with a RBL.
- Unlike the WBL, the RBL is not responsible for issuing the commands to memory; each individual module is responsible for doing this.

Two read ports are shared:

- Between the PREVIEW module dark frame and the CCDC lens-shading compensation input
- Between the PREVIEW module and the CSI1 module data read input

They can only be used by one module at a given time because there is no arbitration mechanism. Software is responsible of enabling only one of the 2 possible features attached to a port and to select the correct multiplexer configuration using the ISP_CTRL [27] SBL_SHARED_RPORTA and ISP_CTRL [28] SBL_SHARED_RPORTB registers.

12.4.9.2.4 Arbitration

The central-resource SBL arbitrates between module requests, based on fixed priorities. Read and write requests are arbitrated independently.

A total of 8 commands can be active at a time. When a new slot opens, the highest-priority transfer enters the command queue.

RBLs/WBLs are ensured access to the read/write buffer memories at least once every other cycle.

Note: The hardware uses burst. All bursts are precise; the total number of transfers in the burst is known at the start of the burst. All writes are posted.

12.4.9.3 Memories

The central-resource shared-buffer module has three memories:

- READ BUFFER: Shared by all modules
- WRITE BUFFER 0: Used only by the resizer module
- WRITE BUFFER 1: Shared by all modules except RESIZER

12.4.9.4 Debug Registers

Some registers are available for debugging data transfers between a module and external memory. The read-only debug registers are divided into two categories:

- 8 global request registers to capture information about any of the 52 module request registers at a given time. Each register provides information about one DU. The number 16 corresponds to the maximum number of outstanding requests, according to the protocol. Each global request register provides the following information:
  - Individual module register command number. For modules with 2 individual requestors, this field displays either 0 or 1. For modules with 4 individual requestors, this field displays 0, 1, 2, or 3.
  - Source or destination module
  - Data-flow direction
  - Valid bit
- 52 individual module request registers (read or write information). Each register provides information about one DU. The number of request registers assigned (2 or 4) depends on memory bandwidth requirements; modules with lower requirements are assigned 2 request registers, while modules with higher bandwidths are assigned 4 request registers.

Table 12-41. Central Resource SBL Number of Request Registers

Port	RD/WR	Nb Request Registers	Description
CSIa	WRITE	4 WR request registers	CSIa output
CSIb	WRITE	4 WR request registers	CSIb output
CCDC	WRITE	4 WR request registers	CCDC output
	READ	2 RD request registers	CCDC fault pixel correction input
PREVIEW	WRITE	4 WR request registers	PREVIEW output
	READ	4 RD request registers	PREVIEW input
	READ	4 RD request registers	PREVIEW dark-frame input
RESIZER	WRITE	4 WR request registers	RESIZER output line 1
	WRITE	4 WR request registers	RESIZER output line 2
	WRITE	4 WR request registers	RESIZER output line 3
	WRITE	4 WR request registers	RESIZER output line 4
	READ	4 RD request registers	RESIZER input port
H3A	WRITE	2 WR request registers	H3A output - AF port
	WRITE	2 WR request registers	H3A output - AE/AWB port
HIST	READ	2 RD request registers	HIST input

Each write-request register provides the following information:

- Current byte count: Number of bytes in the block of data for this command, up to 256 bytes
- Data ready: Block of data confirmed by the module
- Data sent: Data sent to the destination and waiting for status
- Upper 20 bits of the address

Each read-request register provides the following information:

- Valid: Read requested from the module
- Waiting for data: Command accepted from the source
- Data available: Data received from the source and can be read by the module
- Byte count requested: Up to 256 bytes
- Upper 20 bits of the address

12.4.10 Circular Buffer

The circular buffer (CBUFF) maps a virtual space to a physical space by address translation. It does not change the data or store it locally.

Note: Addresses can be further translated by the MMU.

12.4.10.1 Feature List

The features are listed below:

- Two independent circular buffers (CBUFF0 and CBUFF1)
- Linear address space (virtual) mapped into a circular space (physical)
- Fully transparent for accesses out of the configured virtual space
- Maximum physical buffer size of 16x16M bytes:
  - Physical space is composed by 2,4,8 or 16 windows
  - Maximum allowed window size is 16M bytes
- Support for multiline write patterns
  - Used together with the resizer in upscale mode. Each buffer must contain at least ceil (vertical

zoom factor) images lines

- Support of 2D addressing modes
- Strong error detection mechanisms
- Buffer addresses are 64-bit aligned, but window fill level managing is byte accurate.
- Read and write accesses are supported.
- Bandwidth Control Feedback loop connected to the CSI1 receiver flexible input

12.4.10.2 Interrupts

All events generated (see Table 12-18 for details) by the circular buffer are merged into a single event at Camera ISP level. This event can be mapped to MPU SS by enabling the ISP_IRQ0ENABLE [21] CBUFF_IRQ bit or to IVA2.2 by enabling the ISP_IRQ1ENABLE [21] CBUFF_IRQ bit.

12.4.10.3 Functional Description

The CBUFF module maps a virtual address space to a physical space called circular buffer. The CBUFF module can handle up to 2 independent circular buffers CBUFF0 and CBUFF1.

This section gives an overview of typical uses of the module.

12.4.10.3.1 Bandwidth Control Feedback Loop

The bandwidth control feedback (BCF) feature can be used in memory to memory operation when the Camera ISP reads data through the CSIb receivers flexible input.

At a given time a certain number of physical buffers are available for the processor. Depending, if the circular buffer is configured in read or write mode, the processor can, respectively, write or read those physical buffers.

When the number of physical buffers to be processed by the processor increases, the number of physical buffers available to the Camera ISP decreases. When the number of physical buffers available for the Camera ISP becomes too low, the corresponding BCF signal is asserted. This signal can be used to stall the data read flow.

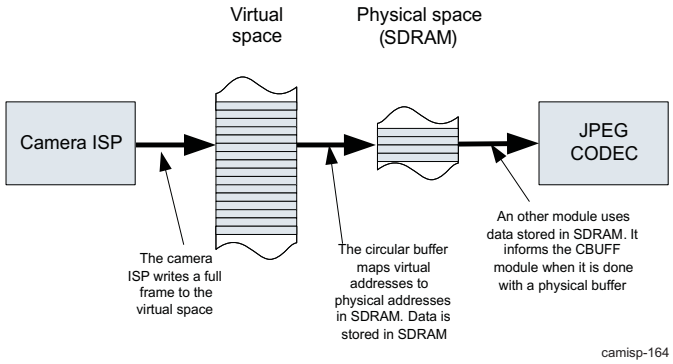
The Camera ISP supports two mechanisms to reduce the processing speed in memory to memory operation:

- Add a fixed delay between memory read requests. Delays can be added:
  - At SBL level for all image data read ports (histogram, preview, CSI1 and resizer)
  - At CSI1 module level by slowing down the video port clock
- Stall memory reads based on the level of the circular buffer. That is the purpose of the BCF feature described in this section.

12.4.10.3.1.1 Single Slice Mode

The Camera ISP writes data with an incremental addressing scheme to the virtual space. The physical buffer is smaller than the virtual space. Therefore, the physical buffer locations is read or written multiple times. See Figure 12-93.

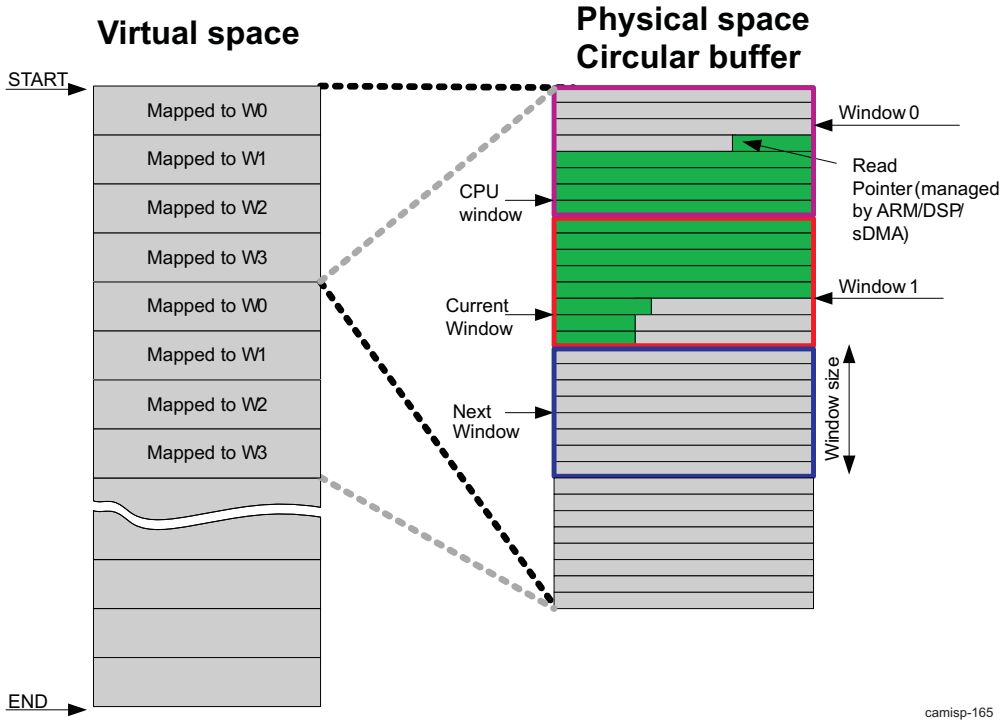
Figure 12-93. Single Slice Buffer (Write Mode)



Physically, this circular buffer is typically in the SDRAM. The virtual space is defined by a start and an end address. The physical buffer is defined by a start address, a window size, and a window count. It is contiguous in memory.

Figure 12-94 shows the buffer organization for a 4-window buffer.

Figure 12-94. Single Slice Buffer Example (Write Mode)



The CBUFF module can manage two independent circular buffers in single slice mode.

In case of memory-to-memory operation, the module using the data (that is, JPEG CODEC) written by the ISP into the circular buffer may be slower than the Camera ISP. That is especially the case when the resizer is used for upscaling.

In this section, it is assumed that the data to be read by the Camera ISP is a full frame completely present in physical memory. Sliced buffer capability is only used the buffer data at the output of the Camera ISP. Therefore, the circular buffer is configured in write mode.

The basic idea is to stall the read data flow at CSIb flexible input level. This is done by blocking the response phase of the CSIb interconnect read master port. Blocking the response phase rather than the request phase allows data prefetch at SBL level.

When the count of full windows is greater than or equal to the value defined in the CBUFFx_CTRL[7:4] BCF register, the data flow between the SBL buffer and the CSI1 data read port is stalled. Data transfers resume when the full window count falls below the defined limit.

Stalling the data flow at this level does not prevent the SBL from fetching data until its buffers are full. Therefore, when data transfer resumes, data are available quickly at the CSI1 read port level, and ISP processing can resume.

Due to SBL buffering, some data may be sent out to the circular buffer when the stall command is activated. Firmware must allocate enough space in the circular buffer to avoid an overflow. The amount of buffered data depends on the ISP configuration:

- The Camera ISP pipeline can contain up to 10 pixels that are sent to the SBL after stall control is activated.
- The SBL can store up to four DUs at the output of the PREVIEW module. Therefore, when the circular buffer is filled from the PREVIEW module there must be enough space to store the extra 1K byte of data.
- The data path between the resizer and preview module goes though the SBL and data is internally buffered. The SBL can buffer up to four DUs between the PREVIEW and RESIZER module. In other words, when the input of the PREVIEW module is stalled, the RESIZER receives up to 2K bytes before it stalls.
- The SBL can store up to four DUs for each of the RESIZERs write ports. The count of ports used by the resizer depends on the vertical scaling factor; it uses ceil(vscale) ports. Each port writes into one image line and can buffer up to 1K byte.
- Depending on the resizing factor, the next write window may already contain up to three full video lines.

The largest amount of buffering is required when all of the following occur:

- The PREVIEW module is used.
- The RESIZER is used with a rescaling factor of 4x in the horizontal and vertical direction.
- The window size is not a multiple of the vertical up scaling factor.

Example #IMPLIED. Example of 4x Digital Zoom Use:

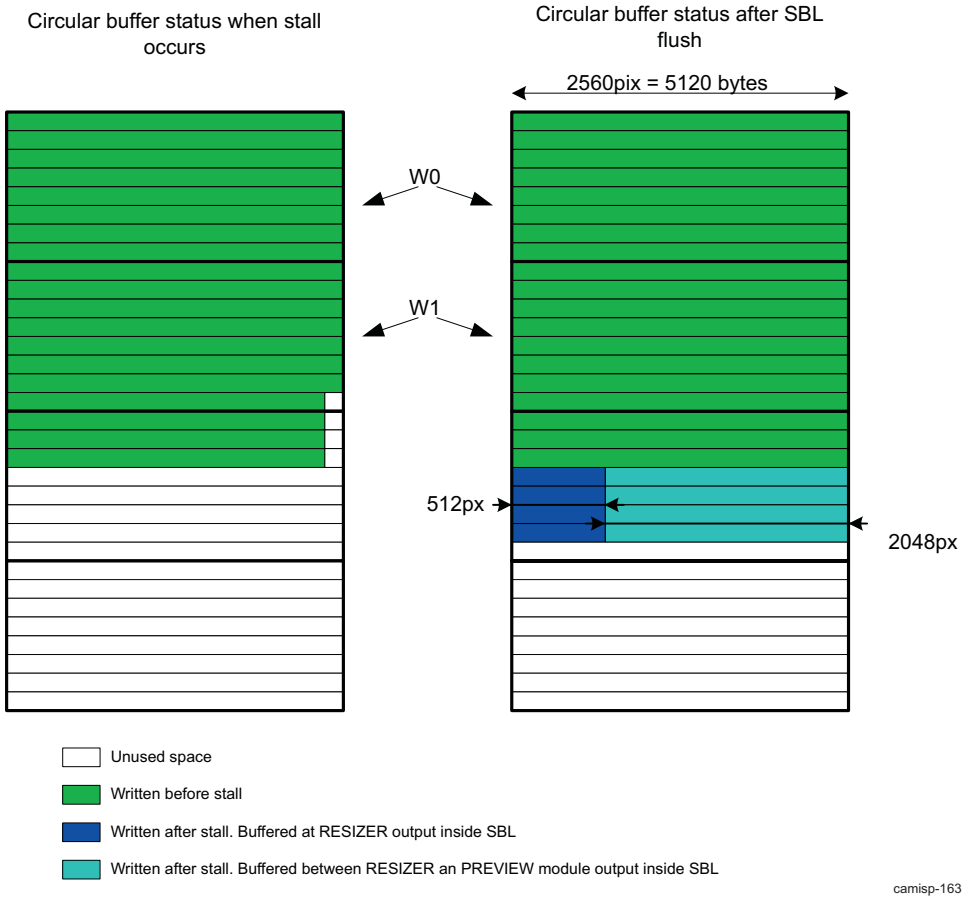
ThIs example shows how to calculate the minimum required window count when this feature is used.

It is assumed that the Camera ISP reads a VGA RAW8 image from memory, processes it in the preview,and scales it up to 5M pix (2560x1920). The circular buffer is used for temporary storage between the JPEG CODEC and the Camera ISP. Each window is configured to contain 8 video lines and the stall command is issued when at least two windows of the circular buffer are full.

Because the circular buffer window size is a multiple of the vertical upscaling factor, the next write window is empty when the stall command is issued. However, up to 1K byte = 512 pixels are stored for each write port of the resizer. This data is written as a 512x4 pixel bloc to the circular buffer by the SBL. Between the PREVIEW and RESIZER up to 1kbyte = 512 pixels are stored. This data expands to a block of 512x4x4 pixels written to the circular buffer by the SBL. In other words, up to 7 lines of the circular buffer may be filled after the stall command is issued. A minimum of four windows must be allocated to avoid an overflow of the circular buffer.

Example #IMPLIED. Example of 4x Digital Zoom Use: (continued)

Figure 12-95. Circular Buffer Control Feedback Loop Example



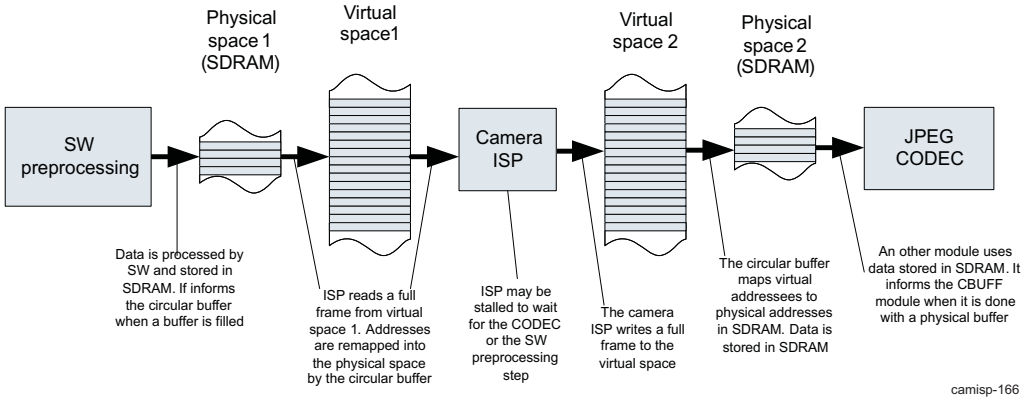
The CODEC reads data from the circular buffer and flag windows as done. In this example, the stall command is released when the CODEC has completed processing W0 and W1. The Camera ISP resumes writing into W3.

Note: Software can control the minimum full window count to issue the stall command using the CBUFFx_CTRL[7:4] BCF register. Setting this value to 1 or 2 in this example may negatively affect the performance, because only up to two or three windows can be used.

12.4.10.3.1.2 Extended Slice Mode

In extended slice mode, both circular buffers managed by the CBUFF module are used together. One provides address translation for the read data flow, and the other for a write data flow.

Figure 12-96. Extended Slice Buffer Example



The Camera ISP prefetches data and buffers it in the SBL. The CSIb flexible input pre-buffers up to 1K byte of data. In other words, the stall control for the CSIb interconnect read master port must be triggered in advance. Otherwise, the Camera ISP may prefetch invalid data.

The last moment when the CSIb interconnect read master port can be safely stalled is when the physical read buffer still contains 1K byte of valid data. The Camera ISP may or may not then read the remaining data into its internal SBL buffer.

The minimum amount of buffered data depends on the circular buffer configuration and the used data format.

When the processor filling the read buffer has no more data to write, the BCF feature must be disabled (clear ISP_CTRL [23:22] CBUFF0_BCF_CTRL or ISP_CTRL[25:24] CBUFF1_BCF_CTRL) for the CBUFF attached to the read data flow. This allows the Camera ISP to prefetch the remaining data from the buffer without stalling.

12.4.10.3.2 Window Management

This section explains the internal address remapping and windows management algorithm. Internally the module maintains some variables in addition to the configuration registers.

The module manages two circular buffers in parallel. Those are called CBUFF0 and CBUFF1.

Table 12-42. Internal Variables

Quantity	Description
CWx	Current window index for buffer x (x = 0, 1). Possible values are 0 to allowed window count. The current value can be read using the CBUFFx_STATUS[11:8] CW register.
NWx	Next window index for buffer x (x = 0, 1). Possible values are 0 to allowed window count. The current value can be read using the CBUFFx_STATUS[19:16] NW register.
CPUWx	Window in the physical buffer that can be accessed by the CPU. Possible values are 0 to allowed window count. The current value can be read using the CBUFFx_STATUS[3:0] CPUW register.
FCOx	Start address, in the virtual space, of the current window. This is an internal quantity that cannot be accessed by software.

Table 12-42. Internal Variables (continued)

Quantity	Description
OFFSETy	This is an internal quantity that cannot be accessed by software. y = 0: Address offset used when the current window of buffer 0 is accessed y = 1: Address offset used when the next window of buffer 0 is accessed y = 2: Address offset used when the current window of buffer 1 is accessed y = 3: Address offset used when the next window of buffer 1 is accessed
LEVELy	This is an internal quantity that cannot be accessed by software. y = 0: Amount of data, in bytes, read or written in the current window of buffer 0 y = 1: Amount of data, in bytes, read or written in the next window of buffer 0 y = 2: Amount of data, in bytes, read or written in the current window of buffer 1 y = 3: Amount of data, in bytes, read or written in the next window of buffer 1

12.4.10.3.2.1 Startup

The status of a circular buffer (CBUFF0 or CBUFF1) is reset when it is disabled. This does not affect the configuration registers or the CBUFF_IRQSTATUS register. Table 12-43 shows the internal state after reset.

Table 12-43. Internal State After Reset

Quantity	Description
CWx	0
NWx	1
CPUWx	0
FCOx	CBUFFx_START
OFFSET0	0
OFSFET1	0
OFFSET2,3	0
LEVELy	0

12.4.10.3.2.2 Access Identification

For each access to the virtual space, the CBUFF module first checks the address (ADDR) to classify the transaction into one of the categories listed in Table 12-44.

Table 12-44. Address Identification

ID	Label	Condition
0	CW_CBUFF0	CBUFFx_CTRL[0] ENABLE = 1 and ADDR>=FCO0 and ADDRFCO0 + CBUFFx_WINDOWSIZE and ADDR= CBUFFx_END (x = 0)
1	NW_CBUFF0	CBUFFx_CTRL[0] ENABLE=1 (x = 0) and ADDR>=FCO0 + CBUFFx_WINDOWSIZE and ADDRFCO0 + 2*CBUFFx_WINDOWSIZE and ADDR= CBUFFx_END (x = 0)
2	CW_CBUFF1	CBUFFx_CTRL[0] ENABLE=1 and ADDR>=FCO1 and ADDRFCO1 + CBUFFx_WINDOWSIZE and ADDR= CBUFFx_END (x = 1)
3	NW_CBUFF1	CBUFFx_CTRL[0] ENABLE=1 and ADDR>=FCO1 + CBUFFx_WINDOWSIZE and ADDRFCO1 + 2*CBUFFx_WINDOWSIZE and ADDR= CBUFFx_END (x = 1)
4	ERR_CBUFF0	CBUFFx_CTRL[0] ENABLE=1 and ADDR>= CBUFFx_START and ADDR= CBUFFx_END (x = 0)

Table 12-44. Address Identification (continued)

ID	Label	Condition
5	ERR_CBUFF1	CBUFFx_CTRL[0] ENABLE=1 and ADDR>= CBUFFx_START and ADDR= CBUFFx_END (x = 1)
6	TRANSPARENT	Always true

Lower IDs correspond to higher priorities in case multiple conditions are true. For example, when the current virtual window of the circular buffer 0 is accessed, at least the tests for categories CW_CBUFF0 and ERR_CBUFF0 are true. The final category is CW_CBUFF0, because it has a higher priority.

Further processing depends on the category:

- TRANSPARENT: Accesses flow through the module without changing its internal state or any translation.
- ERR_CBUFF0 and ERR_CBUFF1: The module goes into error state for the concerned buffer (CBUFF0 or CBUFF1) and set the CBUFF_IRQSTATUS[1] IRQ_CBUFF0_INVALID bit (or CBUFF_IRQSTATUS[4] IRQ_CBUFF1_INVALID). When the module is in error state for CBUFFx, all accesses to that buffer are cancelled. In other words, any access that has an address between CBUFFx_START and CBUFFx_END (x = 0) is not transmitted to the interconnect. There are two ways to leave the error state:
  - Hardware reset
  - Disable and re-enable the buffer (CBUFF0 or CBUFF1) in error state

  Accesses outside of the virtual space from the circular buffer in error state are not affected.
- CW_CBUFF0, NW_CBUFF0, CW_CBUFF1 and NW_CBUFF1: The internal state is updated and address translation is performed when the performed access type (read or write) is compatible with the current mode (read or write). Otherwise, an IRQ_CBUFFx_INVALID event is set and CBUFFx goes into the error state.

12.4.10.3.2.3 Address Translation

An offset is selected depending on the access category (see Section 12.4.10.3.2.1, *Startup*) and the internal state of the accessed buffer. Table 12-45 lists possible cases.

Table 12-45. Address Translation

Condition	Address Translation
CBUFFx_CTRL[0] ENABLE=1 and ADDR>= CBUFFx_START and ADDR= CBUFFx_END and CBUFF0 in error state (x = 0)	Access cancelled
CBUFFx_CTRL[0] ENABLE=1 and ADDR>= CBUFFx_START and ADDR= CBUFFx_END and CBUFF1 in error state (x = 1)	Access cancelled
Category = CW_CBUFF0	ADDROUT = ADDRIN-OFFSET0
Category = NW_CBUFF0	ADDROUT = ADDRIN-OFFSET1
Category = CW_CBUFF1	ADDROUT = ADDRIN-OFFSET2
Category = NW_CBUFF1	ADDROUT = ADDRIN-OFFSET3
Category = ERR_CBUFF0	Access cancelled
Category = ERR_CBUFF1	Access cancelled
Category = TRANSPARENT	ADDROUT = ADDRIN

12.4.10.3.2.4 Window Fill Level

Each time an access is performed into an active window (CW_CBUFF0, NW_CBUFF0, CW_CBUFF1 or NW_CBUFF1) the window level is updated. The corresponding LEVELy is incremented according to the BYTEEN input of the interconnect port. All possible BYTEEN patterns are supported. Table 12-46 shows some examples. The basic idea is to count the number of ones in the BYTEEN input.

Table 12-46. Window Level Increment

BYTEEN	LEVELy Increment	Comment
0x00	+0	No access
0x01	+1	8 bit access
0x02	+1	8 bit access
0x03	+2	16 bit access
...	...	...
0x07	+3	24 bit access
...	...	...
0x0F	+4	32 bit access
...	...	...
0xF0	+4	32 bit access
...	...	...
0xFF	+8	64 bit access

The window level is compared to CBUFFx_THRESHOLD. As listed in Table 12-47, the following situations may occur:

Table 12-47. Window Level Comparison

Condition	Description
LEVEL0>= CBUFFx_THRESHOLD (x = 0)	Current window of buffer 0 full. Internal window indexes, levels, and offsets are updated.
LEVEL1>= CBUFFx_THRESHOLD (x = 0)	Next window of buffer 0 full. An IRQ_CBUFF0_INVALID error event is set and CBUFF0 enters the error state.
LEVEL2>= CBUFFx_THRESHOLD (x = 1)	Current window of buffer 1 full
LEVEL3>= CBUFFx_THRESHOLD (x = 1)	Next window of buffer 1 full. An IRQ_CBUFF1_INVALID error event is set and CBUFF1 enters the error state.

12.4.10.3.2.5 Window Pointer and Offset Update

When the current window of a circular buffer (CBUFFx, x = 0 or 1) is full:

- The "next window" becomes the "current window"
  - CWx <- NW
  - LEVEL(2*x) <- LEVEL (2*x+1)
  - OFFSET(2*x) <- OFFSET(2*x+1)
  - FCOx - FCOx + CBUFFx_WINDOWSIZE
- A new "next window" if opened. The update is done in a circular manner: the first window in the physical space is reused after the last one
  - NW <- (NW+1) modulo WC
  - LEVEL(2*x+1) <- 0
  - When the "next window" is moved from the last buffer to the first:
    - OFFSET(2*x+1) - OFFSET(2*x+1) + WCx * CBUFFx_WINDOWSIZE

Note: WC is the window count defined by the CBUFFx_CTRL[9:8] WCOUNT register.

12.4.10.3.3 CPU Interaction

The CBUFF module sets an IRQ_CBUFFx_READY event to inform the CPU that it can access the CPUx window in the physical buffer. The CBUFF module cannot monitor CPU accesses to the physical buffer. The CPU must indicate when it has completed the processing of the CPUWx window by writing the CBUFFx_CTRL[2] DONE bit. This increments the CPU window index CPUWx by one modulo the window count.

The behavior depends on whether read or write mode has been selected using the CBUFFx_CTRL[1] RWMODE bit.

Note: Some details of this feature are not available in the public domain.

12.4.11 MMU Logic

12.4.11.1 MMU Features

The Camera ISP MMU contains a translation lookaside buffer (TLB) that holds translations and properties for current pages. This TLB can be managed statically through the configuration slave port, or by the internal hardware table-walking logic (TWL), which can autonomously traverse the page table on a TLB miss. The TWL can be enabled or disabled (MMU_CNTL [2] TWLENABLE).

On a TLB miss, the initiator is stalled until a valid address translation is found. If no valid translation is found, a translation fault interrupt is generated to the processor. It is a nonrecoverable situation, which leads to the Camera ISP being reset by the processor software.

The MMU provides up to 4G bytes of virtual memory.

12.4.11.2 MMU Functional Description

For a detailed description of the MMU, see Chapter 8, *Memory Management Units*.

Note: In the Camera ISP MMU, the endianess feature is available for write, but conversion for read is not possible.

12.5 Camera ISP Basic Programming Model

12.5.1 Programming the CSI1 Receiver

This section describes the programming model of the CSI1 receiver.

12.5.1.1 Hardware Setup/Initialization

This section discusses the configuration of the CSI1 receiver required before image capture can begin.

12.5.1.1.1 Reset Behavior

On hardware or software reset of the Camera ISP, all registers in the CSI1 receiver are reset to their reset values.

12.5.1.2 Event and Status Checking

When an event occurs, the corresponding bit in the CSI1_IRQSTATUS register is set. Each event can be individually masked using the CSI1_IRQENABLE register . Masked events are not transmitted to the interrupt lane, but the CSI1_IRQSTATUS register is updated.

Events transmitted to the interrupt lane can be mapped to the ARM or DSP by unmasking the ISP_CSIB bit in the ISP_IRQxENABLE (x = 0, 1). Depending on whether the event is mapped to the ARM or DSP register, to clear an event, the following actions are required:

- Clear the event at the CSI1 receiver level by writing 1 to the corresponding bit in the CSI1_IRQSTATUS register .
- Clear the event at ISP level by writing 1 to the corresponding bit ISP_CSIB in the ISP_IRQxENABLE (x = 0, 1) register.

12.5.1.3 Register Accessibility During Frame Processing

There are two types of register accesses in the CSI1 receiver:

- Shadowed registers:
  - These registers/fields can be read and written (if the field is writable) at any time. However, written values take effect only at the start of a frame. Reads return the most recent write, even though the settings are not used until the next start of frame.
  - The shadowed registers are:
    - CSI1_CTRL1
    - CSI1_CODE
    - CSI1_STAT_START
    - CSI1_STAT_SIZE
    - CSI1_SOF_ADDR
    - CSI1_EOF_ADDR
    - CSI1_DAT_SIZE
    - CSI1_DAT_PING_ADDR
    - CSI1_DAT_PONG_ADDR
    - CSI1_DAT_OFST
- Busy-locked registers:
  - These registers/fields must not be written if the module is busy.
  - All register fields not listed as shadowed are busy-writable registers.

12.5.1.4 Enable/Disable the Hardware

The CSI1 receiver is globally controlled by the CSI1_CTRL register. The bit fields in this register must not be modified when the CSI1 interface is active (except CSI1_CTRL [0] IF_EN):

- To activate the CSI1 interface:
  - CSI1_CTRL [0] IF_EN = 0x1
  - Data acquisition starts on the following FSC synchronization code. Writing CSI1_CTRL [0] IF_EN = 0x1 resets the output FIFO of the module; the reset is caused by the 0-to-1 edge transition.
- To disable the CSI1 interface:
  - CSI1_CTRL [0] IF_EN = 0x0
  - The interface is disabled immediately if CSI1_CTRL [3] FRAME = 0x0.
  - If CSI1_CTRL [3] FRAME = 0x1 the interface is disabled after the FEC synchronization code is received.

12.5.1.5 Complex I/O Control

CAUTION

When using the CSIb receiver, the CONTROL.CONTROL_CSIRXFE[12] CSIb_PWRDN bit must be set to 1 to enable the transceiver, and the CONTROL.CONTROL_CSIRXFE[13] CSIb_RESET bit must be set to 1 to disable the reset before starting the transfer.

The CSI1_CTRL [2] IO_OUT_SEL bit selects the output mode of the CSI1 complex I/O cell, which can be serial or parallel. Frame-shift errors are not detected when parallel mode is selected.

The CSI1_CTRL [10] INV bit selects the clock edge used to sample data:

- If CSI1_CTRL [10] INV = 0x0, use the rising edge.
- If CSI1_CTRL [10] INV = 0x1, use the falling edge.
- The polarity is changed only if the SCM-associated bit is also changed: CONTROL.CONTROL_CSIRXFE CSIx_INV.

12.5.1.6 Debug Mode

Use the CSI1_CTRL [7] DBG_EN bit to enable the debug mode:

- During debug mode, the input comes from the CSI1_DBG register, not from the CSI1 physical interface. The full CSI1 receiver functionality can be debugged in debug mode. Full 32-bit values must always be written to the CSI1_DBG register.
- The following bits have no effect during debug mode:
  - CSI1_CTRL [0] IF_EN
  - CSI1_CTRL [1] PHY_SEL
  - CSI1_CTRL [2] IO_OUT_SEL
- The following examples apply to the CSI1_DBG register:
  - Synchronization codes: CSI1_DBG = 0xFF000000
  - To send the RAW12 pixels 0x673, 0x452, 0x01d, 0xefc, 0xab0, 0x891, 0x326, and 0x547, write CSI1_DBG = 0x01234567 followed by CSI1_DBG = and 0x89abcdef CSI1_DBG = 0x76543210.

Note: Each write to the CSI1_DBG register sends a full 32-bit word through the CSI1 receiver hardware. When 8- or 16-bit writes are performed to the register, the previous 32-bit value is merged with the newly written one. When the driver writes, for example, 0x01234567 followed by 0x000000FF with BYTEEN = 0x1 (this write from the MPU subsystem informs that only 8 bits are written), the CSI1 receiver pipeline gets 0x01234567 followed by 0x012345FF.

12.5.1.7 Video Port

- Set the video-port output frequency:
  - CSI1_CTRL [9:8] VP_OUT_CTRL
  - The video-port output frequency varies from CAM_ICLK to CAM_ICLK/4 MHz. CAM_ICLK is the CSI1 receiver functional and interface clock. The reset value selects CAM_ICLK/2.
- CSI1_CTRL [11] VP_ONLY_EN:
  - Controls whether the video-port output is the only output interface enabled and applies for all channels.
  - When CSI1_CTRL [11] VP_ONLY_EN = 0x1, the data are output only to the video port; the interface master port is not used. The two parts of the frame (embedded data and pixel data) are output to the video port (instead of pixel data to video port and embedded data to interconnect).
    - The video port outputs the embedded data defined by the CSI1_STAT_START and CSI1_STAT_SIZE registers without decompression.

- The video port outputs the pixel data defined by the CSI1_DAT_START and CSI1_DAT_SIZE registers.

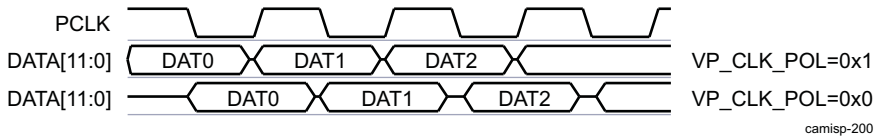
– The pixel data are decompressed according to the settings of the CSI1_CTRL1 [7:3] FORMAT bit field. Table 12-48 summarizes the behavior of the video port as a function of the CSI1_CTRL [7:3] FORMAT bit field.

Table 12-48. CSI1_CTRL1[7:3] FORMAT and CSI1_CTRL[11] VP_ONLY_EN = 1 Settings

CSI1_CTRL1 [7:3] Format	Video-Port Behavior
YUV422 + VP or RAW8 + VP	The incoming data are 8 bits. The pixel data are not compressed. The embedded data are transmitted to the CCDC module on 8 bits (DATA[7:0]). The pixel data are not decompressed. The pixel data are transmitted to the CCDC module on 8 bits (DATA[7:0]).
RAW12 + VP	The incoming data are 12 bits. The pixel data are not compressed. The embedded data are transmitted to the CCDC module on 12 bits (DATA[11:0]). The pixel data are reconstructed in the receiver. The pixel data are transmitted to the CCDC module on 12 bits (DATA[11:0]). The pixel data are not decompressed.

- Control the video-port pixel clock polarity:
  – If CSI1_CTRL [12] VP_CLK_POL = 0x0, the CSI1 receiver module writes the data on the video port on the pixel-clock falling edge. The module connected to the VP samples the data on the pixel clock rising edge.
  – If CSI1_CTRL [12] VP_CLK_POL = 0x1, the CSI1 receiver module writes the data on the video port on the pixel-clock raising edge. The module connected to the VP samples the data on the pixel clock falling edge. Figure 12-97 shows the CSI1_CTRL .VP_CLK_POL settings.

Figure 12-97. CSI1_CTRL.VP_CLK_POL Settings



12.5.1.8 Region of Interest

The CSI1_CTRL1 [1] REGION_EN bit enables the region-of-interest feature (SOF lines, pixel data, and EOF lines):

- If enabled, register settings set the position and size of each region; all data not in a region of interest are ignored.
- If disabled, all data in the frame are output. CSI1_CTRL1 [1] REGION_EN is set to 0x0 for a JPEG bitstream.

12.5.1.9 Destination Format

- Control the destination format:
  – The CSI1 receiver reformats received data to store it in memory or to send it to the video-preprocessing.
  – CSI1_CTRL [7:3] FORMAT controls destination-data format:
    - EXP8 = Data expansion to 8 bits, padding with zeros
    - EXP16 = Data expansion to 16 bits, padding with alpha or zeros CSI1_CTRL [15:8] ALPHA can be used to set an alpha value. For RGB444 + EXP16:
      - data_out [31:28] = ALPHA [3:0]
      - data_out [15:12] = ALPHA [3:0]
    - EXP32 = Data expansion to 32 bits, padding with alpha. CSI1_CTRL [15:8] ALPHA can be used to set an alpha value. For RGB888 + EXP32: data_out [31:24] = ALPHA [7:0]
    - FSP = False synchronization code protection decoding. Applies only to JPEG8 data format.
    - VP = Output to the video-preprocessing is enabled.

12.5.1.10 Frame Acquisition

- Program the number of frames that the CSI1 receiver acquires:
  - CSI1_CTRL1 [31:24] COUNT
  - Writes to the COUNT bit field are controlled by the CSI1_CTRL1 [16] COUNT_UNLOCK bit.
  - If COUNT = 0x0, the counter is free-running; frame acquisition lasts until disabled by the programmer. It is the default value.
  - COUNT controls the number of frames to acquire. The values range between 1 and 255. The COUNT value is decremented after each frame received. The software can read this value to acquire the number of frames that remain to be acquired.

After the correct number of frames is received, acquisition is automatically disabled (CSI1_CTRL1 [0] CHAN_EN = 0x0) and the COUNT_IRQ interrupt is triggered. The programmer can reenable the acquisition by resetting CSI1_CTRL1 [0] CHAN_EN to 0x1.

CSI1_CTRL1 [17] PING_PONG indicates whether the PING address (CSI1_DAT_PING_ADDR) or PONG address (CSI1_DAT_PONG_ADDR was used to store the pixel data of the last frame. After reset or after a 0-to-1 edge transition in CSI1_CTRL [0] IF_EN, the pixel data are written in the PING buffer and CSI1_CTRL1 [17] PING_PONG = 0x1 (PONG). After the first FEC synchronization code is received, the pixel data are written in the PONG buffer and CSI1_CTRL1 [17] PING_PONG = 0x0 (PING). CSI1_CTRL1 [17] PING_PONG toggles after every FEC synchronization code.

12.5.1.11 Synchronization Codes

The CSI1_CODE register enables overwriting of the default values: CSI1_CODE [11:8] FSC, CSI1_CODE [15:12] FEC, CSI1_CODE [3:0] LSC, and CSI1_CODE [7:4] LEC overwrite the 4 LSBs of the 32-bit synchronization codes. The default values should not be modified.

12.5.1.12 Status Data

The SOF and EOF status lines can be output to memory.

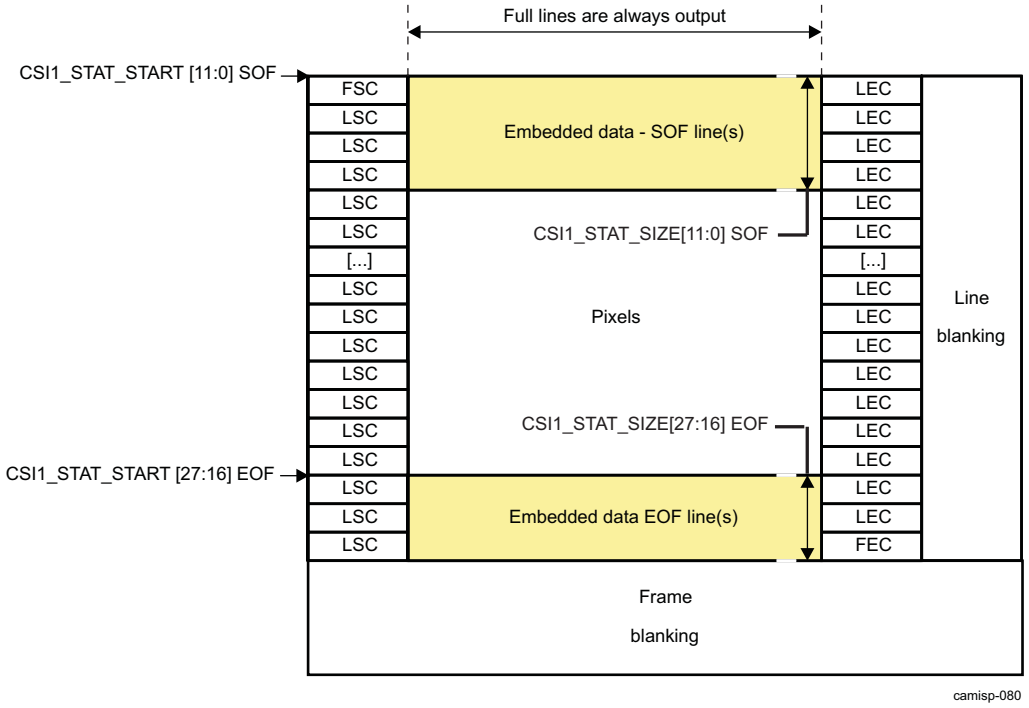
The SOF and EOF status lines always cover full lines. No register settings enable the setting of width.

The CSI1_STAT_START register enables the setting of the vertical start position of the SOF and EOF status lines. Because the SOF status line comes first in the CSI1 frame, CSI1_STAT_START [11:0] SOF = 0x0.

The CSI1_STAT_SIZE register enables the setting of the numbers of SOF and EOF status lines. If CSI1_STAT_SIZE [11:0] SOF = 0x0 and CSI1_STAT_SIZE [27:16] EOF = 0x0, no status data are output.

The SOF and EOF status lines and the pixel data must not overlap, but can be consecutive. Figure 12-98 shows the SOF and EOF region settings.

Figure 12-98. SOF and EOF Region Settings



The 32-bit destination address of the SOF status lines is set by the CSI1_SOF_ADDR register.

Note: The destination address must be aligned on a 32-byte boundary; the address 5 LSBs are ignored. The SOF lines are packed together at the destination address.

The 32-bit destination addresses of the EOF status lines are set by the CSI1_EOF_ADDR register.

Note: The destination address must be aligned on a 32-byte boundary; the address 5 LSBs are ignored. The EOF lines are packed together at the destination address.

Note: The CSI1 receiver does not modify the data in the SOF and EOF status lines. The data are received and written with no modifications.

12.5.1.13 Pixel Data Region

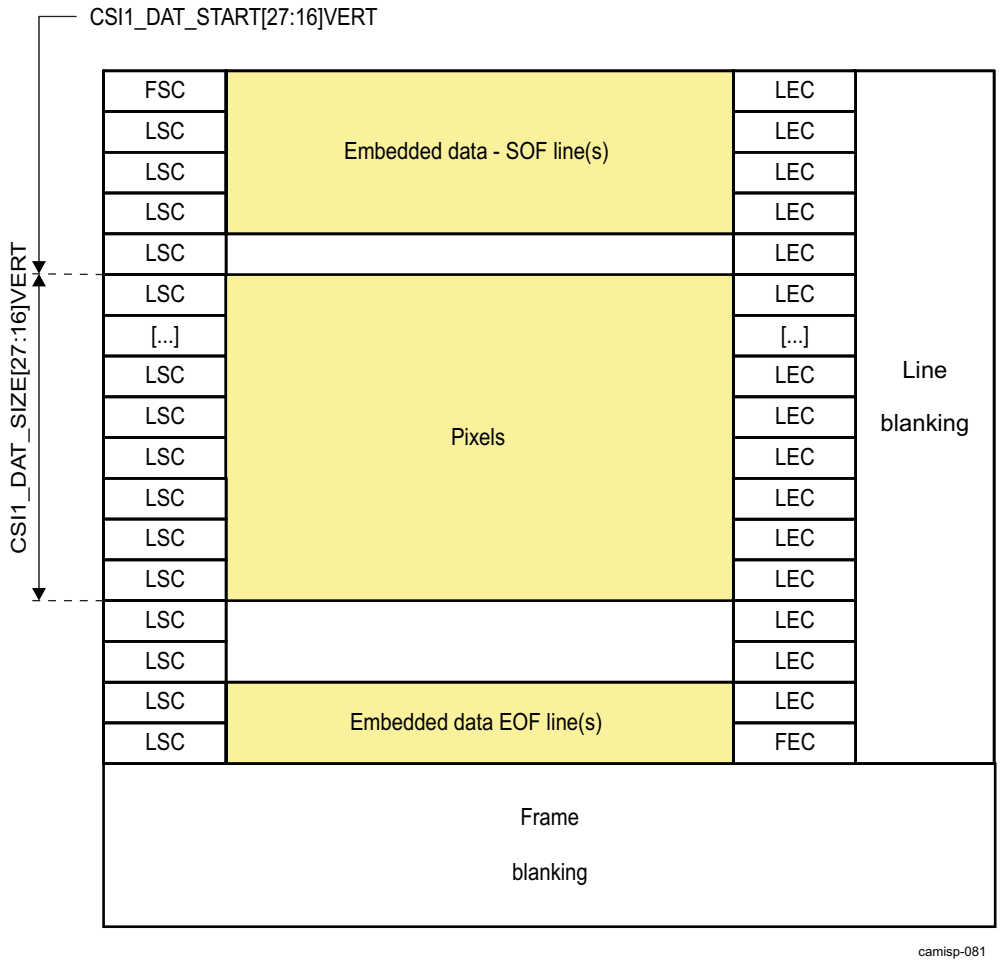
Pixel data can be output to memory or to the video-preprocessing hardware.

The pixel data region covers full lines. The CSI1_DAT_SIZE register sets the horizontal size of the pixel region. The vertical size is expressed in lines.

The CSI1_DAT_START register enables the setting of the vertical start position of the pixel data. The vertical start position is expressed in lines.

Figure 12-99 shows the pixel region settings.

Figure 12-99. Pixel Data Region Settings



The 32-bit destination addresses of the pixel data are set by the CSI1_DAT_PING_ADDR and CSI1_DAT_PONG_ADDR registers.

Note: The destination address must be aligned on a 32-byte boundary; the address 5 LSBs are ignored. The pixel data lines are packed together at the destination address.

It is possible to perform double-buffering (ping-ponging) at the destination by setting different addresses in the CSI1_DAT_PING_ADDR and CSI1_DAT_PONG_ADDR registers. It is possible to disable double-buffering by setting up the same address in both registers. The CSI1_CTRL1 [17] PING_PONG status bit must be used by the software to determine which address contains the latest frame.

A destination pitch controls the address jump between the address of the first pixel of the previous line and the address of the first pixel of the current line. The destination pitch is set in bytes with the CSI1_DAT_OFST register. It applies for CSI1_DAT_PING_ADDR and CSI1_DAT_PONG_ADDR .

Note: The destination pitch must be a multiple of 32 bytes; the address 5 LSBs are ignored.

The use of CSI1_DAT_OFST is limited to the following destination data formats:

- YUV422 little endian
- YUV422 big endian
- RGB444 + EXP16
- RGB565

- RGB888 + EXP32

For all other data formats, the CSI1_CTRL_DAT_OFST register is ignored (equivalent to CSI1_DAT_OFST = 0x0).

The destination data format is set with the CSI1_CTRL1 [7:3] FORMAT bit field.

For the PING frame:

- @Line0 = CSI1_DAT_PING_ADDR
- @Line1 = @Line0 + CSI1_DAT_OFST
- @Line2 = @Line1 + CSI1_DAT_OFST

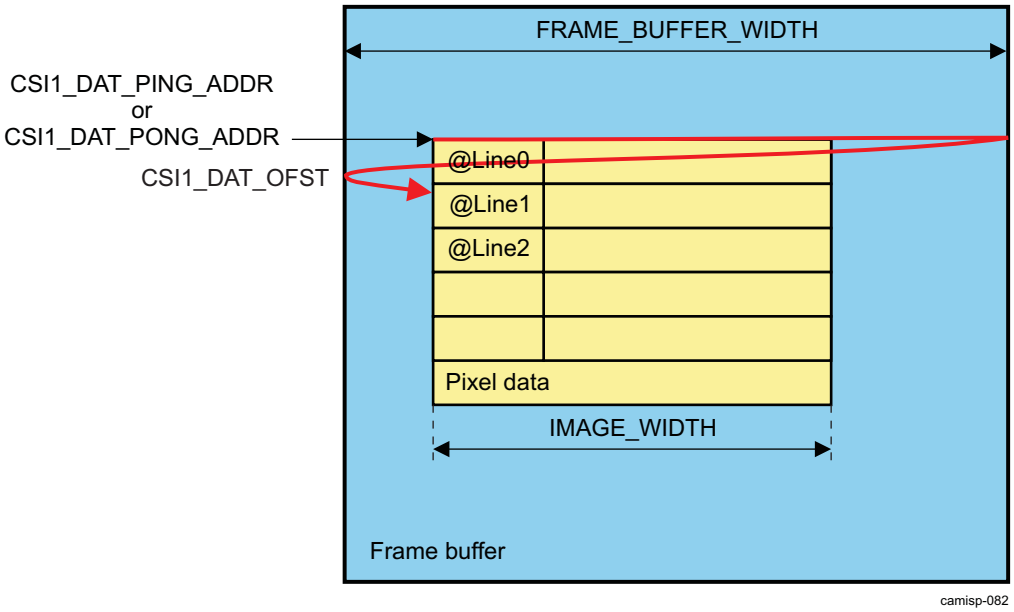
For the PONG frame:

- @Line0 = CSI1_DAT_PONG_ADDR
- @Line1 = @Line0 + CSI1_DAT_OFST
- @Line2 = @Line1 + CSI1_DAT_OFST

When CSI1_DAT_OFST = 0x0, the lines are written contiguously in memory. The destination pitch enables 2D transfers; it is required to write the pixel data directly in the frame buffer, for instance.

In such cases, CSI1_DAT_OFST = FRAME_BUFFER_WIDTH. Figure 12-100 shows the pixel data settings.

Figure 12-100. Pixel Data Destination Settings



12.5.1.14 Memory Read Channel

12.5.1.14.1 Write Data From Sensor to Memory

Data can be captured from the sensor using any logical channel. To keep the native data format, the channel format must be set to YUV422 little-endian format.

12.5.2 Programming the CSI2 Receiver

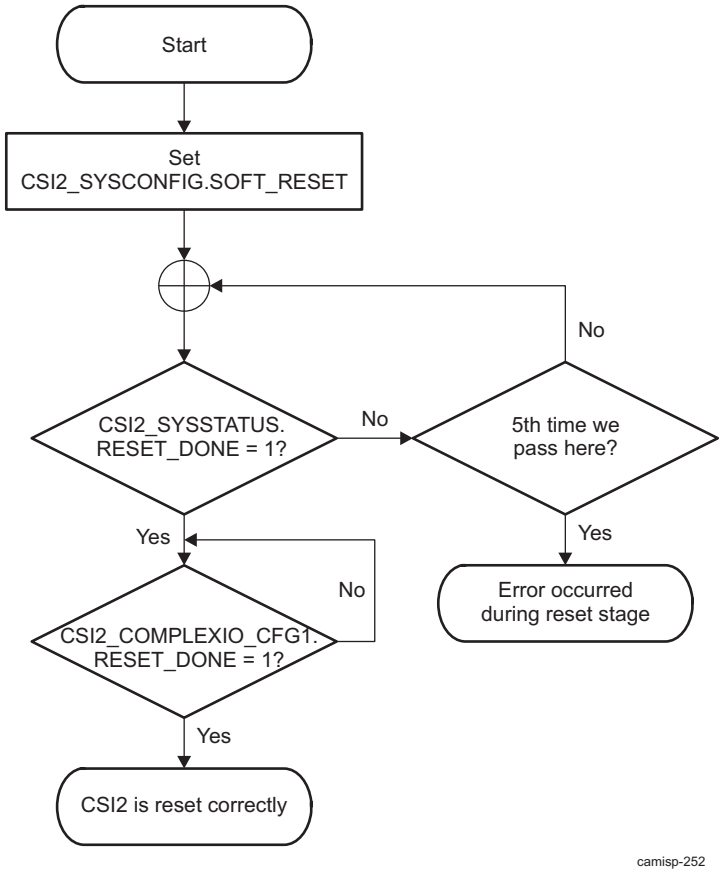
12.5.2.1 Enable CSI2 Interface

The CSI2 interface is clocked by CSI2_96_FCLK, which is enabled by setting the CM_FCLKEN_CAM[1] EN_CSI2 bit register to 1.

12.5.2.2 Reset Management

The CSI2 receiver accepts a general software reset, propagated throughout the hierarchy. This reset can be done to initialize the CSI2 receiver and the complex I/O and has the same effect as a hardware reset. Figure 12-101 shows how to reset the CSI2 globally.

Figure 12-101. CSI2 Receiver Global Reset Flow Chart



Note: CSI2_COMPLEXIO_CFG.RESET_DONE is set to 1 only after the CSI2 receiver, CSI2 complex I/O, and external camera sensor are initialized.

Note: Before setting the software reset bit to 1 in the CSI2_SYSCONFIG register, the user must have access to a CSI2 receiver register.

12.5.2.3 Enable Video/Picture Acquisition

To start a video/picture acquisition, perform the following steps:

1. Reset the CSI2 receiver module (see Section 12.5.2.2, *Reset Management*).
2. Configure the module power management: Set the CSI2_SYSCONFIG[13:12] MSTANDBY_MODE bit field to 0x2 so the module tries to enter smart-standby mode during the vertical blanking period. The CSI2_SYSCONFIG[0] AUTO_IDLE bit field keeps its reset value; by default, an automatic port clock gating strategy is applied based on port interface activity
3. Configure the interrupt generation as required using the CSI2_IRQSTATUS and CSI2_IRQENABLE registers. To enable context and/or complex I/O event reporting, enable the corresponding bit field in

the CSI2_IRQENABLE register. If the enable bit is at 0, logging is still effective if an event occurs but is not reported to a higher level.

4. Configure the complex I/O interrupt generation as required using the CSI2_COMPLEXIO1_IRQSTATUS and CSI2_COMPLEXIO1_IRQENABLE registers. If the enable bit is at 0, logging is still effective if an event occurs but is not reported to a higher level.
5. Start complex I/O: Set the CSI2_COMPLEXIO_CFG1[28:27] PWR_CMD bit to 0x1 to pass the complex I/O to ON state, then check that the state status reaches the ON state (CSI2_COMPLEXIO_CFG1[26:25] PWR_STATUS = 0x1).
6. Configure the complex I/O:
   - The complex I/O is fully functional with CSI2PHY_CFG1 set at its reset value.
   - CSI2PHY_CFG0 must be changed according to the data rate being used.
7. Set the CSI2_TIMING[15] FORCE_RX_MODE_IO1 bit to 1 so the RxMode and StopState FSM go to RxMode state. Users can also configure the delay for the FSM to return from RxMode to NoRxMode when all lines reach StopState.
8. Set the CSI2_CTRL[2] ECC_EN bit to 1 to activate ECC correction and error detection on short packets and packet headers. The ECC check corrects the packet if there is one error and generates an error if there is more than one error (unrecoverable error).
9. Start the CSI2 receiver by setting the CSI2_CTRL[0] IF_EN bit to 1.
10. Configure the different contexts to be used (up to eight):
    a. Link the context to a virtual channel and a data type (see Section 12.5.2.7, *Linking a Context to a Virtual Channel and a Data Type*).
    b. Set the CSI2_CTx_CTRL1[26:23] FEC_NUMBER bit field to 0x1 for a progressive video and to 0x2 for an interlaced video. For more information, see Section 12.4.3.7, *DMA Engine*.
    c. Keep the CSI2_CTx_CTRL1[15:8] COUNT bit field and the CSI2_CTx_CTRL1[4] COUNT_UNLOCK bit at their reset values (0x0 and 0, respectively) to capture an infinite number of frames (until the interface or the context is disabled).
    d. Set the CSI2_CTx_CTRL1[5] CS_EN bit to enable the CRC checksum on long packet payload. This allows detection of errors, but cannot correct errors like the ECC for header and short packet. On error detection, an event is triggered (the CSI2_CTx_IRQSTATUS[5] CS_IRQ bit).
    e. Configure the DMA engine for the current channel:
       - Configure the CSI2_CTx_DAT_PING_ADDR[31:5] ADDR bit field to set the ping frame address in memory.
       - Configure the CSI2_CTx_DAT_PONG_ADDR[31:5] ADDR bit field to set the pong frame address in memory. If not using a double-buffer mechanism, the ping and pong addresses must be equal so that all frames are stored at the same memory address).
       - Set the CSI2_CTx_DAT_OFST[15:5] OFST bit field to 0x0, so consecutive lines are stored consecutively in memory (image width and frame-buffer width are equal).

       Note: Addresses given for PING and PONG frames are virtual addresses. Address translations are made on the Camera MMU and on the Circular Buffer. For more information about the Camera MMU module, see Chapter 8, *Memory Management Units*.

    f. Keep the CSI2_CTx_CTRL3[29:16] ALPHA bit field at its reset value (0x0) for RGB padding.
11. Enable the contexts to be used by setting the CSI2_CTx_CTRL1[0] CTX_EN bit to 1.

12.5.2.4 Disable Video/Picture Acquisition

There are two ways to end picture acquisition:

- Disable the corresponding context by writing 0 to the CSI2_CTx_CTRL1[0] CTX_EN bit. This stops the acquisition for the current context. Other enabled contexts are still capturing frames and writing them in memory.
- Disable the CSI2 receiver interface by writing 0 to the CSI2_CTRL[0] IF_EN bit. This can have an immediate effect if the CSI2_CTRL[3] FRAME bit is 0, or it can be effective after all the enabled contexts receive the FEC if the CSI2_CTRL[3] FRAME bit is 1.

12.5.2.5 Capture a Finite Number of Frames

The CSI2 receiver module can be configured to capture a finite number of frames. To configure the CSI2 receiver in this mode, perform the following steps:

1. Write 1 to the CSI2_CTx_CTRL1[4] COUNT_UNLOCK bit to enable a write to the COUNT bit field.
2. Set the bit field to the number of frames the CSI2 receiver must capture. Valid values are 0 to 255; 0 is infinite capture and 1 to 255 defines the number of frames to capture.
3. Write 0 to the CSI2_CTx_CTRL1[4] COUNT_UNLOCK bit to disable a write to the COUNT bit field.

During frame capture, the COUNT bit field is decremented by 1 at each frame capture. The software reads the COUNT bit field to know how many frames still must be captured.

The COUNT bit can be updated during capture if the COUNT_UNLOCK is set to 1.

12.5.2.6 Configure a Periodic Event During Frame Acquisition

The CSI2 receiver can generate a periodic event. This line number is defined in the CSI2_CTx_CTRL3[15:0] LINE_NUMBER bit field. The event can be generated once or multiple times per frame, depending on the CSI2_CTx_CTRL1[1] LINE_MODULO bit value:

- If the LINE_MODULO bit = 0, the event is generated when the line number corresponding to the LINE_NUMBER bit field is received.
- If the LINE_MODULO bit = 1, the event is generated when the line number received corresponds to a multiple of the LINE_NUMBER value (LINE_NUMBER is used as a modulo).

12.5.2.7 Linking a Context to a Virtual Channel and a Data Type

The CSI2 receiver supports eight contexts and the CSI2 protocol defines four virtual channels. Therefore, a CSI2 receiver context can be associated with a virtual channel and a data type. Virtual channels are defined by a 2-bit field. Valid data types for the CSI2 receiver with their associated values are described in Table 12-49.

Table 12-49. CSI2 Receiver-Supported Data Types

Value	Data Type
0x0	Others
0x12	Embedded 8-bit nonimage data
0x18	YUV420 8-bit
0x19	YUV420 10-bit
0x1A	YUV420 8-bit legacy
0x1C	YUV420 8-bit + CSPS
0x1D	YUV420 10-bit + CSPS
0x1E	YUV422 8-bit
0x1F	YUV422 10-bit
0x22	RGB565
0x24	RGB888
0x28	RAW6
0x29	RWA7
0x2A	RAW8
0x2B	RAW10
0x2C	RAW12
0x2D	RAW14
0x33	RGB666 + EXP32_24
0x40	USER_DEFINED_8_BIT_DATA_TYPE_1
0x41	USER_DEFINED_8_BIT_DATA_TYPE_2
0x42	USER_DEFINED_8_BIT_DATA_TYPE_3

Table 12-49. CSI2 Receiver-Supported Data Types (continued)

Value	Data Type
0x43	USER_DEFINED_8_BIT_DATA_TYPE_4
0x44	USER_DEFINED_8_BIT_DATA_TYPE_5
0x45	USER_DEFINED_8_BIT_DATA_TYPE_6
0x46	USER_DEFINED_8_BIT_DATA_TYPE_7
0x47	USER_DEFINED_8_BIT_DATA_TYPE_8
0x68	RAW6 + EXP8
0x69	RAW7 + EXP8
0x80	USER_DEFINED_8_BIT_DATA_TYPE_1 + EXP8
0x81	USER_DEFINED_8_BIT_DATA_TYPE_2 + EXP8
0x82	USER_DEFINED_8_BIT_DATA_TYPE_3 + EXP8
0x83	USER_DEFINED_8_BIT_DATA_TYPE_4 + EXP8
0x84	USER_DEFINED_8_BIT_DATA_TYPE_5 + EXP8
0x85	USER_DEFINED_8_BIT_DATA_TYPE_6 + EXP8
0x86	USER_DEFINED_8_BIT_DATA_TYPE_7 + EXP8
0x87	USER_DEFINED_8_BIT_DATA_TYPE_8 + EXP8
0x9E	YUV422 8bit + VP
0xA0	RGB444 + EXP16
0xA1	RGB555 + EXP16
0xAB	RAW10 + EXP16
0xAC	RAW12 + EXP16
0xAD	RAW14 + EXP16
0xE3	RGB666 + EXP32
0xE4	RGB888 + EXP32
0xE8	RAW6 + DPCM10 + VP
0x12A	RAW8 + VP
0x12C	RAW12 + VP
0x12D	RAW14 + VP
0x12F	RAW10 + VP
0x229	RAW7 + DPCM10 + EXP16
0x2A8	RAW6 + DPCM10 + EXP16
0x2AA	RAW8 + DPCM10 + EXP16
0x2C0	USER_DEFINED_8_BIT_DATA_TYPE_1 + DPCM10 + EXP16
0x2C1	USER_DEFINED_8_BIT_DATA_TYPE_2 + DPCM10 + EXP16
0x2C2	USER_DEFINED_8_BIT_DATA_TYPE_3 + DPCM10 + EXP16
0x2C3	USER_DEFINED_8_BIT_DATA_TYPE_4 + DPCM10 + EXP16
0x2C4	USER_DEFINED_8_BIT_DATA_TYPE_5 + DPCM10 + EXP16
0x2C5	USER_DEFINED_8_BIT_DATA_TYPE_6 + DPCM10 + EXP16
0x2C6	USER_DEFINED_8_BIT_DATA_TYPE_7 + DPCM10 + EXP16
0x2C7	USER_DEFINED_8_BIT_DATA_TYPE_8 + DPCM10 + EXP16
0x329	RAW7 + DPCM10 + VP
0x32A	RAW8 + DPCM10 + VP
0x340	USER_DEFINED_8_BIT_DATA_TYPE_1 + DPCM10 + VP
0x341	USER_DEFINED_8_BIT_DATA_TYPE_2 + DPCM10 + VP
0x342	USER_DEFINED_8_BIT_DATA_TYPE_3 + DPCM10 + VP
0x343	USER_DEFINED_8_BIT_DATA_TYPE_4 + DPCM10 + VP
0x344	USER_DEFINED_8_BIT_DATA_TYPE_5 + DPCM10 + VP

Table 12-49. CSI2 Receiver-Supported Data Types (continued)

Value	Data Type
0x345	USER_DEFINED_8_BIT_DATA_TYPE_6 + DPCM10 + VP
0x346	USER_DEFINED_8_BIT_DATA_TYPE_7 + DPCM10 + VP
0x347	USER_DEFINED_8_BIT_DATA_TYPE_8 + DPCM10 + VP

For each context, a CSI2_CTx_CTRL2 register defines with which channel and data type the context is associated:

- The VIRTUAL_ID field defines the associated virtual ID transported by the CSI2 protocol from the camera sensor.
- The FORMAT field defines the associated data type. The data type is a combination of the data type transported by the CSI2 protocol and the type of storage in memory. A given data type (RGB888) can be stored in memory in different ways (RGB888 or RGB888 + EXP32). Therefore, the FORMAT field also defines the way DMA stores data in memory.

For example, for the current context to capture a frame from virtual channel 2 and data type RAW12 with data expansion (RAW12 + EXP16), write the value 0x10AC (0x2 11 + 0xAC) in the 16 LSBs of the CSI2_CTx_CTRL2 register.

12.5.2.8 Progressive and Interleaved Frame Configuration

The CSI2 receiver can treat both progressive and interlaced frames. There is no progressive or interleaved mode, but the CSI2_CTx_CTRL1[23:16] FEC_NUMBER field controls the number of FECs before swapping to the other (ping or pong) buffer. Therefore, two modes are possible:

- FEC_NUMBER = 1: This is equivalent to progressive mode. After a FEC on the context, the current buffer is switched (ping to pong or pong to ping). The image in the memory buffer consists of one transmitted frame.
- FEC_NUMBER ! = 1: The current buffer is switched (ping to pong or pong to ping) after the FEC_NUMBER FEC is received for the context. The image in the memory buffer consists of the FEC_NUMBER transmitted frame.

For more information about how data is stored in memory through the DMA, see Section 12.4.3.7, *DMA Engine*.

Note: If FEC_NUMBER != 1, the camera sensor must send the line number information with the current line. Otherwise, the CSI2 receiver cannot calculate each line address.

12.5.3 Programming the Timing CTRL Module

Note: All the following settings must be done before enabling the timing control module.

12.5.3.1 Timing Generator

The cam_xclka clock frequency is set through the TCTRL_CTRL[4:0] DIVA bit field. The cam_xclkb clock frequency is set through the TCTRL_CTRL[9:5] DIVB bit field. One can change the divisor values at any time.

For divisor values 0, 1, and 31, the divider is not enabled. For all other values:

- cam_xclka = cam_mclk/TCTRL_CTRL[4:0] DIVA
- cam_xclkb = cam_mclk/TCTRL_CTRL[9:5] DIVB

12.5.3.2 Camera-Control Signal Generator

Enabling of the SHUTTER, PRESTROBE or STROBE signals generation and actives the counters:

- TCTRL_CTRL[21] SHUTEN = 1
- TCTRL_CTRL[22] PSTRBEN = 1
- TCTRL_CTRL[23] STRBEN = 1

Two configurations apply:

- The control signals are based on the vertical synchronization information coming from the camera module or from the externally generated cam_global_reset signal.
- The control signals are based on the internally generated cam_global_reset.

12.5.3.2.1 Vertical Synchro-Based Control-Signal Generation or Externally-Generated cam_global_reset

Before enabling the control-signal generation, the following registers must be set:

- Select the input that triggers the control signals. The trigger signal can come from the PARALLEL, CSIa, or CSIb interface, or the externally-generated cam_global_reset signal.
  - TCTRL_CTRL[28:27] INSEL
- The signal must be set to INPUT:
  - TCTRL_CTRL[31] GRESETDIR = 0x0
  - Writes to TCTRL_CTRL[29] GRESETEN bit do not trigger the PRESTROBE, STROBE, and SHUTTER signals, and do not generate the cam_global_reset signal.
- The following bits are cleared automatically to 0 after the signal assertion:
  - TCTRL_CTRL[21] SHUTEN
  - TCTRL_CTRL[22] PSTRBEN
  - TCTRL_CTRL[23] STRBEN
- The following bits set the polarity of the SHUTTER, STROBE/PRESTROBE, and cam_global_reset signals. The signals can be active high or active low:
  - TCTRL_CTRL[24] SHUTPOL
  - TCTRL_CTRL[26] STRBPSTRBPOL
  - TCTRL_CTRL[30] GRESETPOL
- The following bit sets the clock divisor value, which generates the CNTCLK clock:
  - TCTRL_CTRL[18:10] DIVC

  The clock is set by CNTCLK = cam_mclk/TCTRL_CTRL[18:10] DIVC. The possible values are 0 to 511. Setting DIVC = 0 disables the CNTCLK clock generation.
- The frame counters are set with (possible values are 0 to 63 frames):
  - TCTRL_FRAME[5:0] SHUT
  - TCTRL_FRAME[11:6] PSTRB
  - TCTRL_FRAME[17:12] STRB

  Note: If the value is zero, the Timing Control module does not delay any frame in input.
- The delay counters are set with:
  - TCTRL_SHUT_DELAY
  - TCTRL_PSTRB_DELAY
  - TCTRL_STRB_DELAY

  The possible values are 0 to 2^{25}- 1 cycles. The cycles are at the CNTCLK clock frequency. The maximum signal duration is (2^{25} -1) x 2.366 s = 79 s (TCTRL_CTRL[18:10] DIVC = 511).
- The signal durations are set with:
  - TCTRL_SHUT_LENGTH
  - TCTRL_PSTRB_LENGTH
  - TCTRL_STRB_LENGTH

  The possible values are 0 to 2^{24} -1 cycles. The cycles are at the CNTCLK clock frequency. The

maximum signal duration is (2^{24} -1) x 2.366 s = 39.69 s (TCTRL_CTRL[18:10] DIVC = 511).

12.5.3.2.2 Internally-Generated cam_global_reset-Based Control-Signal Generation

Before enabling the cam_global_reset control-signal generation by writing TCTRL_CTRL[29] GRESETEN = 1, the following registers must be set:

Note: Setting TCTRL_CTRL[21] SHUTEN, TCTRL_CTRL[22] PSTRBEN, TCTRL_CTRL[23] STRBEN, and TCTRL_CTRL[29] GRESETEN to 1 simultaneously leads to unpredictable behavior. The TCTRL_CTRL[21] SHUTEN, TCTRL_CTRL[22] PSTRBEN, TCTRL_CTRL[23] STRBEN must be set before TCTRL_CTRL[29] GRESETEN is enabled.

- The signal must be set to OUTPUT:
  - TCTRL_CTRL[31] GRESETDIR = 0x1
  - Vertical synchronization events do not trigger the PRESTROBE, STROBE, and SHUTTER signals.
- The following bits are cleared automatically to 0 after the signal assertion:
  - TCTRL_CTRL[21] SHUTEN
  - TCTRL_CTRL[22] PSTRBEN
  - TCTRL_CTRL[23] STRBEN
  - TCTRL_CTRL[29] GRESETEN
- The following bits set the polarity of the SHUTTER, STROBE/PRESTROBE, and cam_global_reset signals. The signals can be active high or active low:
  - TCTRL_CTRL[24] SHUTPOL
  - TCTRL_CTRL[26] STRBPSTRBPOL
  - TCTRL_CTRL[30] GRESETPOL
- The following bit sets the clock divisor value, which generates the CNTCLK clock:
  - TCTRL_CTRL[18:10] DIVC

    The clock is set by CNTCLK = cam_mclk/TCTRL_CTRL[18:10] DIVC. The possible values are 0 to 511. Setting DIVC = 0 disables the CNTCLK clock generation.
- The frame counters bit fields are ignored:
  - TCTRL_FRAME[5:0] SHUT
  - TCTRL_FRAME[11:6] PSTRB
  - TCTRL_FRAME[17:12] STRB
- The delay counters are set with:
  - TCTRL_SHUT_DELAY
  - TCTRL_PSTRB_DELAY
  - TCTRL_STRB_DELAY

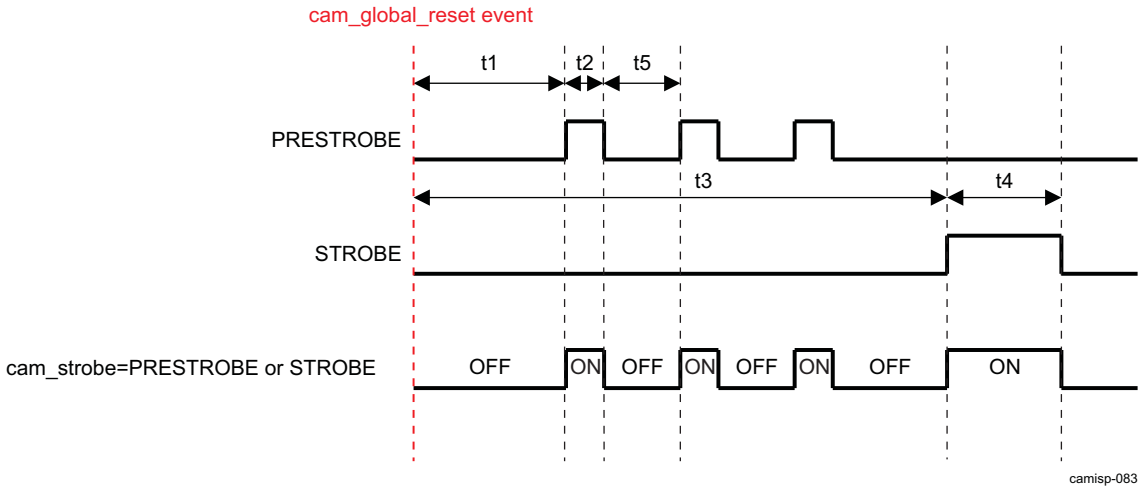
    The possible values are 0 to 225 -1 cycle. The cycles are at the CNTCLK clock frequency. The maximum signal duration is (2^{25} -1) x 2.366 s = 79 s (TCTRL_CTRL[18:10] DIVC = 511).
- The signal durations are set with:
  - TCTRL_SHUT_LENGTH
  - TCTRL_PSTRB_LENGTH
  - TCTRL_STRB_LENGTH

  The possible values are 0 to 224 -1 cycle. The cycles are at the CNTCLK clock frequency. The maximum signal duration is (2^{24} -1) x 2.366 s = 39.69 s (TCTRL_CTRL[18:11] DIVC = 511).
- The cam_global_reset assertion time is set by TCTRL_GRESET_LENGTH. The possible values are 0 to 224 -1 cycle. The cycles are at the CNTCLK clock frequency. The maximum signal duration is (2^{24} -1) x 2.366 s = 39.69 s (TCTRL_CTRL[18:11] DIVC = 511).

12.5.3.2.3 STROBE and PRESTROBE Signal Generation for Red-Eye Removal

The STROBE and PRESTROBE signal generation enables a strobe flash for red eye removal. The process is shown in Figure 12-102. The dotted line corresponds to known timings from which the delay counters start decreasing: cam_global_reset event.

Figure 12-102. cam_strobe Signal-Generation for Red-Eye Removal



- t1: Set by the TCTRL_PSTRB_DELAY register
- t2: set by the TCTRL_PSTRB_LENGTH register
- t5: set by the TCTRL_PSTRB_REPLAY[24:0] DELAY bit field. The number of times the pulse is repeated is controlled by the TCTRL_PSTRB_REPLAY [31:25] COUNTER register.
  In the former example, TCTRL_PSTRB_REPLAY [31:25] COUNTER = 2.
  - The possible delay values are 0 to 2^{25} -1 cycle. The cycles are at the CNTCLK clock frequency. The maximum signal duration is (2^{25} -1) x 2.366 s = 79 s (TCTRL_CTRL[18:11] DIVC = 511).
  - The possible count values are 0 to 127 additional pulses.
- t3: Set by the TCTRL_STRB_DELAY register
- t4: Set by the TCTRL_STRB_LENGTH register

12.5.4 Programming the CCDC

This section discusses issues related to the software control of the CCDC. It lists which registers are required to be programmed in different modes, and describes how to enable and disable the CCDC, how to check the status of the CCDC, the different register access types, and programming constraints.

12.5.4.1 CCDC Hardware Setup/Initialization

This section discusses the configuration of the CCDC required before image processing can begin.

12.5.4.1.1 Reset Behavior

On hardware reset of the Camera ISP, all registers in the CCDC are reset to their reset values.

12.5.4.1.2 Register Setup

Before enabling the CCDC, the hardware must be correctly configured through register writes. Table 12-50 identifies the register parameters that must be programmed before enabling the CCDC.

Table 12-50. CCDC Required Configuration Parameters

Function	Configuration Required
External pin signal configuration	CCDC_SYN_MODE[0] VDHDOUT
	CCDC_SYN_MODE[16] VDHDEN
	CCDC_SYN_MODE[2] VDPOL
	CCDC_SYN_MODE[3] HDPOL
	CCDC_SYN_MODE[7] FLDMODE
	CCDC_SYN_MODE[1] FLDOUT
	CCDC_SYN_MODE[4] FLDPOL
	CCDC_SYN_MODE [5] EXWEN
	CCDC_SYN_MODE[6] DATAPOL
	CCDC_CFG[15] VDLC = 1
	ISP_CTRL[3:2] PAR_BRIDGE
	ISP_CTRL[7:6] SHIFT
	ISP_CTRL[4] PAR_CLK_POL
Input mode	[0] R656ON
	CCDC_SYN_MODE[13 :12] INPMOD
Color pattern	CCDC_COLPTN
Black compensation	CCDC_BLKCMP
Faulty-pixel correction	CCDC_FPC[15] FPCEN
Data-path configuration	CCDC_FMTCFG[15] VPEN
	CCDC_SYN_MODE[18] VP2SDR
	CCDC_SYN_MODE[17] WEN
	CCDC_SYN_MODE[19] SDR2RSZ
Lens-shading compensation	CCDC_LSC_CONFIG[0] ENABLE

Table 12-51 identifies additional configuration requirements depending on whether the corresponding condition is met.

Table 12-51 can be read as: if (**Condition** is TRUE), then **Configuration Required** parameters must be programmed.

Table 12-51. CCDC Conditional Configuration Parameters

Function	Condition	Configuration Required
VD/HD set as outputs	CCDC_SYN_MODE[0] VDHDOUT = 0x1	CCDC_HD_VD_WID CCDC_PIX_LINES
Interlaced fields	CCDC_SYN_MODE[7] FLDMODE = 0x1	CCDC_CFG[7:6] FIDMD
External WEN	CCDC_SYN_MODE[5] EXWEN = 0x1	CCDC_CFG[8] WENLOG
REC656 input	CCDC_REC656IF[0] R656ON = 0x1 ISP_CTRL [3:2] PAR_BRIDGE = 0x0	CCDC_REC656IF[1] ECCFVH CCDC_CFG[5] BW656
YCC input	CCDC_SYN_MODE [13:12] INPMOD != 0 && CCDC_REC656IF[0] R656ON = 0x0	CCDC_CFG[13] MSBINVI CCDC_DCSUB
8bit YCC Input < 75 MHz	ISP_CTRL [3:2] PAR_BRIDGE = 0x0 CCDC_SYN_MODE [13:12] INPMOD == 2 && CCDC_REC656IF[0] R656ON = 0x0	CCDC_CFG[11] Y8POS
8bit YCC Input < 130 MHz	ISP_CTRL[3:2] PAR_BRIDGE = 0x1 || ISP_CTRL[3:2] PAR_BRIDGE = 0x2	
RAW input	CCDC_SYN_MODE [13:12] INPMOD == 0 && CCDC_REC656IF[0] R656ON = 0x0	CCDC_SYN_MODE [10:8] DATSIZ CCDC_CLAMP[31] CLAMPEN

Table 12-51. CCDC Conditional Configuration Parameters (continued)

Function	Condition	Configuration Required
Optical black clamp enabled	CCDC_CLAMP[31] CLAMPEN = 0x1 && CCDC_SYN_MODE[13:12] INPMOD == 0	CCDC_CLAMP[4:0] OBGAIN CCDC_CLAMP[24:10] OBST CCDC_CLAMP[27:25] OBSLN CCDC_CLAMP[30:28] OBSLEN
Optical black clamp disabled	CCDC_CLAMP[31] CLAMPEN = 0x0 && CCDC_SYN_MODE[13:12] INPMOD == 0	CCDC_DCSUB
Faulty-pixel correction	CCDC_FPC[15] FPCEN = 0x1	CCDC_FPC[14:0] FPNUM CCDC_FPC_ADDR Fault Pixel Table must be in memory
Video-port (data formatter) enabled	CCDC_FMTCFG[15] VPEN = 0x1	CCDC_FMTCFG[14:12] VPIN CCDC_FMT_HORZ CCDC_FMT_VERT CCDC_FMTCFG[0] FMTEN CCDC_VP_OUT CCDC_FMTCFG[18:16] VPIF_FRQ
Reformatter enabled	CCDC_FMTCFG[0] FMTEN = 0x1	CCDC_FMTCFG[1] LNALT
Reformatter enabled and line-alternating mode disabled	CCDC_FMTCFG[0] FMTEN = 0x1 && CCDC_FMTCFG[1] LNALT = 0x0	CCDC_FMTCFG[3:2] LNUM CCDC_FMT_ADDR_i (i = 0 to 7) CCDC_FMTCFG[11:8] PLEN_EVEN CCDC_FMTCFG[7:4] PLEN_ODD CCDC_PRGEVEN0 or CCDC_PRGEVEN1 CCDC_PRGODD0 or CCDC_PRGODD1
Write to memory or resizer	CCDC_SYN_MODE[17] WEN = 0x1 || CCDC_SYN_MODE[19] SDR2RSZ = 0x1	CCDC_HORZ_INFO CCDC_VERT_START CCDC_VERT_LINES CCDC_SYN_MODE[14] LPF CCDC_CULLING CCDC_ALAW[3] CCDTBL CCDC_SYN_MODE[11] PACK8 CCDC_CFG[12] BSWD
Write to memory	CCDC_SYN_MODE[17] WEN = 0x1	CCDC_SDR_ADDR CCDC_HSIZE_OFF CCDC_SDOFST
A-Law	CCDC_ALAW[3] CCDTBL = 0x1	CCDC_ALAW[2:0] GWDI
Interrupt usage	VDINT[1:0] Interrupts are enabled	CCDC_VDINT
Lens-shading compensation	CCDC_LSC_CONFIG[0] ENABLE	CCDC_LSC_CONFIG CCSC_LSC_INITIAL CCDC_LSC_TABLE_BASE CCDC_LSC_TABLE_OFFSET

12.5.4.1.3 Pixel Selection (Framing) Register Dependencies

There are three locations in the data flow where the valid frame data can be defined:

- Data formatter input pixel selection
- Video port output pixel selection
- Output formatter pixel selection

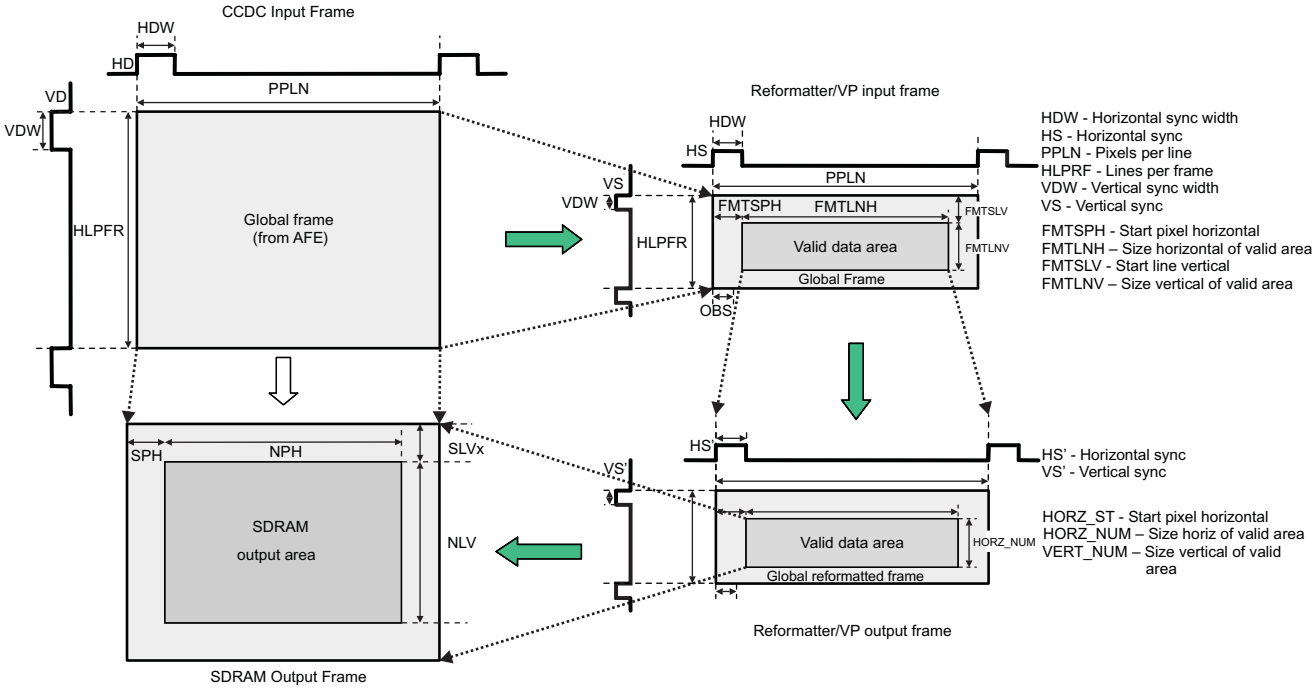
Care must be taken to ensure that the frame definitions correspond to the output of the upstream frame definitions. When the video port is enabled, CCDC_VP_OUT[30:17] VERT_NUM must be less than CCDC_FMT_VERT[12:0] FMTLNV.

Two data paths through the CCDC affect the programming of the pixel-selection registers, depending on the value of the CCDC_SYN_MODE[18] VP2SDR bit:

- VP2SDR = 0: The input data bypasses the data formatter/video port. In this case, only the memory output frame parameters apply. This data path is represented by the white arrow in Figure 12-103.

- VP2SDR = 1: The input data passes through the data formatter/video port. In this case, both data formatter frame definitions apply to the video-port output, and all three frame definitions apply to the memory output. This data path is represented by the green shaded arrow in Figure 12-103.

Figure 12-103. Dependencies Among Framing Settings in Data Flow



12.5.4.2 Enable/Disable Hardware

All required registers mentioned in the previous section must be programmed before setting the CCDC_PCR[0] ENABLE bit.

The CCDC always operates in continuous mode. In other words, after enabling the CCDC, it processes sequential frames until the ENABLE bit is cleared by software. When this happens, the frame being processed completes before the CCDC is disabled.

When the CCDC is in master mode (HS/VS signals set to outputs), fetching and processing of the frame begin immediately on setting the CCDC_PCR[0] ENABLE bit.

When the CCDC is in slave mode (HS/VS signals set to inputs), processing of the frame depends on the input timing of the external sensor/decoder. To ensure that data from the external device is not missed, the CCDC must be enabled before data transmission from the external device. In this way, the CCDC waits for data from the external device.

On setting the CCDC_FPC[15] FPCEN bit, the CCDC begins to fetch and buffer the faulty-pixel table from memory. If faulty-pixel correction is used, the CCDC_FPC[15] FPCEN bit must be set before the CCDC_PCR[0] ENABLE bit, but after the faulty-pixel table is placed in memory and the CCDC_FPC[14:0] FPNUM and CCDC_FPC_ADDR registers are set.

12.5.4.3 Events and Status Checking

The CCDC can generate three different interrupts: CCDC_VD0_IRQ, CCDC_VD1_IRQ, and CCDC_VD2_IRQ.

The CCDC_SYN_MODE[16] VDHDEN bit must be enabled to receive any of the CCDC CCDC_VDx_IRQ interrupts.

12.5.4.3.1 Interrupts

The CCDC module has three programmable events: CCDC_VD0_IRQ, CCDC_VD1_IRQ, and CCDC_VD2_IRQ, and one error event CCDC_ERR_IRQ. Event generation is described in Section 12.5.4.3.2, *CCDC_VD0_IRQ and CCDC_VD1_IRQ Interrupts*, and Section 12.5.4.3.3, *CCDC_VD2_IRQ Interrupt*.

CCDC module events can be mapped to the ARM or the DSP:

- The CCDC_VD0_IRQ, CCDC_VD1_IRQ, CCDC_VD2_IRQ, and CCDC_ERR_IRQ bits in the ISP_IRQ0ENABLE register control whether the CCDC module events trigger an interrupt to the ARM. The ISP_IRQ0STATUS register indicates which event(s) triggered the interrupt. An event is cleared by writing a 1 in its corresponding bit in the ISP_IRQ0STATUS register. To clear the CCDC_ERR_IRQ interrupt, clear the CCDC_FPC[16] FPERR bit before clearing the ISP_IRQ0STATUS[11] CCDC_ERR_IRQ bit.
- The CCDC_VD0_IRQ, CCDC_VD1_IRQ, CCDC_VD2_IRQ, and CCDC_ERR_IRQ bits in the ISP_IRQ1ENABLE register control whether the CCDC module events trigger an interrupt to the DSP. The ISP_IRQ1STATUS register indicates which event(s) triggered the interrupt. An event is cleared by writing a 1 in its corresponding bit in the ISP_IRQ1STATUS register. To clear the CCDC_ERR_IRQ interrupt, clear the CCDC_FPC[16] FPERR bit before clearing the ISP_IRQ1STATUS[11] CCDC_ERR_IRQ bit.

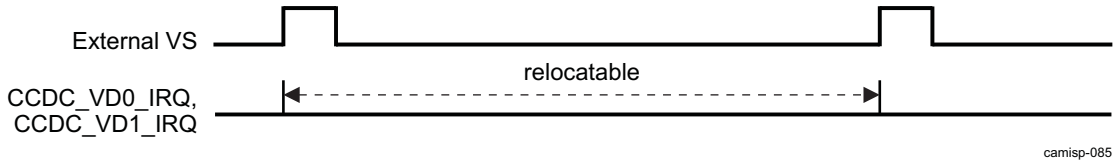
12.5.4.3.2 CCDC_VD0_IRQ and CCDC_VD1_IRQ Interrupts

As shown in Figure 12-104, the CCDC_VD0_IRQ and CCDC_VD1_IRQ interrupts occur relative to the VS pulse. The trigger timing is selected by using the CCDC_SYN_MODE[2] VDPOL setting. CCDC_VD0_IRQ and CCDC_VD1_IRQ occur after receiving the number of horizontal lines (HS pulse signals) set in the CCDC_VDINT[30:16] VDINT0 and CCDC_VDINT[14:0] VDINT1 register fields, respectively.

Note: In the case of BT.656 input mode, there is VS at the beginning of each field. Therefore, there are two interrupts for each frame (one for each field).

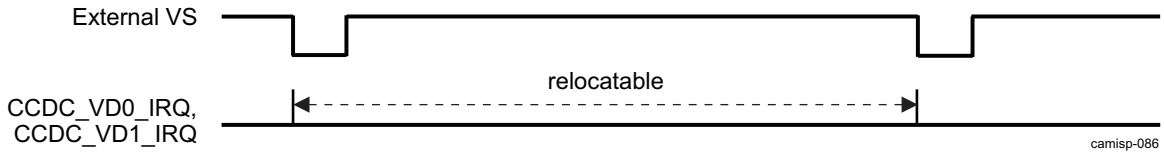
If CCDC_SYN_MODE[2] VDPOL is 0, the CCDC_VD0_IRQ and CCDC_VD1_IRQ HS counters begin counting HS pulses from the rising edge of the external VS, as shown in Figure 12-104.

Figure 12-104. CCDC_VD0_IRQ/CCDC_VD1_IRQ Interrupt Behavior When VDPOL = 0



If CCDC_SYN_MODE[2] VDPOL is 1, the CCDC_VD0_IRQ and CCDC_VD1_IRQ HS counters begin counting HS pulses from the falling edge of the external VS.

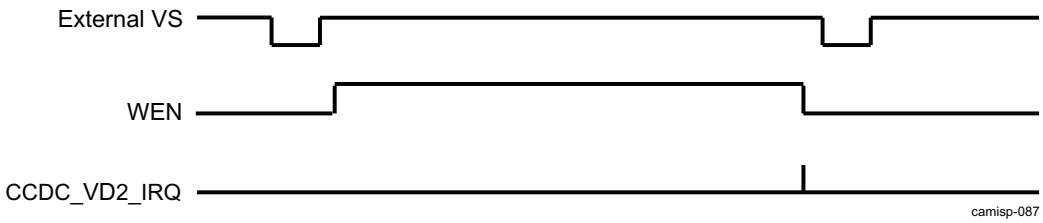
Figure 12-105. CCDC_VD0_IRQ/CCDC_VD1_IRQ Interrupt Behavior When VDPOL = 1



12.5.4.3.3 CCDC_VD2_IRQ Interrupt

In addition to the CCDC_VD0_IRQ and CCDC_VD1_IRQ interrupts, the CCDC has an interrupt called CCDC_VD2_IRQ. This interrupt always occurs at the falling edge of the WEN signal (through external pin). There are no registers in the CCDC module to configure this interrupt (see Figure 12-106).

Figure 12-106. CCDC_VD2_IRQ Interrupt Behavior



12.5.4.3.4 Status Checking

The CCDC_PCR 1] BUSY status bit is set when the start of frame occurs (if the CCDC_PCR[0] ENABLE bit is 1 at that time). It is automatically reset to 0 at the end of a frame. The CCDC_PCR[1] BUSY status bit may be polled to determine end-of-frame status.

The CCDC_FPC[16] FPERR status bit is set when faulty-pixel data fetched from memory arrives late. This bit can be reset by writing a 1 to the bit.

12.5.4.4 Register Accessibility During Frame Processing

There are three types of register access in the CCDC:

- Shadowed registers: In the CCDC, two register fields are shadowed in different ways. Shadowed registers can be read and written at any time, but the written values take effect (are latched) only at certain times, based on some event. Reads return the most recent write, even though the settings are not used until the specific event occurs. The shadowed registers are:
  - CCDC_PCR[0] ENABLE
    - Written values take effect only at the start of a frame event (rising edge of VD if CCDC_SYN_MODE[2] VDPOL is positive, or falling edge of VD if CCDC_SYN_MODE[2] VDPOL is negative).
  - CCDC_SDR_ADDR

  - When CCDC_CFG[15] VDLC is set to 0, written values take effect only at the start of a frame event (rising edge of VD if CCDC_SYN_MODE[2] VDPOL is positive, or falling edge of VD if CCDC_SYN_MODE[2] VDPOL is negative). When CCDC_CFG[15] VDLC is set to 1, written values take effect only at the start of the frame being output to memory (when the input has reached the CCDC_HORZ_INFO[30:16] SPH pixel of the CCDC_VERT_START[30:16] SLV0 or CCDC_VERT_START[14:0] SLV1 line of each field).

- Busy-writable registers: These registers/fields can be read or written even if the module is busy. Changes to the underlying settings occur instantaneously.

  All register fields in the CCDC not formerly listed as shadowed or below as optionally shadowed/busy-writable are busy-writable registers.
- Optionally Shadowed/Busy-Writable registers: All registers/fields listed below can be set as either shadow registers or busy-writable registers:
  - When CCDC_CFG[15] VDLC is 0, these registers are shadowed.
  - When CCDC_CFG[15] VDLC is 1, these registers are busy-writable.

Note: CCDC_CFG[15] VDLC must be set to 1 by software if the CCDC is to be used; therefore, these registers are busy-writable. If CCDC_CFG[15] VDLC remains set to 0 (default), indeterminate results may occur for ANY register access in the CCDC, not just those listed in the following.

  - Optionally shadowed or busy-writable registers
    - CCDC_SYN_MODE[19] SDR2RSZ
    - CCDC_SYN_MODE[18] VP2SDR
    - CCDC_SYN_MODE[16] VDHDEN
    - CCDC_SYN_MODE[17] WEN
    - CCDC_SYN_MODE[14] LPF
    - CCDC_HD_VD_WID
    - CCDC_PIX_LINES
    - CCDC_HORZ_INFO
    - CCDC_VERT_START
    - CCDC_VERT_LINES
    - CCDC_CULLING
    - CCDC_HSIZE_OFF
    - CCDC_SDOFST
    - CCDC_CLAMP[31] CLAMPEN
    - CCDC_FMTCFG[0] FMTEN
    - CCDC_LSC_CONFIG
    - CCDC_LSC_INITIAL
    - CCDC_LSC_TABLE_BASE
    - CCDC_LSC_TABLE_OFFSET

12.5.4.5 Interframe Operations

Between frames, it may be necessary to enable/disable functions or modify memory pointers. Because the CCDC_PCR register and memory pointer registers are shadowed, these modifications can take place any time before the end of the frame, and the data is latched in for the next frame. The MPU subsystem can perform these changes on receiving an interrupt.

12.5.4.6 CCDC Operations

12.5.4.6.1 Image-Sensor Configuration

12.5.4.6.1.1 Input-Mode Selection

The CCDC module supports two modes: SYNC mode and ITU-R BT.656 mode. Specific settings correspond to each mode.

- SYNC mode:
  - In this mode, the input data can be either raw data or YCbCr data. Setting CCDC_SYN_MODE.INPMODE = 0 selects raw data, and CCDC_SYN_MODE[13:12] INPMODE = 1 or 2 selects YCbCr data on 16 or 8 bits. If CCDC_SYN_MODE[13:12] INPMODE = 0, the cam_d signal width is selected through CCDC_SYN_MODE[10:8] DATSIZ: the possible values are 8, 10, 11, and 12 bits.
  - If CCDC_SYN_MODE[13:12] INPMODE = 1, the cam_d signal width is 8 bits, but the internal CCDC module data path is configured to 16 bits. It is mandatory to enable the 8- to 16-bit bridge by setting ISP_CTRL[3:2] PAR_BRIDGE = 2 or 3. The ISP_CTRL [3:2] PAR_BRIDGE bit also controls how the 8-bit data is mapped onto the 16-bit data.
  - The value set in CCDC_SYN_MODE[10:8] DATSIZ does not matter. The position of the Y component can be set with the CCDC_CFG[11] Y8POS bit.
  - If CCDC_SYN_MODE[13:12] INPMODE = 2, the cam_d signal width is 8 bits. The value set in CCDC_SYN_MODE[10:8] DATSIZ does not matter. The position of the Y component can be set with the CCDC_CFG[11] Y8POS bit.
  - The internal timing generator must be enabled with CCDC_SYN_MODE[16] VDHDEN = 1.
- ITU mode:
  - In this mode, the data follows the protocol set by the ITU-R BT.656 protocol. To select it, set CCDC_REC656IF[0] R656ON = 1. When this mode is selected, the values set in CCDC_SYN_MODE[13:12] INPMODE and CCDC_SYN_MODE[10:8] DATSIZ do not matter.
  - To select the 8-bit or 10-bit protocol, set the CCDC_CFG[5] BW656 bit field. Data line cam_d [7:0] are used for 8-bit YCbCr and cam_d[9:0] are used for 10-bit YCbCr.
  - FVH error correction is enabled by setting CCDC_REC656IF[1] ECCFVH = 1.
  - The internal timing generator must be enabled with CCDC_SYN_MODE[16] VDHDEN = 1.

12.5.4.6.1.2 Timing Generator and Frame Settings

The polarities of the cam_hs, cam_vs, and cam_fld signals are controlled by the CCDC_SYN_MODE[3] HDPOL, CCDC_SYN_MODE[2] VDPOL, and CCDC_SYN_MODE[4] FLDPOL bit fields. The polarities can be positive or negative.

The pixel data is presented on cam_d one pixel for every cam_pclk rising edge or falling edge. It is controlled with the ISP_CTRL[4] PAR_CLK_POL bit.

The CCDC_SYN_MODE[7] FLDMODE bit fields set the image-sensor type to progressive or interlaced mode. When the sensor is interlaced, the CCDC_SYN_MODE[15] FLDSTAT status bit indicates whether the current frame is odd or even.

The polarity of the cam_d signal can also be controlled with the CCDC_SYN_MODE[6] DATAPOL bit field. The polarity can be normal mode or ones complement mode.

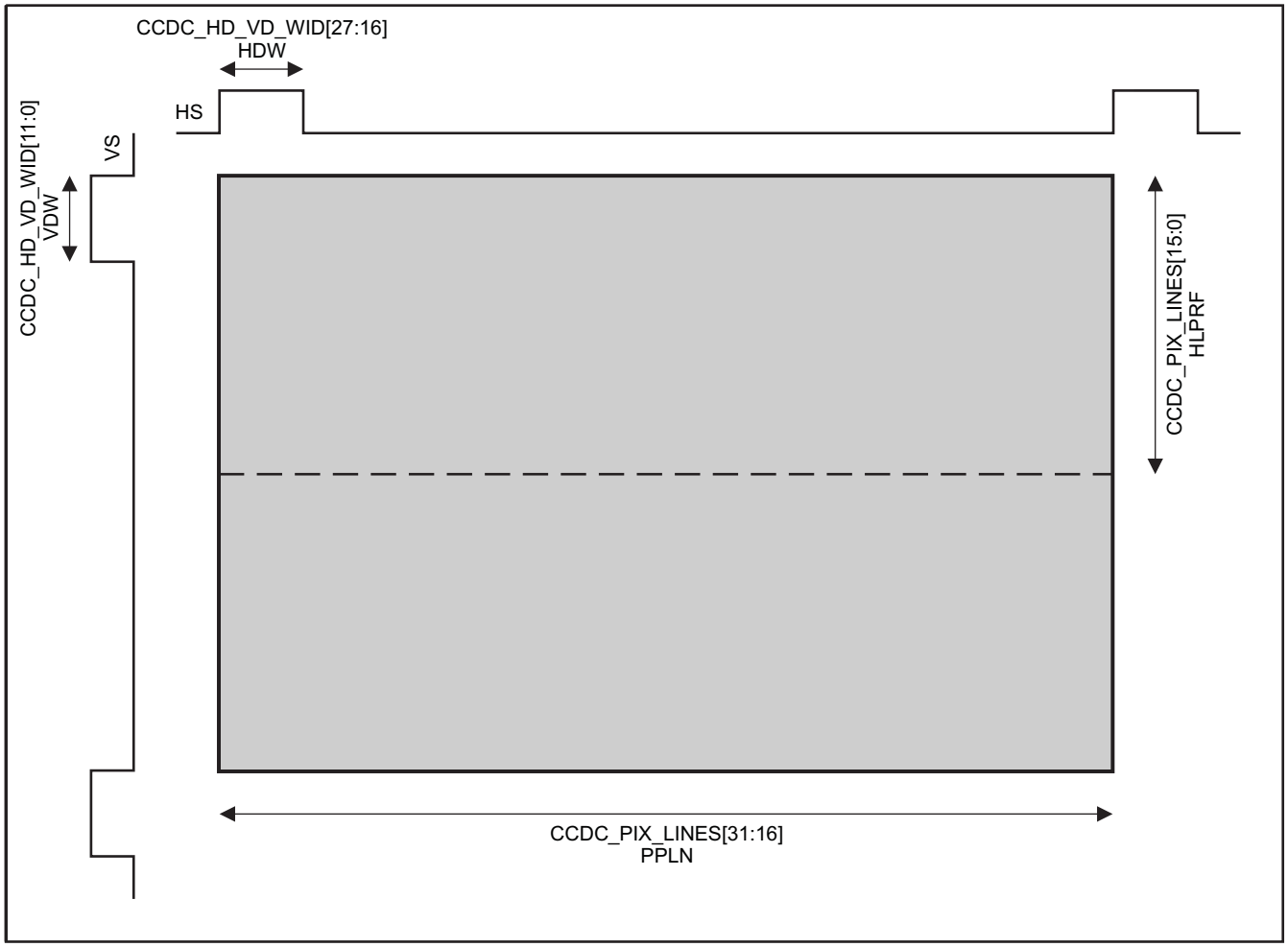
Furthermore, the directions of the cam_fld and cam_hs/cam_vs signals are controlled by the CCDC_SYN_MODE[1] FLDOUT and CCDC_SYN_MODE[0] VDHDOUT bits. If CCDC_SYN_MODE[0] VDHDOUT is set as an output, the CCDC_PIX_LINES register controls the length of the cam_hs and cam_vs signals.

If CCDC_SYN_MODE[0] VDHDOUT = 1:

- The HS sync pulse width is given by CCDC_HD_VD_WID[27:16] HDW. The VS sync pulse width is given by CCDC_HD_VD [11:0] VDW.
- The HS period is given by CCDC_PIX_LINES[31:16] PPLN. The VS period is given by CCDC_PIX_LINES[15:0] HLPRF x 2.

Figure 12-107 shows the HS/VS sync pulse output timings.

Figure 12-107. HS/VS Sync Pulse Output Timings

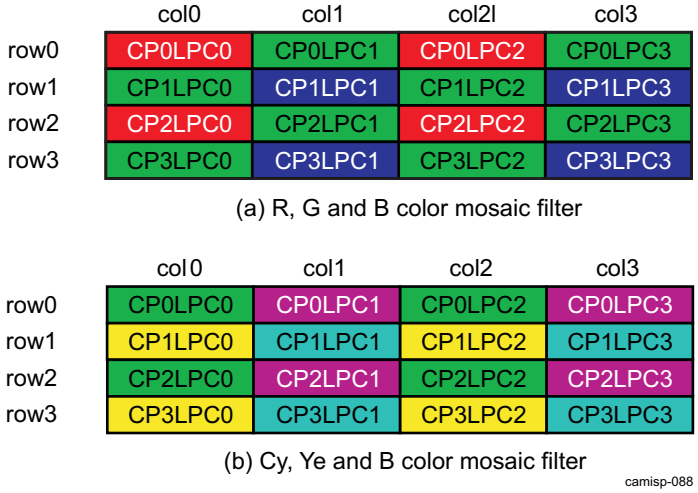


12.5.4.6.1.3 Mosaic Filter Settings

The CCDC module supports image sensors with R, G, and B primary color mosaic filters and Ye, Cy, Mg, and G complementary color mosaic filters. The mosaic filter layout pattern is controlled by the CCDC_COLPTN register. Each bit field in this register controls the color associated to one pixel in a 4x4 region area. The 4x4 area repeats horizontally and vertically.

Figure 12-108 shows configuration examples of the CCDC_COLPTN register. It is assumed that CP0LPC0 is the first pixel output and CP3LPC3 is the last pixel output during frame readout.

Figure 12-108. Mosaic Filter - CCDC_COLPTN Bit Field Settings



12.5.4.6.2 Image-Signal Processing

12.5.4.6.2.1 Digital Clamp

Digital clamp is enabled only if optical black clamping is disabled: CCDC_CLAMP[31] CLAMPEN = 0. The digital clamp DC value to be subtracted from the raw image data (CCDC_SYN_MODE[13:12] INPMOD = 0) or from the luminance (CCDC_SYN_MODE[13:12] INPMOD = 0 or CCDC_REC656IF[0] R656ON = 1) is set with the CCDC_DCSUB register. The DC value can range from 0 to 212-1. The reset value is 0.

12.5.4.6.2.2 Optical Black Clamping

Optical black clamping is enabled by setting CCDC_CLAMP[31] CLAMPEN to 1. When enabled, an average of black-pixel samples is computed over a window. If the height of the window is 2N, the average value multiplied by a programmable gain factor is subtracted from the raw image data for the following 2N lines. Every 2N lines, a new average value is computed. For the first 2N lines, 0 is subtracted.

The size of the window and horizontal position are controlled by the CCDC_CLAMP register. The vertical position of the window is set to line 0 and cannot be modified:

- CCDC_CLAMP[30:28] OBSLEN sets the horizontal size of the window (1, 2, 4 or 8 pixels).
- CCDC_CLAMP[27:25] OBSLN sets the vertical size of the window (2N = 1, 2, 4 or 8 lines).
- CCDC_CLAMP[24:10] OBST sets the horizontal start position of the upper-left corner of the window. It is specified from the start of the HS sync pulse in pixel clocks.

The gain factor is set by the CCDC_CLAMP[4:0] OBGAIN bit field. Its fixed-point representation is U5Q4; the range is 0 to 1.9375. The gain factor reset value is 1.

If optical-black clamping is disabled, digital clamp is enabled.

12.5.4.6.2.3 Black Compensation

Black compensation applies an offset to the raw image data. The offset is applied according to the phase and color for each phase. The black compensation offsets are controlled by the CCDC_BLKCMP register. All offsets are coded in S8Q0 representation (two's complement); the range is -128 to +127.

12.5.4.6.2.4 Faulty-Pixel Correction

Faulty-pixel correction is enabled by setting the CCDC_FPC[15] FPCEN bit to 1. Before activating faulty-pixel correction, set the number of faulty pixels to be corrected in a frame with the CCDC_FPC[14:0] FPNUM bit field, set the faulty-pixel LUT in memory, and set the CCDC_FPC_ADDR register to the LUT address. The address should be aligned to a 64-bit byte boundary; the 6 LSBs are ignored. Reading the register always shows the 6 LSBs as 0.

If the CCDC module cannot fetch the required faulty-pixel entry in time, an error is set in the CCDC_FPC[16] FPERR bit. After the bit is set, no more faulty pixels are corrected in the frame. The bit is automatically cleared on the end of the frame and the feature reenabled for the following frame.

12.5.4.6.2.5 Data Formatter

The data formatter transforms movie mode readout patterns into Bayer readout patterns. It is enabled by setting CCDC_FMTCFG[0] FMTEN to 1.

The CCDC_FMT_HORZ and CCDC_FMT_VERT registers set a clip window at the input of the data reformatter.

The data formatter converts a single line of movie mode sensor into multiple Bayer lines; it is capable of transforming 1 input line into 1, 2, 3, or 4 output lines. The number of lines generated from one input line is set by the CCDC_FMTCFG[3:2] LNUM bit field. The following limitations apply:

- The maximum number of pixels that can be supported in an output line if the input line is transformed into 1 output line is 4x1376 (1376 is the limit of the line memory).
- The maximum number of pixels that can be supported in an output line if the input line is transformed into 2 output lines is 2x1376.
- The maximum number of pixels that can be supported in an output line if the input line is transformed into 3 output lines is 1376.
- The maximum number of pixels that can be supported in an output line if the input line is transformed into 4 output lines is 1376.

The data reformatter gets its flexibility from up to 8 different addresses and a program that can contain up to 16 entries each for the odd and even lines. The program length for even fields is set with the CCDC_FMTCFG[11:8] PLEN_EVEN bit field. The program length for odd fields is set with the CCDC_FMTCFG[7:4] PLEN_ODD bit fields. Each entry refers to one of the 8 addresses and supports autoincrement and autodecrement.

- The 8 addresses are controlled by the CCDC_FMT_ADDR_i registers (i = 0 to 7).
- The 16 program entries for even lines are controlled by the CCDC_PRGEVEN0 and CCDC_PRGEVEN1 registers. The 16 program entries for odd lines are controlled by the CCDC_PRGODD0 and CCDC_PRGODD1 registers.

Modulo addressing is used to access the program entries. Even input lines use the even program and odd input lines use the odd program. Each new pixel in a line uses one program entry.

The CCDC_VP_OUT register sets the frame size at the output of the video port.

Example 1: Conventional readout pattern: 1 input line = 1 output line

The following input-to-output mapping (see Table 12-52) corresponds to a conventional readout pattern. One input line corresponds to 1 output line; the output can be as large as 4x1376 pixels.

Table 12-52. Conventional Readout Pattern 1 to 1

Input	**Pixels order in input line**								
Line [i]	0	1	2	3	[...]	5500	5501	5502	5503
Output	**Pixels order in output line**								
Line [i]	0	1	2	3	[...]	5500	5501	5502	5503

To obtain this mapping between the input and output pixels, the following settings apply:

- CCDC_FMTCFG [3:2] LNUM	= 0	Converts to 1 line
- CCDC_FMTCFG [11:8] PLEN_EVEN	= 0	1 program entry
- CCDC_FMTCFG [7:4] PLEN_ODD	= 0	1 program entry
- CCDC_FMT_ADDR_i [25:24] LINE (i = 0)	= 0	Init ADDR0 pointer to 0th line
- CCDC_FMT_ADDR_i [12:0] INIT (i = 0)	= 0	Init ADDR0 index to 0th pixel

- CCDC_PRGEVEN0 [3:0] EVEN0 = 0 Even prog entry 0: ADDR0++
- CCDC_PRGODD0 [3:0] ODD0 = 0 Odd prog entry 0: ADDR0++
- CCDC_VP_OUT [3:0] HORZ_ST = 0 Horizontal start
- CCDC_VP_OUT [16:4] HORZ_NUM = 0 Horizontal size

The program for even and odd lines is executed as follows. Because there is only one program entry, this instruction is executed repeatedly.

```
ADDR0= (0, 0)
While not end_of_line
{
Write
              incoming pixel to ADDR0
ADDR0+=(1,0)
}
```

Example 2: Dual readout pattern: 1 input line = 1 output line

The following input-to-output mapping (see Table 12-53) corresponds to a conventional dual readout pattern. One input line corresponds to 1 output line; the output can be as large as 4x1376 pixels.

Table 12-53. Dual Readout Pattern 1 to 1

Input	**Pixels order in input line**								
Line [i]	0	1	2	3	[...]	4092	4093	4094	4095
Output	**Pixels order in output line**								
Line [i]	0	2	4	6	[...]	7	5	3	1

To obtain this mapping between the input and output pixels, the following settings apply:

- CCDC_FMTCFG [3:2] LNUM = 0 Converts to 1 line
- CCDC_FMTCFG [11:8] PLEN_EVEN = 1 2 program entry
- CCDC_FMTCFG [7:4] PLEN_ODD = 1 2 program entry
- CCDC_FMT_ADDR_i [25:24] LINE (i = 0) = 0 Init ADDR0 pointer to 0th line
- CCDC_FMT_ADDR_i [12:0] INIT (i = 0) = 0 Init ADDR0 index to 0th pixel
- CCDC_FMT_ADDR_i [25:24] LINE (i = 1) = 0 Init ADDR1 pointer to 0th line
- CCDC_FMT_ADDR_i [12:0] INIT (i = 1) = 4095 Init ADDR1 index to 4095th pixel
- CCDC_PRGEVEN0 [3:0] EVEN0 = 0 Even prog entry 0: ADDR0++
- CCDC_PRGEVEN0 [7:4] EVEN1 = 3 Even prog entry 1: ADDR1- -
- CCDC_PRGODD0 [3:0] ODD0 = 0 Even prog entry 0: ADDR0++
- CCDC_PRGODD0 [7:4] ODD1 = 3 Even prog entry 1: ADDR1- -
- CCDC_VP_OUT [3:0] HORZ_ST = 0 Horizontal start
- CCDC_VP_OUT [16:4] HORZ_NUM = 0 Horizontal size

```
ADDR0=(0,0)
ADDR1=(4095,0)
While not
              end_of_line
{
Write incoming pixel to
              ADDR0
ADDR0+= (1,0)
Write incoming pixel to
              ADDR1
ADDR1 -=(1,0)
}
```

Example 3: Dual readout pattern: 1 input line = 3 output line

The following input-to-output mapping (see Table 12-54) corresponds to a complex pattern. One input line corresponds to 3 output lines; the output can be as large as 1376 pixels.

Table 12-54. Dual Readout Pattern 1 to 3

Input	**Pixels order in input line**								
Line [i]	0	1	2	3	4	5	6	7	8
	9	10	11	[...]					
Output	**Pixels order in output line**								
Line [3i]			0	6	[...]	23	17	11	5
Line [3i + 1]		2	8	14	[...]	15	9	3	
Line [3i + 2]	4	10	16	22	[...]	7	1		

To obtain this mapping between the input and output pixels, the following settings apply:

- CCDC_FMTCFG [3:2] LNUM = 2 Converts to 3 lines
- CCDC_FMTCFG [11:8] PLEN_EVEN = 5 6 program entries
- CCDC_FMTCFG [7:4] PLEN_ODD = 5 6 program entries
- CCDC_FMT_ADDR_i [25:24] LINE (i = 0) = 0 Init ADDR0 pointer to 0th line
- CCDC_FMT_ADDR_i [12:0] INIT (i = 0) = 2 Init ADDR0 index to 2nd pixel
- CCDC_FMT_ADDR_i [25:24] LINE (i = 1) = 2 Init ADDR1 pointer to 2nd line
- CCDC_FMT_ADDR_i [12:0] INIT (i = 1) = 855 Init ADDR1 index to 855th pixel
- CCDC_FMT_ADDR_i [25:24] LINE (i = 2) = 1 Init ADDR2 pointer to first line
- CCDC_FMT_ADDR_i [12:0] INIT (i = 2) = 1 Init ADDR2 index to first pixel
- CCDC_FMT_ADDR_i [25:24] LINE (i = 3) = 1 Init ADDR3 pointer to first line
- CCDC_FMT_ADDR_i [12:0] INIT (i = 3) = 856 Init ADDR3 index to 856th pixel
- CCDC_FMT_ADDR_i [25:24] LINE (i = 4) = 2 Init ADDR4 pointer to second line
- CCDC_FMT_ADDR_i [12:0] INIT (i = 4) = 0 Init ADDR4 index to 0th pixel
- CCDC_FMT_ADDR_i [25:24] LINE (i = 5) = 0 Init ADDR5 pointer to 0th line
- CCDC_FMT_ADDR_i [12:0] INIT (i = 5) = 857 Init ADDR5 index to 857th pixel
- CCDC_PRGEVEN0 [3:0] EVEN0 = 0 Even prog entry 0: ADDR0++
- CCDC_PRGEVEN0 [7:4] EVEN1 = 3 Even prog entry 1: ADDR1- -
- CCDC_PRGEVEN0 [11:8] EVEN2 = 4 Even prog entry 2: ADDR2++
- CCDC_PRGEVEN0 [15:12] EVEN3 = 7 Even prog entry 3: ADDR3- -
- CCDC_PRGEVEN0 [19:16] EVEN4 = 8 Even prog entry 4: ADDR4++
- CCDC_PRGEVEN0 [23:20] EVEN5 = 11 Even prog entry 4: ADDR5- -
- CCDC_PRGODD0 [3:0] ODD0 = 0 Even prog entry 0: ADDR0++
- CCDC_PRGODD0 [7:4] ODD1 = 3 Even prog entry 1: ADDR1- -
- CCDC_PRGODD0 [11:8] ODD2 = 4 Even prog entry 2: ADDR2++
- CCDC_PRGODD0 [15:12] ODD3 = 7 Even prog entry 3: ADDR3- -
- CCDC_PRGODD0 [19:16] ODD4 = 8 Even prog entry 4: ADDR4++
- CCDC_PRGODD0 [23:20] ODD5 = 11 Even prog entry 4: ADDR5- -

- CCDC_VP_OUT [3:0] HORZ_ST = 2 Horizontal start, excludes first 2 pixels
- CCDC_VP_OUT [16:4] HORZ_NUM = 854 Horizontal size

```
ADDR0=(2,0)
ADDR1=(855,2)
ADDR2=(1,1)
ADDR3=(856,1)
ADDR4=(0,2)
ADDR5=(857,0)
While
              not end_of_line
{
Write incoming pixel to
              ADDR0
ADDR0 += (1, 0)
Write incoming pixel to
              ADDR1
ADDR1 -= (1, 0)
Write incoming pixel to
              ADDR2
ADDR2 += (1, 0)
Write incoming pixel to
              ADDR3
ADDR3 -= (1, 0)
Write incoming pixel to
              ADDR4
ADDR4 += (1, 0)
Write incoming pixel to
              ADDR5
ADDR5 -= (1, 0)
}
```

Video Port

The 10-bit video-port output is enabled with CCDC_FMTCFG[15] VPEN = 1. Because the input data can be up to 12 bits, one must select which 10 bits are selected with CCDC_FMTCFG[14:12] VPIN.

The output of the video port goes to the Preview module. At the output of the video port, the data rate is resynchronized; the CCDC_FMTCFG[18:16] VPIF_FRQ bit field selects the video-output data rate as: L3 speed/(CCDC_FMTCFG[18:16]VPIF_FRQ + 2) (from L3 speed/2 MHz to L3 speed/8 MHz). If CCDC_FMTCFG[18:16] VPIF_FRQ frequency is set too low compared to the input pixel clock, overflow can occur.

12.5.4.6.2.6 Output Formatter

12.5.4.6.2.6.1 Low-Pass Filter

The low-pass filter is enabled with the CCDC_SYN_MODE[14] LPF bit. When enabled, two pixels each in the left and right edges of each line are cropped from the output.

12.5.4.6.2.6.2 A-Law Compression

A-Law compression is enabled by setting CCDC_ALAW [3] CCDTBL to 1. The A-Law table is fixed, so no setup is required. When the input is wider than 10 bits, the CCDC_ALAW [2:0] GWDI bit is used to select which 10 bits of the 12 possible bits are selected for compression. See Table 12-55.

Table 12-55. CCDC_ALAW [2:0] GWDI

GWDI	Description
4	Bits 11 to 2
5	Bits 10 to 1
6	Bits 9 to 0
Others	Reserved

12.5.4.6.2.6.3 Culling

Culling performs a horizontal and vertical decimation function. The horizontal and vertical culling patterns are set by the CCDC_CULLING register.

- The 8-bit CCDC_CULLING[31:24] CULHEVN and CCDC_CULLING [23:16] CULHODD bit fields set the horizontal culling pattern for even and odd lines. A 1 means that the pixel is retained; a 0 means that the pixel is skipped. The same number of retained pixels must be set for even and odd lines.
- The 8-bit CCDC_CULLING[7:0] CULV bit field sets the vertical culling pattern. The LSB represents the top line and the MSB represents the bottom line. A 1 means that the line is retained; a 0 means that the line is skipped.

12.5.4.6.2.6.4 Data Packing

Pixel data are stored to memory in little-endian format (bytes at lower addresses have lower significance). By default, pixel data are stored in 16-bit words; unused bits are filled with zeros.

If the input data is 8 bits, or if A-Law compression is enabled, the data can be stored in 8-bit words by setting the CCDC_SYN_MODE[11] PACK8 bit to 1.

If the input data is 12, 11, 10, or 9 bits, and A-Law compression is not enabled, the 8 MSBs are stored to memory.

Figure 12-109 shows data packing and pixel ordering.

Figure 12-109. Data Packing - Pixel Ordering

	31	24	15	07	00
12 bits	0	pix 1	0	pix 0	
11 bits	0	pix 1	0	pix 0	
10 bits	0	pix 1	0	pix 0	
9 bits	0	pix 1	0	pix 0	
8 bits	0	pix 1	0	pix 0	
8 bits packed	pix3	pix 2	pix 1	pix 0	

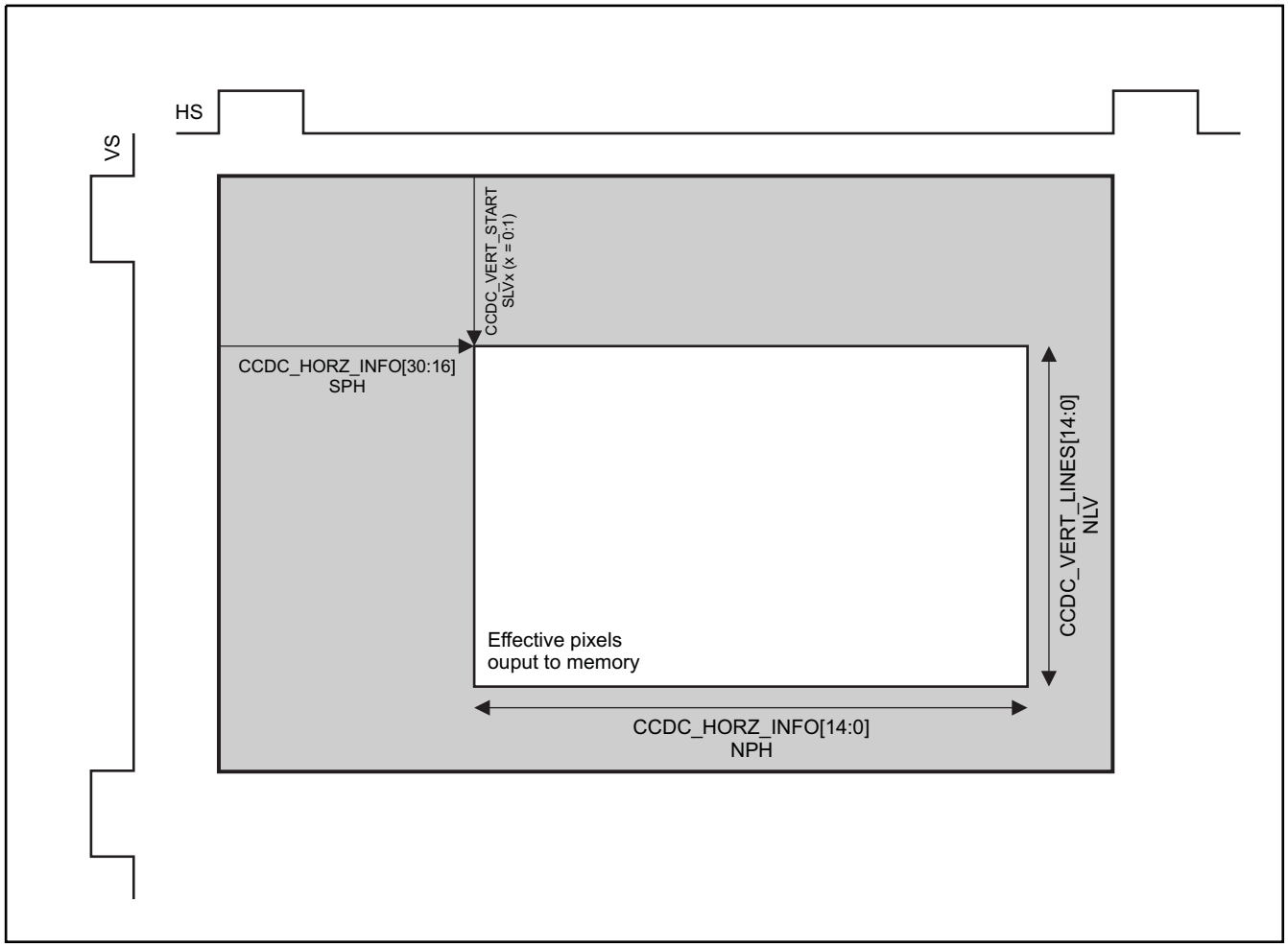
camisp-118

12.5.4.6.2.6.5 Clipping Window

Before data is stored in memory, a clipping window can be set; only a selected sensor area is stored to memory. Figure 12-110 shows the settings; only the white area is stored to memory.

- The valid-data horizontal start position is controlled with the CCDC_HORZ_INFO[30:16] SPH bit field. The valid-data vertical start position is controlled with the CCDC_VERT_START register. If the sensor is interlaced, the vertical start position for even and odd fields can be configured independently with the CCDC_VERT_START register. If the sensor is progressive, the CCDC_VERT_START[14:0] SLV1 bit field is ignored.
- The valid-data horizontal size is controlled with the CCDC_HORZ_INFO[14:0] NPH bit field. The horizontal size must be a multiple of 16 pixels. The valid-data vertical size is controlled with the CCDC_VERT_LINES[14:0] NLV register.

Figure 12-110. Clipping Window Before Output to Memory



12.5.4.6.2.6.6 Output to Memory

The output formatter memory write enable is controlled by the CCDC_SYN_MODE[17] WEN bit. The output of the data reformatter can be written to memory by setting CCDC_SYN_MODE[18] VP2SDR to 1.

The pixel data at the output of the output formatter is written to the address given by the CCDC_SDR_ADDR register. The address should be aligned on a 32-byte boundary. The 5 LSBs are ignored. Reading the register always shows the 5 LSBs as 0.

A destination pitch can be set with the CCDC_HSIZE_OFF register. The offset must be a multiple of 32 bytes. The 5 LSBs are ignored. Reading the register always shows the 5 LSBs as 0. It is required for the pixel line length to be a multiple of 32 bytes to be stored in memory without holes.

The CCDC_SDOFST register controls how the pixels are stored to memory.

Data to be written to memory can be qualified by the external cam_wen signal. This feature can be enabled by setting the CCDC_SYN_MODE[5] EXWEN bit. The CCDC_CFG [8] WENLOG bit configures how the cam_wen signal is used with the internally generated valid signal.

The data can be swapped on a byte basis with the CCDC_CFG[12] BSWD bit.

If CCDC_SYN_MODE[13:12] INPMOD = 1, the MSB of the 8-bit chroma component can be inverted by setting CCDC_CFG[13] MSBINVI to 1.

12.5.4.7 Summary of Constraints

The following is a list of register configuration constraints to adhere to when programming the CCDC. It can be used as a quick checklist. More detailed register setting constraints can be found in the individual register descriptions.

- If the memory output port is enabled, the memory output line offset and address should be on 32-byte boundaries.
- If faulty-pixel correction is enabled, the CCDC_FPC_ADDR address should be on a 64-byte boundary.
- External WEN cannot be used when the VP2SDR path is enabled.
- If the formatter is enabled, in line-alternating mode, the vertical start and end number should be even.
- The horizontal number for the video port must be <= 1376*4.
- If the video port is enabled, CCDC_VP_OUT[30:17] VERT_NUM must be CCDC_FMT_VERT[12:0] FMTLNV.
- The video port must be enabled if the formatter is enabled.
- In YCC input mode:
  - The CCDC_COLPTN must be set to 0s.
  - The CCDC_BLKCMP must be set to 0s.
  - Faulty-pixel correction must be disabled.
  - The video port must be disabled.
  - The formatter must be disabled.
  - The VP2SDR must be disabled.
  - The low-pass filter must be disabled.
  - The A-Law must be disabled.
- In RAW input mode, the resizer output path should not be enabled.

12.5.5 Programming the Preview Engine

This section discusses issues related to software control of the preview engine. It lists which registers are required to be programmed in different modes, how to enable and disable the preview engine, and how to check the status of the preview engine; discusses the different register access types; and enumerates programming constraints.

12.5.5.1 Preview Hardware Setup/Initialization

This section discusses the configuration of the preview engine required before image processing can begin.

12.5.5.1.1 Reset Behavior

On hardware reset of the Camera ISP, all registers in the preview engine are reset to their reset values. However, because the preview engine programmable tables (gamma, noise filter, luminance enhancer, and CFA coefficients) are stored in internal memory, their contents do not have reset values. If the reset is a chip-level power-on reset (reset after power is applied), the contents of these tables are unknown. If the reset is a Camera ISP module reset (when power remains active), the contents of these tables remain the same as before the reset.

12.5.5.1.2 Register Setup

Before enabling the preview engine, the hardware must be correctly configured through register writes. Table 12-56 identifies the register parameters that must be programmed before enabling the preview engine.

Table 12-56. Preview Engine Required Configuration Parameters

Function	Configuration Required
Function enable/disable	PRV_PCR[5] INVALAW
	PRV_PCR[7] DRKFCAP
	PRV_PCR[6] DRKFEN
	PRV_PCR[21] SCOMP_EN
	PRV_PCR[8] HMEDEN
	PRV_PCR[9] NFEN
	PRV_PCR[10] CFAEN
	PRV_PCR[26] GAMMA_BYPASS
	PRV_PCR[15] YNENHEN
	PRV_PCR[16] SUPEN
	PRV_PCR[27] DCOREN
I/O ports	PRV_PCR[2] SOURCE
	PRV_PCR[20] SDRPORT
	PRV_PCR[19] RSZPORT
Input size	PRV_HORZ_INFO
	PRV_VERT_INFO
Averager	PRV_AVE
White balance	PRV_PCR[14:11] CFAFMT
	PRV_WB_DGAIN
	PRV_WBGAIN
	PRV_WBSEL
Black adjustment	PRV_BLKADJOFF
RGB-to-RGB blending	PRV_RGB_MAT1 to PRV_RGB_MAT5
	PRV_RGB_OFF1 to PRV_RGB_OFF2
RGB-to-YCbCr conversion	PRV_CSC0 to PRV_CSC2
Contrast and brightness	PRV_CNT_BRT
YCC output format	PRV_SETUP_YC
	PRV_PCR[18:17] YCPOS

The PRV_PCR register contains control bits that enable or disable optional functions and module I/O ports. If an optional function or port is enabled, there may be more registers or configuration information required for the preview engine to operate correctly.

Table 12-57 can be read as:

If (**Condition** is TRUE) then

Configuration required parameters must be programmed.

Table 12-57. Preview Engine Conditional Configuration Parameters

Function	Condition	Configuration Required
Read from CCDC	PRV_PCR [2] SOURCE = 0x0	PRV_PCR[3] ONESHOT
Read from memory	PRV_PCR [2] SOURCE = 0x1	PRV_PCR[4] WIDTH
		PRV_RSDR_ADDR
		PRV_RADR_OFFSET
		ISP_CTRL[27] SBL_SHARED_RPORTA
Dark frame subtract	PRV_PCR [6] DRKFEN = 0x1	PRV_DSDR_ADDR
		PRV_DRKF_OFFSET
		ISP_CTRL[28] SBL_SHARED_RPORTB
		Dark frame must be in memory

Table 12-57. Preview Engine Conditional Configuration Parameters (continued)

Function	Condition	Configuration Required
Shading correction	PRV_PCR [21] SCOMP_EN = 0x1 && PRV_PCR [6] DRKFEN = 0x1	PRV_PCR[24:22] SCOMP_SFT PRV_DSDR_ADDR PRV_DRKF_OFFSET Dark frame must be in memory
Horizontal median filter	PRV_PCR [8] HMEDEN = 0x1	PRV_HMED
Noise filter	PRV_PCR [9] NFEN = 0x1	PRV_NF[1:0] SPR Setup Noise Filter Tables
Defect correction	PRV_PCR [27] DCOREN = 0x1	PRV_PCR[28] DCOR_METHOD **PRV_CDC_THRx (x = 0 to 3)**
CFA interpolation	PRV_PCR [10] CFAEN = 0x1	PRV_CFA Setup CFA Coefficient Table
Gamma correction	PRV_PCR [26] GAMMA_BYPASS = 0x0	Setup Gamma Correction Tables
Luminance enhancement	PRV_PCR [15] YNENHEN = 0x1	Setup Luminance Enhancement Table
Chrominance suppression	PRV_PCR [16] SUPEN = 0x1	PRV_CSUP
Write to memory	PRV_PCR [20] SDRPORT = 0x1	PRV_WSDR_ADDR PRV_WADD_OFFSET

12.5.5.1.3 Table Setup

The three gamma memories, noise filter threshold memory, noise filter strength memory, luminance enhancer memory, and the CFA coefficient memory must be filled in before the operation of the preview engine, if their respective functions are enabled.

Two registers allow memory contents to be read and written:

- The address register is used to select the specific table entry: PRV_SET_TBL_ADDR.
- The data register contains the data to be written to the specified location: PRV_SET_TBL_DATA.

While the data register is 20 bits wide, only the 8 LSB data is used for the gamma, noise filter, and CFA filter tap memories.

The preview engine supports linear increments on reads and writes automatically. The following examples show how the programmer can read/write the memory. If data is read/written, the address pointer is automatically incremented. For random/noncontiguous reads/writes, PRV_SET_TBL_ADDR register must be modified.

Note: The address is not autoincremented when the preview engine is busy and users try to read/write the tables.

Example:

Read/write all the entires of the CFA table (using linear increment):

- WRITE (SET_TBL_ADDR, 0x1400);
- READ (SET_TBL_DATA, 0xvalue1);
- WRITE (SET_TBL_DATA, 0xvalue2);
- READ (SET_TBL_DATA, 0xvalue3);

Etc. . . .

- READ (SET_TBL_DATA, 0xvalue163F);

Read/write selective entries of the tables (have to program the address separately for each read/write)

- WRITE (SET_TBL_ADDR, 11);
- READ (SET_TBL_DATA, value11);
- WRITE (SET_TBL_ADDR, 564);
- WRITE (SET_TBL_DATA, value564);

12.5.5.2 Enable/Disable Hardware

Setting the PRV_PCR[0] ENABLE bit to 0x1 enables the preview engine. This must be done after all required registers and tables are programmed.

When the input source is the memory, the preview engine always operates in one-shot mode. In other words, after enabling the preview engine, the ENABLE bit is automatically turned off (set to 0) and only a single frame is processed from memory. In this mode, fetching and processing of the frame begin immediately on setting the ENABLE bit.

When the input source is the CCDC, the preview engine can be configured to operate in either one-shot mode or continuous mode (PRV_PCR [3] ONESHOT). Processing of the frame is depends on the timing of the CCDC. To ensure that data from the CCDC is not missed, the preview engine must be enabled before the CCDC so that it waits for CCDC data.

Note: In one-shot mode, on setting the ENABLE bit, the processing of the frame begins and the ENABLE, ONESHOT, and SOURCE bits are reset to their reset values.

When the preview engine is in continuous mode, it can be disabled by clearing the ENABLE bit during the processing of the last frame. The disable is latched in at the end of the frame in which it is written.

12.5.5.3 Events and Status Checking

The preview engine generates an interrupt at the end of each frame.

The status of this interrupt can be checked by reading the ISP_IRQ0STATUS register (or ISP_IRQ1STATUS). When the read of the register ISP_IRQ0STATUS occurs (or ISP_IRQ1STATUS), the register is not automatically reset. To reset the interrupt, a 1 must be written to the PRV_DONE_IRQ bit.

Each event that generates an interrupt can be individually mapped to ARM or DSP using the ISP_IRQ0ENABLE register (or ISP_IRQ1ENABLE). When a particular event is not enabled (for example ISP_IRQ0ENABLE[x] = 0), the correspondent status (ISP_IRQ0STATUS [x] = 1) bit is flagged if the correspondent event occurs. This has no effect on the interrupt line, but can be used by software to poll the status.

The PRV_PCR[1] BUSY status bit is set when the start of frame occurs (if the PRV_PCR[0] ENABLE bit is 1 at that time). It is automatically reset to 0 at the end of a frame. The PRV_PCR[1] BUSY status bit may be polled to determine end-of-frame status.

The PRV_PCR[31] DRK_FAIL status bit is set when dark-frame data fetched from memory arrives late. This bit can be reset by writing a 1 to the bit.

12.5.5.4 Register Accessibility During Frame Processing

There are three types of register access in the preview engine.

- Shadow registers: These registers/fields can be read and written (if the field is writable) at any time. However, the written values take effect only at the start of a frame. Reads return the most recent write, even though the settings are not used until the next start of frame.

The shadowed registers are:

- PRV_PCR
- PRV_RSDR_ADDR
- PRV_RADR_OFFSET
- PRV_DSDR_ADDR
- PRV_DRKF_OFFSET
- PRV_WSDR_ADDR
- PRV_WADD_OFFSET

- Busy-writable registers: These registers/fields can be read or written even if the module is busy. Changes to the underlying settings occur instantaneously.

  The busy-writable registers are:

  - PRV_WB_DGAIN
  - PRV_WBGAIN

- Busy-lock registers:
  - All registers EXCEPT the shadow and busy-writable registers belong to this category. Busy-lock registers cannot be written when the module is busy. Writes are allowed, but no change occurs in the registers (blocked writes from hardware perspective, but allowed writes from software perspective).
  - After the PRV_PCR[1] BUSY bit is reset to 0, busy-lock registers can be written.
  - The PRV_SET_TBL_DATA register cannot be read when the preview engine is busy, because this register is mapped to memories internally. Such reads return indeterminate data. Byte enables are not implemented for reading preview engine memories.

The ideal procedure for changing the preview engine registers is:

```
IF (PRV_PCR[1] BUSY == 0) OR IF
(EOF interrupt occurs)
    DISABLE PREVIEW ENGINE
    CHANGE REGISTERS
    ENABLE PREVIEW ENGINE
```

12.5.5.5 Interframe Operations

Between frames, it may be necessary to enable/disable functions or modify memory pointers. Because the PRV_PCR register and memory pointer registers are shadowed, these modifications can occur any time before the end of the frame, and the data is latched in for the next frame. The MPU subsystem can perform these changes on receiving an interrupt.

12.5.5.6 Summary of Constraints

The following is a list of register configuration constraints to adhere to when programming the preview engine. It can be used as a quick checklist. More detailed register setting constraints can be found in the individual register descriptions.

- A frame can only be read from memory when the SBL read port is affected to the PREVIEW module (ISP_CTRL[27] SBL_SHARED_RPORTA = 0)
- The shading compensation feature can only be used when the SBL read port is affected to the PREVIEW module (ISP_CTRL[28] SBL_SHARED_RPORTB = 0)
- If the memory output port is enabled, the memory output line offset and address should be on 32-byte boundaries.
- The output width must be less than or equal to 3312.
- The output width must be even.
- Input to the horizontal median filter must be even.

- Defect correction can only be used when noise filter is enabled
- Input height must be smaller than CCDC output height.
- The input width of the preview engine must be a multiple of the average count multiplied by the least common multiple of the odd distance and even distance of the averager.
  - (PRV_HORZ_INFO[13:0] EPH - HORZ_INFO [29:16] SPH + 1) MOD ((1 << PRV_AVE [1:0] COUNT)*LeastCommonMultiple(PRV_AVE[5:4] ODDDIST+1, PRV_AVE[3:2] EVENDIST+1)) = 0
- Input width must be at least 4 pixels smaller than CCDC output width:
  - PRV_HORZ_INFO[29:16] SPH at least 2 pixels before last pixel from CCDC
  - PRV_HORZ_INFO[13:0] EPH at least 2 pixels before last pixel from CCDC

12.5.6 Programming the Resizer

This section discusses issues related to software control of the resizer. It lists which registers are required to be programmed in different modes, how to enable and disable the resizer, and how to check the status of the resizer; discusses the different register access types; and enumerates programming constraints.

12.5.6.1 Resizer Hardware Setup/Initialization

This section discusses the configuration of the resizer required before image processing can begin.

12.5.6.1.1 Reset Behavior

On hardware reset of the Camera ISP, all registers in the resizer are reset to their reset values.

12.5.6.1.2 Register Setup

Before enabling the resizer, the hardware must be correctly configured through register writes. Table 12-58 identifies the register parameters that must be programmed before enabling the resizer.

Table 12-58. Resizer Required Configuration Parameters

Function	Configuration Required
Resizer control parameters	RSZ_CNT
I/O sizes	RSZ_OUT_SIZE
	RSZ_IN_START
	RSZ_IN_SIZE
Memory addresses	RSZ_SDR_INADD
	RSZ_SDR_INOFF
	RSZ_SDR_OUTADD
	RSZ_SDR_OUTOFF
Filter coefficients	RSZ_HFILT10 to RSZ_HFILT3130
	RSZ_VFILT10 to RSZ_VFILT3130
Edge enhancement	RSZ_YENH[17:16] ALG0

The edge-enhancement function is optional:

- If it is disabled, the rest of the RSZ_YENH register does not need to be programmed.
- If it is enabled, the edge-enhancement parameters in Table 12-59 must be programmed so that the edge-enhancement function operates correctly.

Table 12-59 can be read as:

If (**Condition** is TRUE) then

Configuration required parameters must be programmed.

Table 12-59. Resizer Conditional Configuration Parameters

Function	Condition	Configuration Required
Edge enhancement	RSZ_YENH[17:16] ALG0 = 0x1 or 0x2	RSZ_YENH[15:12] GAIN
		RSZ_YENH[11:8] SLOP
		RSZ_YENH[7:0] CORE

12.5.6.2 Enable/Disable Hardware

Setting the RSZ_PCR[0] ENABLE bit to 0x1 enables the resizer. This must be done after all required registers are programmed.

When the input source is memory, the resizer always operates in one-shot mode. In other words, after enabling the resizer, the RSZ_PCR[0] ENABLE bit is automatically turned off (set to 0) and only a single frame is processed from memory.

In this mode, fetching and processing of the frame begin immediately on setting the RSZ_PCR [0] ENABLE bit.

When the input source is the CCDC or preview engine, the resizer can be configured to operate in either one-shot mode, or continuous mode (RSZ_PCR[2] ONESHOT). Processing of theframe depends on the timing of the CCDC/Preview. To ensure that data from the CCDC or preview engine is not missed, the resizer must be enabled before to these upstream modules, so it waits for data from the CCDC or the preview engine.

When the resizer is started during an ongoing frame the enable is latched at the end of the frame it was written in.

When the resizer is in continuous mode, it can be disabled by clearing the ENABLE bit during the processing of the last frame. The disable is latched in at the end of the frame in which it was written.

12.5.6.3 Events and Status Checking

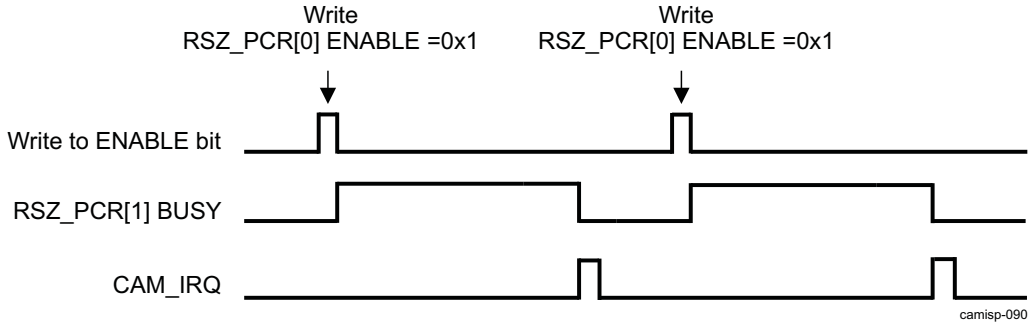
The resizer generates an interrupt event at the end of each frame.

The status of this interrupt can be checked by reading the ISP_IRQ0STATUS register (or ISP_IRQ1STATUS). When the read of the register ISP_IRQ0STATUS occurs (or ISP_IRQ1STATUS), the register is not automatically reset. To reset the interrupt, a 1 must be written to the RSZ_DONE_IRQ bit.

Each event that generates an interrupt can be individually mapped to ARM or DSP using the ISP_IRQ0ENABLE register (or ISP_IRQ1ENABLE). When a particular event is not enabled (for example ISP_IRQ0ENABLE[x] = 0), the correspondent status (ISP_IRQ0STATUS[x] = 1) bit is flagged, if the correspondent event occurs. This has no effect on the interrupt line, but can be used by software to poll the status.

The RSZ_PCR[1] BUSY status bit is set when the start of frame occurs (if the RSZ_PCR[0] ENABLE bit is 1 at that time). It is automatically reset to 0 at the end of a frame. The RSZ_PCR[1] BUSY status bit may be polled to determine end-of-frame status in oneshot mode. Figure 12-111 shows the firmware/hardware interaction. Configuration registers and filter coefficients are programmed in-between or before busy periods, before writing ENABLE to 0x1.

Figure 12-111. Firmware Interactions for Memory-Input Resizing



Note: The SBL_SDR_REQ_EXP[19:10] RSZ_EXP bit field enables the spreading of non-real time read requests over time. It is useful to avoid overflow situations when the resizer module is programmed to work from memory to memory.

12.5.6.4 Register Accessibility During Frame Processing

There are two types of register access in the resizer.

- Shadow registers: These registers/fields can be read and written (if the field is writable) at any time. However, the written values take effect only at the start of a frame. Reads return the most recent write, even though the settings are not used until the next start of frame.

  The shadowed registers are:

  – RSZ_PCR
  – RSZ_SDR_INADD
  – RSZ_SDR_INOFF
  – RSZ_SDR_OUTADD
  – RSZ_SDR_OUTOFF
- Busy-lock registers
  – All registers EXCEPT the shadowed registers belong to this category.
  – Busy-lock registers cannot be written when the module is busy. Writes are allowed, but no change occurs in the registers (blocked writes from hardware perspective; allowed write from software perspective).
  – After the RSZ_PCR[1] BUSY bit is reset to 0, the busy-lock registers can be written.

The ideal procedure for changing the resizer registers is:

```
IF (RSZ_PCR[1] BUSY == 0) OR IF
(EOF interrupt occurs)
    DISABLE RESIZER
    CHANGE REGISTERS
    ENABLE RESIZER
```

12.5.6.5 Inter-Frame Operations

Between frames, it may be necessary to modify the memory pointers before processing the next frame. Because the RSZ_PCR[0] ENABLE bit and memory pointer registers are shadowed, these modifications can occur any time before the end of the frame, and the data is get latched in for the next frame. The MPU subsystem can perform these changes on receiving an interrupt.

Note: The firmware is responsible for computing and uploading the filter coefficients. If polyphase resampling is used, a different set is required when changing between 4-tap and 7-tap modes, and with different downsampling factors; all upsampling factors can share the same set of coefficients. Do not change any busy-lock registers while the resizer is operating. Specifically, when back-to-back resizes require changes in any busy-lock registers (such as the coefficients, resizing ratios, input and output sizes), users must wait for the first resize to complete. The following section describes some scenarios where this is required.

12.5.6.5.1 Multiple Passes for Large Resizing Operations

The resizer supports multiple passes of processing for large resizing operations. "Large" has the following meanings:

- Wider output than 3312 pixels: This works only in memory input mode. Input can be partitioned into multiple resizer blocks, and each block is separately resized and stitched together. Having input/output memory line offsets, input starting pixel and starting phase are essential to make this work. The basic idea is to begin subsequent slices at exactly where previous images leave off. The starting phase and pixel registers can be programmed to this exact location.
- Larger than 4x upsampling: Resizing can be applied in multiple passes. For example, 10x upsampling can be realized by first a 4x upsampling, then a 2.5x upsampling. The first pass can be performed on-the-fly with the preview engine. The second pass can be performed only with input from memory, and for 10x digital zoom; there is time outside the active picture region to perform the second pass.
- Larger than 4:1 downsampling: Although it is rarely necessary to generate a very small image from a large image, this is supported by the hardware. For example, 10x downsampling can be realized first with 4x downsampling on-the-fly with the preview engine, then 2.5x downsampling in the memory-input path. There may not be much time outside the active data region for the second pass, but since the image is already reduced to 1/16 of its original size, not much time is necessary. Typically, sensor or video input has 10 ~ 20 % of usable vertical blanking.

For all these scenarios, the second pass can be configured and initiated from an interrupt service routine triggered by the resizer end-of-frame interrupt: ISP_IRQ0ENABLE[24] RSZ_DONE_IRQ (or ISP_IRQ1ENABLE).

12.5.6.5.2 Processing Time Calculation

The time calculated below is the time it takes for all resizes where the input source is memory (second pass when doing a 10x resize in preview mode; in the case of a 10x resize in preview mode, the first pass is hidden behind the time it takes to capture and process the image from the sensor based on cam_pclk and the number of lines resized).

The following equation can be used to determine the processing time of the resizer when the input is from memory and therefore how much time it takes before it can switch back to preview input mode:

Time = [W x bytes_per_pixel x H]/[L3/2]

Where:

/* If the input is YUV422 and horizontal downsampling is performed: */

if ((RSZ_CNT[27] INPTYP ==0) && ((RSZ_CNT[9:0] HRSZ + 1) >256))

W = average (input_width, output width); /* output width includes extra 4 pixels if edge enhancement is enabled*/

else

W = max(input width, output width); /*output width includes extra 4 pixels if edge enhancement is enabled*/

$$input_width = RSZ_IN_SIZE[12:0]HORZ$$
$$output_width = RSZ_OUT_SIZE[11:0]HORZ$$
$$input_height = RSZ_IN_SIZE[28:16]VERT$$
$$bytes_per_pixel = \begin{cases} 1, \ when\ RSZ_CNT[27]INPTYP = 1\ (color\ separate) \\ 2, \ when\ RSZ_CNT[27]INPTYP = 0\ (YUV422) \end{cases}$$

camisp-E097

This time is the baseline steady state calculation of the hardware and does not include the time it takes for the hardware to fetch the first input and fill the pipeline. It also does not include the time spent from the last output from the resizer to get back to memory when the resizer interrupt occurs.

However, these beginning and ending times are relatively negligible.

Depending on real-time constraints, this processing time may be much faster than is required. (data fetches can be delayed from memory to free more bandwidth for use by other system peripherals; see Section 12.5.9.5.2, *Input from Memory*.

12.5.6.6 Summary of Constraints

The following is a list of register configuration constraints to adhere to when programming the resizer. It can be used as a quick checklist. More detailed register setting constraints can be found in the individual register descriptions.

- Vertical and horizontal resize ratio values must be within the following range: [64..1024].
- Output width:
  - Must be within the maximum limit:
    - width 3312 (if vertical resize value is in range: (RSZ_CNT[19:10] VRSZ + 1) = [64..512])
    - width 1650 (if vertical resize value is in range: (RSZ_CNT[19:10] VRSZ + 1) = [513..1024])
  - Must be even
  - Must be a multiple of 16 bytes (for vertical upsizing)
- When input is from preview engine/CCDC:
  - The input height and width must be <= the output of the preview engine/CCDC.
  - The input address and offset must be zero.
  - The input can not be color-separated data.
- If the source is memory:
  - The vertical start pixel must be zero.
  - The horizontal start pixel must be within the range: 0:15 for color interleaved, 0:31 for color separate data.
  - The memory output line offset and address must be on 32-byte boundaries.
- Input height and width MUST adhere to the equations in Table 12-60.

Table 12-60. How to Set Input Height and Width

	8-phase, 4-tap mode	4-phase, 7-tap mode
RSZ_IN_SIZE [12:0] HORZ	(32*sph + (ow - 1)*hrsz + 16) >> 8 + 7	(64*sph + (ow - 1)*hrsz + 32) >> 8 + 7
RSZ_IN_SIZE [28:16] VERT	(32*spv + (oh - 1)*vrsz + 16) >> 8 + 4	(64*spv + (oh - 1)*vrsz + 32) >> 8 + 7

Where:

- sph = Start phase horizontal (RSZ_CNT[22:20] HSTPH)
- spv = Start phase vertical (RSZ_CNT[25:23] VSTPH)
- ow = Output width (RSZ_OUT_SIZE[11:0] HORZ + extra)

- oh = Output height (RSZ_OUT_SIZE[27:16] VERT)
- hrsz = Horizontal resize value (RSZ_CNT[9:0] HRSZ + 1)
- vrsz = Vertical resize value (RSZ_CNT[19:10] VRSZ +1)

extra = 0 when RSZ_YENH[17:16] ALGO = 0 (edge enhancement disabled)

extra = 4 when RSZ_YENH[17:16] ALGO != 0 (edge enhancement enabled)

Note: Normally, (for example, for a QVGA display or encoded PAL video), the output size, not the input size, matters. The image provided by preview/CCDC/memory must have an adequate output size: at least RSZ_IN_SIZE[12:0] HORZ x RSZ_IN_SIZE[28:16] VERT. If the image is bigger, the resizer can crop extra pixels.

The phase is usually computed to keep the center of the image at the same location. This permits a natural-looking continuous digital zoom.

For this reason, Table 12-60 explains how to compute RSZ_IN_SIZE[12:0] HORZ , RSZ_IN_SIZE[28:16] VERT, not how to compute the output size.

12.5.7 Programming the H3A

This section discusses issues related to software control of the H3A module. It lists which registers are required to be programmed in different modes, how to enable and disable the H3A, and how to check the status of the H3A. It also discusses the different register access types and enumerates programming constraints.

12.5.7.1 Hardware Setup/Initialization

This section discusses the configuration of the H3A required before image processing can begin.

12.5.7.1.1 Reset Behavior

On hardware reset of the Camera ISP, all registers in the H3A are reset (set to their reset values).

12.5.7.1.2 Register Setup

For register configuration, the AF engine and the AEW engine of the H3A can be independently be configured. Because there are separate enable bits for each engine, so this section discusses the AF engine and the AEW engine separately.

12.5.7.1.2.1 AF Engine

Before enabling the AF engine, the hardware must be correctly configured through register writes. Table 12-61 identifies the register parameters that must be programmed before enabling the AF engine of the H3A.

Table 12-61. AF Engine Required Configuration Parameters

Function	Configuration Required
AF optional preprocessing	H3A_PCR[2] AF_MED_EN
	H3A_PCR[1] AF_ALAW_EN
AF mode configuration	H3A_PCR[13:11] RGBPOS
	H3A_PCR[14] FVMODE

Table 12-61. AF Engine Required Configuration Parameters (continued)

Function	Configuration Required
Paxel start and size information	H3A_AFPAX1
	H3A_AFPAX2
	H3A_AFPAXSTART
	H3A_AFIIRSH
Memory address	H3A_AFBUFST
Filter coefficients	H3A_AFCOEF010 to H3A_AFCOEF1010

The horizontal median filter function is optional.

If it is disabled, the H3A_PCR[10:3] MED_TH does not need to be programmed.

However, if it is enabled, the H3A_PCR[10:3] MED_TH parameter must be programmed so that the horizontal median filter function operates correctly.

Table 12-62 can be read as:

If (**Condition** is TRUE) then

Configuration required parameters must be programmed

Table 12-62. AF Engine Conditional Configuration Parameters

Function	Condition	Configuration Required
Horizontal median filter	H3A_PCR[2] AF_MED_EN	H3A_PCR[10:3] MED_TH

12.5.7.1.2.2 AEW Engine

Before enabling the AEW engine, the hardware must be correctly configured through register writes. Table 12-63 identifies the register parameters that must be programmed before enabling the AEW engine of the H3A.

Table 12-63. AEW Engine Required Configuration Parameters

Function	Configuration Required
AEW optional preprocessing	H3A_PCR[17] AEW_ALAW_EN
Saturation limit	H3A_PCR[31:22] AVE2LMT
Window start and size information	H3A_AEWWIN1
	H3A_AEWINSTART
	H3A_AEWINBLK
	H3A_AEWSUBWIN
Memory address	H3A_AEWBUFST

12.5.7.2 Enable/Disable Hardware

Setting the H3A_PCR[0] AF_EN bit enables the AF engine, and the H3A_PCR[16] AEW_EN bit enables the AEW engine. This should be done after all required registers are programmed.

The H3A always operates in continuous mode. Because the input to the H3A module is the video-port interface of the CCDC, processing of the frame depends on the timing signals from the CCDC. To ensure that data from the CCDC is not missed, the H3A should be enabled before the CCDC so that it waits for CCDC data.

The AF engine or the AEW engine can be disabled by clearing the H3A_PCR[0] AF_EN or H3A_PCR [16] AEW_EN bit, respectively, during the processing of the last frame. The disable is latched in at the end of the frame in which it is written.

12.5.7.3 Event and Status Checking

Both the AF engine and the AEW engine generate an interrupt at the end of processing each frame. These interrupt events can be sent to CAM_IRQ0 or CAM_IRQ1 by setting the H3A_AWB_DONE_IRQ or H3A_AF_DONE_IRQ bits in the ISP_IRQ0ENABLE enable register (or ISP_IRQ1ENABLE).

The status of these interrupts can be checked by reading the ISP_IRQ0STATUS register (or ISP_IRQ1STATUS). When the read of the register ISP_IRQ0STATUS occurs (or ISP_IRQ1STATUS), the register is not automatically reset. To reset the interrupt, a 1 must be written to the corresponding bit.

Each event that generates an interrupt can be individually mapped to ARM or DSP using the ISP_IRQ0ENABLE register (or ISP_IRQ1ENABLE). When a particular event is not enabled (for example ISP_IRQ0ENABLE[x] = 0), the correspondent status (ISP_IRQ0STATUS [x] = 1) bit is flagged if the correspondent event occurs. This has no effect on the interrupt line, but can be used by software to poll the status.

12.5.7.4 Register Accessibility During Frame Processing

There are two types of register access in the H3A module:

- Shadow registers: These registers/fields can be read and written (if the field is writable) at any time. However, the written values take effect only at the start of a frame. Reads return the most recent write, even though the settings are not used until the next start of frame.

  The shadowed registers are:

  – H3A_PCR
  – H3A_AFBUFST
  – H3A_AEWBUFST
- Busy-lock registers:
  – All registers EXCEPT the shadowed registers belong to this category.
  – Busy-lock registers cannot be written when the module is busy. Writes are allowed, but no change occurs in the registers (blocked writes from hardware perspective, but allowed writes from software perspective).
  – After the busy bit in the PCR register is reset to 0, the busy-lock registers can be written (H3A_PCR [15] BUSYAF for AF registers and H3A_PCR [18] BUSYAEAWB for AE/AWB registers).

The ideal procedure for changing the H3A registers is:

IF (H3A_PCR [15] BUSYAF == 0 OR H3A_PCR [18] BUSYAEAWB == 0) OR IF (EOF interrupt occurs)

DISABLE AF or AE/AWB

CHANGE REGISTERS AF or AE/AWB

ENABLE AF or AE/AWB

12.5.7.5 Interframe Operations

Between frames, it may be necessary to modify the memory pointers before processing the next frame. Since the H3A_PCR and memory pointer registers are shadowed, these modifications can occur any time before the end of the frame, and the data is latched in for the next frame. The MPU subsystem can perform these changes on receiving an interrupt.

12.5.7.6 Summary of Constraints

The following is a list of register configuration constraints to adhere to when programming the H3A. It can be used as a quick checklist. More detailed register setting constraints can be found in the individual register descriptions.

- The output addresses must be on 64-byte boundaries.

  AF Engine:
- The paxel horizontal start value must be greater than or equal to the IIR horizontal start position.

- The width and height of the paxels must be even numbers.
- The minimum width of the autofocus paxel must be 6 pixels.
- Paxels cannot overlap the last pixel in a line.
- Paxels must be adjacent to one another.

AEW Engine:

- The width and height of the windows must be even numbers.
- Subsampling windows can only start on even numbers.
- The minimum width of the AE/AWB windows is 6 pixels.

12.5.8 Programming the Histogram

This section discusses issues related to software control of the histogram module. It lists which registers are required to be programmed in different modes, how to enable and disable the histogram, and how to check the status of the histogram; discusses the different register access types; and enumerates programming constraints.

12.5.8.1 Hardware Setup/Initialization

This section discusses the configuration of the histogram required before image processing can begin.

12.5.8.1.1 Reset Behavior

On hardware reset of the Camera ISP, all registers in the histogram are reset to their reset values. However, since the histogram output memory is stored in internal memory, its contents do not have reset values. If the reset is a chip-level power-on reset (reset after power is applied), the contents of this memory are unknown. If the reset is a Camera ISP module reset (when power remains active), the contents of this memory remain the same as before the reset.

12.5.8.1.2 Reset of Histogram Output Memory

Clear the output memory before enabling the histogram. This can be done two ways:

- Writing zeros to the memory through software
- If the HIST_CNT [7] CLR bit is set, reading the memory causes it to be reset after the read.

Reads and writes to the output memory are blocked when the HIST_PCR [1] BUSY bit is 1.

12.5.8.1.3 Register Setup

Before enabling the histogram module, the hardware must be correctly configured through register writes. Table 12-64 identifies the register parameters that must be programmed before enabling the histogram.

Table 12-64. Histogram Required Configuration Parameters

Function	Configuration Required
Histogram Control Bits	HIST_CNT [3] SOURCE
	HIST_CNT[6] CFA
	HIST_CNT [5:4] BINS
	HIST_CNT [2:0] SHIFT
	HIST_CNT [7] CLR
White Balance Gain	HIST_WB_GAIN
Region n Size and position (n = 0)	HIST_Rn_HORZ
	HIST_Rn_VERT

Table 12-65 can be read as:

If (**Condition** is TRUE) then

Configuration required parameters should be programmed

Table 12-65. Histogram Conditional Configuration Parameters

Function	Condition	Configuration Required
Input from memory	HIST_CNT [3] SOURCE = 0x1	HIST_CNT [8] DATSIZ
		HIST_RADD
		HIST_RADD_OFF
		HIST_H_V_INFO
Less than 256 bins	HIST_CNT [5:4] BINS 3	HIST_Rn_HORZ (n = 1)
		HIST_Rn_VERT (n = 1)
Less than 128 bins	HIST_CNT [5:4] BINS 2	HIST_Rn_HORZ (n = 2)
		HIST_Rn_VERT (n = 2)
		HIST_Rn_HORZ (n = 3)
		HIST_Rn_VERT (n = 3)

12.5.8.2 Enable/Disable Hardware

Setting the HIST_PCR [0] ENABLE bit enables the histogram module. This must be done after all required registers are programmed and the output memory has been cleared.

When the input source is the memory, the histogram module always operates in one-shot mode. In other words, after enabling the histogram, the ENABLE bit is automatically turned off (set to 0) and only a single frame is processed from memory. In this mode, fetching and processing of the frame begin immediately on setting the ENABLE bit.

When the input source is the CCDC, the histogram always operates in continuous mode. Processing of the frame depends on the timing of the CCDC. To ensure that data from the CCDC is not missed, the histogram must be enabled before CCDC so it waits for CCDC data..

When the histogram is in continuous mode, it can be disabled by clearing the ENABLE bit during the processing of the last frame. The disable is latched in at the end of the frame in which it is written.

12.5.8.3 Event and Status Checking

The histogram generates an interrupt at the end of each frame.

The status of this interrupt can be checked by reading the ISP_IRQ0STATUS register (or ISP_IRQ1STATUS). When the read of the register ISP_IRQ0STATUS occurs (or ISP_IRQ1STATUS), the register is not automatically reset. To reset the interrupt, a 1 must be written to the HIST_DONE_IRQ bit.

Each event that generates an interrupt can be individually mapped to ARM or DSP using the ISP_IRQ0ENABLE register (or ISP_IRQ1ENABLE). When a particular event is not enabled (for example ISP_IRQ0ENABLE[x] = 0), the corresponding status (ISP_IRQ0STATUS [x] = 1) bit is flagged if the corresponding event occurs. This has no effect on the interrupt line, but can be used by software to poll the status.

The HIST_PCR [1] BUSY status bit is set when the start of frame occurs (if the HIST_PCR [0] ENABLE bit is 1 at that time). It is automatically reset to 0 at the end of a frame. The HIST_PCR [1] BUSY status bit may be polled to determine the end-of-frame status.

12.5.8.4 Register Accessibility During Frame Processing

There are two types of register access in the histogram module.

- Shadow registers: These registers/fields can be read and written (if the field is writable) at any time. However, the written values take effect only at the start of a frame. Reads return the most recent write, even though the settings are not used until the next start of frame.

  The shadowed registers are:

  – HIST_PCR
  – HIST_RADD
  – HIST_RADD_OFF
- Busy-lock registers
  – All registers EXCEPT the shadowed registers belong to this category.
  – Busy-lock registers cannot be written when the module is busy. Writes are allowed, but no change occurs in the registers (blocked writes from hardware perspective; allowed write from software perspective).
  – After the HIST_PCR [1] BUSY bit is reset to 0, the busy-lock registers can be written.
  – The HIST_DATA register cannot be read when the histogram is busy, because since this register is mapped to memories internally. Such reads return indeterminate data. Byte enables are not implemented for reading the histogram memory.

The ideal procedure for changing the histogram registers is:

IF (HIST_PCR [1] BUSY == 0) OR IF (EOF interrupt occurs)

DISABLE HISTOGRAM

CHANGE REGISTERS

ENABLE HISTOGRAM

12.5.8.5 Interframe Operations

Between frames read from memory, it may be necessary to modify the input memory pointers before processing the next frame. Since the H3A_PCR and memory pointer registers are shadowed, these modifications can take place any time before the end of the frame, and the data is latched in for the next frame. The MPU Subsystem can perform these changes on receiving an interrupt.

If continuous frames are processed without clearing the histogram output memory, the bin counters contain the counts of however many images are processed since they were last cleared. To read the bin counters for each frame, the bin counters must be read after each frame is completed, but before the next frame begins (since the counters cannot be read while the HIST_PCR [1] BUSY bit is 1).

If the input source is memory (one-shot mode), the HIST_PCR [0] ENABLE bit must be set once for the frame, and after the frame is completed, the bin counters can be read/cleared before enabling the next frame.

When the input source is the video-port interface of the CCDC (continuous mode), the HIST_PCR [0] ENABLE bit must be set to enable processing of the frame, and cleared after the frame processing begins (the disable is latched in at the end of the frame). This procedure allows only one frame to be processed. After the frame is completed, the bin counters can be read/cleared before enabling the histogram for the next frame.

12.5.8.6 Summary of Constraints

The following is a list of register configuration constraints to adhere to when programming the histogram. It can be used as a quick checklist. More detailed register setting constraints can be found in the individual register descriptions.

- The input address and line offset must be on 32-byte boundaries.
- A region dimension of 1 (horizontal or vertical or both) is not allowed.

12.5.9 Programming the Central-Resource SBL

This section discusses issues related to the software control of the central-resource SBL. It lists which registers are required to be programmed in different modes, describes how to check the status of the central resource SBL overflow bits, and enumerates programming constraints.

The central-resource SBL controls data interactions between the Camera ISP modules and the interface to memory. A small number of registers configure maximum data-read bandwidth in memory-to-memory operations.

12.5.9.1 Hardware Setup/Initialization

This section discusses the configuration of the central-resource SBL required before image processing can begin.

12.5.9.1.1 Reset Behavior

On hardware reset of the Camera ISP, all registers in the SBL are reset to their reset values.

12.5.9.1.2 Register Setup

Before enabling any of the Camera ISP modules, the hardware must be correctly configured through register writes to the Camera ISP registers. If the preview engine, resizer, or the histogram is reading from memory, the SBL_SDR_REQ_EXP register must be programmed. The values programmed in each of the three fields (PRV_EXP, RSZ_EXP, HIST_EXP) determines the number of clock cycles required to allow two consecutive read requests from the module.

12.5.9.2 Enable/Disable Hardware

The central-resource SBL functionality is always enabled, unless the PRCM idles the clocks to the Camera ISP.

12.5.9.3 Event and Status Checking

The SBL generates one interrupt for the write buffer overflow events listed below. Software must check which overflow event has raised the interrupt request, by reading the SBL_PCR register.

See Table 12-66.

Table 12-66. SBL Write-Buffer Overflow Events

Bit	Event Description
SBL_PCR [26] CSIB_WBL_OVF	CSIb write-buffer memory overflow
SBL_PCR [25] CSIA_WBL_OVF	CSIa write-buffer memory overflow
SBL_PCR [24] CCDCPRV_2_RSZ_OVF	CCDC/PREVIEW to RESIZER input overflow This bit is set if the RESIZER input source is sent to CCDC/PREVIEW engine when the active data (to be resized) has already showed up at the resizer interface. In such a case, resizing for this frame cannot take place and the bit is set. This scenario can happen when a resize of > 4x is required per frame. Therefore, the RESIZER must operate in two passes. In the first pass, the input data from CCDC/PREVIEW is directly resized and written to memory. In the second pass, the resized data from the first pass is resized again. The next frame from the CCDC/PREVIEW engine should start only after the second pass on the previous frame is complete. This bit indicates the failure status.
SBL_PCR [23] CCDC_WBL_OVF	CCDC write-buffer memory overflow
SBL_PCR [22] PRV_WBL_OVF	PREVIEW write-buffer memory overflow
SBL_PCR [21] RSZ1_WBL_OVF	RESIZER line 1 write-buffer memory overflow
SBL_PCR [20] RSZ2_WBL_OVF	RESIZER line 2 write-buffer memory overflow

Table 12-66. SBL Write-Buffer Overflow Events (continued)

Bit	Event Description
SBL_PCR [19] RSZ3_WBL_OVF	RESIZER line 3 write-buffer memory overflow
SBL_PCR [18] RSZ4_WBL_OVF	RESIZER line 4 write-buffer memory overflow
SBL_PCR [17] H3A_AF_WBL_OVF	H3A AF write-buffer memory overflow
SBL_PCR [16] H3A_AEAWB_WBL_OVF	H3A AE/AWB write-buffer memory overflow

The status of this interrupt can be checked by reading the ISP_IRQ0STATUS register (or ISP_IRQ1STATUS). When the read of the register ISP_IRQ0STATUS occurs (or ISP_IRQ1STATUS), the register is not automatically reset. To reset the interrupt, a 1 must be written:

- To the corresponding bit(s) in the SBL_PCR registers
- To the OVF_IRQ bit in the ISP_IRQ0STATUS register (or ISP_IRQ1STATUS)

Each event that generates an interrupt can be individually mapped to ARM or DSP using the ISP_IRQ0ENABLE register (or ISP_IRQ1ENABLE). When a particular event is not enabled (for example ISP_IRQ0ENABLE[x] = 0), the correspondent status (ISP_IRQ0STATUS [x] = 1) bit is flagged if the correspondent event occurs. This has no effect on the interrupt line, but can be used by software to poll the status.

The SBL flags no read buffer logic underflow events. These are signaled by the reading module. Table 12-67 lists the read port and corresponding events if they exist.

Table 12-67. SBL Read-Buffer Underflow Events

Read port	Description
CCDC faulty pixel	When faulty-pixel correction is used, the pixel frequency is imposed by the camera clock. This imposes read-time constraints to the faulty-pixel table read. When the table read was too slow, the CCDC_FPC [16] FPERR bit is set to 1 and no more faulty pixels are processed for that frame. This error generates an interrupt that can be mapped to DSP or ARM by setting the CCDC_ERR_IRQ bit in the ISP_IRQ0ENABLE register (or ISP_IRQ1ENABLE). To clear this interrupt, clear the CCDC_FPC [16] FPERR bit before clearing the ISP_IRQ0STATUS [11] CCDC_ERR_IRQ bit (or ISP_IRQ1STATUS).
CCDC lens-shading compensation	There must be adequate memory bandwidth if this feature is enabled. If the data fetched from memory arrives late, then the CCDC_LSC_PREFETCH_ERROR event is triggered and an interrupt generated.
Preview dark frame	There must be adequate memory bandwidth if this feature is enabled. If the data fetched from memory arrives late, the PRV_PCR [31] DRK_FAIL status bit is set to indicate a fail. No interrupt is generated by this event.
Preview image from memory	No error can occur on this port because the preview module stops processing when no image data is ready.
Resizer image from memory	No error can occur on this port because the resizer module stops processing when no image data is ready.
Histogram image from memory	No error can occur on this port because the histogram module stops processing when no image data is ready.
CSI1 image from memory	No error can occur on this port because the CSI1 module stops processing when no image data is ready.

12.5.9.4 Register Accessibility During Frame Processing

The central resource SBL registers are all busy-writable registers.

- Busy-writable registers
  - These registers/fields can be read or written even if the module is busy. Changes to the underlying settings occur instantaneously.

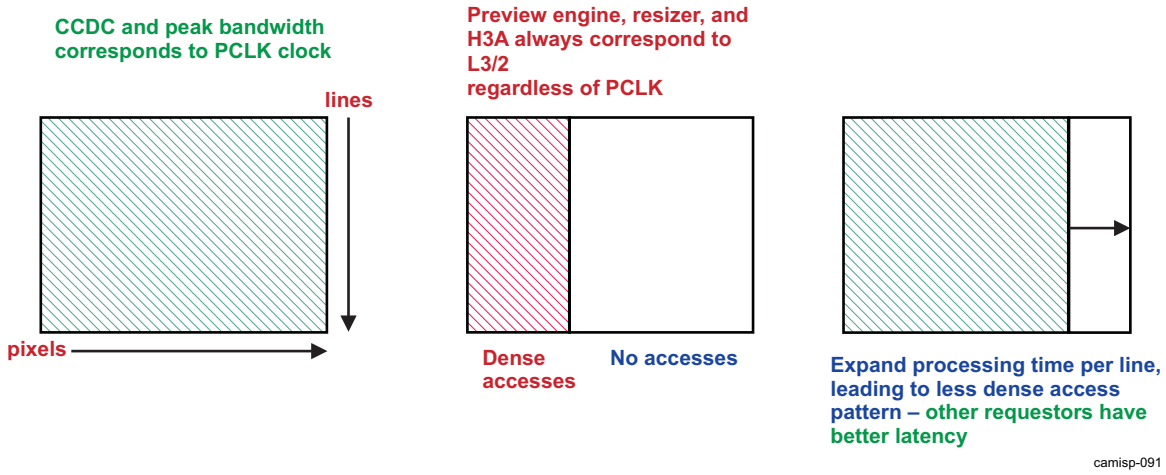
12.5.9.5 Camera ISP Bandwidth Adjustments

For memory-to-memory operation, the Camera ISP processes data at the highest possible data rate. If this processing returns results long before real-time deadlines, the performance of other peripherals in the system may be negatively affected. The Camera ISP offers two kinds of adjustments that can slow down data processing in this situation. One can be made when the sensor input to the CCDC is the input source, and the other can be made when the memory is the source of the input image.

12.5.9.5.1 Input From CCDC Video-Port Interface

The video-port interface delivers data at a rate independent of the pixel clock when the data reformatter is enabled. By default, this rate is set to 83 MHz, which is fast enough to support a parallel interface clock of 75 MHz or a CSI clock of 83 MHz. When the pixel clock is at a lower frequency, it is unnecessary for the video-port interface to operate at such a high frequency. The CCDC_FMTCFG [18:16] VPIF_FRQ field of the CCDC can be programmed to reduce the rate at which the video port delivers new data to the other modules (Preview, H3A, and histogram). In effect, this register indirectly controls the output bandwidth of the preview engine, resizer, and H3A. Depending on the input sensor clock, Users can set this field appropriately and balance the bandwidth requirements to memory. Figure 12-112 demonstrates how this register can expand processing time per line for lower PCLK frequencies.

Figure 12-112. Video-Port Interface Bandwidth Balancing



12.5.9.5.2 Input From Memory

When the input image is from memory, data is fetched from memory and processed at a steady state rate of 166 MB/sec. Depending on the image size and real-time deadline for each frame, this may be much faster than necessary. Such activity can also starve other processes in the system. Internally, when a CAMERA ISP module receives input from memory, the CAMERA ISP makes a read request to the L3 interconnect whenever there is available memory in its internal buffers.

The SBL_SDR_REQ_EXP register can be programmed to control the rate at which a Camera ISP module (Preview, resizer, or histogram) reads the input frame from memory. This indirectly controls the output bandwidth of the preview engine and resizer. Depending on the size of the images and the real-time deadlines, users can set this field appropriately and balance the bandwidth requirements to memory.

The minimum number of cycles (L3) in between read requests used to program the SBL_SDR_REQ_EXP register can be determined based on the frame size and real-time requirement using the following equation:

Number of cycles/request = (DMA cycles/frame) / (DMA read requests/frame)

In the previous equation, (DMA cycles/frame) is based on a real-time requirement. For example, if the real-time requirement is a frame rate of 1/30 sec and the L3 clock is 166 MHz, this can be calculated as:

(DMA cycles/frame) = L3 clock * frame rate = 166 MHz * 1/30 = 5.53M cycles

The (DMA read requests/frame) is based on the frame size and the alignment in memory. Assuming a VGA (640 x 480) frame size and optimal alignment conditions:

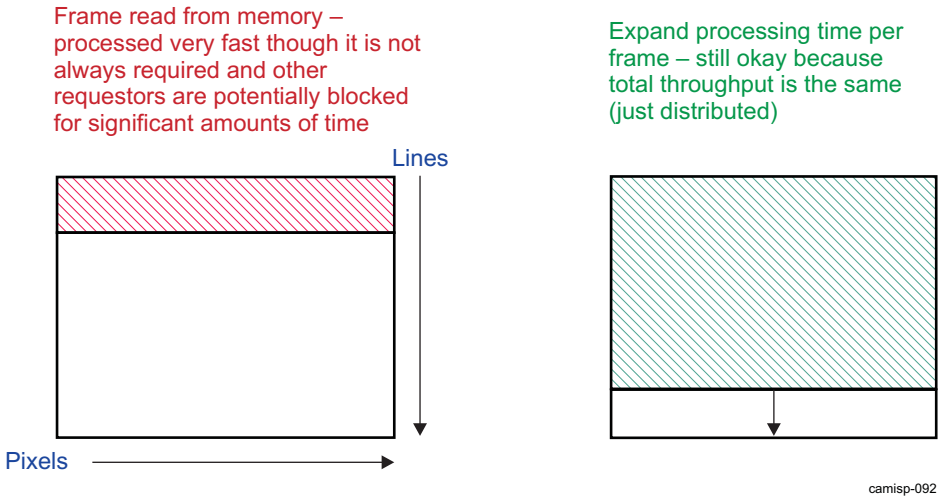
(DMA read requests/frame) = Transfers per line * number of lines = 640 pix/line*2 bytes/pix/256 bytes/xfer * 480 lines = 2400 requests/frame

In this example, the final equation can now be solved:

Number of cycles/request = 5.53M cyles/2400 requests = 2306 cycles/request

Figure 12-113 demonstrates how this register can expand processing time for lower real-time requirements.

Figure 12-113. Memory Read Bandwidth Balancing



The maximum values that can be written to the SBL_SDR_REQ_EXP register for the different read requesters is 1023. For the histogram and preview engine, this should be sufficient for the typical size of RAW data frames. However, because the resizer can read a variety of video frame sizes, the field for the resizer is internally multiplied by 1024. Therefore, for this example, the SBL_SDR_REQ_EXP[19:10] RSZ_EXP bit field can be programmed to FLOOR(2306/1024) = 2.

The previous equations provide an estimate or a starting point for programming this register. Depending on the system loads and available bandwidth, it may be necessary to reduce this number to compensate for a heavily loaded system.

Note: The granularity for the resizer is 1024, and for the preview and HIST it is 32.

12.5.10 Programming the Circular Buffer (CBUFF)

12.5.10.1 Hardware Setup/Initialization

This section discusses the configuration of the circular buffer required before address translation can begin.

12.5.10.2 Reset Behavior

Upon hardware reset of the circular buffer, all of the registers in the circular buffer are reset to their reset values.

12.5.10.3 Register Setup

All registers of the circular buffer to be used (CBUFFx, x=0 or 1) have to be initialized for correct operation.

The CBUFFx_START and CBUFFx_END register define the virtual address range managed by the circular buffer. It usually corresponds to the address region where one image frame is written by the Camera ISP.

The window count and size are set through the CBUFFx_CTRL [9:8] WCOUNT and CBUFFx_WINDOWSIZE registers. The window size usually depends on the utilization of the buffer. 8 or 16 video lines correspond to a current size for JPEG video compression. A higher window count provides better latency related overflow protection.

When the Camera ISP accesses data in an incremental addressing scheme, the next window is never used. In this case the overflow event generation, when the processor window falls into the "next window", can be disabled by setting the CBUFFx_CTRL [3] ALLOW_NW_EQ_CPUW flag.

When the 2D addressing capability isn't used the CBUFFx_THRESHOLD register is set to the window size. Otherwise it is set to a smaller value depending on the buffer organization. For example, when each window corresponds to 8 lines by 4096 pixel but the Camera ISP only send lines of 2560 pixels the CBUFFx_WINDOWSIZE=8*4096 and CBUFFx_THRESHOLD=8*2560.

When the register setup is completed the module is enabled using the CBUFFx_CTRL [0] EN bit.

It can be disabled by clearing the CBUFFx_CTRL [0] EN bit. This must only be done when there are no more outstanding requests to the virtual space managed by CBUFFx. All internal FSMs and counters of the circular buffer are reset when it is disabled. Pending interrupts are not affected.

12.5.10.4 Event and status Checking

12.5.10.4.1 Interrupts

All events generated by the circular buffer are mapped to an unique event at Camera ISP level: CBUFF_IRQ,

The CBUFF module event can be mapped to the MPU SS or to the IVA SS.

The CBUFF_IRQ bit in the ISP_IRQ0ENABLE [21] CBUFF_IRQ register control whether the CBUFF module event triggers an interrupt to the MPU SS. The CBUFF_IRQ bit in the ISP_IRQ0ENABLE [21] CBUFF_IRQ register control whether the CBUFF module event triggers an interrupt to the IVA2.2 SS.

When an event has been triggered the ISP_IRQ0STATUS [21] CBUFF_IRQ bit is set (or ISP_IRQ1STATUS). SW must than read the CBUFF_IRQSTATUS register to know which circular buffer event has triggered the interrupt. SW must clear the event first in the circular buffer module by writing 1 to the proper bit in CBUFF_IRQSTATUS register and then clear the event at Camera ISP level by writing 1 to the ISP_IRQ0STATUS [21] CBUFF_IRQ bit (or ISP_IRQ1STATUS). If another event is pending at circular buffer level, the CBUFF_IRQ interrupt is triggered again.

12.5.10.4.2 Status Checking

The event status can be checked through the CBUFFx_READY_IRQ, CBUFFx_INVALID_IRQ and CBUFFx_OVR_IRQ bits in the CBUFF_IRQSTATUS registers.

In addition to those status bits the circular buffer module provides read only access to the "current window", "next window" and "CPU windows" indexes through the CBUFFx_STATUS register. The "CPU window" index can for example be used by the processor to compute the address of the physical buffer. Those indexes can also be used to evaluate latency margins.

12.5.10.5 Register Accessibility During Frame Processing

All registers are Busy-writeable registers. These registers/fields can be read or written even if the module is busy. Changes to the underlying settings takes place instantaneously. However the module behavior is unpredictable when registers are changed during processing.

For correct operation software must follow the following steps:

- Disable all accesses to the virtual space managed by CBUFFx. For example when the circular buffer relocates data provided by the CCDC module SW must disable the CCDC module and check SBL status registers to make sure there are no more outstanding transactions.
- Disable circular buffer x by clearing the CBUFFx_CTRL [0] ENABLE bit.
- Change the configuration.
- Re-enable CBUFFx by setting the CBUFFx_CTRL [0] ENABLE bit.

12.5.10.6 Operations

A CBUFFx_READY_IRQ event is generated each time processor can read data from the circular buffer. Processor can clear the event when it starts processing the data to avoid masking of other events. Processor can keep trace of the location on the data internally or use the circular buffer registers to compute it.

The formula used is:

ADDR = CBUFFx_STATUS [3:0] CPUW x CBUFFx_WINDOWSIZE + CBUFFx_START

When processor is done with processing, it must free the buffer by setting the CBUFFx_CTRL [2] DONE bit. Otherwise an overflow event may occur.

Note that the circular buffer does not keep trace of end of frame events. They have to be managed by the processor using the end of frame event of the module that writes into the circular buffer. At the end of the frame there may remain data in the "current write" and "next write" windows. For example, when the window size is set to 8 lines and the image size is 20 lines only 2 window ready events are generated for a linear addressing scheme. The remaining 4 lines can be read after the end of frame event.

No automatic reset of the CBUFF FSM occurs at the end of the Camera ISP frame. Software must reset the CBUFF by clearing the CBUFFx_CTRL [0] ENABLE bit when the frame has been completely processed. A new frame can only start when CBUFFx_CTRL [0] ENABLE has been set.

12.6 Camera ISP Register Manual

12.6.1 Camera ISP Instance Summary

Table 12-68. Camera ISP Instance Summary

Module Name	Base Address (Hex)	Size
ISP	0x480B C000	256 bytes
ISP_CBUFF	0x480B C100	256 bytes
ISP_CSI1B	0x480B C400	512 bytes
ISP_CCDC	0x480B C600	512 bytes
ISP_HIST	0x480B CA00	512 bytes
ISP_H3A	0x480B CC00	512 bytes
ISP_PREVIEW	0x480B CE00	512 bytes
ISP_RESIZER	0x480B D000	512 bytes
ISP_SBL	0x480B D200	512 bytes
ISP_CSI2A	0x480B D800	1024 bytes
CSI2PHY_SCP	0x480B D970	512 bytes

12.6.2 Camera ISP Registers

12.6.2.1 Camera ISP Register Summary

Table 12-69. ISP Register Summary

Register Name	Type	Register Width (Bits)	Address Offset	Physical Address
ISP_REVISION	R	32	0x0000 0000	0x480B C000
ISP_SYSCONFIG	RW	32	0x0000 0004	0x480B C004
ISP_SYSSTATUS	R	32	0x0000 0008	0x480B C008
ISP_IRQ0ENABLE	RW	32	0x0000 000C	0x480B C00C
ISP_IRQ0STATUS	RW	32	0x0000 0010	0x480B C010
ISP_IRQ1ENABLE	RW	32	0x0000 0014	0x480B C014
ISP_IRQ1STATUS	RW	32	0x0000 0018	0x480B C018
TCTRL_GRESET_LENGTH	RW	32	0x0000 0030	0x480B C030
TCTRL_PSTRB_REPLAY	RW	32	0x0000 0034	0x480B C034
ISP_CTRL	RW	32	0x0000 0040	0x480B C040
TCTRL_CTRL	RW	32	0x0000 0050	0x480B C050
TCTRL_FRAME	RW	32	0x0000 0054	0x480B C054
TCTRL_PSTRB_DELAY	RW	32	0x0000 0058	0x480B C058
TCTRL_STRB_DELAY	RW	32	0x0000 005C	0x480B C05C
TCTRL_SHUT_DELAY	RW	32	0x0000 0060	0x480B C060
TCTRL_PSTRB_LENGTH	RW	32	0x0000 0064	0x480B C064
TCTRL_STRB_LENGTH	RW	32	0x0000 0068	0x480B C068
TCTRL_SHUT_LENGTH	RW	32	0x0000 006C	0x480B C06C

12.6.2.2 Camera ISP Register Description

Table 12-70. ISP_REVISION

Address Offset	0x0000 0000		
Physical Address	0x480B C000	**Instance**	ISP
Description	ISP REVISION REGISTER This register contains the IP revision code in binary coded digital. For example, we have: 0x01 = revision 0.1 and 0x21 = revision 2.1		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																								REV							

Bits	Field Name	Description	Type	Reset
31:8	RESERVED	Write 0s for future compatibility. Reads return 0.	R	0x000000
7:0	REV	IP revision. [7:4] major revision [3:0] minor revision	R	TI internal data

Table 12-71. Register Call Summary for Register ISP_REVISION

Camera ISP Register Manual

- Camera ISP Register Summary: [0]

Table 12-72. ISP_SYSCONFIG

Address Offset	0x0000 0004		
Physical Address	0x480B C004	**Instance**	ISP
Description	ISP SYSTEM CONFIGURATION REGISTER		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																		MIDLE_MODE		RESERVED										SOFT_RESET	AUTO_IDLE

Bits	Field Name	Description	Type	Reset
31:14	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x00000
13:12	MIDLE_MODE	Master interface power management, MSTANDBY / WAIT protocol. 0x0: Force-standby: the MSTANDBY signal is only asserted to the power and reset clock manager when the module is disabled. 0x1: No-standby: the MSTANDBY signal is never asserted to the power and reset clock manager. 0x2: Smart-standby: the MSTANDBY signal is asserted to the power and reset clock manager based on the internal activity of the module. The ISP clocks are not disabled during smart standby.	RW	0x0
11:2	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x000
1	SOFT_RESET	Software reset. Set the bit to 1 to trigger the module reset. The bit is automatically reset be the HW. During reads return 0. 0x0: Normal mode. 0x1: The module is reset.	RW	0x0

Bits	Field Name	Description	Type	Reset
0	AUTO_IDLE	Internal interconnect and functional clock gating strategy 0x0: Interconnect and functional clock is free running. 0x1: Automatic interface clock gating strategy is applied based on the Interconnect interface activity for the interface clock and on the functional activity for functional clocks.	RW	0x1

Table 12-73. Register Call Summary for Register ISP_SYSCONFIG

Camera ISP Integration • Power Management: [0] [1] • Resets: [2]
Camera ISP Use Case and Tips • Programming Flow :
Camera ISP Register Manual • Camera ISP Register Summary: [5]

Table 12-74. ISP_SYSSTATUS

Address Offset	0x0000 0008		
Physical Address	0x480B C008	**Instance**	ISP
Description	ISP SYSTEM STATUS REGISTER		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																															RESET_DONE

Bits	Field Name	Description	Type	Reset
31:1	RESERVED	Write 0s for future compatibility. Reads return 0.	R	0x00000000
0	RESET_DONE	Internal reset monitoring 0x0: Internal module reset is ongoing. 0x1: Reset completed.	R	0x1

Table 12-75. Register Call Summary for Register ISP_SYSSTATUS

Camera ISP Use Case and Tips • Programming Flow :
Camera ISP Register Manual • Camera ISP Register Summary: [2]

Table 12-76. ISP_IRQ0ENABLE

Address Offset	0x0000 000C		
Physical Address	0x480B C00C	**Instance**	ISP
Description	INTERRUPT ENABLE REGISTER TO MCU. IRQ0 STATUS LINE. The same events are mapped in IRQ1. However, one event should be mapped to only one target.		
Type	RW		

Bits	Field Name	Description	Type	Reset
31	HS_VS_IRQ	HS or VS synchro event This event is triggered if a rising or falling edge is detected on the HS or VS signal. The rising or falling edge and the HS or VS signal selection is chosen with the ISP_CTRL.SYNC_DTECT bit field. 0x0: Event is masked 0x1: Event generates an interrupt when it occurs.	RW	0x0
30	RESERVED	RESERVED	RW	0x0
29	OCP_ERR_IRQ	ISP Interconnect error. This event is triggered when an Interconnect error occurs: SResp = ERR. 0x0: Event is masked 0x1: Event generates an interrupt when it occurs.	RW	0x0
28	MMU_ERR_IRQ	MMU error. 0x0: Event is masked 0x1: Event generates an interrupt when it occurs.	RW	0x0
27:26	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x0
25	OVF_IRQ	Central Resource SBL overflow This event is triggered when one of the buffer in the central resource SBL overflows. 0x0: Event is masked 0x1: Event generates an interrupt when it occurs.	RW	0x0
24	RSZ_DONE_IRQ	RESIZER module - resizer processing done event. This event is triggered at the end of the frame when the processing is completed for the current frame. 0x0: Event is masked 0x1: Event generates an interrupt when it occurs.	RW	0x0
23:22	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x0
21	CBUFF_IRQ	Circular buffer interrupt 0x0: Event is masked 0x1: Event generates an interrupt when it occurs.	RW	0x0
20	PRV_DONE_IRQ	PREVIEW module - processing done event. This event is triggered at the end of the frame when the processing is completed for the current frame. 0x0: Event is masked 0x1: Event generates an interrupt when it occurs.	RW	0x0
19	CCDC_LSC_PREFETCH_ERROR	The prefetch error indicates when the gain table was read to slowly from SDRAM. When this event is pending the module goes into transparent mode (output=input). Normal operation can be resumed at the start of the next frame after 1) clearing this event 2) disabling the LSC module 3) enabling it 0x0: Event is masked 0x1: Event generates an interrupt when it occurs.	RW	0x0

Bits	Field Name	Description	Type	Reset
18	CCDC_LSC_PREFETCH_COMPLETED	Indicates current state of the prefetch buffer. Can be used to start sending the data once the buffer is full to minimize the risk of an underflow. This event is triggered when the buffer contains 3 full paxel rows. It can be used to minimize buffer underflow risks. 0x0: Event is masked 0x1: Event generates an interrupt when it occurs.	RW	0x0
17	CCDC_LSC_DONE	The event is triggered when the internal state of LSC toggles from BUSY to IDLE. 0x0: Event is masked 0x1: Event generates an interrupt when it occurs.	RW	0x0
16	HIST_DONE_IRQ	HIST module - processing done event. This event is triggered at the end of the frame when the processing is completed for the current frame. 0x0: Event is masked 0x1: Event generates an interrupt when it occurs.	RW	0x0
15:14	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x0
13	H3A_AWB_DONE_IRQ	H3A module - auto exposure and auto white balance processing done event. This event is triggered at the end of the frame when the processing is completed for the current frame. 0x0: Event is masked 0x1: Event generates an interrupt when it occurs.	RW	0x0
12	H3A_AF_DONE_IRQ	H3A module - autofocus processing done event. This event is triggered at the end of the frame when the processing is completed for the current frame. 0x0: Event is masked 0x1: Event generates an interrupt when it occurs.	RW	0x0
11	CCDC_ERR_IRQ	CCDC module - faulty pixel correction memory underflow 0x0: Event is masked 0x1: Event generates an interrupt when it occurs.	RW	0x0
10	CCDC_VD2_IRQ	CCDC module - programmable event 2 0x0: Event is masked 0x1: Event generates an interrupt when it occurs.	RW	0x0
9	CCDC_VD1_IRQ	CCDC module - programmable event 1. 0x0: Event is masked 0x1: Event generates an interrupt when it occurs.	RW	0x0
8	CCDC_VD0_IRQ	CCDC module - programmable event 0. 0x0: Event is masked 0x1: Event generates an interrupt when it occurs.	RW	0x0
7	CSIB_LC3_IRQ	CSI1receiver module–event on logical channel 3 0x0: Event is masked. 0x1: Event generates an interrupt when it occurs.	RW	0x0
6	CSIB_LC2_IRQ	CSI1 receiver module–event on logical channel 2 0x0: Event is masked. 0x1: Event generates an interrupt when it occurs.	RW	0x0
5	CSIB_LC1_IRQ	CSI1 receiver module–event on logical channel 1 0x0: Event is masked. 0x1: Event generates an interrupt when it occurs.	RW	0x0
4	CSIB_LC0_IRQ	CSI1 receiver module–event on logical channel 0 0x0: Event is masked. 0x1: Event generates an interrupt when it occurs.	RW	0X0

Bits	Field Name	Description	Type	Reset
3	CSIB_LCM_IRQ	CSI1 receiver module–event on memory channel 0x0: Event is masked. 0x1: Event generates an interrupt when it occurs.		
2:1	RESERVED	Reserved bit field. Write the reset value.	RW	0x0
0	CSIA_IRQ	CSIA_RECEIVER module event. 0x0: Event is masked. 0x1: Event generates an interrupt when it occurs.	RW	0x0

Table 12-77. Register Call Summary for Register ISP_IRQ0ENABLE

Camera ISP Integration
- Interrupt Requests: [0] [1] [2] [3] [4] [5] [6] [7] [8] [9] [10] [11] [12] [13] [14] [15] [16] [17] [18]

Camera ISP Functional Description
- Interrupts: [19]

Camera ISP Basic Programming Model
- Events and Status Checking: [20]
- Events and Status Checking: [21] [22]
- Events and Status Checking: [23] [24]
- Inter-Frame Operations: [25]
- Event and Status Checking: [26] [27] [28]
- Event and Status Checking: [29] [30]
- Event and Status Checking: [31] [32] [33]
- Event and status Checking: [34] [35]

Camera ISP Register Manual
- Camera ISP Register Summary: [36]
- Camera ISP Register Description: [37] [38]

Table 12-78. ISP_IRQ0STATUS

Address Offset	0x0000 0010		
Physical Address	0x480B C010	**Instance**	ISP
Description	INTERRUPT STATUS REGISTER TO MCU. IRQ0 STATUS LINE.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
HS_VS_IRQ	RESERVED	OCP_ERR_IRQ	MMU_ERR_IRQ	RESERVED		OVF_IRQ	RSZ_DONE_IRQ	RESERVED		CBUFF_IRQ	PRV_DONE_IRQ	CCDC_LSC_PREFETCH_ERROR	CCDC_LSC_PREFETCH_COMPLETED	CCDC_LSC_DONE	HIST_DONE_IRQ	RESERVED		H3A_AWB_DONE_IRQ	H3A_AF_DONE_IRQ	CCDC_ERR_IRQ	CCDC_VD2_IRQ	CCDC_VD1_IRQ	CCDC_VD0_IRQ	CSIB_LC3_IRQ	CSIB_LC2Q	CSIB_LC1_IRQ	CSIB_LC0_IRQ	CSIB_LCM_IRQ	RESERVED		CSIA_IRQ

Bits	Field Name	Description	Type	Reset
31	HS_VS_IRQ	HS or VS sync event Reads: 0: Event is false. 1: Event is true. Writes: 0: Status bit unchanged 1: Status bit reset	R/W/1to Clr	0x0
30	RESERVED	Reserved	R/W	0x0
29	OCP_ERR_IRQ	ISP Interconnect error. Reads: 0: Event is false. 1: Event is true. Writes: 0: Status bit unchanged 1: Status bit reset	R/W/1to Clr	0x0
28	MMU_ERR_IRQ	MMU error. If event is true, one needs to read the MMU_IRQSTATUS register to know the event source. Write in MMU_IRQSTATUS to clear the bit. Reads: 0: Event is false. 1: Event is true.	R/W/1to Clr	0x0
27:26	RESERVED	Write 0s for future compatibility. Reads return 0.	R/W/1to Clr	0x0
25	OVF_IRQ	Central Resource SBL overflow If event is true, one needs to check the SBL_PCR register to know the source. One needs to clear the SBL_PCR register first before clearing this bit. Reads: 0: Event is false. 1: Event is true. Writes: 0: Status bit unchanged 1: Status bit reset	R/W/1to Clr	0x0
24	RSZ_DONE_IRQ	RESIZER module - resizer processing done event. Reads: 0: Event is false. 1: Event is true. Writes: 0: Status bit unchanged 1: Status bit reset 0x0: Event is masked 0x1: Event generates an interrupt when it occurs.	R/W/1to Clr	0x0
23:22	RESERVED	Write 0s for future compatibility. Reads return 0.	R/W/1to Clr	0x0
21	CBUFF_IRQ	A circular buffer event is pending. Check submodule's interrupt status register. Reads: 0: Event is false. 1: Event is true. Writes: 0: Status bit unchanged 1: Status bit reset	R/W/1to Clr	0x0
20	PRV_DONE_IRQ	PREVIEW module - processing done event. Reads: 0: Event is false. 1: Event is true. Writes: 0: Status bit unchanged 1: Status bit reset	R/W/1to Clr	0x0

Bits	Field Name	Description	Type	Reset
19	CCDC_LSC_PREFETCH_ ERROR	The prefetch error indicates when the gain table was read to slowly from SDRAM. Reads: 0: Event is false. 1: Event is true. Writes: 0: Status bit unchanged 1: Status bit reset	R/W/1to Clr	0x0
18	CCDC_LSC_PREFETCH_ COMPLETED	Indicates current state of the prefetch buffer. Reads: 0: Event is false. 1: Event is true. Writes: 0: Status bit unchanged 1: Status bit reset	R/W/1to Clr	0x0
17	CCDC_LSC_DONE	The event is triggered when the internal state of LSC toggles from BUSY to IDLE. Reads: 0: Event is false. 1: Event is true. Writes: 0: Status bit unchanged 1: Status bit reset	R/W/1to Clr	0x0
16	HIST_DONE_IRQ	HIST module - processing done event. Reads: 0: Event is false. 1: Event is true. Writes: 0: Status bit unchanged 1: Status bit reset	R/W/1to Clr	0x0
15:14	RESERVED	Write 0s for future compatibility. Reads return 0.	R/W/1to Clr	0x0
13	H3A_AWB_DONE_IRQ	H3A module - auto exposure and auto white balance processing done event. Reads: 0: Event is false. 1: Event is true. Writes: 0: Status bit unchanged 1: Status bit reset	R/W/1to Clr	0x0
12	H3A_AF_DONE_IRQ	H3A module - autofocus processing done event. Reads: 0: Event is false. 1: Event is true. Writes: 0: Status bit unchanged 1: Status bit reset	R/W/1to Clr	0x0
11	CCDC_ERR_IRQ	CCDC module - faulty pixel correction memory underflow If event is true, one needs to clear the CCDC_FPC.FPERR bit first before clearing this bit. Reads: 0: Event is false. 1: Event is true. Writes: 0: Status bit unchanged 1: Status bit reset	R/W/1to Clr	0x0
10	CCDC_VD2_IRQ	CCDC module - programmable event 2 Reads: 0: Event is false. 1: Event is true. Writes: 0: Status bit unchanged 1: Status bit reset	R/W/1to Clr	0x0

Bits	Field Name	Description	Type	Reset
9	CCDC_VD1_IRQ	CCDC module - programmable event 1. Reads: 0: Event is false. 1: Event is true. Writes: 0: Status bit unchanged 1: Status bit reset	R/W/1to Clr	0x0
8	CCDC_VD0_IRQ	CCDC module - programmable event 0. Reads: 0: Event is false. 1: Event is true. Writes: 0: Status bit unchanged 1: Status bit reset	R/W/1to Clr	0x0
7	CSIB_LC3_IRQ	CSI1 receiver module–event on logical channel 3 Reads: 0: Event is false. 1: Event is true. Writes: 0: Status bit unchanged 1: Status bit reset	R/W/1to Clr	0x0
6	CSIB_LC2_IRQ	CSI1receiver module–event on logical channel 2 Reads: 0: Event is false. 1: Event is true. Writes: 0: Status bit unchanged 1: Status bit reset	R/W/1to Clr	0x0
5	CSIB_LC1_IRQ	CSI1 receiver module– event on logical channel 1 Reads: 0: Event is false. 1: Event is true. Writes: 0: Status bit unchanged 1: Status bit reset	R/W/1to Clr	0x0
4	CSIB_LC0_IRQ	CSI1 receiver module–event on logical channel 0 Reads: 0: Event is false. 1: Event is true. Writes: 0: Status bit unchanged 1: Status bit reset	R/W/1to Clr	0x0
3	CSIB_LCM_IRQ	CSI1 receiver module–event on memory channel Reads: 0: Event is false. 1: Event is true. Writes: 0: Status bit unchanged 1: Status bit reset	R/W/1to Clr	0x0
2	RESERVED	Reserved bit field. Write the reset value.	R/W/1to Clr	0x0
1	RESERVED	Reserved bit field. Write the reset value.	R/W/1to Clr	0x0

Bits	Field Name	Description	Type	Reset
0	CSIA_IRQ	CSIA RECEIVER module event. If event is true, one needs to read the CSIA_IRQSTATUS register to know the event source. Write in CSIA_IRQSTATUS to clear the bit. Reads: 0: Event is false. 1: Event is true. Writes: 0: Status bit unchanged 1: Status bit reset	R/W/1to Clr	0x0

Table 12-79. Register Call Summary for Register ISP_IRQ0STATUS

Camera ISP Integration

- Interrupt Requests: [0] [1] [2] [3] [4] [5] [6] [7] [8] [9] [10] [11] [12] [13] [14] [15] [16] [17] [18]

Camera ISP Basic Programming Model

- Events and Status Checking: [19] [20] [21]
- Events and Status Checking: [22] [23] [24]
- Events and Status Checking: [25] [26] [27]
- Event and Status Checking: [28] [29] [30]
- Event and Status Checking: [31] [32] [33]
- Event and Status Checking: [34] [35] [36] [37] [38]
- Event and status Checking: [39] [40]

Camera ISP Register Manual

- Camera ISP Register Summary: [41]

Table 12-80. ISP_IRQ1ENABLE

Address Offset	0x0000 0014		
Physical Address	0x480B C014	**Instance**	ISP
Description	INTERRUPT ENABLE REGISTER TO DSP. IRQ1 STATUS LINE. The same events are mapped in IRQ0. However, one event should be mapped to only one target.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
HS_VS_IRQ	RESERVED	OCP_ERR_IRQ	MMU_ERR_IRQ	RESERVED		OVF_IRQ	RSZ_DONE_IRQ	RESERVED		CBUFF_IRQ	PRV_DONE_IRQ	CCDC_LSC_PREFETCH_ERROR	CCDC_LSC_PREFETCH_COMPLETED	CCDC_LSC_DONE	HIST_DONE_IRQ	RESERVED		H3A_AWB_DONE_IRQ	H3A_AF_DONE_IRQ	CCDC_ERR_IRQ	CCDC_VD2_IRQ	CCDC_VD1_IRQ	CCDC_VD0_IRQ	CSIB_LC3_IRQ	CSIB_LC2_IRQ	CSIB_LC1_IRQ	CSIB_LC0_IRQ	CSIB_LCM_IRQ	RESERVED		CSIA_IRQ

Bits	Field Name	Description	Type	Reset
31	HS_VS_IRQ	HS or VS synchro event This event is triggered if a rising or falling edge is detected on the HS or VS signal. The rising or falling edge and the HS or VS signal selection is chosen with the ISP_CTRL.SYNC_DTECT bit field. 0x0: Event is masked. 0x1: Event generates an interrupt when it occurs.	RW	0x0
30	RESERVED	RESERVED	RW	0x0

Bits	Field Name	Description	Type	Reset
29	OCP_ERR_IRQ	ISP Interconnect error. This event is triggered when an Interconnect error occurs: SResp = ERR. 0x0: Event is masked. 0x1: Event generates an interrupt when it occurs.	RW	0x0
28	MMU_ERR_IRQ	MMU error. 0x0: Event is masked. 0x1: Event generates an interrupt when it occurs.	RW	0x0
27:26	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x0
25	OVF_IRQ	Central Resource SBL overflow This event is triggered when one of the buffer in the central resource SBL overflows. 0x0: Event is masked. 0x1: Event generates an interrupt when it occurs.	RW	0x0
24	RSZ_DONE_IRQ	RESIZER module - resizer processing done event. This event is triggered at the end of the frame when the processing is completed for the current frame. 0x0: Event is masked. 0x1: Event generates an interrupt when it occurs.	RW	0x0
23:22	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x0
21	CBUFF_IRQ	Circular buffer interrupt 0x0: Event is masked. 0x1: Event generates an interrupt when it occurs.	RW	0x0
20	PRV_DONE_IRQ	PREVIEW module - processing done event. This event is triggered at the end of the frame when the processing is completed for the current frame. 0x0: Event is masked. 0x1: Event generates an interrupt when it occurs.	RW	0x0
19	CCDC_LSC_PREFETCH_ ERROR	The prefetch error indicates when the gain table was read to slowly from SDRAM. When this event is pending the module goes into transparent mode (output=input). Normal operation can be resumed at the start of the next frame after 1) clearing this event 2) disabling the LSC module 3) enabling it 0x0: Event is masked. 0x1: Event generates an interrupt when it occurs.	RW	0x0
18	CCDC_LSC_PREFETCH_ COMPLETED	Indicates current state of the prefetch buffer. Can be used to start sending the data once the buffer is full to minimize the risk of an underflow. This event is triggered when the buffer contains 3 full paxel rows. It can be used to minimize buffer underflow risks. 0x0: Event is masked. 0x1: Event generates an interrupt when it occurs.	RW	0x0
17	CCDC_LSC_DONE	The event is triggered when the internal state of LSC toggles from BUSY to IDLE. 0x0: Event is masked. 0x1: Event generates an interrupt when it occurs.	RW	0x0
16	HIST_DONE_IRQ	HIST module - processing done event. This event is triggered at the end of the frame when the processing is completed for the current frame. 0x0: Event is masked. 0x1: Event generates an interrupt when it occurs.	RW	0x0

Bits	Field Name	Description	Type	Reset
15:14	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x0
13	H3A_AWB_DONE_IRQ	H3A module - auto exposure and auto white balance processing done event. This event is triggered at the end of the frame when the processing is completed for the current frame. 0x0: Event is masked. 0x1: Event generates an interrupt when it occurs.	RW	0x0
12	H3A_AF_DONE_IRQ	H3A module - autofocus processing done event. This event is triggered at the end of the frame when the processing is completed for the current frame. 0x0: Event is masked. 0x1: Event generates an interrupt when it occurs.	RW	0x0
11	CCDC_ERR_IRQ	CCDC module - faulty pixel correction memory underflow 0x0: Event is masked 0x1: Event generates an interrupt when it occurs.	RW	0x0
10	CCDC_VD2_IRQ	CCDC module - programmable event 2 0x0: Event is masked. 0x1: Event generates an interrupt when it occurs.	RW	0x0
9	CCDC_VD1_IRQ	CCDC module - programmable event 1. 0x0: Event is masked. 0x1: Event generates an interrupt when it occurs.	RW	0x0
8	CCDC_VD0_IRQ	CCDC module - programmable event 0. 0x0: Event is masked. 0x1: Event generates an interrupt when it occurs.	RW	0x0
7	CSIB_LC3_IRQ	CSI1receiver module - event on logical channel 3 0x0: Event is masked. 0x1: Event generates an interrupt when it occurs.	RW	0x0
6	CSIB_LC2_IRQ	CSI1 receiver module - event on logical channel 2 0x0: Event is masked. 0x1: Event generates an interrupt when it occurs.	RW	0x0
5	CSIB_LC1_IRQ	CSI1 receiver module - event on logical channel 1 0x0: Event is masked. 0x1: Event generates an interrupt when it occurs.	RW	0x0
4	CSIB_LC0_IRQ	CSI1 receiver module–event on logical channel 0 0x0: Event is masked. 0x1: Event generates an interrupt when it occurs.	RW	0x0
3	CSIB_LCM_IRQ	CSI1 receiver module–event on memory channel 0x0: Event is masked. 0x1: Event generates an interrupt when it occurs.	RW	0x0
2	RESERVED	Reserved bit field. Write the reset value.	RW	0x0
1	RESERVED	Reserved bit field. Write the reset value.	RW	0x0
0	CSIA_IRQ	CSI2A RECEIVER module event. 0x0: Event is masked. 0x1: Event generates an interrupt when it occurs.	RW	0x0

Table 12-81. Register Call Summary for Register ISP_IRQ1ENABLE

Camera ISP Functional Description

- Interrupts: [0]

Table 12-81. Register Call Summary for Register ISP_IRQ1ENABLE (continued)

Camera ISP Basic Programming Model

- Events and Status Checking: [1]
- Events and Status Checking: [2]
- Events and Status Checking: [3]
- Inter-Frame Operations: [4]
- Event and Status Checking: [5] [6]
- Event and Status Checking: [7]
- Event and Status Checking: [8] [9]

Camera ISP Register Manual

- Camera ISP Register Summary: [10]

Table 12-82. ISP_IRQ1STATUS

Address Offset	0x0000 0018		
Physical Address	0x480B C018	**Instance**	ISP
Description	INTERRUPT STATUS REGISTER TO DSP. IRQ1 STATUS LINE.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
HS_VS_IRQ	RESERVED	OCP_ERR_IRQ	MMU_ERR_IRQ	RESERVED		OVF_IRQ	RSZ_DONE_IRQ	RESERVED		CBUFF_IRQ	PRV_DONE_IRQ	CCDC_LSC_PREFETCH_ERROR	CCDC_LSC_PREFETCH_COMPLETED	CCDC_LSC_DONE	HIST_DONE_IRQ	RESERVED		H3A_AWB_DONE_IRQ	H3A_AF_DONE_IRQ	CCDC_ERR_IRQ	CCDC_VD2_IRQ	CCDC_VD1_IRQ	CCDC_VD0_IRQ	CSIB_LC3_IRQ	CSIB_LC2_IRQ	CSIB_LC1_IRQ	CSIB_LC0_IRQ	CSIB_LCM_IRQ	RESERVED		CSIA_IRQ

Bits	Field Name	Description	Type	Reset
31	HS_VS_IRQ	HS or VS synchro event READS: 0: event is false 1: event is true WRITES 0: status bit unchanged 1: status bit reset	R/W/1to Clr	0x0
30	RESERVED	RESERVED	R/W	0x0
29	OCP_ERR_IRQ	ISP Interconnect error. READS: 0: event is false 1: event is true WRITES 0: status bit unchanged 1: status bit reset	R/W/1to Clr	0x0
28	MMU_ERR_IRQ	MMU error. If event is true, one needs to read the MMU_IRQSTATUS register to know the event source. Write in MMU_IRQSTATUS to clear the bit. READS: 0: event is false 1: event is true	R/W/1to Clr	0x0

Bits	Field Name	Description	Type	Reset
27:26	RESERVED	Write 0s for future compatibility. Reads return 0.	R/W/1to Clr	0x0
25	OVF_IRQ	Central Resource SBL overflow If event is true, one needs to check the SBL_PCR register to know the source. One needs to clear the SBL_PCR register first before clearing this bit. READS: 0: event is false 1: event is true WRITES 0: status bit unchanged 1: status bit reset	R/W/1to Clr	0x0
24	RSZ_DONE_IRQ	RESIZER module - resizer processing done event. READS: 0: event is false 1: event is true WRITES 0: status bit unchanged 1: status bit reset 0x0: Event is masked 0x1: Event generates an interrupt when it occurs.	R/W/1to Clr	0x0
23:22	RESERVED	Write 0s for future compatibility. Reads return 0.	R/W/1to Clr	0x0
21	CBUFF_IRQ	A circular buffer event is pending. Check submodule's interrupt status register. READS: 0: event is false 1: event is true WRITES 0: status bit unchanged 1: status bit reset	R/W/1to Clr	0x0
20	PRV_DONE_IRQ	PREVIEW module - processing done event. READS: 0: event is false 1: event is true WRITES 0: status bit unchanged 1: status bit reset	R/W/1to Clr	0x0
19	CCDC_LSC_PREFETCH_ ERROR	The prefetch error indicates when the gain table was read to slowly from SDRAM. READS: 0: event is false 1: event is true WRITES 0: status bit unchanged 1: status bit reset	R/W/1to Clr	0x0
18	CCDC_LSC_PREFETCH_ COMPLETED	Indicates current state of the prefetch buffer. READS: 0: event is false 1: event is true WRITES 0: status bit unchanged 1: status bit reset	R/W/1to Clr	0x0
17	CCDC_LSC_DONE	The event is triggered when the internal state of LSC toggles from BUSY to IDLE. READS: 0: event is false 1: event is true WRITES 0: status bit unchanged 1: status bit reset	R/W/1to Clr	0x0

Bits	Field Name	Description	Type	Reset
16	HIST_DONE_IRQ	HIST module - processing done event. READS: 0: event is false 1: event is true WRITES 0: status bit unchanged 1: status bit reset	R/W/1to Clr	0x0
15:14	RESERVED	Write 0s for future compatibility. Reads return 0.	R/W/1to Clr	0x0
13	H3A_AWB_DONE_IRQ	H3A module - auto exposure and auto white balance processing done event. READS: 0: event is false 1: event is true WRITES 0: status bit unchanged 1: status bit reset	R/W/1to Clr	0x0
12	H3A_AF_DONE_IRQ	H3A module - autofocus processing done event. READS: 0: event is false 1: event is true WRITES 0: status bit unchanged 1: status bit reset	R/W/1to Clr	0x0
11	CCDC_ERR_IRQ	CCDC module - faulty pixel correction memory underflow If event is true, one needs to clear the CCDC_FPC.FPERR bit first before clearing this bit. READS: 0: event is false 1: event is true WRITES 0: status bit unchanged 1: status bit reset	R/W/1to Clr	0x0
10	CCDC_VD2_IRQ	CCDC module - programmable event 2 READS: 0: event is false 1: event is true WRITES 0: status bit unchanged 1: status bit reset	R/W/1to Clr	0x0
9	CCDC_VD1_IRQ	CCDC module - programmable event 1. READS: 0: event is false 1: event is true WRITES 0: status bit unchanged 1: status bit reset	R/W/1to Clr	0x0
8	CCDC_VD0_IRQ	CCDC module - programmable event 0. READS: 0: event is false 1: event is true WRITES 0: status bit unchanged 1: status bit reset	R/W/1to Clr	0x0
7	CSIB_LC3_IRQ	CSI1receiver module–event on logical channel 3 Reads: 0: Event is false. 1: Event is true. Writes 0: Status bit unchanged 1: Status bit reset	R/W/1to Clr	0x0

Bits	Field Name	Description	Type	Reset
6	CSIB_LC2_IRQ	CSI1 receiver module–event on logical channel 2 Reads: 0: Event is false. 1: Event is true. Writes 0: Status bit unchanged 1: Status bit reset	R/W/1to Clr	0x0
5	CSIB_LC1_IRQ	CSI1 receiver module–event on logical channel 1 Reads: 0: Event is false. 1: Event is true. Writes 0: Status bit unchanged 1: Status bit reset	R/W/1to Clr	0x0
4	CSIB_LC0_IRQ	CSI1 receiver module–event on logical channel 0 Reads: 0: Event is false. 1: Event is true. Writes 0: Status bit unchanged 1: Status bit reset	R/W/1to Clr	0x0
3	CSIB_LCM_IRQ	CSI1 receiver module - event on memory channel. READS: 0: event is false 1: event is true WRITES 0: status bit unchanged 1: status bit reset	R/W/1to Clr	0x0
2	RESERVED	Reserved bit field. Write the reset value.	R/W/1to Clr	0x0
1	RESERVED	Reserved bit field. Write the reset value.	R/W/1to Clr	0x0
0	CSIA_IRQ	CSIA RECEIVER module event. If event is true, one needs to read the CSIA_IRQSTATUS register to know the event source. Write in CSIA_IRQSTATUS to clear the bit. READS: 0: event is false 1: event is true WRITES 0: status bit unchanged 1: status bit reset	R/W/1to Clr	0x0

Table 12-83. Register Call Summary for Register ISP_IRQ1STATUS

Camera ISP Basic Programming Model

- Events and Status Checking: [0] [1] [2]
- Events and Status Checking: [3] [4]
- Events and Status Checking: [5] [6]
- Event and Status Checking: [7] [8]
- Event and Status Checking: [9] [10]
- Event and Status Checking: [11] [12] [13] [14]
- Event and status Checking: [15] [16]

Camera ISP Register Manual

- Camera ISP Register Summary: [17]

Table 12-84. TCTRL_GRESET_LENGTH

Address Offset	0x0000 0030		
Physical Address	0x480B C030	**Instance**	ISP
Description	TIMING CONTROL - GLOBAL SHUTTER LENGTH REGISTER This register is used by the TIMING CTRL module to generate the cam_global_reset signal.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								LENGTH																							

Bits	Field Name	Description	Type	Reset
31:24	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x00
23:0	LENGTH	Sets the length of the cam_global_reset signal assertion in cycles of the CNTCLK clock. The CNTCLK frequency is generated with the TCTRL_CTRL.DIVC bit field. After signal assertion, the TCTRL_CTRL.GRESETEN bit is automatically cleared. The possible values are 0 to 2^24-1 cycles. The polarity of the cam_global_reset signal is set by the TCTRL_CTRL.GRESETPOL bit.	RW	0x000000

Table 12-85. Register Call Summary for Register TCTRL_GRESET_LENGTH

Camera ISP Basic Programming Model

- Camera-Control Signal Generator: [0]

Camera ISP Register Manual

- Camera ISP Register Summary: [1]
- Camera ISP Register Description: [2]

Table 12-86. TCTRL_PSTRB_REPLAY

Address Offset	0x0000 0034		
Physical Address	0x480B C034	**Instance**	ISP
Description	TIMING CONTROL - PRESTROBE REPLAY REGISTER This register is used by the TIMING CTRL module to generate the prestrobe signal.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
COUNTER								DELAY																							

Bits	Field Name	Description	Type	Reset
31:25	COUNTER	Sets the number of PRESTROBE pulses after the original pulse. If this bit is set to 0, the PRESTROBE signal behavior is only controlled by TCTRL_FRAME.STRB, TCTRL_PSTRB_DELAY and TCTRL_PSTRB_LENGTH. If TCTRL_PSTRB_LENGTH=0, there is no replay. This bit is useful when one wants to enable red-eye removal.	RW	0x00
24:0	DELAY	Sets the delay for the PRESTROBE signal re-assertion in cycles of the CNTCLK clock. The CNTCLK frequency is generated with the TCTRL_CTRL.DIVC bit field. The possible values are 0 to 2^25-1 cycles. If TCTRL_PSTRB_LENGTH=0, there is no replay. This bit field must not be set to 0 if the COUNTER is set to a value different of 0. This bit is useful when one wants to enable red-eye removal.	RW	0x0000000

Table 12-87. Register Call Summary for Register TCTRL_PSTRB_REPLAY

Camera ISP Basic Programming Model

- Camera-Control Signal Generator: [0] [1] [2]

Camera ISP Register Manual

- Camera ISP Register Summary: [3]

Table 12-88. ISP_CTRL

Address Offset	0x0000 0040		
Physical Address	0x480B C040	**Instance**	ISP
Description	CONTROL REGISTER After reset, the parallel interface is selected and only the CCDC module data write port is enabled.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FLUSH	JPEG_FLUSH	CCDC_WEN_POL	SBL_SHARED_RPORTB	SBL_SHARED_RPORTA	RESERVED	CBUFF1_BCF_CTRL		CBUFF0_BCF_CTRL		SBL_AUTOIDLE	SBL_WR0_RAM_EN	SBL_WR1_RAM_EN	SBL_RD_RAM_EN	PREV_RAM_EN	CCDC_RAM_EN	SYNC_DETECT		RSZ_CLK_EN	PRV_CLK_EN	HIST_CLK_EN	H3A_CLK_EN	CBUFF_AUTOGATING	CCDC_CLK_EN		SHIFT		RESERVED	PAR_CLK_POL		PAR_BRIDGE		PAR_SER_CLK_SEL	

Bits	Field Name	Description	Type	Reset
31	FLUSH	CCDC memory flush Writing 1 in this bit flushes the CCDC memories in the central resource SBL. The SBL memories are always flushed by the end of frame. However, there are cases where the end of frame cannot be detected.	RW	0x0
30	JPEG_FLUSH	JPEG flush When a camera module outputs a JPEG bit stream, this bit must be set because the bit stream length may not be a multiple of 32 bits. Enabling this bit ensures that no data stay in the design internal FIFOS.	RW	0x0
29	CCDC_WEN_POL	Sets the polarity of the CCDC WEN bit. 0x0: Active low 0x1: Active high	RW	0x0
28	SBL_SHARED_RPORTB	Controls SBL shared read port B access 0x0: Read port used by preview module dark frame read 0x1: Read port used by CCDC module lens-shading compensation data read	RW	0x0
27	SBL_SHARED_RPORTA	Controls SBL shared read port A access 0x0: Read port used by preview module data read 0x1: Read port used by CSI1 module data read	RW	0x0
26	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x0
25:24	CBUFF1_BCF_CTRL	Bandwidth control feedback loop configuration register 0x0: Disabled. 0x1: The BCF signal of CBUFF1 stalls the response phase of the CSIb OCP read master port. 0x2: The BCF signal of CBUFF1 stalls the request phase of the CSIb OCP read master port. 0x3: The BCF signal of CBUFF1 stalls the request and response phase of the CSIb OCP read master port.	RW	0x0

Bits	Field Name	Description	Type	Reset
23:22	CBUFF0_BCF_CTRL	Bandwidth control feedback loop configuration register 0x0: Disabled. 0x1: The BCF signal of CBUFF0 stalls the response phase of the CSIb OCP read master port. 0x2: The BCF signal of CBUFF0 stalls the request phase of the CSIb OCP read master port. 0x3: The BCF signal of CBUFF0 stalls the request and response phase of the CSIb OCP read master port.	RW	0x0
21	SBL_AUTOIDLE	Sets the SBL autoidle mode 0x0: Disabled 0x1: Enabled	RW	0x1
20	SBL_WR0_RAM_EN	This bit controls the SBL module WRITE0 RAM used by the RESIZER module. If the RESIZER module is disabled, this bit should be set to 0 to save power. 0x0: RAM is disabled 0x1: RAM is enabled	RW	0x0
19	SBL_WR1_RAM_EN	This bit controls the SBL module WRITE1 RAM. If the RESIZER module is the only module enabled to perform memory to memory resize operations, this bit should be set to 0 to save power. 0x0: RAM is disabled 0x1: RAM is enabled	RW	0x0
18	SBL_RD_RAM_EN	This bit controls the SBL module READ RAM. If no read requests are generated, this bit should be set to 0 to save power. 0x0: RAM is disabled 0x1: RAM is enabled	RW	0x0
17	PREV_RAM_EN	This bit controls the PREVIEW module RAM. If the PREVIEW module is not used, this bit should be set to 0 to save power. 0x0: RAM is disabled 0x1: RAM is enabled	RW	0x0
16	CCDC_RAM_EN	This bit controls the CCDC module RAM. If the CCDC module is not used, the bit should be set to 0 to save power. 0x0: RAM is disabled 0x1: RAM is enabled	RW	0x0
15:14	SYNC_DETECT	HS or VS synchronization signal detection It is sometimes necessary to detect the rising or falling edge of the horizontal and vertical synchro signals. When such event is detected, an interrupt is triggered if ISP_IRQ0ENABLE.HS_VS_IRQ = 1 or ISP_IRQ0ENABLE.HS_VS_IRQ = 1. 0x0: HS falling edge 0x1: HS rising edge 0x2: VS falling edge 0x3: VS rising edge	RW	0x0
13	RSZ_CLK_EN	RSZ module clock enable. This bit controls the clock distribution to the RSZ module. 0x0: Disable clock. The module is not active. However, accesses on the module slave port to configure it are still possible. 0x1: Enable clock. The module is fully functional.	RW	0x0

Bits	Field Name	Description	Type	Reset
12	PRV_CLK_EN	PRV module clock enable. This bit controls the clock distribution to the PRV module. 0x0: Disable clock. The module is not active. However, accesses on the module slave port to configure it are still possible. 0x1: Enable clock. The module is fully functional.	RW	0x0
11	HIST_CLK_EN	HIST module clock enable. This bit controls the clock distribution to the HIST module. 0x0: Disable clock. The module is not active. However, accesses on the module slave port to configure it are still possible. 0x1: Enable clock. The module is fully functional.	RW	0x0
10	H3A_CLK_EN	H3A module clock enable. This bit controls the clock distribution to the H3A module. 0x0: Disable clock. The module is not active. However, accesses on the module slave port to configure it are still possible. 0x1: Enable clock. The module is fully functional.	RW	0x0
9	CBUFF_AUTOGATING	CBUFF module autogating feature control 0x0: CBUFF autogating feature is disabled. The CBUFF internal clock is free running. 0x1: CBUFF autogating feature is enabled. The CBUFF internal clock is only enabled when it is requested by the CBUFF module.	RW	0x1
8	CCDC_CLK_EN	CCDC module clock enable. This bit controls the clock distribution to the CCDC module. 0x0: Disable clock. The module is not active. However, accesses on the module slave port to configure it are still possible. 0x1: Enable clock. The module is fully functional.	RW	0x0
7:6	SHIFT	Data lane shifter The parallel interface is a 12-bit interface, The video port of CSI1 is a 12-bit interface, The video port of CSI2 is a 14-bit interface. The Camera ISP has 14 data lanes but the full imaging pipeline only supports 10 bits. There are 2 main utilizations of the data lane shifter 1) Dynamic reduction: For example data from a 12 bit sensor can be converted into 10bit. 2) When a camera module as fewer than 12 data lanes, the ISP requires the pins to be connected on the least significant lanes. An issue occurs when a n-bit camera parallel interface can work in a m-bit mode with m<n. The ISP expects the m bits to be on the least significant data lanes whereas it is not correct. The data lane shifter takes place before the CCDC module 0x0: No shift. CAMEXT[13:0] -> CAM [13:0] 0x1: Shift by 2. CAMEXT[13:2] -> CAM [11:0] 0x2: Shift by 4 CAMEXT[13:4] -> CAM [9:0] 0x3: Shift by 6 CAMEXT[13:6] -> CAM [7:0]	RW	0x0
5	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x0
4	PAR_CLK_POL	This bit sets the pixel clock polarity on the parallel interface. The pixel clock is used for latching the pixel data into the CCDC module. 0x0: Clock not inverted. The data are sampled on the rising edge of the clock. 0x1: Clock inverted. The data are sampled on the falling edge of the clock.	RW	0x0

Bits	Field Name	Description	Type	Reset
3:2	PAR_BRIDGE	This bit field controls the 8 to 16-bit bridge at the input of the CCDC module. 0x0: The bridge is disabled: no conversion. 0x1: Reserved 0x2: The bridge is enabled. The first byte is written to cam_d[7:0], the second byte is written to cam_d[15:8] 0x3: The bridge is enabled. The first byte is written to cam_d[15:8], the second byte is written to cam_d[7:0]	RW	0x0
1:0	PAR_SER_CLK_SEL	Selects the serial or parallel interface as the input to the preview hardware. 0x0: Selects the 12-bit parallel interface as the input to the CCDC module. 0x1: Selects the CSIA serial interface as the input to the CCDC module. 0x2: Selects the CSIB serial interface as the input to the CCDC module.	RW	0x0

Table 12-89. Register Call Summary for Register ISP_CTRL

Camera ISP Environment
- Parallel Generic Configuration: JPEG Sensor Connection on the Parallel Interface: [0]

Camera ISP Integration
- Power Management: [1] [2] [3] [4] [5] [6]
- Interrupt Requests: [7]

Camera ISP Functional Description
- Bridge-Lane Shifter: [8] [9]
- CCDC: [10]
- Preview Engine Features: [11] [12]
- Functional Operations: [13] [14]
- Functional Description: [15] [16]

Camera ISP Basic Programming Model
- Memory Read Channel:
- CCDC Hardware Setup/Initialization: [18] [19] [20] [21] [22] [23] [24]
- CCDC Operations: [25] [26] [27]
- Preview Hardware Setup/Initialization: [29] [30]
- Summary of Constraints: [31] [32]

Camera ISP Register Manual
- Camera ISP Register Summary: [33]
- Camera ISP Register Description: [34] [35] [36]
- Camera ISP_CCDC Register Description: [37] [38]

Table 12-90. TCTRL_CTRL

Address Offset	0x0000 0050		
Physical Address	0x480B C050	**Instance**	ISP
Description	TIMING CONTROL - CONTROL REGISTER		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GRESETDIR	GRESETPOL	GRESETEN	INSEL		STRBPSTRBPOL	RESERVED	SHUTPOL	STRBEN	PSTRBEN	SHUTEN	RESERVED	DIVC										DIVB						DIVA			

Bits	Field Name	Description	Type	Reset
31	GRESETDIR	Sets the direction of the cam_global_reset signal. 0x0: INPUT. cam_global_reset is an input to the TIMING CONTROL module. cam_global_reset is externally generated. 0x1: OUTPUT. cam_global_reset is an output of the TIMING CONTROL module. cam_global_reset is internally generated. If GRESETEN is set to 1, the internally generated cam_global_reset triggers the generation of the PRESTROBE, STROBE and SHUTTER signals. The frame counters are ignored.	RW	0x0
30	GRESETPOL	Sets the polarity of the global reset signal: cam_global_reset. It applies whatever the direction of the cam_global_reset signal: input or output. 0x0: active high 0x1: active low	RW	0x0
29	GRESETEN	Triggers the generation of the cam_global_reset signal. The signal is asserted immediately. If enabled, the cam_global_reset signal is asserted for TCTRL_GRESET_LENGTH cycles. After the signal assertion, the enable bit is automatically cleared to 0. The polarity of the cam_global_reset signal is set with TCTRL_CTRL.GRESETPOL. Enabling this bit triggers the generation of the cam_shutter and cam_strobe signals (if previously enabled). The frame counters must be set to 0 when this bit is set to 1 and GRESETDIR is set a OUTPUT.	RW	0x0
28:27	INSEL	Sets the mode that triggers the SHUTTER, PRESTROBE and STROBE signals. 0x0: Video Port. The VS sync pulse at the input of the CCDC module is used to count the frames. The source of the VS pulse is selected by the ISP_CTRL [1:0] PAR_SER_CLK_SEL register. 0x1: CSIA interface. The frame start code (FSC) and frame end code (FEC) sync codes are used to count the frames. 0x2: CSIB interface. The frame start code (FSC) and frame end code (FEC) sync codes are used to count the frames. 0x3: GRESET. The cam_global_reset input signal triggers the SHUTTER, PRESTROBE and STROBE signals. In this mode, there are no frame counters. The delay counters start decrementing as soon as the cam_global_reset signal is asserted. The polarity of the cam_global_reset signal is set with TCTRL_CTRL.GRESETPOL.	RW	0x0
26	STRBPSTRBPOL	Sets the polarity of the strobe and prestrobe signals. 0x0: Active high 0x1: Active low	RW	0x0
25	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x0
24	SHUTPOL	Sets the polarity of the mechanical shutter signal: cam_shutter 0x0: Active high 0x1: Active low	RW	0x0
23	STRBEN	Flash strobe signal enable. If enabled, the STROBE signal is asserted after TCTRL_FRAME.STRB frames have been received and a delay of TCTRL_STRB_DELAY cycles have passed. The STROBE signal is asserted for TCTRL_STRB_LENGTH cycles. After the signal assertion, the enable bit is automatically cleared to 0. This signal must not be disabled by software.	RW	0x0

Bits	Field Name	Description	Type	Reset
22	PSTRBEN	Flash prestrobe signal enable. If enabled, the PRESTROBE signal is asserted after TCTRL_FRAME.PSTRB frames have been received and a delay of TCTRL_PSTRB_DELAY cycles have passed. The PRESTROBE signal is asserted for TCTRL_PSTRB_LENGTH cycles. After the signal assertion, the enable bit is automatically cleared to 0. This signal must not be disabled by software.	RW	0x0
21	SHUTEN	Mechanical shutter signal enable. If enabled, the SHUTTER signal is asserted after TCTRL_FRAME.SHUT frames have been received and a delay of TCTRL_SHUT_DELAY cycles have passed. The SHUTTER signal is asserted for TCTRL_SHUT_LENGTH cycles. After the signal assertion, the enable bit is automatically cleared to 0. This signal must not be disabled by software.	RW	0x0
20:19	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x0
18:10	DIVC	Sets the clock divisor value for the CNTCLK clock generation based on the CAM_MCLK input clock. CNTCLK is an internal clock used by the TIMING CTRL module counters. Usually, CNTCLK = CAM_MCLK/DIVC, except for some particular values shown hereafter. 0x0: No clock. CNTCLK is gated.	RW	0x000
9:5	DIVB	Sets the clock divisor value for the cam_xclkb clock generation based on the CAM_MCLK input clock. Usually, cam_xclkb = CAM_MCLK/DIVB, except for some particular values shown hereafter. This bit field is not reset by a soft reset; a hard reset is required. It enables to keep the clock configuration stable through a soft reset. 0x0: cam_xclkb = stable low level. Divider disabled. 0x1: cam_xclkb = stable high level. Divider disabled. 0x1F: cam_xclkb = cam_xclk. Bypass.	RW	0x00
4:0	DIVA	Sets the clock divisor value for the cam_xclka clock generation based on the CAM_MCLK input clock. Usually, cam_xclka = CAM_MCLK/DIVA, except for some particular values shown hereafter. This bit field is not reset by a soft reset; a hard reset is required. It enables to keep the clock configuration stable through a soft reset. 0x0: cam_xclka = stable low level. Divider disabled. 0x1: cam_xclka = stable high level. Divider disabled. 0x1F: cam_xclka = cam_xclk. Bypass.	RW	0x00

Table 12-91. Register Call Summary for Register TCTRL_CTRL

Camera ISP Integration
- Clocks: [0] [1]

Camera ISP Functional Description
- Timing Control: [2] [3] [4] [5] [6] [7] [8] [9] [10] [11] [12]

Camera ISP Basic Programming Model
- Timing Generator: [13] [14] [15] [16]
- Camera-Control Signal Generator: [17] [18] [19] [20] [21] [22] [23] [24] [25] [26] [27] [28] [29] [30] [31] [32] [33] [34] [35] [36] [37] [38] [39] [40] [41] [42] [43] [44] [45] [46] [47] [48] [49] [50] [51] [52] [53] [54] [55]

Camera ISP Register Manual
- Camera ISP Register Summary: [56]
- Camera ISP Register Description: [57] [58] [59] [60] [61] [62] [63] [64] [65] [66] [67] [68] [69] [70] [71]

Table 12-92. TCTRL_FRAME

Address Offset	0x0000 0054		
Physical Address	0x480B C054	**Instance**	ISP
Description	TIMING CONTROL - FRAME REGISTER This register is used by the TIMING CTRL module to generate the SHUTTER, PRESTROBE and STROBE signals.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								STRB								PSTRB								SHUT							

Bits	Field Name	Description	Type	Reset
31:18	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x0000
17:12	STRB	Frame counter for the STROBE signal generation. From 0 to 63 frames. This bit field is ignored if TCTRL.INSEL=GRESET.	RW	0x00
11:6	PSTRB	Frame counter for the PRESTROBE signal generation. From 0 to 63 frames. This bit field is ignored if TCTRL.INSEL=GRESET.	RW	0x00
5:0	SHUT	Frame counter for the SHUTTER signal generation. From 0 to 63 frames. This bit field is ignored if TCTRL.INSEL=GRESET.	RW	0x00

Table 12-93. Register Call Summary for Register TCTRL_FRAME

Camera ISP Basic Programming Model

- Camera-Control Signal Generator: [0] [1] [2] [3] [4] [5]

Camera ISP Register Manual

- Camera ISP Register Summary: [6]
- Camera ISP Register Description: [7] [8] [9] [10]

Table 12-94. TCTRL_PSTRB_DELAY

Address Offset	0x0000 0058		
Physical Address	0x480B C058	**Instance**	ISP
Description	TIMING CONTROL - PRE STROBE DELAY REGISTER This register is used by the TIMING CTRL module to generate the PRESTROBE signal.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								DELAY																							

Bits	Field Name	Description	Type	Reset
31:25	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x00
24:0	DELAY	Sets the delay for the PRESTROBE signal assertion in cycles of the CNTCLK clock. The CNTCLK frequency is generated with the TCTRL_CTRL.DIVC bit field. The possible values are 0 to 2^25-1 cycles.	RW	0x0000000

Table 12-95. Register Call Summary for Register TCTRL_PSTRB_DELAY

Camera ISP Basic Programming Model

- Camera-Control Signal Generator: [0] [1] [2]

Table 12-95. Register Call Summary for Register TCTRL_PSTRB_DELAY (continued)

Camera ISP Register Manual

- Camera ISP Register Summary: [3]
- Camera ISP Register Description: [4] [5]

Table 12-96. TCTRL_STRB_DELAY

Address Offset	0x0000 005C		
Physical Address	0x480B C05C	**Instance**	ISP
Description	TIMING CONTROL - STROBE DELAY REGISTER This register is used by the TIMING CTRL module to generate the STROBE signal.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								DELAY																							

Bits	Field Name	Description	Type	Reset
31:25	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x00
24:0	DELAY	Sets the delay for the cam_strobe signal assertion in cycles of the CNTCLK clock. The CNTCLK frequency is generated with the TCTRL_CTRL.DIVC bit field. The possible values are 0 to 2^25-1 cycles.	RW	0x0000000

Table 12-97. Register Call Summary for Register TCTRL_STRB_DELAY

Camera ISP Basic Programming Model

- Camera-Control Signal Generator: [0] [1] [2]

Camera ISP Register Manual

- Camera ISP Register Summary: [3]
- Camera ISP Register Description: [4]

Table 12-98. TCTRL_SHUT_DELAY

Address Offset	0x0000 0060		
Physical Address	0x480B C060	**Instance**	ISP
Description	TIMING CONTROL - SHUTTER DELAY REGISTER This register is used by the TIMING CTRL module to generate the SHUTTER signal.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								DELAY																							

Bits	Field Name	Description	Type	Reset
31:25	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x00
24:0	DELAY	Sets the delay for the cam_shutter signal assertion in cycles of the CNTCLK clock. The CNTCLK frequency is generated with the TCTRL_CTRL.DIVC bit field. The possible values are 0 to 2^25-1 cycles.	RW	0x0000000

Table 12-99. Register Call Summary for Register TCTRL_SHUT_DELAY

Camera ISP Basic Programming Model

- Camera-Control Signal Generator: [0] [1]

Camera ISP Register Manual

- Camera ISP Register Summary: [2]
- Camera ISP Register Description: [3]

Table 12-100. TCTRL_PSTRB_LENGTH

Address Offset	0x0000 0064		
Physical Address	0x480B C064	**Instance**	ISP
Description	TIMING CONTROL - PRESTROBE LENGTH REGISTER This register is used by the TIMING CTRL module to generate the PRESTROBE signal.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								LENGTH																							

Bits	Field Name	Description	Type	Reset
31:24	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x00
23:0	LENGTH	Sets the length of the PRESTROBE signal assertion in cycles of the CNTCLK clock. The CNTCLK frequency is generated with the TCTRL_CTRL.DIVC bit field. After signal assertion, the TCTRL_CTRL.PSTRBEN bit is automatically cleared. The possible values are 0 to 2^24-1 cycles.	RW	0x000000

Table 12-101. Register Call Summary for Register TCTRL_PSTRB_LENGTH

Camera ISP Basic Programming Model

- Camera-Control Signal Generator: [0] [1] [2]

Camera ISP Register Manual

- Camera ISP Register Summary: [3]
- Camera ISP Register Description: [4] [5] [6] [7]

Table 12-102. TCTRL_STRB_LENGTH

Address Offset	0x0000 0068		
Physical Address	0x480B C068	**Instance**	ISP
Description	TIMING CONTROL - STROBE LENGTH REGISTER This register is used by the TIMING CTRL module to generate the STROBE signal.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								LENGTH																							

Bits	Field Name	Description	Type	Reset
31:24	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x00
23:0	LENGTH	Sets the length of the cam_strobe signal assertion in cycles of the CNTCLK clock. The CNTCLK frequency is generated with the TCTRL_CTRL.DIVC bit field. After signal assertion, the TCTRL_CTRL.STRBEN bit is automatically cleared. The possible values are 0 to 2^24-1 cycles.	RW	0x000000

Table 12-103. Register Call Summary for Register TCTRL_STRB_LENGTH

Camera ISP Basic Programming Model • Camera-Control Signal Generator: [0] [1] [2]
Camera ISP Register Manual • Camera ISP Register Summary: [3] • Camera ISP Register Description: [4]

Table 12-104. TCTRL_SHUT_LENGTH

Address Offset	0x0000 006C		
Physical Address	0x480B C06C	**Instance**	ISP
Description	TIMING CONTROL - SHUTTER LENGTH REGISTER This register is used by the TIMING CTRL module to generate the SHUTTER signal.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								LENGTH																							

Bits	Field Name	Description	Type	Reset
31:24	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x00
23:0	LENGTH	Sets the length of the cam_shutter signal assertion in cycles of the CNTCLK clock. The CNTCLK frequency is generated with the TCTRL_CTRL.DIVC bit field. After signal assertion, the TCTRL_CTRL.SHUTEN bit is automatically cleared. The possible values are 0 to 2^24-1 cycles.	RW	0x000000

Table 12-105. Register Call Summary for Register TCTRL_SHUT_LENGTH

Camera ISP Basic Programming Model • Camera-Control Signal Generator: [0] [1]
Camera ISP Register Manual • Camera ISP Register Summary: [2] • Camera ISP Register Description: [3]

12.6.3 Camera ISP_CBUFF Registers

12.6.3.1 Camera ISP_CBUFF Register Summary

Table 12-106. ISP_CBUFF Register Summary

Register Name	Type	Register Width (Bits)	Address Offset	Physical Address
CBUFF_REVISION	R	32	0x0000 0000	0x480B C100
CBUFF_SYSCONFIG	RW	32	0x0000 0010	0x480B C110
CBUFF_SYSSTATUS	R	32	0x0000 0014	0x480B C114
CBUFF_IRQSTATUS	RW	32	0x0000 0018	0x480B C118
CBUFF_IRQENABLE	RW	32	0x0000 001C	0x480B C11C
CBUFFx_CTRL(1)	RW	32	0x0000 0020 + (0x4 * x)	0x480B C120 + (0x4 * x)
CBUFFx_STATUS(1)	R	32	0x0000 0030 + (0x4 * x)	0x480B C130 + (0x4 * x)
CBUFFx_START(1)	RW	32	0x0000 0040 + (0x4 * x)	0x480B C140 + (0x4 * x)
CBUFFx_END(1)	RW	32	0x0000 0050 + (0x4 * x)	0x480B C150 + (0x4 * x)
CBUFFx_WINDOWSIZE(1)	RW	32	0x0000 0060 + (0x4 * x)	0x480B C160 + (0x4 * x)

(1) x= 0 to 1

Table 12-106. ISP_CBUFF Register Summary (continued)

Register Name	Type	Register Width (Bits)	Address Offset	Physical Address
CBUFFx_THRESHOLD[1]	RW	32	0x0000 0070 + (0x4 * x)	0x480B C170 + (0x4 * x)

12.6.3.2 Camera ISP_CBUFF Register Description

Table 12-107. CBUFF_REVISION

Address Offset	0x0000 0000		
Physical Address	0x480B C100	**Instance**	ISP_CBUFF
Description	This register contains the IP revision code		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																								REV							

Bits	Field Name	Description	Type	Reset
31:8	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x000000
7:0	REV	IP revision [7:4] Major revision [3:0] Minor revision	R	TI internal data

Table 12-108. Register Call Summary for Register CBUFF_REVISION

Camera ISP Register Manual

- Camera ISP_CBUFF Register Summary: [0]

Table 12-109. CBUFF_SYSCONFIG

Address Offset	0x0000 0010		
Physical Address	0x480B C110	**Instance**	ISP_CBUFF
Description	This register allows controlling various parameters of the Interconnect interface.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																															

Bits	Field Name	Description	Type	Reset
31:0	RESERVED	Write 0s for future compatibility. Reads return zero.	RW	0x00000000

Table 12-110. Register Call Summary for Register CBUFF_SYSCONFIG

Camera ISP Register Manual

- Camera ISP_CBUFF Register Summary: [0]

Table 12-111. CBUFF_SYSSTATUS

Address Offset	0x0000 0014		
Physical Address	0x480B C114	**Instance**	ISP_CBUFF
Description	The register provides status information about the module, excluding the interrupt status information		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																															

Bits	Field Name	Description	Type	Reset
31:0	RESERVED	Reserved for module-specific status information. Reads return 0	R	0x00000000

Table 12-112. Register Call Summary for Register CBUFF_SYSSTATUS

Camera ISP Register Manual

- Camera ISP_CBUFF Register Summary: [0]

Table 12-113. CBUFF_IRQSTATUS

Address Offset	0x0000 0018		
Physical Address	0x480B C118	**Instance**	ISP_CBUFF
Description	The interrupt status register regroups all the status of the module internal events that can generate an interrupt.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																										IRQ_CBUFF1_OVR	IRQ_CBUFF1_INVALID	IRQ_CBUFF1_READY	IRQ_CBUFF0_OVR	IRQ_CBUFF0_INVALID	IRQ_CBUFF0_READY

Bits	Field Name	Description	Type	Reset
31:6	RESERVED	Write 0s for future compatibility. Reads return zero.	RW	0x0000000
5	IRQ_CBUFF1_OVR	Buffer overflow event. 0x0: No done interrupt pending (r); Status unchanged (w). 0x1: Done interrupt pending (r); Status bit cleared (w).	R/W/1to Clr	0x0
4	IRQ_CBUFF1_INVALID	Invalid access 0x0: No done interrupt pending (r); Status unchanged (w). 0x1: Done interrupt pending (r); Status bit cleared (w).	R/W/1to Clr	0x0
3	IRQ_CBUFF1_READY	The CPUW1 physical buffer is ready to be accessed by the CPU. 0x0: No done interrupt pending (r); Status unchanged (w). 0x1: Done interrupt pending (r); Status bit cleared (w).	R/W/1to Clr	0x0
2	IRQ_CBUFF0_OVR	Buffer overflow event 0x0: No done interrupt pending (r); Status unchanged (w). 0x1: Done interrupt pending (r); Status bit cleared (w).	R/W/1to Clr	0x0

Bits	Field Name	Description	Type	Reset
1	IRQ_CBUFF0_INVALID	Invalid access 0x0: No YUV buffer done interrupt pending (r); Status unchanged (w). 0x1: YUV buffer done interrupt pending (r); Status bit cleared (w).	R/W/1to Clr	0x0
0	IRQ_CBUFF0_READY	The CPUW0 physical buffer is ready to be accessed by the CPU. 0x0: No done interrupt pending (r); Status unchanged (w). 0x1: Done interrupt pending (r); Status bit cleared (w).	R/W/1to Clr	0x0

Table 12-114. Register Call Summary for Register CBUFF_IRQSTATUS

Camera ISP Integration • Interrupt Requests: [0] [1] [2] [3] [4] [5] [6]
Camera ISP Functional Description • Functional Description: [7] [8] [9]
Camera ISP Basic Programming Model • Event and status Checking: [10] [11] [12]
Camera ISP Register Manual • Camera ISP_CBUFF Register Summary: [13]

Table 12-115. CBUFF_IRQENABLE

Address Offset	0x0000 001C		
Physical Address	0x480B C11C	**Instance**	ISP_CBUFF
Description	The interrupt enable register allows to enable/disable the module internal sources of interrupt, on an event-by-event basis.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																										IRQ_CBUFF1_OVR	IRQ_CBUFF1_INVALID	IRQ_CBUFF1_READY	IRQ_CBUFF0_OVR	IRQ_CBUFF0_INVALID	IRQ_CBUFF0_READY

Bits	Field Name	Description	Type	Reset
31:6	RESERVED	Write 0s for future compatibility. Reads return zero.	RW	0x0000000
5	IRQ_CBUFF1_OVR	Buffer overflow event 0x0: Interrupt is masked. 0x1: Interrupt is enabled.	RW	0x0
4	IRQ_CBUFF1_INVALID	Invalid access 0x0: Interrupt is masked. 0x1: Interrupt is enabled.	RW	0x0
3	IRQ_CBUFF1_READY	The CPUW1 physical buffer is ready to be accessed by the CPU. 0x0: Interrupt is masked. 0x1: Interrupt is enabled.	RW	0x0

Bits	Field Name	Description	Type	Reset
2	IRQ_CBUFF0_OVR	Buffer overflow event 0x0: Interrupt is masked. 0x1: Interrupt is enabled.	RW	0x0
1	IRQ_CBUFF0_INVALID	Invalid access 0x0: Interrupt is masked. 0x1: Interrupt is enabled.	RW	0x0
0	IRQ_CBUFF0_READY	The CPUW0 physical buffer is ready to be accessed by the CPU. 0x0: interrupt is masked 0x1: Interrupt is enabled	RW	0x0

Table 12-116. Register Call Summary for Register CBUFF_IRQENABLE

Camera ISP Integration
- Interrupt Requests: [0] [1] [2] [3] [4] [5]

Camera ISP Register Manual
- Camera ISP_CBUFF Register Summary: [6]

Table 12-117. CBUFFx_CTRL

Address Offset	0x0000 0020 + (0x4 * x)	**Index**	x = 0 to 1
Physical Address	0x480B C120 + (0x4 * x)	**Instance**	ISP_CBUFF
Description	Circular buffer x control register		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																						WCOUNT		BCF				ALLOW_NW_EQ_CPUW	DONE	RWMODE	ENABLE

Bits	Field Name	Description	Type	Reset
31:10	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x000000
9:8	WCOUNT	Window count 0x0: 2 windows 0x1: 4 windows 0x2: 8 windows 0x3: 16 windows	RW	0x0
7:4	BCF	This register controls the bandwidth control feedback loop output. Functionality depends on subsystem integration. 0: Control loop disabled. Data read from memory is free running. 1-15: The control feedback loop signal is asserted when the window count available for ISP is below (<) the threshold. In other words at least (>=) BCF windows are available for ISP access when this signal is released.	RW	0x0

Bits	Field Name	Description	Type	Reset
3	ALLOW_NW_EQ_CPUW	Allow NW=CPUW. Better buffer utilization when ISP does not use the next write window. 0x0: When the CPUW and the NW pointers designate the same window and accesses are effectively performed to those windows an overflow event occurs. This happens when - the CPU has received an READY IRQ for that window indicating that it can be accessed and - the ISP performs an access to that window. ISP accesses are tracked based on OCPI activity. 0x1: When the CPUW and the CW pointers designate the same window and accesses are effectively performed to those windows an overflow event occurs. This happens when - the CPU has received an READY IRQ for that window indicating that it can be accessed and - the ISP performs an access to that window. ISP accesses are tracked based on OCPI activity.	RW	0x0
2	DONE	Write this bit to 1 to indicate the CPU has finished processing its physical buffer. This bit is automatically cleared by hardware, reads always return 0. 0x0: No effect. 0x1: The CPU has completely processed the CPUW physical buffer.	W	0x0
1	RWMODE	Selects read or write mode 0x0: Write mode. HW writes and CPU reads the physical space. CPU accesses are out of CBUFF module's scope, therefore only writes are permitted between CBUFF0_START and CBUFF0_END. 0x1: Read mode. HW reads and CPU writes the physical space. CPU accesses are out of CBUFF module's scope; therefore only reads are permitted between CBUFF0_START and CBUFF0_END.	RW	0x0
0	ENABLE	Enable/disable 0x0: Disables the circular buffer 0; this resets the internal state of circular buffer 0. All accesses received on OCPI are transmitted to OCPO without modification. Disabling the module takes effect immediately. It is SW responsibility to ensure that no more accesses to CBUFF0 are outstanding before disabling the module. Otherwise memory corruption may occur. 0x1: Enable the circular buffer 0. All accesses between CBUFF0_START and CBUFF0_END are processed by the module.	RW	0x0

Table 12-118. Register Call Summary for Register CBUFFx_CTRL

Camera ISP Integration • Interrupt Requests: [0]
Camera ISP Functional Description • Functional Description: [1] [2] [3] [4] [5] [6] [7] [8] [9] [10] [11] [12] [13]
Camera ISP Basic Programming Model • Register Setup: [18] [19] [20] [21] • Register Accessibility During Frame Processing: [22] [23] • Operations: [24] [25] [26]
Camera ISP Register Manual • Camera ISP_CBUFF Register Summary: [27]

Table 12-119. CBUFFx_STATUS

Address Offset	0x0000 0030 + (0x4 * x)	**Index**	x = 0 to 1
Physical Address	0x480B C130 + (0x4 * x)	**Instance**	ISP_CBUFF
Description	Threshold value used to check if the CW or NW windows are full.		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								RESERVED				NW				RESERVED				CW				RESERVED				CPUW			

Bits	Field Name	Description	Type	Reset
31:24	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x00
23:20	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x0
19:16	NW	Next window number. Valid values depend on the CBUFF_CTRL.WCOUNT register.	R	0x1
15:12	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x0
11:8	CW	Current window number. Valid values depend on the CBUFF_CTRL.WCOUNT register.	R	0x0
7:4	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x0
3:0	CPUW	Current CPU window number. Valid values depend on the CBUFF_CTRL.WCOUNT register.	R	0x0

Table 12-120. Register Call Summary for Register CBUFFx_STATUS

Camera ISP Functional Description
- Functional Description: [0] [1] [2]

Camera ISP Basic Programming Model
- Event and status Checking: [3]

Camera ISP Register Manual
- Camera ISP_CBUFF Register Summary: [4]

Table 12-121. CBUFFx_START

Address Offset	0x0000 0040 + (0x4 * x)	**Index**	x = 0 to 1
Physical Address	0x480B C140 + (0x4 * x)	**Instance**	ISP_CBUFF
Description	Start address of the virtual space managed by circular buffer x. Start address of the 1st physical buffer managed by circular buffer x.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDR																													RESERVED		

Bits	Field Name	Description	Type	Reset
31:3	ADDR	Address, in 64 bit words.	RW	0x00000000
2:0	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x0

Table 12-122. Register Call Summary for Register CBUFFx_START

Camera ISP Functional Description • Functional Description: [0] [1] [2] [3] [4] [5]
Camera ISP Basic Programming Model • Register Setup: [6]
Camera ISP Register Manual • Camera ISP_CBUFF Register Summary: [7]

Table 12-123. CBUFFx_END

Address Offset	0x0000 0050 + (0x4 * x)	**Index**	x = 0 to 1
Physical Address	0x480B C150 + (0x4 * x)	**Instance**	ISP_CBUFF
Description	End address of the virtual space managed by circular buffer x.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDR																														RESERVED	

Bits	Field Name	Description	Type	Reset
31:3	ADDR	Address, in 64 bit words.	RW	0x00000000
2:0	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x0

Table 12-124. Register Call Summary for Register CBUFFx_END

Camera ISP Functional Description • Functional Description: [0] [1] [2] [3] [4] [5] [6] [7] [8]
Camera ISP Basic Programming Model • Register Setup: [9]
Camera ISP Register Manual • Camera ISP_CBUFF Register Summary: [10]

Table 12-125. CBUFFx_WINDOWSIZE

Address Offset	0x0000 0060 + (0x4 * x)	**Index**	x = 0 to 1
Physical Address	0x480B C160 + (0x4 * x)	**Instance**	ISP_CBUFF
Description	Defines the window size.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								SIZE																						RESERVED	

Bits	Field Name	Description	Type	Reset
31:24	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x00
23:3	SIZE	Size, in 64 bit words.	RW	0x000000
2:0	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x0

Table 12-126. Register Call Summary for Register CBUFFx_WINDOWSIZE

Camera ISP Functional Description • Functional Description: [0] [1] [2] [3] [4] [5] [6] [7]
Camera ISP Basic Programming Model • Register Setup: [8] [9]
Camera ISP Register Manual • Camera ISP_CBUFF Register Summary: [10]

Table 12-127. CBUFFx_THRESHOLD

Address Offset	0x0000 0070 + (0x4 * x)	**Index**	x = 0 to 1
Physical Address	0x480B C170 + (0x4 * x)	**Instance**	ISP_CBUFF
Description	Threshold value used to check if a write window is full.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								THRESHOLD																							

Bits	Field Name	Description	Type	Reset
31:24	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x00
23:0	THRESHOLD	Threshold value, in bytes.	RW	0x000000

Table 12-128. Register Call Summary for Register CBUFFx_THRESHOLD

Camera ISP Functional Description • Functional Description: [0] [1] [2] [3] [4]
Camera ISP Basic Programming Model • Register Setup: [5] [6]
Camera ISP Register Manual • Camera ISP_CBUFF Register Summary: [7]

12.6.4 Camera ISP_CSI1B Registers

12.6.4.1 Camera ISP_CSI1B Register Summary

Table 12-129. ISP_CSI1B Register Summary

Register Name	Type	Register Width (Bits)	Address Offset	Physical Address
CSI1B_REVISION	R	32	0x0000 0000	0x480B C400
CSI1B_SYSCONFIG	RW	32	0x0000 0004	0x480B C404
CSI1B_SYSSTATUS	R	32	0x0000 0008	0x480B C408
CSI1B_IRQENABLE	RW	32	0x0000 000C	0x480B C40C
CSI1B_IRQSTATUS	RW 1toClr	32	0x0000 0010	0x480B C410
CSI1B_LCM_IRQENABLE	RW	32	0x0000 002C	0x480B C42C
CSI1B_LCM_IRQSTATUS	RW 1toClr	32	0x0000 0030	0x480B C430
CSI1B_CTRL	RW	32	0x0000 0040	0x480B C440
CSI1B_DBG	W	32	0x0000 0044	0x480B C444
CSI1B_GNQ	R	32	0x0000 0048	0x480B C448
CSI1B_CTRL1	RW	32	0x0000 0050	0x480B C450
CSI1B_CODE	RW	32	0x0000 0054	0x480B C454

Table 12-129. ISP_CSI1B Register Summary (continued)

Register Name	Type	Register Width (Bits)	Address Offset	Physical Address
CSI1B_STAT_START	RW	32	0x0000 0058	0x480B C458
CSI1B_STAT_SIZE	RW	32	0x0000 005C	0x480B C45C
CSI1B_SOF_ADDR	RW	32	0x0000 0060	0x480B C460
CSI1B_EOF_ADDR	RW	32	0x0000 0064	0x480B C464
CSI1B_DAT_START	RW	32	0x0000 0068	0x480B C468
CSI1B_DAT_SIZE	RW	32	0x0000 006C	0x480B C46C
CSI1B_DAT_PING_ADDR	RW	32	0x0000 0070	0x480B C470
CSI1B_DAT_PONG_ADDR	RW	32	0x0000 0074	0x480B C474
CSI1B_DAT_OFST	RW	32	0x0000 0078	0x480B C478
CSI1B_LCM_CTRL	RW	32	0x0000 01D0	0x480B C5D0
CSI1B_LCM_VSIZE	RW	32	0x0000 01D4	0x480B C5D4
CSI1B_LCM_HSIZE	RW	32	0x0000 01D8	0x480B C5D8
CSI1B_LCM_PREFETCH	RW	32	0x0000 01DC	0x480B C5DC
CSI1B_LCM_SRC_ADDR	RW	32	0x0000 01E0	0x480B C5E0
CSI1B_LCM_SRC_OFST	RW	32	0x0000 01E4	0x480B C5E4
CSI1B_LCM_DST_ADDR	RW	32	0x0000 01E8	0x480B C5E8
CSI1B_LCM_DST_OFST	RW	32	0x0000 01EC	0x480B C5EC

12.6.4.2 Camera ISP_CSI1B Register Description

Table 12-130. CSI1B_REVISION

Address Offset	0x0000 0000		
Physical Address	0x480B C400	**Instance**	ISP_CSI1B
Description	MODULE REVISION This register contains the IP revision code in binary coded digital. For example, we have: 0x01 = revision 0.1 and 0x21 = revision 2.1		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																								REV							

Bits	Field Name	Description	Type	Reset
31:8	RESERVED	Write 0s for future compatibility. Reads return 0.	R	0x000000
7:0	REV	IP revision [7:4] Major revision [3:0] Minor revision	R	TI internal data

Table 12-131. Register Call Summary for Register CSI1B_REVISION

Camera ISP Register Manual

- Camera ISP_CSI1B Register Summary: [0]

Table 12-132. CSI1B_SYSCONFIG

Address Offset	0x0000 0004		
Physical Address	0x480B C404	**Instance**	ISP_CSI1B
Description	SYSTEM CONFIGURATION REGISTER This register is the CSI1 system configuration register.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																		MSTANDBY_MODE		RESERVED										SOFT_RESET	AUTO_IDLE

Bits	Field Name	Description	Type	Reset
31:14	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x00000
13:12	MSTANDBY_MODE	Sets the behavior of the master port power management signals. 0x0: Force-standby. Hardware signal is only asserted when the module is disabled. 0x1: No-standby. Hardware signal is never asserted. 0x2: Smart-standby: Hardware signal is asserted based on the activity of the module. The module tries to go to standby during the vertical blanking period.	RW	0x0
11:2	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x000
1	SOFT_RESET	Software reset. Set the bit to 1 to trigger a module reset. The bit is automatically reset by the HW. During reads return 0. 0x0: Normal mode. 0x1: The module is reset	RW	0x0
0	AUTO_IDLE	Internal interface clock gating strategy. 0x0: Interface clock is free-running. 0x1: Automatic Interface clock gating strategy is applied based on the Interconnect interface activity.	RW	0x1

Table 12-133. Register Call Summary for Register CSI1B_SYSCONFIG

Camera ISP Register Manual

- Camera ISP_CSI1B Register Summary: [0]

Table 12-134. CSI1B_SYSSTATUS

Address Offset	0x0000 0008		
Physical Address	0x480B C408	**Instance**	ISP_CSI1B
Description	SYSTEM STATUS REGISTER This register provides status information about the module, excluding the interrupt status register.		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																															RESET_DONE

Bits	Field Name	Description	Type	Reset
31:1	RESERVED	Write 0s for future compatibility. Reads return 0.	R	0x00000000
0	RESET_DONE	Internal reset monitoring 0x0: Internal module reset is on going. 0x1: Reset completed.	R	0x1

Table 12-135. Register Call Summary for Register CSI1B_SYSSTATUS

Camera ISP Register Manual

- Camera ISP_CSI1B Register Summary: [0]

Table 12-136. CSI1B_IRQENABLE

Address Offset	0x0000 000C		
Physical Address	0x480B C40C	**Instance**	ISP_CSI1B
Description	INTERRUPT ENABLE REGISTER		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																				FS_IRQ	LE_IRQ	LS_IRQ	FE_IRQ	COUNT_IRQ	RESERVED	FIFO_OVF_IRQ	RESERVED	FSP_IRQ	FW_IRQ	FSC_IRQ	SSC_IRQ

Bits	Field Name	Description	Type	Reset
31:12	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x00000
11	FS_IRQ	Frame start sync code detection. 0x0: Event is masked 0x1: Event generates an interrupt when it occurs	RW	0x0
10	LE_IRQ	Line end sync code detection. 0x0: Event is masked 0x1: Event generates an interrupt when it occurs	RW	0x0
9	LS_IRQ	Line start sync code detection. 0x0: Event is masked 0x1: Event generates an interrupt when it occurs	RW	0x0
8	FE_IRQ	Frame end sync code detection. 0x0: Event is masked 0x1: Event generates an interrupt when it occurs	RW	0x0
7	COUNT_IRQ	Frame counter reached. 0x0: Event is masked 0x1: Event generates an interrupt when it occurs	RW	0x0
6	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x0
5	FIFO_OVF_IRQ	FIFO overflow error. 0x0: Event is masked 0x1: Event generates an interrupt when it occurs	RW	0x0
4	RESERVED	Write 0s for future compatibility. Reads return 0. 0x0: Event is masked 0x1: Event generates an interrupt when it occurs	RW	0x0

Bits	Field Name	Description	Type	Reset
3	FSP_IRQ	FSP error. 0x0: Event is masked 0x1: Event generates an interrupt when it occurs	RW	0x0
2	FW_IRQ	Frame width error. 0x0: Event is masked 0x1: Event generates an interrupt when it occurs	RW	0x0
1	FSC_IRQ	False sync code error. 0x0: Event is masked 0x1: Event generates an interrupt when it occurs	RW	0x0
0	SSC_IRQ	Shifted sync code error. This interrupt cannot be triggered if the complex I/O is set in parallel mode (CSI1_CTRL.IO_OUT_SEL = 1). 0x0: Event is masked 0x1: Event generates an interrupt when it occurs	RW	0x0

Table 12-137. Register Call Summary for Register CSI1B_IRQENABLE

Camera ISP Register Manual

- Camera ISP_CSI1B Register Summary: [0]

Table 12-138. CSI1B_IRQSTATUS

Address Offset	0x0000 0010		
Physical Address	0x480B C410	**Instance**	ISP_CSI1B
Description	INTERRUPT STATUS REGISTER		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																				FS_IRQ	LE_IRQ	LS_IRQ	FE_IRQ	COUNT_IRQ	RESERVED	FIFO_OVF_IRQ	RESERVED	FSP_IRQ	FW_IRQ	FSC_IRQ	SSC_IRQ

Bits	Field Name	Description	Type	Reset
31:12	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x00000
11	FS_IRQ	Frame start sync code detection status. 0x0: READS: Event is false. WRITES: Status bit unchanged. 0x1: READS: Event is true (pending). WRITES: Status bit is reset.	RW1toClr	0x0
10	LE_IRQ	Line end sync code detection status. 0x0: READS: Event is false. WRITES: Status bit unchanged. 0x1: READS: Event is true (pending). WRITES: Status bit is reset.	RW1toClr	0x0
9	LS_IRQ	Line start sync code detection status. 0x0: READS: Event is false. WRITES: Status bit unchanged. 0x1: READS: Event is true (pending). WRITES: Status bit is reset.	RW1toClr	0x0

Bits	Field Name	Description	Type	Reset
8	FE_IRQ	Frame end sync code detection status. 0x0: READS: Event is false. WRITES: Status bit unchanged. 0x1: READS: Event is true (pending). WRITES: Status bit is reset.	RW1toClr	0x0
7	COUNT_IRQ	Frame counter reached status 0x0: READS: Event is false. WRITES: Status bit unchanged. 0x1: READS: Event is true (pending). WRITES: Status bit is reset.	RW1toClr	0x0
6	RESERVED	Write 0s for future compatibility. Reads return 0.	RW1toClr	0x0
5	FIFO_OVF_IRQ	FIFO overflow error status. 0x0: READS: Event is false. WRITES: Status bit unchanged. 0x1: READS: Event is true (pending). WRITES: Status bit is reset.	RW1toClr	0x0
4	RESERVED	Write 0s for future compatibility. Reads return 0. 0x0: READS: Event is false. WRITES: Status bit unchanged. 0x1: READS: Event is true (pending). WRITES: Status bit is reset.	RW1toClr	0x0
3	FSP_IRQ	FSP error status. 0x0: READS: Event is false. WRITES: Status bit unchanged. 0x1: READS: Event is true (pending). WRITES: Status bit is reset.	RW1toClr	0x0
2	FW_IRQ	Frame width error status. 0x0: READS: Event is false. WRITES: Status bit unchanged. 0x1: READS: Event is true (pending). WRITES: Status bit is reset.	RW1toClr	0x0
1	FSC_IRQ	False sync code error status. 0x0: READS: Event is false. WRITES: Status bit unchanged. 0x1: READS: Event is true (pending). WRITES: Status bit is reset.	RW1toClr	0x0
0	SSC_IRQ	Shifted sync code error status. 0x0: READS: Event is false. WRITES: Status bit unchanged. 0x1: READS: Event is true (pending). WRITES: Status bit is reset.	RW1toClr	0x0

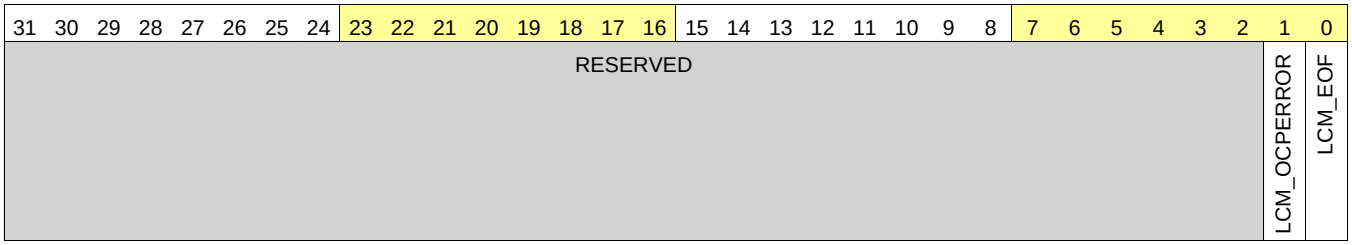
Table 12-139. Register Call Summary for Register CSI1B_IRQSTATUS

Camera ISP Register Manual

- Camera ISP_CSI1B Register Summary: [0]

Table 12-140. CSI1B_LCM_IRQENABLE

Address Offset	0x0000 002C		
Physical Address	0x480B C42C	**Instance**	ISP_CSI1B
Description	INTERRUPT ENABLE REGISTER - Memory channel This register regroups all the events related to the memory channel 2. The events related to memory channel trigger SINTERRUPTN[8]. The channel must be enabled for events to be generated on that channel.		
Type	RW		



Bits	Field Name	Description	Type	Reset
31:2	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x00000000
1	LCM_OCPERROR	An OCP error occurred on the master read port. 0x0: Event is masked 0x1: Event generates an interrupt when it occurs	RW	0x0
0	LCM_EOF	Memory read channel - End of frame 0x0: Event generates an interrupt when it occurs 0x1: Event is masked	RW	0x0

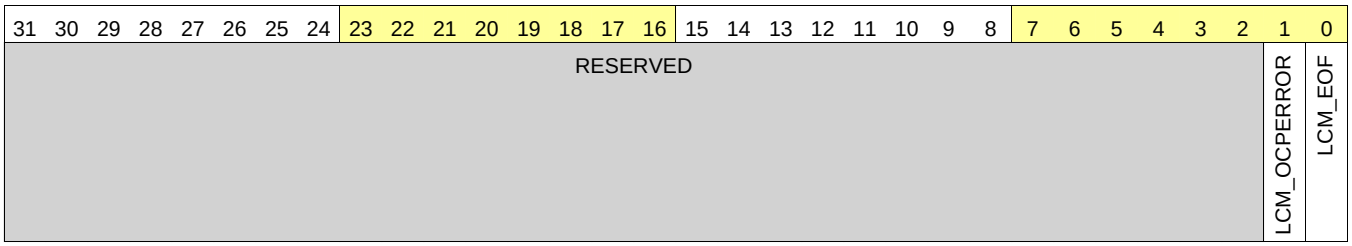
Table 12-141. Register Call Summary for Register CSI1B_LCM_IRQENABLE

Camera ISP Register Manual

- Camera ISP_CSI1B Register Summary: [0]

Table 12-142. CSI1B_LCM_IRQSTATUS

Address Offset	0x0000 0030		
Physical Address	0x480B C430	**Instance**	ISP_CSI1B
Description	INTERRUPT STATUS REGISTER - Memory channel This register regroups all the events related to memory channel. The events related to memory channel trigger SINTERRUPTN[8]. The channel must be enabled for events to be generated on that channel.		
Type	RW 1toClr		



Bits	Field Name	Description	Type	Reset
31:2	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x00000000
1	LCM_OCPERROR	An OCP error occurred on the master read port. 0x0: READS: Event is true (pending). WRITES: Status bit is reset. 0x1: READS: Event is false. WRITES: Status bit unchanged.	RW 1toClr	0x0

Bits	Field Name	Description	Type	Reset
0	LCM_EOF	Memory read channel - End of frame 0x0: READS: Event is false. WRITES: Status bit unchanged. 0x1: READS: Event is true (pending). WRITES: Status bit is reset.	RW 1toClr	0x0

Table 12-143. Register Call Summary for Register CSI1B_LCM_IRQSTATUS

Camera ISP Register Manual

- Camera ISP_CSI1B Register Summary: [0]

Table 12-144. CSI1B_CTRL

Address Offset	0x0000 0040		
Physical Address	0x480B C440	**Instance**	ISP_CSI1B
Description	GLOBAL CONTROL REGISTER This register controls the CSI1 RECEIVER module. This register must not be modified dynamically (except IF_EN bit field).		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																			VP_CLK_POL	VP_ONLY_EN	INV	VP_OUT_CTRL		DBG_EN	BURST		RESERVED	FRAME	IO_OUT_SEL	PHY_SEL	IF_EN

Bits	Field Name	Description	Type	Reset
31:13	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x00000
12	VP_CLK_POL	VP clock polarity 0x0: The RECEIVER module writes the data on the VP on the pixel clock falling edge. The module connected to the VP must sample the data on the pixel clock rising edge. 0x1: The RECEIVER module writes the data on the VP on the pixel clock raising edge. The module connected to the VP must sample the data on the pixel clock falling edge.	RW	0x0
11	VP_ONLY_EN	VP only enable. 0x0: The VP is enabled and the Interconnect master port are enabled. 0x1: The VP is enabled and the Interconnect master port is disabled. The embedded data and pixel data are output on the VP.	RW	0x0
10	INV	Strobe/clock inversion control signal 0x0: Not inverted. 0x1: Inverted.	RW	0x0
9:8	VP_OUT_CTRL	Video port output clock control. Sets the video port output clock as a function of the interface clock (OCPCLK). 0x0: No division: video port clock = OCPCLK. 0x1: Division by 2: video port clock = OCPCLK/2. 0x2: Division by 3: video port clock = OCPCLK/3. 0x3: Division by 4: video port clock = OCPCLK/4.	RW	0x1

Bits	Field Name	Description	Type	Reset
7	DBG_EN	Enables the debug mode. 0x0: Disable 0x1: Enable	RW	0x0
6:5	BURST	Forces the write burst size used by the module. The write burst size must never exceed the output FIFO size. The output FIFO size can be read with the CSI1_GNQ.FIFODEPTH bit field. 0x0: 1x 64-bit burst = single request. 0x1: 2x 64-bit bursts. 0x2: 4x 64-bit bursts. 0x3: 8x 64-bit bursts.	RW	0x0
4	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x0
3	FRAME	Set the modality in which IF_EN works. 0x0: When SW writes IF_EN = 0 the interface is disabled immediately. 0x1: When SW writes IF_EN = 1 the interface is disabled after the next FEC sync code.	RW	0x0
2	IO_OUT_SEL	I/O cell output mode selection. 0x0: I/O output is serial: DATA, STROBE 0x1: I/O output is parallel: DATA32, DATATOG, SYNTOG. Note that the SSC_IRQ error cannot be triggered in this configuration.	RW	0x0
1	PHY_SEL	Physical layer protocol selection. 0x0: Data/Clock physical layer 0x1: Data/Strobe physical layer	RW	0x0
0	IF_EN	Enables the physical interface to the module. 0x0: The interface is disabled. If FRAME = 0, it is disabled immediately. If FRAME = 1, it is disabled on the next FEC sync code. 0x1: The interface is enabled immediately, the data acquisition starts on the next FSC sync code. Writing 1 to this register when the current value is 0 has the effect to clear the output FIFO. The pixel data of the following frame is written in the PING buffer, that is, the CSI1_CTRL.PING_PONG bits are reset to 1 as well.	RW	0x0

Table 12-145. Register Call Summary for Register CSI1B_CTRL

Camera ISP Register Manual

- Camera ISP_CSI1B Register Summary: [0]

Table 12-146. CSI1B_DBG

Address Offset	0x0000 0044		
Physical Address	0x480B C444	**Instance**	ISP_CSI1B
Description	DEBUG REGISTER This register provides a way to debug the CSI1 RECEIVER module with no image sensor connected to the module. The debug mode is enabled by CSI1_CTRL.DBG_EN. Only full 32-bit values must be written.		
Type	W		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DBG																															

Bits	Field Name	Description	Type	Reset
31:0	DBG	32-bit input value.	W	0x00000000

Table 12-147. Register Call Summary for Register CSI1B_DBG

Camera ISP Register Manual

- Camera ISP_CSI1B Register Summary: [0]

Table 12-148. CSI1B_GNQ

Address Offset	0x0000 0048		
Physical Address	0x480B C448	**Instance**	ISP_CSI1B
Description	GENERIC PARAMETER REGISTER This register provide a way to read the generic parameters used in the design.		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																								RESERVED		OCPREADPORT	FIFODEPTH			NBCHANNELS	

Bits	Field Name	Description	Type	Reset
31:5	RESERVED	Reads return 0.	R	0x0000000
4:2	FIFODEPTH	Output FIFO size in multiple of 64 bits. 0x0: 2x 64 bits 0x1: 4x 64 bits 0x2: 8x 64 bits 0x3: 16x 64 bits 0x4: 32x 64 bits 0x5: 64x 64 bits	R	0x1
1:0	RESERVED	Read returns 2.	R	0x2

Table 12-149. Register Call Summary for Register CSI1B_GNQ

Camera ISP Register Manual

- Camera ISP_CSI1B Register Summary: [0]

Table 12-150. CSI1B_CTRL1

Address Offset	0x0000 0050		
Physical Address	0x480B C450	**Instance**	ISP_CSI1B
Description	CONTROL REGISTER This register is shadowed: modifications are taken into account after the next FSC sync code.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
COUNT								RESERVED					RESERVED	PING_PONG	COUNT_UNLOCK	ALPHA								FORMAT						RESERVED	REGION_EN	RESERVED

Bits	Field Name	Description	Type	Reset
31:24	COUNT	Sets the number of frame to acquire. Once the frame acquisition starts, the COUNT value is decremented after every frame. When COUNT reaches 0, the COUNT_IRQ interrupt is triggered and CHAN_EN is set to 0. Writes to this bit field are controlled by the COUNT_UNLOCK bit. COUNT can be overwritten dynamically with a new count value." 0: Infinite number of frames (no count). 1: 1 frame to acquire ... 255: 255 frames to acquire.	RW	0x00
23:19	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x00
18	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x0
17	PING_PONG	Indicates whether the PING or PONG destination address (CSI1_DAT_PING_ADDR or CSI1_DAT_PONG_ADDR) was used to write the last frame. This bit field toggles after every FEC sync code. 0x0: PING buffer 0x1: PONG buffer	R	0x1
16	COUNT_UNLOCK	Unlock writes to the COUNT bit field. 0x0: COUNT bit field is locked. Writes have no effect 0x1: COUNT bit field is unlocked. Writes are possible.	W	0x0
15:8	ALPHA	Alpha value for RGB888 and RBG444.	RW	0x00
7:3	FORMAT	Data format selection. 0x0: YUV422 BIG ENDIAN 0x1: YUV422 LITTLE ENDIAN 0x2: YUV420 0x3: YUV422 + VP or RAW8 + VP 0x4: RGB444 + EXP16 0x5: RGB565 0x6: RGB888 0x7: RGB888 + EXP32 0x10: RAW8 0x14: RAW10 0x15: RAW10 + EXP16 0x16: RAW10 + VP 0x18: RAW12 0x19: RAW12 + EXP16 0x1A: RAW12 + VP 0x1C: JPEG8 + FSP 0x1D: JPEG8 0x1E: RAW10 -> RAW8 RAW10 data from sensor is right shifted to produce RAW8 before it is sent to memory.	RW	0x00
2	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x0
1	REGION_EN	Enables the setting of regions of interest in the frame: SOF region, EOF region and DAT region. 0x0: Disabled 0x1: Enabled	RW	0x0
0	RESERVED	Write 1 for future compatibility. Reads returns 1.	RW	0x1

Table 12-151. Register Call Summary for Register CSI1B_CTRL1

Camera ISP Register Manual

- Camera ISP_CSI1B Register Summary: [0]

Table 12-152. CSI1B_CODE

Address Offset	0x0000 0054		
Physical Address	0x480B C454	**Instance**	ISP_CSI1B
Description	CODE REGISTER This register sets the codes that are used in the 32-bit synchronization codes to recognize frame start, frame end, line start and line end codes. The default values are usually not supposed to be modified. Updating this register with new codes under a flowing serial transmission causes unexpected results.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED												RESERVED				FEC				FSC				LEC				LSC			

Bits	Field Name	Description	Type	Reset
31:20	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x000
19:16	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x0
15:12	FEC	End sync code identifier. The sync code identifier is located between bits 0 to 3 in the 32-bit synchronization codes.	RW	0x3
11:8	FSC	Start sync code identifier. The sync code identifier is located between bits 0 to 3 in the 32-bit synchronization codes.	RW	0x2
7:4	LEC	End sync code identifier. The sync code identifier is located between bits 0 to 3 in the 32-bit synchronization codes.	RW	0x1
3:0	LSC	Start sync code identifier. The sync code identifier is located between bits 0 to 3 in the 32-bit synchronization codes.	RW	0x0

Table 12-153. Register Call Summary for Register CSI1B_CODE

Camera ISP Register Manual

- Camera ISP_CSI1B Register Summary: [0]

Table 12-154. CSI1B_STAT_START

Address Offset	0x0000 0058		
Physical Address	0x480B C458	**Instance**	ISP_CSI1B
Description	STATUS LINE START REGISTER This register is shadowed: modifications are taken into account after the next FSC sync code.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								EOF								RESERVED				SOF											

Bits	Field Name	Description	Type	Reset
31:28	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x0
27:16	EOF	Sets the vertical position of the EOF status lines in regards of the FSC sync code. From 0 to 4095.	RW	0x000
15:12	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x0

Bits	Field Name	Description	Type	Reset
11:0	SOF	Sets the vertical position of the EOF status lines in regards of the FSC sync code. Should always be 0.	RW	0x000

Table 12-155. Register Call Summary for Register CSI1B_STAT_START

Camera ISP Register Manual

- Camera ISP_CSI1B Register Summary: [0]

Table 12-156. CSI1B_STAT_SIZE

Address Offset	0x0000 005C		
Physical Address	0x480B C45C	**Instance**	ISP_CSI1B
Description	STATUS LINE SIZE REGISTER This register is shadowed: modifications are taken into account after the next FSC sync code.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								EOF								RESERVED								SOF							

Bits	Field Name	Description	Type	Reset
31:28	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x0
27:16	EOF	Sets the number of EOF status lines From 0 to 4095	RW	0x000
15:12	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x0
11:0	SOF	Sets the number of SOF status line(s) From 0 to 4095	RW	0x000

Table 12-157. Register Call Summary for Register CSI1B_STAT_SIZE

Camera ISP Register Manual

- Camera ISP_CSI1B Register Summary: [0]

Table 12-158. CSI1B_SOF_ADDR

Address Offset	0x0000 0060		
Physical Address	0x480B C460	**Instance**	ISP_CSI1B
Description	SOF STATUS LINE MEM ADDRESS REGISTER This register sets the 32-bit memory address where the SOF data are stored. Note that the 5 LSBs are ignored: the address should be aligned on a 32-byte boundary. This register is shadowed: modifications are taken into account after the next FSC sync code.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDR																											RESERVED			

Bits	Field Name	Description	Type	Reset
31:5	ADDR	27 most significant bits of the 32-bit address.	RW	0x0000000
4:0	RESERVED	5 least significant bits of the 32-bit address. Write 0s for future compatibility. Reads return 0.	RW	0x00

Table 12-159. Register Call Summary for Register CSI1B_SOF_ADDR

Camera ISP Register Manual

- Camera ISP_CSI1B Register Summary: [0]

Table 12-160. CSI1B_EOF_ADDR

Address Offset	0x0000 0064		
Physical Address	0x480B C464	**Instance**	ISP_CSI1B
Description	EOF STATUS LINE MEM ADDRESS REGISTER This register sets the 32-bit memory address where the EOF data are stored. Note that the 5 LSBs are ignored: the address should be aligned on a 32-byte boundary. This register is shadowed: modifications are taken into account after the next FSC sync code.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDR																											RESERVED			

Bits	Field Name	Description	Type	Reset
31:5	ADDR	27 most significant bits of the 32-bit address.	RW	0x0000000
4:0	RESERVED	5 least significant bits of the 32-bit address. Write 0s for future compatibility. Reads return 0.	RW	0x00

Table 12-161. Register Call Summary for Register CSI1B_EOF_ADDR

Camera ISP Register Manual

- Camera ISP_CSI1B Register Summary: [0]

Table 12-162. CSI1B_DAT_START

Address Offset	0x0000 0068		
Physical Address	0x480B C468	**Instance**	ISP_CSI1B
Description	DATA START REGISTER This register is shadowed: modifications are taken into account after the next FSC sync code.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED				VERT												RESERVED															

Bits	Field Name	Description	Type	Reset
31:28	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x0
27:16	VERT	Sets the vertical position of the data in regards of the FSC sync code. From 0 to 4095 lines.	RW	0x000
15:0	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x0000

Table 12-163. Register Call Summary for Register CSI1B_DAT_START

Camera ISP Register Manual

- Camera ISP_CSI1B Register Summary: [0]

Table 12-164. CSI1B_DAT_SIZE

Address Offset	0x0000 006C		
Physical Address	0x480B C46C	**Instance**	ISP_CSI1B
Description	DATA SIZE REGISTER This register is shadowed: modifications are taken into account after the next FSC sync code.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								VERT								RESERVED															

Bits	Field Name	Description	Type	Reset
31:28	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x0
27:16	VERT	Sets the vertical size of the data window. From 0 to 4095 lines. If VERT = "0", no data is output.	RW	0x000
15:0	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x0000

Table 12-165. Register Call Summary for Register CSI1B_DAT_SIZE

Camera ISP Register Manual

- Camera ISP_CSI1B Register Summary: [0]

Table 12-166. CSI1B_DAT_PING_ADDR

Address Offset	0x0000 0070		
Physical Address	0x480B C470	**Instance**	ISP_CSI1B
Description	DATA MEM PING ADDRESS REGISTER This register sets the 32-bit memory address where the pixel data are stored. The destination is double buffered: this register sets the PING address. Double buffering is enabled when the addresses CSI1_DAT_PING_ADDR and CSI1_DAT_PONG_ADDR are different. Note that the 5 LSBs are ignored: the address should be aligned on a 32-byte boundary. This register is shadowed: modifications are taken into account after the next FSC sync code.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDR																											RESERVED			

Bits	Field Name	Description	Type	Reset
31:5	ADDR	27 most significant bits of the 32-bit address.	RW	0x0000000
4:0	RESERVED	5 least significant bits of the 32-bit address. Write 0s for future compatibility. Reads return 0.	RW	0x00

Table 12-167. Register Call Summary for Register CSI1B_DAT_PING_ADDR

Camera ISP Register Manual

- Camera ISP_CSI1B Register Summary: [0]

Table 12-168. CSI1B_DAT_PONG_ADDR

Address Offset	0x0000 0074		
Physical Address	0x480B C474	**Instance**	ISP_CSI1B
Description	DATA MEM PONG ADDRESS REGISTER - LOG CHAN 0 This register sets the 32-bit memory address where the pixel data are stored. The destination is double buffered: this register sets the PONG address. Double buffering is enabled when the addresses CSI1_DAT_PING_ADDR andCSI1_DAT_PONG_ADDR are different. Note that the 5 LSBs are ignored: the address should be aligned on a 32-byte boundary. This register is shadowed: modifications are taken into account after the next FSC sync code.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDR																								RESERVED							

Bits	Field Name	Description	Type	Reset
31:5	ADDR	27 most significant bits of the 32-bit address.	RW	0x0000000
4:0	RESERVED	5 least significant bits of the 32-bit address. Write 0s for future compatibility. Reads return 0.	RW	0x00

Table 12-169. Register Call Summary for Register CSI1B_DAT_PONG_ADDR

Camera ISP Register Manual

- Camera ISP_CSI1B Register Summary: [0]

Table 12-170. CSI1B_DAT_OFST

Address Offset	0x0000 0078		
Physical Address	0x480B C478	**Instance**	ISP_CSI1B
Description	DATA MEM ADDRESS OFFSET REGISTER This register sets the offset, which is applied on the destination address after each line is written to memory. This register applies for both CSI1_DAT_PING_ADDR and CSI1_DAT_PONG_ADDR. For example, it enables to perform 2D data transfers of the pixel data into a frame buffer. In such case, the pixel data and frame buffer data must have the same data format. Note that the 5 LSBs are ignored: the offset must be a multiple of 32 bytes. The usage of this register is limited to the following data formats: YUV422 LITTLE ENDIAN, YUV422 BIG ENDIAN, RGB565, RGB444 + EXP16, RGB888 + EXP32. This register is shadowed: modifications are taken into account after the next FSC sync code.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
OFST																								RESERVED							

Bits	Field Name	Description	Type	Reset
31:5	OFST	Line offset programmed in bytes. If OFST = 0, the data is written contiguously in memory. Otherwise, OFST sets the destination offset between the first pixel of the previous line and the first pixel of the current line.	RW	0x0000000
4:0	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x00

Table 12-171. Register Call Summary for Register CSI1B_DAT_OFST

Camera ISP Register Manual

- Camera ISP_CSI1B Register Summary: [0]

Table 12-172. CSI1B_LCM_CTRL

Address Offset	0x0000 01D0		
Physical Address	0x480B C5D0	**Instance**	ISP_CSI1B
Description	Control register for the memory channel. It defines the data format of the source frame stored in memory and how this frame is processed.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DST_PACK	RESERVED	RESERVED	RESERVED		DST_FORMAT			SRC_PACK	RESERVED	RESERVED	RESERVED		SRC_FORMAT			RESERVED								BURST_SIZE			READ_THROTTLE		DST_PORT	RESERVED	CHAN_EN

Bits	Field Name	Description	Type	Reset
31	DST_PACK	Data is packed before it is sent to memory. 0x0: Enabled 0x1: Disabled	RW	0x0
30	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x0
29	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x0
28:27	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x0
26:24	DST_FORMAT	Output format selection. Not every combination between input and output formats are possible. Check the specification for allowed modes. 0x0: Reserved 0x1: Reserved 0x2: RAW8 0x3: RAW10 0x4: RAW12	RW	0x0
23	SRC_PACK	Data stored in memory is packed and must be unpacked. 0x0: Enabled 0x1: Disabled	RW	0x0
22	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x0
21	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x0
20:19	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x0
18:16	SRC_FORMAT	Data format of the data stored in memory. As there is no header embedded in the data sent to memory, users are responsible of choosing the proper format. 0x0: Reserved 0x1: Reserved 0x2: RAW8 0x3: RAW10 0x4: RAW12	RW	0x0
15:8	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x00

Bits	Field Name	Description	Type	Reset
7:5	BURST_SIZE	Defines the burst size of the master read port 0x0: 1x 64-bit burst = single request. 0x1: 2x 64-bit bursts. 0x2: 4x 64-bit bursts. 0x3: 8x 64-bit bursts. 0x4: 16x 64-bit bursts. 0x5: 32x 64-bit bursts.	RW	0x0
4:3	READ_THROTTLE	Limit maximum data read speed for memory to memory operation 0x0: Full speed. Throughput is limited by internal processing capabilities. 0x1: speed 0x3: 1/4 speed 0x4: 1/8 speed	RW	0x0
2	DST_PORT	Select the destination port 0x0: Data is sent to memory. 0x1: Data is sent to video port, it is always sent without compression or packing.	RW	0x0
1	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x0
0	CHAN_EN	Enables the read from memory channel. Before enabling the memory read channel SW must: - disable the physical interface using the IF_EN bit - wait until disabling of the physical interface is effective (depends on the FRAME bit) Read from memory starts as soon as this bit is set, therefore all CSI1_LCM_x registers must be configured correctly before. This bit is cleared by HW at the end of the frame. 0x0: Disabled 0x1: Enabled	RW	0x0

Table 12-173. Register Call Summary for Register CSI1B_LCM_CTRL

Camera ISP Register Manual

- Camera ISP_CSI1B Register Summary: [0]

Table 12-174. CSI1B_LCM_VSIZE

Address Offset	0x0000 01D4		
Physical Address	0x480B C5D4	**Instance**	ISP_CSI1B
Description	Memory channel vertical framing register		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								COUNT								RESERVED															

Bits	Field Name	Description	Type	Reset
31:28	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x0
27:16	COUNT	Defines the line count to be read from memory. From 1 to 4095 lines.	RW	0x001
15:0	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x0000

Table 12-175. Register Call Summary for Register CSI1B_LCM_VSIZE

Camera ISP Register Manual

- Camera ISP_CSI1B Register Summary: [0]

Table 12-176. CSI1B_LCM_HSIZE

Address Offset	0x0000 01D8		
Physical Address	0x480B C5D8	**Instance**	ISP_CSI1B
Description	Memory read channel horizontal framing register.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								COUNT								RESERVED								SKIP							

Bits	Field Name	Description	Type	Reset
31:28	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x0
27:16	COUNT	Horizontal count of pixels to output after the skipped pixels. Valid values: 1-4095.	RW	0x000
15:12	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x0
11:0	SKIP	Horizontal count of pixels to skip after the start of the line. Valid values: 0-4095. 0 disables pixel skipping	RW	0x000

Table 12-177. Register Call Summary for Register CSI1B_LCM_HSIZE

Camera ISP Register Manual

- Camera ISP_CSI1B Register Summary: [0]

Table 12-178. CSI1B_LCM_PREFETCH

Address Offset	0x0000 01DC		
Physical Address	0x480B C5DC	**Instance**	ISP_CSI1B
Description	This register defines the amount of data to be fetched from memory. Is must be consistent with the CSI1_LCM_HSIZE register (check programing model)		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																HWORDS													RESERVED		

Bits	Field Name	Description	Type	Reset
31:14	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x00000
13:3	HWORDS	64 bit words to read from memory for each line of the image. Possible values 1..2047	RW	0x001
2:0	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x0

Table 12-179. Register Call Summary for Register CSI1B_LCM_PREFETCH

Camera ISP Register Manual

- Camera ISP_CSI1B Register Summary: [0]

Table 12-180. CSI1B_LCM_SRC_ADDR

Address Offset	0x0000 01E0		
Physical Address	0x480B C5E0	**Instance**	ISP_CSI1B
Description	Memory channel source address register This register sets the 32-bit memory address where the pixel data are stored. Note that the 5 LSBs are ignored: the address should be aligned on a 32-byte boundary.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDR																											RESERVED			

Bits	Field Name	Description	Type	Reset
31:5	ADDR	27 most significant bits of the 32-bit address.	RW	0x0000000
4:0	RESERVED	5 least significant bits of the 32-bit address. Write 0s for future compatibility. Reads return 0.	RW	0x00

Table 12-181. Register Call Summary for Register CSI1B_LCM_SRC_ADDR

Camera ISP Register Manual

- Camera ISP_CSI1B Register Summary: [0]

Table 12-182. CSI1B_LCM_SRC_OFST

Address Offset	0x0000 01E4		
Physical Address	0x480B C5E4	**Instance**	ISP_CSI1B
Description	Memory channel source offset register. This register sets the offset, which is applied on the source address after each line is read from memory. For example, it enables to perform 2D data transfers of the pixel data from a frame buffer. In such case, the pixel data and frame buffer data must have the same data format. Note that the 5 LSBs are ignored: the offset must be a multiple of 32 bytes.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
OFST																											RESERVED			

Bits	Field Name	Description	Type	Reset
31:5	OFST	Line offset programmed in bytes. If OFST = 0, the data is read contiguously from memory. Otherwise, OFST sets the source offset between the first pixel of the previous line and the first pixel of the current line.	RW	0x0000000
4:0	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x00

Table 12-183. Register Call Summary for Register CSI1B_LCM_SRC_OFST

Camera ISP Register Manual

- Camera ISP_CSI1B Register Summary: [0]

Table 12-184. CSI1B_LCM_DST_ADDR

Address Offset	0x0000 01E8		
Physical Address	0x480B C5E8	**Instance**	ISP_CSI1B
Description	Memory channel destination address. This register sets the 32-bit memory address where the pixel data are stored. Note that the 5 LSBs are ignored: the address should be aligned on a 32-byte boundary.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDR																								RESERVED							

Bits	Field Name	Description	Type	Reset
31:5	ADDR	27 most significant bits of the 32-bit address.	RW	0x0000000
4:0	RESERVED	5 least significant bits of the 32-bit address. Write 0s for future compatibility. Reads return 0.	RW	0x00

Table 12-185. Register Call Summary for Register CSI1B_LCM_DST_ADDR

Camera ISP Register Manual

- Camera ISP_CSI1B Register Summary: [0]

Table 12-186. CSI1B_LCM_DST_OFST

Address Offset	0x0000 01EC		
Physical Address	0x480B C5EC	**Instance**	ISP_CSI1B
Description	Memory channel destination offset register. This register sets the offset, which is applied on the destination address after each line is written to memory. For example, it enables to perform 2D data transfers of the pixel data into a frame buffer. In such case, the pixel data and frame buffer data must have the same data format. Note that the 5 LSBs are ignored: the offset must be a multiple of 32 bytes.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
OFST																								RESERVED							

Bits	Field Name	Description	Type	Reset
31:5	OFST	Line offset programmed in bytes. If OFST = 0, the data is written contiguously to memory if possible. At the end of a line only full 32 bit words are written, creating eventually gaps at the end of lines. Otherwise, OFST sets the destination offset between the first pixel of the previous line and the first pixel of the current line.	RW	0x0000000
4:0	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x00

Table 12-187. Register Call Summary for Register CSI1B_LCM_DST_OFST

Camera ISP Register Manual

- Camera ISP_CSI1B Register Summary: [0]

12.6.5 Camera ISP_CCDC Registers

12.6.5.1 Camera ISP_CCDC Register Summary

Table 12-188. ISP_CCDC Register Summary

Register Name	Type	Register Width (Bits)	Address Offset	Physical Address
CCDC_PID	R	32	0x0000 0000	0x480B C600
CCDC_PCR	RW	32	0x0000 0004	0x480B C604
CCDC_SYN_MODE	RW	32	0x0000 0008	0x480B C608
CCDC_HD_VD_WID	RW	32	0x0000 000C	0x480B C60C
CCDC_PIX_LINES	RW	32	0x0000 0010	0x480B C610
CCDC_HORZ_INFO	RW	32	0x0000 0014	0x480B C614
CCDC_VERT_START	RW	32	0x0000 0018	0x480B C618
CCDC_VERT_LINES	RW	32	0x0000 001C	0x480B C61C
CCDC_CULLING	RW	32	0x0000 0020	0x480B C620
CCDC_HSIZE_OFF	RW	32	0x0000 0024	0x480B C624
CCDC_SDOFST	RW	32	0x0000 0028	0x480B C628
CCDC_SDR_ADDR	RW	32	0x0000 002C	0x480B C62C
CCDC_CLAMP	RW	32	0x0000 0030	0x480B C630
CCDC_DCSUB	RW	32	0x0000 0034	0x480B C634
CCDC_COLPTN	RW	32	0x0000 0038	0x480B C638
CCDC_BLKCMP	RW	32	0x0000 003C	0x480B C63C
CCDC_FPC	RW	32	0x0000 0040	0x480B C640
CCDC_FPC_ADDR	RW	32	0x0000 0044	0x480B C644
CCDC_VDINT	RW	32	0x0000 0048	0x480B C648
CCDC_ALAW	RW	32	0x0000 004C	0x480B C64C
CCDC_REC656IF	RW	32	0x0000 0050	0x480B C650
CCDC_CFG	RW	32	0x0000 0054	0x480B C654
CCDC_FMTCFG	RW	32	0x0000 0058	0x480B C658
CCDC_FMT_HORZ	RW	32	0x0000 005C	0x480B C65C
CCDC_FMT_VERT	RW	32	0x0000 0060	0x480B C660
CCDC_FMT_ADDR_i[1]	RW	32	0x0000 0064 + (*i 0x4)	0x480B C664 + (i * 0x4)
CCDC_PRGEVEN0	RW	32	0x0000 0084	0x480B C684
CCDC_PRGEVEN1	RW	32	0x0000 0088	0x480B C688
CCDC_PRGODD0	RW	32	0x0000 008C	0x480B C68C
CCDC_PRGODD1	RW	32	0x0000 0090	0x480B C690
CCDC_VP_OUT	RW	32	0x0000 0094	0x480B C694
CCDC_LSC_CONFIG	RW	32	0x0000 0098	0x480B C698
CCDC_LSC_INITIAL	RW	32	0x0000 009C	0x480B C69C
CCDC_LSC_TABLE_BASE	RW	32	0x0000 00A0	0x480B C6A0
CCDC_LSC_TABLE_OFFSET	RW	32	0x0000 00A4	0x480B C6A4

(1) i = 0 to 7

12.6.5.2 Camera ISP_CCDC Register Description

Table 12-189. CCDC_PID

Address Offset	0x0000 0000		
Physical Address	0x480B C600	**Instance**	ISP_CCDC
Description	PERIPHERAL ID REGISTER		
Type	R		

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10 9 8	7 6 5 4 3 2 1 0
RESERVED	TID	CID	PREV

Bits	Field Name	Description	Type	Reset
31:24	RESERVED	Write 0s for future compatibility. Reads return 0.	R	0x00
23:16	TID	Peripheral identification: CCDC module	R	0x01
15:8	CID	Class identification: Camera ISP	R	0xFE
7:0	PREV	Peripheral revision number	R	TI internal data

Table 12-190. Register Call Summary for Register CCDC_PID

Camera ISP Register Manual

- Camera ISP_CCDC Register Summary: [0]

Table 12-191. CCDC_PCR

Address Offset	0x0000 0004		
Physical Address	0x480B C604	**Instance**	ISP_CCDC
Description	PERIPHERAL CONTROL REGISTER		
Type	RW		

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10 9 8	7 6 5 4 3 2 1 0
RESERVED			RESERVED / RESERVED / BUSY / ENABLE

Bits	Field Name	Description	Type	Reset
31:4	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x0000000
3	RESERVED	Write 0s for future compatibility. Read returns 0.	RW	0x0
2	RESERVED	Reserved bit field. Write the reset value.	RW	0x0
1	BUSY	CCDC module busy. 0x0: Module is not busy. 0x1: Module is busy.	R	0x0
0	ENABLE	CCDC module enable. This bit is latched by VD (start of frame) 0x0: Disable module. 0x1: Enable module.	RW	0x0

Table 12-192. Register Call Summary for Register CCDC_PCR

Camera ISP Basic Programming Model

- Enable/Disable Hardware: [0] [1] [2]
- Events and Status Checking: [3] [4] [5]
- Register Accessibility During Frame Processing: [6]
- Interframe Operations: [7]

Camera ISP Use Case and Tips

- Programming Flow :

Camera ISP Register Manual

- Camera ISP_CCDC Register Summary: [10]

Table 12-193. CCDC_SYN_MODE

Address Offset	0x0000 0008		
Physical Address	0x480B C608	**Instance**	ISP_CCDC
Description	SYNC and mode set register		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED												SDR2RSZ	VP2SDR	WEN	VDHDEN	FLDSTAT	LPF	INPMOD		PACK8	DATSIZ			FLDMODE	DATAPOL	EXWEN	FLDPOL	HDPOL	VDPOL	FLDOUT	VDHDOUT

Bits	Field Name	Description	Type	Reset
31:20	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x000
19	SDR2RSZ	Memory port output into the RESIZER input. Controls whether or not the memory port output data are forwarded to the RESIZER module input port. This does not depend on the state of the CCDC_SYN_MODE.WEN bit. This bit must only be set if the CCDC module receives directly YUV422 data. The input frame size to the RESIZER module is the same as the output frame size to the memory port. The data are simultaneously written to memory if the WEN bit is set while sending the same data to the RESIZER module. The PREVIEW module can also write to the RESIZER module: this bit takes precedence over the PREVIEW module settings. This bit is latched by the VS sync pulse. 0x0: Disable 0x1: Enable	RW	0x0
18	VP2SDR	Video port output enable to the output formatter.. Controls whether the video port data is forwarded to the output formatter or not. If CCDC_SYN_MODE .WEN= 1, the video port data is written to memory. Note that if field is set, then SDRAM line (VERT_START.SLVx) and pixel start (HORZ_INFO.SPH) are with respect to the video port output (and not the original input) This bit field is latched by the VS sync pulse. 0x0: Disable 0x1: Enable	RW	0x0

Bits	Field Name	Description	Type	Reset
17	WEN	Data write enable. Controls whether the CCDC module output data are written to memory or not. This bit field is latched by the VS sync pulse. 0x0: Disable 0x1: Enable	RW	0x0
16	VDHDEN	Timing generator enable. If HS/VS sync pulses are defined as output signals, activates the internal timing generator. If HS/VS sync pulses are defined as input signals, activates internal timing generator to synchronize with HS/VS. This bit must be set to 1 when HS and VS signals are used. 0x0: Disable 0x1: Enable	RW	0x0
15	FLDSTAT	cam_fld signal status. This bit field applies only if the CCDC module is configured to work in interlaced mode: CCDC_SYN_MODE.FLDMODE = "interlaced". It indicates the status of the current field. 0x0: Odd field 0x1: Even field	R	0x0
14	LPF	Three-tap low pass (antialiasing) filter enable. This bit field is latched by the VS sync pulse. 0x0: Filter is disabled. 0x1: Filter is enabled.	RW	0x0
13:12	INPMOD	cam_d format in SYNC mode. Sets the data input format. 0x0: Raw data 0x1: YCbCr data on 16 bits. It is required to enable the 8 to 16-bit bridge in the ISP_CTRL register. 0x2: YCbCr data on 8 bits.	RW	0x0
11	PACK8	Data packing. Sets the data packing configuration when the data is written to memory. 0x0: Normal mode: 16 bits/pixel. 0x1: Pack mode: 8 bits/pixel.	RW	0x0
10:8	DATSIZ	cam_d signal width in SYNC mode. Valid only when CCDC_SYN_MODE.INPMOD = "raw data". 0x0: cam_d is 8 bits but the 8 to 16-bit bridge is enabled in the ISP_CTRL register. 0x4: cam_d is 12 bits 0x5: cam_d is 11 bits 0x6: cam_d is 10 bits 0x7: cam_d is 8 bits	RW	0x0
7	FLDMODE	cam_fld signal mode. 0x0: Progressive mode. cam_fld not used. 0x1: Interlaced mode. cam_fld used.	RW	0x0
6	DATAPOL	cam_d signal polarity. 0x0: Normal 0x1: One's complement	RW	0x0

Bits	Field Name	Description	Type	Reset
5	EXWEN	External write enable selection. The cam_wen signal can be used as an external memory write-enable signal. The data is stored to memory only if cam_hs, cam_vs and cam_wen signals are asserted. 0x0: cam_wen is not used. 0x1: cam_wen is used	RW	0x0
4	FLDPOL	cam_fld signal polarity. 0x0: Positive 0x1: Negative	RW	0x0
3	HDPOL	Sets the cam_hs signal polarity. 0x0: Positive 0x1: Negative	RW	0x0
2	VDPOL	cam_vs signal polarity 0x0: Positive 0x1: Negative	RW	0x0
1	FLDOUT	cam_fld signal direction. 0x0: Input 0x1: Output	RW	0x0
0	VDHDOUT	cam_hs and cam_vs signal directions. 0x0: Input 0x1: Output	RW	0x0

Table 12-194. Register Call Summary for Register CCDC_SYN_MODE

Camera ISP Environment
- ITU-R BT.656 Protocol and Data Format (8, 10 Bits): [0]

Camera ISP Functional Description
- CCDC: [1] [2] [3] [4] [5] [6] [7] [8] [9] [10] [11] [12] [13] [14] [15] [16] [17] [18]
- Resizer: [19]

Camera ISP Basic Programming Model
- CCDC Hardware Setup/Initialization: [20] [21] [22] [23] [24] [25] [26] [27] [28] [29] [30] [31] [32] [33] [34] [35] [36] [37] [38] [39] [40] [41] [42] [43] [44] [45] [46] [47]
- Events and Status Checking: [48] [49] [50] [51]
- Register Accessibility During Frame Processing: [52] [53] [54] [55] [56] [57] [58] [59] [60]
- CCDC Operations: [61] [62] [63] [64] [65] [66] [67] [68] [69] [70] [71] [72] [73] [74] [75] [76] [77] [78] [79] [80] [81] [82] [83] [84] [85] [86] [87] [88] [89] [90]

Camera ISP Use Case and Tips
- Programming Flow :

Camera ISP Register Manual
- Camera ISP_CCDC Register Summary: [94]
- Camera ISP_CCDC Register Description: [95] [96] [97] [98] [99] [100] [101] [102] [103]

Table 12-195. CCDC_HD_VD_WID

Address Offset	0x0000 000C		
Physical Address	0x480B C60C	**Instance**	ISP_CCDC
Description	SYNC WIDTH CONTROL REGISTER		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								HDW								RESERVED				VDW											

Bits	Field Name	Description	Type	Reset
31:28	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x0
27:16	HDW	Sets the width of the HS sync pulse if set as output. The width of the pulse is (HDW+1) pixel clocks. Not used when HS/VS sync pulses are input signals (CCDC_SYN_MODE.VDHDOUT = 0). This bit field is latched by the VS sync pulse.	RW	0x000
15:12	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x0
11:0	VDW	Sets the width of the VS sync pulse is set as output. The width of the pulse is (VDW+1) lines. Not used when HS/VS sync pulses are input signals (CCDC_SYN_MODE.VDHDOUT = 0). This bit field is latched by the VS sync pulse.	RW	0x000

Table 12-196. Register Call Summary for Register CCDC_HD_VD_WID

Camera ISP Functional Description
- CCDC: [0] [1]

Camera ISP Basic Programming Model
- CCDC Hardware Setup/Initialization: [2]
- Register Accessibility During Frame Processing: [3]
- CCDC Operations: [4]

Camera ISP Register Manual
- Camera ISP_CCDC Register Summary: [5]

Table 12-197. CCDC_PIX_LINES

Address Offset	0x0000 0010		
Physical Address	0x480B C610	**Instance**	ISP_CCDC
Description	SIZE CONTROL REGISTER		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PPLN																HLPRF															

Bits	Field Name	Description	Type	Reset
31:16	PPLN	Pixels per line Sets the number of pixel clock periods in one line. HD period = (PPLN + 1) pixel clocks. Not used when HS/VS sync pulses are input signals (CCDC_SYN_MODE.VDHDOUT = 0). This bit field is latched by the VS sync pulse	RW	0x0000
15:0	HLPRF	Half line per field or frame Sets the number of half lines per frame or field. VD period = (HLPFR + 1)/2 lines. Not used when HS/VS sync pulses are input signals (CCDC_SYN_MODE.VDHDOUT = 0). Sets the internal timing generator to generate the correct number of HS pulses in between two VS pulses. If the sensor is interlaced, with a total of N lines, then this field must be set to N. This means that N half lines are written for each field. If the sensor is progressive, then this register must be set to be twice the number of lines to be written. For example, if sensor outputs N lines, this field must be set to 2N. This bit field is latched by the VS sync pulse.	RW	0x0000

Table 12-198. Register Call Summary for Register CCDC_PIX_LINES

Camera ISP Functional Description • CCDC: [0] [1]
Camera ISP Basic Programming Model • CCDC Hardware Setup/Initialization: [2] • Register Accessibility During Frame Processing: [3] • CCDC Operations: [4] [5] [6]
Camera ISP Register Manual • Camera ISP_CCDC Register Summary: [7]

Table 12-199. CCDC_HORZ_INFO

Address Offset	0x0000 0014		
Physical Address	0x480B C614	**Instance**	ISP_CCDC
Description	HORIZONTAL PIXEL INFO REGISTER		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								SPH								RESERVED								NPH							

Bits	Field Name	Description	Type	Reset
31	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x0
30:16	SPH	Start pixel horizontal Sets the pixel clock position at which data output to memory begins. It is measured from the start of HS. This bit field is latched by the VS sync pulse.	RW	0x0000
15	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x0
14:0	NPH	Number of pixels horizontal Sets the number of horizontal pixels output to memory. The number of pixels output is (NPH + 1). This bit field is latched by the VS sync pulse.	RW	0x0100

Table 12-200. Register Call Summary for Register CCDC_HORZ_INFO

Camera ISP Functional Description • CCDC: [0] [1]
Camera ISP Basic Programming Model • CCDC Hardware Setup/Initialization: [2] • Register Accessibility During Frame Processing: [3] [4] • CCDC Operations: [5] [6]
Camera ISP Register Manual • Camera ISP_CCDC Register Summary: [7]

Table 12-201. CCDC_VERT_START

Address Offset	0x0000 0018		
Physical Address	0x480B C618	**Instance**	ISP_CCDC
Description	VERTICAL LINE START REGISTER		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								SLV0								RESERVED								SLV1							

Bits	Field Name	Description	Type	Reset
31	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x0
30:16	SLV0	Start line vertical - field0 Sets the line at which data output to memory begins. It is measured from the start of the VS sync pulse. This bit field is latched by the VS sync pulse.	RW	0x0000
15	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x0
14:0	SLV1	Start line vertical - field1 Sets the line at which data output to memory begins. It is measured from the start of the VS sync pulse. For a progressive sensor, this bit field is ignored. This bit field is latched by the VS sync pulse.	RW	0x0000

Table 12-202. Register Call Summary for Register CCDC_VERT_START

Camera ISP Functional Description
- CCDC: [0] [1] [2]

Camera ISP Basic Programming Model
- CCDC Hardware Setup/Initialization: [3]
- Register Accessibility During Frame Processing: [4] [5] [6]
- CCDC Operations: [7] [8] [9]

Camera ISP Register Manual
- Camera ISP_CCDC Register Summary: [10]

Table 12-203. CCDC_VERT_LINES

Address Offset	0x0000 001C		
Physical Address	0x480B C61C	**Instance**	ISP_CCDC
Description	VERTICAL LINE NUMBER REGISTER		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																NLV															

Bits	Field Name	Description	Type	Reset
31:15	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x00000
14:0	NLV	Number of lines - vertical direction Sets the number of vertical lines output to memory. The number of lines output is (NLV + 1). This bit is latched by the VS sync pulse.	RW	0x0000

Table 12-204. Register Call Summary for Register CCDC_VERT_LINES

Camera ISP Functional Description • CCDC: [0] [1] [2]
Camera ISP Basic Programming Model • CCDC Hardware Setup/Initialization: [3] • Register Accessibility During Frame Processing: [4] • CCDC Operations: [5]
Camera ISP Register Manual • Camera ISP_CCDC Register Summary: [6]

Table 12-205. CCDC_CULLING

Address Offset	0x0000 0020		
Physical Address	0x480B C620	**Instance**	ISP_CCDC
Description	CULL CONTROL REGISTER		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CULHEVN								CULHODD								RESERVED								CULV							

Bits	Field Name	Description	Type	Reset
31:24	CULHEVN	Horizontal culling patterns for even lines. Sets an 8-bit mask (0:cull, 1:retain). The LSB is the 1st pixel and the MSB is the 8th pixel. Then, the pattern repeats. This bit field is latched by the VS sync pulse.	RW	0xFF
23:16	CULHODD	Horizontal culling patterns for odd lines. Sets an 8-bit mask (0:cull, 1:retain). The LSB is the 1st pixel and the MSB is the 8th pixel. Then, the pattern repeats. This bit field is latched by the VS sync pulse.	RW	0xFF
15:8	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x00
7:0	CULV	Vertical culling pattern. Sets an 8-bit mask (0:cull, 1:retain). The LSB is the 1st line and the MSB is the 8th line. Then, the pattern repeats. This bit field is latched by the VS sync pulse.	RW	0xFF

Table 12-206. Register Call Summary for Register CCDC_CULLING

Camera ISP Functional Description • CCDC: [0] [1] [2] [3] [4] [5] [6] [7]
Camera ISP Basic Programming Model • CCDC Hardware Setup/Initialization: [8] • Register Accessibility During Frame Processing: [9] • CCDC Operations: [10] [11] [12] [13]
Camera ISP Register Manual • Camera ISP_CCDC Register Summary: [14]

Table 12-207. CCDC_HSIZE_OFF

Address Offset	0x0000 0024		
Physical Address	0x480B C624	**Instance**	ISP_CCDC
Description	HORIZONTAL SIZE REGISTER		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																LNOFST															

Bits	Field Name	Description	Type	Reset
31:16	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x0000
15:0	LNOFST	Line offset. Sets the offset for each output line to memory. The offset must be a multiple of 32 bytes: the 5 least significant bits are ignored. Usually the line offset is equal to the line length in bytes, that is, the line length must be a multiple of 32 bytes. If LNOFST = 0, the data is written again and again over the same line. For optimal performance in the system, the address offset must be on a 256-byte boundary. This bit field is latched by the VS sync pulse.	RW	0x0000

Table 12-208. Register Call Summary for Register CCDC_HSIZE_OFF

Camera ISP Functional Description
- CCDC: [0]

Camera ISP Basic Programming Model
- CCDC Hardware Setup/Initialization: [1]
- Register Accessibility During Frame Processing: [2]
- CCDC Operations: [3]

Camera ISP Register Manual
- Camera ISP_CCDC Register Summary: [4]

Table 12-209. CCDC_SDOFST

Address Offset	0x0000 0028		
Physical Address	0x480B C628	**Instance**	ISP_CCDC
Description	MEMORY OFFSET REGISTER		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																FIINV	FOFST	LOFST0				LOFST1				LOFST2				LOFST3	

Bits	Field Name	Description	Type	Reset
31:15	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x00000
14	FIINV	Field identification signal inverse This bit field is latched by the VS sync pulse. 0x0: Non inverse 0x1: Inverse	RW	0x0

Bits	Field Name	Description	Type	Reset
13:12	FOFST	Line offset value This bit field is latched by the VS sync pulse. 0x0: +1 line 0x1: +2 lines 0x2: +3 lines 0x3: +4 lines	RW	0x0
11:9	LOFST0	Line offset values of even lines and even fields (field id = 0). This bit field is latched by the VS sync pulse. 0x0: +1 line 0x1: +2 lines 0x2: +3 lines 0x3: +4 lines 0x4: -1 line 0x5: -2 lines 0x6: -3 lines 0x7: -4 lines	RW	0x0
8:6	LOFST1	Line offset values of odd lines and even fields (field id = 0). This bit field is latched by the VS sync pulse. 0x0: +1 line 0x1: +2 lines 0x2: +3 lines 0x3: +4 lines 0x4: -1 line 0x5: -2 lines 0x6: -3 lines 0x7: -4 lines	RW	0x0
5:3	LOFST2	Line offset values of even lines and odd fields (field id = 1). This bit field is latched by the VS sync pulse. 0x0: +1 line 0x1: +2 lines 0x2: +3 lines 0x3: +4 lines 0x4: -1 line 0x5: -2 lines 0x6: -3 lines 0x7: -4 lines	RW	0x0
2:0	LOFST3	Line offset values of odd lines and odd fields (field id = 1). This bit field is latched by the VS sync pulse. 0x0: +1 line 0x1: +2 lines 0x2: +3 lines 0x3: +4 lines 0x4: -1 line 0x5: -2 lines 0x6: -3 lines 0x7: -4 lines	RW	0x0

Table 12-210. Register Call Summary for Register CCDC_SDOFST

Camera ISP Functional Description • CCDC: [0] [1] [2] [3] [4] [5] [6] [7]
Camera ISP Basic Programming Model • CCDC Hardware Setup/Initialization: [8] • Register Accessibility During Frame Processing: [9] • CCDC Operations: [10]
Camera ISP Register Manual • Camera ISP_CCDC Register Summary: [11]

Table 12-211. CCDC_SDR_ADDR

Address Offset	0x0000 002C		
Physical Address	0x480B C62C	**Instance**	ISP_CCDC
Description	MEMORY ADDRESS REGISTER		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDR																															

Bits	Field Name	Description	Type	Reset
31:0	ADDR	Memory address Sets the CCDC module output address. The address should be aligned on a 32-byte boundary: the 5 least significant bits are ignored. For optimal performance in the system, the address must be on a 256-byte boundary. This bit field is latched by the VS sync pulse.	RW	0x00000000

Table 12-212. Register Call Summary for Register CCDC_SDR_ADDR

Camera ISP Functional Description • CCDC: [0] [1]
Camera ISP Basic Programming Model • CCDC Hardware Setup/Initialization: [2] • Register Accessibility During Frame Processing: [3] • CCDC Operations: [4]
Camera ISP Register Manual • Camera ISP_CCDC Register Summary: [5]

Table 12-213. CCDC_CLAMP

Address Offset	0x0000 0030		
Physical Address	0x480B C630	**Instance**	ISP_CCDC
Description	CLAMP CONTROL REGISTER		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CLAMPEN	OBSLEN			OBSLN				OBST								RESERVED								OBGAIN							

Bits	Field Name	Description	Type	Reset
31	CLAMPEN	Clamp enable Enables clamping based on the calculated average of optical black sample. This bit is latched by the VS sync pulse. 0x0: Disable 0x1: Enable	RW	0x0
30:28	OBSLEN	Optical black sample length Sets the number of optical black sample pixels per line to include in the average calculation. 0x0: 1 pixel 0x1: 2 pixels 0x2: 4 pixels 0x3: 8 pixels 0x4: 16 pixels	RW	0x0
27:25	OBSLN	Optical black sample lines Sets the number of optical black sample lines to include in the average calculation. 0x0: 1 line 0x1: 2 lines 0x2: 4 lines 0x3: 8 lines 0x4: 16 lines	RW	0x0
24:10	OBST	Start pixel of optical black samples Start pixel position of optical black samples specified from the start of HS in pixel clocks.	RW	0x0000
9:5	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x00
4:0	OBGAIN	Gain to apply to the optical black average The gain value is in U5Q4 fixed point representation (0 to 1.9375).	RW	0x10

Table 12-214. Register Call Summary for Register CCDC_CLAMP

Camera ISP Functional Description
- CCDC: [0] [1] [2] [3] [4]

Camera ISP Basic Programming Model
- CCDC Hardware Setup/Initialization: [5] [6] [7] [8] [9] [10] [11]
- Register Accessibility During Frame Processing: [12]
- CCDC Operations: [13] [14] [15] [16] [17] [18] [19]

Camera ISP Register Manual
- Camera ISP_CCDC Register Summary: [20]
- Camera ISP_CCDC Register Description: [21]

Table 12-215. CCDC_DCSUB

Address Offset	0x0000 0034		
Physical Address	0x480B C634	**Instance**	ISP_CCDC
Description	DC CLAMP REGISTER		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																DCSUB															

Bits	Field Name	Description	Type	Reset
31:14	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x00000
13:0	DCSUB	DC value to subtract from the data. Sets the DC value to be subtracted from the data when optical black sampling is disabled: CCDC_CLAMP.CLAMPEN = 0 NOTE: In OMAP3430, this function does not clip negative results to 0 for YUV 8 bit input or REC656 input modes (SYN_ MODE.INPMOD == 2 || REC656IF.REC656ON == 1).	RW	0x0000

Table 12-216. Register Call Summary for Register CCDC_DCSUB

Camera ISP Functional Description • CCDC: [0] [1]
Camera ISP Basic Programming Model • CCDC Hardware Setup/Initialization: [2] [3] • CCDC Operations: [4]
Camera ISP Register Manual • Camera ISP_CCDC Register Summary: [5]

Table 12-217. CCDC_COLPTN

Address Offset	0x0000 0038		
Physical Address	0x480B C638	**Instance**	ISP_CCDC
Description	COLOR PATTERN REGISTER		
Type	RW		

31 30	29 28	27 26	25 24	23 22	21 20	19 18	17 16	15 14	13 12	11 10	9 8	7 6	5 4	3 2	1 0
CP3LPC3	CP3LPC2	CP3LPC1	CP3LPC0	CP2PLC3	CP2PLC2	CP2PLC1	CP2PLC0	CP1PLC3	CP1PLC2	CP1PLC1	CP1PLC0	CP0PLC3	CP0PLC2	CP0PLC1	CP0PLC0

Bits	Field Name	Description	Type	Reset
31:30	CP3LPC3	Color pattern, 3rd line, pixel counter = 0 0x0: R/Ye 0x1: Gr/Cy 0x2: Gb/G 0x3: B/Mg	RW	0x0
29:28	CP3LPC2	Color pattern, 3rd line, pixel counter = 2 0x0: R/Ye 0x1: Gr/Cy 0x2: Gb/G 0x3: B/Mg	RW	0x0
27:26	CP3LPC1	Color pattern, 3rd line, pixel counter = 1 0x0: R/Ye 0x1: Gr/Cy 0x2: Gb/G 0x3: B/Mg	RW	0x0
25:24	CP3LPC0	Color pattern, 3rd line, pixel counter = 0 0x0: R/Ye 0x1: Gr/Cy 0x2: Gb/G 0x3: B/Mg	RW	0x0

Bits	Field Name	Description	Type	Reset
23:22	CP2PLC3	Color pattern, 2nd line, pixel counter = 3 0x0: R/Ye 0x1: Gr/Cy 0x2: Gb/G 0x3: B/Mg	RW	0x0
21:20	CP2PLC2	Color pattern, 2nd line, pixel counter = 2 0x0: R/Ye 0x1: Gr/Cy 0x2: Gb/G 0x3: B/Mg	RW	0x0
19:18	CP2PLC1	Color pattern, 2nd line, pixel counter = 1 0x0: R/Ye 0x1: Gr/Cy 0x2: Gb/G 0x3: B/Mg	RW	0x0
17:16	CP2PLC0	Color pattern, 2nd line, pixel counter = 0 0x0: R/Ye 0x1: Gr/Cy 0x2: Gb/G 0x3: B/Mg	RW	0x0
15:14	CP1PLC3	Color pattern, 1st line, pixel counter = 3 0x0: R/Ye 0x1: Gr/Cy 0x2: Gb/G 0x3: B/Mg	RW	0x0
13:12	CP1PLC2	Color pattern, 1st line, pixel counter = 2 0x0: R/Ye 0x1: Gr/Cy 0x2: Gb/G 0x3: B/Mg	RW	0x0
11:10	CP1PLC1	Color pattern, 1st line, pixel counter = 1 0x0: R/Ye 0x1: Gr/Cy 0x2: Gb/G 0x3: B/Mg	RW	0x0
9:8	CP1PLC0	Color pattern, 1st line, pixel counter = 0 0x0: R/Ye 0x1: Gr/Cy 0x2: Gb/G 0x3: B/Mg	RW	0x0
7:6	CP0PLC3	Color pattern, 0th line, pixel counter = 3 0x0: R/Ye 0x1: Gr/Cy 0x2: Gb/G 0x3: B/Mg	RW	0x0

Bits	Field Name	Description	Type	Reset
5:4	CP0PLC2	Color pattern, 0th line, pixel counter = 2 0x0: R/Ye 0x1: Gr/Cy 0x2: Gb/G 0x3: B/Mg	RW	0x0
3:2	CP0PLC1	Color pattern, 0th line, pixel counter = 1 0x0: R/Ye 0x1: Gr/Cy 0x2: Gb/G 0x3: B/Mg	RW	0x0
1:0	CP0PLC0	Color pattern, 0th line, pixel counter = 0 0x0: R/Ye 0x1: Gr/Cy 0x2: Gb/G 0x3: B/Mg	RW	0x0

Table 12-218. Register Call Summary for Register CCDC_COLPTN

Camera ISP Functional Description
- CCDC: [0]

Camera ISP Basic Programming Model
- CCDC Hardware Setup/Initialization: [1]
- CCDC Operations: [2] [3]
- Summary of Constraints: [4]

Camera ISP Register Manual
- Camera ISP_CCDC Register Summary: [5]

Table 12-219. CCDC_BLKCMP

Address Offset	0x0000 003C		
Physical Address	0x480B C63C	**Instance**	ISP_CCDC
Description	BLACK COMPENSATION REGISTER		
Type	RW		

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10 9 8	7 6 5 4 3 2 1 0
R_YE	GR_CY	GB_G	B_MG

Bits	Field Name	Description	Type	Reset
31:24	R_YE	Black-level compensation, R/Ye pixels. 2's complement, MSB is sign bit. The range is -128 to +127.	RW	0x00
23:16	GR_CY	Black-level compensation, Gr/Cy pixels. 2's complement, MSB is sign bit. The range is -128 to +127.	RW	0x00
15:8	GB_G	Black-level compensation, Gb/G pixels. 2's complement, MSB is sign bit. The range is -128 to +127.	RW	0x00
7:0	B_MG	Black-level compensation, B/Mg pixels. 2's complement, MSB is sign bit. The range is -128 to +127.	RW	0x00

Table 12-220. Register Call Summary for Register CCDC_BLKCMP

Camera ISP Functional Description

- CCDC: [0]

Camera ISP Basic Programming Model

- CCDC Hardware Setup/Initialization: [2]
- CCDC Operations: [3]
- Summary of Constraints: [4]

Camera ISP Register Manual

- Camera ISP_CCDC Register Summary: [5]

Table 12-221. CCDC_FPC

Address Offset	0x0000 0040		
Physical Address	0x480B C640	**Instance**	ISP_CCDC
Description	FAULT PIXEL CORRECTION REGISTER		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																FPERR	FPCEN	FPNUM													

Bits	Field Name	Description	Type	Reset
31:17	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x0000
16	FPERR	Fault pixel correction error This bit is set when the CCDC module is unable to fetch the required fault pixel table entry in time. Write 1 to clear the error or end_of_frame clears it automatically for the next frame. For example, the current pixel being processed has coordinates of 256/512 (256th line and 512th pixel in that line) and it must be corrected. If the entry in the fault pixel table that must be used has coordinates 256/256, then the current pixel cannot be corrected since the correct entry is not loaded in time. Note that there is no error recovery mechanism in the CCDC; if this bit is set at anytime in a frame, there are no more fault pixels corrected in that frame. Firmware is responsible for making sure that there is enough bandwidth in the system to allow for loading of the fault pixel table. Alternately, decreasing the frequency of the fault pixels to be corrected enhances the chances of this bit not being set. 0x0: No error 0x1: Error	RW	0x0

Bits	Field Name	Description	Type	Reset
15	FPCEN	Fault pixel correction enable. Upon setting this bit, and as long as it remains enabled, the fault pixel logic continues to request data and just start over for next frame once the last data of current frame has been received. As soon as the register is set the data are fetched. To disable fault pixel correction, users can write a 0 at any time. However, the disabling only applies after the current frame is processed (busy bit for current frame is 0) This bit should only be written after the FPC_ADDR register below has been set. Also, the other fields in this register have to be set prior to enabling this bit. The required process is: Write(FPC_ADDR) Write(FPC) with this bit (FPC.FPCEN) turned off Write(FPC) with this bit (FPC.FPCEN) turned on while other fields are same as previous write 0x0: Disable 0x1: Enable	RW	0x0
14:0	FPNUM	Number of fault pixels to be corrected in the frame This field should not be changed when the FPCEN is enabled at any time	RW	0x0000

Table 12-222. Register Call Summary for Register CCDC_FPC

Camera ISP Functional Description • CCDC: [0] [3]
Camera ISP Basic Programming Model • CCDC Hardware Setup/Initialization: [4] [5] [6] • Enable/Disable Hardware: [7] [8] [9] • Events and Status Checking: [10] [11] [12] • CCDC Operations: [13] [14] [15] • Event and Status Checking: [16] [17]
Camera ISP Register Manual • Camera ISP Register Description: [18] [19] • Camera ISP_CCDC Register Summary: [20]

Table 12-223. CCDC_FPC_ADDR

Address Offset	0x0000 0044		
Physical Address	0x480B C644	**Instance**	ISP_CCDC
Description	FAULT PIXEL CORRECTION MEMORY ADDRESS		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDR																															

Bits	Field Name	Description	Type	Reset
31:0	ADDR	Memory address Set the memory address of the fault pixel correction table. The address should be aligned to a 64-byte boundary: the 6 LSBs are ignored. Each of the 32-bit table entry contains a 13-bit vertical position, a 14-bit horizontal position and a 5-bit operation field.	RW	0x00000000

Table 12-224. Register Call Summary for Register CCDC_FPC_ADDR

Camera ISP Functional Description • CCDC: [0]

Table 12-224. Register Call Summary for Register CCDC_FPC_ADDR (continued)

Camera ISP Basic Programming Model • CCDC Hardware Setup/Initialization: [1] • Enable/Disable Hardware: [2] • CCDC Operations: [3] • Summary of Constraints: [4]
Camera ISP Register Manual • Camera ISP_CCDC Register Summary: [5]

Table 12-225. CCDC_VDINT

Address Offset	0x0000 0048		
Physical Address	0x480B C648	**Instance**	ISP_CCDC
Description	VD INTERRUPT REGISTER		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								VDINT0								RESERVED								VDINT1							

Bits	Field Name	Description	Type	Reset
31	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x0
30:16	VDINT0	VD0 interrupt timing Specified VDINT0 in units of horizontal lines from the start of the VS sync pulse. The resulting value is VDINT0+1. Note that if the rising edge (or falling edge if programmed) of the HS sync pulse lines up with the rising edge (or falling edge if programmed) of VS sync pulse, the first HS sync pulse is not counted.	RW	0x0000
15	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x0
14:0	VDINT1	VD1 interrupt timing Specifies VDINT1 in units of horizontal lines from the start of the VS sync pulse. The resulting value is VDINT1+1. Note that if the rising edge (or falling edge if programmed) of the HS sync pulse lines up with the rising edge (or falling edge if programmed) of VS sync pulse, the first HS sync pulse is not counted.	RW	0x0000

Table 12-226. Register Call Summary for Register CCDC_VDINT

Camera ISP Basic Programming Model • CCDC Hardware Setup/Initialization: [0] • Events and Status Checking: [1] [2]
Camera ISP Use Case and Tips • Programming Flow :
Camera ISP Register Manual • Camera ISP_CCDC Register Summary: [5]

Table 12-227. CCDC_ALAW

Address Offset	0x0000 004C		
Physical Address	0x480B C64C	**Instance**	ISP_CCDC
Description	ALAW SETTINGS REGISTER		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																												CCDTBL	GWDI		

Bits	Field Name	Description	Type	Reset
31:4	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x0000000
3	CCDTBL	Apply A-Law compression to data saved to memory. 0x0: Disable 0x1: Enable	RW	0x0
2:0	GWDI	A-Law input width 0x3: Bits 12 to 3 0x4: Bits 11 to 2 0x5: Bits 10 to 1 0x6: Bits 9 to 0	RW	0x4

Table 12-228. Register Call Summary for Register CCDC_ALAW

Camera ISP Functional Description
- CCDC: [0] [1] [2]

Camera ISP Basic Programming Model
- CCDC Hardware Setup/Initialization: [3] [4] [5]
- CCDC Operations: [6] [7]

Camera ISP Register Manual
- Camera ISP_CCDC Register Summary: [8]

Table 12-229. CCDC_REC656IF

Address Offset	0x0000 0050		
Physical Address	0x480B C650	**Instance**	ISP_CCDC
Description	ITU-R BT.656 CONFIGURATION REGISTER		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																														ECCFVH	R656ON

Bits	Field Name	Description	Type	Reset
31:2	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x00000000
1	ECCFVH	FVH error correction enable 0x0: Disable 0x1: Enable	RW	0x0
0	R656ON	ITU-R BT656 interface enable 0x0: Disable 0x1: Enable	RW	0x0

Table 12-230. Register Call Summary for Register CCDC_REC656IF

Camera ISP Environment • ITU-R BT.656 Protocol and Data Format (8, 10 Bits): [0]
Camera ISP Functional Description • CCDC: [1] [2] [3]
Camera ISP Basic Programming Model • CCDC Hardware Setup/Initialization: [4] [5] [6] [7] [8] • CCDC Operations: [9] [10] [11]
Camera ISP Register Manual • Camera ISP_CCDC Register Summary: [12] • Camera ISP_CCDC Register Description: [13]

Table 12-231. CCDC_CFG

Address Offset	0x0000 0054		
Physical Address	0x480B C654	**Instance**	ISP_CCDC
Description	CONFIGURATION REGISTER		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																VDLC	RESERVED	MSBINVI	BSWD	Y8POS	RESERVED		WENLOG	FIDMD		BW656	RESERVED	RESERVED	RESERVED	RESERVED	

Bits	Field Name	Description	Type	Reset
31:16	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x0000
15	VDLC	Enable latching function registers on the internal VS sync pulse. If this bit is set, all the register fields that are VS pulse latched take on new values immediately. Care should be taken not to alter fields that can cause undesired behavior to the output data NOTE: In OMAP3430, this bit must be set to 1 by software if the CCDC is to be used. If CCDCFG.VDLC remains set to 0 (default), indeterminate results may occur for ANY register access in the CCDC. For details, see Section 12.5, *Basic Programming Model*. 0x0: Latched on VS 0x1: Not latched on VS	RW	0x0
14	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x0
13	MSBINVI	MSB of chroma input signal stored to memory inverted. 0x0: Normal 0x1: MSB inverted	RW	0x0
12	BSWD	Byte swap data stored to memory. 0x0: Normal 0x1: Swap bytes	RW	0x0
11	Y8POS	Location of Y color component when YCbCr 8-bit data is input. 0x0: Even pixel 0x1: Odd pixel	RW	0x0
10:9	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x0

Bits	Field Name	Description	Type	Reset
8	WENLOG	Valid area settings 0x0: Internal valid signal and WEN signal are ANDed logically. 0x1: Internal valid signal and WEN signal are ORed logically.	RW	0x0
7:6	FIDMD	Settings of field identification detection function. 0x0: FLD signal is latched at the VS timing. The external Field signal is latched when the VD is active and the active edge of the HD signal 0x1: FLD signal is not latched. The field signal is not latched at all 0x2: FLD signal is latched at edge of VS. The field signal is latched on the active edge of the VD signal 0x3: FLD signal is latched on phase of VS and HS. The field signal is latched when the VD signal is active and the HD signal is inactive (opposite phase)	RW	0x0
5	BW656	The data width in ITU-R BT656 input mode. This bit field takes precedence over the CCDC_SYN_MODE.INPMOD and CCDC_DATSIZ bit fields if the ITU mode is enabled with CCDC_REC656IF.R656ON = 1. 0x0: 8 bits 0x1: 10 bits	RW	0x0
4	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x0
3	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x0
2	RESERVED	Reserved. Write 0s for future compatibility. Reads return 0.	RW	0x0
1:0	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x0

Table 12-232. Register Call Summary for Register CCDC_CFG

Camera ISP Functional Description • CCDC: [0]
Camera ISP Basic Programming Model • CCDC Hardware Setup/Initialization: [1] [2] [3] [4] [5] [6] [7] • Register Accessibility During Frame Processing: [8] [9] [10] [11] [12] [13] • CCDC Operations: [14] [15] [16] [17] [18] [19]
Camera ISP Register Manual • Camera ISP_CCDC Register Summary: [20]

Table 12-233. CCDC_FMTCFG

Address Offset	0x0000 0058		
Physical Address	0x480B C658	**Instance**	ISP_CCDC
Description	DATA REFORMATTER/VIDEO IF CONFIG REGISTER		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED													VPIF_FRQ			VPEN	VPIN			PLEN_EVEN				PLEN_ODD				LNUM		LNALT	FMTEN

Bits	Field Name	Description	Type	Reset
31:19	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x0000
18:16	VPIF_FRQ	Video port data ready frequency. This field allows the firmware to control the rate at which the video port delivers new data to the other modules (PREVIEW, H3A, and HIST modules). In effect, this register controls the raw output bandwidth of the PREVIEW, H3A, and HIST. Depending on the input sensor clock, users can set this field appropriately and balance the bandwidth requirements to memory. Given a pixel clock, one must set this bit field with the higher divisor value to lower the memory bandwidth requirement. The video port clock is Interconnect/(VPIF_FRQ + 2). The valid range for VPIF FRQ is 0..6.	RW	0x0
15	VPEN	Video port enable. This bit must be enabled to send data to the PREVIEW, H3A and HIST modules. 0x0: Disable 0x1: Enable	RW	0
14:12	VPIN	10-bit input select for video port. 0x3: bits 12-3 0x4: bits 11-2 0x5: bits 10-1 0x6: bits 9-0	RW	0x4
11:8	PLEN_EVEN	Number of program entries in even line minus 1. If the desired number of program entries is 8, then the value must be set to 7.	RW	0x0
7:4	PLEN_ODD	Number of program entries in odd line minus 1. If the desired number of program entries is 8, then the value must be set to 7.	RW	0x0
3:2	LNUM	Number of output lines from 1 input line 0x0: 1 line 0x1: 2 lines 0x2: 3 lines 0x3: 4 lines	RW	0x0
1	LNALT	Line alternating mode enable. In Line Alternating Mode, even and odd lines are swapped. 0th line output as 1st line and 1st line output as 0th line, and so on. If this bit field is set, the start and number of lines for the formatter (FMTVERT below) must be even. The FMTEN field below must be set in addition for this field to function 0x0: Enable. Normal mode 0x1: Disable. Line alternating mode	RW	0x0
0	FMTEN	Formatter enable. This bit is latched by the VS sync pulse. 0x0: Disable 0x1: Enable	RW	0x0

Table 12-234. Register Call Summary for Register CCDC_FMTCFG

Camera ISP Functional Description

- CCDC: [0] [1] [2] [3] [4]

Table 12-234. Register Call Summary for Register CCDC_FMTCFG (continued)

Camera ISP Basic Programming Model • CCDC Hardware Setup/Initialization: [5] [6] [7] [8] [9] [10] [11] [12] [13] [14] [15] [16] • Register Accessibility During Frame Processing: [17] • CCDC Operations: [18] [19] [20] [21] [22] [23] [24] [25] [26] [27] [28] [29] [30] [31] [32] [33] [34] [35] • Camera ISP Bandwidth Adjustments: [36]
Camera ISP Use Case and Tips • Programming Flow :
Camera ISP Register Manual • Camera ISP_CCDC Register Summary: [40] • Camera ISP_CCDC Register Description: [41] [42] [43] [44] [45]

Table 12-235. CCDC_FMT_HORZ

Address Offset	0x0000 005C		
Physical Address	0x480B C65C	**Instance**	ISP_CCDC
Description	DATA REFORMATTER HORIZ INFO REGISTER		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								FMTSPH								RESERVED								FMTLNH							

Bits	Field Name	Description	Type	Reset
31:29	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x0
28:16	FMTSPH	Start pixel horizontal from start of the HS sync pulse.	RW	0x0000
15:13	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x0
12:0	FMTLNH	Number of pixels in horizontal direction to use for the data reformatter (minimum is 2 pixels).	RW	0x0000

Table 12-236. Register Call Summary for Register CCDC_FMT_HORZ

Camera ISP Functional Description • CCDC: [0]
Camera ISP Basic Programming Model • CCDC Hardware Setup/Initialization: [1] • CCDC Operations: [6]
Camera ISP Register Manual • Camera ISP_CCDC Register Summary: [7]

Table 12-237. CCDC_FMT_VERT

Address Offset	0x0000 0060		
Physical Address	0x480B C660	**Instance**	ISP_CCDC
Description	DATA REFORMATTER VERT INFO REGISTER		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								FMTSLV								RESERVED								FMTLNV							

Bits	Field Name	Description	Type	Reset
31:29	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x0
28:16	FMTSLV	Start line from start of VS sync pulse for the data reformatter.	RW	0x0000
15:13	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x0
12:0	FMTLNV	Number of lines in vertical direction for the data reformatter	RW	0x0000

Table 12-238. Register Call Summary for Register CCDC_FMT_VERT

Camera ISP Functional Description

- CCDC: [0]

Camera ISP Basic Programming Model

- CCDC Hardware Setup/Initialization: [1] [2]
- CCDC Operations: [7]
- Summary of Constraints: [8]

Camera ISP Register Manual

- Camera ISP_CCDC Register Summary: [9]
- Camera ISP_CCDC Register Description: [10]

Table 12-239. CCDC_FMT_ADDR_i

Address Offset	0x0000 0064 + (i * 0x4)	**Index**	i = 0 to 7
Physical Address	0x480B C664 + (i *0x4)	**Instance**	ISP_CCDC
Description	DATA REFORMATTER ADDR PTR x SETUP REGISTER		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED						LINE		RESERVED																INIT							

Bits	Field Name	Description	Type	Reset
31:26	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x00
25:24	LINE	The output line the address belongs to is the 1st, 2nd, 3rd or 4th line. 0x0: 1st line 0x1: 2nd line 0x2: 3rd line 0x3: 4th line	RW	0x0
23:13	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x000

Bits	Field Name	Description	Type	Reset
12:0	INIT	Initial address value If CCDC_FMTCFG.FMTCFG.LNUM = 0 (1 line), the max value of the address must not exceed (4x1376 - 1) including the updates on it during processing. If CCDC_FMTCFG. FMTCFG.LNUM = 1 (2 lines), the max value of the address must not exceed (3x1376 - 1) including the updates on it during processing. If CCDC_FMTCFG.LNUM = 2 (3 lines), the max value of the address must not exceed (2x1376 - 1) including the updates on it during processing. If CCDC_FMTCFG.LNUM = 3 (4 lines), the max value of the address must not exceed (1x1376 - 1) including the updates on it during processing. The address must always be a positive number during the updates.	RW	0x0000

Table 12-240. Register Call Summary for Register CCDC_FMT_ADDR_i

Camera ISP Functional Description • CCDC: [0] [1]
Camera ISP Basic Programming Model • CCDC Hardware Setup/Initialization: [2] • CCDC Operations: [3] [4] [5] [6] [7] [8] [9] [10] [11] [12] [13] [14] [15] [16] [17] [18] [19] [20] [21]
Camera ISP Register Manual • Camera ISP_CCDC Register Summary: [22]

Table 12-241. CCDC_PRGEVEN0

Address Offset	0x0000 0084		
Physical Address	0x480B C684	**Instance**	ISP_CCDC
Description	PROGRAM ENTRIES 0-7 FOR EVEN LINES REGISTER Each bit field in this register is programmed in the same way. The following definition applies, where n ranges from 0 to 8. Bits [4*n+3:4*n+1] 000: ADDR0 001: ADDR1 010: ADDR2 011: ADDR3 100: ADDR4 101: ADDR5 110: ADDR6 111: ADDR7 Bit [4*n]: 0: Auto increment 1: Auto decrement		
Type	RW		

31 30 29 28	27 26 25 24	23 22 21 20	19 18 17 16	15 14 13 12	11 10 9 8	7 6 5 4	3 2 1 0
EVEN7	EVEN6	EVEN5	EVEN4	EVEN3	EVEN2	EVEN1	EVEN0

Bits	Field Name	Description	Type	Reset
31:28	EVEN7	Address update. See register description.	RW	0x0
27:24	EVEN6	Address update. See register description.	RW	0x0
23:20	EVEN5	Address update. See register description.	RW	0x0
19:16	EVEN4	Address update. See register description.	RW	0x0
15:12	EVEN3	Address update. See register description.	RW	0x0
11:8	EVEN2	Address update. See register description.	RW	0x0
7:4	EVEN1	Address update. See register description.	RW	0x0
3:0	EVEN0	Address update. See register description.	RW	0x0

Table 12-242. Register Call Summary for Register CCDC_PRGEVEN0

Camera ISP Functional Description • CCDC: [0]
Camera ISP Basic Programming Model • CCDC Hardware Setup/Initialization: [1] • CCDC Operations: [2] [3] [4] [5] [6] [7] [8] [9] [10] [11]

Table 12-242. Register Call Summary for Register CCDC_PRGEVEN0 (continued)

Camera ISP Register Manual • Camera ISP_CCDC Register Summary: [12]

Table 12-243. CCDC_PRGEVEN1

Address Offset	0x0000 0088		
Physical Address	0x480B C688	**Instance**	ISP_CCDC
Description	PROGRAM ENTRIES 8-15 FOR EVEN LINES REGISTER Each bit field in this register is programmed in the same way. The following definition applies, where n ranges from 0 to 8. Bits [4*n+3:4*n+1] 000: ADDR0 001: ADDR1 010: ADDR2 011: ADDR3 100: ADDR4 101: ADDR5 110: ADDR6 111: ADDR7 Bit [4*n]: 0: Auto increment 1: Auto decrement		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EVEN15				EVEN14				EVEN13				EVEN12				EVEN11				EVEN10				EVEN9				EVEN8			

Bits	Field Name	Description	Type	Reset
31:28	EVEN15	Address update. See register description.	RW	0x0
27:24	EVEN14	Address update. See register description.	RW	0x0
23:20	EVEN13	Address update. See register description.	RW	0x0
19:16	EVEN12	Address update. See register description.	RW	0x0
15:12	EVEN11	Address update. See register description.	RW	0x0
11:8	EVEN10	Address update. See register description.	RW	0x0
7:4	EVEN9	Address update. See register description.	RW	0x0
3:0	EVEN8	Address update. See register description.	RW	0x0

Table 12-244. Register Call Summary for Register CCDC_PRGEVEN1

Camera ISP Functional Description • CCDC: [0]
Camera ISP Basic Programming Model • CCDC Hardware Setup/Initialization: [1] • CCDC Operations: [2]
Camera ISP Register Manual • Camera ISP_CCDC Register Summary: [3]

Table 12-245. CCDC_PRGODD0

Address Offset	0x0000 008C		
Physical Address	0x480B C68C	**Instance**	ISP_CCDC
Description	PROGRAM ENTRIES 0-7 FOR ODD LINES REGISTER Each bit field in this register is programmed in the same way. The following definition applies, where n ranges from 0 to 8. Bits [4*n+3:4*n+1] 000: ADDR0 001: ADDR1 010: ADDR2 011: ADDR3 100: ADDR4 101: ADDR5 110: ADDR6 111: ADDR7 Bit [4*n]: 0: Auto increment 1: Auto decrement		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ODD7				ODD6				ODD5				ODD4				ODD3				ODD2				ODD1				ODD0			

Bits	Field Name	Description	Type	Reset
31:28	ODD7	Address update. See register description.	RW	0x0
27:24	ODD6	Address update. See register description.	RW	0x0
23:20	ODD5	Address update. See register description.	RW	0x0
19:16	ODD4	Address update. See register description.	RW	0x0

Bits	Field Name	Description	Type	Reset
15:12	ODD3	Address update. See register description.	RW	0x0
11:8	ODD2	Address update. See register description.	RW	0x0
7:4	ODD1	Address update. See register description.	RW	0x0
3:0	ODD0	Address update. See register description.	RW	0x0

Table 12-246. Register Call Summary for Register CCDC_PRGODD0

Camera ISP Functional Description • CCDC: [0]
Camera ISP Basic Programming Model • CCDC Hardware Setup/Initialization: [1] • CCDC Operations: [2] [3] [4] [5] [6] [7] [8] [9] [10] [11]
Camera ISP Register Manual • Camera ISP_CCDC Register Summary: [12]

Table 12-247. CCDC_PRGODD1

Address Offset	0x0000 0090		
Physical Address	0x480B C690	**Instance**	ISP_CCDC
Description	PROGRAM ENTRIES 8-15 FOR ODD LINES REGISTER Each bit field in this register is programmed in the same way. The following definition applies, where n ranges from 0 to 8. Bits [4*n+3:4*n+1] 000: ADDR0 001: ADDR1 010: ADDR2 011: ADDR3 100: ADDR4 101: ADDR5 110: ADDR6 111: ADDR7 Bit [4*n]: 0: Auto increment 1: Auto decrement		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ODD15				ODD14				ODD13				ODD12				ODD11				ODD10				ODD9				ODD8			

Bits	Field Name	Description	Type	Reset
31:28	ODD15	Address update. See register description.	RW	0x0
27:24	ODD14	Address update. See register description.	RW	0x0
23:20	ODD13	Address update. See register description.	RW	0x0
19:16	ODD12	Address update. See register description.	RW	0x0
15:12	ODD11	Address update. See register description.	RW	0x0
11:8	ODD10	Address update. See register description.	RW	0x0
7:4	ODD9	Address update. See register description.	RW	0x0
3:0	ODD8	Address update. See register description.	RW	0x0

Table 12-248. Register Call Summary for Register CCDC_PRGODD1

Camera ISP Functional Description • CCDC: [0]
Camera ISP Basic Programming Model • CCDC Hardware Setup/Initialization: [1] • CCDC Operations: [2]
Camera ISP Register Manual • Camera ISP_CCDC Register Summary: [3]

Table 12-249. CCDC_VP_OUT

Address Offset	0x0000 0094		
Physical Address	0x480B C694	**Instance**	ISP_CCDC
Description	VIDEO PORT OUTPUT REGISTER		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								VERT_NUM																HORZ_NUM							
																												HORZ_ST			

Bits	Field Name	Description	Type	Reset
31	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x0
30:17	VERT_NUM	Number of vertical lines to clock out the video. If the data reformatter is turned off, then the number of lines that can be clocked out of the video port must be at least 1 line less than the number of lines input from the sensor. If the data reformatter is turned on, then the number of lines that can be clocked out of the video port must be less than (CCDC_FMT_VERT.FMTLNV - 1) * (CCDC_FMTCFG.LNUM + 1) The video port output VS pulse is generated right from the first video port input VS itself.	RW	0x0000
16:4	HORZ_NUM	Number of horizontal pixel to clock out the video port. The minimum value allowed is 2 pixels. The maximum offset allowed is 1376 if original input is broken down to 4 lines. The maximum offset allowed is 1376 if original input is broken down to 3 lines. The maximum offset allowed is 2*1376 if original input is broken down to 2 lines. The maximum offset allowed is 4*1376 if original input is broken down to 1 line.	RW	0x0000
3:0	HORZ_ST	Horizontal start pixel in each output line. The maximum value allowed is 15. The video port output HSYNC pulse is generated from this position on for each line. To be able to select an offset higher than 15, the input settings to the data reformatter must be configured appropriately. The purpose of this parameter is to allow for sensors that can read out a parallelogram image rather than a rectangular image.	RW	0x0

Table 12-250. Register Call Summary for Register CCDC_VP_OUT

Camera ISP Functional Description
- CCDC:

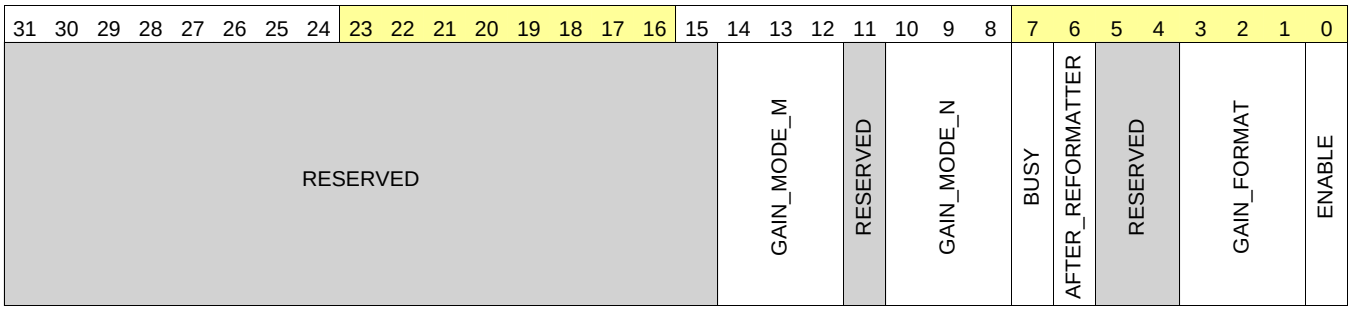
Camera ISP Basic Programming Model
- CCDC Hardware Setup/Initialization: [1] [2]
- CCDC Operations: [6] [7] [8] [9] [10] [11] [12]
- Summary of Constraints: [13]

Camera ISP Use Case and Tips
- Programming Flow :

Camera ISP Register Manual
- Camera ISP_CCDC Register Summary: [17]

Table 12-251. CCDC_LSC_CONFIG

Address Offset	0x0000 0098		
Physical Address	0x480B C698	**Instance**	ISP_CCDC
Description	LENS SHADING COMPENSATION CONTROL AND STATUS REGISTER		
Type	RW		



Bits	Field Name	Description	Type	Reset
31:15	RESERVED	Write 0s for future compatibility. Read returns 0.	RW	0x00000
14:12	GAIN_MODE_M	Define the horizontal dimension of a paxel. Possible values are listed below	RW	0x6
		0x2: Paxel is 4 pixels tall (M=4)		
		0x3: Paxel is 8 pixels tall (M=8)		
		0x4: Paxel is 16 pixels tall (M=16)		
		0x5: Paxel is 32 pixels tall (M=32)		
		0x6: Paxel is 64 pixels tall (M=64)		
11	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x0
10:8	GAIN_MODE_N	Define the vertical dimension of a paxel. Possible values are listed below	RW	0x6
		0x2: Paxel is 4 pixels tall (N=4)		
		0x3: Paxel is 8 pixels tall (N=8)		
		0x4: Paxel is 16 pixels tall (N=16)		
		0x5: Paxel is 32 pixels tall (N=32)		
		0x6: Paxel is 64 pixels tall (N=64)		
7	BUSY	Module busy or idle	R	0x0
		0x0: The module is idle		
		0x1: The module is busy		
6	AFTER_REFORMATTER	Chooses if lens-shading compensation is done before or after data reformatting	RW	0x0
		0x0: Lens-shading is done before data reformatting. H3A, HIST and PREVIEW receives shading compensated data		
		0x1: Data received by H3A isn't compensated. Data received by PREVIEW and HIST is compensated.		
5:4	RESERVED	Write 0s for future compatibility. Reads return zero.	RW	0x0

Bits	Field Name	Description	Type	Reset
3:1	GAIN_FORMAT	Sets gain table format 0x0: Coded as 8-bit fraction Range from 0 to 255/256 0x1: Coded as 8-bit fraction + 1.0 of base. Range from 1 to 1+255/256 0x2: Coded as 1-bit integer, 7-bit fraction. Range from 0 to 1+127/128 0x3: Coded as 1-bit integer, 7-bit fraction + 1.0 Range from 1 to 2+127/128 0x4: Coded as 2-bit integer, 6-bit fraction Range from 0 to 3+63/64 0x5: Coded as 2-bit integer, 6-bit fraction + 1.0 Range from 1 to 4+63/64 0x6: Coded as 3-bit integer, 5-bit fraction Range from 0 to 7+31/32 0x7: Coded as 3-bit integer, 5-bit fraction + 1.0 Range from 1 to 8+31/32	RW	0x0
0	ENABLE	Enables/disables LSC 0x0: Disables the module at the end of the current frame. Video data is transmitted without modification when LSC is disabled. The BUSY bit can be used to poll when access to SBL buffer can be used by the preview module. 0x1: Enables the module. Module starts fetching the gain table as soon it is started. Firmware has to make sure it has been correctly initialized before starting the module. Data processing is only effective at the start of the next frame. Note that preview module dark frame subtract must be disabled because there is only one shared read port.	RW	0x0

Table 12-252. Register Call Summary for Register CCDC_LSC_CONFIG

Camera ISP Functional Description
- CCDC:

Camera ISP Basic Programming Model
- CCDC Hardware Setup/Initialization: [2] [3] [4]
- Register Accessibility During Frame Processing: [5]
- CCDC Operations:

Camera ISP Register Manual
- Camera ISP_CCDC Register Summary: [23]

Table 12-253. CCDC_LSC_INITIAL

Address Offset	0x0000 009C		
Physical Address	0x480B C69C	**Instance**	ISP_CCDC
Description	LENS SHADING COMPENSATION INITIAL X/Y REGISTER		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								Y								RESERVED								X							

Bits	Field Name	Description	Type	Reset
31:22	RESERVED	Write 0s for future compatibility. Reads return zero.	RW	0x000
21:16	Y	Y position, in pixels, of the first active pixel in reference to the first active paxel. Must be an even number.	RW	0x00
15:6	RESERVED	Write 0s for future compatibility. Reads return zero.	RW	0x000

Bits	Field Name	Description	Type	Reset
5:0	X	X position, in pixels, of the first active pixel in reference to the first active paxel. Must be an even number.	RW	0x00

Table 12-254. Register Call Summary for Register CCDC_LSC_INITIAL

Camera ISP Functional Description
- CCDC:

Camera ISP Basic Programming Model
- Register Accessibility During Frame Processing: [10]
- CCDC Operations:

Camera ISP Register Manual
- Camera ISP_CCDC Register Summary: [15]

Table 12-255. CCDC_LSC_TABLE_BASE

Address Offset	0x0000 00A0		
Physical Address	0x480B C6A0	**Instance**	ISP_CCDC
Description	LENS SHADING COMPENSATION TABLE BASE ADDRESS REGISTER		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BASE																															

Bits	Field Name	Description	Type	Reset
31:0	BASE	Table address in bytes. Table is 32-bit aligned so this register must be a multiple of 4. This bit field sets the address of the gain table in memory.	RW	0x00000000

Table 12-256. Register Call Summary for Register CCDC_LSC_TABLE_BASE

Camera ISP Functional Description
- CCDC:

Camera ISP Basic Programming Model
- CCDC Hardware Setup/Initialization: [3]
- Register Accessibility During Frame Processing: [4]
- CCDC Operations:

Camera ISP Register Manual
- Camera ISP_CCDC Register Summary: [6]

Table 12-257. CCDC_LSC_TABLE_OFFSET

Address Offset	0x0000 00A4		
Physical Address	0x480B C6A4	**Instance**	ISP_CCDC
Description	LENS SHADING COMPENSATION TABLE OFFSET REGISTER		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																OFFSET															

Bits	Field Name	Description	Type	Reset
31:16	RESERVED	Write 0s for future compatibility. Read returns 0.	RW	0x0000
15:0	OFFSET	Defines the length, in bytes, of one row of the gain table. Gain table is 32-bit aligned, so this value must be a multiple of 4. Note that the row in memory must be longer or equal to what LSC uses.	RW	0x0000

Table 12-258. Register Call Summary for Register CCDC_LSC_TABLE_OFFSET

Camera ISP Functional Description • CCDC:
Camera ISP Basic Programming Model • CCDC Hardware Setup/Initialization: [4] • Register Accessibility During Frame Processing: [5] • CCDC Operations:
Camera ISP Register Manual • Camera ISP_CCDC Register Summary: [7]

12.6.6 Camera ISP_HIST Registers

12.6.6.1 Camera ISP_HIST Register Summary

Table 12-259. ISP_HIST Register Summary

Register Name	Type	Register Width (Bits)	Address Offset	Physical Address
HIST_PID	R	32	0x0000 0000	0x480B CA00
HIST_PCR	RW	32	0x0000 0004	0x480B CA04
HIST_CNT	RW	32	0x0000 0008	0x480B CA08
HIST_WB_GAIN	RW	32	0x0000 000C	0x480B CA0C
HIST_Rn_HORZ(1)	RW	32	0x0000 0010 + (n*0x8)	0x480B CA10 + (n*0x8)
HIST_Rn_VERT(1)	RW	32	0x0000 0014 + (n*0x8)	0x480B CA14 + (n*0x8)
HIST_ADDR	RW	32	0x0000 0030	0x480B CA30
HIST_DATA	RW	32	0x0000 0034	0x480B CA34
HIST_RADD	RW	32	0x0000 0038	0x480B CA38
HIST_RADD_OFF	RW	32	0x0000 003C	0x480B CA3C
HIST_H_V_INFO	RW	32	0x0000 0040	0x480B CA40

(1) n = 0 to 3

12.6.6.2 Camera ISP_HIST Register Description

Table 12-260. HIST_PID

Address Offset	0x0000 0000		
Physical Address	0x480B CA00	**Instance**	ISP_HIST
Description	PERIPHERAL ID REGISTER		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								TID								CID								PREV							

Bits	Field Name	Description	Type	Reset
31:24	RESERVED	Write 0s for future compatibility. Reads return 0.	R	0x00
23:16	TID	Peripheral identification: HIST MODULE	R	0x08
15:8	CID	Class identification: Camera ISP	R	0xFE
7:0	PREV	Peripheral revision number	R	TI internal data

Table 12-261. Register Call Summary for Register HIST_PID

Camera ISP Register Manual

- Camera ISP_HIST Register Summary: [0]

Table 12-262. HIST_PCR

Address Offset	0x0000 0004		
Physical Address	0x480B CA04	**Instance**	ISP_HIST
Description	PERIPHERAL CONTROL REGISTER		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																														BUSY	ENABLE

Bits	Field Name	Description	Type	Reset
31:2	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x00000000
1	BUSY	HIST module busy. 0x0: Module is not busy. 0x1: Module is busy.	RW	0x0
0	ENABLE	HIST module enable. 0x0: Disable module 0x1: Enable module	RW	0x0

Table 12-263. Register Call Summary for Register HIST_PCR

Camera ISP Functional Description

- Statistics Collection: Histogram:

Camera ISP Basic Programming Model

- Hardware Setup/Initialization: [2]
- Enable/Disable Hardware: [3]
- Event and Status Checking: [4] [5] [6]
- Register Accessibility During Frame Processing: [7] [8] [9]
- Interframe Operations: [10] [11] [12]

Camera ISP Register Manual

- Camera ISP_HIST Register Summary: [13]

Table 12-264. HIST_CNT

Address Offset	0x0000 0008		
Physical Address	0x480B CA08	**Instance**	ISP_HIST
Description	HISTOGRAM CONTROL REGISTER		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																							DATSIZ	CLR	CFA	BINS		SOURCE	SHIFT		

Bits	Field Name	Description	Type	Reset
31:9	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x000000
8	DATSIZ	Input data width 0x0: The pixels are coded on more than 8 bits. 0x1: The pixels are coded on 8 bits.	RW	0x0
7	CLR	Clear histogram data after read. 0x0: Don't clear the data after read. 0x1: Clear the data after read.	RW	0x0
6	CFA	CFA pattern. 0x0: Bayer pattern. 0x1: Reserved.	RW	0x0
5:4	BINS	Number of bins. 0x0: 32 bins, REGIONS 0, 1, 2 and 3 are active. 0x1: 64 bins, REGIONS 0, 1, 2 and 3 are active. 0x2: 128 bins, REGIONS 0 and 1 are active. 0x3: 256 bins, REGION 0 is active.	RW	0x0
3	SOURCE	Input source. 0x0: The input data comes from the CCDC module. 0x1: The input data comes from memory.	RW	0x0
2:0	SHIFT	Shift value. The pixel data is right shifted before the binning operation. The shift value can vary from 0 to 7.	RW	0x0

Table 12-265. Register Call Summary for Register HIST_CNT

Camera ISP Functional Description
- Statistics Collection: Histogram: [0] [1] [2] [3]

Camera ISP Basic Programming Model
- Hardware Setup/Initialization: [7] [8] [9] [10] [11] [12] [13] [14] [15] [16]

Camera ISP Register Manual
- Camera ISP_HIST Register Summary: [17]

Table 12-266. HIST_WB_GAIN

Address Offset	0x0000 000C		
Physical Address	0x480B CA0C	**Instance**	ISP_HIST
Description	HISTOGRAM WHITE BALANCE GAIN REGISTER		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
WG00								WG01								WG02								WG03							

Bits	Field Name	Description	Type	Reset
31:24	WG00	White balance gain 00. The gain value is unsigned and in 3Q5 representation. It varies from 0 to 7.96875.	RW	0x20
23:16	WG01	White balance gain 01. The gain value is unsigned and in 3Q5 representation. It varies from 0 to 7.96875.	RW	0x20
15:8	WG02	White balance gain 02. The gain value is unsigned and in 3Q5 representation. It varies from 0 to 7.96875.	RW	0x20
7:0	WG03	White balance gain 03. The gain value is unsigned and in 3Q5 representation. It varies from 0 to 7.96875.	RW	0x20

Table 12-267. Register Call Summary for Register HIST_WB_GAIN

Camera ISP Functional Description
- Statistics Collection: Histogram: [0] [2]

Camera ISP Basic Programming Model
- Hardware Setup/Initialization: [3]

Camera ISP Register Manual
- Camera ISP_HIST Register Summary: [4]

Table 12-268. HIST_Rn_HORZ

Address Offset	0x0000 0010 + (n*0x8)	**Index**	n = 0 to 3
Physical Address	0x480B CA10 + (n*0x8)	**Instance**	ISP_HIST
Description	REGION n HORIZONTAL REGISTER		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								HSTART								RESERVED								HEND							

Bits	Field Name	Description	Type	Reset
31:30	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x0
29:16	HSTART	Horizontal start position for REGION n. From 0 to 16383.	RW	0x0000
15:14	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x0
13:0	HEND	Horizontal end position for REGION n. From 0 to 16383.	RW	0x0000

Table 12-269. Register Call Summary for Register HIST_Rn_HORZ

Camera ISP Functional Description • Statistics Collection: Histogram: [0] [1]
Camera ISP Basic Programming Model • Hardware Setup/Initialization: [2] [3] [4] [5]
Camera ISP Register Manual • Camera ISP_HIST Register Summary: [6]

Table 12-270. HIST_Rn_VERT

Address Offset	0x0000 0014 + (n*0x8)	**Index**	n = 0 to 3
Physical Address	0x480B CA14 + (n*0x8)	**Instance**	ISP_HIST
Description	REGION n VERTICAL REGISTER		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								VSTART								RESERVED								VEND							

Bits	Field Name	Description	Type	Reset
31:30	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x0
29:16	VSTART	Vertical start position for REGION n. From 0 to 16383.	RW	0x0000
15:14	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x0
13:0	VEND	Vertical end position for REGION n. From 0 to 16383.	RW	0x0000

Table 12-271. Register Call Summary for Register HIST_Rn_VERT

Camera ISP Functional Description • Statistics Collection: Histogram: [0] [1]
Camera ISP Basic Programming Model • Hardware Setup/Initialization: [2] [3] [4] [5]
Camera ISP Register Manual • Camera ISP_HIST Register Summary: [6]

Table 12-272. HIST_ADDR

Address Offset	0x0000 0030		
Physical Address	0x480B CA30	**Instance**	ISP_HIST
Description	HISTOGRAM ADDRESS REGISTER		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																								ADDR							

Bits	Field Name	Description	Type	Reset
31:10	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x000000
9:0	ADDR	Histogram memory address. The histogram memory has 1024 entries. Each entry is coded on 20 bits.	RW	0x000

Table 12-273. Register Call Summary for Register HIST_ADDR

Camera ISP Functional Description
- Statistics Collection: Histogram:

Camera ISP Register Manual
- Camera ISP_HIST Register Summary: [4]

Table 12-274. HIST_DATA

Address Offset	0x0000 0034		
Physical Address	0x480B CA34	**Instance**	ISP_HIST
Description			
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								RDATA																							

Bits	Field Name	Description	Type	Reset
31:20	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x000
19:0	RDATA	Histogram data. The histogram memory has 1024 entries. Each entry is coded on 20 bits.	RW	0x-----

Table 12-275. Register Call Summary for Register HIST_DATA

Camera ISP Functional Description
- Statistics Collection: Histogram:

Camera ISP Basic Programming Model
- Register Accessibility During Frame Processing: [3]

Camera ISP Register Manual
- Camera ISP_HIST Register Summary: [4]

Table 12-276. HIST_RADD

Address Offset	0x0000 0038		
Physical Address	0x480B CA38	**Instance**	ISP_HIST
Description	ADDRESS REGISTER This register is used only if the HIST module input data comes from memory instead of the CCDC module.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RADD																															

Bits	Field Name	Description	Type	Reset
31:0	RADD	32-bit address. The 5 LSBs are ignored: the starting address should be aligned on a 32-byte boundary.	RW	0x00000000

Table 12-277. Register Call Summary for Register HIST_RADD

Camera ISP Functional Description
- Statistics Collection: Histogram: [0]

Camera ISP Basic Programming Model
- Hardware Setup/Initialization: [1]
- Register Accessibility During Frame Processing: [2]

Camera ISP Register Manual
- Camera ISP_HIST Register Summary: [3]

Table 12-278. HIST_RADD_OFF

Address Offset	0x0000 003C		
Physical Address	0x480B CA3C	**Instance**	ISP_HIST
Description	ADDRESS OFFSET REGISTER This register is used only if the HIST module input data comes from memory instead of the CCDC module.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																OFFSET															

Bits	Field Name	Description	Type	Reset
31:16	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x0000
15:0	OFFSET	Offset value. The 5 LSBs are ignored: the offset must be a multiple of 32-bytes.	RW	0x0000

Table 12-279. Register Call Summary for Register HIST_RADD_OFF

Camera ISP Functional Description
- Statistics Collection: Histogram: [0]

Camera ISP Basic Programming Model
- Hardware Setup/Initialization: [1]
- Register Accessibility During Frame Processing: [2]

Camera ISP Register Manual
- Camera ISP_HIST Register Summary: [3]

Table 12-280. HIST_H_V_INFO

Address Offset	0x0000 0040		
Physical Address	0x480B CA40	**Instance**	ISP_HIST
Description	IMAGE SIZE REGISTER This register is used only if the HIST module input data comes from memory instead of the CCDC module.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								HSIZE								RESERVED								VSIZE							

Bits	Field Name	Description	Type	Reset
31:30	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x0
29:16	HSIZE	Horizontal size	RW	0x0000
15:14	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x0
13:0	VSIZE	Vertical size	RW	0x0000

Table 12-281. Register Call Summary for Register HIST_H_V_INFO

Camera ISP Functional Description
- Statistics Collection: Histogram: [0] [1]

Camera ISP Basic Programming Model
- Hardware Setup/Initialization: [2]

Table 12-281. Register Call Summary for Register HIST_H_V_INFO (continued)

Camera ISP Register Manual • Camera ISP_HIST Register Summary: [3]

12.6.7 Camera ISP_H3A Registers

12.6.7.1 Camera ISP_H3A Register Summary

Table 12-282. ISP_H3A Register Summary

Register Name	Type	Register Width (Bits)	Address Offset	Physical Address
H3A_PID	R	32	0x0000 0000	0x480B CC00
H3A_PCR	RW	32	0x0000 0004	0x480B CC04
H3A_AFPAX1	RW	32	0x0000 0008	0x480B CC08
H3A_AFPAX2	RW	32	0x0000 000C	0x480B CC0C
H3A_AFPAXSTART	RW	32	0x0000 0010	0x480B CC10
H3A_AFIIRSH	RW	32	0x0000 0014	0x480B CC14
H3A_AFBUFST	RW	32	0x0000 0018	0x480B CC18
H3A_AFCOEF010	RW	32	0x0000 001C	0x480B CC1C
H3A_AFCOEF032	RW	32	0x0000 0020	0x480B CC20
H3A_AFCOEF054	RW	32	0x0000 0024	0x480B CC24
H3A_AFCOEF076	RW	32	0x0000 0028	0x480B CC28
H3A_AFCOEF098	RW	32	0x0000 002C	0x480B CC2C
H3A_AFCOEF0010	RW	32	0x0000 0030	0x480B CC30
H3A_AFCOEF110	RW	32	0x0000 0034	0x480B CC34
H3A_AFCOEF132	RW	32	0x0000 0038	0x480B CC38
H3A_AFCOEF154	RW	32	0x0000 003C	0x480B CC3C
H3A_AFCOEF176	RW	32	0x0000 0040	0x480B CC40
H3A_AFCOEF198	RW	32	0x0000 0044	0x480B CC44
H3A_AFCOEF1010	RW	32	0x0000 0048	0x480B CC48
H3A_AEWWIN1	RW	32	0x0000 004C	0x480B CC4C
H3A_AEWINSTART	RW	32	0x0000 0050	0x480B CC50
H3A_AEWINBLK	RW	32	0x0000 0054	0x480B CC54
H3A_AEWSUBWIN	RW	32	0x0000 0058	0x480B CC58
H3A_AEWBUFST	RW	32	0x0000 005C	0x480B CC5C

12.6.7.2 Camera ISP_H3A Register Description

Table 12-283. H3A_PID

Address Offset	0x0000 0000		
Physical Address	0x480B CC00	**Instance**	ISP_H3A
Description	PERIPHERAL ID REGISTER		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								TID								CID								PREV							

Bits	Field Name	Description	Type	Reset
31:24	RESERVED	Write 0s for future compatibility. Reads return 0.	R	0x00
23:16	TID	Peripheral identification: H3A module.	R	0x08
15:8	CID	Class identification: Camera ISP	R	0xFE
7:0	PREV	Peripheral revision number	R	TI internal data

Table 12-284. Register Call Summary for Register H3A_PID

Camera ISP Register Manual

- Camera ISP_H3A Register Summary: [0]

Table 12-285. H3A_PCR

Address Offset	0x0000 0004		
Physical Address	0x480B CC04	**Instance**	ISP_H3A
Description	PERIPHERAL CONTROL REGISTER		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
AVE2LMT										RESERVED			BUSYAEAWB	AEW_ALAW_EN	AEW_EN	BUSYAF	FVMODE	RGBPOS				MED_TH							AF_MED_EN	AF_ALAW_EN	AF_EN

Bits	Field Name	Description	Type	Reset
31:22	AVE2LMT	AE AWB saturation limit. All pixels in a block are compared to this limit. If one pixel is above the limit, the block is considered saturated.	RW	0x3FF
21:19	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x0
18	BUSYAEAWB	AE AWB busy 0x0: AE AWB not busy 0x1: AE AWB busy	R	0x0
17	AEW_ALAW_EN	AE AWB A-Law enable 0x0: Disable AE AWB A-Law table. 0x1: Disable AE AWB A-Law table.	RW	0x0
16	AEW_EN	AE AWB enable 0x0: Disable AE AWB. 0x1: Enable AE AWB.	RW	0x0
15	BUSYAF	AF busy 0x0: AF not busy. 0x1: AF busy.	R	0x0

Bits	Field Name	Description	Type	Reset
14	FVMODE	Focus value accumulation mode 0x0: Sum mode. 0x1: Peak mode.	RW	0x0
13:11	RGBPOS	RGB pixel position in the AF windows 0x0: GR and GB as Bayer pattern. 0x1: RG and GB as Bayer pattern. 0x2: GR and BG as Bayer pattern. 0x3: RG and BG as Bayer pattern. 0x4: GG and RB as Bayer pattern. 0x5: RB and GG as Bayer pattern.	RW	0x0
10:3	MED_TH	Median filter threshold	RW	0xFF
2	AF_MED_EN	AF median filter enable 0x0: Disable autofocus median filter 0x1: Enable autofocus median filter	RW	0x0
1	AF_ALAW_EN	AF A-Law table enable 0x0: Disable autofocus A-Law table. 0x1: Enable autofocus A-Law table.	RW	0x0
0	AF_EN	AF enable 0x0: Disable autofocus 0x1: Enable autofocus	RW	0x0

Table 12-286. Register Call Summary for Register H3A_PCR

Camera ISP Functional Description
- Statistics Collection: H3A:

Camera ISP Basic Programming Model
- Hardware Setup/Initialization: [7] [8] [9] [10] [11] [12] [13] [14] [15] [16]
- Enable/Disable Hardware: [17] [18] [19] [20]
- Register Accessibility During Frame Processing: [21] [22] [23] [24] [25]
- Interframe Operations: [26]
- Interframe Operations: [27]

Camera ISP Register Manual
- Camera ISP_H3A Register Summary: [28]

Table 12-287. H3A_AFPAX1

Address Offset	0x0000 0008		
Physical Address	0x480B CC08	**Instance**	ISP_H3A
Description	AF PAXEL CONFIGURATION		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								PAXW								RESERVED								PAXH							

Bits	Field Name	Description	Type	Reset
31:23	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x000
22:16	PAXW	Paxel width. The paxel width is set by 2 x (PAXW+1). The paxel-width range varies from 2 to 256. The paxel width must be set to a minimum value of 16 pixels.	RW	0x00
15:7	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x000

Bits	Field Name	Description	Type	Reset
6:0	PAXH	Paxel height. The paxel height is set by 2 x (PAXH+1). The paxel-height range varies from 2 to 256.	RW	0x00

Table 12-288. Register Call Summary for Register H3A_AFPAX1

Camera ISP Functional Description • Statistics Collection: H3A:
Camera ISP Basic Programming Model • Hardware Setup/Initialization: [2]
Camera ISP Register Manual • Camera ISP_H3A Register Summary: [3]

Table 12-289. H3A_AFPAX2

Address Offset	0x0000 000C		
Physical Address	0x480B CC0C	**Instance**	ISP_H3A
Description	AF PAXEL CONFIGURATION		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																AFINCV			PAXVC										PAXHC		

Bits	Field Name	Description	Type	Reset
31:17	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x0000
16:13	AFINCV	AF line increments. The number of lines to skip in a paxel is set by 2 x (AFINCV+1).	RW	0x0
12:6	PAXVC	Paxel count in the vertical direction. The number of paxels in the vertical direction is set by PAXVC+1. It is illegal to have more than 128 paxels in the vertical direction. We have: 0<= PAXVC <= 127.	RW	0x00
5:0	PAXHC	Paxel count in the horizontal direction. The number of paxels in the horizontal direction is set by PAXHC+1. It is illegal to have more than 36 paxels in the horizontal direction. We have: 0<= PAXHC <= 35.	RW	0x00

Table 12-290. Register Call Summary for Register H3A_AFPAX2

Camera ISP Functional Description • Statistics Collection: H3A:
Camera ISP Basic Programming Model • Hardware Setup/Initialization: [3]
Camera ISP Register Manual • Camera ISP_H3A Register Summary: [4]

Table 12-291. H3A_AFPAXSTART

Address Offset	0x0000 0010		
Physical Address	0x480B CC10	**Instance**	ISP_H3A
Description	AF PAXEL START POSITION REGISTER		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								PAXSH								RESERVED								PAXSV							

Bits	Field Name	Description	Type	Reset
31:28	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x0
27:16	PAXSH	AF paxel horizontal start position. Sets the horizontal position for the first pixel. The range is 1 to 4095. PAXSH must be equal to or greater than (H3A_AFIIRSH.AFIIRSH + 1).	RW	0x000
15:12	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x0
11:0	PAXSV	AF paxel vertical start position. Sets the vertical position for the first pixel. The range is 0 to 4095.	RW	0x000

Table 12-292. Register Call Summary for Register H3A_AFPAXSTART

Camera ISP Functional Description • Statistics Collection: H3A:
Camera ISP Basic Programming Model • Hardware Setup/Initialization: [2]
Camera ISP Register Manual • Camera ISP_H3A Register Summary: [3]

Table 12-293. H3A_AFIIRSH

Address Offset	0x0000 0014		
Physical Address	0x480B CC14	**Instance**	ISP_H3A
Description	AF IIR HORIZONTAL START POSITION REGISTER		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																				IIRSH											

Bits	Field Name	Description	Type	Reset
31:12	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x00000
11:0	IIRSH	AF IIR horizontal start position. The range is 0 to 4095. When the horizontal position of a line equals this value the shift registers are cleared on the next pixel.	RW	0x000

Table 12-294. Register Call Summary for Register H3A_AFIIRSH

Camera ISP Functional Description • Statistics Collection: H3A:
Camera ISP Basic Programming Model • Hardware Setup/Initialization: [2]

Table 12-294. Register Call Summary for Register H3A_AFIIRSH (continued)

Camera ISP Register Manual

- Camera ISP_H3A Register Summary: [3]
- Camera ISP_H3A Register Description: [4]

Table 12-295. H3A_AFBUFST

Address Offset	0x0000 0018		
Physical Address	0x480B CC18	**Instance**	ISP_H3A
Description	AF MEMORY ADDRESS		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
AFBUFST																								RESERVED							

Bits	Field Name	Description	Type	Reset
31:5	AFBUFST	Address	RW	0x0000000
4:0	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x00

Table 12-296. Register Call Summary for Register H3A_AFBUFST

Camera ISP Functional Description

- Statistics Collection: H3A:

Camera ISP Basic Programming Model

- Hardware Setup/Initialization: [1]
- Register Accessibility During Frame Processing: [2]

Camera ISP Register Manual

- Camera ISP_H3A Register Summary: [3]

Table 12-297. H3A_AFCOEF010

Address Offset	0x0000 001C		
Physical Address	0x480B CC1C	**Instance**	ISP_H3A
Description	IIR FILTER COEF DATA REGISTER - SET 0		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED				COEFF1												RESERVED				COEFF0											

Bits	Field Name	Description	Type	Reset
31:28	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x0
27:16	COEFF1	AF IIR filter coefficient 1 (set0) The coefficient value is signed in S12Q6 representation. The range is -32 <= coeff <= 31.96875.	RW	0x000
15:12	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x0
11:0	COEFF0	AF IIR filter coefficient 0 (set0) The coefficient value is signed in S12Q6 representation. The range is -32 <= coeff <= 31.96875.	RW	0x000

Table 12-298. Register Call Summary for Register H3A_AFCOEF010

Camera ISP Functional Description
- Statistics Collection: H3A:

Camera ISP Basic Programming Model
- Hardware Setup/Initialization: [1]

Camera ISP Register Manual
- Camera ISP_H3A Register Summary: [2]

Table 12-299. H3A_AFCOEF032

Address Offset	0x0000 0020		
Physical Address	0x480B CC20	**Instance**	ISP_H3A
Description	IIR FILTER COEF DATA REGISTER - SET 0		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								COEFF3								RESERVED								COEFF2							

Bits	Field Name	Description	Type	Reset
31:28	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x0
27:16	COEFF3	AF IIR filter coefficient 3 (set0) The coefficient value is signed in S12Q6 representation. The range is -32 <= coeff <= 31.96875.	RW	0x000
15:12	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x0
11:0	COEFF2	AF IIR filter coefficient 2 (set0) The coefficient value is signed in S12Q6 representation. The range is -32 <= coeff <= 31.96875.	RW	0x000

Table 12-300. Register Call Summary for Register H3A_AFCOEF032

Camera ISP Register Manual
- Camera ISP_H3A Register Summary: [0]

Table 12-301. H3A_AFCOEF054

Address Offset	0x0000 0024		
Physical Address	0x480B CC24	**Instance**	ISP_H3A
Description	IIR FILTER COEF DATA REGISTER - SET 0		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								COEFF5								RESERVED								COEFF4							

Bits	Field Name	Description	Type	Reset
31:28	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x0
27:16	COEFF5	AF IIR filter coefficient 5 (set0) The coefficient value is signed in S12Q6 representation. The range is -32 <= coeff <= 31.96875.	RW	0x000
15:12	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x0
11:0	COEFF4	AF IIR filter coefficient 4 (set0) The coefficient value is signed in S12Q6 representation. The range is -32 <= coeff <= 31.96875.	RW	0x000

Table 12-302. Register Call Summary for Register H3A_AFCOEF054

Camera ISP Register Manual

- Camera ISP_H3A Register Summary: [0]

Table 12-303. H3A_AFCOEF076

Address Offset	0x0000 0028		
Physical Address	0x480B CC28	**Instance**	ISP_H3A
Description	IIR FILTER COEF DATA REGISTER - SET 0		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								COEFF7								RESERVED								COEFF6							

Bits	Field Name	Description	Type	Reset
31:28	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x0
27:16	COEFF7	AF IIR filter coefficient 7 (set0) The coefficient value is signed in S12Q6 representation. The range is -32 <= coeff <= 31.96875.	RW	0x000
15:12	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x0
11:0	COEFF6	AF IIR filter coefficient 6 (set0) The coefficient value is signed in S12Q6 representation. The range is -32 <= coeff <= 31.96875.	RW	0x000

Table 12-304. Register Call Summary for Register H3A_AFCOEF076

Camera ISP Register Manual

- Camera ISP_H3A Register Summary: [0]

Table 12-305. H3A_AFCOEF098

Address Offset	0x0000 002C		
Physical Address	0x480B CC2C	**Instance**	ISP_H3A
Description	IIR FILTER COEF DATA REGISTER - SET 0		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								COEFF9								RESERVED								COEFF8							

Bits	Field Name	Description	Type	Reset
31:28	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x0
27:16	COEFF9	AF IIR filter coefficient 8 (set0) The coefficient value is signed in S12Q6 representation. The range is -32 <= coeff <= 31.96875.	RW	0x000
15:12	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x0
11:0	COEFF8	AF IIR filter coefficient 9 (set0) The coefficient value is signed in S12Q6 representation. The range is -32 <= coeff <= 31.96875.	RW	0x000

Table 12-306. Register Call Summary for Register H3A_AFCOEF098

Camera ISP Register Manual

- Camera ISP_H3A Register Summary: [0]

Table 12-307. H3A_AFCOEF0010

Address Offset	0x0000 0030		
Physical Address	0x480B CC30	**Instance**	ISP_H3A
Description	IIR FILTER COEF DATA REGISTER - SET 0		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																				COEFF10											

Bits	Field Name	Description	Type	Reset
31:12	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x00000
11:0	COEFF10	AF IIR filter coefficient 10 (set0) The coefficient value is signed in S12Q6 representation. The range is -32 <= coeff <= 31.96875.	RW	0x000

Table 12-308. Register Call Summary for Register H3A_AFCOEF0010

Camera ISP Functional Description

- Statistics Collection: H3A:

Camera ISP Register Manual

- Camera ISP_H3A Register Summary: [1]

Table 12-309. H3A_AFCOEF110

Address Offset	0x0000 0034		
Physical Address	0x480B CC34	**Instance**	ISP_H3A
Description	IIR FILTER COEF DATA REGISTER - SET 1		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED				COEFF1												RESERVED				COEFF0											

Bits	Field Name	Description	Type	Reset
31:28	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x0
27:16	COEFF1	AF IIR filter coefficient 1 (set1) The coefficient value is signed in S12Q6 representation. The range is -32 <= coeff <= 31.96875.	RW	0x000
15:12	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x0
11:0	COEFF0	AF IIR filter coefficient 0 (set1) The coefficient value is signed in S12Q6 representation. The range is -32 <= coeff <= 31.96875.	RW	0x000

Table 12-310. Register Call Summary for Register H3A_AFCOEF110

Camera ISP Functional Description

- Statistics Collection: H3A:

Camera ISP Register Manual

- Camera ISP_H3A Register Summary: [1]

Table 12-311. H3A_AFCOEF132

Address Offset	0x0000 0038		
Physical Address	0x480B CC38	**Instance**	ISP_H3A
Description	IIR FILTER COEF DATA REGISTER - SET 1		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								COEFF3								RESERVED								COEFF2							

Bits	Field Name	Description	Type	Reset
31:28	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x0
27:16	COEFF3	AF IIR filter coefficient 3 (set1) The coefficient value is signed in S12Q6 representation. The range is -32 <= coeff <= 31.96875.	RW	0x000
15:12	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x0
11:0	COEFF2	AF IIR filter coefficient 2 (set1) The coefficient value is signed in S12Q6 representation. The range is -32 <= coeff <= 31.96875.	RW	0x000

Table 12-312. Register Call Summary for Register H3A_AFCOEF132

Camera ISP Register Manual

- Camera ISP_H3A Register Summary: [0]

Table 12-313. H3A_AFCOEF154

Address Offset	0x0000 003C		
Physical Address	0x480B CC3C	**Instance**	ISP_H3A
Description	IIR FILTER COEF DATA REGISTER - SET 1		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								COEFF5								RESERVED								COEFF4							

Bits	Field Name	Description	Type	Reset
31:28	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x0
27:16	COEFF5	AF IIR filter coefficient 5 (set1) The coefficient value is signed in S12Q6 representation. The range is -32 <= coeff <= 31.96875.	RW	0x000
15:12	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x0
11:0	COEFF4	AF IIR filter coefficient 4 (set1) The coefficient value is signed in S12Q6 representation. The range is -32 <= coeff <= 31.96875.	RW	0x000

Table 12-314. Register Call Summary for Register H3A_AFCOEF154

Camera ISP Register Manual

- Camera ISP_H3A Register Summary: [0]

Table 12-315. H3A_AFCOEF176

Address Offset	0x0000 0040		
Physical Address	0x480B CC40	**Instance**	ISP_H3A
Description	IIR FILTER COEF DATA REGISTER - SET 1		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								COEFF7								RESERVED								COEFF6							

Bits	Field Name	Description	Type	Reset
31:28	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x0
27:16	COEFF7	AF IIR filter coefficient 7 (set1) The coefficient value is signed in S12Q6 representation. The range is -32 <= coeff <= 31.96875.	RW	0x000
15:12	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x0
11:0	COEFF6	AF IIR filter coefficient 6 (set1) The coefficient value is signed in S12Q6 representation. The range is -32 <= coeff <= 31.96875.	RW	0x000

Table 12-316. Register Call Summary for Register H3A_AFCOEF176

Camera ISP Register Manual

- Camera ISP_H3A Register Summary: [0]

Table 12-317. H3A_AFCOEF198

Address Offset	0x0000 0044		
Physical Address	0x480B CC44	**Instance**	ISP_H3A
Description	IIR FILTER COEF DATA REGISTER - SET 1		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								COEFF9								RESERVED								COEFF8							

Bits	Field Name	Description	Type	Reset
31:28	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x0
27:16	COEFF9	AF IIR filter coefficient 9 (set0) The coefficient value is signed in S12Q6 representation. The range is -32 <= coeff <= 31.96875.	RW	0x000
15:12	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x0
11:0	COEFF8	AF IIR filter coefficient 8 (set0) The coefficient value is signed in S12Q6 representation. The range is -32 <= coeff <= 31.96875.	RW	0x000

Table 12-318. Register Call Summary for Register H3A_AFCOEF198

Camera ISP Register Manual

- Camera ISP_H3A Register Summary: [0]

Table 12-319. H3A_AFCOEF1010

Address Offset	0x0000 0048		
Physical Address	0x480B CC48	**Instance**	ISP_H3A
Description	IIR FILTER COEF DATA REGISTER - SET 1		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																				COEFF10											

Bits	Field Name	Description	Type	Reset
31:12	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x00000
11:0	COEFF10	AF IIR filter coefficient 10 (set1) The coefficient value is signed in S12Q6 representation. The range is -32 <= coeff <= 31.96875.	RW	0x000

Table 12-320. Register Call Summary for Register H3A_AFCOEF1010

Camera ISP Functional Description • Statistics Collection: H3A:
Camera ISP Basic Programming Model • Hardware Setup/Initialization: [1]
Camera ISP Register Manual • Camera ISP_H3A Register Summary: [2]

Table 12-321. H3A_AEWWIN1

Address Offset	0x0000 004C		
Physical Address	0x480B CC4C	**Instance**	ISP_H3A
Description	AE AWB CONTROL REGISTER		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED	WINH							RESERVED	WINW							WINVC								WINHC							

Bits	Field Name	Description	Type	Reset
31	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x0
30:24	WINH	AE AWB window height. The window height is given by 2 x (WINH+1). The final value ranges between 2 to 256 (even values only).	RW	0x00
23:20	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x0
19:13	WINW	AE AWB window width. The window width is given by 2 x (WINW+1). The final value ranges between 2 to 256 (even values only).	RW	0x00
12:6	WINVC	AE AWB vertical window count The number of windows in the vertical direction is set by WINVC + 1. The maximum number of vertical windows in a frame must not exceed 128.	RW	0x00
5:0	WINHC	AE AWB horizontal window count. The number of horizontal windows is set by WINHC + 1. The maximum number of horizontal windows in a frame must not exceed 36.	RW	0x00

Table 12-322. Register Call Summary for Register H3A_AEWWIN1

Camera ISP Functional Description • Statistics Collection: H3A:
Camera ISP Basic Programming Model • Hardware Setup/Initialization: [4]
Camera ISP Register Manual • Camera ISP_H3A Register Summary: [5]

Table 12-323. H3A_AEWINSTART

Address Offset	0x0000 0050		
Physical Address	0x480B CC50	**Instance**	ISP_H3A
Description	AE AWB START POSITION REGISTER		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								WINSV								RESERVED				WINSH											

Bits	Field Name	Description	Type	Reset
31:28	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x0
27:16	WINSV	AE AWB vertical window start position. Sets the first line for the first window. The range is 0 to 4095.	RW	0x000
15:12	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x0
11:0	WINSH	AE AWB horizontal window start position. Sets the horizontal position for the first window on each line. The range is 0 to 4095.	RW	0x000

Table 12-324. Register Call Summary for Register H3A_AEWINSTART

Camera ISP Functional Description • Statistics Collection: H3A:
Camera ISP Basic Programming Model • Hardware Setup/Initialization: [2]
Camera ISP Register Manual • Camera ISP_H3A Register Summary: [3]

Table 12-325. H3A_AEWINBLK

Address Offset	0x0000 0054		
Physical Address	0x480B CC54	**Instance**	ISP_H3A
Description	BLACK LINE REGISTER		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								WINSV								RESERVED								WINH							

Bits	Field Name	Description	Type	Reset
31:28	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x0
27:16	WINSV	AE AWB vertical window start position for the single black line of windows. Sets the first line for the single black line of windows. The range is 0 to 4095. Note that the horizontal start and the horizontal number of windows is similar to the regular windows.	RW	0x000
15:7	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x000
6:0	WINH	AE AWB window height for the single black line of windows. The window height is set by 2 x (WINH + 1). The range is 2 to 256 (even values only).	RW	0x00

Table 12-326. Register Call Summary for Register H3A_AEWINBLK

Camera ISP Functional Description • Statistics Collection: H3A:
Camera ISP Basic Programming Model • Hardware Setup/Initialization: [2]
Camera ISP Register Manual • Camera ISP_H3A Register Summary: [3]

Table 12-327. H3A_AEWSUBWIN

Address Offset	0x0000 0058		
Physical Address	0x480B CC58	**Instance**	ISP_H3A
Description	AE AWB REGISTER		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																				AEWINCV				RESERVED				AEWINCH			

Bits	Field Name	Description	Type	Reset
31:12	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x00000
11:8	AEWINCV	AE AWB vertical sampling point increment. Sets the vertical distance between subsamples. The increment is set by 2 x (AEWINCV+1). The range is 2 to 32 (even values only).	RW	0x0
7:4	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x0
3:0	AEWINCH	AE AWB horizontal sampling point increment. Sets the horizontal distance between subsamples. The increment is set by 2 x (AEWINCH+1). The range is 2 to 32 (even values only).	RW	0x0

Table 12-328. Register Call Summary for Register H3A_AEWSUBWIN

Camera ISP Functional Description • Statistics Collection: H3A:
Camera ISP Basic Programming Model • Hardware Setup/Initialization: [2]
Camera ISP Register Manual • Camera ISP_H3A Register Summary: [3]

Table 12-329. H3A_AEWBUFST

Address Offset	0x0000 005C		
Physical Address	0x480B CC5C	**Instance**	ISP_H3A
Description	AE AWB MEMORY ADDRESS		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
AEWBUFST																								RESERVED							

Bits	Field Name	Description	Type	Reset
31:5	AEWBUFST	AE AWB memory start address The start address in memory where the AE/AWB data are written. This field can be modified even when the AE/AWB submodule is busy. The change takes place only for the next frame. However, register reads always give the latest value.	RW	0x0000000
4:0	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x00

Table 12-330. Register Call Summary for Register H3A_AEWBUFST

Camera ISP Functional Description • Statistics Collection: H3A:
Camera ISP Basic Programming Model • Hardware Setup/Initialization: [1] • Register Accessibility During Frame Processing: [2]
Camera ISP Register Manual • Camera ISP_H3A Register Summary: [3]

12.6.8 Camera ISP_PREVIEW Registers

12.6.8.1 Camera ISP_PREVIEW Register Summary

Table 12-331. ISP_PREVIEW Register Summary

Register Name	Type	Register Width (Bits)	Address Offset	Physical Address
PRV_PID	R	32	0x0000 0000	0x480B CE00
PRV_PCR	RW	32	0x0000 0004	0x480B CE04
PRV_HORZ_INFO	RW	32	0x0000 0008	0x480B CE08
PRV_VERT_INFO	RW	32	0x0000 000C	0x480B CE0C
PRV_RSDR_ADDR	RW	32	0x0000 0010	0x480B CE10
PRV_RADR_OFFSET	RW	32	0x0000 0014	0x480B CE14
PRV_DSDR_ADDR	RW	32	0x0000 0018	0x480B CE18
PRV_DRKF_OFFSET	RW	32	0x0000 001C	0x480B CE1C
PRV_WSDR_ADDR	RW	32	0x0000 0020	0x480B CE20
PRV_WADD_OFFSET	RW	32	0x0000 0024	0x480B CE24
PRV_AVE	RW	32	0x0000 0028	0x480B CE28
PRV_HMED	RW	32	0x0000 002C	0x480B CE2C
PRV_NF	RW	32	0x0000 0030	0x480B CE30
PRV_WB_DGAIN	RW	32	0x0000 0034	0x480B CE34
PRV_WBGAIN	RW	32	0x0000 0038	0x480B CE38

Table 12-331. ISP_PREVIEW Register Summary (continued)

Register Name	Type	Register Width (Bits)	Address Offset	Physical Address
PRV_WBSEL	RW	32	0x0000 003C	0x480B CE3C
PRV_CFA	RW	32	0x0000 0040	0x480B CE40
PRV_BLKADJOFF	RW	32	0x0000 0044	0x480B CE44
PRV_RGB_MAT1	RW	32	0x0000 0048	0x480B CE48
PRV_RGB_MAT2	RW	32	0x0000 004C	0x480B CE4C
PRV_RGB_MAT3	RW	32	0x0000 0050	0x480B CE50
PRV_RGB_MAT4	RW	32	0x0000 0054	0x480B CE54
PRV_RGB_MAT5	RW	32	0x0000 0058	0x480B CE58
PRV_RGB_OFF1	RW	32	0x0000 005C	0x480B CE5C
PRV_RGB_OFF2	RW	32	0x0000 0060	0x480B CE60
PRV_CSC0	RW	32	0x0000 0064	0x480B CE64
PRV_CSC1	RW	32	0x0000 0068	0x480B CE68
PRV_CSC2	RW	32	0x0000 006C	0x480B CE6C
PRV_CSC_OFFSET	RW	32	0x0000 0070	0x480B CE70
PRV_CNT_BRT	RW	32	0x0000 0074	0x480B CE74
PRV_CSUP	RW	32	0x0000 0078	0x480B CE78
PRV_SETUP_YC	RW	32	0x0000 007C	0x480B CE7C
PRV_SET_TBL_ADDR	RW	32	0x0000 0080	0x480B CE80
PRV_SET_TBL_DATA	RW	32	0x0000 0084	0x480B CE84
PRV_CDC_THRx(1)	RW	32	0x0000 0090 + (0x4 * x)	0x480B CE90 + (0x4 * x)

(1) x = 0 to 3

12.6.8.2 Camera ISP_PREVIEW Register Description

Table 12-332. PRV_PID

Address Offset	0x0000 0000		
Physical Address	0x480B CE00	**Instance**	ISP_PREVIEW
Description	PERIPHERAL ID REGISTER		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								TID								CID								PREV							

Bits	Field Name	Description	Type	Reset
31:24	RESERVED	Write 0s for future compatibility. Reads return 0.	R	0x00
23:16	TID	Peripheral identification: PREVIEW module.	R	0x02
15:8	CID	Class identification:	R	0xFE
7:0	PREV	Peripheral revision number.	R	TI internal data

Table 12-333. Register Call Summary for Register PRV_PID

Camera ISP Register Manual

- Camera ISP_PREVIEW Register Summary: [0]

Table 12-334. PRV_PCR

Address Offset	0x0000 0004		
Physical Address	0x480B CE04	**Instance**	ISP_PREVIEW
Description	PERIPHERAL CONTROL REGISTER All the fields in this register can be altered even when the PREVIEW module is busy. Changes take place only for the next frame.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DRK_FAIL	RESERVED		DCOR_METHOD	DCOREN	GAMMA_BYPASS	RESERVED		SCOMP_SFT			SCOMP_EN	SDRPORT	RSZPORT	YCPOS		SUPEN	YNENHEN	CFAFMT				CFAEN	NFEN	HMEDEN	DRKFCAP	DRKFEN	INVALAW	WIDTH	ONESHOT	SOURCE	BUSY	ENABLE

Bits	Field Name	Description	Type	Reset
31	DRK_FAIL	Dark frame subtract fail status. Write 1 to clear this bit. Reset to zero for the next frame. When the error is triggered, dark frame subtract is abandoned for the current frame, dark frame subtract resumes for next frame (but bit is not cleared unless explicitly done by firmware). 0x0: No error 0x1: Error	RW	0x0
30:29	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x0
28	DCOR_METHOD	Defect correction method. 0x0: MinMax 0x1: MinMax2 (Couplet defect correction)	RW	0x0
27	DCOREN	Defect correction enable This bit enables or disables the defect correction. The PRV_PCR.DCOR_METHOD and PRV_CDC_THRx registers must be configured for correct operation. 0x0: Disable defect correction 0x1: Enable defect correction	RW	0x0
26	GAMMA_BYPASS	Bypass, the output is set to the 8 MSB of the 10-bit input. 0x0: No bypass. 0x1: Bypass, the output is set to the 8 MSB of the 10-bit input.	RW	0x0
25	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x0
24:22	SCOMP_SFT	Shading compensation shift value after multiplication. The right-shift range is 0 to 7.	RW	0x0
21	SCOMP_EN	Shading compensation enable instead of dark frame The 8-bit value loaded from memory is multiplied by the current pixel instead of subtracting it. Note that the dark frame subtract (DRKFEN) must be enabled in addition to this bit being active to perform shading compensation. 0x0: Disable. Dark frame subtract can be used. 0x1: Enable. Dark frame subtract cannot be used.	RW	0x0
20	SDRPORT	PREVIEW module memory output port enable. This bit enables or disables the data transfer from the PREVIEW module to the memory. 0x0: Disable 0x1: Enable	RW	0x1

Bits	Field Name	Description	Type	Reset
19	RSZPORT	RESIZER module output port enable. This bit enables or disables the data transfer between the PREVIEW and RESIZER modules. Controls whether or not SDRAM output data is forwarded to the resizer input port. This control bit does not depend on the state of the former SDRPORT bit. Data is simultaneously written to SDRAM (if SDRPORT bit is set) while sending the same data to the resizer as input. Note that the CCDC is also capable or directly writing to the resizer input port. The CCDC setting takes precedence over the preview engine setting. 0x0: Disable 0x1: Enable	RW	0x0
18:17	YCPOS	(CRYCBY) Cr0(31:24) Y1(23:16) Cb0(15:8) Y0(7:0) 0x0: (YCRYCB) Y1(31:24) Cr0(23:16) Y0(15:8) Cb0(7:0) 0x1: (YCBYCR) Y1(31:24) Cb0(23:16) Y0(15:8) Cr0(7:0) 0x2: (CBYCRY) Cb0(31:24) Y1(23:16) Cr(15:8) Y0(7:0) 0x3: (CRYCBY) Cr0(31:24) Y1(23:16) Cb0(15:8) Y0(7:0)	RW	0x0
16	SUPEN	Color suppression. 0x0: Disable 0x1: Enable	RW	0x0
15	YNENHEN	Non-linear enhancer 0x0: Disable 0x1: Enable	RW	0x0
14:11	CFAFMT	CFA format 0x0: Mode 0: conventional Bayer. 0x1: Mode 1: horizontal 2x downsample. 0x2: Mode 2: bypass CFA stage 0x3: Mode 3: horizontal and vertical 2x downsample. 0x4: Mode 4: Super CCD Honeycom movie mode sensor. 0x5: Mode5: bypass CFA stage .	RW	0x0
10	CFAEN	CFA enable 0x0: Disable 0x1: Enable	RW	0x0
9	NFEN	Noise filter enable 0x0: Disable 0x1: Enable	RW	0x0
8	HMEDEN	Horizontal median filter enable. 0x0: Disable 0x1: Enable	RW	0x0
7	DRKFCAP	Dark frame capture enable. 0x0: Normal processing. 0x1: Capture dark frame.	RW	0x0
6	DRKFEN	Subtract dark frame enable. 0x0: Disable 0x1: Enable	RW	0x0
5	INVALAW	Inverse A-Law enable. 0x0: Disable 0x1: Enable	RW	0x0

Bits	Field Name	Description	Type	Reset
4	WIDTH	Input data width selection. 0x0: 10-bit mode 0x1: 8-bit mode	RW	0x0
3	ONESHOT	One-shot mode selection. If this bit is set to 1, it is reset to 0 after the ENABLE bit is asserted. 0x0: Continuous mode (through the video port). 0x1: One shot mode.	RW	0x0
2	SOURCE	Input source selection. If this bit is set to 1, it is reset to 0 after the ENABLE bit is asserted. 0x0: Video port (through the CCDC) 0x1: Memory.	RW	0x0
1	BUSY	Busy bit. 0x0: PREVIEW module not busy. 0x1: PREVIEW module busy.	R	0x0
0	ENABLE	Enable bit. If the ONESHOT or SOURCE bit is 1, this bit is reset to 0 after it is asserted 0x0: PREVIEW module disabled. 0x1: PREVIEW module enabled.	RW	0x0

Table 12-335. Register Call Summary for Register PRV_PCR

Camera ISP Functional Description
- Preview Engine Features: [0] [1] [2] [3] [4] [5] [6] [7] [8] [9] [10] [18] [19] [22] [23] [25] [26] [27] [28] [29] [30]
- Resizer: [31]

Camera ISP Basic Programming Model
- Preview Hardware Setup/Initialization: [32] [33] [34] [35] [36] [37] [38] [39] [40] [41] [42] [43] [44] [45] [46] [47] [48] [49] [50] [51] [52] [53] [54] [55] [56] [57] [58] [59] [60] [61] [62] [63] [64] [65]
- Enable/Disable Hardware: [66] [67]
- Events and Status Checking: [68] [69] [70] [71]
- Register Accessibility During Frame Processing: [72] [73] [74]
- Interframe Operations: [75]
- Event and Status Checking: [76]

Camera ISP Use Case and Tips
- Programming Flow :

Camera ISP Register Manual
- Camera ISP_PREVIEW Register Summary: [87]
- Camera ISP_PREVIEW Register Description: [88] [89]

Table 12-336. PRV_HORZ_INFO

Address Offset	0x0000 0008		
Physical Address	0x480B CE08	**Instance**	ISP_PREVIEW
Description	HORIZONTAL SETUP REGISTER		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								SPH								RESERVED								EPH							

Bits	Field Name	Description	Type	Reset
31:30	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x0
29:16	SPH	Start pixel horizontal. If PCR.SOURCE == 0 (CCDC input) SPH must be >= 2.	RW	0x0000
15:14	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x0
13:0	EPH	End pixel horizontal. The input width of the preview engine must be a multiple of the average count multiplied by the least common multiple of the odd distance and even distance of the AVE register settings. If PRV_PCR.SOURCE == 0 (CCDC input) EPH must be >= 2 pixels before the last pixel sent from the CCDC. PRV_HORZ_INFO.EPH - PRV_HORZ_INFO.SPH + 1) MOD ((1 PRV_AVE.COUNT) * LeastCommonMultiple(PRV_AVE.ODDDIST+1, PRV_AVE.EVENDIST+1)) = 0	RW	0x0000

Table 12-337. Register Call Summary for Register PRV_HORZ_INFO

Camera ISP Functional Description • Preview Engine Features: [0]
Camera ISP Basic Programming Model • Preview Hardware Setup/Initialization: [1] • Summary of Constraints: [2] [3] [4]
Camera ISP Use Case and Tips • Programming Flow :
Camera ISP Register Manual • Camera ISP_PREVIEW Register Summary: [9] • Camera ISP_PREVIEW Register Description: [10] [11]

Table 12-338. PRV_VERT_INFO

Address Offset	0x0000 000C		
Physical Address	0x480B CE0C	**Instance**	ISP_PREVIEW
Description	VERTICAL SETUP REGISTER		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								SLV								RESERVED								ELV							

Bits	Field Name	Description	Type	Reset
31:30	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x0
29:16	SLV	Start line vertical	RW	0x0000
15:14	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x0
13:0	ELV	End line vertical	RW	0x0000

Table 12-339. Register Call Summary for Register PRV_VERT_INFO

Camera ISP Functional Description • Preview Engine Features: [0]

Table 12-339. Register Call Summary for Register PRV_VERT_INFO (continued)

Camera ISP Basic Programming Model • Preview Hardware Setup/Initialization: [1]
Camera ISP Use Case and Tips • Programming Flow :
Camera ISP Register Manual • Camera ISP_PREVIEW Register Summary: [5]

Table 12-340. PRV_RSDR_ADDR

Address Offset	0x0000 0010		
Physical Address	0x480B CE10	**Instance**	ISP_PREVIEW
Description	MEMORY READ ADDRESS REGISTER		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RADR																															

Bits	Field Name	Description	Type	Reset
31:0	RADR	32-bit read address. Specifies the 32-bit address in memory of the input frame. The lower 5 bits of this register are always treated as 0s. The offset should be aligned on a 32-byte boundary. This field can be altered even when the PREVIEW module is busy. The change takes place only for the next frame (next VS pulse). However, note that reading this register always gives the latest value.	RW	0x00000000

Table 12-341. Register Call Summary for Register PRV_RSDR_ADDR

Camera ISP Functional Description • Preview Engine Features: [0]
Camera ISP Basic Programming Model • Preview Hardware Setup/Initialization: [1] • Register Accessibility During Frame Processing: [2]
Camera ISP Register Manual • Camera ISP_PREVIEW Register Summary: [3]

Table 12-342. PRV_RADR_OFFSET

Address Offset	0x0000 0014		
Physical Address	0x480B CE14	**Instance**	ISP_PREVIEW
Description	MEMORY READ ADDRESS OFFSET REGISTER		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																OFFSET															

Bits	Field Name	Description	Type	Reset
31:16	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x0000
15:0	OFFSET	Line offset. Specifies the offset for each line relatively to the previous line. The lower 5 bits of this register are always treated as 0s. The offset should be aligned on a 32-byte boundary. This field can be altered even when the PREVIEW module is busy. The change takes place only for the next frame (next VS sync pulse). However, note that reading this register always gives the latest value.	RW	0x0000

Table 12-343. Register Call Summary for Register PRV_RADR_OFFSET

Camera ISP Functional Description • Preview Engine Features: [0]
Camera ISP Basic Programming Model • Preview Hardware Setup/Initialization: [1] • Register Accessibility During Frame Processing: [2]
Camera ISP Register Manual • Camera ISP_PREVIEW Register Summary: [3]

Table 12-344. PRV_DSDR_ADDR

Address Offset	0x0000 0018		
Physical Address	0x480B CE18	**Instance**	ISP_PREVIEW
Description	DARK FRAME MEMORY ADDRESS REGISTER		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DRKF																															

Bits	Field Name	Description	Type	Reset
31:0	DRKF	Dark frame address. Specifies the dark frame start address in memory. The lower 5 bits of this register are always treated as 0s. The offset should be aligned on a 32-byte boundary. This field can be altered even when the PREVIEW module is busy. The change takes place only for the next frame (next VS sync pulse). However, note that reading this register always gives the latest value.	RW	0x00000000

Table 12-345. Register Call Summary for Register PRV_DSDR_ADDR

Camera ISP Functional Description • Preview Engine Features: [0]
Camera ISP Basic Programming Model • Preview Hardware Setup/Initialization: [1] [2] • Register Accessibility During Frame Processing: [3]
Camera ISP Register Manual • Camera ISP_PREVIEW Register Summary: [4]

Table 12-346. PRV_DRKF_OFFSET

Address Offset	0x0000 001C		
Physical Address	0x480B CE1C	**Instance**	ISP_PREVIEW
Description	DARK FRAME MEMORY OFFSET REGISTER		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																OFFSET															

Bits	Field Name	Description	Type	Reset
31:16	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x0000
15:0	OFFSET	Dark frame line offset. Specifies the offset for each line relatively to the previous line. The lower 5 bits of this register are always treated as 0s. The offset should be aligned on a 32-byte boundary. This field can be altered even when the PREVIEW module is busy. The change takes place only for the next frame (next VS pulse). However, note that reading this register always gives the latest value.	RW	0x0000

Table 12-347. Register Call Summary for Register PRV_DRKF_OFFSET

Camera ISP Functional Description
- Preview Engine Features: [0]

Camera ISP Basic Programming Model
- Preview Hardware Setup/Initialization: [1] [2]
- Register Accessibility During Frame Processing: [3]

Camera ISP Register Manual
- Camera ISP_PREVIEW Register Summary: [4]

Table 12-348. PRV_WSDR_ADDR

Address Offset	0x0000 0020		
Physical Address	0x480B CE20	**Instance**	ISP_PREVIEW
Description	MEMORY WRITE ADDRESS REGISTER		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDR																															

Bits	Field Name	Description	Type	Reset
31:0	ADDR	Write address. Specifies the 32-bit address in memory of the output frame. The lower 5 bits of this register are always treated as 0s. The offset should be aligned on a 32-byte boundary. For optimum performance in the system, the starting address must be on a 256-byte boundary This field can be altered even when the PREVIEW module is busy. The change takes place only for the next frame (next VS pulse). However, note that reading this register always gives the latest value.	RW	0x00000000

Table 12-349. Register Call Summary for Register PRV_WSDR_ADDR

Camera ISP Functional Description
- Preview Engine Features: [0] [1]

Table 12-349. Register Call Summary for Register PRV_WSDR_ADDR (continued)

Camera ISP Basic Programming Model • Preview Hardware Setup/Initialization: [2] • Register Accessibility During Frame Processing: [3]
Camera ISP Use Case and Tips • Programming Flow :
Camera ISP Register Manual • Camera ISP_PREVIEW Register Summary: [5]

Table 12-350. PRV_WADD_OFFSET

Address Offset	0x0000 0024		
Physical Address	0x480B CE24	**Instance**	ISP_PREVIEW
Description	MEMORY WRITE OFFSET REGISTER		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																OFFSET															

Bits	Field Name	Description	Type	Reset
31:16	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x0000
15:0	OFFSET	Line offset. The lower 5 bits of this register are always treated as 0s. The offset should be aligned on a 32-byte boundary. For optimum performance in the system, the starting address must be on a 256-byte boundary This field can be altered even when the PREVIEW module is busy. The change takes place only for the next frame (next VS pulse). However, note that reading this register always gives the latest value.	RW	0x0000

Table 12-351. Register Call Summary for Register PRV_WADD_OFFSET

Camera ISP Functional Description • Preview Engine Features: [0] [1]
Camera ISP Basic Programming Model • Preview Hardware Setup/Initialization: [2] • Register Accessibility During Frame Processing: [3]
Camera ISP Use Case and Tips • Programming Flow :
Camera ISP Register Manual • Camera ISP_PREVIEW Register Summary: [5]

Table 12-352. PRV_AVE

Address Offset	0x0000 0028		
Physical Address	0x480B CE28	**Instance**	ISP_PREVIEW
Description	INPUT FORMATTER REGISTER		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																										ODDDIST		EVENDIST		COUNT	

Bits	Field Name	Description	Type	Reset
31:6	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x0000000
5:4	ODDDIST	Distance between consecutive pixels of the same color in the odd line. 0x0: 1 0x1: 2 0x2: 3 0x3: 4	RW	0x0
3:2	EVENDIST	Distance between consecutive pixels of the same color in the even line. 0x0: 1 0x1: 2 0x2: 3 0x3: 4	RW	0x0
1:0	COUNT	Number of horizontal pixels to average. This field should not be changed when the PCR.DRKFEN bit is set to 1 The input width must be divisible by the number of horizontal pixels to average indicated by this field (. if 4 pixel average is selected, then the input width must be a multiple of 4). 0x0: No averaging. 0x1: 2-pixel average 0x2: 4-pixel average 0x3: 8-pixel average	RW	0x0

Table 12-353. Register Call Summary for Register PRV_AVE

Camera ISP Functional Description • Preview Engine Features: [0] [1] [2]
Camera ISP Basic Programming Model • Preview Hardware Setup/Initialization: [3] • Summary of Constraints: [4] [5] [6]
Camera ISP Register Manual • Camera ISP_PREVIEW Register Summary: [7] • Camera ISP_PREVIEW Register Description: [8] [9] [10]

Table 12-354. PRV_HMED

Address Offset	0x0000 002C		
Physical Address	0x480B CE2C	**Instance**	ISP_PREVIEW
Description	HORIZONTAL MEDIAN FILTER REGISTER		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																						ODDDIST	EVENDIST	THRESHOLD							

Bits	Field Name	Description	Type	Reset
31:10	RESERVED	Reserved for module specific status information. Reads returns 0	RW	0x000000
9	ODDDIST	Distance between consecutive pixels of the same color in the odd line. 0x0: 1 0x1: 2	RW	0x0
8	EVENDIST	Distance between consecutive pixels of the same color in even line. 0x0: 1 0x1: 2	RW	0x0
7:0	THRESHOLD	Horizontal median filter threshold.	RW	0x00

Table 12-355. Register Call Summary for Register PRV_HMED

Camera ISP Functional Description • Preview Engine Features:
Camera ISP Basic Programming Model • Preview Hardware Setup/Initialization: [3]
Camera ISP Register Manual • Camera ISP_PREVIEW Register Summary: [4]

Table 12-356. PRV_NF

Address Offset	0x0000 0030		
Physical Address	0x480B CE30	**Instance**	ISP_PREVIEW
Description	NOISE FILTER REGISTER		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																														SPR	

Bits	Field Name	Description	Type	Reset
31:2	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x00000000
1:0	SPR	The spread value in noise filter algorithm	RW	0x0

Table 12-357. Register Call Summary for Register PRV_NF

Camera ISP Functional Description • Preview Engine Features:
Camera ISP Basic Programming Model • Preview Hardware Setup/Initialization: [1]
Camera ISP Use Case and Tips • Programming Flow :
Camera ISP Register Manual • Camera ISP_PREVIEW Register Summary: [4]

Table 12-358. PRV_WB_DGAIN

Address Offset	0x0000 0034		
Physical Address	0x480B CE34	**Instance**	ISP_PREVIEW
Description	WHITE BALANCE COEF REGISTER		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																								DGAIN							

Bits	Field Name	Description	Type	Reset
31:10	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x000000
9:0	DGAIN	Digital gain for the white balance. The data is in U10Q8 representation. The value can change anytime following the start of a frame.	RW	0x100

Table 12-359. Register Call Summary for Register PRV_WB_DGAIN

Camera ISP Functional Description
- Preview Engine Features: [0]

Camera ISP Basic Programming Model
- Preview Hardware Setup/Initialization: [1]
- Register Accessibility During Frame Processing: [2]

Camera ISP Use Case and Tips
- Programming Flow :

Camera ISP Register Manual
- Camera ISP_PREVIEW Register Summary: [5]

Table 12-360. PRV_WBGAIN

Address Offset	0x0000 0038		
Physical Address	0x480B CE38	**Instance**	ISP_PREVIEW
Description	WHITE BALANCE COEF REGISTER		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
COEF3								COEF2								COEF1								COEF0							

Bits	Field Name	Description	Type	Reset
31:24	COEF3	White balance gain - COEF3. The data is in U8Q5 representation. The value can change anytime following the start of a frame.	RW	0x20
23:16	COEF2	White balance gain - COEF2. The data is in U8Q5 representation. The value can change anytime following the start of a frame.	RW	0x20
15:8	COEF1	White balance gain - COEF1. The data is in U8Q5 representation. The value can change anytime following the start of a frame.	RW	0x20
7:0	COEF0	White balance gain - COEF0. The data is in U8Q5 representation. The value can change anytime following the start of a frame.	RW	0x20

Table 12-361. Register Call Summary for Register PRV_WBGAIN

Camera ISP Functional Description • Preview Engine Features: [0]
Camera ISP Basic Programming Model • Preview Hardware Setup/Initialization: [2] • Register Accessibility During Frame Processing: [3]
Camera ISP Use Case and Tips • Programming Flow :
Camera ISP Register Manual • Camera ISP_PREVIEW Register Summary: [9]

Table 12-362. PRV_WBSEL

Address Offset	0x0000 003C		
Physical Address	0x480B CE3C	**Instance**	ISP_PREVIEW
Description	WHITE BALANCE COEF SELECTION REGISTER		
Type	RW		

31 30	29 28	27 26	25 24	23 22	21 20	19 18	17 16	15 14	13 12	11 10	9 8	7 6	5 4	3 2	1 0
N3_3	N3_2	N3_1	N3_0	N2_3	N2_2	N2_1	N2_0	N1_3	N1_2	N1_1	N1_0	N0_3	N0_2	N0_1	N0_0

Bits	Field Name	Description	Type	Reset
31:30	N3_3	Coefficient selection for 3rd line, 3rd pixel. 0x0: COEF0 0x1: COEF1 0x2: COEF2 0x3: COEF3	RW	0x3
29:28	N3_2	Coefficient selection for 3rd line, 2rd pixel. 0x0: COEF0 0x1: COEF1 0x2: COEF2 0x3: COEF3	RW	0x2
27:26	N3_1	Coefficient selection for 3rd line, 1st pixel. 0x0: COEF0 0x1: COEF1 0x2: COEF2 0x3: COEF3	RW	0x3
25:24	N3_0	Coefficient selection for 3rd line, 0th pixel. 0x0: COEF0 0x1: COEF1 0x2: COEF2 0x3: COEF3	RW	0x2
23:22	N2_3	Coefficient selection for 2nd line, 3rd pixel. 0x0: COEF0 0x1: COEF1 0x2: COEF2 0x3: COEF3	RW	0x1

Bits	Field Name	Description	Type	Reset
21:20	N2_2	Coefficient selection for 2nd line, 2nd pixel. 0x0: COEF0 0x1: COEF1 0x2: COEF2 0x3: COEF3	RW	0x0
19:18	N2_1	Coefficient selection for 2nd line, 1st pixel. 0x0: COEF0 0x1: COEF1 0x2: COEF2 0x3: COEF3	RW	0x1
17:16	N2_0	Coefficient selection for 2nd line, 0th pixel. 0x0: COEF0 0x1: COEF1 0x2: COEF2 0x3: COEF3	RW	0x0
15:14	N1_3	Coefficient selection for 1st line, 3rd pixel. 0x0: COEF0 0x1: COEF1 0x2: COEF2 0x3: COEF3	RW	0x3
13:12	N1_2	Coefficient selection for 1st line, 2nd pixel. 0x0: COEF0 0x1: COEF1 0x2: COEF2 0x3: COEF3	RW	0x2
11:10	N1_1	Coefficient selection for 1st line, 1st pixel. 0x0: COEF0 0x1: COEF1 0x2: COEF2 0x3: COEF3	RW	0x3
9:8	N1_0	Coefficient selection for 1st line, 0th pixel. 0x0: COEF0 0x1: COEF1 0x2: COEF2 0x3: COEF3	RW	0x2
7:6	N0_3	Coefficient selection for 0th line, 3rd pixel. 0x0: COEF0 0x1: COEF1 0x2: COEF2 0x3: COEF3	RW	0x1
5:4	N0_2	Coefficient selection for 0th line, 2nd pixel. 0x0: COEF0 0x1: COEF1 0x2: COEF2 0x3: COEF3	RW	0x0

Bits	Field Name	Description	Type	Reset
3:2	N0_1	Coefficient selection for 0th line, 1st pixel. 0x0: COEF0 0x1: COEF1 0x2: COEF2 0x3: COEF3	RW	0x1
1:0	N0_0	Coefficient selection for 0th line, 0th pixel. 0x0: COEF0 0x1: COEF1 0x2: COEF2 0x3: COEF3	RW	0x0

Table 12-363. Register Call Summary for Register PRV_WBSEL

Camera ISP Functional Description • Preview Engine Features:
Camera ISP Basic Programming Model • Preview Hardware Setup/Initialization: [1]
Camera ISP Use Case and Tips • Programming Flow :
Camera ISP Register Manual • Camera ISP_PREVIEW Register Summary: [5]

Table 12-364. PRV_CFA

Address Offset	0x0000 0040		
Physical Address	0x480B CE40	**Instance**	ISP_PREVIEW
Description	COLOR FILTER ARRAY REGISTER		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																GRADTH_VER								GRADTH_HOR							

Bits	Field Name	Description	Type	Reset
31:16	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x0000
15:8	GRADTH_VER	Gradient threshold vertical.	RW	0x00
7:0	GRADTH_HOR	Gradient threshold horizontal.	RW	0x00

Table 12-365. Register Call Summary for Register PRV_CFA

Camera ISP Functional Description • Preview Engine Features:
Camera ISP Basic Programming Model • Preview Hardware Setup/Initialization: [2]
Camera ISP Register Manual • Camera ISP_PREVIEW Register Summary: [3]

Table 12-366. PRV_BLKADJOFF

Address Offset	0x0000 0044		
Physical Address	0x480B CE44	**Instance**	ISP_PREVIEW
Description	BLACK ADJUSTMENT OFFSET REGISTER		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								R								G								B							

Bits	Field Name	Description	Type	Reset
31:24	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x00
23:16	R	Black-level offset adjustment for RED. The data is in 2's complement format.	RW	0x00
15:8	G	Black-level offset adjustment for GREEN. The data is in 2's complement format.	RW	0x00
7:0	B	Black-level offset adjustment for BLUE. The data is in 2's complement format.	RW	0x00

Table 12-367. Register Call Summary for Register PRV_BLKADJOFF

Camera ISP Functional Description
- Preview Engine Features: [0]

Camera ISP Basic Programming Model
- Preview Hardware Setup/Initialization: [1]

Camera ISP Register Manual
- Camera ISP_PREVIEW Register Summary: [2]

Table 12-368. PRV_RGB_MAT1

Address Offset	0x0000 0048		
Physical Address	0x480B CE48	**Instance**	ISP_PREVIEW
Description	RGB TO RGB MATRIX COEF REGISTER		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								MTX_GR								RESERVED				MTX_RR											

Bits	Field Name	Description	Type	Reset
31:28	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x0
27:16	MTX_GR	Blending value for GR position. The format is in S12Q8 representation.	RW	0x100
15:12	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x0
11:0	MTX_RR	Blending value for RR position. The format is in S12Q8 representation.	RW	0x100

Table 12-369. Register Call Summary for Register PRV_RGB_MAT1

Camera ISP Functional Description
- Preview Engine Features: [0]

Camera ISP Basic Programming Model
- Preview Hardware Setup/Initialization: [1]

Camera ISP Use Case and Tips
- Programming Flow :

Table 12-369. Register Call Summary for Register PRV_RGB_MAT1 (continued)

Camera ISP Register Manual • Camera ISP_PREVIEW Register Summary: [4]

Table 12-370. PRV_RGB_MAT2

Address Offset	0x0000 004C		
Physical Address	0x480B CE4C	**Instance**	ISP_PREVIEW
Description	RGB TO RGB MATRIX COEF REGISTER		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								MTX_RG								RESERVED								MTX_BR							

Bits	Field Name	Description	Type	Reset
31:28	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x0
27:16	MTX_RG	Blending value for RG position. The format is in S12Q8 representation.	RW	0x100
15:12	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x0
11:0	MTX_BR	Blending value for BR position. The format is in S12Q8 representation.	RW	0x100

Table 12-371. Register Call Summary for Register PRV_RGB_MAT2

Camera ISP Functional Description • Preview Engine Features: [0]
Camera ISP Use Case and Tips • Programming Flow :
Camera ISP Register Manual • Camera ISP_PREVIEW Register Summary: [3]

Table 12-372. PRV_RGB_MAT3

Address Offset	0x0000 0050		
Physical Address	0x480B CE50	**Instance**	ISP_PREVIEW
Description	RGB TO RGB MATRIX COEF REGISTER		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								MTX_BG								RESERVED								MTX_GG							

Bits	Field Name	Description	Type	Reset
31:28	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x0
27:16	MTX_BG	Blending value for BG position. The format is in S12Q8 representation.	RW	0x100
15:12	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x0
11:0	MTX_GG	Blending value for GG position. The format is in S12Q8 representation.	RW	0x100

Table 12-373. Register Call Summary for Register PRV_RGB_MAT3

Camera ISP Functional Description • Preview Engine Features: [0]
Camera ISP Use Case and Tips • Programming Flow :
Camera ISP Register Manual • Camera ISP_PREVIEW Register Summary: [3]

Table 12-374. PRV_RGB_MAT4

Address Offset	0x0000 0054		
Physical Address	0x480B CE54	**Instance**	ISP_PREVIEW
Description	RGB TO RGB MATRIX COEF REGISTER		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								MTX_GB								RESERVED								MTX_RB							

Bits	Field Name	Description	Type	Reset
31:28	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x0
27:16	MTX_GB	Blending value for GB position. The format is in S12Q8 representation.	RW	0x100
15:12	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x0
11:0	MTX_RB	Blending value for RB position. The format is in S12Q8 representation.	RW	0x100

Table 12-375. Register Call Summary for Register PRV_RGB_MAT4

Camera ISP Functional Description • Preview Engine Features: [0]
Camera ISP Use Case and Tips • Programming Flow :
Camera ISP Register Manual • Camera ISP_PREVIEW Register Summary: [3]

Table 12-376. PRV_RGB_MAT5

Address Offset	0x0000 0058		
Physical Address	0x480B CE58	**Instance**	ISP_PREVIEW
Description	RGB TO RGB MATRIX COEF REGISTER		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																								MTX_BB							

Bits	Field Name	Description	Type	Reset
31:12	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x00000
11:0	MTX_BB	Blending value for BB position. The format is in S12Q8 representation.	RW	0x100

Table 12-377. Register Call Summary for Register PRV_RGB_MAT5

Camera ISP Functional Description • Preview Engine Features: [0]
Camera ISP Basic Programming Model • Preview Hardware Setup/Initialization: [1]
Camera ISP Use Case and Tips • Programming Flow :
Camera ISP Register Manual • Camera ISP_PREVIEW Register Summary: [4]

Table 12-378. PRV_RGB_OFF1

Address Offset	0x0000 005C		
Physical Address	0x480B CE5C	**Instance**	ISP_PREVIEW
Description	RGB TO RGB MATRIX OFFSET REGISTER		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								MTX_OFFR								RESERVED								MTX_OFFG							

Bits	Field Name	Description	Type	Reset
31:26	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x00
25:16	MTX_OFFR	Blending offset value for RED. The data is in 2's complement format.	RW	0x000
15:10	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x00
9:0	MTX_OFFG	Blending offset value for GREEN. The data is in 2's complement format.	RW	0x000

Table 12-379. Register Call Summary for Register PRV_RGB_OFF1

Camera ISP Functional Description • Preview Engine Features: [0]
Camera ISP Basic Programming Model • Preview Hardware Setup/Initialization: [1]
Camera ISP Use Case and Tips • Programming Flow :
Camera ISP Register Manual • Camera ISP_PREVIEW Register Summary: [4]

Table 12-380. PRV_RGB_OFF2

Address Offset	0x0000 0060		
Physical Address	0x480B CE60	**Instance**	ISP_PREVIEW
Description	RGB TO RGB MATRIX OFFSET REGISTER		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																								MTX_OFFB							

Bits	Field Name	Description	Type	Reset
31:10	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x000000
9:0	MTX_OFFB	Blending offset value for BLUE (in 2's complemented representation).	RW	0x000

Table 12-381. Register Call Summary for Register PRV_RGB_OFF2

Camera ISP Functional Description • Preview Engine Features: [0]
Camera ISP Basic Programming Model • Preview Hardware Setup/Initialization: [1]
Camera ISP Use Case and Tips • Programming Flow :
Camera ISP Register Manual • Camera ISP_PREVIEW Register Summary: [4]

Table 12-382. PRV_CSC0

Address Offset	0x0000 0064		
Physical Address	0x480B CE64	**Instance**	ISP_PREVIEW
Description	COLOR SPACE CONVERSION COEF REGISTER		
Type	RW		

31 30	29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10	9 8	7 6 5 4 3 2 1 0
RESERVED	CSCBY	CSCGY		CSCRY	

Bits	Field Name	Description	Type	Reset
31:30	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x0
29:20	CSCBY	Color space conversion coefficient of BLUE for computing Y. The format is in S10Q8 representation.	RW	0x01C
19:10	CSCGY	Color space conversion coefficient of GREEN for computing Y. The format is in S10Q8 representation.	RW	0x098
9:0	CSCRY	Color space conversion coefficient of RED for computing Y. The format is in S10Q8 representation.	RW	0x04C

Table 12-383. Register Call Summary for Register PRV_CSC0

Camera ISP Functional Description • Preview Engine Features: [0]
Camera ISP Basic Programming Model • Preview Hardware Setup/Initialization: [1]
Camera ISP Use Case and Tips • Programming Flow :
Camera ISP Register Manual • Camera ISP_PREVIEW Register Summary: [4]

Table 12-384. PRV_CSC1

Address Offset	0x0000 0068		
Physical Address	0x480B CE68	**Instance**	ISP_PREVIEW
Description	COLOR SPACE CONVERSION COEF REGISTER		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								CSCBCB								CSCGCB								CSCRCB							

Bits	Field Name	Description	Type	Reset
31:30	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x0
29:20	CSCBCB	Color space conversion coefficient of BLUE for computing Y. The format is in S10Q8 representation.	RW	0x080
19:10	CSCGCB	Color space conversion coefficient of GREEN for computing Y. The format is in S10Q8 representation.	RW	0x3AC
9:0	CSCRCB	Color space conversion coefficient of RED for computing Cb. The format is in S10Q8 representation.	RW	0x3D4

Table 12-385. Register Call Summary for Register PRV_CSC1

Camera ISP Functional Description
- Preview Engine Features: [0]

Camera ISP Use Case and Tips
- Programming Flow :

Camera ISP Register Manual
- Camera ISP_PREVIEW Register Summary: [3]

Table 12-386. PRV_CSC2

Address Offset	0x0000 006C		
Physical Address	0x480B CE6C	**Instance**	ISP_PREVIEW
Description	COLOR SPACE CONVERSION COEF REGISTER		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								CSCBCR								CSCGCR								CSCRCR							

Bits	Field Name	Description	Type	Reset
31:30	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x0
29:20	CSCBCR	Color space conversion coefficient of BLUE for computing Cr. The format is in S10Q8 representation.	RW	0x3EC
19:10	CSCGCR	Color space conversion coefficient of GREEN for computing Cr. The format is in S10Q8 representation.	RW	0x080

Bits	Field Name	Description	Type	Reset
9:0	CSCRCR	Color space conversion coefficient of RED for computing Cr. The format is in S10Q8 representation.	RW	0x39E

Table 12-387. Register Call Summary for Register PRV_CSC2

Camera ISP Functional Description • Preview Engine Features: [0] [1] [2]
Camera ISP Basic Programming Model • Preview Hardware Setup/Initialization: [3]
Camera ISP Use Case and Tips • Programming Flow :
Camera ISP Register Manual • Camera ISP_PREVIEW Register Summary: [6]

Table 12-388. PRV_CSC_OFFSET

Address Offset	0x0000 0070		
Physical Address	0x480B CE70	**Instance**	ISP_PREVIEW
Description	COLOR SPACE CONVERSION OFFSET REGISTER		
Type	RW		

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10 9 8	7 6 5 4 3 2 1 0
RESERVED / RESERVED	YOFST	OFSTCB	OFSTCR

Bits	Field Name	Description	Type	Reset
31:26	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x00
25:24	RESERVED	Write 0s for future compatibility. Reads returns written value.	RW	0x0
23:16	YOFST	DC offset value for Y component. The data is in S8Q0 representation. Yout = Yin + YOFST	RW	0x00
15:8	OFSTCB	DC offset value for Cb component. The data is in S8Q0 representation. Cout = Cin + OFSTCB	RW	0x00
7:0	OFSTCR	DC offset value for Cr component. The data is in S8Q0 representation. Cout = Cin + OFSTCR	RW	0x00

Table 12-389. Register Call Summary for Register PRV_CSC_OFFSET

Camera ISP Functional Description • Preview Engine Features: [0]
Camera ISP Use Case and Tips • Programming Flow :
Camera ISP Register Manual • Camera ISP_PREVIEW Register Summary: [3]

Table 12-390. PRV_CNT_BRT

Address Offset	0x0000 0074		
Physical Address	0x480B CE74	**Instance**	ISP_PREVIEW
Description	CONTRAST SET REGISTER		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																CNT								BRT							

Bits	Field Name	Description	Type	Reset
31:16	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x0000
15:8	CNT	Contrast adjustment. Sets the contrast of the Y data. The data is in U8Q4 representation i.e (0 to 15.9375).. Applied after offset adjustment.	RW	0x10
7:0	BRT	Brightness adjustment. Sets the brightness of Y data. The data is in U8Q0 representation (0 to 255).. Applied after contrast adjustment.	RW	0x00

Table 12-391. Register Call Summary for Register PRV_CNT_BRT

Camera ISP Functional Description
- Preview Engine Features: [0] [1]

Camera ISP Basic Programming Model
- Preview Hardware Setup/Initialization: [2]

Camera ISP Use Case and Tips
- Programming Flow :

Camera ISP Register Manual
- Camera ISP_PREVIEW Register Summary: [6]

Table 12-392. PRV_CSUP

Address Offset	0x0000 0078		
Physical Address	0x480B CE78	**Instance**	ISP_PREVIEW
Description	CHROMINANCE SUPPRESSION SET REGISTER		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED															HPYF	CSUPTH								CSUPG							

Bits	Field Name	Description	Type	Reset
31:17	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x0000
16	HPYF	Use high-pass filter of luminance for chroma suppression 0x0: Disable. Use luminance without high-pass filter. 0x1: Enable	RW	0x0
15:8	CSUPTH	Chroma suppression threshold.	RW	0x00
7:0	CSUPG	Gain value for chroma suppression function. The data format is in U8Q8 representation.	RW	0x00

Table 12-393. Register Call Summary for Register PRV_CSUP

Camera ISP Functional Description • Preview Engine Features:
Camera ISP Basic Programming Model • Preview Hardware Setup/Initialization: [4]
Camera ISP Use Case and Tips • Programming Flow :
Camera ISP Register Manual • Camera ISP_PREVIEW Register Summary: [8]

Table 12-394. PRV_SETUP_YC

Address Offset	0x0000 007C		
Physical Address	0x480B CE7C	**Instance**	ISP_PREVIEW
Description	Y AND C SET REGISTER		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MAXY								MINY								MAXC								MINC							

Bits	Field Name	Description	Type	Reset
31:24	MAXY	Maximum Y value. The values greater than MAXY are clipped to MAXY.	RW	0xFF
23:16	MINY	Minimum Y value. The values smaller than MINY are clipped to MINY.	RW	0x00
15:8	MAXC	Maximum Cb and Cr values. The values greater than MAXC are clipped to MAXC.	RW	0xFF
7:0	MINC	Minimum Cb and Cr values. The values smaller than MINC are clipped to MINC.	RW	0x00

Table 12-395. Register Call Summary for Register PRV_SETUP_YC

Camera ISP Functional Description • Preview Engine Features: [0]
Camera ISP Basic Programming Model • Preview Hardware Setup/Initialization: [1]
Camera ISP Register Manual • Camera ISP_PREVIEW Register Summary: [2]

Table 12-396. PRV_SET_TBL_ADDR

Address Offset	0x0000 0080		
Physical Address	0x480B CE80	**Instance**	ISP_PREVIEW
Description	SET TABLE ADDRESS REGISTER		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																								ADDR							

Bits	Field Name	Description	Type	Reset
31:13	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x00000
12:0	ADDR	13-bit address.	RW	0x0000

Table 12-397. Register Call Summary for Register PRV_SET_TBL_ADDR

Camera ISP Basic Programming Model

- Preview Hardware Setup/Initialization: [0] [1]

Camera ISP Register Manual

- Camera ISP_PREVIEW Register Summary: [2]

Table 12-398. PRV_SET_TBL_DATA

Address Offset	0x0000 0084		
Physical Address	0x480B CE84	**Instance**	ISP_PREVIEW
Description	SETUP TABLE DATA REGISTER		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								DATA																							

Bits	Field Name	Description	Type	Reset
31:20	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x000
19:0	DATA	Data to be written. All 20 bits are valid to set up the non-linear enhancer table. Only 8 LSBs are valid to set up the gamma, noise filter and CFA coefficient tables.	RW	0x-----

Table 12-399. Register Call Summary for Register PRV_SET_TBL_DATA

Camera ISP Basic Programming Model

- Preview Hardware Setup/Initialization: [0]
- Register Accessibility During Frame Processing: [1]

Camera ISP Register Manual

- Camera ISP_PREVIEW Register Summary: [2]

Table 12-400. PRV_CDC_THRx

Address Offset	0x0000 0090 + (0x4 * x)	Index	x = 0 to 4
Physical Address	0x480B CE90 + (0x4 * x)	**Instance**	ISP_PREVIEW
Description	COUPLET DEFECT CORRECTION THRESHOLD REGISTER FOR COLORx		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								CORRECT								RESERVED								DETECT							

Bits	Field Name	Description	Type	Reset
31:26	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x00
25:16	CORRECT	Correction threshold when couplet defect correction is selected. It must be set to 1023 for single defect correction.	RW	0x000
15:10	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x00
9:0	DETECT	Detection threshold when couplet defect correction is selected. It must be set to 0 for single defect correction.	RW	0x000

Table 12-401. Register Call Summary for Register PRV_CDC_THRx

Camera ISP Functional Description • Preview Engine Features:
Camera ISP Basic Programming Model • Preview Hardware Setup/Initialization: [9]
Camera ISP Register Manual • Camera ISP_PREVIEW Register Summary: [10] • Camera ISP_PREVIEW Register Description: [11]

12.6.9 Camera ISP_RESIZER Registers

12.6.9.1 Camera ISP_RESIZER Register Summary

Table 12-402. ISP_RESIZER Register Summary

Register Name	Type	Register Width (Bits)	Address Offset	Physical Address
RSZ_PID	R	32	0x0000 0000	0x480B D000
RSZ_PCR	RW	32	0x0000 0004	0x480B D004
RSZ_CNT	RW	32	0x0000 0008	0x480B D008
RSZ_OUT_SIZE	RW	32	0x0000 000C	0x480B D00C
RSZ_IN_START	RW	32	0x0000 0010	0x480B D010
RSZ_IN_SIZE	RW	32	0x0000 0014	0x480B D014
RSZ_SDR_INADD	RW	32	0x0000 0018	0x480B D018
RSZ_SDR_INOFF	RW	32	0x0000 001C	0x480B D01C
RSZ_SDR_OUTADD	RW	32	0x0000 0020	0x480B D020
RSZ_SDR_OUTOFF	RW	32	0x0000 0024	0x480B D024
RSZ_HFILT10	RW	32	0x0000 0028	0x480B D028
RSZ_HFILT32	RW	32	0x0000 002C	0x480B D02C
RSZ_HFILT54	RW	32	0x0000 0030	0x480B D030
RSZ_HFILT76	RW	32	0x0000 0034	0x480B D034
RSZ_HFILT98	RW	32	0x0000 0038	0x480B D038
RSZ_HFILT1110	RW	32	0x0000 003C	0x480B D03C
RSZ_HFILT1312	RW	32	0x0000 0040	0x480B D040
RSZ_HFILT1514	RW	32	0x0000 0044	0x480B D044
RSZ_HFILT1716	RW	32	0x0000 0048	0x480B D048
RSZ_HFILT1918	RW	32	0x0000 004C	0x480B D04C
RSZ_HFILT2120	RW	32	0x0000 0050	0x480B D050
RSZ_HFILT2322	RW	32	0x0000 0054	0x480B D054
RSZ_HFILT2524	RW	32	0x0000 0058	0x480B D058
RSZ_HFILT2726	RW	32	0x0000 005C	0x480B D05C
RSZ_HFILT2928	RW	32	0x0000 0060	0x480B D060
RSZ_HFILT3130	RW	32	0x0000 0064	0x480B D064
RSZ_VFILT10	RW	32	0x0000 0068	0x480B D068
RSZ_VFILT32	RW	32	0x0000 006C	0x480B D06C
RSZ_VFILT54	RW	32	0x0000 0070	0x480B D070
RSZ_VFILT76	RW	32	0x0000 0074	0x480B D074
RSZ_VFILT98	RW	32	0x0000 0078	0x480B D078
RSZ_VFILT1110	RW	32	0x0000 007C	0x480B D07C
RSZ_VFILT1312	RW	32	0x0000 0080	0x480B D080

Table 12-402. ISP_RESIZER Register Summary (continued)

Register Name	Type	Register Width (Bits)	Address Offset	Physical Address
RSZ_VFILT1514	RW	32	0x0000 0084	0x480B D084
RSZ_VFILT1716	RW	32	0x0000 0088	0x480B D088
RSZ_VFILT1918	RW	32	0x0000 008C	0x480B D08C
RSZ_VFILT2120	RW	32	0x0000 0090	0x480B D090
RSZ_VFILT2322	RW	32	0x0000 0094	0x480B D094
RSZ_VFILT2524	RW	32	0x0000 0098	0x480B D098
RSZ_VFILT2726	RW	32	0x0000 009C	0x480B D09C
RSZ_VFILT2928	RW	32	0x0000 00A0	0x480B D0A0
RSZ_VFILT3130	RW	32	0x0000 00A4	0x480B D0A4
RSZ_YENH	RW	32	0x0000 00A8	0x480B D0A8

12.6.9.2 Camera ISP_RESIZER Register Description

Table 12-403. RSZ_PID

Address Offset	0x0000 0000		
Physical Address	0x480B D000	**Instance**	ISP_RESIZER
Description	PERIPHERAL ID REGISTER		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								TID								CID								PREV							

Bits	Field Name	Description	Type	Reset
31:24	RESERVED	Write 0s for future compatibility. Reads return 0.	R	0x00
23:16	TID	Peripheral identification: RESIZER module.	R	0x10
15:8	CID	Class identification: Camera ISP.	R	0xFE
7:0	PREV	Peripheral revision number.	R	TI internal data

Table 12-404. Register Call Summary for Register RSZ_PID

Camera ISP Register Manual

- Camera ISP_RESIZER Register Summary: [0]

Table 12-405. RSZ_PCR

Address Offset	0x0000 0004		
Physical Address	0x480B D004	**Instance**	ISP_RESIZER
Description	PERIPHERAL CONTROL REGISTER		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																													ONESHOT	BUSY	ENABLE

Bits	Field Name	Description	Type	Reset
31:3	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x00000000
2	ONESHOT	One-shot or continuous mode selection. This bit only applies when the image source is CCDC or PREVIEW (RSZ_CNT INPSRC =0) 0x0: Continuous mode. The module must be disabled by software. The disable module is latched at the end of the frame it was written in. 0x1: One shot mode. Enable bit is reset to 0 once the busy bit turns to 1.	RW	0x1
1	BUSY	RESIZER module busy. 0x0: RESIZER module not busy. 0x1: RESIZER module busy.	R	0x0
0	ENABLE	RESIZER module enable. Memory to memory operation: Resizer always operates in one-shot mode in memory to memory operation. Enable bit is reset to 0 once the busy bit turns to 1. CCDC/PREVIEW to memory operation: The resizer operates either in one-shot or continuous mode. The behavior is defined by the RSZ_PCR [2] ONESHOT bit. The enable bit MUST be the last field written to resize a frame. The enable bit can be written when the resizer is busy. 0x0: Disable RESIZER module. 0x1: Enable RESIZER module.	RW	0x0

Table 12-406. Register Call Summary for Register RSZ_PCR

Camera ISP Basic Programming Model
- Enable/Disable Hardware: [0] [1] [2] [3]
- Events and Status Checking: [4] [5] [6]
- Register Accessibility During Frame Processing: [7] [8] [9]
- Inter-Frame Operations: [10]

Camera ISP Use Case and Tips
- Programming Flow :

Camera ISP Register Manual
- Camera ISP_RESIZER Register Summary: [12]
- Camera ISP_RESIZER Register Description: [13]

Table 12-407. RSZ_CNT

Address Offset	0x0000 0008		
Physical Address	0x480B D008	**Instance**	ISP_RESIZER
Description	RESIZER CONTROL REGISTER		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED		CBILIN	INPSRC	INPTYP	YCPOS	VSTPH		HSTPH				VRSZ												HRSZ							

Bits	Field Name	Description	Type	Reset
31:30	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x0
29	CBILIN	Chrominance horizontal algorithm select. Filtering that is same as luminance processing is intended only for downsampling, and bilinear interpolation is intended only for upsampling. 0x0: The chrominance uses the same processing as the luminance. 0x1: The chrominance uses bilinear interpolation processing.	RW	0x0
28	INPSRC	Input source select. NOTE: If this field is set to zero, the resizer input is fed from either the preview engine or the CCDC, but not both. The CCDC and preview engine modules have individual control to feed their output to the resizer input. If both the CCDC and preview engine modules are set to drive the resizer input, the CCDC output takes precedence. 0x0: PREVIEW or CCDC module. 0x1: Memory.	RW	0x0
27	INPTYP	Input type select. 0x0: YUV422 color is interleaved 0x1: Color separate data on 8 bits.	RW	0x0
26	YCPOS	Luminance and chrominance position in 16-bit word 0x0: YC 0x1: CY	RW	0x0
25:23	VSTPH	Vertical starting phase. The phase range is 0 to 7.	RW	0x0
22:20	HSTPH	Horizontal starting phase. The phase range is 0 to 7.	RW	0x0
19:10	VRSZ	Vertical resizing value. (range from 64 - 1024) plus 1 Vertical resizing ratio is 256/(VRSZ+1) For have the correct functionality, must enter the desired value minus 1. For example for a ratio of 1, must enter 255	RW	0x0FF
9:0	HRSZ	Horizontal resizing value. (range from 64 - 1024) plus 1 Horizontal resizing ratio is 256/(HRSZ+1) For have the correct functionality, must enter the desired value minus 1. For example for a ratio of 1, must enter 255	RW	0x0FF

Table 12-408. Register Call Summary for Register RSZ_CNT

Camera ISP Functional Description
- Resizer: [0] [1] [2] [3] [4] [5] [6] [7] [8] [9] [10] [11] [12] [13] [14] [15] [16] [17] [18] [19] [20] [21] [22] [23] [24] [25] [26] [27]

Camera ISP Basic Programming Model
- Resizer Hardware Setup/Initialization: [28]
- Inter-Frame Operations: [29] [30]
- Summary of Constraints: [31] [32] [33] [34] [35] [36]

Camera ISP Use Case and Tips
- Programming Flow :

Camera ISP Register Manual
- Camera ISP_RESIZER Register Summary: [40]
- Camera ISP_RESIZER Register Description: [41]

Table 12-409. RSZ_OUT_SIZE

Address Offset	0x0000 000C		
Physical Address	0x480B D00C	**Instance**	ISP_RESIZER
Description	OUTPUT SIZE REGISTER		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								VERT								RESERVED								HORZ							

Bits	Field Name	Description	Type	Reset
31:28	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x00
27:16	VERT	Output height.	RW	0x000
15:12	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x00
11:0	HORZ	Output (width in the horizontal direction) The maximum output width cannot be greater than 3312 pixels wide (1650 if downsampling greater than 2 is used with 7 filter taps) This value must be EVEN and the number of bytes written to SDRAM must be a multiple of 16-bytes if the vertical resizing factor is greater than 1x (upsizing)	RW	0x000

Table 12-410. Register Call Summary for Register RSZ_OUT_SIZE

Camera ISP Functional Description
- Resizer: [0] [1] [2] [3] [4] [5] [6]

Camera ISP Basic Programming Model
- Resizer Hardware Setup/Initialization: [7]
- Summary of Constraints: [8] [9]

Camera ISP Use Case and Tips
- Programming Flow :

Camera ISP Register Manual
- Camera ISP_RESIZER Register Summary: [12]

Table 12-411. RSZ_IN_START

Address Offset	0x0000 0010		
Physical Address	0x480B D010	**Instance**	ISP_RESIZER
Description	INPUT CONFIGURATION REGISTER This register must be used when the input pixel data comes from the PREVIEW module. must be set to 0 is the input pixel data comes from memory.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								VERT_ST								RESERVED								HORZ_ST							

Bits	Field Name	Description	Type	Reset
31:29	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x0
28:16	VERT_ST	Vertical start line. This field makes sense when the resizer obtains its input from the preview engine/CCDC When the resizer gets its input from SDRAM, this field must be set to 0	RW	0x0000

Bits	Field Name	Description	Type	Reset
15:13	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x0
12:0	HORZ_ST	Horizontal start pixel. Pixels are coded on 16 bits for YUV and 8 bits for color separate data. When the resizer gets its input from SDRAM, this field must be set to <= 15 for YUV 16-bit data and <= 31 for 8-bit color separate data Horizontal starting pixel value is in number of pixels if input is from SDRAM. If the input to the resizer is from CCDC/preview engine, this field must be programmed as follows: (1) Program this field using number of bytes (twice number of pixels). (2) Change the lowest bit to reflect start position in pixels (effectively change from a value 0 to a value 1 if required)	RW	0x0000

Table 12-412. Register Call Summary for Register RSZ_IN_START

Camera ISP Functional Description • Resizer: [0] [1] [2] [3] [4]
Camera ISP Basic Programming Model • Resizer Hardware Setup/Initialization: [5]
Camera ISP Register Manual • Camera ISP_RESIZER Register Summary: [6]

Table 12-413. RSZ_IN_SIZE

Address Offset	0x0000 0014		
Physical Address	0x480B D014	**Instance**	ISP_RESIZER
Description	INPUT SIZE REGISTER		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								VERT								RESERVED								HORZ							

Bits	Field Name	Description	Type	Reset
31:29	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x0
28:16	VERT	Input height. The range is 0 to 4095 lines.	RW	0x0000
15:13	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x0
12:0	HORZ	Input width. The range is 0 to 4095 pixels.	RW	0x0000

Table 12-414. Register Call Summary for Register RSZ_IN_SIZE

Camera ISP Functional Description • Resizer: [0] [1] [2] [3] [4] [5]
Camera ISP Basic Programming Model • Resizer Hardware Setup/Initialization: [6] • Summary of Constraints: [7] [8] [9] [10] [11] [12]
Camera ISP Use Case and Tips • Programming Flow :

Table 12-414. Register Call Summary for Register RSZ_IN_SIZE (continued)

Camera ISP Register Manual • Camera ISP_RESIZER Register Summary: [15]

Table 12-415. RSZ_SDR_INADD

Address Offset	0x0000 0018		
Physical Address	0x480B D018	**Instance**	ISP_RESIZER
Description	INPUT ADDRESS REGISTER This register must be set only if the input pixel data comes from memory.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDR																															

Bits	Field Name	Description	Type	Reset
31:0	ADDR	32-bit input memory address. The 5 LSBs are forced to be zeros by the hardware to align on a 32-byte boundary; the 5 LSBs are read-only. This field must be programmed to be 0x0 if the resizer input is from preview engine/CCDC. * This field can be altered even when the resizer is busy. The change takes place only for the next frame. However, note that reading this register always gives the latest value.	RW	0x00000000

Table 12-416. Register Call Summary for Register RSZ_SDR_INADD

Camera ISP Functional Description • Resizer: [0] [1] [2]
Camera ISP Basic Programming Model • Resizer Hardware Setup/Initialization: [3] • Register Accessibility During Frame Processing: [4]
Camera ISP Register Manual • Camera ISP_RESIZER Register Summary: [5]

Table 12-417. RSZ_SDR_INOFF

Address Offset	0x0000 001C		
Physical Address	0x480B D01C	**Instance**	ISP_RESIZER
Description	INPUT OFFSET REGISTER This register must be set only if the input pixel data comes from memory.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																OFFSET															

Bits	Field Name	Description	Type	Reset
31:16	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x0000
15:0	OFFSET	Memory offset for the input lines. The 5 LSBs are forced to be zeros by the hardware to align on a 32-byte boundary; the 5 LSBs are read-only. This field must be programmed to be 0x0 if the resizer input is from preview engine/CCDC. * This field can be altered even when the resizer is busy. The change takes place only for the next frame. However, note that reading this register always gives the latest value.	RW	0x0000

Table 12-418. Register Call Summary for Register RSZ_SDR_INOFF

Camera ISP Functional Description • Resizer: [0] [1] [2]
Camera ISP Basic Programming Model • Resizer Hardware Setup/Initialization: [3] • Register Accessibility During Frame Processing: [4]
Camera ISP Register Manual • Camera ISP_RESIZER Register Summary: [5]

Table 12-419. RSZ_SDR_OUTADD

Address Offset	0x0000 0020		
Physical Address	0x480B D020	**Instance**	ISP_RESIZER
Description	OUTPUT ADDRESS REGISTER		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDR																															

Bits	Field Name	Description	Type	Reset
31:0	ADDR	32-bit output memory address. The 5 LSBs are forced to be zeros by the hardware to align on a 32-byte boundary; the 5 LSBs are read-only. For optimal use of SDRAM bandwidth, the SDRAM address must be set on a 256-byte boundary * This field can be altered even when the resizer is busy. The change takes place only for the next frame. However, note that reading this register always gives the latest value.	RW	0x00000000

Table 12-420. Register Call Summary for Register RSZ_SDR_OUTADD

Camera ISP Functional Description • Resizer: [0] [1]
Camera ISP Basic Programming Model • Resizer Hardware Setup/Initialization: [2] • Register Accessibility During Frame Processing: [3]
Camera ISP Use Case and Tips • Programming Flow :
Camera ISP Register Manual • Camera ISP_RESIZER Register Summary: [6]

Table 12-421. RSZ_SDR_OUTOFF

Address Offset	0x0000 0024		
Physical Address	0x480B D024	**Instance**	ISP_RESIZER
Description	OUTPUT OFFSET REGISTER		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																OFFSET															

Bits	Field Name	Description	Type	Reset
31:16	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x0000
15:0	OFFSET	Memory offset for the output lines. The 5 LSBs are forced to be zeros by the hardware to align on a 32-byte boundary; the 5 LSBs are read-only. For optimal use of SDRAM bandwidth, the SDRAM line offset must be set on a 256-byte boundary. * This field can be altered even when the resizer is busy. The change takes place only for the next frame. However, note that reading this register always gives the latest value.	RW	0x0000

Table 12-422. Register Call Summary for Register RSZ_SDR_OUTOFF

Camera ISP Functional Description • Resizer: [0] [1]
Camera ISP Basic Programming Model • Resizer Hardware Setup/Initialization: [2] • Register Accessibility During Frame Processing: [3]
Camera ISP Use Case and Tips • Programming Flow :
Camera ISP Register Manual • Camera ISP_RESIZER Register Summary: [6]

Table 12-423. RSZ_HFILT10

Address Offset	0x0000 0028		
Physical Address	0x480B D028	**Instance**	ISP_RESIZER
Description	HORIZONTAL FILTER COEFFICIENTS 0 AND 1 REGISTER		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								COEF1								RESERVED								COEF0							

Bits	Field Name	Description	Type	Reset
31:26	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x00
25:16	COEF1	10-bit coefficient (S10Q8 format -> range of -2 to 1.255/256, 1 is 0x100) - Phase 0, tap 1	RW	0x000
15:10	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x00
9:0	COEF0	10-bit coefficient (S10Q8 format -> range of -2 to 1.255/256, 1 is 0x100) - Phase 0, tap 0	RW	0x000

Table 12-424. Register Call Summary for Register RSZ_HFILT10

Camera ISP Functional Description • Resizer: [0]
Camera ISP Basic Programming Model • Resizer Hardware Setup/Initialization: [1]
Camera ISP Use Case and Tips • Programming Flow :
Camera ISP Register Manual • Camera ISP_RESIZER Register Summary: [3]

Table 12-425. RSZ_HFILT32

Address Offset	0x0000 002C		
Physical Address	0x480B D02C	**Instance**	ISP_RESIZER
Description	HORIZONTAL FILTER COEFFICIENTS 2 AND 3 REGISTER		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								COEF3								RESERVED								COEF2							

Bits	Field Name	Description	Type	Reset
31:26	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x00
25:16	COEF3	10-bit coefficient (S10Q8 format -> range of -2 to 1.255/256, 1 is 0x100) - Phase 0, tap 3	RW	0x000
15:10	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x00
9:0	COEF2	10-bit coefficient (S10Q8 format -> range of -2 to 1.255/256, 1 is 0x100) - Phase 0, tap 2	RW	0x000

Table 12-426. Register Call Summary for Register RSZ_HFILT32

Camera ISP Use Case and Tips

- Programming Flow :

Camera ISP Register Manual

- Camera ISP_RESIZER Register Summary: [1]

Table 12-427. RSZ_HFILT54

Address Offset	0x0000 0030		
Physical Address	0x480B D030	**Instance**	ISP_RESIZER
Description	HORIZONTAL FILTER COEFFICIENTS 4 AND 5 REGISTER		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								COEF5								RESERVED								COEF4							

Bits	Field Name	Description	Type	Reset
31:26	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x00
25:16	COEF5	10-bit coefficient (S10Q8 format -> range of -2 to 1.255/256, 1 is 0x100) - Phase 0/1, tap 5/1	RW	0x000
15:10	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x00
9:0	COEF4	10-bit coefficient (S10Q8 format -> range of -2 to 1.255/256, 1 is 0x100) - Phase 0/1, tap 4/0	RW	0x000

Table 12-428. Register Call Summary for Register RSZ_HFILT54

Camera ISP Use Case and Tips

- Programming Flow :

Camera ISP Register Manual

- Camera ISP_RESIZER Register Summary: [1]

Table 12-429. RSZ_HFILT76

Address Offset	0x0000 0034		
Physical Address	0x480B D034	**Instance**	ISP_RESIZER
Description	HORIZONTAL FILTER COEFFICIENTS 6 AND 37REGISTER		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								COEF7								RESERVED								COEF6							

Bits	Field Name	Description	Type	Reset
31:26	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x00
25:16	COEF7	10-bit coefficient (S10Q8 format -> range of -2 to 1.255/256, 1 is 0x100) - Phase 0/1, tap 7/3	RW	0x000
15:10	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x00
9:0	COEF6	10-bit coefficient (S10Q8 format -> range of -2 to 1.255/256, 1 is 0x100) - Phase 0/1, tap 6/2	RW	0x000

Table 12-430. Register Call Summary for Register RSZ_HFILT76

Camera ISP Use Case and Tips

- Programming Flow :

Camera ISP Register Manual

- Camera ISP_RESIZER Register Summary: [1]

Table 12-431. RSZ_HFILT98

Address Offset	0x0000 0038		
Physical Address	0x480B D038	**Instance**	ISP_RESIZER
Description	HORIZONTAL FILTER COEFFICIENTS 8 AND 9 REGISTER		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								COEF9								RESERVED								COEF8							

Bits	Field Name	Description	Type	Reset
31:26	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x00
25:16	COEF9	10-bit coefficient (S10Q8 format -> range of -2 to 1.255/256, 1 is 0x100) - Phase 1/2, tap 1/1	RW	0x000
15:10	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x00
9:0	COEF8	10-bit coefficient (S10Q8 format -> range of -2 to 1.255/256, 1 is 0x100) - Phase 1/2, tap 0/0	RW	0x000

Table 12-432. Register Call Summary for Register RSZ_HFILT98

Camera ISP Use Case and Tips

- Programming Flow :

Camera ISP Register Manual

- Camera ISP_RESIZER Register Summary: [1]

Table 12-433. RSZ_HFILT1110

Address Offset	0x0000 003C		
Physical Address	0x480B D03C	**Instance**	ISP_RESIZER
Description	HORIZONTAL FILTER COEFFICIENTS 10 AND 11 REGISTER		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								COEF11								RESERVED								COEF10							

Bits	Field Name	Description	Type	Reset
31:26	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x00
25:16	COEF11	10-bit coefficient (S10Q8 format -> range of -2 to 1.255/256, 1 is 0x100) - Phase 1/2, tap 3/3	RW	0x000
15:10	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x00
9:0	COEF10	10-bit coefficient (S10Q8 format -> range of -2 to 1.255/256, 1 is 0x100) - Phase 1/2, tap 2/2	RW	0x000

Table 12-434. Register Call Summary for Register RSZ_HFILT1110

Camera ISP Use Case and Tips

- Programming Flow :

Camera ISP Register Manual

- Camera ISP_RESIZER Register Summary: [1]

Table 12-435. RSZ_HFILT1312

Address Offset	0x0000 0040		
Physical Address	0x480B D040	**Instance**	ISP_RESIZER
Description	HORIZONTAL FILTER COEFFICIENTS 12 AND 13 REGISTER		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								COEF13								RESERVED								COEF12							

Bits	Field Name	Description	Type	Reset
31:26	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x00
25:16	COEF13	10-bit coefficient (S10Q8 format -> range of -2 to 1.255/256, 1 is 0x100) - Phase 1/3, tap 5/1	RW	0x000
15:10	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x00
9:0	COEF12	10-bit coefficient (S10Q8 format -> range of -2 to 1.255/256, 1 is 0x100) - Phase 1/3, tap 4/0	RW	0x000

Table 12-436. Register Call Summary for Register RSZ_HFILT1312

Camera ISP Use Case and Tips

- Programming Flow :

Camera ISP Register Manual

- Camera ISP_RESIZER Register Summary: [1]

Table 12-437. RSZ_HFILT1514

Address Offset	0x0000 0044		
Physical Address	0x480B D044	**Instance**	ISP_RESIZER
Description	HORIZONTAL FILTER COEFFICIENTS 14 AND 15 REGISTER		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								COEF15								RESERVED								COEF14							

Bits	Field Name	Description	Type	Reset
31:26	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x00
25:16	COEF15	10-bit coefficient (S10Q8 format -> range of -2 to 1.255/256, 1 is 0x100) - Phase 1/3, tap 7/3	RW	0x000
15:10	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x00
9:0	COEF14	10-bit coefficient (S10Q8 format -> range of -2 to 1.255/256, 1 is 0x100) - Phase 1/3, tap 6/2	RW	0x000

Table 12-438. Register Call Summary for Register RSZ_HFILT1514

Camera ISP Use Case and Tips

- Programming Flow :

Camera ISP Register Manual

- Camera ISP_RESIZER Register Summary: [1]

Table 12-439. RSZ_HFILT1716

Address Offset	0x0000 0048		
Physical Address	0x480B D048	**Instance**	ISP_RESIZER
Description	HORIZONTAL FILTER COEFFICIENTS 17 AND 16 REGISTER		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								COEF17								RESERVED								COEF16							

Bits	Field Name	Description	Type	Reset
31:26	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x00
25:16	COEF17	10-bit coefficient (S10Q8 format -> range of -2 to 1.255/256, 1 is 0x100) - Phase 2/4, tap 1/1	RW	0x000
15:10	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x00
9:0	COEF16	10-bit coefficient (S10Q8 format -> range of -2 to 1.255/256, 1 is 0x100) - Phase 2/4, tap 0/0	RW	0x000

Table 12-440. Register Call Summary for Register RSZ_HFILT1716

Camera ISP Use Case and Tips

- Programming Flow :

Camera ISP Register Manual

- Camera ISP_RESIZER Register Summary: [1]

Table 12-441. RSZ_HFILT1918

Address Offset	0x0000 004C		
Physical Address	0x480B D04C	**Instance**	ISP_RESIZER
Description	HORIZONTAL FILTER COEFFICIENTS 18 AND 19 REGISTER		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								COEF19								RESERVED								COEF18							

Bits	Field Name	Description	Type	Reset
31:26	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x00
25:16	COEF19	10-bit coefficient (S10Q8 format -> range of -2 to 1.255/256, 1 is 0x100) - Phase 2/4, tap 3/3	RW	0x000
15:10	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x00
9:0	COEF18	10-bit coefficient (S10Q8 format -> range of -2 to 1.255/256, 1 is 0x100) - Phase 2/4, tap 2/2	RW	0x000

Table 12-442. Register Call Summary for Register RSZ_HFILT1918

Camera ISP Use Case and Tips

- Programming Flow :

Camera ISP Register Manual

- Camera ISP_RESIZER Register Summary: [1]

Table 12-443. RSZ_HFILT2120

Address Offset	0x0000 0050		
Physical Address	0x480B D050	**Instance**	ISP_RESIZER
Description	HORIZONTAL FILTER COEFFICIENTS 20 AND 21 REGISTER		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								COEF21								RESERVED								COEF20							

Bits	Field Name	Description	Type	Reset
31:26	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x00
25:16	COEF21	10-bit coefficient (S10Q8 format -> range of -2 to 1.255/256, 1 is 0x100) - Phase 2/5, tap 5/1	RW	0x000
15:10	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x00
9:0	COEF20	10-bit coefficient (S10Q8 format -> range of -2 to 1.255/256, 1 is 0x100) - Phase 2/5, tap 4/0	RW	0x000

Table 12-444. Register Call Summary for Register RSZ_HFILT2120

Camera ISP Use Case and Tips

- Programming Flow :

Camera ISP Register Manual

- Camera ISP_RESIZER Register Summary: [1]

Table 12-445. RSZ_HFILT2322

Address Offset	0x0000 0054		
Physical Address	0x480B D054	**Instance**	ISP_RESIZER
Description	HORIZONTAL FILTER COEFFICIENTS 22 AND 23 REGISTER		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								COEF23								RESERVED								COEF22							

Bits	Field Name	Description	Type	Reset
31:26	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x00
25:16	COEF23	10-bit coefficient (S10Q8 format -> range of -2 to 1.255/256, 1 is 0x100) - Phase 2/5, tap 7/3	RW	0x000
15:10	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x00
9:0	COEF22	10-bit coefficient (S10Q8 format -> range of -2 to 1.255/256, 1 is 0x100) - Phase 2/5, tap 6/2	RW	0x000

Table 12-446. Register Call Summary for Register RSZ_HFILT2322

Camera ISP Use Case and Tips

- Programming Flow :

Camera ISP Register Manual

- Camera ISP_RESIZER Register Summary: [1]

Table 12-447. RSZ_HFILT2524

Address Offset	0x0000 0058		
Physical Address	0x480B D058	**Instance**	ISP_RESIZER
Description	HORIZONTAL FILTER COEFFICIENTS 24 AND 25 REGISTER		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								COEF25								RESERVED								COEF24							

Bits	Field Name	Description	Type	Reset
31:26	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x00
25:16	COEF25	10-bit coefficient (S10Q8 format -> range of -2 to 1.255/256, 1 is 0x100) - Phase 3/6, tap 1/1	RW	0x000
15:10	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x00
9:0	COEF24	10-bit coefficient (S10Q8 format -> range of -2 to 1.255/256, 1 is 0x100) - Phase 3/6, tap 0/0	RW	0x000

Table 12-448. Register Call Summary for Register RSZ_HFILT2524

Camera ISP Use Case and Tips

- Programming Flow :

Camera ISP Register Manual

- Camera ISP_RESIZER Register Summary: [1]

Table 12-449. RSZ_HFILT2726

Address Offset	0x0000 005C		
Physical Address	0x480B D05C	**Instance**	ISP_RESIZER
Description	HORIZONTAL FILTER COEFFICIENTS 26 AND 27 REGISTER		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								COEF27								RESERVED								COEF26							

Bits	Field Name	Description	Type	Reset
31:26	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x00
25:16	COEF27	10-bit coefficient (S10Q8 format -> range of -2 to 1.255/256, 1 is 0x100) - Phase 3/6, tap 3/3	RW	0x000
15:10	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x00
9:0	COEF26	10-bit coefficient (S10Q8 format -> range of -2 to 1.255/256, 1 is 0x100) - Phase 3/6, tap 2/2	RW	0x000

Table 12-450. Register Call Summary for Register RSZ_HFILT2726

Camera ISP Use Case and Tips

- Programming Flow :

Camera ISP Register Manual

- Camera ISP_RESIZER Register Summary: [1]

Table 12-451. RSZ_HFILT2928

Address Offset	0x0000 0060		
Physical Address	0x480B D060	**Instance**	ISP_RESIZER
Description	HORIZONTAL FILTER COEFFICIENTS 28 AND 29 REGISTER		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								COEF29								RESERVED								COEF28							

Bits	Field Name	Description	Type	Reset
31:26	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x00
25:16	COEF29	10-bit coefficient (S10Q8 format -> range of -2 to 1.255/256, 1 is 0x100) - Phase 3/7, tap 5/1	RW	0x000
15:10	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x00
9:0	COEF28	10-bit coefficient (S10Q8 format -> range of -2 to 1.255/256, 1 is 0x100) - Phase 3/7, tap 4/0	RW	0x000

Table 12-452. Register Call Summary for Register RSZ_HFILT2928

Camera ISP Use Case and Tips

- Programming Flow :

Camera ISP Register Manual

- Camera ISP_RESIZER Register Summary: [1]

Table 12-453. RSZ_HFILT3130

Address Offset	0x0000 0064		
Physical Address	0x480B D064	**Instance**	ISP_RESIZER
Description	HORIZONTAL FILTER COEFFICIENTS 30 AND 31 REGISTER		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								COEF31								RESERVED								COEF30							

Bits	Field Name	Description	Type	Reset
31:26	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x00
25:16	COEF31	10-bit coefficient (S10Q8 format -> range of -2 to 1.255/256, 1 is 0x100) - Phase 3/7, tap 7/3	RW	0x000
15:10	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x00
9:0	COEF30	10-bit coefficient (S10Q8 format -> range of -2 to 1.255/256, 1 is 0x100) - Phase 3/7, tap 6/2	RW	0x000

Table 12-454. Register Call Summary for Register RSZ_HFILT3130

Camera ISP Functional Description
- Resizer: [0]

Camera ISP Basic Programming Model
- Resizer Hardware Setup/Initialization: [1]

Camera ISP Use Case and Tips
- Programming Flow :

Camera ISP Register Manual
- Camera ISP_RESIZER Register Summary: [3]

Table 12-455. RSZ_VFILT10

Address Offset	0x0000 0068		
Physical Address	0x480B D068	**Instance**	ISP_RESIZER
Description	VERTICAL FILTER COEFFICIENTS 0 AND 1 REGISTER		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								COEF1								RESERVED								COEF0							

Bits	Field Name	Description	Type	Reset
31:26	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x00
25:16	COEF1	10-bit coefficient (S10Q8 format -> range of -2 to 1.255/256, 1 is 0x100) - Phase 0, tap 1	RW	0x000
15:10	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x00
9:0	COEF0	10-bit coefficient (S10Q8 format -> range of -2 to 1.255/256, 1 is 0x100) - Phase 0, tap 0	RW	0x000

Table 12-456. Register Call Summary for Register RSZ_VFILT10

Camera ISP Functional Description
- Resizer: [0]

Table 12-456. Register Call Summary for Register RSZ_VFILT10 (continued)

Camera ISP Basic Programming Model • Resizer Hardware Setup/Initialization: [1]
Camera ISP Use Case and Tips • Programming Flow :
Camera ISP Register Manual • Camera ISP_RESIZER Register Summary: [3]

Table 12-457. RSZ_VFILT32

Address Offset	0x0000 006C		
Physical Address	0x480B D06C	**Instance**	ISP_RESIZER
Description	VERTICAL FILTER COEFFICIENTS 2 AND 3 REGISTER		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								COEF3								RESERVED								COEF2							

Bits	Field Name	Description	Type	Reset
31:26	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x00
25:16	COEF3	10-bit coefficient (S10Q8 format -> range of -2 to 1.255/256, 1 is 0x100) - Phase 0, tap 3	RW	0x000
15:10	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x00
9:0	COEF2	10-bit coefficient (S10Q8 format -> range of -2 to 1.255/256, 1 is 0x100) - Phase 0, tap 2	RW	0x000

Table 12-458. Register Call Summary for Register RSZ_VFILT32

Camera ISP Use Case and Tips • Programming Flow :
Camera ISP Register Manual • Camera ISP_RESIZER Register Summary: [1]

Table 12-459. RSZ_VFILT54

Address Offset	0x0000 0070		
Physical Address	0x480B D070	**Instance**	ISP_RESIZER
Description	VERTICAL FILTER COEFFICIENTS 4 AND 5 REGISTER		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								COEF5								RESERVED								COEF4							

Bits	Field Name	Description	Type	Reset
31:26	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x00
25:16	COEF5	10-bit coefficient (S10Q8 format -> range of -2 to 1.255/256, 1 is 0x100) - Phase 0/1, tap 5/1	RW	0x000
15:10	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x00
9:0	COEF4	10-bit coefficient (S10Q8 format -> range of -2 to 1.255/256, 1 is 0x100) - Phase 0/1, tap 4/0	RW	0x000

Table 12-460. Register Call Summary for Register RSZ_VFILT54

Camera ISP Use Case and Tips

- Programming Flow :

Camera ISP Register Manual

- Camera ISP_RESIZER Register Summary: [1]

Table 12-461. RSZ_VFILT76

Address Offset	0x0000 0074		
Physical Address	0x480B D074	**Instance**	ISP_RESIZER
Description	VERTICAL FILTER COEFFICIENTS 6 AND 7 REGISTER		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								COEF7								RESERVED								COEF6							

Bits	Field Name	Description	Type	Reset
31:26	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x00
25:16	COEF7	10-bit coefficient (S10Q8 format -> range of -2 to 1.255/256, 1 is 0x100) - Phase 0/1, tap 7/3	RW	0x000
15:10	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x00
9:0	COEF6	10-bit coefficient (S10Q8 format -> range of -2 to 1.255/256, 1 is 0x100) - Phase 0/1, tap 6/2	RW	0x000

Table 12-462. Register Call Summary for Register RSZ_VFILT76

Camera ISP Use Case and Tips

- Programming Flow :

Camera ISP Register Manual

- Camera ISP_RESIZER Register Summary: [1]

Table 12-463. RSZ_VFILT98

Address Offset	0x0000 0078		
Physical Address	0x480B D078	**Instance**	ISP_RESIZER
Description	VERTICAL FILTER COEFFICIENTS 8 AND 9 REGISTER		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								COEF9								RESERVED								COEF8							

Bits	Field Name	Description	Type	Reset
31:26	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x00
25:16	COEF9	10-bit coefficient (S10Q8 format -> range of -2 to 1.255/256, 1 is 0x100) - Phase 1/2, tap 1/1	RW	0x000
15:10	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x00
9:0	COEF8	10-bit coefficient (S10Q8 format -> range of -2 to 1.255/256, 1 is 0x100) - Phase 1/2, tap 0/0	RW	0x000

Table 12-464. Register Call Summary for Register RSZ_VFILT98

Camera ISP Use Case and Tips • Programming Flow :
Camera ISP Register Manual • Camera ISP_RESIZER Register Summary: [1]

Table 12-465. RSZ_VFILT1110

Address Offset	0x0000 007C		
Physical Address	0x480B D07C	**Instance**	ISP_RESIZER
Description	VERTICAL FILTER COEFFICIENTS 10 AND 11 REGISTER		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								COEF11								RESERVED								COEF10							

Bits	Field Name	Description	Type	Reset
31:26	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x00
25:16	COEF11	10-bit coefficient (S10Q8 format -> range of -2 to 1.255/256, 1 is 0x100) - Phase 1/2, tap 3/3	RW	0x000
15:10	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x00
9:0	COEF10	10-bit coefficient (S10Q8 format -> range of -2 to 1.255/256, 1 is 0x100) - Phase 1/2, tap 2/2	RW	0x000

Table 12-466. Register Call Summary for Register RSZ_VFILT1110

Camera ISP Use Case and Tips • Programming Flow :
Camera ISP Register Manual • Camera ISP_RESIZER Register Summary: [1]

Table 12-467. RSZ_VFILT1312

Address Offset	0x0000 0080		
Physical Address	0x480B D080	**Instance**	ISP_RESIZER
Description	VERTICAL FILTER COEFFICIENTS 12 AND 13 REGISTER		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								COEF13								RESERVED								COEF12							

Bits	Field Name	Description	Type	Reset
31:26	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x00
25:16	COEF13	10-bit coefficient (S10Q8 format -> range of -2 to 1.255/256, 1 is 0x100) - Phase 1/3, tap 5/1	RW	0x000
15:10	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x00
9:0	COEF12	10-bit coefficient (S10Q8 format -> range of -2 to 1.255/256, 1 is 0x100) - Phase 1/3, tap 4/0	RW	0x000

Table 12-468. Register Call Summary for Register RSZ_VFILT1312

Camera ISP Use Case and Tips

- Programming Flow :

Camera ISP Register Manual

- Camera ISP_RESIZER Register Summary: [1]

Table 12-469. RSZ_VFILT1514

Address Offset	0x0000 0084		
Physical Address	0x480B D084	**Instance**	ISP_RESIZER
Description	VERTICAL FILTER COEFFICIENTS 14 AND 15 REGISTER		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								COEF15								RESERVED								COEF14							

Bits	Field Name	Description	Type	Reset
31:26	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x00
25:16	COEF15	10-bit coefficient (S10Q8 format -> range of -2 to 1.255/256, 1 is 0x100) - Phase 1/3, tap 7/3	RW	0x000
15:10	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x00
9:0	COEF14	10-bit coefficient (S10Q8 format -> range of -2 to 1.255/256, 1 is 0x100) - Phase 1/3, tap 6/2	RW	0x000

Table 12-470. Register Call Summary for Register RSZ_VFILT1514

Camera ISP Use Case and Tips

- Programming Flow :

Camera ISP Register Manual

- Camera ISP_RESIZER Register Summary: [1]

Table 12-471. RSZ_VFILT1716

Address Offset	0x0000 0088		
Physical Address	0x480B D088	**Instance**	ISP_RESIZER
Description	VERTICAL FILTER COEFFICIENTS 16 AND 17 REGISTER		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								COEF17								RESERVED								COEF16							

Bits	Field Name	Description	Type	Reset
31:26	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x00
25:16	COEF17	10-bit coefficient (S10Q8 format -> range of -2 to 1.255/256, 1 is 0x100) - Phase 2/4, tap 1/1	RW	0x000
15:10	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x00
9:0	COEF16	10-bit coefficient (S10Q8 format -> range of -2 to 1.255/256, 1 is 0x100) - Phase 2/4, tap 0/0	RW	0x000

Table 12-472. Register Call Summary for Register RSZ_VFILT1716

Camera ISP Use Case and Tips • Programming Flow :
Camera ISP Register Manual • Camera ISP_RESIZER Register Summary: [1]

Table 12-473. RSZ_VFILT1918

Address Offset	0x0000 008C		
Physical Address	0x480B D08C	**Instance**	ISP_RESIZER
Description	VERTICAL FILTER COEFFICIENTS 18 AND 19 REGISTER		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								COEF19								RESERVED								COEF18							

Bits	Field Name	Description	Type	Reset
31:26	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x00
25:16	COEF19	10-bit coefficient (S10Q8 format -> range of -2 to 1.255/256, 1 is 0x100) - Phase 2/4, tap 3/3	RW	0x000
15:10	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x00
9:0	COEF18	10-bit coefficient (S10Q8 format -> range of -2 to 1.255/256, 1 is 0x100) - Phase 2/4, tap 2/2	RW	0x000

Table 12-474. Register Call Summary for Register RSZ_VFILT1918

Camera ISP Use Case and Tips • Programming Flow :
Camera ISP Register Manual • Camera ISP_RESIZER Register Summary: [1]

Table 12-475. RSZ_VFILT2120

Address Offset	0x0000 0090		
Physical Address	0x480B D090	**Instance**	ISP_RESIZER
Description	VERTICAL FILTER COEFFICIENTS 20 AND 21 REGISTER		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								COEF21								RESERVED								COEF20							

Bits	Field Name	Description	Type	Reset
31:26	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x00
25:16	COEF21	10-bit coefficient (S10Q8 format -> range of -2 to 1.255/256, 1 is 0x100) - Phase 2/5, tap 5/1	RW	0x000
15:10	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x00
9:0	COEF20	10-bit coefficient (S10Q8 format -> range of -2 to 1.255/256, 1 is 0x100) - Phase 2/5, tap 4/0	RW	0x000

Table 12-476. Register Call Summary for Register RSZ_VFILT2120

Camera ISP Use Case and Tips

- Programming Flow :

Camera ISP Register Manual

- Camera ISP_RESIZER Register Summary: [1]

Table 12-477. RSZ_VFILT2322

Address Offset	0x0000 0094		
Physical Address	0x480B D094	**Instance**	ISP_RESIZER
Description	VERTICAL FILTER COEFFICIENTS 22 AND 23 REGISTER		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								COEF23								RESERVED								COEF22							

Bits	Field Name	Description	Type	Reset
31:26	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x00
25:16	COEF23	10-bit coefficient (S10Q8 format -> range of -2 to 1.255/256, 1 is 0x100) - Phase 2/5, tap 7/3	RW	0x000
15:10	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x00
9:0	COEF22	10-bit coefficient (S10Q8 format -> range of -2 to 1.255/256, 1 is 0x100) - Phase 2/5, tap 6/2	RW	0x000

Table 12-478. Register Call Summary for Register RSZ_VFILT2322

Camera ISP Use Case and Tips

- Programming Flow :

Camera ISP Register Manual

- Camera ISP_RESIZER Register Summary: [1]

Table 12-479. RSZ_VFILT2524

Address Offset	0x0000 0098		
Physical Address	0x480B D098	**Instance**	ISP_RESIZER
Description	VERTICAL FILTER COEFFICIENTS 24 AND 25 REGISTER		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								COEF25								RESERVED								COEF24							

Bits	Field Name	Description	Type	Reset
31:26	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x00
25:16	COEF25	10-bit coefficient (S10Q8 format -> range of -2 to 1.255/256, 1 is 0x100) - Phase 3/6, tap 1/1	RW	0x000
15:10	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x00
9:0	COEF24	10-bit coefficient (S10Q8 format -> range of -2 to 1.255/256, 1 is 0x100) - Phase 3/6, tap 0/0	RW	0x000

Table 12-480. Register Call Summary for Register RSZ_VFILT2524

Camera ISP Use Case and Tips

- Programming Flow :

Camera ISP Register Manual

- Camera ISP_RESIZER Register Summary: [1]

Table 12-481. RSZ_VFILT2726

Address Offset	0x0000 009C		
Physical Address	0x480B D09C	**Instance**	ISP_RESIZER
Description	VERTICAL FILTER COEFFICIENTS 26 AND 27 REGISTER		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								COEF27								RESERVED								COEF26							

Bits	Field Name	Description	Type	Reset
31:26	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x00
25:16	COEF27	10-bit coefficient (S10Q8 format -> range of -2 to 1.255/256, 1 is 0x100) - Phase 3/6, tap 3/3	RW	0x000
15:10	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x00
9:0	COEF26	10-bit coefficient (S10Q8 format -> range of -2 to 1.255/256, 1 is 0x100) - Phase 3/6, tap 2/2	RW	0x000

Table 12-482. Register Call Summary for Register RSZ_VFILT2726

Camera ISP Use Case and Tips

- Programming Flow :

Camera ISP Register Manual

- Camera ISP_RESIZER Register Summary: [1]

Table 12-483. RSZ_VFILT2928

Address Offset	0x0000 00A0		
Physical Address	0x480B D0A0	**Instance**	ISP_RESIZER
Description	VERTICAL FILTER COEFFICIENTS 28 AND 29 REGISTER		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								COEF29								RESERVED								COEF28							

Bits	Field Name	Description	Type	Reset
31:26	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x00
25:16	COEF29	10-bit coefficient (S10Q8 format -> range of -2 to 1.255/256, 1 is 0x100) - Phase 3/7, tap 5/1	RW	0x000
15:10	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x00
9:0	COEF28	10-bit coefficient (S10Q8 format -> range of -2 to 1.255/256, 1 is 0x100) - Phase 3/7, tap 4/0	RW	0x000

Table 12-484. Register Call Summary for Register RSZ_VFILT2928

Camera ISP Use Case and Tips
- Programming Flow :

Camera ISP Register Manual
- Camera ISP_RESIZER Register Summary: [1]

Table 12-485. RSZ_VFILT3130

Address Offset	0x0000 00A4		
Physical Address	0x480B D0A4	**Instance**	ISP_RESIZER
Description	VERTICAL FILTER COEFFICIENTS 30 AND 31 REGISTER		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								COEF31								RESERVED								COEF30							

Bits	Field Name	Description	Type	Reset
31:26	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x00
25:16	COEF31	10-bit coefficient (S10Q8 format -> range of -2 to 1.255/256, 1 is 0x100) - Phase 3/7, tap 7/3	RW	0x000
15:10	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x00
9:0	COEF30	10-bit coefficient (S10Q8 format -> range of -2 to 1.255/256, 1 is 0x100) - Phase 3/7, tap 6/2	RW	0x000

Table 12-486. Register Call Summary for Register RSZ_VFILT3130

Camera ISP Functional Description
- Resizer: [0]

Camera ISP Basic Programming Model
- Resizer Hardware Setup/Initialization: [1]

Camera ISP Use Case and Tips
- Programming Flow :

Camera ISP Register Manual
- Camera ISP_RESIZER Register Summary: [3]

Table 12-487. RSZ_YENH

Address Offset	0x0000 00A8		
Physical Address	0x480B D0A8	**Instance**	ISP_RESIZER
Description	LUMINANCE ENHANCER REGISTER The new luminance value is computed as: Y += HPF(Y) x max(GAIN, (|HPF(Y) - CORE) x SLOP + 8) >> 4.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED														ALGO		GAIN				SLOP				CORE							

Bits	Field Name	Description	Type	Reset
31:18	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x0000
17:16	ALGO	Algorithm select. 0x0: Disable. 0x1: [-1 2 -1]/2 high-pass filter. 0x2: [-1 -2 6 -2 -1]/4 high-pass filter.	RW	0x0

Bits	Field Name	Description	Type	Reset
15:12	GAIN	Maximum gain. The data is coded in U4Q4 representation.	RW	0x0
11:8	SLOP	Slope. The data is coded in U4Q4 representation.	RW	0x0
7:0	CORE	Coring offset. The data is coded in U8Q0 representation.	RW	0x00

Table 12-488. Register Call Summary for Register RSZ_YENH

Camera ISP Functional Description
- Resizer: [0] [1] [2] [3] [4] [5]

Camera ISP Basic Programming Model
- Resizer Hardware Setup/Initialization: [6] [7] [8] [9] [10] [11]
- Summary of Constraints: [12] [13]

Camera ISP Register Manual
- Camera ISP_RESIZER Register Summary: [14]

12.6.10 Camera ISP_SBL Registers

12.6.10.1 Camera ISP_SBL Register Summary

Table 12-489. ISP_SBL Register Summary

Register Name	Type	Register Width (Bits)	Address Offset	Physical Address
SBL_PID	R	32	0x0000 0000	0x480B D200
SBL_PCR	RW	32	0x0000 0004	0x480B D204
SBL_GLB_REG_0	R	32	0x0000 0008	0x480B D208
SBL_GLB_REG_1	R	32	0x0000 000C	0x480B D20C
SBL_GLB_REG_2	R	32	0x0000 0010	0x480B D210
SBL_GLB_REG_3	R	32	0x0000 0014	0x480B D214
SBL_GLB_REG_4	R	32	0x0000 0018	0x480B D218
SBL_GLB_REG_5	R	32	0x0000 001C	0x480B D21C
SBL_GLB_REG_6	R	32	0x0000 0020	0x480B D220
SBL_GLB_REG_7	R	32	0x0000 0024	0x480B D224
SBL_CCDC_WR_0	R	32	0x0000 0028	0x480B D228
SBL_CCDC_WR_1	R	32	0x0000 002C	0x480B D22C
SBL_CCDC_WR_2	R	32	0x0000 0030	0x480B D230
SBL_CCDC_WR_3	R	32	0x0000 0034	0x480B D234
SBL_CCDC_FP_RD_0	R	32	0x0000 0038	0x480B D238
SBL_CCDC_FP_RD_1	R	32	0x0000 003C	0x480B D23C
SBL_PRV_RD_0	R	32	0x0000 0040	0x480B D240
SBL_PRV_RD_1	R	32	0x0000 0044	0x480B D244
SBL_PRV_RD_2	R	32	0x0000 0048	0x480B D248
SBL_PRV_RD_3	R	32	0x0000 004C	0x480B D24C
SBL_PRV_WR_0	R	32	0x0000 0050	0x480B D250
SBL_PRV_WR_1	R	32	0x0000 0054	0x480B D254
SBL_PRV_WR_2	R	32	0x0000 0058	0x480B D258
SBL_PRV_WR_3	R	32	0x0000 005C	0x480B D25C
SBL_PRV_DK_RD_0	R	32	0x0000 0060	0x480B D260
SBL_PRV_DK_RD_1	R	32	0x0000 0064	0x480B D264

Table 12-489. ISP_SBL Register Summary (continued)

Register Name	Type	Register Width (Bits)	Address Offset	Physical Address
SBL_PRV_DK_RD_2	R	32	0x0000 0068	0x480B D268
SBL_PRV_DK_RD_3	R	32	0x0000 006C	0x480B D26C
SBL_RSZ_RD_0	R	32	0x0000 0070	0x480B D270
SBL_RSZ_RD_1	R	32	0x0000 0074	0x480B D274
SBL_RSZ_RD_2	R	32	0x0000 0078	0x480B D278
SBL_RSZ_RD_3	R	32	0x0000 007C	0x480B D27C
SBL_RSZ1_WR_0	R	32	0x0000 0080	0x480B D280
SBL_RSZ1_WR_1	R	32	0x0000 0084	0x480B D284
SBL_RSZ1_WR_2	R	32	0x0000 0088	0x480B D288
SBL_RSZ1_WR_3	R	32	0x0000 008C	0x480B D28C
SBL_RSZ2_WR_0	R	32	0x0000 0090	0x480B D290
SBL_RSZ2_WR_1	R	32	0x0000 0094	0x480B D294
SBL_RSZ2_WR_2	R	32	0x0000 0098	0x480B D298
SBL_RSZ2_WR_3	R	32	0x0000 009C	0x480B D29C
SBL_RSZ3_WR_0	R	32	0x0000 00A0	0x480B D2A0
SBL_RSZ3_WR_1	R	32	0x0000 00A4	0x480B D2A4
SBL_RSZ3_WR_2	R	32	0x0000 00A8	0x480B D2A8
SBL_RSZ3_WR_3	R	32	0x0000 00AC	0x480B D2AC
SBL_RSZ4_WR_0	R	32	0x0000 00B0	0x480B D2B0
SBL_RSZ4_WR_1	R	32	0x0000 00B4	0x480B D2B4
SBL_RSZ4_WR_2	R	32	0x0000 00B8	0x480B D2B8
SBL_RSZ4_WR_3	R	32	0x0000 00BC	0x480B D2BC
SBL_HIST_RD_0	R	32	0x0000 00C0	0x480B D2C0
SBL_HIST_RD_1	R	32	0x0000 00C4	0x480B D2C4
SBL_H3A_AF_WR_0	R	32	0x0000 00C8	0x480B D2C8
SBL_H3A_AF_WR_1	R	32	0x0000 00CC	0x480B D2CC
SBL_H3A_AEAWB_WR_0	R	32	0x0000 00D0	0x480B D2D0
SBL_H3A_AEAWB_WR_1	R	32	0x0000 00D4	0x480B D2D4
SBL_CSIA_WR_0	R	32	0x0000 00D8	0x480B D2D8
SBL_CSIA_WR_1	R	32	0x0000 00DC	0x480B D2DC
SBL_CSIA_WR_2	R	32	0x0000 00E0	0x480B D2E0
SBL_CSIA_WR_3	R	32	0x0000 00E4	0x480B D2E4
SBL_CSIB_WR_0	R	32	0x0000 00E8	0x480B D2E8
SBL_CSIB_WR_1	R	32	0x0000 00EC	0x480B D2EC
SBL_CSIB_WR_2	R	32	0x0000 00F0	0x480B D2F0
SBL_CSIB_WR_3	R	32	0x0000 00F4	0x480B D2F4
SBL_SDR_REQ_EXP	RW	32	0x0000 00F8	0x480B D2F8

12.6.10.2 Camera ISP_SBL Register Description

Table 12-490. SBL_PID

Address Offset	0x0000 0000		
Physical Address	See Table 12-489.	**Instance**	ISP_SBL
Description	PERIPHERAL IDENTIFICATION REGISTER		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								TID								CID								PREV							

Bits	Field Name	Description	Type	Reset
31:24	RESERVED	Write 0s for future compatibility. Reads return 0.	R	0x00
23:16	TID	Peripheral identification: SBL	R	0x01
15:8	CID	Class identification: Camera ISP	R	0xFB
7:0	PREV	Peripheral revision number.	R	TI internal data

Table 12-491. Register Call Summary for Register SBL_PID

Camera ISP Register Manual

- Camera ISP_SBL Register Summary: [0]

Table 12-492. SBL_PCR

Address Offset	0x0000 0004		
Physical Address	See Table 12-489.	**Instance**	ISP_SBL
Description	PERIPHERAL CONTROL REGISTER Note on the overflow bits: the overflow does not prevent the modules to make further requests. The only way to clear the overflow is to reset the bit. After an overflow, the data are corrupted and the application layer should disregard the data. No software reset is required after an overflow. If the overflow condition is cleared, the data continue to be acquired normally.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED					CSIB_WBL_OVF	CSIA_WBL_OVF	CCDCPRV_2_RSZ_OVF	CCDC_WBL_OVF	PRV_WBL_OVF	RSZ1_WBL_OVF	RSZ2_WBL_OVF	RSZ3_WBL_OVF	RSZ4_WBL_OVF	H3A_AF_WBL_OVF	H3A_AEAWB_WBL_OVF	RESERVED																RESERVED

Bits	Field Name	Description	Type	Reset
31:27	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x00
26	CSIB_WBL_OVF	CSIB Write buffer memory overflow All DUs have been filled up: overflow. Software has to write 1 to clear the bit. 0x0: No overflow 0x1: Overflow	RW	0x0
25	CSIA_WBL_OVF	CSIA Write buffer memory overflow All DUs have been filled up: overflow. Software has to write 1 to clear the bit. 0x0: No overflow 0x1: Overflow	RW	0x0

Bits	Field Name	Description	Type	Reset
24	CCDCPRV_2_RSZ_OVF	CCDC/PRV to RESIZER input overflow This bit is set if the RESIZER input source is set to CCDC/PREVIEW engine when the active data (to be resized) has already showed up at the resizer interface. In such a case, resizing for this frame cannot take place and the bit is set. This scenario can happen when a resize of > 4x is required per frame. Therefore, the RESIZER needs to operate in two passes. In the first pass the input data from CCDC/PREVIEW is directly resized and written to memory. In the second pass, the resized data from the first pass is resized again. The next frame from the CCDC/PREVIEW engine should only start after the second pass on the previous frame is complete. This bit indicated the failure status. Software has to write 1 to clear the bit. **Note:** Ignore the behavior of this field when RSZ_CNT[28] INPSRC = 0x1 (when data comes from memory). 0x0: No overflow 0x1: Overflow	RW	0x0
23	CCDC_WBL_OVF	CCDC Write buffer memory overflow All DUs have been filled up: overflow. Software has to write 1 to clear the bit. 0x0: No overflow 0x1: Overflow	RW	0x0
22	PRV_WBL_OVF	PREVIEW Write buffer memory overflow All DUs have been filled up: overflow. Software has to write 1 to clear the bit. 0x0: No overflow 0x1: Overflow	RW	0x0
21	RSZ1_WBL_OVF	RESIZER line 1 Write buffer memory overflow All DUs have been filled up: overflow. Software has to write 1 to clear the bit. 0x0: No overflow 0x1: Overflow	RW	0x0
20	RSZ2_WBL_OVF	RESIZER line 2 Write buffer memory overflow All DUs have been filled up: overflow. Software has to write 1 to clear the bit. 0x0: No overflow 0x1: Overflow	RW	0x0
19	RSZ3_WBL_OVF	RESIZER line 3 Write buffer memory overflow All DUs have been filled up: overflow. Software has to write 1 to clear the bit. 0x0: No overflow 0x1: Overflow	RW	0x0
18	RSZ4_WBL_OVF	RESIZER line 4 Write buffer memory overflow All DUs have been filled up: overflow. Software has to write 1 to clear the bit. 0x0: No overflow 0x1: Overflow	RW	0x0
17	H3A_AF_WBL_OVF	H3A AF Write buffer memory overflow All DUs have been filled up: overflow. Software has to write 1 to clear the bit. 0x0: No overflow 0x1: Overflow	RW	0x0

Bits	Field Name	Description	Type	Reset
16	H3A_AEAWB_WBL_OVF	H3A AE AWB Write buffer memory overflow All DUs have been filled up: overflow. Software has to write 1 to clear the bit. 0x0: No overflow 0x1: Overflow	RW	0x0
15:3	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x0000
2:0	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x0

Table 12-493. Register Call Summary for Register SBL_PCR

Camera ISP Basic Programming Model

- Event and Status Checking: [0] [1] [2] [3] [4] [5] [6] [7] [8] [9] [10] [11] [12]

Camera ISP Register Manual

- Camera ISP Register Description: [13] [14] [15] [16]
- Camera ISP_SBL Register Summary: [17]

Table 12-494. SBL_GLB_REG_0

Address Offset	0x0000 0008		
Physical Address	See Table 12-489.	**Instance**	ISP_SBL
Description	GLOBAL REGISTER 0		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																							SRC_DST_ID	SRC_DST_M						DIRECTION	VALID

Bits	Field Name	Description	Type	Reset
31:9	RESERVED	Write 0s for future compatibility. Reads return 0.	R	0x000000
8:7	SRC_DST_ID	Individual module register command number. For the modules that only have 2 individual requestors, this field only displays either 0 or 1. For modules that have 4 individual requestors, this field displays 0, 1, 2 or 3. 0x0: Read/write module requestor #1 0x1: Read/write module requestor #2 0x2: Read/write module requestor #3 0x3: Read/write module requestor #4	R	0x0

Bits	Field Name	Description	Type	Reset
6:2	SRC_DST_M	Source or destination module 0x0: CCDC module output 0x1: CCDC module fault pixel correction input 0x2: PREVIEW module input 0x3: PREVIEW module output 0x4: PREVIEW module dark frame subtract input 0x5: RESIZER module input 0x6: RESIZER module output line 1 0x7: RESIZER module output line 2 0x8: RESIZER module output line 3 0x9: RESIZER module output line 4 0xA: HISTOGRAM module input 0xB: H3A module output - autofocus 0xC: H3A module output - auto exposure and auto white balance 0xD: CSIa module output 0xE: CSIb module output	R	0x00
1	DIRECTION	Direction 0x0: Read 0x1: Write	R	0x0
0	VALID	Valid bit 0x0: Not valid. 0x1: Valid	R	0x0

Table 12-495. Register Call Summary for Register SBL_GLB_REG_0

Camera ISP Register Manual

- Camera ISP_SBL Register Summary: [0]

Table 12-496. SBL_GLB_REG_1

Address Offset	0x0000 000C		
Physical Address	See Table 12-489.	**Instance**	ISP_SBL
Description	GLOBAL REGISTER 1		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																								SRC_DST_ID	SRC_DST_M					DIRECTION	VALID

Bits	Field Name	Description	Type	Reset
31:9	RESERVED	Write 0s for future compatibility. Reads return 0.	R	0x000000
8:7	SRC_DST_ID	Individual module register command number. For the modules that only have 2 individual requestors, this field only displays either 0 or 1. For modules that have 4 individual requestors, this field displays 0, 1, 2 or 3. 0x0: Read/write module requestor #1 0x1: Read/write module requestor #2 0x2: Read/write module requestor #3 0x3: Read/write module requestor #4	R	0x0
6:2	SRC_DST_M	Source or destination module 0x0: CCDC module output 0x1: CCDC module fault pixel correction input 0x2: PREVIEW module input 0x3: PREVIEW module output 0x4: PREVIEW module dark frame subtract input 0x5: RESIZER module input 0x6: RESIZER module output line 1 0x7: RESIZER module output line 2 0x8: RESIZER module output line 3 0x9: RESIZER module output line 4 0xA: HISTOGRAM module input 0xB: H3A module output - autofocus 0xC: H3A module output - auto exposure and auto white balance 0xD: CSIa module output 0xE: CSIb module output	R	0x00
1	DIRECTION	Direction 0x0: Read 0x1: Write	R	0x0
0	VALID	Valid bit 0x0: Not valid. 0x1: Valid	R	0x0

Table 12-497. Register Call Summary for Register SBL_GLB_REG_1

Camera ISP Register Manual

- Camera ISP_SBL Register Summary: [0]

Table 12-498. SBL_GLB_REG_2

Address Offset	0x0000 0010		
Physical Address	See Table 12-489.	**Instance**	ISP_SBL
Description	GLOBAL REGISTER 2		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																								SRC_DST_ID		SRC_DST_M				DIRECTION	VALID

Bits	Field Name	Description	Type	Reset
31:9	RESERVED	Write 0s for future compatibility. Reads return 0.	R	0x000000
8:7	SRC_DST_ID	Individual module register command number. For the modules that only have 2 individual requestors, this field only displays either 0 or 1. For modules that have 4 individual requestors, this field displays 0, 1, 2 or 3. 0x0: Read/write module requestor #1 0x1: Read/write module requestor #2 0x2: Read/write module requestor #3 0x3: Read/write module requestor #4	R	0x0
6:2	SRC_DST_M	Source or destination module 0x0: CCDC module output 0x1: CCDC module fault pixel correction input 0x2: PREVIEW module input 0x3: PREVIEW module output 0x4: PREVIEW module dark frame subtract input 0x5: RESIZER module input 0x6: RESIZER module output line 1 0x7: RESIZER module output line 2 0x8: RESIZER module output line 3 0x9: RESIZER module output line 4 0xA: HISTOGRAM module input 0xB: H3A module output - autofocus 0xC: H3A module output - auto exposure and auto white balance 0xD: CSIa module output 0xE: CSIb module output	R	0x00
1	DIRECTION	Direction 0x0: Read 0x1: Write	R	0x0
0	VALID	Valid bit 0x0: Not valid. 0x1: Valid	R	0x0

Table 12-499. Register Call Summary for Register SBL_GLB_REG_2

Camera ISP Register Manual

- Camera ISP_SBL Register Summary: [0]

Table 12-500. SBL_GLB_REG_3

Address Offset	0x0000 0014		
Physical Address	See Table 12-489.	**Instance**	ISP_SBL
Description	GLOBAL REGISTER 3		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																								SRC_DST_ID		SRC_DST_M				DIRECTION	VALID

Bits	Field Name	Description	Type	Reset
31:9	RESERVED	Write 0s for future compatibility. Reads return 0.	R	0x000000
8:7	SRC_DST_ID	Individual module register command number. For the modules that only have 2 individual requestors, this field only displays either 0 or 1. For modules that have 4 individual requestors, this field displays 0, 1, 2 or 3. 0x0: Read/write module requestor #1 0x1: Read/write module requestor #2 0x2: Read/write module requestor #3 0x3: Read/write module requestor #4	R	0x0
6:2	SRC_DST_M	Source or destination module 0x0: CCDC module output 0x1: CCDC module fault pixel correction input 0x2: PREVIEW module input 0x3: PREVIEW module output 0x4: PREVIEW module dark frame subtract input 0x5: RESIZER module input 0x6: RESIZER module output line 1 0x7: RESIZER module output line 2 0x8: RESIZER module output line 3 0x9: RESIZER module output line 4 0xA: HISTOGRAM module input 0xB: H3A module output - autofocus 0xC: H3A module output - auto exposure and auto white balance 0xD: CSIa module output 0xE: CSIb module output	R	0x00
1	DIRECTION	Direction 0x0: Read 0x1: Write	R	0x0
0	VALID	Valid bit 0x0: Not valid. 0x1: Valid	R	0x0

Table 12-501. Register Call Summary for Register SBL_GLB_REG_3

Camera ISP Register Manual

- Camera ISP_SBL Register Summary: [0]

Table 12-502. SBL_GLB_REG_4

Address Offset	0x0000 0018		
Physical Address	See Table 12-489.	**Instance**	ISP_SBL
Description	GLOBAL REGISTER 4		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																								SRC_DST_ID		SRC_DST_M				DIRECTION	VALID

Bits	Field Name	Description	Type	Reset
31:9	RESERVED	Write 0s for future compatibility. Reads return 0.	R	0x000000
8:7	SRC_DST_ID	Individual module register command number. For the modules that only have 2 individual requestors, this field only displays either 0 or 1. For modules that have 4 individual requestors, this field displays 0, 1, 2 or 3. 0x0: Read/write module requestor #1 0x1: Read/write module requestor #2 0x2: Read/write module requestor #3 0x3: Read/write module requestor #4	R	0x0
6:2	SRC_DST_M	Source or destination module 0x0: CCDC module output 0x1: CCDC module fault pixel correction input 0x2: PREVIEW module input 0x3: PREVIEW module output 0x4: PREVIEW module dark frame subtract input 0x5: RESIZER module input 0x6: RESIZER module output line 1 0x7: RESIZER module output line 2 0x8: RESIZER module output line 3 0x9: RESIZER module output line 4 0xA: HISTOGRAM module input 0xB: H3A module output - autofocus 0xC: H3A module output - auto exposure and auto white balance 0xD: CSIa module output 0xE: CSIb module output	R	0x00
1	DIRECTION	Direction 0x0: Read 0x1: Write	R	0x0
0	VALID	Valid bit 0x0: Not valid. 0x1: Valid	R	0x0

Table 12-503. Register Call Summary for Register SBL_GLB_REG_4

Camera ISP Register Manual

- Camera ISP_SBL Register Summary: [0]

Table 12-504. SBL_GLB_REG_5

Address Offset	0x0000 001C		
Physical Address	See Table 12-489.	**Instance**	ISP_SBL
Description	GLOBAL REGISTER 5		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																								SRC_DST_ID		SRC_DST_M				DIRECTION	VALID

Bits	Field Name	Description	Type	Reset
31:9	RESERVED	Write 0s for future compatibility. Reads return 0.	R	0x000000
8:7	SRC_DST_ID	Individual module register command number. For the modules that only have 2 individual requestors, this field only displays either 0 or 1. For modules that have 4 individual requestors, this field displays 0, 1, 2 or 3. 0x0: Read/write module requestor #1 0x1: Read/write module requestor #2 0x2: Read/write module requestor #3 0x3: Read/write module requestor #4	R	0x0
6:2	SRC_DST_M	Source or destination module 0x0: CCDC module output 0x1: CCDC module fault pixel correction input 0x2: PREVIEW module input 0x3: PREVIEW module output 0x4: PREVIEW module dark frame subtract input 0x5: RESIZER module input 0x6: RESIZER module output line 1 0x7: RESIZER module output line 2 0x8: RESIZER module output line 3 0x9: RESIZER module output line 4 0xA: HISTOGRAM module input 0xB: H3A module output - autofocus 0xC: H3A module output - auto exposure and auto white balance 0xD: CSIa module output 0xE: CSIb module output	R	0x00
1	DIRECTION	Direction 0x0: Read 0x1: Write	R	0x0
0	VALID	Valid bit 0x0: Not valid. 0x1: Valid	R	0x0

Table 12-505. Register Call Summary for Register SBL_GLB_REG_5

Camera ISP Register Manual

- Camera ISP_SBL Register Summary: [0]

Table 12-506. SBL_GLB_REG_6

Address Offset	0x0000 0020		
Physical Address	See Table 12-489.	**Instance**	ISP_SBL
Description	GLOBAL REGISTER 6		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																								SRC_DST_ID		SRC_DST_M				DIRECTION	VALID

Bits	Field Name	Description	Type	Reset
31:9	RESERVED	Write 0s for future compatibility. Reads return 0.	R	0x000000
8:7	SRC_DST_ID	Individual module register command number. For the modules that only have 2 individual requestors, this field only displays either 0 or 1. For modules that have 4 individual requestors, this field displays 0, 1, 2 or 3. 0x0: Read/write module requestor #1 0x1: Read/write module requestor #2 0x2: Read/write module requestor #3 0x3: Read/write module requestor #4	R	0x0
6:2	SRC_DST_M	Source or destination module 0x0: CCDC module output 0x1: CCDC module fault pixel correction input 0x2: PREVIEW module input 0x3: PREVIEW module output 0x4: PREVIEW module dark frame subtract input 0x5: RESIZER module input 0x6: RESIZER module output line 1 0x7: RESIZER module output line 2 0x8: RESIZER module output line 3 0x9: RESIZER module output line 4 0xA: HISTOGRAM module input 0xB: H3A module output - autofocus 0xC: H3A module output - auto exposure and auto white balance 0xD: CSIa module output 0xE: CSIb module output	R	0x00
1	DIRECTION	Direction 0x0: Read 0x1: Write	R	0x0
0	VALID	Valid bit 0x0: Not valid. 0x1: Valid	R	0x0

Table 12-507. Register Call Summary for Register SBL_GLB_REG_6

Camera ISP Register Manual

- Camera ISP_SBL Register Summary: [0]

Table 12-508. SBL_GLB_REG_7

Address Offset	0x0000 0024		
Physical Address	See Table 12-489.	**Instance**	ISP_SBL
Description	GLOBAL REGISTER 7		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																								SRC_DST_ID		SRC_DST_M				DIRECTION	VALID

Bits	Field Name	Description	Type	Reset
31:9	RESERVED	Write 0s for future compatibility. Reads return 0.	R	0x000000
8:7	SRC_DST_ID	Individual module register command number. For the modules that only have 2 individual requestors, this field only displays either 0 or 1. For modules that have 4 individual requestors, this field displays 0, 1, 2 or 3. 0x0: Read/write module requestor #1 0x1: Read/write module requestor #2 0x2: Read/write module requestor #3 0x3: Read/write module requestor #4	R	0x0
6:2	SRC_DST_M	Source or destination module 0x0: CCDC module output 0x1: CCDC module fault pixel correction input 0x2: PREVIEW module input 0x3: PREVIEW module output 0x4: PREVIEW module dark frame subtract input 0x5: RESIZER module input 0x6: RESIZER module output line 1 0x7: RESIZER module output line 2 0x8: RESIZER module output line 3 0x9: RESIZER module output line 4 0xA: HISTOGRAM module input 0xB: H3A module output - autofocus 0xC: H3A module output - auto exposure and auto white balance 0xD: CSIa module output 0xE: CSIb module output	R	0x00
1	DIRECTION	Direction 0x0: Read 0x1: Write	R	0x0
0	VALID	Valid bit 0x0: Not valid. 0x1: Valid	R	0x0

Table 12-509. Register Call Summary for Register SBL_GLB_REG_7

Camera ISP Register Manual

- Camera ISP_SBL Register Summary: [0]

Table 12-510. SBL_CCDC_WR_0

Address Offset	0x0000 0028		
Physical Address	See Table 12-489.	**Instance**	ISP_SBL
Description	CCDC WRITE REQUEST 1 REGISTER		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								BYTE_CNT								DATA_READY	DATA_SENT	ADDR													

Bits	Field Name	Description	Type	Reset
31:30	RESERVED	Write 0s for future compatibility. Reads return 0.	R	0x0
29:22	BYTE_CNT	Current byte count	R	0x00
21	DATA_READY	Data ready 0x0: No 0x1: Yes	R	0x0
20	DATA_SENT	Data sent to the destination, waiting for status 0x0: No 0x1: Yes	R	0x0
19:0	ADDR	Upper 20 bits of the write address	R	0x00000

Table 12-511. Register Call Summary for Register SBL_CCDC_WR_0

Camera ISP Register Manual

- Camera ISP_SBL Register Summary: [0]

Table 12-512. SBL_CCDC_WR_1

Address Offset	0x0000 002C		
Physical Address	See Table 12-489.	**Instance**	ISP_SBL
Description	CCDC WRITE REQUEST 2 REGISTER		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								BYTE_CNT																DATA_READY	DATA_SENT	ADDR					

Bits	Field Name	Description	Type	Reset
31:30	RESERVED	Write 0s for future compatibility. Reads return 0.	R	0x0
29:22	BYTE_CNT	Current byte count	R	0x00
21	DATA_READY	Data ready 0x0: No 0x1: Yes	R	0x0
20	DATA_SENT	Data sent to the destination, waiting for status 0x0: No 0x1: Yes	R	0x0
19:0	ADDR	Upper 20 bits of the write address	R	0x00000

Table 12-513. Register Call Summary for Register SBL_CCDC_WR_1

Camera ISP Register Manual

- Camera ISP_SBL Register Summary: [0]

Table 12-514. SBL_CCDC_WR_2

Address Offset	0x0000 0030		
Physical Address	See Table 12-489.	**Instance**	ISP_SBL
Description	CCDC WRITE REQUEST 3 REGISTER		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								BYTE_CNT								DATA_READY	DATA_SENT	ADDR													

Bits	Field Name	Description	Type	Reset
31:30	RESERVED	Write 0s for future compatibility. Reads return 0.	R	0x0
29:22	BYTE_CNT	Current byte count	R	0x00
21	DATA_READY	Data ready 0x0: No 0x1: Yes	R	0x0
20	DATA_SENT	Data sent to the destination, waiting for status 0x0: No 0x1: Yes	R	0x0
19:0	ADDR	Upper 20 bits of the write address	R	0x00000

Table 12-515. Register Call Summary for Register SBL_CCDC_WR_2

Camera ISP Register Manual

- Camera ISP_SBL Register Summary: [0]

Table 12-516. SBL_CCDC_WR_3

Address Offset	0x0000 0034		
Physical Address	See Table 12-489.	**Instance**	ISP_SBL
Description	CCDC WRITE REQUEST 4 REGISTER		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								BYTE_CNT								DATA_READY	DATA_SENT	ADDR													

Bits	Field Name	Description	Type	Reset
31:30	RESERVED	Write 0s for future compatibility. Reads return 0.	R	0x0
29:22	BYTE_CNT	Current byte count.	R	0x00
21	DATA_READY	Data ready 0x0: No 0x1: Yes	R	0x0
20	DATA_SENT	Data sent to the destination, waiting for status 0x0: No 0x1: Yes	R	0x0

Bits	Field Name	Description	Type	Reset
19:0	ADDR	Upper 20 bits of the write address	R	0x00000

Table 12-517. Register Call Summary for Register SBL_CCDC_WR_3

Camera ISP Register Manual

- Camera ISP_SBL Register Summary: [0]

Table 12-518. SBL_CCDC_FP_RD_0

Address Offset	0x0000 0038		
Physical Address	See Table 12-489.	**Instance**	ISP_SBL
Description	CCDC FAULT PIXEL CORRECTION READ REQUEST 1 REGISTER		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED	VALID	DATA_WAIT	DATA _AVL	BYTE_CNT												ADDR															

Bits	Field Name	Description	Type	Reset
31	RESERVED	Write 0s for future compatibility. Reads return 0.	R	0
30	VALID	Valid bi Read 0x0: No Read 0x1: Yes	R	0
29	DATA_WAIT	Waiting for data Read 0x0: No Read 0x1: Yes	R	0
28	DATA _AVL	Data available. Received from source and can be read by the module. Read 0x0: No Read 0x1: Yes	R	0
27:20	BYTE_CNT	Byte count requested	R	0x00
19:0	ADDR	Upper 20 bits of the read address	R	0x00000

Table 12-519. Register Call Summary for Register SBL_CCDC_FP_RD_0

Camera ISP Register Manual

- Camera ISP_SBL Register Summary: [0]

Table 12-520. SBL_CCDC_FP_RD_1

Address Offset	0x0000 003C		
Physical Address	See Table 12-489.	**Instance**	ISP_SBL
Description	CCDC FAULT PIXEL CORRECTION READ REQUEST 2 REGISTER		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED	VALID	DATA_WAIT	DATA_AVL	BYTE_CNT												ADDR															

Bits	Field Name	Description	Type	Reset
31	RESERVED	Write 0s for future compatibility. Reads return 0.	R	0x0
30	VALID	Valid bit Read 0x0: No Read 0x1: Yes	R	0
29	DATA_WAIT	Waiting for data Read 0x0: No Read 0x1: Yes	R	0
28	DATA_AVL	Data available. Received from source and can be read by the module. Read 0x0: No Read 0x1: Yes	R	0
27:0	BYTE_CNT	Byte count requested	R	0x00
19:0	ADDR	Upper 20 bits of the read address	R	0x00000

Table 12-521. Register Call Summary for Register SBL_CCDC_FP_RD_1

Camera ISP Register Manual

- Camera ISP_SBL Register Summary: [0]

Table 12-522. SBL_PRV_RD_0

Address Offset	0x0000 0040		
Physical Address	See Table 12-489.	**Instance**	ISP_SBL
Description	PREVIEW READ REQUEST 1 REGISTER		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED	VALID	DATA_WAIT	DATA_AVL	BYTE_CNT												ADDR															

Bits	Field Name	Description	Type	Reset
31	RESERVED	Write 0s for future compatibility. Reads return 0.	R	0x0
30	VALID	Valid bit Read 0x0: No Read 0x1: Yes	R	0
29	DATA _WAIT	Waiting for data Read 0x0: No Read 0x1: Yes	R	0
28	DATA _AVL	Data available. Received from source and can be read by the module Read 0x0: No Read 0x1: Yes	R	0
27:20	BYTE CNT	Byte count requested	R	0x00
19:0	ADDR	Upper 20 bits of the read address	R	0x00000

Table 12-523. Register Call Summary for Register SBL_PRV_RD_0

Camera ISP Register Manual

- Camera ISP_SBL Register Summary: [0]

Table 12-524. SBL_PRV_RD_1

Address Offset	0x0000 0044		
Physical Address	See Table 12-489.	**Instance**	ISP_SBL
Description	PREVIEW READ REQUEST 2 REGISTER		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED	VALID	DATA_WAIT	DATA_AVL	BYTE_CNT												ADDR															

Bits	Field Name	Description	Type	Reset
31	RESERVED	Write 0s for future compatibility. Reads return 0.	R	0x0
30	VALID	Valid bit Read 0x0: No Read 0x1: Yes	R	0
29	DATA_WAIT	Waiting for data Read 0x0: No Read 0x1: Yes	R	0
28	DATA_AVL	Data available. Received from source and can be read by the module. Read 0x0: No Read 0x1: Yes	R	0
27:20	BYTE_CNT	Byte count requested	R	0x00
19:0	ADDR	Upper 20 bits of the read address	R	0x00000

Table 12-525. Register Call Summary for Register SBL_PRV_RD_1

Camera ISP Register Manual

- Camera ISP_SBL Register Summary: [0]

Table 12-526. SBL_PRV_RD_2

Address Offset	0x0000 0048		
Physical Address	See Table 12-489.	**Instance**	ISP_SBL
Description	PREVIEW READ REQUEST 3 REGISTER		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED	VALID	DATA_WAI T	DATA_AVL	BYTE_CNT												ADDR															

Bits	Field Name	Description	Type	Reset
31	RESERVED	Write 0s for future compatibility. Reads return 0.	R	0x0
30	VALID	Valid bit Read 0x0: No Read 0x1: Yes	R	0
29	DATA_WAI T	Waiting for data Read 0x0: No Read 0x1: Yes	R	0
28	DATA _AVL	Data available. Received from source and can be read by the module. Read 0x0: No Read 0x1: Yes	R	0
27:20	BYTE_CNT	Byte count requested	R	0x00
19:0	ADDR	Upper 20 bits of the read address	R	0x00000

Table 12-527. Register Call Summary for Register SBL_PRV_RD_2

Camera ISP Register Manual

- Camera ISP_SBL Register Summary: [0]

Table 12-528. SBL_PRV_RD_3

Address Offset	0x0000 004C		
Physical Address	See Table 12-489.	**Instance**	ISP_SBL
Description	PREVIEW READ REQUEST 4 REGISTER		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED	VALID	DATA_WAI T	DATA_AVL	BYTE_CNT												ADDR															

Bits	Field Name	Description	Type	Reset
31	RESERVED	Write 0s for future compatibility. Reads return 0.	R	0x0
30	VALID	Valid bit Read 0x0: No Read 0x1: Yes	R	0
29	DATA_WAI T	Waiting for data Read 0x0: No Read 0x1: Yes	R	0
28	DATA_AVL	Data available. Received from source and can be read by the module. Read 0x0: No Read 0x1: Yes	R	0
27:20	BYTE_CNT	Byte count requested	R	0x00
19:0	ADDR	Upper 20 bits of the read address	R	0x00000

Table 12-529. Register Call Summary for Register SBL_PRV_RD_3

Camera ISP Register Manual

- Camera ISP_SBL Register Summary: [0]

Table 12-530. SBL_PRV_WR_0

Address Offset	0x0000 0050		
Physical Address	See Table 12-489.	**Instance**	ISP_SBL
Description	PREVIEW WRITE REQUEST 1 REGISTER		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								BYTE_CNT								DATA_READY	DATA_SENT	ADDR													

Bits	Field Name	Description	Type	Reset
31:30	RESERVED	Write 0s for future compatibility. Reads return 0.	R	0x0
29:22	BYTE_CNT	Current byte count	R	0x00
21	DATA_READY	Data ready 0x0: No 0x1: Yes	R	0x0
20	DATA_SENT	Data sent to the destination, waiting for status 0x0: No 0x1: Yes	R	0x0
19:0	ADDR	Upper 20 bits of the write address	R	0x00000

Table 12-531. Register Call Summary for Register SBL_PRV_WR_0

Camera ISP Register Manual

- Camera ISP_SBL Register Summary: [0]

Table 12-532. SBL_PRV_WR_1

Address Offset	0x0000 0054		
Physical Address	See Table 12-489.	**Instance**	ISP_SBL
Description	PREVIEW WRITE REQUEST 2 REGISTER		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								BYTE_CNT								DATA_READY	DATA_SENT	ADDR													

Bits	Field Name	Description	Type	Reset
31:30	RESERVED	Write 0s for future compatibility. Reads return 0.	R	0x0
29:22	BYTE_CNT	Current byte count	R	0x00
21	DATA_READY	Data ready 0x0: No 0x1: Yes	R	0x0
20	DATA_SENT	Data sent to the destination, waiting for status 0x0: No 0x1: Yes	R	0x0
19:0	ADDR	Upper 20 bits of the write address	R	0x00000

Table 12-533. Register Call Summary for Register SBL_PRV_WR_1

Camera ISP Register Manual

- Camera ISP_SBL Register Summary: [0]

Table 12-534. SBL_PRV_WR_2

Address Offset	0x0000 0058		
Physical Address	See Table 12-489.	**Instance**	ISP_SBL
Description	PREVIEW WRITE REQUEST 3 REGISTER		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								BYTE_CNT								DATA_READY	DATA_SENT	ADDR													

Bits	Field Name	Description	Type	Reset
31:30	RESERVED	Write 0s for future compatibility. Reads return 0.	R	0x0
29:22	BYTE_CNT	Current byte count	R	0X00
21	DATA_READY	Data ready Read 0x0: No Write 0x1: Yes	R	0
20	DATA_SENT	Data sent to the destination, waiting for status Read 0x0: No Read 0x1:Yes	R	0
19:0	ADDR	Upper 20 bits of the write address	R	0x00000

Table 12-535. Register Call Summary for Register SBL_PRV_WR_2

Camera ISP Register Manual

- Camera ISP_SBL Register Summary: [0]

Table 12-536. SBL_PRV_WR_3

Address Offset	0x0000 005C		
Physical Address	See Table 12-489.	**Instance**	ISP_SBL
Description	PREVIEW WRITE REQUEST 4 REGISTER		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED		VALID		DATA_WAIT		DATA_AVL		BYTE_CNT								ADDR															

Bits	Field Name	Description	Type	Reset
31:30	RESERVED	Write 0s for future compatibility. Reads return 0.	R	0x0
29:22	BYTE_CNT	Current byte count	R	0x00
21	DATA_READY	Data ready Read 0x0: No Read 0x1: Yes	R	0
20	DATA_SENT	Data sent to the destination, waiting for status Read 0x0: No Read 0x1: Yes	R	0
19:0	ADDR	Upper 20 bits of the write address	R	0x00000

Table 12-537. Register Call Summary for Register SBL_PRV_WR_3

Camera ISP Register Manual

- Camera ISP_SBL Register Summary: [0]

Table 12-538. SBL_PRV_DK_RD_0

Address Offset	0x0000 0060		
Physical Address	See Table 12-489.	**Instance**	ISP_SBL
Description	PREVIEW READ REQUEST 1 REGISTER		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED		VALID		DATA_WAIT		DATA_AVL		BYTE_CNT								ADDR															

Bits	Field Name	Description	Type	Reset
31	RESERVED	Write 0s for future compatibility. Reads return 0.	R	0x0
30	VALID	Valid bit 0x0: No 0x1: Yes	R	0x0
29	DATA_WAIT	Waiting for data 0x0: No 0x1: Yes	R	0x0

Bits	Field Name	Description	Type	Reset
28	DATA_AVL	Data available. Received from source and can be read by the module. 0x0: No 0x1: Yes	R	0x0
27:20	BYTE_CNT	Byte count requested.	R	0x00
19:0	ADDR	Upper 20 bits of the read address.	R	0x00000

Table 12-539. Register Call Summary for Register SBL_PRV_DK_RD_0

Camera ISP Register Manual

- Camera ISP_SBL Register Summary: [0]

Table 12-540. SBL_PRV_DK_RD_1

Address Offset	0x0000 0064		
Physical Address	See Table 12-489.	**Instance**	ISP_SBL
Description	PREVIEW READ REQUEST 2 REGISTER		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED	VALID	DATA_WAIT	DATA_AVL	BYTE_CNT												ADDR															

Bits	Field Name	Description	Type	Reset
31	RESERVED	Write 0s for future compatibility. Reads return 0.	R	0x0
30	VALID	Valid bit 0x0: No 0x1: Yes	R	0x0
29	DATA_WAIT	Waiting for data 0x0: No 0x1: Yes	R	0x0
28	DATA_AVL	Data available. Received from source and can be read by the module. 0x0: No 0x1: Yes	R	0x0
27:20	BYTE_CNT	Byte count requested.	R	0x00
19:0	ADDR	Upper 20 bits of the read address.	R	0x00000

Table 12-541. Register Call Summary for Register SBL_PRV_DK_RD_1

Camera ISP Register Manual

- Camera ISP_SBL Register Summary: [0]

Table 12-542. SBL_PRV_DK_RD_2

Address Offset	0x0000 0068		
Physical Address	See Table 12-489.	**Instance**	ISP_SBL
Description	PREVIEW READ REQUEST 3 REGISTER		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED	VALID	DATA_WAIT	DATA_AVL	BYTE_CNT												ADDR															

Bits	Field Name	Description	Type	Reset
31	RESERVED	Write 0s for future compatibility. Reads return 0.	R	0x0
30	VALID	Valid bit 0x0: No 0x1: Yes	R	0x0
29	DATA_WAIT	Waiting for data 0x0: No 0x1: Yes	R	0x0
28	DATA_AVL	Data available. Received from source and can be read by the module. 0x0: No 0x1: Yes	R	0x0
27:20	BYTE_CNT	Byte count requested.	R	0x00
19:0	ADDR	Upper 20 bits of the read address.	R	0x00000

Table 12-543. Register Call Summary for Register SBL_PRV_DK_RD_2

Camera ISP Register Manual

- Camera ISP_SBL Register Summary: [0]

Table 12-544. SBL_PRV_DK_RD_3

Address Offset	0x0000 006C		
Physical Address	See Table 12-489.	**Instance**	ISP_SBL
Description	PREVIEW READ REQUEST 4 REGISTER		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED	VALID	DATA_WAIT	DATA_AVL	BYTE_CNT												ADDR															

Bits	Field Name	Description	Type	Reset
31	RESERVED	Write 0s for future compatibility. Reads return 0.	R	0x0
30	VALID	Valid bit 0x0: No 0x1: Yes	R	0x0

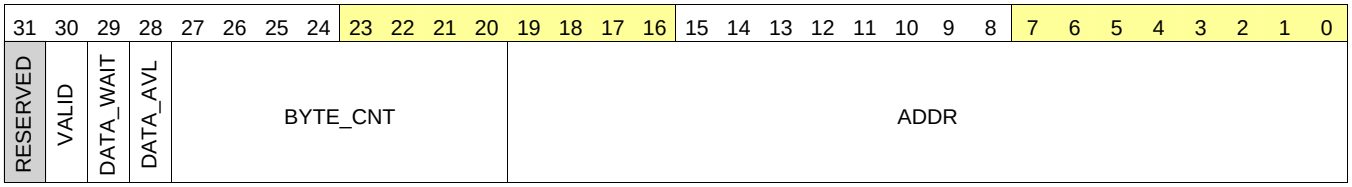
Bits	Field Name	Description	Type	Reset
29	DATA_WAIT	Waiting for data 0x0: No 0x1: Yes	R	0x0
28	DATA_AVL	Data available. Received from source and can be read by the module. 0x0: No 0x1: Yes	R	0x0
27:20	BYTE_CNT	Byte count requested.	R	0x00
19:0	ADDR	Upper 20 bits of the read address.	R	0x00000

Table 12-545. Register Call Summary for Register SBL_PRV_DK_RD_3

Camera ISP Register Manual
- Camera ISP_SBL Register Summary: [0]

Table 12-546. SBL_RSZ_RD_0

Address Offset	0x0000 0070		
Physical Address	See Table 12-489.	**Instance**	ISP_SBL
Description	RESIZER READ REQUEST 1 REGISTER		
Type	R		



Bits	Field Name	Description	Type	Reset
31	RESERVED	Write 0s for future compatibility. Reads return 0.	R	0x0
30	VALID	Valid bit Read 0x0: No Read 0x1: Yes	R	0
29	DATA_WAIT	Waiting for data Read 0x0: No Read 0x1: Yes	R	0
28	DATA_AVL	Data available. Received from source and can be read by the module. Read 0x0: No Read 0x1: Yes	R	0
27:20	BYTE_CNT	Byte count requested	R	0x00
19:0	ADDR	Upper 20 bits of the read address	R	0x00000

Table 12-547. Register Call Summary for Register SBL_RSZ_RD_0

Camera ISP Register Manual
- Camera ISP_SBL Register Summary: [0]

Table 12-548. SBL_RSZ_RD_1

Address Offset	0x0000 0074		
Physical Address	See Table 12-489.	**Instance**	ISP_SBL
Description	RESIZER READ REQUEST 2 REGISTER		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED	VALID	DATA_WAIT	DATA_AVL	BYTE_CNT												ADDR															

Bits	Field Name	Description	Type	Reset
31	RESERVED	Write 0s for future compatibility. Reads return 0.	R	0x0
30	VALID	Valid bit Read 0x0: No Read 0x1: Yes	R	0
29	DATA_WAIT	Waiting for data Read 0x0: No Read 0x1: Yes	R	0
28	DATA_AVL	Data available. Received from source and can be read by the module. Read 0x0: No Read 0x1: Yes	R	0
27:20	BYTE_CNT	Byte count requested	R	0x00
19:0	ADDR	Upper 20 bits of the read address	R	0x00000

Table 12-549. Register Call Summary for Register SBL_RSZ_RD_1

Camera ISP Register Manual

- Camera ISP_SBL Register Summary: [0]

Table 12-550. SBL_RSZ_RD_2

Address Offset	0x0000 0078		
Physical Address	See Table 12-489.	**Instance**	ISP_SBL
Description	RESIZER READ REQUEST 3 REGISTER		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED	VALID	DATA_WAIT	DATA_AVL	BYTE_CNT												ADDR															

Bits	Field Name	Description	Type	Reset
31	RESERVED	Write 0s for future compatibility. Reads return 0.	R	0x0
30	VALID	Valid bit Read 0x0: No Read 0x1: Yes	R	0

Bits	Field Name	Description	Type	Reset
29	DATA_WAIT	Waiting for data Read 0x0: No Read 0x1: Yes	R	0
28	DATA _AVL	Data available. Received from source and can be read by the module. Read 0x0: No Read 0x1: Yes	R	0
27:20	BYTE_CNT	Byte count requested	R	0x00
19:0	ADDR	Upper 20 bits of the read address	R	0x00000

Table 12-551. Register Call Summary for Register SBL_RSZ_RD_2

Camera ISP Register Manual
- Camera ISP_SBL Register Summary: [0]

Table 12-552. SBL_RSZ_RD_3

Address Offset	0x0000 007C		
Physical Address	See Table 12-489.	**Instance**	ISP_SBL
Description	RESIZER READ REQUEST 4 REGISTER		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED		VALID		DATA_WAIT		DATA_AVL		BYTE_CNT								ADDR															

Bits	Field Name	Description	Type	Reset
31	RESERVED	Write 0s for future compatibility. Reads return 0.	R	0x0
30	VALID	Valid bit Read 0x0: No Read 0x1: Yes	R	0
29	DATA_WAIT	Waiting for data Read 0x0: No Read 0x1: Yes	R	0
28	DATA_AVL	Data available. Received from source and can be read by the module. Read 0x0: No Read 0x1: Yes	R	0
27:20	BYTE_CNT	Byte count requested	R	0x00
19:0	ADDR	Upper 20 bits of the read address	R	0x00000

Table 12-553. Register Call Summary for Register SBL_RSZ_RD_3

Camera ISP Register Manual
- Camera ISP_SBL Register Summary: [0]

Table 12-554. SBL_RSZ1_WR_0

Address Offset	0x0000 0080		
Physical Address	See Table 12-489.	**Instance**	ISP_SBL
Description	RESIZER LINE 1 WRITE REQUEST 1 REGISTER		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								BYTE_CNT								DATA_READY	DATA_SENT	ADDR													

Bits	Field Name	Description	Type	Reset
31:30	RESERVED	Write 0s for future compatibility. Reads return 0.	R	0x0
29:22	BYTE_CNT	Current byte count	R	0x00
21	DATA_READY	Data ready Read 0x0: No Read 0x1: Yes	R	0
20	DATA_SENT	Data sent to the destination, waiting for status Read 0x0: No Read 0x1: Yes	R	0
19:0	ADDR	Upper 20 bits of the write address.	R	0x00000

Table 12-555. Register Call Summary for Register SBL_RSZ1_WR_0

Camera ISP Register Manual

- Camera ISP_SBL Register Summary: [0]

Table 12-556. SBL_RSZ1_WR_1

Address Offset	0x0000 0084		
Physical Address	See Table 12-489.	**Instance**	ISP_SBL
Description	RESIZER LINE 1 WRITE REQUEST 2 REGISTER		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								BYTE_CNT								DATA_READY	DATA_SENT	ADDR													

Bits	Field Name	Description	Type	Reset
31:30	RESERVED	Write 0s for future compatibility. Reads return 0.	R	0
29:22	BYTE_CNT	Current byte count	R	0x00
21	DATA_READY	Data ready Read 0x0: No Read 0x1: Yes	R	0
20	DATA_SENT	Data sent to the destination, waiting for status Read 0x0: No Read 0x1: Yes	R	0

Bits	Field Name	Description	Type	Reset
19:0	ADDR	Upper 20 bits of the write address.	R	0x00000

Table 12-557. Register Call Summary for Register SBL_RSZ1_WR_1

Camera ISP Register Manual

- Camera ISP_SBL Register Summary: [0]

Table 12-558. SBL_RSZ1_WR_2

Address Offset	0x0000 0088		
Physical Address	See Table 12-489.	**Instance**	ISP_SBL
Description	RESIZER LINE 1 WRITE REQUEST 3 REGISTER		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								BYTE_CNT								DATA_READY	DATA_SENT	ADDR													

Bits	Field Name	Description	Type	Reset
31:30	RESERVED	Write 0s for future compatibility. Reads return 0.	R	0x0
29:22	BYTE_CNT	Current byte count	R	0x00
21	DATA_READY	Data ready Read 0x0: No Read 0x1: Yes	R	0
20	DATA_SENT	Data sent to the destination, waiting for status Read 0x0: No Read 0x1: Yes	R	0
19:0	ADDR	Upper 20 bits of the write address	R	0x00000

Table 12-559. Register Call Summary for Register SBL_RSZ1_WR_2

Camera ISP Register Manual

- Camera ISP_SBL Register Summary: [0]

Table 12-560. SBL_RSZ1_WR_3

Address Offset	0x0000 008C		
Physical Address	See Table 12-489.	**Instance**	ISP_SBL
Description	RESIZER LINE 1 WRITE REQUEST 4 REGISTER		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								BYTE_CNT								DATA_READY	DATA_SENT	ADDR													

Bits	Field Name	Description	Type	Reset
31:30	RESERVED	Write 0s for future compatibility. Reads return 0.	R	0x0
29:22	BYTE_CNT	Current byte count	R	0x00
21	DATA_READY	Data ready Read 0x0: No Read 0x1: Yes	R	0x0
20	DATA_SENT	Data sent to the destination, waiting for status Read 0x0: No Read 0x1: Yes	R	0
19:0	ADDR	Upper 20 bits of the write address	R	0x00000

Table 12-561. Register Call Summary for Register SBL_RSZ1_WR_3

Camera ISP Register Manual

- Camera ISP_SBL Register Summary: [0]

Table 12-562. SBL_RSZ2_WR_0

Address Offset	0x0000 0090		
Physical Address	See Table 12-489.	**Instance**	ISP_SBL
Description	RESIZER LINE 2 WRITE REQUEST 1 REGISTER		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								BYTE_CNT																DATA_READY	DATA_SENT	ADDR					

Bits	Field Name	Description	Type	Reset
31:30	RESERVED	Write 0s for future compatibility. Reads return 0.	R	0x0
29:22	BYTE_CNT	Current byte count	R	0x00
21	DATA_READY	Data ready Read 0x0: No Read 0x1: Yes	R	0
20	DATA_SENT	Data sent to the destination, waiting for status Read 0x0: No Read 0x1: Yes	R	0
19:0	ADDR	Upper 20 bits of the write address	R	0x00000

Table 12-563. Register Call Summary for Register SBL_RSZ2_WR_0

Camera ISP Register Manual

- Camera ISP_SBL Register Summary: [0]

Table 12-564. SBL_RSZ2_WR_1

Address Offset	0x0000 0094		
Physical Address	See Table 12-489.	**Instance**	ISP_SBL
Description	RESIZER LINE 2 WRITE REQUEST 2 REGISTER		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								BYTE_CNT								DATA_READY	DATA_SENT	ADDR													

Bits	Field Name	Description	Type	Reset
31:30	RESERVED	Write 0s for future compatibility. Reads return 0.	R	0x0
29:22	BYTE_CNT	Current byte count	R	0x00
21	DATA_READY	Data ready Read 0x0: No Read 0x1: Yes	R	0
20	DATA_SENT	Data sent to the destination, waiting for status Read 0x0: No Read 0x1: Yes	R	0
19:0	ADDR	Upper 20 bits of the write address	R	0x00000

Table 12-565. Register Call Summary for Register SBL_RSZ2_WR_1

Camera ISP Register Manual

- Camera ISP_SBL Register Summary: [0]

Table 12-566. SBL_RSZ2_WR_2

Address Offset	0x0000 0098		
Physical Address	See Table 12-489.	**Instance**	ISP_SBL
Description	RESIZER LINE 2 WRITE REQUEST 3 REGISTER		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								BYTE_CNT								DATA_READY	DATA_SENT	ADDR													

Bits	Field Name	Description	Type	Reset
31:30	RESERVED	Write 0s for future compatibility. Reads return 0.	R	0x0
29:22	BYTE_CNT	Current byte count	R	0x00
21	DATA_READY	Data ready Read 0x0: No Read 0x1: Yes	R	0
20	DATA_SENT	Data sent to the destination, waiting for status Read 0x0: No Read 0x1: Yes	R	0

Bits	Field Name	Description	Type	Reset
19:0	ADDR	Upper 20 bits of the write address	R	0x00000

Table 12-567. Register Call Summary for Register SBL_RSZ2_WR_2

Camera ISP Register Manual

- Camera ISP_SBL Register Summary: [0]

Table 12-568. SBL_RSZ2_WR_3

Address Offset	0x0000 009C		
Physical Address	See Table 12-489.	**Instance**	ISP_SBL
Description	RESIZER LINE 2 WRITE REQUEST 4 REGISTER		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								BYTE_CNT								DATA_READY	DATA_SENT	ADDR													

Bits	Field Name	Description	Type	Reset
31:30	RESERVED	Write 0s for future compatibility. Reads return 0.	R	0x0
29:22	BYTE_CNT	Current byte count	R	0x00
21	DATA_READY	Data ready Read 0x0: No Read 0x1: Yes	R	0
20	DATA_SENT	Data sent to the destination, waiting for status Read 0x0: No Read 0x1: Yes	R	0
19:0	ADDR	Upper 20 bits of the write address	R	0x00000

Table 12-569. Register Call Summary for Register SBL_RSZ2_WR_3

Camera ISP Register Manual

- Camera ISP_SBL Register Summary: [0]

Table 12-570. SBL_RSZ3_WR_0

Address Offset	0x0000 00A0		
Physical Address	See Table 12-489.	**Instance**	ISP_SBL
Description	RESIZER LINE 3 WRITE REQUEST 1 REGISTER		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								BYTE_CNT								DATA_READY	DATA_SENT	ADDR													

Bits	Field Name	Description	Type	Reset
31:30	RESERVED	Write 0s for future compatibility. Reads return 0.	R	0x0
29:22	BYTE_CNT	Current byte count.	R	0x00
21	DATA_READY	Data ready 0x0: No 0x1: Yes	R	0x0
20	DATA_SENT	Data sent to the destination, waiting for status. 0x0: No 0x1: Yes	R	0x0
19:0	ADDR	Upper 20 bits of the write address	R	0x00000

Table 12-571. Register Call Summary for Register SBL_RSZ3_WR_0

Camera ISP Register Manual

- Camera ISP_SBL Register Summary: [0]

Table 12-572. SBL_RSZ3_WR_1

Address Offset	0x0000 00A4		
Physical Address	See Table 12-489.	**Instance**	ISP_SBL
Description	RESIZER LINE 3 WRITE REQUEST 2 REGISTER		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								BYTE_CNT																DATA_READY	DATA_SENT	ADDR					

Bits	Field Name	Description	Type	Reset
31:30	RESERVED	Write 0s for future compatibility. Reads return 0.	R	0x0
29:22	BYTE_CNT	Current byte count.	R	0x00
21	DATA_READY	Data ready 0x0: No 0x1: Yes	R	0x0
20	DATA_SENT	Data sent to the destination, waiting for status. 0x0: No 0x1: Yes	R	0x0
19:0	ADDR	Upper 20 bits of the write address.	R	0x00000

Table 12-573. Register Call Summary for Register SBL_RSZ3_WR_1

Camera ISP Register Manual

- Camera ISP_SBL Register Summary: [0]

Table 12-574. SBL_RSZ3_WR_2

Address Offset	0x0000 00A8		
Physical Address	See Table 12-489.	**Instance**	ISP_SBL
Description	RESIZER LINE 3 WRITE REQUEST 3 REGISTER		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								BYTE_CNT								DATA_READY	DATA_SENT	ADDR													

Bits	Field Name	Description	Type	Reset
31:30	RESERVED	Write 0s for future compatibility. Reads return 0.	R	0x0
29:22	BYTE_CNT	Current byte count.	R	0x00
21	DATA_READY	Data ready 0x0: No 0x1: Yes	R	0x0
20	DATA_SENT	Data sent to the destination, waiting for status. 0x0: No 0x1: Yes	R	0x0
19:0	ADDR	Upper 20 bits of the write address	R	0x00000

Table 12-575. Register Call Summary for Register SBL_RSZ3_WR_2

Camera ISP Register Manual

- Camera ISP_SBL Register Summary: [0]

Table 12-576. SBL_RSZ3_WR_3

Address Offset	0x0000 00AC		
Physical Address	See Table 12-489.	**Instance**	ISP_SBL
Description	RESIZER LINE 3 WRITE REQUEST 4 REGISTER		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								BYTE_CNT								DATA_READY	DATA_SENT	ADDR													

Bits	Field Name	Description	Type	Reset
31:30	RESERVED	Write 0s for future compatibility. Reads return 0.	R	0x0
29:22	BYTE_CNT	Current byte count.	R	0x00
21	DATA_READY	Data ready 0x0: No 0x1: Yes	R	0x0
20	DATA_SENT	Data sent to the destination, waiting for status. 0x0: No 0x1: Yes	R	0x0

Bits	Field Name	Description	Type	Reset
19:0	ADDR	Upper 20 bits of the write address	R	0x00000

Table 12-577. Register Call Summary for Register SBL_RSZ3_WR_3

Camera ISP Register Manual

- Camera ISP_SBL Register Summary: [0]

Table 12-578. SBL_RSZ4_WR_0

Address Offset	0x0000 00B0		
Physical Address	See Table 12-489.	**Instance**	ISP_SBL
Description	RESIZER LINE 4 WRITE REQUEST 1 REGISTER		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								BYTE_CNT								DATA_READY	DATA_SENT	ADDR													

Bits	Field Name	Description	Type	Reset
31:30	RESERVED	Write 0s for future compatibility. Reads return 0.	R	0x0
29:22	BYTE_CNT	Current byte count.	R	0x00
21	DATA_READY	Data ready 0x0: No 0x1: Yes	R	0x0
20	DATA_SENT	Data sent to the destination, waiting for status. 0x0: No 0x1: Yes	R	0x0
19:0	ADDR	Upper 20 bits of the write address	R	0x00000

Table 12-579. Register Call Summary for Register SBL_RSZ4_WR_0

Camera ISP Register Manual

- Camera ISP_SBL Register Summary: [0]

Table 12-580. SBL_RSZ4_WR_1

Address Offset	0x0000 00B4		
Physical Address	See Table 12-489.	**Instance**	ISP_SBL
Description	RESIZER LINE 4 WRITE REQUEST 2 REGISTER		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								BYTE_CNT								DATA_READY	DATA_SENT	ADDR													

Bits	Field Name	Description	Type	Reset
31:30	RESERVED	Write 0s for future compatibility. Reads return 0.	R	0x0
29:22	BYTE_CNT	Current byte count.	R	0x00
21	DATA_READY	Data ready 0x0: No 0x1: Yes	R	0x0
20	DATA_SENT	Data sent to the destination, waiting for status. 0x0: No 0x1: Yes	R	0x0
19:0	ADDR	Upper 20 bits of the write address.	R	0x00000

Table 12-581. Register Call Summary for Register SBL_RSZ4_WR_1

Camera ISP Register Manual

- Camera ISP_SBL Register Summary: [0]

Table 12-582. SBL_RSZ4_WR_2

Address Offset	0x0000 00B8		
Physical Address	See Table 12-489.	**Instance**	ISP_SBL
Description	RESIZER LINE 4 WRITE REQUEST 3 REGISTER		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								BYTE_CNT																DATA_READY	DATA_SENT	ADDR					

Bits	Field Name	Description	Type	Reset
31:30	RESERVED	Write 0s for future compatibility. Reads return 0.	R	0x0
29:22	BYTE_CNT	Current byte count.	R	0x00
21	DATA_READY	Data ready 0x0: No 0x1: Yes	R	0x0
20	DATA_SENT	Data sent to the destination, waiting for status. 0x0: No 0x1: Yes	R	0x0
19:0	ADDR	Upper 20 bits of the write address	R	0x00000

Table 12-583. Register Call Summary for Register SBL_RSZ4_WR_2

Camera ISP Register Manual

- Camera ISP_SBL Register Summary: [0]

Table 12-584. SBL_RSZ4_WR_3

Address Offset	0x0000 00BC		
Physical Address	See Table 12-489.	**Instance**	ISP_SBL
Description	RESIZER LINE 4 WRITE REQUEST 4 REGISTER		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								BYTE_CNT								DATA_READY	DATA_SENT	ADDR													

Bits	Field Name	Description	Type	Reset
31:30	RESERVED	Write 0s for future compatibility. Reads return 0.	R	0x0
29:22	BYTE_CNT	Current byte count.	R	0x00
21	DATA_READY	Data ready 0x0: No 0x1: Yes	R	0x0
20	DATA_SENT	Data sent to the destination, waiting for status. 0x0: No 0x1: Yes	R	0x0
19:0	ADDR	Upper 20 bits of the write address.	R	0x00000

Table 12-585. Register Call Summary for Register SBL_RSZ4_WR_3

Camera ISP Register Manual

- Camera ISP_SBL Register Summary: [0]

Table 12-586. SBL_HIST_RD_0

Address Offset	0x0000 00C0		
Physical Address	See Table 12-489.	**Instance**	ISP_SBL
Description	HISTOGRAM READ REQUEST 1 REGISTER		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED	VALID	DATA_WAIT	DATA_AVL	BYTE_CNT												ADDR															

Bits	Field Name	Description	Type	Reset
31	RESERVED	Write 0s for future compatibility. Reads return 0.	R	0x0
30	VALID	Valid bit Read 0x0: No Read 0x1: Yes	R	0
29	DATA_WAIT	Waiting for data Read 0x0: No Read 0x1: Yes	R	0
28	DATA_AVL	Data available. Received from source and can be read by the module.	R	0

Bits	Field Name	Description	Type	Reset
		Read 0x0: No		
		Read 0x1: Yes		
27:20	BYTE_CNT	Byte count requested	R	0x00
19:0	ADDR	Upper 20 bits of the read address	R	0x00000

Table 12-587. Register Call Summary for Register SBL_HIST_RD_0

Camera ISP Register Manual

- Camera ISP_SBL Register Summary: [0]

Table 12-588. SBL_HIST_RD_1

Address Offset	0x0000 00C4		
Physical Address	See Table 12-489.	**Instance**	ISP_SBL
Description	HISTOGRAM READ REQUEST 2 REGISTER		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED	VALID	DATA_WAIT	DATA_AVL	BYTE_CNT												ADDR															

Bits	Field Name	Description	Type	Reset
31	RESERVED	Write 0s for future compatibility. Reads return 0.	R	0x0
30	VALID	Valid bit	R	0
		Read 0x0: No		
		Read 0x1: Yes		
29	DATA_WAIT	Waiting for data	R	0
		Read 0x0: No		
		Read 0x1: Yes		
28	DATA_AVL	Data available. Received from source and can be read by the mdoule.	R	0
		Read 0x0: No		
		Read 0x1: Yes		
27:20	BYTE_CNT	Byte count requested	R	0x00
19:0	ADDR	Upper 20 bits of the read address	R	0x00000

Table 12-589. Register Call Summary for Register SBL_HIST_RD_1

Camera ISP Register Manual

- Camera ISP_SBL Register Summary: [0]

Table 12-590. SBL_H3A_AF_WR_0

Address Offset	0x0000 00C8		
Physical Address	See Table 12-489.	**Instance**	ISP_SBL
Description	H3A AF WRITE REQUEST 1 REGISTER		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								BYTE_CNT								DATA_READY	DATA_SENT	ADDR													

Bits	Field Name	Description	Type	Reset
31:30	RESERVED	Write 0s for future compatibility. Reads return 0.	R	0x0
29:22	BYTE_CNT	Current byte count.	R	0x00
21	DATA_READY	Data ready 0x0: No 0x1: Yes	R	0x0
20	DATA_SENT	Data sent to the destination, waiting for status. 0x0: No 0x1: Yes	R	0x0
19:0	ADDR	Upper 20 bits of the write address.	R	0x00000

Table 12-591. Register Call Summary for Register SBL_H3A_AF_WR_0

Camera ISP Register Manual

- Camera ISP_SBL Register Summary: [0]

Table 12-592. SBL_H3A_AF_WR_1

Address Offset	0x0000 00CC		
Physical Address	See Table 12-489.	**Instance**	ISP_SBL
Description	H3A AF WRITE REQUEST 2 REGISTER		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								BYTE_CNT								DATA_READY	DATA_SENT	ADDR													

Bits	Field Name	Description	Type	Reset
31:30	RESERVED	Write 0s for future compatibility. Reads return 0.	R	0x0
29:22	BYTE_CNT	Current byte count.	R	0x00
21	DATA_READY	Data ready 0x0: No 0x1: Yes	R	0x0
20	DATA_SENT	Data sent to the destination, waiting for status. 0x0: No 0x1: Yes	R	0x0

Bits	Field Name	Description	Type	Reset
19:0	ADDR	Upper 20 bits of the write address.	R	0x00000

Table 12-593. Register Call Summary for Register SBL_H3A_AF_WR_1

Camera ISP Register Manual

- Camera ISP_SBL Register Summary: [0]

Table 12-594. SBL_H3A_AEAWB_WR_0

Address Offset	0x0000 00D0		
Physical Address	See Table 12-489.	**Instance**	ISP_SBL
Description	H3A AEAWB WRITE REQUEST 1 REGISTER		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								BYTE_CNT								DATA_READY	DATA_SENT	ADDR													

Bits	Field Name	Description	Type	Reset
31:30	RESERVED	Write 0s for future compatibility. Reads return 0.	R	0x0
29:22	BYTE_CNT	Current byte count.	R	0x00
21	DATA_READY	Data ready 0x0: No 0x1: Yes	R	0x0
20	DATA_SENT	Data sent to the destination, waiting for status. 0x0: No 0x1: Yes	R	0x0
19:0	ADDR	Upper 20 bits of the write address.	R	0x00000

Table 12-595. Register Call Summary for Register SBL_H3A_AEAWB_WR_0

Camera ISP Register Manual

- Camera ISP_SBL Register Summary: [0]

Table 12-596. SBL_H3A_AEAWB_WR_1

Address Offset	0x0000 00D4		
Physical Address	See Table 12-489.	**Instance**	ISP_SBL
Description	H3A AEAWB WRITE REQUEST 2 REGISTER		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								BYTE_CNT								DATA_READY	DATA_SENT	ADDR													

Bits	Field Name	Description	Type	Reset
31:30	RESERVED	Write 0s for future compatibility. Reads return 0.	R	0x0
29:22	BYTE_CNT	Current byte count.	R	0x00
21	DATA_READY	Data ready 0x0: No 0x1: Yes	R	0x0
20	DATA_SENT	Data sent to the destination, waiting for status. 0x0: No 0x1: Yes	R	0x0
19:0	ADDR	Upper 20 bits of the write address.	R	0x00000

Table 12-597. Register Call Summary for Register SBL_H3A_AEAWB_WR_1

Camera ISP Register Manual

- Camera ISP_SBL Register Summary: [0]

Table 12-598. SBL_CSIA_WR_0

Address Offset	0x0000 00D8		
Physical Address	See Table 12-489.	**Instance**	ISP_SBL
Description	CSIA WRITE REQUEST 1 REGISTER		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								BYTE_CNT																DATA_READY	DATA_SENT	ADDR					

Bits	Field Name	Description	Type	Reset
31:30	RESERVED	Write 0s for future compatibility. Reads return 0.	R	0x0
29:22	BYTE_CNT	Current byte count.	R	0x00
21	DATA_READY	Data ready 0x0: No 0x1: Yes	R	0x0
20	DATA_SENT	Data sent to the destination, waiting for status. 0x0: No 0x1: Yes	R	0x0
19:0	ADDR	Upper 20 bits of the write address.	R	0x00000

Table 12-599. Register Call Summary for Register SBL_CSIA_WR_0

Camera ISP Register Manual

- Camera ISP_SBL Register Summary: [0]

Table 12-600. SBL_CSIA_WR_1

Address Offset	0x0000 00DC		
Physical Address	See Table 12-489.	**Instance**	ISP_SBL
Description	CSIA WRITE REQUEST 2 REGISTER		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								BYTE_CNT								DATA_READY	DATA_SENT	ADDR													

Bits	Field Name	Description	Type	Reset
31:30	RESERVED	Write 0s for future compatibility. Reads return 0.	R	0x0
29:22	BYTE_CNT	Current byte count.	R	0x00
21	DATA_READY	Data ready 0x0: No 0x1: Yes	R	0x0
20	DATA_SENT	Data sent to the destination, waiting for status. 0x0: No 0x1: Yes	R	0x0
19:0	ADDR	Upper 20 bits of the write address.	R	0x00000

Table 12-601. Register Call Summary for Register SBL_CSIA_WR_1

Camera ISP Register Manual

- Camera ISP_SBL Register Summary: [0]

Table 12-602. SBL_CSIA_WR_2

Address Offset	0x0000 00E0		
Physical Address	See Table 12-489.	**Instance**	ISP_SBL
Description	CSIA WRITE REQUEST 3 REGISTER		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								BYTE_CNT								DATA_READY	DATA_SENT	ADDR													

Bits	Field Name	Description	Type	Reset
31:30	RESERVED	Write 0s for future compatibility. Reads return 0.	R	0x0
29:22	BYTE_CNT	Current byte count	R	0x00
21	DATA_READY	Data ready Read 0x0: No Read 0x1: Yes	R	0
20	DATA_SENT	Data sent to the destination, waiting for status Read 0x0: No Read 0x1: Yes	R	0

Bits	Field Name	Description	Type	Reset
19:0	ADDR	Upper 20 bits of the write address	R	0x00000

Table 12-603. Register Call Summary for Register SBL_CSIA_WR_2

Camera ISP Register Manual

- Camera ISP_SBL Register Summary: [0]

Table 12-604. SBL_CSIA_WR_3

Address Offset	0x0000 00E4		
Physical Address	See Table 12-489.	**Instance**	ISP_SBL
Description	CSIA WRITE REQUEST 4 REGISTER		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								BYTE_CNT								DATA_READY	DATA_SENT	ADDR													

Bits	Field Name	Description	Type	Reset
31:30	RESERVED	Write 0s for future compatibility. Reads return 0.	R	0x0
29:22	BYTE_CNT	Current byte count	R	0x00
21	DATA_READY	Data ready Read 0x0: No Read 0x1: Yes	R	0
20	DATA_SENT	Data sent to the destination, waiting for status Read 0x0: No Read 0x1: Yes	R	0
19:0	ADDR	Upper 20 bits of the write address	R	0x00000

Table 12-605. Register Call Summary for Register SBL_CSIA_WR_3

Camera ISP Register Manual

- Camera ISP_SBL Register Summary: [0]

Table 12-606. SBL_CSIB_WR_0

Address Offset	0x0000 00E8		
Physical Address	See Table 12-489.	**Instance**	ISP_SBL
Description	CSIB WRITE REQUEST 1 REGISTER		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								BYTE_CNT								DATA_READY	DATA_SENT	ADDR													

Bits	Field Name	Description	Type	Reset
31:30	RESERVED	Write 0s for future compatibility. Reads return 0.	R	0x0
29:22	BYTE_CNT	Current byte count.	R	0x00
21	DATA_READY	Data ready 0x0: No 0x1: Yes	R	0x0
20	DATA_SENT	Data sent to the destination, waiting for status. 0x0: No 0x1: Yes	R	0x0
19:0	ADDR	Upper 20 bits of the write address.	R	0x00000

Table 12-607. Register Call Summary for Register SBL_CSIB_WR_0

Camera ISP Register Manual

- Camera ISP_SBL Register Summary: [0]

Table 12-608. SBL_CSIB_WR_1

Address Offset	0x0000 00EC		
Physical Address	See Table 12-489.	**Instance**	ISP_SBL
Description	CSIB WRITE REQUEST 2 REGISTER		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								BYTE_CNT								DATA_READY	DATA_SENT	ADDR													

Bits	Field Name	Description	Type	Reset
31:30	RESERVED	Write 0s for future compatibility. Reads return 0.	R	0x0
29:22	BYTE_CNT	Current byte count.	R	0x00
21	DATA_READY	Data ready 0x0: No 0x1: Yes	R	0x0
20	DATA_SENT	Data sent to the destination, waiting for status. 0x0: No 0x1: Yes	R	0x0
19:0	ADDR	Upper 20 bits of the write address.	R	0x00000

Table 12-609. Register Call Summary for Register SBL_CSIB_WR_1

Camera ISP Register Manual

- Camera ISP_SBL Register Summary: [0]

Table 12-610. SBL_CSIB_WR_2

Address Offset	0x0000 00F0		
Physical Address	See Table 12-489.	**Instance**	ISP_SBL
Description	CSIB WRITE REQUEST 3 REGISTER		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								BYTE_CNT								DATA_READY	DATA_SENT	ADDR													

Bits	Field Name	Description	Type	Reset
31:30	RESERVED	Write 0s for future compatibility. Reads return 0.	R	0x0
29:22	BYTE_CNT	Current byte count.	R	0x00
21	DATA_READY	Data ready 0x0: No 0x1: Yes	R	0x0
20	DATA_SENT	Data sent to the destination, waiting for status. 0x0: No 0x1: Yes	R	0x0
19:0	ADDR	Upper 20 bits of the write address.	R	0x00000

Table 12-611. Register Call Summary for Register SBL_CSIB_WR_2

Camera ISP Register Manual

- Camera ISP_SBL Register Summary: [0]

Table 12-612. SBL_CSIB_WR_3

Address Offset	0x0000 00F4		
Physical Address	See Table 12-489.	**Instance**	ISP_SBL
Description	CSIB WRITE REQUEST 4 REGISTER		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								BYTE_CNT								DATA_READY	DATA_SENT	ADDR													

Bits	Field Name	Description	Type	Reset
31:30	RESERVED	Write 0s for future compatibility. Reads return 0.	R	0x0
29:22	BYTE_CNT	Current byte count.	R	0x00
21	DATA_READY	Data ready 0x0: No 0x1: Yes	R	0x0
20	DATA_SENT	Data sent to the destination, waiting for status. 0x0: No 0x1: Yes	R	0x0

Bits	Field Name	Description	Type	Reset
19:0	ADDR	Upper 20 bits of the write address.	R	0x00000

Table 12-613. Register Call Summary for Register SBL_CSIB_WR_3

Camera ISP Register Manual
- Camera ISP_SBL Register Summary: [0]

Table 12-614. SBL_SDR_REQ_EXP

Address Offset	0x0000 00F8		
Physical Address	See Table 12-489.	**Instance**	ISP_SBL
Description	MEMORY NON REAL TIME READ REQUEST EXPAND		
Type	RW		

31 30	29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10	9 8	7 6 5 4 3 2 1 0
RESERVED	PRV_EXP	RSZ_EXP	HIST_EXP		

Bits	Field Name	Description	Type	Reset
31:30	RESERVED		RW	0x0
29:20	PRV_EXP	PREVIEW module READ request expand Sets the number of clock cycles (func clock cycles) to allow between two consecutive READ requests from the module. It spreads non-real time read requests over time. It enables less priority in the system not to be locked out. At maximum, the PRV can read 256 bytes every 1*PRV_EXP functional clock cycles.	RW	0x000
19:10	RSZ_EXP	RESIZER module READ request expand Sets the number of clock cycles (func clock cycles) to allow between two consecutive READ requests from the module. It spreads non-real time read requests over time. It enables less priority in the system not to be locked out. At maximum, the RSZ can read 256 bytes every 1024*RSZ_EXP functional clock cycles.	RW	0x000
9:0	HIST_EXP	HISTOGRAM module READ request expand Sets the number of clock cycles (func clock cycles) to allow between two consecutive READ requests from the module. It spreads non-real time read requests over time. It enables less priority in the system not to be locked out. At maximum, the HIST can read 256 bytes every HIST_EXP functional clock cycles	RW	0x000

Table 12-615. Register Call Summary for Register SBL_SDR_REQ_EXP

Camera ISP Basic Programming Model
- Events and Status Checking: [0]
- Hardware Setup/Initialization: [1]
- Camera ISP Bandwidth Adjustments: [2] [3] [4] [5]

Camera ISP Register Manual
- Camera ISP_SBL Register Summary: [6]

12.6.11 Camera ISP_CSI2A Registers

12.6.11.1 Camera ISP_CSI2A Register Summary

Table 12-616. ISP_CSI2A Register Summary

Register Name	Type	Register Width (Bits)	Address Offset	Physical Address
CSI2_REVISION	R	32	0x0000 0000	0x480B D800
CSI2_SYSCONFIG	RW	32	0x0000 0010	0x480B D810
CSI2_SYSSTATUS	R	32	0x0000 0014	0x480B D814
CSI2_IRQSTATUS	RW	32	0x0000 0018	0x480B D818
CSI2_IRQENABLE	RW	32	0x0000 001C	0x480B D81C
CSI2_CTRL	RW	32	0x0000 0040	0x480B D840
CSI2_DBG_H	W	32	0x0000 0044	0x480B D844
CSI2_GNQ	R	32	0x0000 0048	0x480B D848
CSI2_COMPLEXIO_CFG 1	RW	32	0x0000 0050	0x480B D850
CSI2_COMPLEXIO1_IRQ STATUS	RW	32	0x0000 0054	0x480B D854
CSI2_SHORT_PACKET	R	32	0x0000 005C	0x480B D85C
CSI2_COMPLEXIO1_IRQ ENABLE	RW	32	0x0000 0060	0x480B D860
CSI2_DBG_P	W	32	0x0000 0068	0x480B D868
CSI2_TIMING	RW	32	0x0000 006C	0x480B D86C
CSI2_CTx_CTRL1	RW	32	0x0000 0070 + (x*0x20)	0x480B D870 + (x*0x20)
CSI2_CTx_CTRL2	RW	32	0x0000 0074 + (x*0x20)	0x480B D874 + (x*0x20)
CSI2_CTx_DAT_OFST	RW	32	0x0000 0078 + (x*0x20)	0x480B D878 + (x*0x20)
CSI2_CTx_DAT_PING_A DDR	RW	32	0x0000 007C + (x*0x20)	0x480B D87C + (x*0x20)
CSI2_CTx_DAT_PONG_ ADDR	RW	32	0x0000 0080 + (x*0x20)	0x480B D880 + (x*0x20)
CSI2_CTx_IRQENABLE	RW	32	0x0000 0084 + (x*0x20)	0x480B D884 + (x*0x20)
CSI2_CTx_IRQSTATUS	RW	32	0x0000 0088 + (x*0x20)	0x480B D888 + (x*0x20)
CSI2_CTx_CTRL3	RW	32	0x0000 008C + (x*0x20)	0x480B D88C + (x*0x20)

12.6.11.2 Camera ISP_CSI2A Register Description

Table 12-617. CSI2_REVISION

Address Offset	0x0000 0000		
Physical Address	0x480B D800	**Instance**	ISP_CSI2A
Description	MODULE REVISION This register contains the IP revision code in binary coded digital. For example, we have: 0x01 = revision 0.1 and 0x21 = revision 2.1		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																								REV							

Bits	Field Name	Description	Type	Reset
31:8	RESERVED	Write 0s for future compatibility. Reads return 0.	R	0x000000
7:0	REV	IP revision [7:4] Major revision [3:0] Minor revision	R	TI internal data

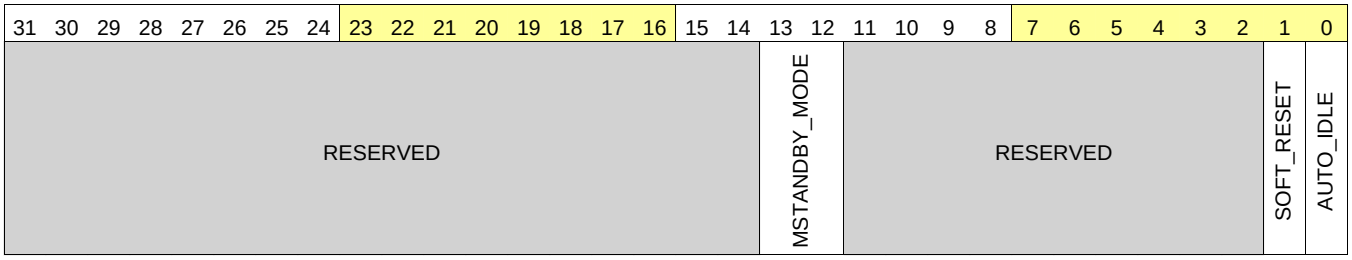
Table 12-618. Register Call Summary for Register CSI2_REVISION

Camera ISP Register Manual

- Camera ISP_CSI2A Register Summary: [0]

Table 12-619. CSI2_SYSCONFIG

Address Offset	0x0000 0010		
Physical Address	0x480B D810	**Instance**	ISP_CSI2A
Description	SYSTEM CONFIGURATION REGISTER This register is the OCP-socket system configuration register.		
Type	RW		



Bits	Field Name	Description	Type	Reset
31:14	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x00000
13:12	MSTANDBY_MODE	Sets the behavior of the master port power management signals. 0x0: Force-standby. Hardware signal is only asserted when the module is disabled. 0x1: No-standby. Hardware signal is never asserted. 0x2: Smart-standby: Hardware signal is asserted based on the activity of the module. The module tries to go to standby during the vertical blanking period.	RW	0x0
11:2	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x000
1	SOFT_RESET	Software reset. Set the bit to 1 to trigger a module reset. The bit is automatically reset by the HW. During reads return 0. 0x0: Normal mode. 0x1: The module is reset.(1)	RW	0x0
0	AUTO_IDLE	Internal OCP gating strategy 0x0: OCP clock is free-running. 0x1: Automatic OCP clock gating strategy is applied based on the OCP interface activity.	RW	0x1

(1) Before setting the software reset bit to 1 in the CSI2_SYSCONFIG register, the user must have access to a CSI2 receiver register.

Table 12-620. Register Call Summary for Register CSI2_SYSCONFIG

Camera ISP Integration

- Power Management: [0] [1]
- Resets: [2]

Camera ISP Basic Programming Model

- Reset Management: [3]
- Enable Video/Picture Acquisition: [4] [5]

Camera ISP Register Manual

- Camera ISP_CSI2A Register Summary: [6]

Table 12-621. CSI2_SYSSTATUS

Address Offset	0x0000 0014		
Physical Address	0x480B D814	**Instance**	ISP_CSI2A
Description	SYSTEM STATUS REGISTER This register provides status information about the module, excluding the interrupt status register.		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																															RESET_DONE

Bits	Field Name	Description	Type	Reset
31:1	RESERVED	Reads return 0.	R	0x00000000
0	RESET_DONE	Internal reset monitoring 0x0: Internal module reset is on going. 0x1: Reset completed.	R	0x1

Table 12-622. Register Call Summary for Register CSI2_SYSSTATUS

Camera ISP Integration
- Resets: [0] [1]

Camera ISP Register Manual
- Camera ISP_CSI2A Register Summary: [2]

Table 12-623. CSI2_IRQSTATUS

Address Offset	0x0000 0018		
Physical Address	0x480B D818	**Instance**	ISP_CSI2A
Description	INTERRUPT STATUS REGISTER - All contexts This register associates one bit for each context to determine which context has generated the interrupt. The context must be enabled for events to be generated on that context. If the context is disabled, the interrupt is not generated.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																OCP_ERR_IRQ	SHORT_PACKET_IRQ	ECC_CORRECTION_IRQ	ECC_NO_CORRECTION_IRQ	RESERVED	COMPLEXIO1_ERR_IRQ	FIFO_OVF_IRQ	CONTEXT7	CONTEXT6	CONTEXT5	CONTEXT4	CONTEXT3	CONTEXT2	CONTEXT1	CONTEXT0

Bits	Field Name	Description	Type	Reset
31:15	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x00000
14	OCP_ERR_IRQ	OCP Error Interrupt 0x0: READS: Event is false. WRITES: Status bit unchanged. 0x1: READS: Event is true (pending). WRITES: Status bit is reset.	RW	0x0
13	SHORT_PACKET_IRQ	Short packet reception status (other than sync events: Line Start, Line End, Frame Start, and Frame End: data type between 0x8 and x0F only must be considered). 0x0: READS: Event is false. WRITES: Status bit unchanged. 0x1: READS: Event is true (pending). WRITES: Status bit is reset.	RW	0x0
12	ECC_CORRECTION_IRQ	ECC has been used to do the correction of the only 1-bit error status (short packet only). 0x0: READS: Event is false. WRITES: Status bit unchanged. 0x1: READS: Event is true (pending). WRITES: Status bit is reset.	RW	0x0
11	ECC_NO_CORRECTION_IRQ	ECC error status (short and long packets). No correction of the header because of more than 1-bit error. 0x0: READS: Event is false. WRITES: Status bit unchanged. 0x1: READS: Event is true (pending). WRITES: Status bit is reset.	RW	0x0
10	RESERVED	Reserved	R	0x0
9	COMPLEXIO1_ERR_IRQ	Error signaling from Complex I/O #1: status of the PHY errors received from the complex I/O #1 (events are defined in CSI2_COMPLEXIO1_IRQSTATUS for the 1st complex I/O). 0x0: READS: Event is false. 0x1: READS: Event is true (pending).	R	0x0
8	FIFO_OVF_IRQ	FIFO overflow error status. 0x0: READS: Event is false. WRITES: Status bit unchanged. 0x1: READS: Event is true (pending). WRITES: Status bit is reset.	RW	0x0
7	CONTEXT7	Context 7 0x0: READS: Event is false. 0x1: READS: Event is true (pending).	R	0x0
6	CONTEXT6	Context 6 0x0: READS: Event is false. 0x1: READS: Event is true (pending).	R	0x0
5	CONTEXT5	Context 5 0x0: READS: Event is false. 0x1: READS: Event is true (pending).	R	0x0
4	CONTEXT4	Context 4 0x0: READS: Event is false. 0x1: READS: Event is true (pending).	R	0x0
3	CONTEXT3	Context 3 0x0: READS: Event is false. 0x1: READS: Event is true (pending).	R	0x0

Bits	Field Name	Description	Type	Reset
2	CONTEXT2	Context 2 0x0: READS: Event is false. 0x1: READS: Event is true (pending).	R	0x0
1	CONTEXT1	Context 1 0x0: READS: Event is false. 0x1: READS: Event is true (pending).	R	0x0
0	CONTEXT0	Context 0 0x0: READS: Event is false. 0x1: READS: Event is true (pending).	R	0x0

Table 12-624. Register Call Summary for Register CSI2_IRQSTATUS

Camera ISP Integration • Interrupt Requests: [0]
Camera ISP Functional Description • ECC and Checksum Generation: [1] [2] [3] • Short Packet: [4]
Camera ISP Basic Programming Model • Enable Video/Picture Acquisition: [5]
Camera ISP Register Manual • Camera ISP_CSI2A Register Summary: [6]

Table 12-625. CSI2_IRQENABLE

Address Offset	0x0000 001C		
Physical Address	0x480B D81C	**Instance**	ISP_CSI2A
Description	INTERRUPT ENABLE REGISTER - All contexts This register associates one bit for each context to enable/disable each context individually.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																OCP_ERR_IRQ	SHORT_PACKET_IRQ	ECC_CORRECTION_IRQ	ECC_NO_CORRECTION_IRQ	RESERVED	COMPLEXIO1_ERR_IRQ	FIFO_OVF_IRQ	CONTEXT7	CONTEXT6	CONTEXT5	CONTEXT4	CONTEXT3	CONTEXT2	CONTEXT1	CONTEXT0	

Bits	Field Name	Description	Type	Reset
31:15	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x00000
14	OCP_ERR_IRQ	OCP Error Interrupt 0x0: Event is masked 0x1: Event generates an interrupt when it occurs	RW	0x0
13	SHORT_PACKET_IRQ	Short packet reception (other than sync events: Line Start, Line End, Frame Start, and Frame End: data type between 0x8 and x0F only must be considered). 0x0: Event is masked 0x1: Event generates an interrupt when it occurs	RW	0x0

Bits	Field Name	Description	Type	Reset
12	ECC_CORRECTION_IRQ	ECC has been used to correct the only 1-bit error (short packet only). 0x0: Event is masked 0x1: Event generates an interrupt when it occurs	RW	0x0
11	ECC_NO_CORRECTION_IRQ	ECC error (short and long packets). No correction of the header because of more than 1-bit error. 0x0: Event is masked 0x1: Event generates an interrupt when it occurs	RW	0x0
10	RESERVED	Reserved	RW	0x0
9	COMPLEXIO1_ERR_IRQ	Error signaling from Complex I/O #1: the interrupt is triggered when any error is received from the complex I/O #1 (events are defined in CSI2_COMPLEXIO1_IRQSTATUS for the 1st complex I/O). 0x0: Event is masked 0x1: Event generates an interrupt when it occurs	RW	0x0
8	FIFO_OVF_IRQ	FIFO overflow enable 0x0: Event is masked 0x1: Event generates an interrupt when it occurs	RW	0x0
7	CONTEXT7	Context 7 0x0: Event is masked 0x1: Event generates an interrupt when it occurs	RW	0x0
6	CONTEXT6	Context 6 0x0: Event is masked 0x1: Event generates an interrupt when it occurs	RW	0x0
5	CONTEXT5	Context 5 0x0: Event is masked 0x1: Event generates an interrupt when it occurs	RW	0x0
4	CONTEXT4	Context 4 0x0: Event is masked 0x1: Event generates an interrupt when it occurs	RW	0x0
3	CONTEXT3	Context 3 0x0: Event is masked 0x1: Event generates an interrupt when it occurs	RW	0x0
2	CONTEXT2	Context 2 0x0: Event is masked 0x1: Event generates an interrupt when it occurs	RW	0x0
1	CONTEXT1	Context 1 0x0: Event is masked 0x1: Event generates an interrupt when it occurs	RW	0x0
0	CONTEXT0	Context 0 0x0: Event is masked 0x1: Event generates an interrupt when it occurs	RW	0x0

Table 12-626. Register Call Summary for Register CSI2_IRQENABLE

Camera ISP Functional Description • ECC and Checksum Generation: [0] • Short Packet: [1]
Camera ISP Basic Programming Model • Enable Video/Picture Acquisition: [2] [3]

Table 12-626. Register Call Summary for Register CSI2_IRQENABLE (continued)

Camera ISP Register Manual

- Camera ISP_CSI2A Register Summary: [4]

Table 12-627. CSI2_CTRL

Address Offset	0x0000 0040		
Physical Address	0x480B D840	**Instance**	ISP_CSI2A
Description	GLOBAL CONTROL REGISTER This register controls the CSI2 RECEIVER module. This register must not be modified dynamically (except IF_EN bit field).		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																VP_CLK_EN	RESERVED	RESERVED	RESERVED	VP_ONLY_EN	RESERVED	VP_OUT_CTRL		DBG_EN	RESERVED		RESERVED	FRAME	ECC_EN	RESERVED	IF_EN

Bits	Field Name	Description	Type	Reset
31:16	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x0000
15	VP_CLK_EN	VP clock enable. 0x0: The VP clock is disabled. 0x1: The VP clock is enabled.	RW	0x0
14	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x0
13	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x0
12	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x0
11	VP_ONLY_EN	VP only enable. 0x0: The VP is enabled and the OCP master port is enabled. 0x1: The VP is enabled and the OCP master port is disabled.	RW	0x0
10	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x0
9:8	VP_OUT_CTRL	Video port output clock control. Sets the video port output clock as a function of the interface clock (OCPCLK). 0x0: No division: video port clock = OCPCLK. 0x1: Division by 2: video port clock = OCPCLK/2. 0x2: Division by 3: video port clock = OCPCLK/3. 0x3: Division by 4: video port clock = OCPCLK/4.	RW	0x1
7	DBG_EN	Enables the debug mode. 0x0: Disable 0x1: Enable	RW	0x0
6:5	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x0
4	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x0

Bits	Field Name	Description	Type	Reset
3	FRAME	Set the modality in which IF_EN works. 0x0: If IF_EN = 0 the interface is disabled immediately. 0x1: If IF_EN = 1 the interface is disabled after all FEC sync code have been received for the active contexts.	RW	0x0
2	ECC_EN	Enables the Error Correction Code check for the received header (short and long packets for all virtual channel ids). 0x0: Disabled 0x1: Enabled	RW	0x0
1	RESERVED	RESERVED	RW	0x0
0	IF_EN	Enables the physical interface to the module. 0x0: The interface is disabled. If FRAME = 0, it is disabled immediately. If FRAME = 1, it is disabled when each context has received the FEC sync code. 0x1: The interface is enabled immediately, the data acquisition starts on the next FSC sync code. Writing 1 to this register when the current value is 0 has the effect to clear the output FIFO. The pixel data of the following frame are written in the PING buffer, that is, the CSI2_CTx_CTRL1.PING_PONG bits are reset to 0 as well.	RW	0x0

Table 12-628. Register Call Summary for Register CSI2_CTRL

Camera ISP Integration • Power Management: [0]
Camera ISP Functional Description • ECC and Checksum Generation: [1] • DMA Engine: [2] [3]
Camera ISP Basic Programming Model • Enable Video/Picture Acquisition: [4] [5] • Disable Video/Picture Acquisition: [6] [7] [8]
Camera ISP Register Manual • Camera ISP_CSI2A Register Summary: [9] • Camera ISP_CSI2A Register Description: [10] [11] [12]

Table 12-629. CSI2_DBG_H

Address Offset	0x0000 0044		
Physical Address	0x480B D844	**Instance**	ISP_CSI2A
Description	DEBUG REGISTER (Header) This register provides a way to debug the CSI2 RECEIVER module with no image sensor connected to the module. The debug mode is enabled by CSI2_CTRL.DBG_EN. Only full 32-bit values must be written. The register is used to write short packets and header of long packets.		
Type	W		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DBG																															

Bits	Field Name	Description	Type	Reset
31:0	DBG	32-bit input value.	W	0x00000000

Table 12-630. Register Call Summary for Register CSI2_DBG_H

Camera ISP Register Manual • Camera ISP_CSI2A Register Summary: [0]

Table 12-631. CSI2_GNQ

Address Offset	0x0000 0048		
Physical Address	0x480B D848	**Instance**	ISP_CSI2A
Description	GENERIC PARAMETER REGISTER This register provide a way to read the generic parameters used in the design.		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																								FIFODEPTH						NBCONTEXTS	

Bits	Field Name	Description	Type	Reset
31:6	RESERVED	Write 0s for future compatibility. Reads return 0.	R	0x0000000
5:2	FIFODEPTH	Output FIFO size in multiple of 68 bits. 0x2: 8x 68 bits 0x3: 16x 68 bits 0x4: 32x 68 bits 0x5: 64x 68 bits 0x6: 128 x 68 bits 0x7: 256 x 68 bits	R	0x2
1:0	NBCONTEXTS	Number of contexts supported by the module. 0x0: 1 Context 0x1: 2 Contexts 0x2: 4 Contexts 0x3: 8 Contexts	R	0x3

Table 12-632. Register Call Summary for Register CSI2_GNQ

Camera ISP Register Manual

- Camera ISP_CSI2A Register Summary: [0]

Table 12-633. CSI2_COMPLEXIO_CFG1

Address Offset	0x0000 0050		
Physical Address	0x480B D850	**Instance**	ISP_CSI2A
Description	COMPLEXIO CONFIGURATION REGISTER for the complex I/O #1 This register contains the lane configuration for the order and position of the lanes (clock and data) and the polarity order for the control of the PHY differential signals in addition to the control bit for the power FSM.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED		RESET_DONE	PWR_CMD		PWR_STATUS		PWR_AUTO	RESERVED												DATA2_POL	DATA2_POSITION			DATA1_POL	DATA1_POSITION			CLOCK_POL	CLOCK_POSITION		

Bits	Field Name	Description	Type	Reset
31:30	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x0
29	RESET_DONE	Internal reset monitoring of the power domain using the PPI byte clock from complex I/O 0x0: Internal module reset is on-going. 0x1: Reset completed.	R	0x0
28:27	PWR_CMD	Command for power control of complex I/O 0x0: Command to change to OFF state 0x1: Command to change to ON state 0x2: Command to change to ultra-low power state	RW	0x0
26:25	PWR_STATUS	Status of the power control of complex I/O 0x0: Complex I/O in OFF state 0x1: Complex I/O in ON state 0x2: Complex I/O in ultra-low power state	R	0x0
24	PWR_AUTO	Automatic switch between ULP and ON states based on ULPM signals from complex I/O 0x0: Disable 0x1: Enable	RW	0x0
23:12	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x0
11	DATA2_POL	+/- differential pin order of data lane 2. 0x0: +/- pin order (csi2_dx=+ and csi2_dy=-) 0x1: -/+ pin order (csi2_dx=- and csi2_dy=+)	RW	0x0
10:8	DATA2_POSITION	Position and order of data lane 2. 0x0: Not used/connected 0x1: Data lane 2 is at position 1. 0x2: Data lane 2 is at position 2. 0x3: Data lane 2 is at position 3.	RW	0x0
7	DATA1_POL	+/- differential pin order of data lane 1. 0x0: +/- pin order (csi2_dx=+ and csi2_dy=-) 0x1: -/+ pin order (csi2_dx=- and csi2_dy=+)	RW	0x0
6:4	DATA1_POSITION	Position and order of data lane 1. Data lane 1 is always present. 0x0: Not used/connected 0x1: Data lane 1 is at position 1. 0x2: Data lane 1 is at position 2. 0x3: Data lane 1 is at position 3.	RW	0x0
3	CLOCK_POL	+/- differential pin order of the clock lane. 0x0: +/- pin order (csi2_dx=+ and csi2_dy=-) 0x1: -/+ pin order (csi2_dx=- and csi2_dy=+)	RW	0x0
2:0	CLOCK_POSITION	Position and order of the clock lane. The clock lane is always present. 0x0: Not used/connected 0x1: Clock lane is at position 1. 0x2: Clock lane is at position 2. 0x3: Clock lane is at position 3.	RW	0x0

Table 12-634. Register Call Summary for Register CSI2_COMPLEXIO_CFG1

Camera ISP Functional Description

- Physical Layer Lane Configuration: [0] [1]
- Complex I/O: [2] [3]

Table 12-634. Register Call Summary for Register CSI2_COMPLEXIO_CFG1 (continued)

Camera ISP Basic Programming Model • Enable Video/Picture Acquisition: [4] [5]
Camera ISP Register Manual • Camera ISP_CSI2A Register Summary: [6]

Table 12-635. CSI2_COMPLEXIO1_IRQSTATUS

Address Offset	0x0000 0054		
Physical Address	0x480B D854	**Instance**	ISP_CSI2A
Description	INTERRUPT STATUS REGISTER - All errors from complex I/O #1		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED					STATEALLULPMEXIT	STATEALLULPMENTER	STATEULPM5	STATEULPM4	STATEULPM3	STATEULPM2	STATEULPM1	ERRCONTROL5	ERRCONTROL4	ERRCONTROL3	ERRCONTROL2	ERRCONTROL1	ERRESC5	ERRESC4	ERRESC3	ERRESC2	ERRESC1	ERRSOTSYNCHS5	ERRSOTSYNCHS4	ERRSOTSYNCHS3	ERRSOTSYNCHS2	ERRSOTSYNCHS1	ERRSOTHS5	ERRSOTHS4	ERRSOTHS3	ERRSOTHS2	ERRSOTHS1

Bits	Field Name	Description	Type	Reset
31:27	RESERVED	Write 0s for future compatibility. Reads return 0.	R	0x00
26	STATEALLULPMEXIT	At least one of the active lanes has exit the ULPM 0x0: READS: Event is false. WRITES: Status bit unchanged. 0x1: READS: Event is true (pending). WRITES: Status bit is reset.	RW W1toClr	0x0
25	STATEALLULPMENTER	All active lanes are entering in ULPM. 0x0: READS: Event is false. WRITES: Status bit unchanged. 0x1: READS: Event is true (pending). WRITES: Status bit is reset.	RW W1toClr	0x0
24	STATEULPM5	Lane #5 in Ultra Low Power Mode 0x0: READS: Event is false. WRITES: Status bit unchanged. 0x1: READS: Event is true (pending). WRITES: Status bit is reset.	RW W1toClr	0x0
23	STATEULPM4	Lane #4 in Ultra Low Power Mode 0x0: READS: Event is false. WRITES: Status bit unchanged. 0x1: READS: Event is true (pending). WRITES: Status bit is reset.	RW W1toClr	0x0
22	STATEULPM3	Lane #3 in Ultra Low Power Mode 0x0: READS: Event is false. WRITES: Status bit unchanged. 0x1: READS: Event is true (pending). WRITES: Status bit is reset.	RW W1toClr	0x0

Bits	Field Name	Description	Type	Reset
21	STATEULPM2	Lane #2 in Ultra Low Power Mode 0x0: READS: Event is false. WRITES: Status bit unchanged. 0x1: READS: Event is true (pending). WRITES: Status bit is reset.	RW W1toClr	0x0
20	STATEULPM1	Lane #1 in Ultra Low Power Mode 0x0: READS: Event is false. WRITES: Status bit unchanged. 0x1: READS: Event is true (pending). WRITES: Status bit is reset.	RW W1toClr	0x0
19	ERRCONTROL5	Control error for lane #5 0x0: READS: Event is false. WRITES: Status bit unchanged. 0x1: READS: Event is true (pending). WRITES: Status bit is reset.	RW W1toClr	0x0
18	ERRCONTROL4	Control error for lane #4 0x0: READS: Event is false. WRITES: Status bit unchanged. 0x1: READS: Event is true (pending). WRITES: Status bit is reset.	RW W1toClr	0x0
17	ERRCONTROL3	Control error for lane #3 0x0: READS: Event is false. WRITES: Status bit unchanged. 0x1: READS: Event is true (pending). WRITES: Status bit is reset.	RW W1toClr	0x0
16	ERRCONTROL2	Control error for lane #2 0x0: READS: Event is false. WRITES: Status bit unchanged. 0x1: READS: Event is true (pending). WRITES: Status bit is reset.	RW W1toClr	0x0
15	ERRCONTROL1	Control error for lane #1 0x0: READS: Event is false. WRITES: Status bit unchanged. 0x1: READS: Event is true (pending). WRITES: Status bit is reset.	RW W1toClr	0x0
14	ERRESC5	Escape entry error for lane #5 0x0: READS: Event is false. WRITES: Status bit unchanged. 0x1: READS: Event is true (pending). WRITES: Status bit is reset.	RW W1toClr	0x0
13	ERRESC4	Escape entry error for lane #4 0x0: READS: Event is false. WRITES: Status bit unchanged. 0x1: READS: Event is true (pending). WRITES: Status bit is reset.	RW W1toClr	0x0
12	ERRESC3	Escape entry error for lane #3 0x0: READS: Event is false. WRITES: Status bit unchanged. 0x1: READS: Event is true (pending). WRITES: Status bit is reset.	RW W1toClr	0x0
11	ERRESC2	Escape entry error for lane #2 0x0: READS: Event is false. WRITES: Status bit unchanged. 0x1: READS: Event is true (pending). WRITES: Status bit is reset.	RW W1toClr	0x0

Bits	Field Name	Description	Type	Reset
10	ERRESC1	Escape entry error for lane #1 0x0: READS: Event is false. WRITES: Status bit unchanged. 0x1: READS: Event is true (pending). WRITES: Status bit is reset.	RW W1toClr	0x0
9	ERRSOTSYNCHS5	Start of transmission sync error for lane #5 0x0: READS: Event is false. WRITES: Status bit unchanged. 0x1: READS: Event is true (pending). WRITES: Status bit is reset.	RW W1toClr	0x0
8	ERRSOTSYNCHS4	Start of transmission sync error for lane #4 0x0: READS: Event is false. WRITES: Status bit unchanged. 0x1: READS: Event is true (pending). WRITES: Status bit is reset.	RW W1toClr	0x0
7	ERRSOTSYNCHS3	Start of transmission sync error for lane #3 0x0: READS: Event is false. WRITES: Status bit unchanged. 0x1: READS: Event is true (pending). WRITES: Status bit is reset.	RW W1toClr	0x0
6	ERRSOTSYNCHS2	Start of transmission sync error for lane #2 0x0: READS: Event is false. WRITES: Status bit unchanged. 0x1: READS: Event is true (pending). WRITES: Status bit is reset.	RW W1toClr	0x0
5	ERRSOTSYNCHS1	Start of transmission sync error for lane #1 0x0: READS: Event is false. WRITES: Status bit unchanged. 0x1: READS: Event is true (pending). WRITES: Status bit is reset.	RW W1toClr	0x0
4	ERRSOTHS5	Start of transmission error for lane #5 0x0: READS: Event is false. WRITES: Status bit unchanged. 0x1: READS: Event is true (pending). WRITES: Status bit is reset.	RW W1toClr	0x0
3	ERRSOTHS4	Start of transmission error for lane #4 0x0: READS: Event is false. WRITES: Status bit unchanged. 0x1: READS: Event is true (pending). WRITES: Status bit is reset.	RW W1toClr	0x0
2	ERRSOTHS3	Start of transmission error for lane #3 0x0: READS: Event is false. WRITES: Status bit unchanged. 0x1: READS: Event is true (pending). WRITES: Status bit is reset.	RW W1toClr	0x0
1	ERRSOTHS2	Start of transmission error for lane #2 0x0: READS: Event is false. WRITES: Status bit unchanged. 0x1: READS: Event is true (pending). WRITES: Status bit is reset.	RW W1toClr	0x0
0	ERRSOTHS1	Start of transmission error for lane #1 0x0: READS: Event is false. WRITES: Status bit unchanged. 0x1: READS: Event is true (pending). WRITES: Status bit is reset.	RW W1toClr	0x0

Table 12-636. Register Call Summary for Register CSI2_COMPLEXIO1_IRQSTATUS

Camera ISP Integration • Interrupt Requests: [0]
Camera ISP Functional Description • Complex I/O: [1]
Camera ISP Basic Programming Model • Enable Video/Picture Acquisition: [2]
Camera ISP Register Manual • Camera ISP_CSI2A Register Summary: [3] • Camera ISP_CSI2A Register Description: [4] [5]

Table 12-637. CSI2_SHORT_PACKET

Address Offset	0x0000 005C		
Physical Address	0x480B D85C	**Instance**	ISP_CSI2A
Description	SHORT PACKET INFORMATION - This register sets the 24-bit DATA_ID + Short Packet Data Field when the data type is between 0x8 and x0F		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								SHORT_PACKET																							

Bits	Field Name	Description	Type	Reset
31:24	RESERVED	Reads return 0.	R	0x00
23:0	SHORT_PACKET	Short Packet information: DATA ID + DATA FIELD	R	0x000000

Table 12-638. Register Call Summary for Register CSI2_SHORT_PACKET

Camera ISP Functional Description • Short Packet: [0] [1]
Camera ISP Register Manual • Camera ISP_CSI2A Register Summary: [2]

Table 12-639. CSI2_COMPLEXIO1_IRQENABLE

Address Offset	0x0000 0060		
Physical Address	0x480B D860	**Instance**	ISP_CSI2A
Description	INTERRUPT ENABLE REGISTER - All errors from complex I/O #1		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED					STATEALLULPMEXIT	STATEALLULPMENTER	STATEULPM5	STATEULPM4	STATEULPM3	STATEULPM2	STATEULPM1	ERRCONTROL5	ERRCONTROL4	ERRCONTROL3	ERRCONTROL2	ERRCONTROL1	ERRESC5	ERRESC4	ERRESC3	ERRESC2	ERRESC1	ERRSOTSYNCHS5	ERRSOTSYNCHS4	ERRSOTSYNCHS3	ERRSOTSYNCHS2	ERRSOTSYNCHS1	ERRSOTHS5	ERRSOTHS4	ERRSOTHS3	ERRSOTHS2	ERRSOTHS1

Bits	Field Name	Description	Type	Reset
31:27	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x00
26	STATEALLULPMEXIT	At least one of the active lanes has exit the ULPM 0x0: Event is masked 0x1: Event generates an interrupt when it occurs	RW	0x0
25	STATEALLULPMENTER	All active lanes are entering in ULPM. 0x0: Event is masked 0x1: Event generates an interrupt when it occurs	RW	0x0
24	STATEULPM5	Lane #5 in Ultra Low Power Mode 0x0: Event is masked 0x1: Event generates an interrupt when it occurs	RW	0x0
23	STATEULPM4	Lane #4 in Ultra Low Power Mode 0x0: Event is masked 0x1: Event generates an interrupt when it occurs	RW	0x0
22	STATEULPM3	Lane #3 in Ultra Low Power Mode 0x0: Event is masked 0x1: Event generates an interrupt when it occurs	RW	0x0
21	STATEULPM2	Lane #2 in Ultra Low Power Mode 0x0: Event is masked 0x1: Event generates an interrupt when it occurs	RW	0x0
20	STATEULPM1	Lane #1 in Ultra Low Power Mode 0x0: Event is masked 0x1: Event generates an interrupt when it occurs	RW	0x0
19	ERRCONTROL5	Control error for lane #5 0x0: Event is masked 0x1: Event generates an interrupt when it occurs	RW	0x0
18	ERRCONTROL4	Control error for lane #4 0x0: Event is masked 0x1: Event generates an interrupt when it occurs	RW	0x0
17	ERRCONTROL3	Control error for lane #3 0x0: Event is masked 0x1: Event generates an interrupt when it occurs	RW	0x0
16	ERRCONTROL2	Control error for lane #2 0x0: Event is masked 0x1: Event generates an interrupt when it occurs	RW	0x0
15	ERRCONTROL1	Control error for lane #1 0x0: Event is masked 0x1: Event generates an interrupt when it occurs	RW	0x0
14	ERRESC5	Escape entry error for lane #5 0x0: Event is masked 0x1: Event generates an interrupt when it occurs	RW	0x0
13	ERRESC4	Escape entry error for lane #4 0x0: Event is masked 0x1: Event generates an interrupt when it occurs	RW	0x0
12	ERRESC3	Escape entry error for lane #3 0x0: Event is masked 0x1: Event generates an interrupt when it occurs	RW	0x0

Bits	Field Name	Description	Type	Reset
11	ERRESC2	Escape entry error for lane #2 0x0: Event is masked 0x1: Event generates an interrupt when it occurs	RW	0x0
10	ERRESC1	Escape entry error for lane #1 0x0: Event is masked 0x1: Event generates an interrupt when it occurs	RW	0x0
9	ERRSOTSYNCHS5	Start of transmission sync error for lane #5 0x0: Event is masked 0x1: Event generates an interrupt when it occurs	RW	0x0
8	ERRSOTSYNCHS4	Start of transmission sync error for lane #4 0x0: Event is masked 0x1: Event generates an interrupt when it occurs	RW	0x0
7	ERRSOTSYNCHS3	Start of transmission sync error for lane #3 0x0: Event is masked 0x1: Event generates an interrupt when it occurs	RW	0x0
6	ERRSOTSYNCHS2	Start of transmission sync error for lane #2 0x0: Event is masked 0x1: Event generates an interrupt when it occurs	RW	0x0
5	ERRSOTSYNCHS1	Start of transmission sync error for lane #1 0x0: Event is masked 0x1: Event generates an interrupt when it occurs	RW	0x0
4	ERRSOTHS5	Start of transmission error for lane #5 0x0: Event is masked 0x1: Event generates an interrupt when it occurs	RW	0x0
3	ERRSOTHS4	Start of transmission error for lane #4 0x0: Event is masked 0x1: Event generates an interrupt when it occurs	RW	0x0
2	ERRSOTHS3	Start of transmission error for lane #3 0x0: Event is masked 0x1: Event generates an interrupt when it occurs	RW	0x0
1	ERRSOTHS2	Start of transmission error for lane #2 0x0: Event is masked 0x1: Event generates an interrupt when it occurs	RW	0x0
0	ERRSOTHS1	Start of transmission error for lane #1 0x0: Event is masked 0x1: Event generates an interrupt when it occurs	RW	0x0

Table 12-640. Register Call Summary for Register CSI2_COMPLEXIO1_IRQENABLE

Camera ISP Basic Programming Model

- Enable Video/Picture Acquisition: [0]

Camera ISP Register Manual

- Camera ISP_CSI2A Register Summary: [1]

Table 12-641. CSI2_DBG_P

Address Offset	0x0000 0068		
Physical Address	0x480B D868	**Instance**	ISP_CSI2A
Description	DEBUG REGISTER (Payload) This register provides a way to debug the CSI2 RECEIVER module with no image sensor connected to the module. The debug mode is enabled by CSI2_CTRL.DBG_EN. The register is used to write payload of long packets.		
Type	W		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DBG																															

Bits	Field Name	Description	Type	Reset
31:0	DBG	32-bit input value.	W	0x00000000

Table 12-642. Register Call Summary for Register CSI2_DBG_P

Camera ISP Register Manual

- Camera ISP_CSI2A Register Summary: [0]

Table 12-643. CSI2_TIMING

Address Offset	0x0000 006C		
Physical Address	0x480B D86C	**Instance**	ISP_CSI2A
Description	TIMING REGISTER This register controls the CSI2 RECEIVER module. This register must not be modified while CSI2_CTRL.IF_EN is set to 1. It is used to indicate the number of L3 cycles for the Stop State monitoring.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																FORCE_RX_MODE_IO1	STOP_STATE_X16_IO1	STOP_STATE_X4_IO1	STOP_STATE_COUNTER_IO1														

Bits	Field Name	Description	Type	Reset
31	RESERVED	Reserved	RW	0x0
30	RESERVED	Reserved	RW	0x1
29	RESERVED	Reserved	RW	0x1
28:16	RESERVED	Reserved	RW	0x1FFF
15	FORCE_RX_MODE_IO1	Control of ForceRxMode signal 0x0: De-assertion of ForceRxMode. The HW reset the bit at the end of the Force RX Mode assertion. The SW can reset the bit to stop the assertion of the ForceRXMode signal prior to the completion of the period. 0x1: Assertion of ForceRxMode	RW	0x1
14	STOP_STATE_X16_IO1	Multiplication factor for the number of L3 cycles defined in STOP_STATE_COUNTER_IO1 bit-field 0x0: The number of L3 cycles defined in STOP_STATE _COUNTER_IO1 is multiplied by 1x 0x1: The number of L3 cycles defined in STOP_STATE _COUNTER_IO1 is multiplied by 16x	RW	0x1

Bits	Field Name	Description	Type	Reset
13	STOP_STATE_X4_IO1	Multiplication factor for the number of L3 cycles defined in STOP_STATE_COUNTER_IO1 bit-field 0x0: The number of L3 cycles defined in STOP_STATE _COUNTER_IO1 is multiplied by 1x 0x1: The number of L3 cycles defined in STOP_STATE _COUNTER_IO1 is multiplied by 4x	RW	0x1
12:0	STOP_STATE_COUNTER_IO1	Stop State counter for monitoring. It indicates the number of L3 cycles to monitor for Stop State before de-asserting ForceRxMode (Complex I/O #1). The value is from 0 to 8191.	RW	0x1FFF

Table 12-644. Register Call Summary for Register CSI2_TIMING

Camera ISP Functional Description • Complex I/O: [0] [1] [2] [3] [4] [5] [6] [7]
Camera ISP Basic Programming Model • Enable Video/Picture Acquisition: [8]
Camera ISP Register Manual • Camera ISP_CSI2A Register Summary: [9]

Table 12-645. CSI2_CTx_CTRL1

Address Offset	0x0000 0070 + (x*0x20)	**Index**	x = 0 to 7
Physical Address	0x480B D870 + (x*0x20)	**Instance**	ISP_CSI2A
Description	CONTROL REGISTER - Context This register controls the Context. This register is shadowed: modifications are taken into account after the next FSC sync code.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								FEC_NUMBER								COUNT								EOF_EN	EOL_EN	CS_EN	COUNT_UNLOCK	PING_PONG	RESERVED	LINE_MODULO	CTX_EN

Bits	Field Name	Description	Type	Reset
31:24	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x00
23:16	FEC_NUMBER	Number of FEC to receive between using swap of CSI2_CTx_DAT_PING_ADDR and CSI2_CTx_DAT_PONG_ADDR for the calculation of the address in memory. (must be used only in interlace mode, otherwise set to 1)	RW	0x01
15:8	COUNT	Sets the number of frame to acquire. Once the frame acquisition starts, the COUNT value is decremented after every frame. When COUNT reaches 0, the FRAME_NUMBER_IRQ interrupt is triggered and CTX_EN is set to 0. Writes to this bit field are controlled by the COUNT_UNLOCK bit. During the same OCP write access , the bit-field COUNT_UNLOCK must be written in addition to COUNT bit-field to change the COUNT value. COUNT can be overwritten dynamically with a new count value." 0: Infinite number of frames (no count). 1: 1 frame to acquire ... 255: 255 frames to acquire.	RW	0x00

Bits	Field Name	Description	Type	Reset
7	EOF_EN	Indicates if the end of frame signal must be asserted at the end of the frame. 0x0: The end of frame signal is not asserted at the end of each frame. 0x1: The end of frame signal is asserted at the end of each frame.	RW	0x0
6	EOL_EN	Indicates if the end of line signal must be asserted at the end of the line. 0x0: The end of line signal is not asserted at the end of each line. 0x1: The end of line signal is asserted at the end of each line.	RW	0x0
5	CS_EN	Enables the checksum check for the received payload (long packet only). 0x0: Disabled 0x1: Enabled	RW	0x0
4	COUNT_UNLOCK	Unlock writes to the COUNT bit field. 0x0: COUNT bit field is locked. Writes have no effect 0x1: COUNT bit field is unlocked. Writes are possible.	W	0x0
3	PING_PONG	Indicates whether the PING or PONG destination address (CSI2_CTx_DAT_PING_ADDR or CSI2_CTx_DAT_PONG_ADDR) was used to write the last frame. This bit field toggles after every FEC sync code received for the current context. 0x0: PING buffer 0x1: PONG buffer	R	0x1
2	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x0
1	LINE_MODULO	Line modulo configuration 0x0: CSI2_CTx_CTRL3.LINE_NUMBER is used once per frame for the generation of the LINE_NUMBER_IRQ. 0x1: CSI2_CTx_CTRL3.LINE_NUMBER is used as a modulo number for the generation of the LINE_NUMBER_IRQ (multiple times the interrupt can be generated for each frame)	RW	0x0
0	CTX_EN	Enables the Context 0x0: Disabled 0x1: Enabled	RW	0x0

Table 12-646. Register Call Summary for Register CSI2_CTx_CTRL1

Camera ISP Integration
- Interrupt Requests: [0]

Camera ISP Functional Description
- ECC and Checksum Generation: [1]
- Virtual Channel and Context: [2] [3] [4] [5]
- DMA Engine: [6] [7] [8] [9] [10] [11]

Camera ISP Basic Programming Model
- Enable Video/Picture Acquisition: [12] [13] [14] [15] [16]
- Disable Video/Picture Acquisition: [17]
- Capture a Finite Number of Frames: [18] [19]
- Configure a Periodic Event During Frame Acquisition: [20]
- Progressive and Interleaved Frame Configuration: [21]

Camera ISP Register Manual
- Camera ISP_CSI2A Register Summary: [22]
- Camera ISP_CSI2A Register Description: [23] [24]

Table 12-647. CSI2_CTx_CTRL2

Address Offset	0x0000 0074 + (x*0x20)	**Index**	x = 0 to 7
Physical Address	0x480B D874 + (x*0x20)	**Instance**	CSI2_RECEIVER
Description	CONTROL REGISTER - Context This register controls the Context. This register is shadowed: modifications are taken into account after the next FSC sync code (except for VIRTUAL_ID and FORMAT fields). The change of VIRTUAL_ID and FORMAT has to occur only when the context is disabled (CSI2_CTx_CTRL1.CTX_EN).		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FRAME_NUMBER																RESERVED	USER_DEF_MAPPING		VIRTUAL_ID		RESERVED			FORMAT							

Bits	Field Name	Description	Type	Reset
31:16	FRAME_NUMBER	Frame number received	R	0x0000
15	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x0
14:13	USER_DEF_MAPPING	Selects the pixel format of USER_DEFINED in FORMAT 0x0: RAW6 0x1: RAW7 0x2: RAW8 (not valid if FORMAT is USER_DEFINED_8_BIT_DATA_TYPE_x_EXP8 with x from 1 to 8)	RW	0x0
12:11	VIRTUAL_ID	Virtual channel ID 0x0: Virtual Channel ID 0 0x1: Virtual Channel ID 1 0x2: Virtual Channel ID 2 0x3: Virtual Channel ID 3	RW	0x0
10	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x0

Bits	Field Name	Description	Type	Reset
9:0	FORMAT	Data format selection. 0x0: OTHERS (except NULL and BLANKING packets) 0x12: Embedded 8-bit non-image data (that is, JPEG) 0x18: YUV420 8bit 0x19: YUV420 10bit 0x1A: YUV420 8bit legacy 0x1C: YUV420 8bit + CSPS 0x1D: YUV420 10bit + CSPS 0x1E: YUV422 8bit 0x1F: YUV422 10bit 0x22: RGB565 0x24: RGB888 0x28: RAW6 0x29: RAW7 0x2A: RAW8 0x2B: RAW10 0x2C: RAW12 0x2D: RAW14 0x33: RGB666 + EXP32_24 0x40: USER_DEFINED_8_BIT_DATA_TYPE_1 0x41: USER_DEFINED_8_BIT_DATA_TYPE_2 0x42: USER_DEFINED_8_BIT_DATA_TYPE_3 0x43: USER_DEFINED_8_BIT_DATA_TYPE_4 0x44: USER_DEFINED_8_BIT_DATA_TYPE_5 0x45: USER_DEFINED_8_BIT_DATA_TYPE_6 0x46: USER_DEFINED_8_BIT_DATA_TYPE_7 0x47: USER_DEFINED_8_BIT_DATA_TYPE_8 0x68: RAW6 + EXP8 0x69: RAW7 + EXP8 0x80: USER_DEFINED_8_BIT_DATA_TYPE_1 + EXP8 0x81: USER_DEFINED_8_BIT_DATA_TYPE_2 + EXP8 0x82: USER_DEFINED_8_BIT_DATA_TYPE_3 + EXP8 0x83: USER_DEFINED_8_BIT_DATA_TYPE_4 + EXP8 0x84: USER_DEFINED_8_BIT_DATA_TYPE_5 + EXP8 0x85: USER_DEFINED_8_BIT_DATA_TYPE_6 + EXP8 0x86: USER_DEFINED_8_BIT_DATA_TYPE_7 + EXP8 0x87: USER_DEFINED_8_BIT_DATA_TYPE_8 + EXP8 0x9E: YUV422 8bit + VP 0xA0: RGB444 + EXP16 0xA1: RGB555 + EXP16 0xAB: RAW10 + EXP16 0xAC: RAW12 + EXP16 0xAD: RAW14 + EXP16 0xE3: RGB666 + EXP32 0xE4: RGB888 + EXP32 0xE8: RESERVED 0x12A: RAW8 + VP 0x12C: RAW12 + VP 0x12D: RAW14 + VP	RW	0x000

Bits	Field Name	Description	Type	Reset
		0x12F: RAW10 + VP		
		0x229: RESERVED		
		0x2A8: RESERVED		
		0x2AA: RESERVED		
		0x2C0: RESERVED		
		0x2C1: RESERVED		
		0x2C2: RESERVED		
		0x2C3: RESERVED		
		0x2C4: RESERVED		
		0x2C5: RESERVED		
		0x2C6: RESERVED		
		0x2C7: RESERVED		
		0x329: RESERVED		
		0x32A: RESERVED		
		0x340: RESERVED		
		0x341: RESERVED		
		0x342: RESERVED		
		0x343: RESERVED		
		0x344: RESERVED		
		0x345: RESERVED		
		0x346: RESERVED		
		0x347: RESERVED		

Table 12-648. Register Call Summary for Register CSI2_CTx_CTRL2

Camera ISP Functional Description • Virtual Channel and Context: [0] [1] [2] • DMA Engine: [3] • Data Decompression :
Camera ISP Basic Programming Model • Linking a Context to a Virtual Channel and a Data Type: [5] [6]
Camera ISP Register Manual • Camera ISP_CSI2A Register Summary: [7]

Table 12-649. CSI2_CTx_DAT_OFST

Address Offset	0x0000 0078 + (x*0x20)	**Index**	x = 0 to 7
Physical Address	0x480B D878 + (x*0x20)	**Instance**	CSI2_RECEIVER
Description	DATA MEM ADDRESS OFFSET REGISTER - Context This register sets the offset, which is applied on the destination address after each line is written to memory. This register applies for both CSI2_CTx_DAT_PING_ADDR and CSI2_CTx_DAT_PONG_ADDR. For example, it enables to perform 2D data transfers of the pixel data into a frame buffer. In such case, the pixel data and frame buffer data must have the same data format. Note that the 5 LSBs are ignored: the offset must be a multiple of 32 bytes. This register is shadowed: modifications are taken into account after the next FSC sync code. Only full 32-bit values must be written.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																OFST														RESERVED	

Bits	Field Name	Description	Type	Reset
31:16	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x0000
15:5	OFST	Line offset programmed in bytes (signed value 2's complement). If OFST = 0, the data is written contiguously in memory. Otherwise, OFST sets the destination offset between the first pixel of the previous line and the first pixel of the current line.	RW	0x000
4:0	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x00

Table 12-650. Register Call Summary for Register CSI2_CTx_DAT_OFST

Camera ISP Basic Programming Model
- Enable Video/Picture Acquisition: [0]

Camera ISP Register Manual
- Camera ISP_CSI2A Register Summary: [1]

Table 12-651. CSI2_CTx_DAT_PING_ADDR

Address Offset	0x0000 007C + (x*0x20)	**Index**	x = 0 to 7
Physical Address	0x480B D87C + (x*0x20)	**Instance**	CSI2_RECEIVER
Description	DATA MEM PING ADDRESS REGISTER - Context This register sets the 32-bit memory address where the pixel data are stored. The destination is double buffered: this register sets the PING address. Double buffering is enabled when the addresses CSI2_CTx_DAT_PING_ADDR and CSI2_CTx_DAT_PONG_ADDR are different. Note that the 5 LSBs are ignored: the address should be aligned on a 32-byte boundary. This register is shadowed: modifications are taken into account after the next FSC sync code. Only full 32-bit values must be written.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDR																											RESERVED			

Bits	Field Name	Description	Type	Reset
31:5	ADDR	27 most significant bits of the 32-bit address.	RW	0x0000000
4:0	RESERVED	5 least significant bits of the 32-bit address. Write 0s for future compatibility. Reads return 0.	RW	0x00

Table 12-652. Register Call Summary for Register CSI2_CTx_DAT_PING_ADDR

Camera ISP Functional Description
- DMA Engine: [0] [1]

Camera ISP Basic Programming Model
- Enable Video/Picture Acquisition: [2]

Camera ISP Register Manual
- Camera ISP_CSI2A Register Summary: [3]
- Camera ISP_CSI2A Register Description: [4] [5] [6] [7] [8]

Table 12-653. CSI2_CTx_DAT_PONG_ADDR

Address Offset	0x0000 0080 + (x*0x20)	**Index**	x = 0 to 7
Physical Address	0x480B D880 + (x*0x20)	**Instance**	CSI2_RECEIVER
Description	DATA MEM PONG ADDRESS REGISTER - Context This register sets the 32-bit memory address where the pixel data are stored. The destination is double buffered: this register sets the PONG address. Double buffering is enabled when the addresses CSI2_CTx_DAT_PING_ADDR and CSI2_CTx_DAT_PONG_ADDR are different. Note that the 5 LSBs are ignored: the address should be aligned on a 32-byte boundary. This register is shadowed: modifications are taken into account after the next FSC sync code. Only full 32-bit values must be written.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDR																								RESERVED							

Bits	Field Name	Description	Type	Reset
31:5	ADDR	27 most significant bits of the 32-bit address.	RW	0x0000000
4:0	RESERVED	5 least significant bits of the 32-bit address. Write 0s for future compatibility. Reads return 0.	RW	0x00

Table 12-654. Register Call Summary for Register CSI2_CTx_DAT_PONG_ADDR

Camera ISP Functional Description

- DMA Engine: [0] [1]

Camera ISP Basic Programming Model

- Enable Video/Picture Acquisition: [2]

Camera ISP Register Manual

- Camera ISP_CSI2A Register Summary: [3]
- Camera ISP_CSI2A Register Description: [4] [5] [6] [7] [8]

Table 12-655. CSI2_CTx_IRQENABLE

Address Offset	0x0000 0084 + (x*0x20)	**Index**	x = 0 to 7
Physical Address	0x480B D884 + (x*0x20)	**Instance**	CSI2_RECEIVER
Description	INTERRUPT ENABLE REGISTER - Context This register regroups all the events related to Context.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																								ECC_CORRECTION_IRQ	LINE_NUMBER_IRQ	FRAME_NUMBER_IRQ	CS_IRQ	RESERVED	LE_IRQ	LS_IRQ	FE_IRQ	FS_IRQ

Bits	Field Name	Description	Type	Reset
31:9	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x000000
8	ECC_CORRECTION_IRQ	Context - ECC has been used to correct the only 1-bit error (long packet only). 0x0: Event is masked 0x1: Event generates an interrupt when it occurs	RW	0x0

Bits	Field Name	Description	Type	Reset
7	LINE_NUMBER_IRQ	Context - Line number is reached. 0x0: Event is masked 0x1: Event generates an interrupt when it occurs	RW	0x0
6	FRAME_NUMBER_IRQ	Context - Frame counter reached. 0x0: Event is masked 0x1: Event generates an interrupt when it occurs	RW	0x0
5	CS_IRQ	Context - Check-Sum of the payload mismatch detection 0x0: Event is masked 0x1: Event generates an interrupt when it occurs	RW	0x0
4	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x0
3	LE_IRQ	Context - Line end sync code detection. 0x0: Event is masked 0x1: Event generates an interrupt when it occurs	RW	0x0
2	LS_IRQ	Context - Line start sync code detection. 0x0: Event is masked 0x1: Event generates an interrupt when it occurs	RW	0x0
1	FE_IRQ	Context - Frame end sync code detection. 0x0: Event is masked 0x1: Event generates an interrupt when it occurs	RW	0x0
0	FS_IRQ	Context - Frame start sync code detection. 0x0: Event is masked 0x1: Event generates an interrupt when it occurs	RW	0x0

Table 12-656. Register Call Summary for Register CSI2_CTx_IRQENABLE

Camera ISP Integration • Interrupt Requests: [0]
Camera ISP Functional Description • ECC and Checksum Generation: [1] [2]
Camera ISP Register Manual • Camera ISP_CSI2A Register Summary: [3]

Table 12-657. CSI2_CTx_IRQSTATUS

Address Offset	0x0000 0088 + (x*0x20)	**Index**	x = 0 to 7
Physical Address	0x480B D888 + (x*0x20)	**Instance**	CSI2_RECEIVER
Description	INTERRUPT STATUS REGISTER - Context This register regroups all the events related to Context.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																							ECC_CORRECTION_IRQ	LINE_NUMBER_IRQ	FRAME_NUMBER_IRQ	CS_IRQ	RESERVED	LE_IRQ	LS_IRQ	FE_IRQ	FS_IRQ

Bits	Field Name	Description	Type	Reset
31:9	RESERVED	Write 0s for future compatibility. Reads return 0.	R	0x000000
8	ECC_CORRECTION_IRQ	Context - ECC has been used to do the correction of the only 1-bit error status (long packet only). 0x0: READS: Event is false. WRITES: Status bit unchanged. 0x1: READS: Event is true (pending). WRITES: Status bit is reset.	RW W1toClr	0x0
7	LINE_NUMBER_IRQ	Context - Line number reached status. 0x0: READS: Event is false. WRITES: Status bit unchanged. 0x1: READS: Event is true (pending). WRITES: Status bit is reset.	RW W1toClr	0x0
6	FRAME_NUMBER_IRQ	Context - Frame counter reached status 0x0: READS: Event is false. WRITES: Status bit unchanged. 0x1: READS: Event is true (pending). WRITES: Status bit is reset.	RW W1toClr	0x0
5	CS_IRQ	Context - Check-Sum mismatch status. 0x0: READS: Event is false. WRITES: Status bit unchanged 0x1: READS: Event is true (pending). WRITES: Status bit is reset.	RW W1toClr	0x0
4	RESERVED	Write 0s for future compatibility. Reads return 0.	R	0x0
3	LE_IRQ	Context - Line end sync code detection status. 0x0: READS: Event is false. WRITES: Status bit unchanged. 0x1: READS: Event is true (pending). WRITES: Status bit is reset.	RW W1toClr	0x0
2	LS_IRQ	Context - Line start sync code detection status. 0x0: READS: Event is false. WRITES: Status bit unchanged. 0x1: READS: Event is true (pending). WRITES: Status bit is reset.	RW W1toClr	0x0
1	FE_IRQ	Context - Frame end sync code detection status. 0x0: READS: Event is false. WRITES: Status bit unchanged. 0x1: READS: Event is true (pending). WRITES: Status bit is reset.	RW W1toClr	0x0
0	FS_IRQ	Context - Frame start sync code detection status. 0x0: READS: Event is false. WRITES: Status bit unchanged. 0x1: READS: Event is true (pending). WRITES: Status bit is reset.	RW W1toClr	0x0

Table 12-658. Register Call Summary for Register CSI2_CTx_IRQSTATUS

Camera ISP Integration • Interrupt Requests: [0]
Camera ISP Functional Description • ECC and Checksum Generation: [1] [2] • Virtual Channel and Context: [3] [4]
Camera ISP Basic Programming Model • Enable Video/Picture Acquisition: [5]

Table 12-658. Register Call Summary for Register CSI2_CTx_IRQSTATUS (continued)

Camera ISP Register Manual • Camera ISP_CSI2A Register Summary: [6]

Table 12-659. CSI2_CTx_CTRL3

Address Offset	0x0000 008C + (x*0x20)	**Index**	x = 0 to 7
Physical Address	0x480B D88C + (x*0x20)	**Instance**	CSI2_RECEIVER
Description	CONTROL REGISTER - Context This register controls the Context. This register is shadowed: modifications are taken into account after the next FSC sync code.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								ALPHA								LINE_NUMBER															

Bits	Field Name	Description	Type	Reset
31:30	RESERVED	Write 0s for future compatibility. Reads return 0.	RW	0x0
29:16	ALPHA	Alpha value for RGB888, RGB666 and RBG444.	RW	0x0000
15:0	LINE_NUMBER	Line number for the interrupt generation	RW	0x0000

Table 12-660. Register Call Summary for Register CSI2_CTx_CTRL3

Camera ISP Environment • Camera Serial Interface (CSI2) Protocol and Data Format: [0] [1] [2]
Camera ISP Integration • Interrupt Requests: [3]
Camera ISP Functional Description • Virtual Channel and Context: [4]
Camera ISP Basic Programming Model • Enable Video/Picture Acquisition: [5] • Configure a Periodic Event During Frame Acquisition: [6]
Camera ISP Register Manual • Camera ISP_CSI2A Register Summary: [7] • Camera ISP_CSI2A Register Description: [8] [9]

12.6.12 Camera CSI2PHY_SCP Registers

12.6.12.1 Camera CSI2PHY_SCP Register Summary

Table 12-661. CSI2PHY_SCP Register Summary

Register Name	Type	Register Width (Bits)	Address Offset	Physical Address
CSI2PHY_CFG0	RW	32	0x0000 0000	0x480B D970
CSI2PHY_CFG1	RW	32	0x0000 0004	0x480B D974

12.6.12.2 Camera CSI2PHY_SCP Register Description

Table 12-662. CSI2PHY_CFG0

Address Offset	0x0000 0000		
Physical Address	0x480B D970	**Instance**	CSI2PHY_SCP
Description	Configuration register for Ths-term & Ths-settle		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED							RESERVED	RESERVED								THS_TERM								THS_SETTLE							

Bits	Field Name	Description	Type	Reset
31:25	RESERVED	Write only 0 for future compatibility.	RW	0x00
24	RESERVED	Write only 0 for future compatibility.	RW	0
23:16	RESERVED	Write only 0 for future compatibility.	RW	0x00
15:8	THS_TERM	Ths-term timing parameter in multiples of CSI2_96M_FCLK period. Requirement from DSI_PHY spec = (Dn Voltage < 450 mV) - 35 ns + 4 UI. Effective time for enabling termination = synchoniser delay + timer delay + LPRx delay + combinatorial routing delay ~ (1-2)*DDRCLK + Ths-term + ~ (1-15) ns. Programmed value = ceil(12.5 ns/DDRClk period)-1). Default value: 4 for 400 MHz. **Note:** 20% clock frequency tolerance.	RW	0x04
7:0	THS_SETTLE	Ths-settle timing parameter in multiples of DDR clock period. Derived requirement from DSI_PHY spec = (90 ns + 6 UI) - (145 ns + 10 UI). Effective Ths-settle seen on the line (starting to look for sync pattern) = synchoniser delay + timer delay + LPRx delay + combinatorial routing delay - pipeline delay in HS data path. ~ (1-2)*DDRCLK + Ths-settle + ~ (1-15) ns - 1*DDRClk Programmed value = ceil(90 ns/DDR clock period)+3. Default value: 39 for 400 MHz. **Note:** (1) Minimum supported Ths-settle preprogrammed value = 3. (2) One clock delay in datapath from HSRx needs to be compensated. Hence + 3.	RW	0x27

Table 12-663. Register Call Summary for Register CSI2PHY_CFG0

Camera ISP Functional Description
- Complex I/O: [0]

Camera ISP Basic Programming Model
- Enable Video/Picture Acquisition: [1]

Camera ISP Register Manual
- Camera CSI2PHY_SCP Register Summary: [2]

Table 12-664. CSI2PHY_CFG1

Address Offset	0x0000 0004		
Physical Address	0x480B D974	**Instance**	CSI2PHY_SCP
Description	Configuration register for Tclk-term, Tclk-settle & Tclk-miss. Reset done status for the CLKIN4DDR clock domains.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED		RESETDONECSI2_96M_FCLK	RESETDONERXBYTECLK	RESERVED		RESERVED		TCLK_TERM								RESERVED						TCLK_MISS		TCLK_SETTLE							

Bits	Field Name	Description	Type	Reset
31	RESERVED	Reserved	R	0x1
30	RESERVED	Reserved	R	0x1
29	RESETDONECSI2_96M_FCLK	Reset Done flag for the CSI2_96M_FCLK domain Read 0x0: Reset in progress Read 0x1: Reset completed	R	0x-
28	RESETDONERXBYTECLK	Reset Done flag for the RxByteClkHS clock domain Read 0x0: Reset in progress Read 0x1: Reset completed	R	0x-
27:26	RESERVED	Read returns 0. Write only 0 for future compatibility.	R	0x0
25	RESERVED	Reserved	R	0x-
24:18	TCLK_TERM	Tclk-term timing parameter in multiples of CSI2_96M_FCLK period. Requirement from DSI_PHY spec = (Dn Voltage < 450 mV) - 55 ns. Effective time for enabling termination = synchonizer delay + timer delay + LPRx delay + combinatorial routing delay ~ (1-2)*CSI2_96M_FCLK + Tclk-term + ~ (1-15) ns. Programmed value = ceil(9.5 ns/CSI2_96M_FCLK period)-1). Default value: 0 for 96 MHz. **Note:** 5% clock frequency tolerance.	RW	0x00
17:10	RESERVED	Must be written with reset value	RW	0xB8
9:8	TCLK_MISS	Tclk-miss timing parameter in multiples of CSI2_96M_FCLK period Programmed value = ceil(15 ns/CSI2_96M_FCLK period)-1 Default value: 1 for 96 MHz	RW	0x1
7:0	TCLK_SETTLE	Tclk-settle timing parameter in multiples of CSI2_96M_FCLK period. Derived requirement from DSI_PHY spec = 145 ns - 435 ns. Effective Ths-settle = synchonizer delay + timer delay + LPRx delay + combinatorial routing delay ~ (1-2)*CSI2_96M_FCLK + Tclk-settle + ~ (1-15) ns. Programmed value = max(3, ceil(155 ns/CSI2_96M_FCLK period)-1). Default value: 14 for 96 MHz. **Note:** 5% clock frequency tolerance.	RW	0x0E

Table 12-665. Register Call Summary for Register CSI2PHY_CFG1

Camera ISP Functional Description • Complex I/O: [0]
Camera ISP Basic Programming Model • Enable Video/Picture Acquisition: [1]
Camera ISP Register Manual • Camera CSI2PHY_SCP Register Summary: [2]

2D/3D Graphics Accelerator

This chapter describes the 2D/3D graphics accelerator (SGX) for the device.

Note: The SGX subsystem is an instantiation by Texas Instruments of the POWERVR™ SGX530 core from Imagination Technologies Ltd.

This document contains materials that are ©2003-2007 Imagination Technologies Ltd.

POWERVR and USSE are trademarks or registered trademarks of Imagination Technologies Ltd.

Topic **Page**

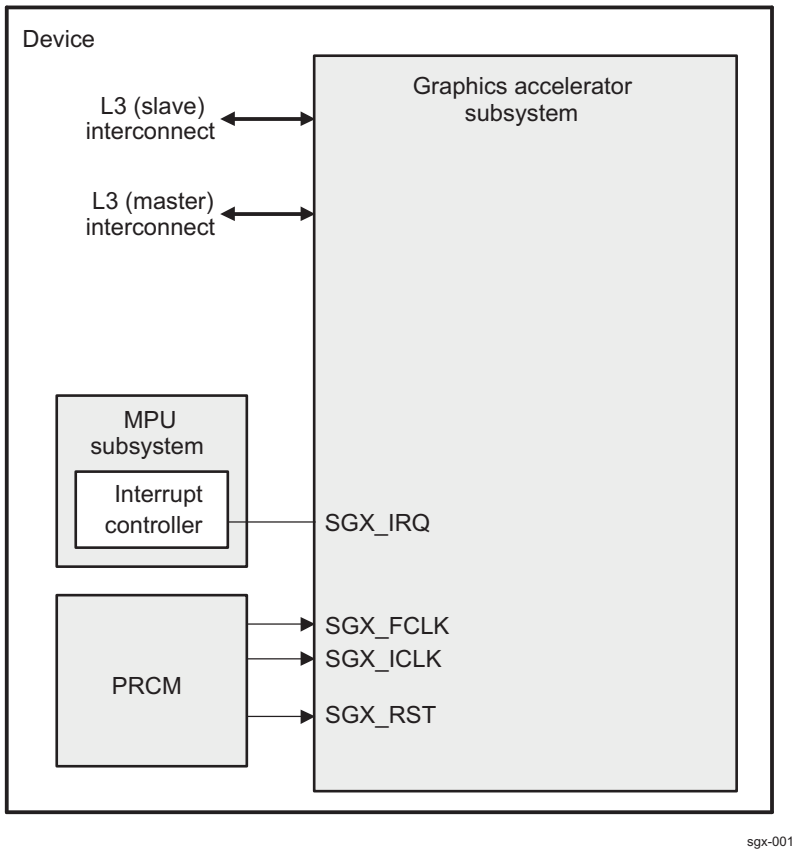
13.1 SGX Overview **1732**
13.2 SGX Integration **1735**
13.3 SGX Functional Description **1737**
13.4 SGX Register Mapping **1738**

13.1 SGX Overview

The 2D/3D graphics accelerator (SGX) subsystem accelerates 2-dimensional (2D) and 3-dimensional (3D) graphics applications. The SGX subsystem is based on the POWERVR™ SGX core from Imagination Technologies. SGX is a new generation of programmable POWERVR graphic cores. The POWERVR™ SGX530 v1.2.1 architecture is scalable and can target all market segments from mainstream mobile devices to high-end desktop graphics. Targeted applications include feature phone, PDA, and hand-held games.

Figure 13-1 shows the SGX subsystem in the device.

Figure 13-1. Graphics Accelerator Highlight



The SGX graphics accelerator efficiently processes a number of various multimedia data types concurrently:

- Pixel data
- Vertex data
- Video data
- General-purpose processing

This is achieved using a multithreaded architecture using two levels of scheduling and data partitioning enabling zero overhead task switching.

The SGX subsystem is connected by a 64-bit master and a 32-bit slave interface to the L3 interconnect.

13.1.1 POWERVR SGX Main Features

- 2D graphics, 3D graphics, vector graphics, and programming support for GP-GPU functions
- Tile-based architecture
- USSE™ – multithreaded engine incorporating pixel and vertex shader functionality
- Advanced shader feature set – in excess of Microsoft VS3.0, PS3.0, and OpenGL2.0
- Industry standard API support – Direct3D Mobile, OpenGL ES 1.1 and 2.0, OpenVG v1.0.1
- Fine-grained task switching, load balancing, and power management
- Advanced geometry direct memory access (DMA) driven operation for minimum CPU interaction
- Programmable high-quality image anti-aliasing
- POWERVR SGX core MMU for address translation from the core virtual address to the external physical address (up to 4GB address range)
- Fully virtualised memory addressing for OS operation in a unified memory architecture
- Advanced and standard 2D operations [e.g., vector graphics, BLTs (block level transfers), ROPs (raster operations)]

13.1.2 SGX 3D Features

- Deferred pixel shading
- On-chip tile floating point depth buffer
- 8-bit stencil with on-chip tile stencil buffer
- 8 parallel depth/stencil tests per clock
- Scissor test
- Texture support:
  - Cube map
  - Projected textures
  - 2D textures
  - Nonsquare textures
- Texture formats:
  - RGBA 8888, 565, 1555
  - Monochromatic 8, 16, 16f, 32f, 32int
  - Dual channel, 8:8, 16:16, 16f:16f
  - Compressed textures PVR-TC1, PVR-TC2, ETC1
  - Programmable support for all YUV formats
- Resolution support:
  - Frame buffer maximum size = 2048 x 2048
  - Texture maximum size = 2048 x 2048
- Texture filtering:
  - Bilinear, trilinear, anisotropic
  - Independent minimum and maximum control
- Antialiasing:
  - 4x multisampling
  - Up to 16x full scene anti-aliasing
  - Programmable sample positions
- Indexed primitive list support
  - Bus mastered
- Programmable vertex DMA
- Render to texture:
  - Including twiddled formats
  - Auto MipMap generation
- Multiple on-chip render targets (MRT).
  Note: Performance is limited when the on-chip memory is not available.

13.1.3 Universal Scalable Shader Engine (USSE) – Key Features

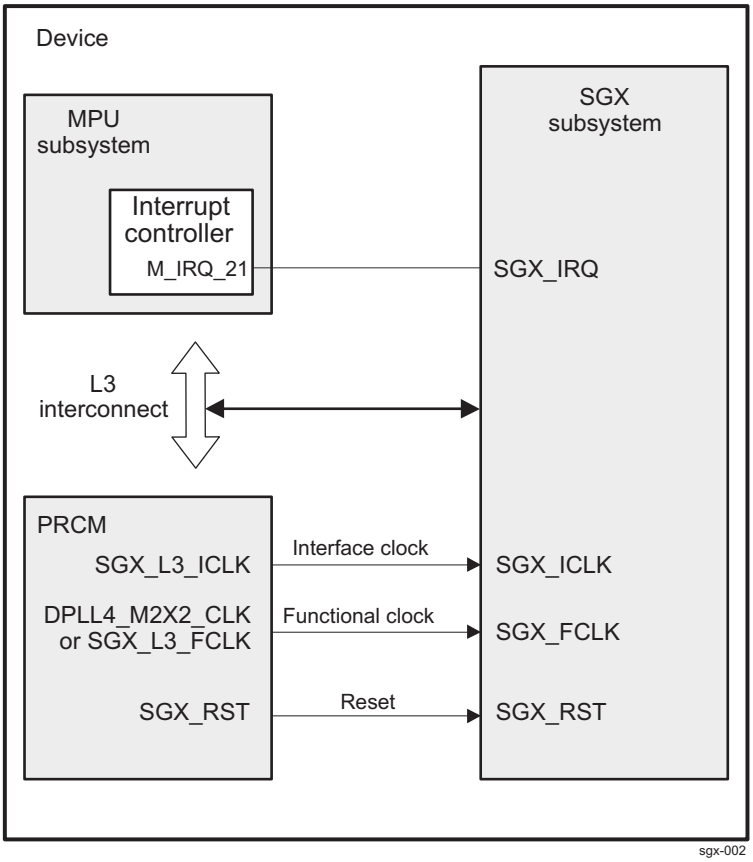
The USSE is the engine core of the POWERVR SGX architecture and supports a broad range of instructions.

- Single programming model:
  - Multithreaded with 16 simultaneous execution threads and up to 64 simultaneous data instances
  - Zero-cost swapping in, and out, of threads
  - Cached program execution model
  - Dedicated pixel processing instructions
  - Dedicated video encode/decode instructions
- SIMD execution unit supporting operations in:
  - 32-bit IEEE float
  - 2-way 16-bit fixed point
  - 4-way 8-bit integer
  - 32-bit bit-wise (logical only)
- Static and dynamic flow control:
  - Subroutine calls
  - Loops
  - Conditional branches
  - Zero-cost instruction predication
- Procedural geometry:
  - Allows generation of primitives
  - Effective geometry compression
  - High-order surface support
- External data access:
  - Permits reads from main memory using cache
  - Permits writes to main memory
  - Data fence facility
  - Dependent texture reads

13.2 SGX Integration

Figure 13-2 highlights the SGX subsystem integration in the device.

Figure 13-2. SGX Subsystem Integration



13.2.1 Clocking, Reset, and Power-Management Scheme

13.2.1.1 Clocks

The SGX subsystem operates from two clocks: an interface clock (SGX_ICLK) and a functional clock (SGX_FCLK). The power, reset, and clock management (PRCM) module generates and distributes both clocks inside the device. The SGX clock tree is depicted in Figure 4-44, *SGX Power Domain Clocking Scheme*, in Chapter 4, *Power, Reset, and Clock Management*.

Table 13-1. Clock Descriptions

Signal Name	I/O[1]	Description
SGX_FCLK	I	Functional clock (two possible clock sources) -> Functional clock domain
SGX_ICLK	I	Interface clock (L3 interconnect clock domain) -> Interface clock domain

(1) I =Input, O=Output,

- The SGX_ICLK interface clock manages the data transfer on the L3 master and slave ports.

  The source of SGX_ICLK is the PRCM clock (SGX_L3_ICLK), which belongs to the SGX clock domain and runs at the L3 interconnect clock speed. The SGX_ICLK frequency is selected based on the whole device L3 interconnect clock frequency. For more information on the interface clock, see Chapter 4, *Power, Reset, and Clock Management*.

  When no longer required by the SGX subsystem, SGX_ICLK can be disabled by software at the PRCM level by setting the PRCM.CM_ICLKEN_SGX[0] EN_SGX bit to 0. For more information, see

Figure 4-60, *SGX Power Domain Clock Controls*, in Chapter 4, *Power, Reset, and Clock Management*.

Note: SGX_ICLK is cut only if the SGX is ready to go into idle state. For more information, see Chapter 4, *Power, Reset, and Clock Management*.

- SGX_FCLK is the functional clock and is used inside the SGX subsystem to generate SGX 2D and 3D domain clock signals.

  The source of SGX_FCLK is either the PRCM clock (SGX_L3_FCLK, which is derived from SGX_ICLK) or the peripheral DPLL clock DPLL4_M2X2_CLK as depicted in Figure 4-60, *SGX Power Domain Clock Controls*. Selection is made at the PRCM level by setting the PRCM.CM_CLKSEL_SGX[2:0] CLKSEL_SGX bit field. A divider of 3, 4, or 6 is applied to the SGX_FCLK frequency with regard to the PRCM SGX_L3_FCLK frequency. By default the SGX_FCLK clock is SGX_L3_FCLK / 3.

  Whether SGX_FCLK is provided by SGX_L3_FCLK or DPLL4_M2X2_CLK, its gating depends on the PRCM.CM_FCLKEN_SGX[1] EN_SGX bit. Clearing this bit to 0 indicates that both SGX clocks are no longer needed and can be cut at the PRCM level, if the module is ready to enter the idle state.For more information, see Section 4.5.5.8, *SGX Power Domain*, in Chapter 4, *Power, Reset, and Clock Management*

13.2.1.2 Resets

The SGX subsystem has its own reset domain. Global reset of the SGX is performed by activating the SGX_RST signal in the SGX_RST domain. Software controls the release of SGX_RST using the PRCM.RM_RSTCTRL_SGX[0] SGX_RST bit.

13.2.1.3 Power Management

The SGX subsystem has its own power domain (SGX power domain). See Chapter 4, *Power, Reset, and Clock Management*, for additional information about the SGX power domain.

As described in Section 13.2, *SGX Integration*, the SGX subsystem receives two clock signals from the PRCM module. The functional clock is either a division of the interface clock with a ratio of 1:3, 1:4, or 1:6, or a 96–MHz clock issued from a DPLL4. The functional clock is used inside the SGX to generate clock signals to the multiple internal module SGX clock domains. The division ratio depends on the PRCM registers setting. For more information, see Chapter 4, *Power, Reset, and Clock Management*.

Three power-management modes are defined:

- Deep power sleep (All clocks are gated.)
- Idle (2D and 3D clocks are gated.)
- 3D (No clock is gated.)

The SGX handles the automatic clock gating performed on the multiple internal module clock domains.

13.2.2 Hardware Requests

13.2.2.1 Interrupt Request

The SGX subsystem can generate one interrupt (SGX_IRQ) to the MPU subsystem interrupt controller mapped on M_IRQ_21.

13.3 SGX Functional Description

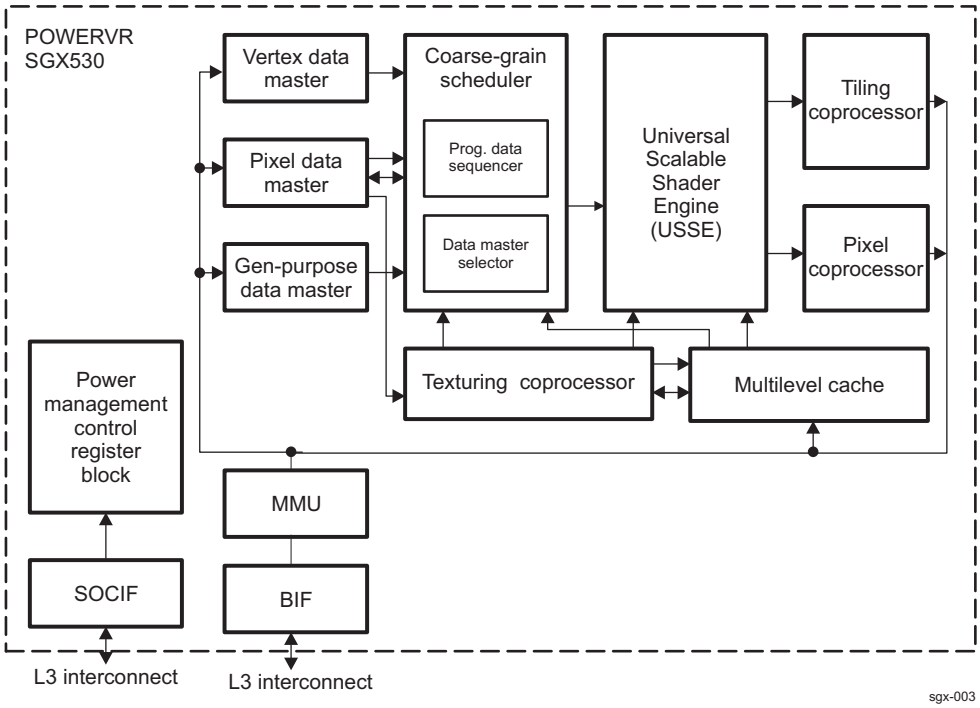
13.3.1 SGX Block Diagram

The SGX subsystem is based on the POWERVR SGX530 core from Imagination Technologies. The architecture uses both programmable and hard coded pipelines to perform various processing tasks required in 2D, 3D, and video processing. The SGX architecture comprises the following elements:

- Coarse grain scheduler
  - Programmable Data Sequencer (PDS)
  - Data Master Selector (DMS)
- Vertex data master (VDM)
- Pixel data master (PDM)
- General-purpose data master
- USSE
- Tiling coprocessor
- Pixel coprocessor
- Texturing coprocessor
- Multilevel cache

Figure 13-3 shows a block diagram of the SGX cores.

Figure 13-3. SGX Block Diagram



13.3.2 SGX Elements Description

The coarse grain scheduler (CGS) is the main system controller for the POWERVR SGX Architecture. It consists of two stages, the DMS and the PDS. The DMS processes requests from the data masters and determines which tasks can be executed given the resource requirements. The PDS then controls the loading and processing of data on the USSE.

There are three data masters in the SGX core:

- The VDM is the initiator of transform and lighting processing within the system. The VDM reads an input control stream, which contains triangle index data, and state data. The state data indicates the PDS program, size of the vertices and the amount of USSE output buffer resource available to the VDM. The triangle data is parsed to determine unique indices that must be processed by the USSE. These are grouped together according to the configuration provided by the driver and presented to the DMS.
- The PDM is the initiator of rasterization processing within the system. Each pixel pipeline processes pixels for a different half of a given tile which allows for optimum efficiency within each pipe due to locality of data. For each task it determines the amount of resource required within the USSE. It merges this with the state address and issues a request for execution on the USSE to the DMS.
- The general-purpose data master responds to events within the system (such as end of a pass of triangles from the ISP, end of a tile from the ISP, end of render, or parameter stream breakpoint event). Each event causes either an interrupt to the host or synchronized execution of a program on the PDS. The program may, or may not cause a subsequent task to be executed on the USSE.

The USSE is a user-programmable processing unit. Although general in nature, its instructions and features are optimized for three types of task: processing vertices (vertex shading), processing pixels (pixel shading), and video/imaging processing.

The multilevel cache is a 2-level cache consisting of two modules: the main cache and the mux/arbiter/demux/decompression unit (MADD). The MADD is a wrapper around the main cache module designed to manage and format requests to and from the cache, as well as providing Level 0 caching for texture and USSE requests. The MADD can accept requests from the PDS, USSE, and texture address generator modules. Arbitration is performed between the three data streams; any required texture decompression is also performed.

The texturing coprocessor performs texture address generation and formatting of texture data. It receives requests from either the iterators or USSE modules and translates these into requests into the multilevel cache. Data returned from the cache are then formatted according to the texture format selected and sent to the USSE for pixel-shading operations.

To process pixels in a tiled manner, the screen is divided into tiles and arranged as groups of tiles by the tiling coprocessor. An inherent advantage of tiling architecture is that a large amount of vertex data can be rejected at this stage, thus reducing both the memory storage requirements and the amount of pixel processing to be performed.

The pixel coprocessor is the final stage of the pixel-processing pipeline and controls the format of the final pixel data sent to the memory. It supplies the USSE with an address into the output buffer, and the USSE returns the relevant pixel data. The address order is determined by the frame buffer mode. The pixel coprocessor contains a dithering and packing function.

13.4 SGX Register Mapping

Table 13-2 describes the SGX memory mapping in the device.

Table 13-2. Instance Summary

Module Name	Start Address	End Address	Size
SGX	0x5000 0000	0x5000 FFFF	64K bytes

CAUTION

The SGX registers are limited to 32-bit data accesses; 8- and 16-bit accesses are not allowed because they can corrupt register content.

IVA2.2 Subsystem

This chapter describes the image video and audio accelerator (IVA2.2) subsystem for the OMAP34xx multimedia device.

Topic **Page**

14.1 IVA2.2 Subsystem Overview 1740
14.2 IVA2.2 Subsystem Integration 1742
14.3 IVA2.2 Subsystem Functional Description 1751
14.4 IVA2.2 Subsystem Basic Programming Model 1829
14.5 IVA2.2 Subsystem Register Manual 1884

14.1 IVA2.2 Subsystem Overview

The device includes the high-performance Texas Instruments image video and audio accelerator (IVA2.2), based on the TMS320DMC64X+ VLIW digital signal processor (DSP) core.

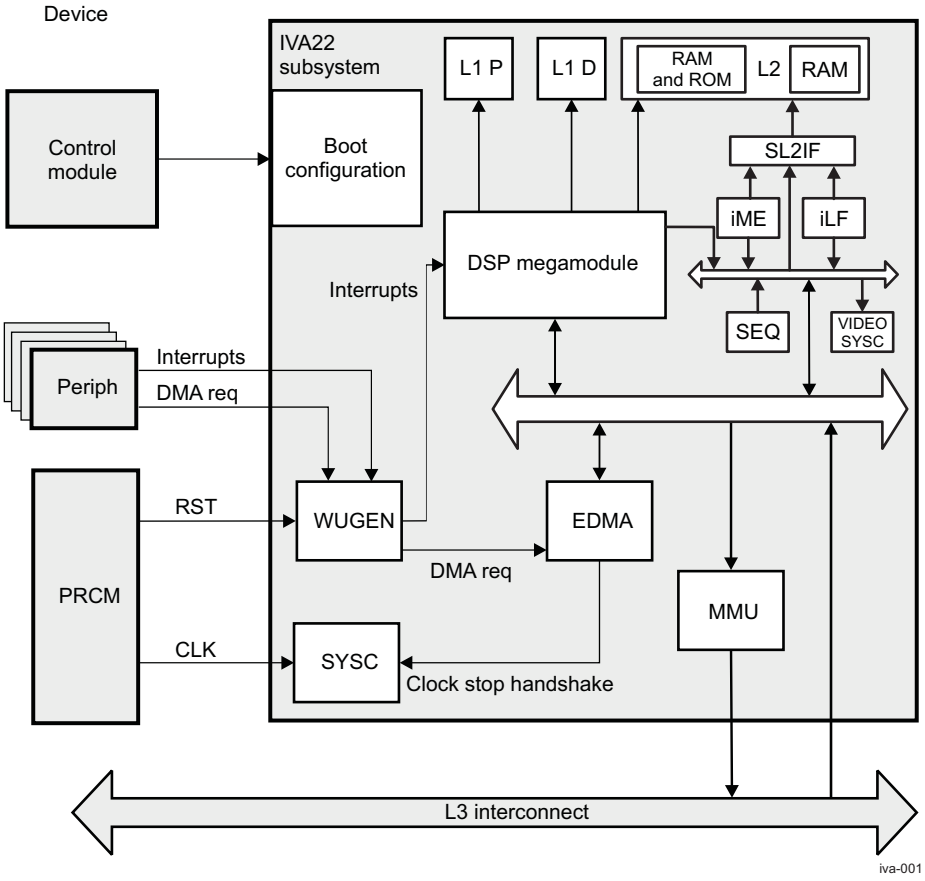
The internal architecture is an assembly of the following components:

- High-performance TI DSP (TMS320DMC64X+) integrated in a megamodule, including local L1/L2 cache and memory controllers
- L1 RAM and L2 RAM and ROM
- Video hardware accelerator module, including local sequencer
- Dedicated enhanced data memory access (EDMA) engine to download/upload data from/to memories and peripherals external to the subchip
- Dedicated memory management unit (MMU) for accessing level 3 (L3) interconnect address space
- Local interconnect network
- Dedicated modules SYSC and WUGEN in charge of power management, clock generation, and connection to the power, reset, and clock manager (PRCM) module

For information about C64x to C64x+ core migration see external document Section 14.3.1.8.

Figure 14-1 shows the IVA2.2 subsystem top-level architecture.

Figure 14-1. IVA2.2 Subsystem Highlight



14.1.1 IVA2.2 Subsystem Key Features

The IVA2.2 subsystem has the following main features:

- 32-bit fixed-point media processor

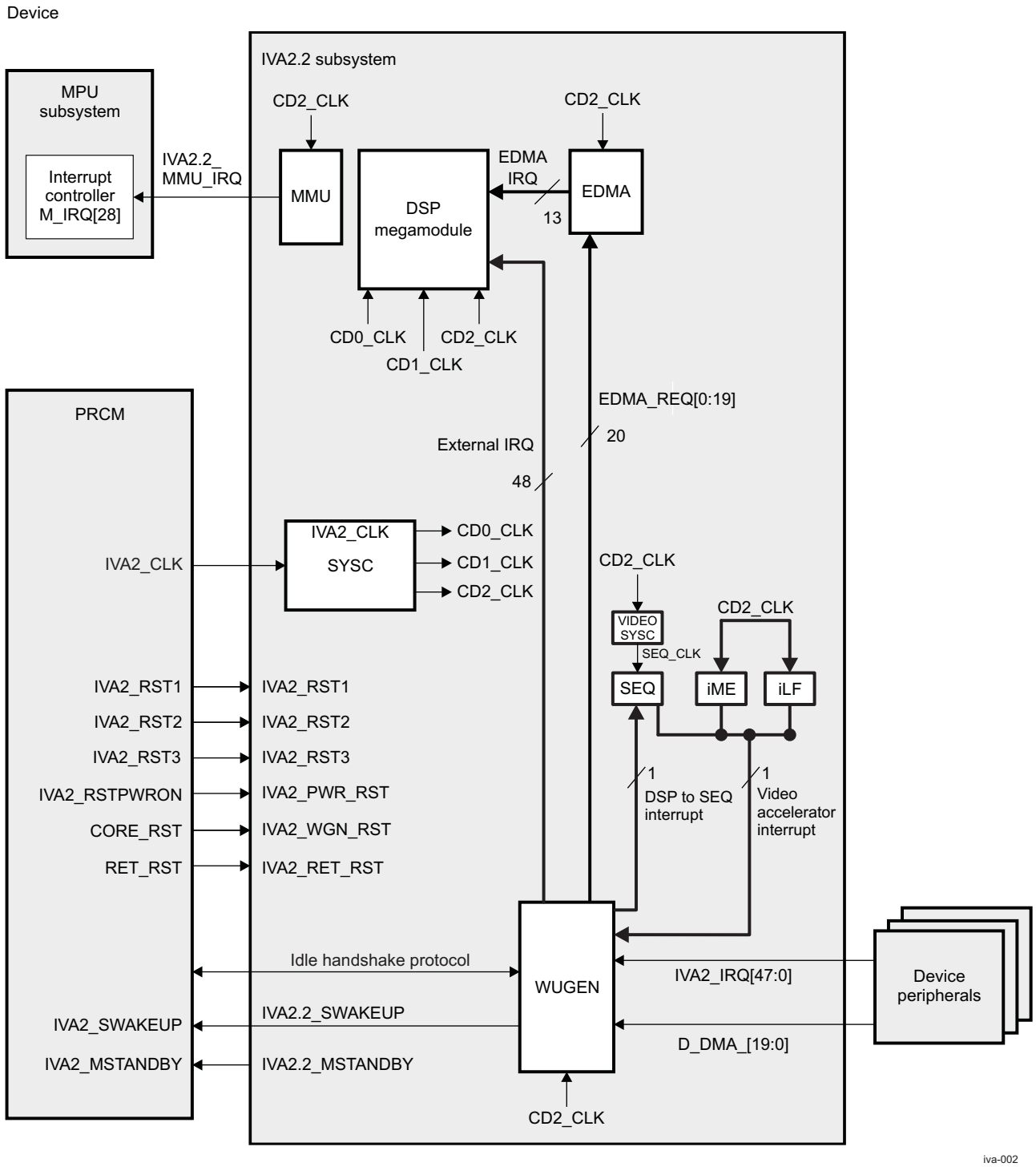
- VLIW architecture based on programmable enhanced version of C64x DSP core
- 8 instructions/cycle, 8 execution units:
  - Optimized instruction set for video and image processing
  - Eight 8 x 8 or 16 x 16 MAC per cycle
  - Eight SAD per cycle
  - Eight interpolations (a + b + 1) >>1 per cycle
  - Two (32-bit x 32-bit > 64-bit) multiply operations per cycle
- Low-power processor and megamodule:
  - Dynamically mixed 32-bit and 16-bit instruction sets
  - Software pipelined loop (SPLOOP) instruction buffer
  - Separate power domain
  - Supported multiple power-down states
- 2-level memory subsystem hierarchy:
  - L1P (program):
    - 32KB direct-mapped cache-32-byte cache line, configurable as cache or memory-mapped (possible values are: 0KB cache/32KB memory, 4KB/28KB, 8KB/24KB, 16KB/16KB, or 32KB/0KB)
  - L1D (data):
    - 32KB 2-way set associative cache-64-byte cache line, configurable as cache or memory-mapped (possible values are: 0KB cache/32KB memory, 4KB/28KB, 8KB/24KB, 16KB/16KB, or 32KB/0KB)
    - 48-KB memory-mapped SRAM
  - L2 (program and data):
    - 64KB 2-way set associative cache-128-byte cache line, configurable as cache or memory-mapped (possible values are: 0KB cache/64KB memory, 32KB/32KB, or 64KB/0KB)
    - 32-KB memory-mapped SRAM
    - 16-KB ROM
- Video hardware accelerator:
  - Improved motion estimation dedicated hardware
  - Improved loop-filtering dedicated hardware
  - Dedicated sequencer
  - Local interconnect
  - Shared level 2 (L2) memory interface/arbiter
- Private direct memory access (DMA) controller:
  - 128 logical channels
  - 1D/2D addressing
  - Chaining capability
  - Fully pipelined, two 64-bit read ports, two 64-bit write ports
  - Single-access 32-byte or 64-byte incrementing bursts
- Level 1 (L1) interrupt controller
- Local IVA digital phase-locked loop (DPLL) supplying IVA subsystem clocking
- 32-entry MMU for seamless integration in high-level operating system (OS) environment
- IVA2.2 system interfaces:
  - 64-bit L3 port shared for external memory accesses:
    - Multithreaded link shared by DSP core and DMA accesses
    - Interface with the L3 interconnect that can be synchronous or asynchronous for clock decoupling between IVA2.2 and L3
    - Incrementing burst support
    - Critical line first, to reduce line-fetch latency to the processor
  - Host port interface (HPI) for MMU programming and access to IVA2.2 internal memories. Can be synchronous or asynchronous.

  - System interfaces: clocking, power management
- C-friendly environment (state-of-the-art C-compiler for VLIW architecture)
- Texas Instruments low-overhead DSP-BIOS operating system

14.2 IVA2.2 Subsystem Integration

Figure 14-2 shows IVA2.2 subsystem integration in the device.

Figure 14-2. IVA2.2 Subsystem Integration



14.2.1 Clocking, Reset, and Power-Management Scheme

14.2.1.1 Clocks

14.2.1.1.1 IVA2.2 Clocks

The IVA2.2 subsystem receives one single clock signal (IVA_CLK) from the PRCM.

Note: When the IVA2.2 subsystem does not require an internal clock, the internal clocks can be disabled at the PRCM level by setting the PRCM.CM_FCLKEN_IVA2 EN_IVA2 bit to 0.

From this clock (IVA_CLK), three internal clocks are generated by the IVA2.2 system control module (SYSC):

- CD0_CLK: Fastest IVA2.2 subsystem functional clock, dedicated to the DSP. Its frequency can be adjusted at the PRCM level by setting the PRCM.CM_CLKSEL1_PLL_IVA2 and PRCM.CM_CLKSEL2_PLL_IVA2 registers.
- CD1_CLK: Divide-by-two of the CD0_CLK clock
- CD2_CLK: Divide-by-two of the CD0_CLK clock

Generation of these three clocks is handled internally to the IVA2.2 subsystem by the SYSC module under direct control of the PRCM.

Table 14-1 lists the clock domains in the IVA2.2 subsystem and shows the roles of the clocks.

Table 14-1. IVA2.2 Internal Clock

CD0	CD0_CLK	DSP core + PMC + DMC
CD1	CD1_CLK	UMC + power-down + interrupt controller
CD2	CD2_CLK	EMC + IDMA + DSP megamodule external interfaces + local interconnect + EDMA + MMU + SYSC + iME + iLF

Note: The internal clocks can be shut down by the PRCM module after the handshake protocol is complete. To configure the PRCM so that internal clocks are hardware-supervised, set the PRCM.CM_AUTOIDLE_PLL_IVA2[2:0] AUTO_IVA2_DPLL field to 0x1. For more information, see Chapter 4, *Power, Reset, and Clock Management*.

The video sequencer module receives SEQ_CLK from CD2_CLK. This divided clock is generated in the video accelerator/sequencer system configuration. For more information about the divided clock of the sequencer, see Section 14.4.9.1, *Clock Management*.

CAUTION

Clock configurations depend on core voltage, and maximum clock frequencies may not apply to production.

14.2.1.2 Resets

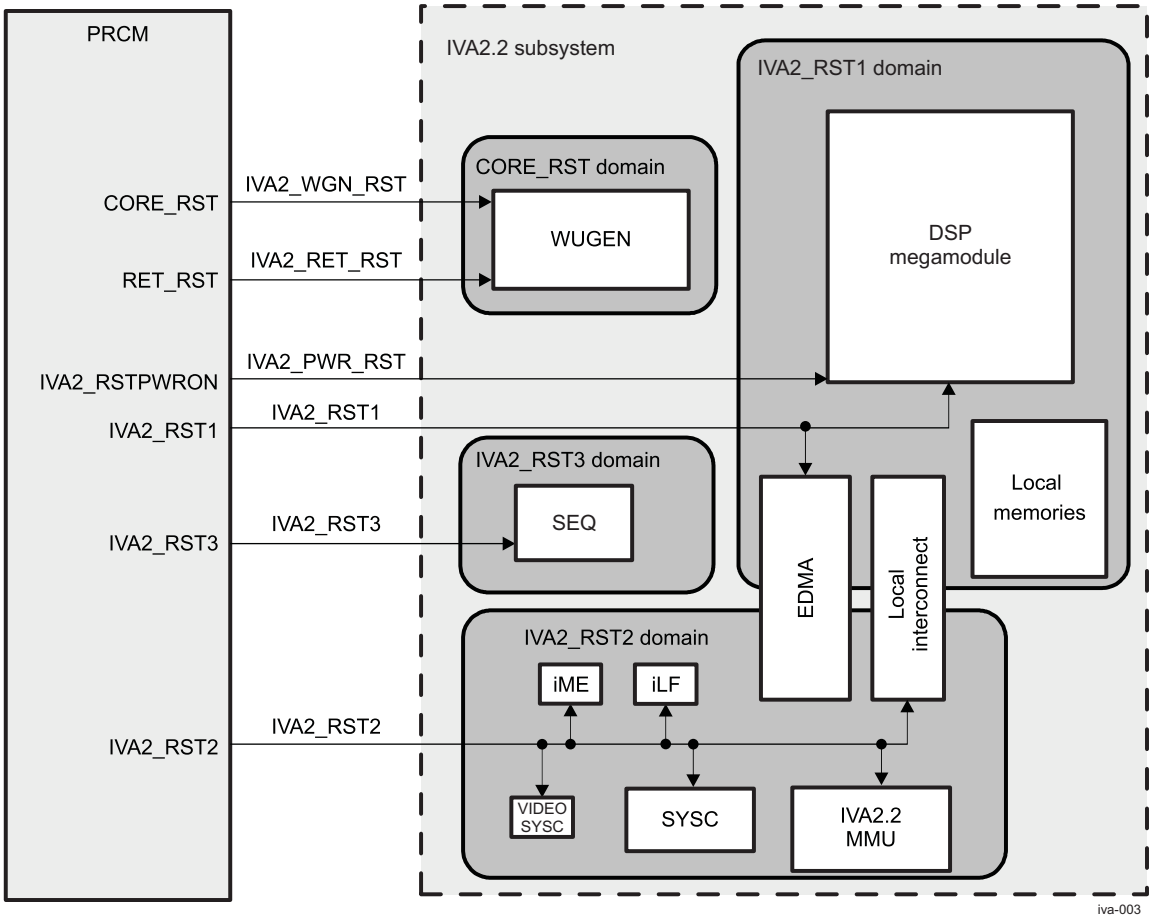
14.2.1.2.1 Hardware Resets

Figure 14-3 shows the seven hardware input reset signals at the boundary of the IVA2.2 subsystem:

- IVA2_RST1, connected to the DSP megamodule and EDMA modules
- IVA2_RST2, connected to the SYSC, MMU, iME, iLF, and local interconnect
- IVA2_RST3, connected to the sequencer
- CORE_RST, connected to the IVA2.2 subsystem wake-up generator (WUGEN) module

- RET_RST, connected to the subsystem WUGEN retention logic to reset registers that keep their value across a warm reset sequence
- IVA2_RSTPWRON, which performs a power-on initialization of IVA2.2 logic

Figure 14-3. IVA2.2 Subsystem Resets



After chip power on, the IVA2.2 subsystem is kept under reset and only the RET_RST is released from reset. Then, only the WUGEN is released from reset as part of the core domain. The IVA2.2 remains under reset until the microprocessor unit (MPU) clears the PRCM.RM_RSTCTRL_IVA2[1] RST2_IVA2 bit. On this action, the PRCM switches on the IVA2.2 power domain, sets the clocks back, and releases the power-on reset. When the IVA2.2 power-on sequence completes (hardware handshake), the PRCM releases the IVA2_RST2 reset. At this stage, the DSP megamodule is kept under reset (unless the MPU also cleared the PRCM.RM_RSTCTRL_IVA2[0] RST1_IVA2 bit); the MPU can upload some code and data in the C64x+ memory. When the MPU has uploaded the code in the C64x+ memory, the MPU clears the RST1_IVA2 bit, releasing the DSP megamodule from reset. At this point, the sequencer is kept under reset (unless the PRCM.RM_RSTCTRL_IVA2[2] RST3_IVA2 bit was cleared); the DSP can upload some code and data in the sequencer memory. When the DSP has uploaded the coded in the sequencer memory, the DSP clears the RST3_IVA2 bit, releasing the sequencer from reset.

The MPU can apply a software reset to the IVA2.2. For a software reset to be safe, the IVA2.2 must be in clock-off mode (DSP in idle mode with clocks shut down by the PRCM; for information about the clock-off state transition, see Section 14.4.9.3*, Power-Down and Wake-Up Management*.) For instance, the MPU can use a mailbox interrupt to ask the IVA2.2 to go to IDLE state. In that case, only IVA2_RST1 and/or IVA2_RST2 and/or IVA2_RST3 are applied, depending on which bits are set, and CORE_RST and IVA2_RSTPWRON are never applied.

The IVA2.2 can also program the PRCM to go to OFF state automatically when the IVA2.2 is idle. Then, in response to a wake-up event (for example, an interrupt), the PRCM can switch on the IVA2.2 power domain, set the clocks back, and release the power-on reset. When the IVA2.2 power-on sequence completes (hardware handshake), the PRCM releases the IVA2_RST3, IVA2_RST2, and IVA2_RST1 resets (RST1, RST2, and RST3 SW bits are assumed to be clear). However, for the IDLE instruction (to transition to OFF state) to be executed, the user must program the PRCM to prevent the sequencer (IVA2_RST3) from being released from reset after a wakeup of the IVA2 from POWER OFF state. CORE_RST can be applied only if the device is coming out of CHIP OFF state (CORE power domain also in OFF state). In this case, CORE_RST is released at the same time as the IVA power-on reset.

For some detected severe issues or if the user applies an external system reset, the IVA2.2 can receive a warm reset. IVA2_RST1, IVA2_RST2, IVA2_RST3, DPLL_RST, RET_RST, and CORE_RST are applied. When the warm reset source is released, IVA2_RST1, IVA2_RST2, IVA2_RST3, and CORE_RST are released. Power-on reset is not applied in this case.

14.2.1.2.2 Software Resets

IVA2.2 subsystem reset signals are sourced from the PRCM module. Some modules of the DSP subsystem can also be reset by software control.

The IVA2_RST1 signal maps to the PRCM.RM_RSTCTL_IVA2 RST1_IVA2 bit, the IVA2_RST2 signal maps to the PRCM.RM_RSTCTL_IVA2 RST2_IVA2 bit, and the IVA2_RST3 signal maps to the PRCM.RM_RSTCTL_IVA2 RST3_IVA2 bit in the PRCM register RM_RSTCTRL_IVA2. Writing these three bit fields lets the software reset the IVA2.2 subsystem.

Note: Software reset can be applied only while the IVA2.2 subsystem is in clock-off mode.

The WUGEN reset signal (CORE_RST) is reset with the CORE power domain, which is reset at the occurrence of either global cold or global warm reset events. For more information about global reset sources, see Chapter 4, *Power, Reset, and Clock Management*.

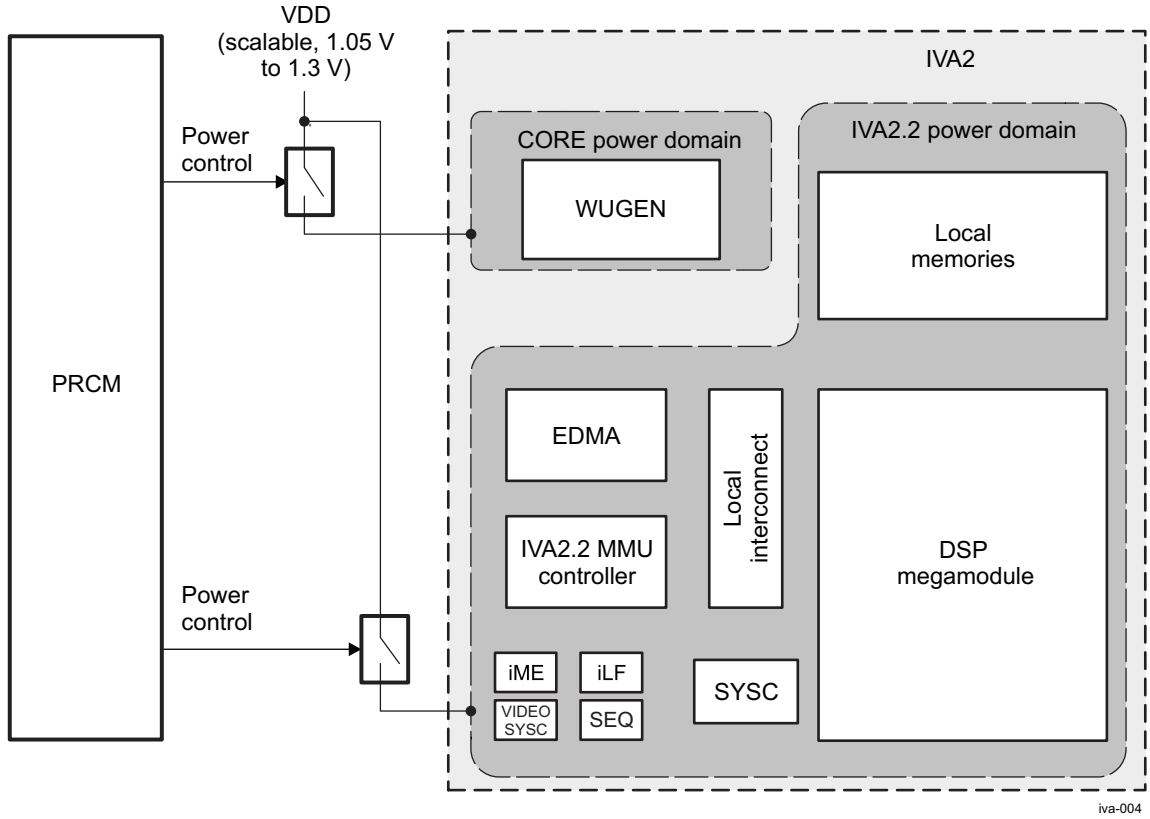
For more information about software resets, see Section 14.4.9.2, *Reset Management*.

14.2.1.3 Power Domain

The IVA2.2 subsystem comprises three power domains that can be shared with other subsystems at the chip level (see Figure 14-4):

- DSP power domain: Everything specified in the IVA2.2 subsystem, except the wake-up generator
- CORE power domain: Includes the WUGEN

Figure 14-4. IVA2.2 Power Domain



The DSP power domain can be powered off if the CORE power domain is powered on. The PRCM ensures that the DSP power domain is always powered off when the CORE power domain is powered off.

The CORE power domain includes several other system-level components of the device. The WUGEN module is included in the CORE power domain to enable the powered-off DSP subsystem to be awakened after receiving an interrupt, a DMA request, or an L3 slave port access.

Memory voltage can be reduced locally when memory is not being used, to reduce memory leakage:

- If IVA2.2 is not active, this is controlled by the PRCM.PM_PWSTCTRL_IVA2 register by setting the MEMONSTATE and/or MEMRETSTATE bit fields.

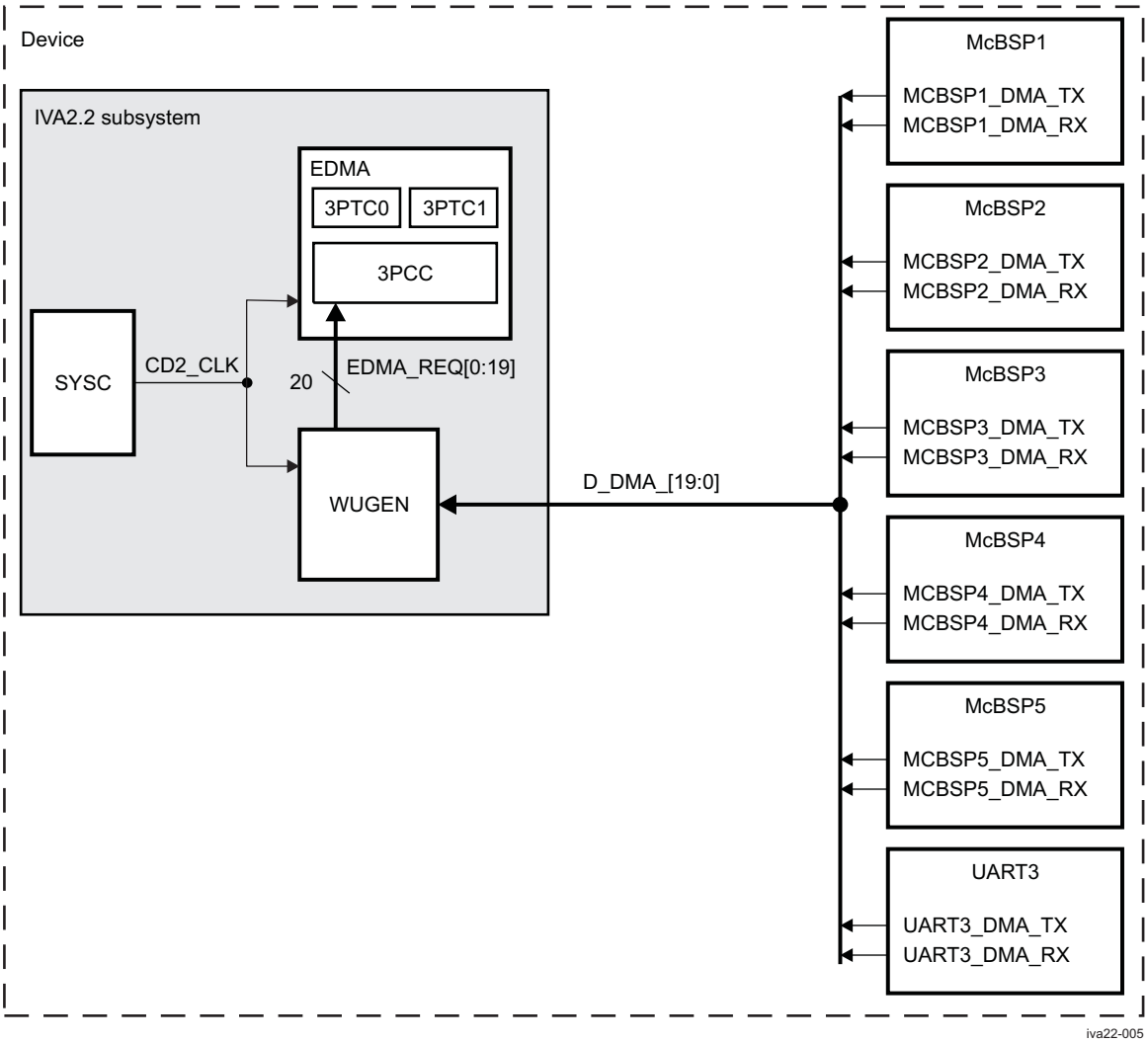
For a detailed description of power domains and controls in the device, see Chapter 4, *Power, Reset, and Clock Management*.

14.2.2 Hardware Requests

14.2.2.1 DMA Requests

The IVA2.2 subsystem receives 14 DMA requests from specific peripherals, such as the McBSP module and the UART module (see Figure 14-5).

Figure 14-5. IVA2.2 EDMA Requests



For a description of the EDMA controller, see Section 14.3.2.1, *EDMA*.

Table 14-2 lists EDMA request mappings to the IVA2.2 subsystem EDMA controller.

Table 14-2. IVA2.2 Subsystem EDMA Request Mappings

DMA	Source	Description
D_DMA_0	MCBSP1_DMA_TX	MCBSP module 1 - transmit request
D_DMA_1	MCBSP1_DMA_RX	MCBSP module 1 - receive request
D_DMA_2	MCBSP2_DMA_TX	MCBSP module 2 - transmit request
D_DMA_3	MCBSP2_DMA_RX	MCBSP module 2 - receive request
D_DMA_4	MCBSP3_DMA_TX	MCBSP module 3 - transmit request
D_DMA_5	MCBSP3_DMA_RX	MCBSP module 3 - receive request
D_DMA_6	MCBSP4_DMA_TX	MCBSP module 4 - transmit request

Table 14-2. IVA2.2 Subsystem EDMA Request Mappings (continued)

DMA	Source	Description
D_DMA_7	MCBSP4_DMA_RX	MCBSP module 4 - receive request
D_DMA_8	MCBSP5_DMA_TX	MCBSP module 5 - transmit request
D_DMA_9	MCBSP5_DMA_RX	MCBSP module 5 - receive request
D_DMA_10	UART3_DMA_TX	UART module 3 - transmit request
D_DMA_11	UART3_DMA_RX	UART module 3 - receive request
D_DMA_12	Reserved	
D_DMA_13	Reserved	
D_DMA_14	Reserved	
D_DMA_15	Reserved	
D_DMA_16	Reserved	
D_DMA_17	Reserved	
D_DMA_18	Reserved	
D_DMA_19	Reserved	

Note: All EDMA requests are shared DMA requests; they are also mapped on the system DMA (sDMA). For more information, see Chapter 9, *DMA*.

For more information about EDMA request management, see Section 14.4.4.3, *Programming an EDMA Transfer*.

14.2.2.2 Interrupt Requests

The IVA2.2 subsystem manages three types of interrupts (see Figure 14-6):

- Internal interrupts: Requests generated by modules in the IVA2.2 subsystem or in the DSP megamodule included in the IVA2.2 subsystem
- External interrupts: Requests generated by peripherals external to the IVA2.2 subsystem, like SPI, display subsystem, or camera subsystem. Peripherals that generate interrupts at the IVA2.2 level use the IVA2_IRQ[47:0] input lines of the IVA2.2 subsystem.
- MMU interrupt: The IVA2.2 MMU can generate an interrupt to an external host outside the IVA2.2 subsystem. As shown in Figure 14-2, the interrupt line of the MMU, IVA2_MMU_IRQ, is connected to M_IRQ_28 of the MPU subsystem interrupt controller.

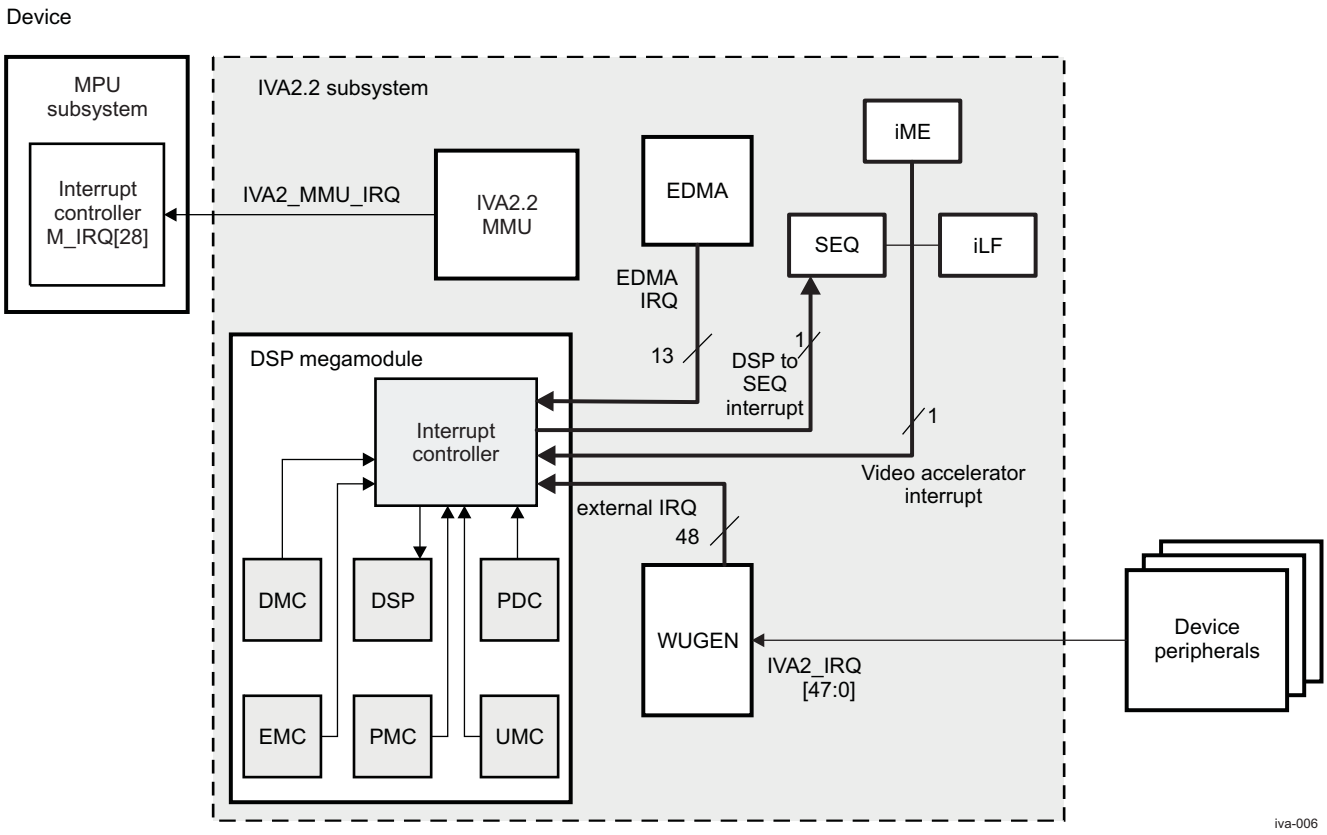
For more information about the functionality of the IVA2.2 MMU, see Section 14.3, *IVA2.2 Subsystem Functional Description*.

To manage and expand the interrupt capabilities of the DSP megamodule (internal and external interrupt requests), the IVA2.2 subsystem includes two levels of interrupt control (see Figure 14-6):

- One interrupt controller (INTC), integrated in the DSP megamodule. This module controls all interrupt requests (internal and external) except the MMU interrupt.

  For more information about the INTC, see Section 14.3.1.7.
- One WUGEN: Responsible at the IVA2.2 subsystem level for detecting wake-up events (interrupts, DMA requests, and slave port access) when the IVA2.2 is powered down. The WUGEN also handles formatting interrupts from device peripherals to the DSP megamodule INTC.

Figure 14-6. IVA2.2 Interrupt Management



In addition to interrupt sources internal to the IVA2.2 subsystem, several interrupt sources coming from device peripherals are connected to the DSP megamodule INTC, but not directly.

These external interrupt sources are first resynchronized in the WUGEN module, synchronously with the CD2 clock domain clock (CD2_CLK). Then the resynchronized versions are connected to the INTC.

Table 14-3 lists the global interrupt mappings of the IVA2.2 subsystem.

Table 14-3. IVA2.2 Interrupt Mappings

EVT	Interrupts/Events	IVA2.2 Pin Name	Interrupt Source	Interrupt Description
0	EVT0	N/A (internal)	DSP INT CTL	Output of event combiner 0, for events 4-31
1	EVT1	N/A (internal)	DSP INT CTL	Output of event combiner 1, for events 32-63
2	EVT2	N/A (internal)	DSP INT CTL	Output of event combiner 2, for events 64-95
3	EVT3	N/A (internal)	DSP INT CTL	Output of event combiner 3, for events 96-127
4-8	Reserved	N/A (internal)	N/A	N/A
9	EMU_DTDMA	N/A (internal)	DSP ECM	ECM interrupt (host scan access, DTDMA)
10	Reserved	N/A (internal)	N/A	N/A
11	EMU_RTDXRX	N/A (internal)	DSP RTDX	RTDX receive complete
12	EMU_RTDXTX	N/A (internal)	DSP RTDX	RTDX transmit complete
13	IDMAINT0	N/A (internal)	DSP IDMA	Internal DMA (IDMA) channel 0 interrupt
14	IDMAINT1	N/A (internal)	DSP IDMA	IDMA channel 1 interrupt
15-27	Reserved	N/A (internal)	N/A	N/A
28	CCMPINT	N/A (internal)	EDMA TPCC	TPCC memory protection interrupt
29	CCINTG	N/A (internal)	EDMA TPCC	TPCC global interrupt
30	CCINT3	N/A (internal)	EDMA TPCC	TPCC region 3 interrupt

Table 14-3. IVA2.2 Interrupt Mappings (continued)

EVT	Interrupts/Events	IVA2.2 Pin Name	Interrupt Source	Interrupt Description
31	CCINT4	N/A (internal)	EDMA TPCC	TPCC region 4 interrupt
32	CCINT5	N/A (internal)	EDMA TPCC	TPCC region 5 interrupt
33	CCINT6	N/A (internal)	EDMA TPCC	TPCC region 6 interrupt
34	CCINT7	N/A (internal)	EDMA TPCC	TPCC region 7 interrupt
35	CCINT8	N/A (internal)	EDMA TPCC	TPCC region 8 interrupt
36	CCINT1	N/A (internal)	EDMA TPCC	TPCC region 1 interrupt
37	CCINT2	N/A (internal)	EDMA TPCC	TPCC region 2 interrupt
38	CCERRINT	N/A (internal)	EDMA TPCC	TPCC error interrupt
39	TCERRINT0	N/A (internal)	EDMA TPTC0	TPTC0 error interrupt
40	TCERRINT1	N/A (internal)	EDMA TPTC1	TPTC1 error interrupt
41-44	Reserved	N/A (internal)	N/A	N/A
45-50	Reserved	IVA2_IRQ[0-5]	N/A	Reserved
51	GPT5_IRQ	IVA2_IRQ[6]	GPTIMER5	General-purpose timer module 5
52	GPT6_IRQ	IVA2_IRQ[7]	GPTIMER6	General-purpose timer module 6
53	GPT7_IRQ	IVA2_IRQ[8]	GPTIMER7	General-purpose timer module 7
54	GPT8_IRQ	IVA2_IRQ[9]	GPTIMER8	General-purpose timer module 8
55	MAIL_U1_IVA2_IRQ	IVA2_IRQ[10]	MAILBOX	Mailbox user 1 interrupt request
56	CAM_IRQ1	IVA2_IRQ[11]	Camera subsystem	Camera module interrupt request 1
57	PRCM_IVA_IRQ	IVA2_IRQ[12]	PRCM	PRCM module
58	DSS_IRQ	IVA2_IRQ[13]	Display subsystem	Display subsystem module
59	Reserved	IVA2_IRQ[14]	N/A	N/A
60	UART3_IRQ	IVA2_IRQ[15]	UART3	UART module 3 (also infrared)
61	MCBSP1_IRQ_TX	IVA2_IRQ[16]	MCBSP1	MCBSP module 1 transmit
62	MCBSP1_IRQ_RX	IVA2_IRQ[17]	MCBSP1	MCBSP module 1 receive
63	Reserved	IVA2_IRQ[18]	N/A	N/A
64	MCBSP2_IRQ_TX	IVA2_IRQ[19]	MCBSP2	MCBSP module 2 transmit
65	MCBSP2_IRQ_RX	IVA2_IRQ[20]	MCBSP2	MCBSP module 2 receive
66	MCBSP3_IRQ_TX	IVA2_IRQ[21]	MCBSP3	MCBSP module 3 transmit
67	MCBSP3_IRQ_RX	IVA2_IRQ[22]	MCBSP3	MCBSP module 3 receive
68	MCBSP4_IRQ_TX	IVA2_IRQ[23]	MCBSP4	MCBSP module 4 transmit
69	MCBSP4_IRQ_RX	IVA2_IRQ[24]	MCBSP4	MCBSP module 4 receive
70	MCBSP5_IRQ_TX	IVA2_IRQ[25]	MCBSP5	MCBSP module 5 transmit
71	MCBSP5_IRQ_RX	IVA2_IRQ[26]	MCBSP5	MCBSP module 5 receive
72	Reserved	IVA2_IRQ[27]	Reserved	Reserved
73	GPIO1_IVA2_IRQ	IVA2_IRQ[28]	GPIO1	GPIO module 1
74	GPIO2_IVA2_IRQ	IVA2_IRQ[29]	GPIO2	GPIO module 2
75	GPIO3_IVA2_IRQ	IVA2_IRQ[30]	GPIO3	GPIO module 3
76	GPIO4_IVA2_IRQ	IVA2_IRQ[31]	GPIO4	GPIO module 4
77	GPIO5_IVA2_IRQ	IVA2_IRQ[32]	GPIO5	GPIO module 5
78	MCBSP1_IRQ	IVA2_IRQ[33]	MCBSP1	MCBSP module 1
79	MCBSP2_IRQ	IVA2_IRQ[34]	MCBSP2	MCBSP module 2
80	MCBSP3_IRQ	IVA2_IRQ[35]	MCBSP3	MCBSP module 3
81	MCBSP4_IRQ	IVA2_IRQ[36]	MCBSP4	MCBSP module 4
82	MCBSP5_IRQ	IVA2_IRQ[37]	MCBSP5	MCBSP module 5
83	Reserved	IVA2_IRQ[38]	Reserved	Reserved

Table 14-3. IVA2.2 Interrupt Mappings (continued)

EVT	Interrupts/Events	IVA2.2 Pin Name	Interrupt Source	Interrupt Description
84	L3_IVA2_Error_IRQ	IVA2_IRQ[39]	L3	L3 interconnect out-of-band error interrupt
85	D2DSPINT	IVA2_IRQ[40]	3G modem	For speech data processing
86-87	Reserved	IVA2_IRQ[41-42]	Reserved	Reserved
88	GPIO6_IVA2_IRQ	IVA2_IRQ[43]	GPIO6	GPIO module 6
89	SDMA_IRQ_0	IVA2_IRQ[44]	SDMA	sDMA interrupt request 0
90	SDMA_IRQ_1	IVA2_IRQ[45]	SDMA	sDMA interrupt request 1
91-92	Reserved	IVA2_IRQ[46-47]	Reserved	Spare interrupts (provision)
93	Reserved	N/A (internal)	N/A	N/A
94	Reserved	N/A (internal)	N/A	N/A(1)
95	VIDEO_INT	N/A (internal)	Video accelerator	Video accelerator interrupt (managed by VIDEOSYSC)
96	INTERR	N/A (internal)	DSP INT CTL	Dropped CPU interrupt event
97	EMC_IDMAERR	N/A (internal)	EMC	Invalid IDMA parameters
98	Reserved	N/A (internal)	Reserved	Reserved
99	Reserved	N/A (internal)	N/A	N/A
100	EFIINTA	N/A (internal)	EFI	EFI interrupt from side A
101	EFIINTB	N/A (internal)	EFI	EFI interrupt from side B
102-112	Reserved	N/A (internal)	N/A	N/A
113	EMC_ED	N/A (internal)	DSP PMC	Single bit error detected during DMA read
114-115	Reserved	N/A (internal)	N/A	N/A
116	UMC_ED1	N/A (internal)	DSP UMC	Corrected bit error detected
117	UMC_ED2	N/A (internal)	DSP UMC	Uncorrected bit error detected
118	PDC_INT	N/A (internal)	DSP PDC	PDC sleep interrupt
119	SYS_CMPA	N/A (internal)	DSP PMC	SYS CPU memory protection fault
120	PMC_CMPA	N/A (internal)	DSP PMC	CPU memory protection fault
121	PMC_DMPA	N/A (internal)	DSP PMC	DMA memory protection fault
122	DMC_CMPA	N/A (internal)	DSP DMC	CPU memory protection fault
123	DMC_DMPA	N/A (internal)	DSP DMC	DMA memory protection fault
124	UMC_CMPA	N/A (internal)	DSP UMC	CPU memory protection fault
125	UMC_DMPA	N/A (internal)	DSP UMC	DMA memory protection fault
126	EMC_CMPA	N/A (internal)	DSP EMC	CPU memory protection fault
127	EMC_BUSERR	N/A (internal)	DSP EMC	BUSERR interrupt

(1) EVT94 is used by the DSP to generate an interrupt to the sequencer by means of the HOST_MBX in the SEQ_IRQSTATE register.

The interrupts generated by the WUGEN module (interrupts from device peripherals) are programmable using a set of registers accessible by the user.

Events related to these external interrupts can be masked by writing to the IVA2.WUGEN_MEVTSET0 or IVA2.WUGEN_MEVTSET1 registers. The interrupts can also be individually unmasked by using the IVA2.WUGEN_MEVTCLR0 or IVA2.WUGEN_MEVTCLR1 registers. By default (after power on), these external interrupts are masked in WUGEN.

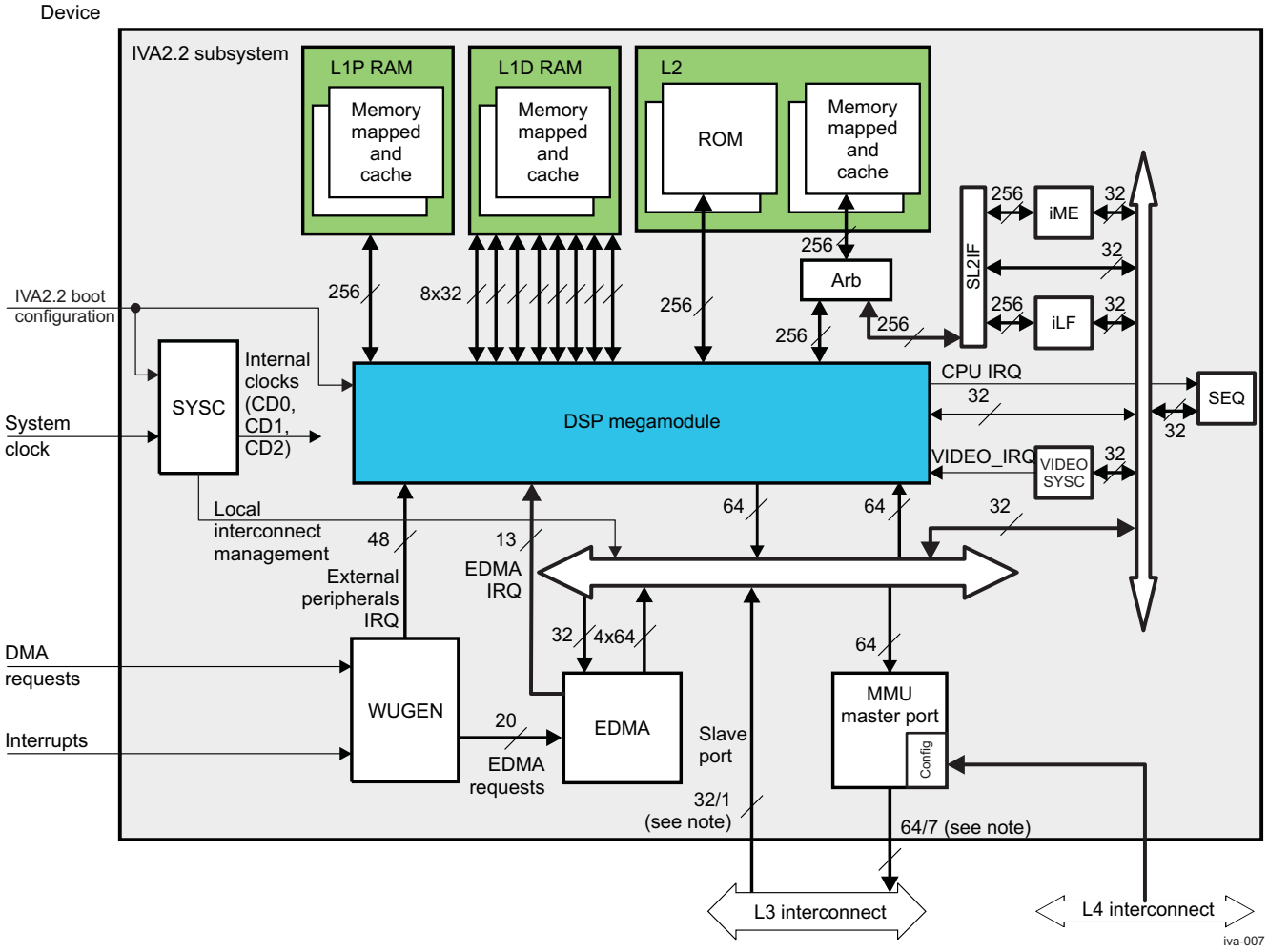
All other interrupts are directly managed by the IVA2.2 DSP INTC, and the DSP registers are set. For information about the DSP INTC, see the documents listed in Section 14.3.1.8, *Other DSP Reference Documents*.

14.3 IVA2.2 Subsystem Functional Description

The IVA2.2 subsystem is composed of a DSP megamodule coupled with several submodules that enable its integration in the device architecture. The IVA2.2 subsystem provides one slave port and one master port; both ports are connected to the L3 interconnect.

Figure 14-7 is a block diagram of the IVA2.2 subsystem.

Figure 14-7. IVA2.2 Subsystem Block Diagram



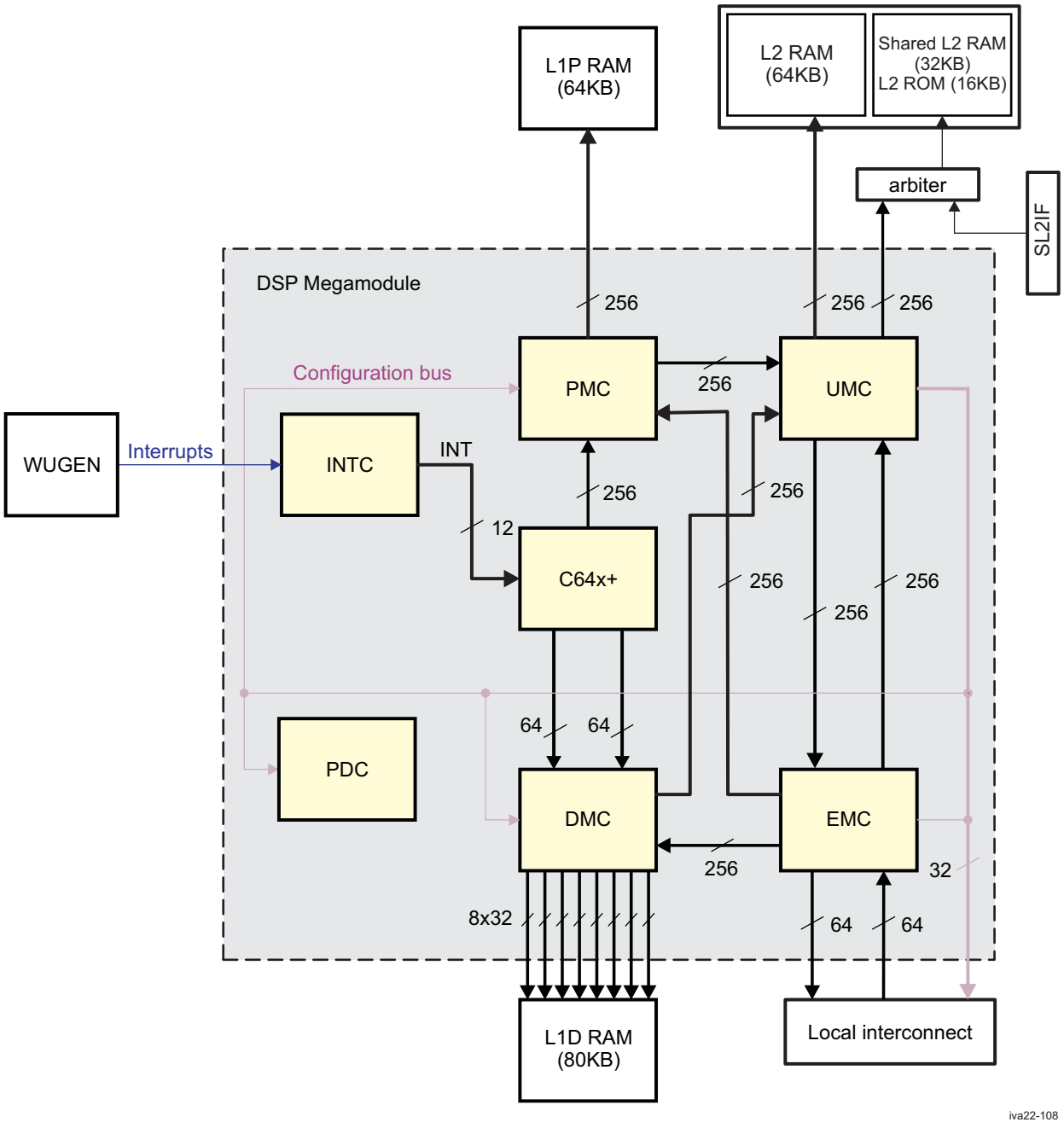
NOTE: This indicates the number of threads for IVA2.2 master/slave port interrupts. For details, see Chapter 5, *Interconnect*.

14.3.1 DSP Megamodule

The C64x+ DSP megamodule is a class of derivative sections of the generalized embedded megamodule. DSP megamodule is a hardware-configurable megamodule module that comprises a version of the C64x+, an L1 program memory controller (PMC), an L1 data memory controller (DMC), a unified memory controller (UMC), an extended memory controller (EMC), an INTC, and a power-down controller (PDC).

Figure 14-8 is a block diagram of the DSP megamodule.

Figure 14-8. DSP Megamodule Block Diagram



14.3.1.1 DSP Overview

The C64x+ is an extension of the first C64x DSP section (also named Kelvin).

The C64x features the following:

- 32-bit fixed-point media processor
- VLIW architecture
- 8 instructions/cycle, 8 execution units
  - Optimized instruction set for video and image processing
  - Eight 8 x 8 or 16 x 16 MAC per cycle
  - Eight SAD per cycle

- Eight interpolations (a + b + 1) >>1 per cycle
- Two (32-bit x 32-bit > 64-bit) multiply operations per cycle

In addition to existing C64x features, the C64x+ includes the following:

- SPLOOP
- Compact instructions
- Extended function support
- Exception handling
- Privilege mode support
- HLOS time-stamp counter register

C64x+ operates to up to 365 MHz in the context of IVA2.2.

CAUTION

Clock configurations depend on core voltage, and maximum clock frequencies might not apply to production.

For information about C64x+ DSP, see the C64x+ reference manual (Section 14.3.1.8).

14.3.1.2 Program Memory Controller Overview

The PMC is the DSP megamodule component that delivers program fetch packets when they are requested by the DSP.

The PMC supports the following features:

- One 256-bit fetch packet per cycle (sustainable)
- 32KB 0-wait state least-significant bit (LSB)-banked SRAM
- L1 program cache controller - 32KB maximum
- Software-programmable allocation of SRAM to cache or memory-mapped SRAM
- DMA transfer from/to SRAM
- Fair priority-based arbitration between DSP, DMA, and cache controller for access to the SRAM
- Block and global program-initiated cache coherence support (invalidate)
- Freeze mode support

In the IVA2.2 subsystem, the PMC controls the 32-KB SRAM typically used as cache RAM. However, the allocation of SRAM can be software-configured to keep a section of the memory as memory-mapped SRAM.

For more information, see Section 14.4.8, *Memory Management*.

14.3.1.3 DMC Overview

The DMC is the DSP megamodule component that reads or writes data from/to local memories, as requested by the DSP.

The DMC supports the following features:

- Two 64-bit memory accesses per cycle (sustainable)
- A memory access pair can be any combination of read and write, with no incurred penalty.
- 80KB 0-wait state LSB-banked SRAM
- L1 data cache controller (capabilities listed in Table 14-90) - 32KB maximum
- Software-programmable allocation of SRAM to cache or memory-mapped SRAM
- DMA transfer from/to SRAM
- Fair priority-based arbitration between DSP, DMA, and cache controller for access to the SRAM
- Hardware coherence maintenance with L2 (snooping)
- Block and global program-initiated cache coherence support (write-back, invalidate, and

write-back-invalidate)
- Freeze and bypass mode support
- Per-page memory-protection attribute check support

The DMC operates to up to 365 MHz in the context of the IVA2.2. The DMC always operates at the same frequency as the DSP.

CAUTION

Clock configurations depend on core voltage, and maximum clock frequencies might not apply to production.

14.3.1.4 UMC Overview

The UMC is the DSP megamodule component that reads or writes program and data from/to memory, as requested by the DMC and/or the PMC.

The UMC supports the following features:
- 64KB low-latency state SRAM (first port)
- 32KB low-latency state memory-mapped only SRAM (second port)
- 16KB low-latency ROM (second port)
- L2 unified cache controller (capabilities listed in Table 14-91) - 64KB maximum
- Software-programmable allocation of SRAM to cache or memory-mapped SRAM
- DMA transfer from/to SRAM and from ROM
- Fair priority-based arbitration between DSP, DMA, and cache controller for access to the SRAM
- Block and global program-initiated cache coherence support (write-back, invalidate, and write-back-invalidate)
- Freeze and bypass modes support
- Per-page memory protection attribute check support
- Emulation support
- Long-distance access support (noncacheable data accesses)
- Page-based memory automatic power-down (to clock-off state) and wake-up support

The UMC contains the interfaces to L2 memory, the L1D, the L1P, and the EMC. The EMC houses the interfaces to the EDMA and internal data memory access (IDMA) engines, and thus allows the UMC and ultimately the CPU to communicate with peripherals and external memory.

The UMC has two L2 memory ports, each of which is 256 bits wide. Although accesses to the two ports are initiated serially for each requestor, only one new access is initiated per UMC clock cycle, to avoid memory-bank conflicts between the ports.

The UMC has three requestor ports: the PMC, the DMC, and the EMC. Peripheral configuration and DMA/IDMA interfacing are handled by the EMC. The PMC interface to the UMC consists of a 256-bit read-only path (L1P fetch only). The DMC and the EMC interface to the UMC, and each has separate 256-bit read and write paths of its own.

The UMC operates to up to 182.5 MHz in the context of the IVA2.2. The UMC always operates at half the frequency of the DSP.

CAUTION

Clock configurations depend on core voltage, and maximum clock frequencies might not apply to production.

14.3.1.5 EMC Overview

The EMC is the megamodule component that reads or writes program and data external memory and configuration registers, as requested by the UMC.

The EMC supports the following features:

- IDMA
- 64-bit slave port for accesses from an external DMA engine and/or host to L1 or L2 memory
- 64-bit master port to serve accesses from the UMC to external memory or configuration registers

The EMC operates to up to 182.5 MHz in the context the of the IVA2.2. The EMC can operate at half or one third the frequency of the DSP.

CAUTION

Clock configurations depend on core voltage, and maximum clock frequencies might not apply to production.

14.3.1.6 Memory Protection Overview

The DSP megamodule memory protection architecture divides the memory map into pages. Each page has an associated set of permissions.

Memory protection provides many benefits to a system:

- Protects operating system data structures from poorly behaving code
- Helps in debugging by providing information about illegal memory accesses
- Prevents unauthorized access to sensitive data
- Allows the OS to enforce clearly defined boundaries between supervisor and user mode accesses, leading to greater system robustness

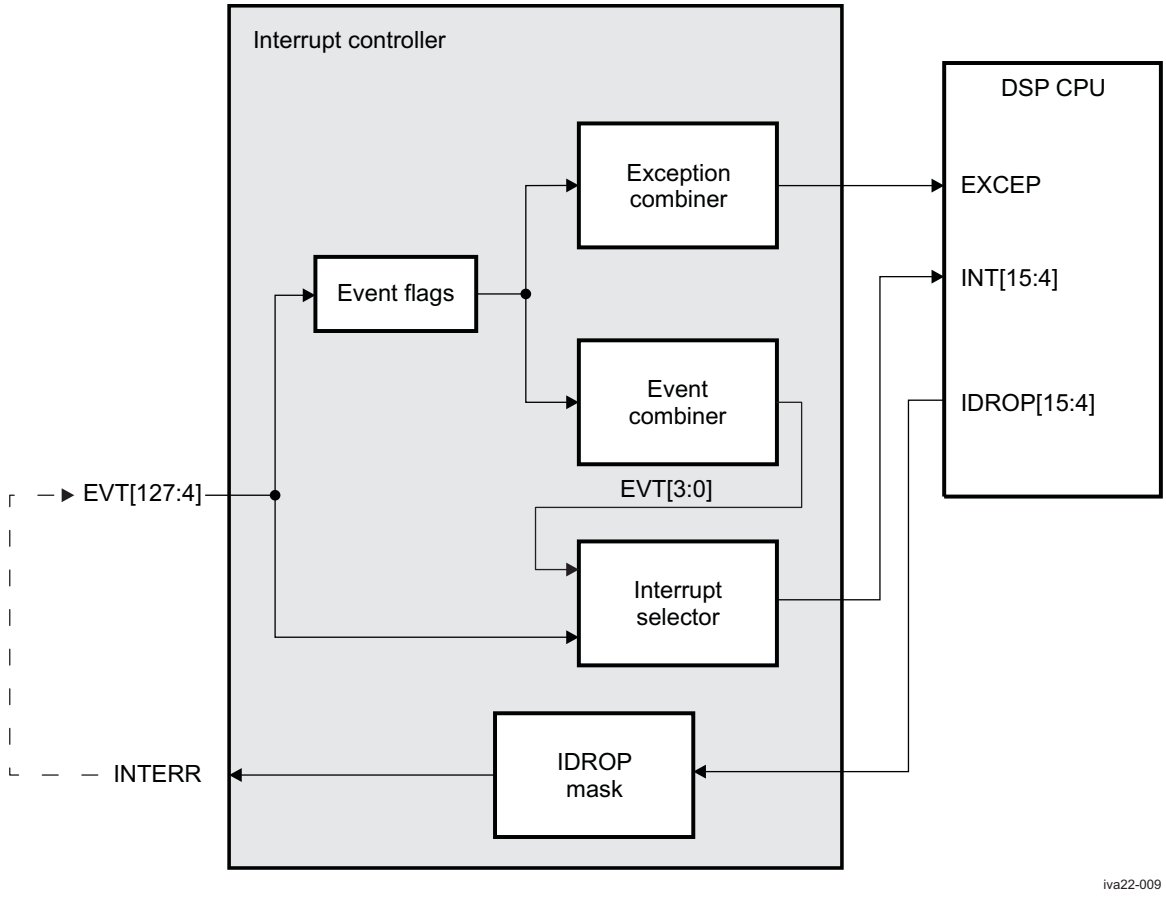
Memory protection is implemented for the PMC, DMC, UMC, and EMC, and also for the IDMA and EDMA modules.

14.3.1.7 INTC

The DSP megamodule INTC detects, potentially combines, and routes up to 128 system events (internal and external) to the DSP CPU interrupt lines. Table 14-5 lists the global interrupt mappings of the IVA2.2 subsystem (internal and external interrupts).

The DSP CPU has 12 maskable interrupts and one exception input. The INTC includes an interrupt selector, an exception combiner, and an event combiner. The interrupt selector allows the routing of any of the 128 system events (or a combination of them) to the 12 maskable interrupts of the DSP CPU, and software determines the priorities of those system events. To handle potential conflicts, the 12 CPU interrupts have fixed priorities. The exception combiner allows the combination of any of the 128 system events to the single exception input of the DSP CPU (see Figure 14-9).

Figure 14-9. DSP Megamodule INTC Block Diagram



Note: Not all of the 128 event inputs of the INTC are connected to an internal or external event line. Table 14-3 lists the global interrupt mapping of the DSP INTC. Some interrupts at the IVA2.2 boundary are reserved for future use or are not used by the IVA2.2 subsystem. For information about the DSP core INTC, see the C64x+ DSP documents listed in Section 14.3.1.8, *Other DSP Reference Documents*.

14.3.1.7.1 Event Type

The interrupt controller provides three types of interrupt sources to the DSP CPU:

- Single event
  - A single system event can be directly routed to a CPU interrupt input.
  - A dropped system event is supported through the CPU dropped interrupt reporting mechanism; that is, if the CPU misses the interrupt input, the system event is missed.
- Combined event
  - Up to four event combiners
  - Each combiner allows up to 32 system events to be logically ORed into a single event that can be routed to the DSP CPU.
- Exception event: This event is considered a single event, but it is directly connected to the exception input of the DSP CPU.

14.3.1.7.2 Event Behavior

- Single event source: The interrupt selector routes 1 of 124 events to a DSP CPU interrupt, as programmed in the interrupt selector registers (IC.INTMUXj registers (where j is 1 to 3). Logic in the CPU and the INTC combine to detect instances where an interrupt request is asserted before an earlier interrupt request was serviced. The INTC records the interrupt number of the first interrupt and keeps this information until directed to release the interrupt, either through reset or by application software. This record can be used as an additional system event to notify the application of an interrupt failure. Interrupts that qualify for dropped detection are defined through an IDROP mask that sets the IC.INTDMASK register. The masked IDROP event output (EVT96; see Table 14-3) is available as a system event that can be selected as either a DSP CPU interrupt or an exception event.
- Combined source: The event combiners create a combined event from the logical OR of 32 system-event flags qualified by a mask provided through programmable registers (IC.EVTMASKi where i = 0 to 3). The combination of the event flags creates an event that is asserted when any of the event flags included in its generation is active. Software must clear the event flags (IC.EVTCLRi where i = 0 to 3) to deassert a combined event.

  Note: The use of event flags makes it impossible for the CPU to detect the dropping of independent system events; therefore, dropped interrupt capability is not directly supported for combined events.

  The interrupt event combiners create four shared interrupt sources:
  - EVT0: Logical OR of EVTFLAGi[31:04] (i = 0) masked by EVTMASKi[31:04] (i = 0)
  - EVT1: Logical OR of EVTFLAGi[63:32] (i = 1) masked by EVTMASKi[63:32] (i = 1)
  - EVT2: Logical OR of EVTFLAGi[95:64] (i = 2) masked by EVTMASKi[95:64] (i = 2)
  - EVT3: Logical OR of EVTFLAGi[127:96] (i = 3) masked by EVTMASKi[127:96] (i = 3)

  EVTFLAGi are the event flag registers and EVTMASKi are the event mask registers. These combined events are presented to the interrupt selection logic.
- Exception event source: As with the event combiner, the exception combiner allows multiple system events to be grouped as a single event input to the CPU, and the combiner provides mask registers to remove undesirable events. Because only one exception is input to the CPU, all mask registers work together to combine up to 128 events as a single EXCEP output. This lets the CPU service all available system exceptions.

  The shared exception source is created as follows:

  EXCEP: Logical OR of EF[127:04] masked by XM[127:04]

  Where EF[i * 32 + j] = EVTFLAGi[j] and XM[i * 32 + j] = EXPMASKi[j] (i = 0, 1, 2, 3 and j = 0 to 31).

The logic is similar to that of the event combiners, except that only one combined event is routed to EXCEP.

14.3.1.7.3 Event Detection

The INTC contains a set of status and control registers to manage the status of system events received by the controller. These include set, flag, and clear registers covering all 128 system events. Enabling of the events is managed in the CPU for direct mapped interrupts, in the event combiner for combined events, and in the exception combiner for system exceptions. Events to the interrupt controller can be enabled/disabled only at the event source.

The event flag registers (IC.EVTFLAGi where i = 0 to 3) capture every system event received, regardless of its destination: event combiner, exception combiner, or interrupt selector. Events that are not masked by either the event combiner or the exception combiner are captured in the IC.EVTFLAGi (where i = 0 to 3) register corresponding to the event, and are used to determine when the combined event is generated. Events that are masked by either the event combiner or the exception combiner are captured in the IC.EVTFLAGi register, but do not affect when the DSP CPU interrupt is generated.

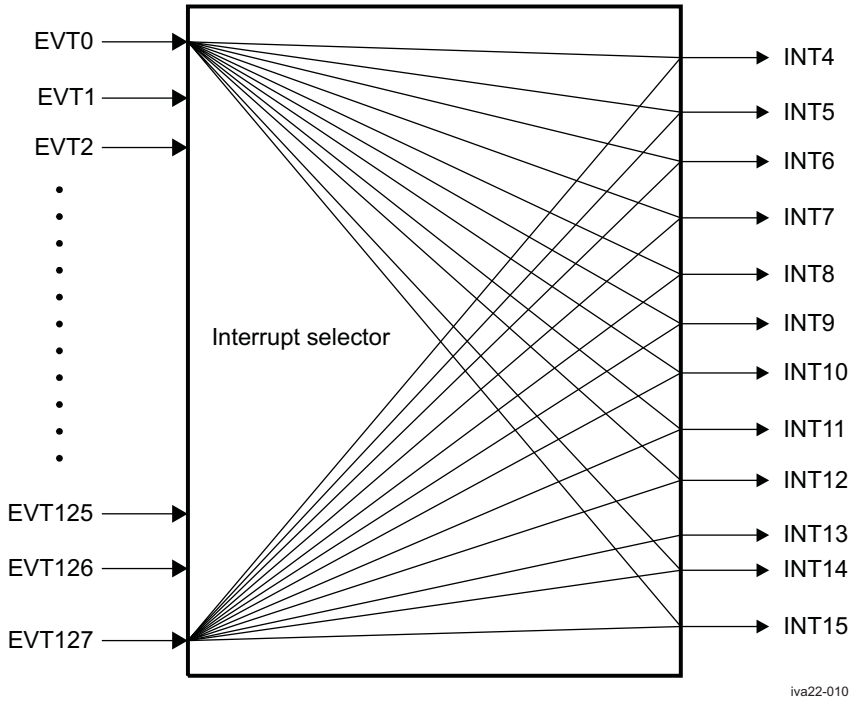
The event flags in the IC.EVTFLAGi registers retain the value of 1 for any event received. These registers are read-only and must be cleared through the write-only IC.EVTCLRi registers. The IC.EVTSETi registers can be used to manually set bits in the IC.EVTFLAGi registers, including those that are masked.

Note: Because external events are maintained by the peripheral until they are explicitly cleared by software, drop event detection does not work for external events (interrupts from peripherals external to the IVA2.2 subsystem).

14.3.1.7.4 Event Selection

The DSP CPU has 12 available maskable interrupts. The interrupt selector allows any of the 128 system events, either from the event inputs or from the event combiners, to be routed to any of the 12 CPU interrupt inputs (see Figure 14-10).

Figure 14-10. Interrupt Selector Block Diagram



The user can choose which of the 128 input events is mapped to each of the 12 CPU interrupts by writing the event number in the bit field corresponding to the CPU interrupt in the INTMUXj register. CPU priority is fixed. This event -> interrupt mapping allows software to define the priority of the event. For more information, see Section 14.4, *IVA2.2 Subsystem Basic Programming Model*.

14.3.1.7.5 Event Combination

The event combiner allows multiple system events to be combined as single event for routing to the interrupt selector. This allows the CPU to service all available system events, even though only 12 are available.

A set of event mask registers (IC.EVTMASKi where i = 0 to 3), is used to program the event combiner. These registers allow up to 32 events to be combined as a single event output that is used as a single DSP CPU interrupt. The event mask bits in the EVTMASKi registers act as enablers for the received system events combined on the event outputs. There are four event outputs to the interrupt selector (EVT[3:0]).

By default, every system event is unmasked and combined with its associated EVTx. To mask out an event source (for example, to disable an event from being combined), the corresponding mask bit (the IC.EVTMASKi[y] EMy bit for EVTy) must be set to 1.

Note: Because the event masks for events 0 through 3 are combined events and thus cannot contribute to the generation of a combined event, they are reserved.

For more information, see Section 14.4.7, *Interrupt Management*.

14.3.1.7.6 Interrupt Event Error

The INTC can generate a system event internally routed to system event input EVT96. This event is generated when a DSP CPU interrupt is received while the interrupt flag (IFR, internal DSP register) is already set in the CPU. This signals possible problems in the code, such as whether interrupts were disabled for an extended time, or whether pipelined (noninterruptible) code sections were too long.

Note: The same strategy of register settings (events, event set, event clear, event mask) is used in the WUGEN (which is not really an INTC), which enables defining wake-up events so that they can wake up the IVA2.2 subsystem. For more information, see Section 14.3.10, *Wake-Up Generator*.

For more information, see Section 14.4.7, *Interrupt Management*. See Section 14.4, *IVA2.2 Subsystem Basic Programming Model*, for information on associated programming.

14.3.1.7.7 PDC Overview

The DSP megamodule PDC allows power management under software control. Different power-down states are defined and can be reached during a period of DSP inactivity. The PDC ensures handshakes with DSP modules so that they go to power-down state in the correct order, on request from the user, and publishes the relevant standby status to the IVA2.2 SYSC module.

The power-down attributes of the DSP components are mapped through the PDC, through the SYS.PDCCMD register.

For information about power-down settings, see Section 14.4.9.3*, Power-Down and Wake-Up Management*.

14.3.1.8 Other DSP Reference Documents

For more information about the DSP megamodule architecture, the instruction set, and the DSP core interrupt controller, and for a complete description of the DSP memory-mapped registers, see the following documents (www.ti.com):

- *TMS320C6000 DSP Peripherals Overview Reference Guide* (TI literature number SPRU190) describes the peripherals available on the TMS320C6000 DSPs.
- *TMS320C64x Technical Overview* (TI literature number SPRU395) introduces the TMS320C64x DSP and discusses the application areas enhanced by the TMS320C64x VelociT.
- *TMS320C64x+ DSP Megamodule Reference Guide* (TI literature number SPRU871) describes the C64x+ megamodule peripherals.
- *TMS320C64x/C64x+ DSP CPU and Instruction Set Reference Guide* (TI literature number SPRU732)
- *TMS320C6000 Programmers Guide* (TI literature number SPRU198) describes ways to optimize C and assembly code for the TMS320C6000 DSPs, and includes application program examples.
- *TMS320C64x to TMS320C64x+ CPU Migration Guide* application note (TI literature number SPRAA84A)

14.3.2 DMA Engines

14.3.2.1 EDMA

The IVA2.2 subsystem has an EDMA to transfer instructions and data from/to any external memory connected to the L3 interconnect to/from the DSP megamodule L1D and L2 and, potentially, L1P (if not configured as full cache) local memories. The EDMA can also perform transfers from external memory to external memory and from DSP megamodule internal memory to DSP megamodule internal memory, with some performance loss caused by resource sharing between the read and write ports. For DSP megamodule internal memory to internal memory, use the IDMA.

For the IDMA functional description and a description of IDMA relative programming, see Section 14.3.2.3, *IDMA*, and Section 14.4.4.2, *Internal Memory-to-Memory Transfer (IDMA)*, respectively.

The EDMA is based on two primary components:

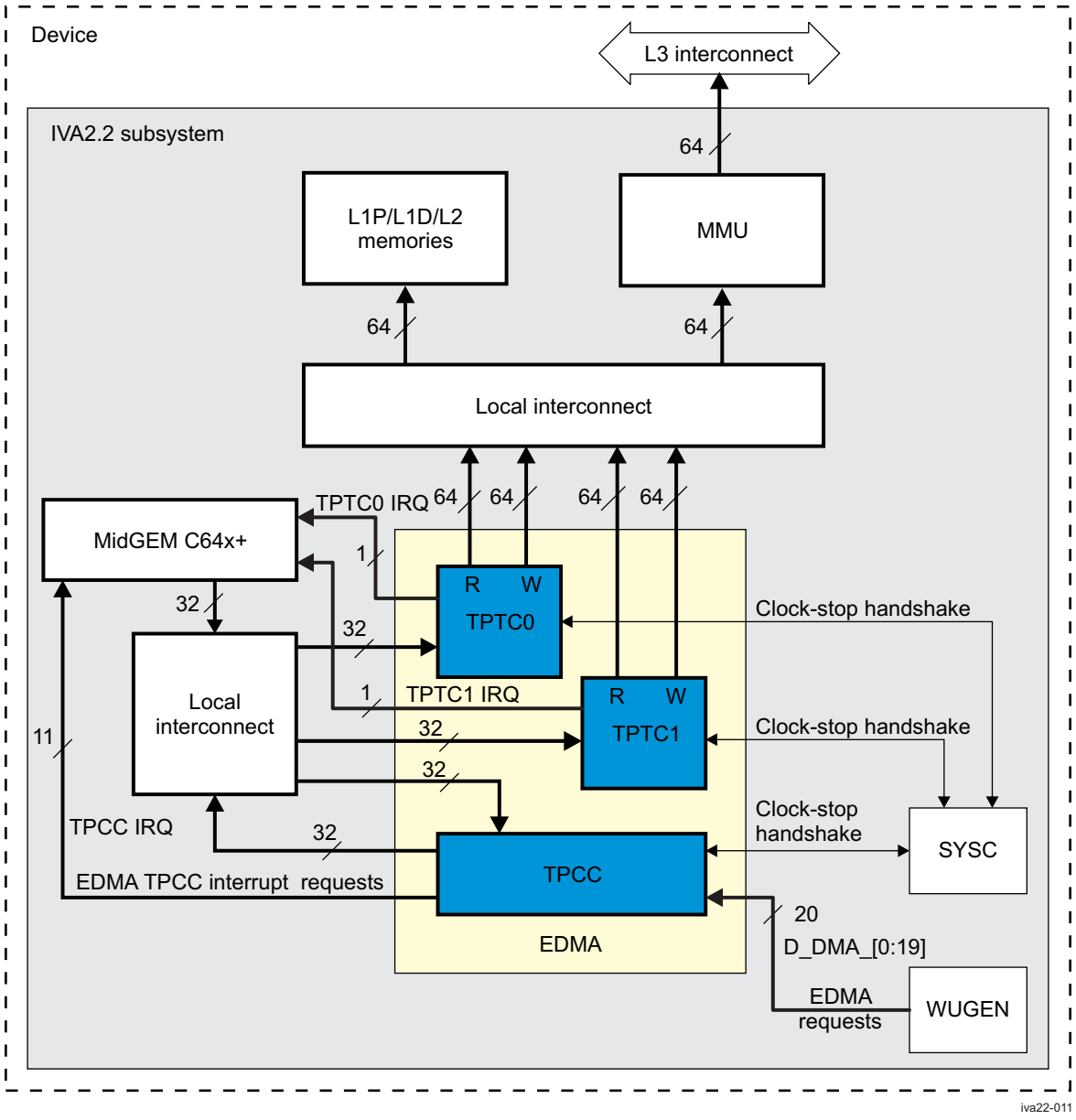
- Third-party DMA channel controller (TPCC)
- Third-party DMA transfer controller (TPTC)

  There are two instances of the TPTC in the IVA2.2 subsystem.

Figure 14-11 shows how the EDMA is integrated in the IVA2.2 subsystem:

- Code running on the DSP can configure the EDMA; through the DSP megamodule configuration port and the local interconnect, the code can program DMA transfers and software-trigger them by writing to the TPCC configuration registers.
- Preprogrammed DMA transfers can be triggered by external events, referred to in this chapter as DMA requests.
- The TPCC schedules DMA transfers to the TPTC DMA engines (TPTC0 and TPTC1) through dedicated local interconnect 32-bit configuration ports.
- Each TPTC issues concurrent traffic on the local interconnect through dedicated read and write 64-bit ports.
- Interrupts generated by the EDMA are routed to the DSP INTC.
- Power-management handshakes are exchanged between EDMA components and the SYSC module.

Figure 14-11. IVA2.2 EDMA Overview



14.3.2.1.1 Third-Party Channel Controller

The TPCC is the DMA transfer scheduler responsible for scheduling, arbitrating, and issuing user-programmed transfers to the two TPTCs.

14.3.2.1.1.1 TPCC Features

The TPCC features are as follows:

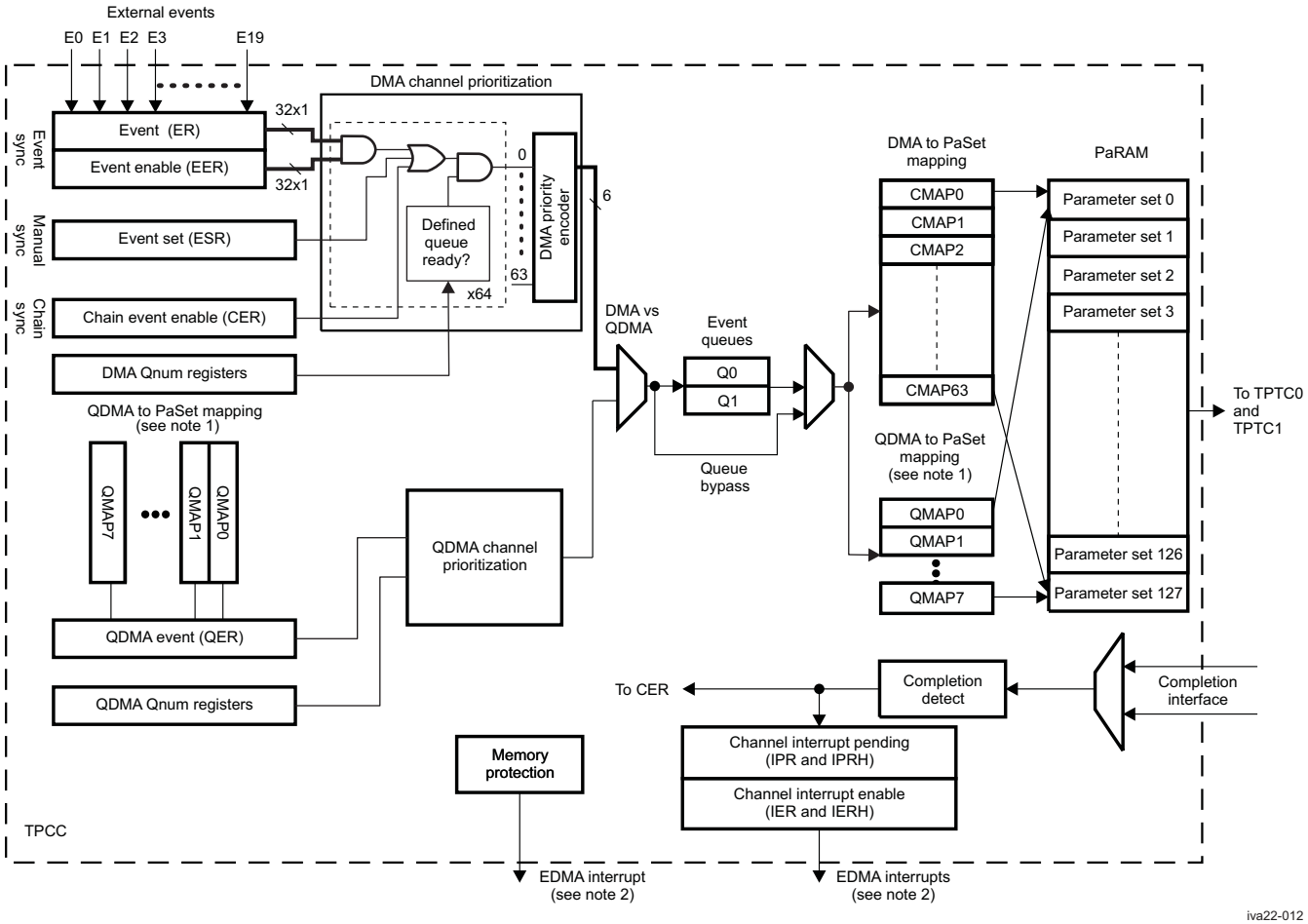
- Parameter RAM (PaRAM entires) holding up to 128 transfer contexts, with the following capabilities:
  - Ping-pong and circular buffering
  - Channel chaining
  - Auto-reloading

Each PaRAM entry can be used as a DMA entry (up to 64), QDMA entry (up to 8), or link entry (remaining):

- Up to 64 DMA logical channels:
  - 64 channels can be triggered explicitly by the DSP CPU (DMA entry).
  - 64 channels can be triggered by 20 external events (see Table 14-2).
  - All channels can be triggered by completion of a previous transfer in a user-programmed chain of transfers (linked entry).
  - 8 channels can be triggered automatically and indirectly by the CPU using the IDMA, reducing CPU offload for DMA configuration (QDMA entry).
- 12 interrupt lines connected to the DSP CPU
- Boundaries for synchronization with the CPU are programmable.
- Two queues holding events waiting for submission to TPTC
- Memory protection
- Each transfer can be programmed to a priority level (three possible levels).

Figure 14-12 is a block diagram of the TPCC.

Figure 14-12. TPCC Block Diagram



Notes:

1. Although it is depicted twice in Figure 14-12, there is only one physical register set for the QDMA to PaRAM set mapping block.
2. For more information, see Table 14-3.

A channel is a specific event that can cause a transfer to be submitted to the TPTCs as a transfer request. The TPCC supports up to 64 DMA channels and up to 8 QDMA channels. These channels are identical, except for the ways they are triggered:

- DMA channels can be triggered by external events (such as McBSP TX Evt and McBSP RX Evt) by the software writing 1 to a given bit location or channel, by the event set register, or through chaining.
- QDMA channels are triggered automatically (auto-triggered) by the CPU using the IDMA. QDMAs allow a minimum number of linear writes (optimized for the DSP IDMA feature) to be issued to the TPCC to force a series of transfers to occur.

The TPCC arbitrates among all pending DMA/QDMA events with a fixed 64:1 and 4:1 priority encoder for DMA and QDMA events, respectively (a low channel number corresponds to a high priority). DMA events are always higher priority than QDMA events. The higher-priority event is placed in the event queue to await submission to the transfer controller, which occurs at the earliest opportunity. Each event queue is serviced in FIFO order, with a maximum of 16 queued events per event queue. If more than one TPTC is ready to be programmed with a transmission request (TR), the event queues are serviced with fixed priority, Q0 higher than Q1. When an event is ready to be queued and both the event queue and the TC channel are empty, the event bypasses the event queue and goes directly to the PaRAM processing logic for submission to the appropriate TC. If the TR bus/PaRAM processing is busy, the bypass path is not used. The bypass is not used to dequeue for a higher-priority event.

Events are extracted from the event queue when the TPTC is available for a new TR to be programmed into the TPTC (signaled with the empty signal, indicating an empty program register set). As an event is extracted from the event queue, the associated PaRAM entry is processed and submitted to the TPTC as a TR. The TPCC updates the appropriate counts and addresses in the PaRAM entry in anticipation of the next trigger event for that PaRAM entry. The PaRAM entry consists of eight words of DMA context, including source address, destination address, count, indexes, etc.

14.3.2.1.1.2 DMA Versus QDMA

The only difference between a QDMA and a DMA transfer is the specific means of generating/recognizing TR synchronization. From the user point of view, DMA and QDMA transfer types can be combined to perform various types of transfers.

DMA channel TR synchronization can be generated from one of three sources:

- Event-triggered: The event register (TPCC_ER) channel *n* bit is set as the result of an external event. An external event is latched in the event register. If the corresponding event is enabled through the event enable register (TPCC_EER), this is recognized as a TR synchronization.
- Manually triggered: The event set register channel *n* bit is set manually to the event set register (TPCC_ESR and TPCC_ESRH registers) for channel *n* (TPCC_ESR[*n*] E*n* bit or TPCC_ESRH[*n*] E*n* bit). Manually set events do not need to be enabled to be recognized as a TR synchronization.
- Chain-triggered: The chain event register channel *n* (TPCC_CER and TPCC_CERH registers) bit is set when a chaining completion code is detected on the completion interface for channel *n*. Chain events do not need to be enabled. If a chain trigger is detected, this is always recognized as a TR synchronization.

QDMA TR synchronization occurs one of two ways:

- Auto-triggered: The QDMA event register channel *n* (TPCC_QER and TPCC_QERH[*n*] E*n*) bit is set when a CPU write address matches the address defined by the QDMA mapping register for channel *n* (TPCC_QCHMAPj) and the QDMA channel is enabled (TPCC_QEER[*n*] E*n* bit field).
- Link-triggered: The TPCC_QER[*n*] E*n* bit is set when a link update is performed on a PaRAM address that matches the TPCC_QCHMAPj setting, and the TPCC_QEER[*n*] E*n* bit is set.

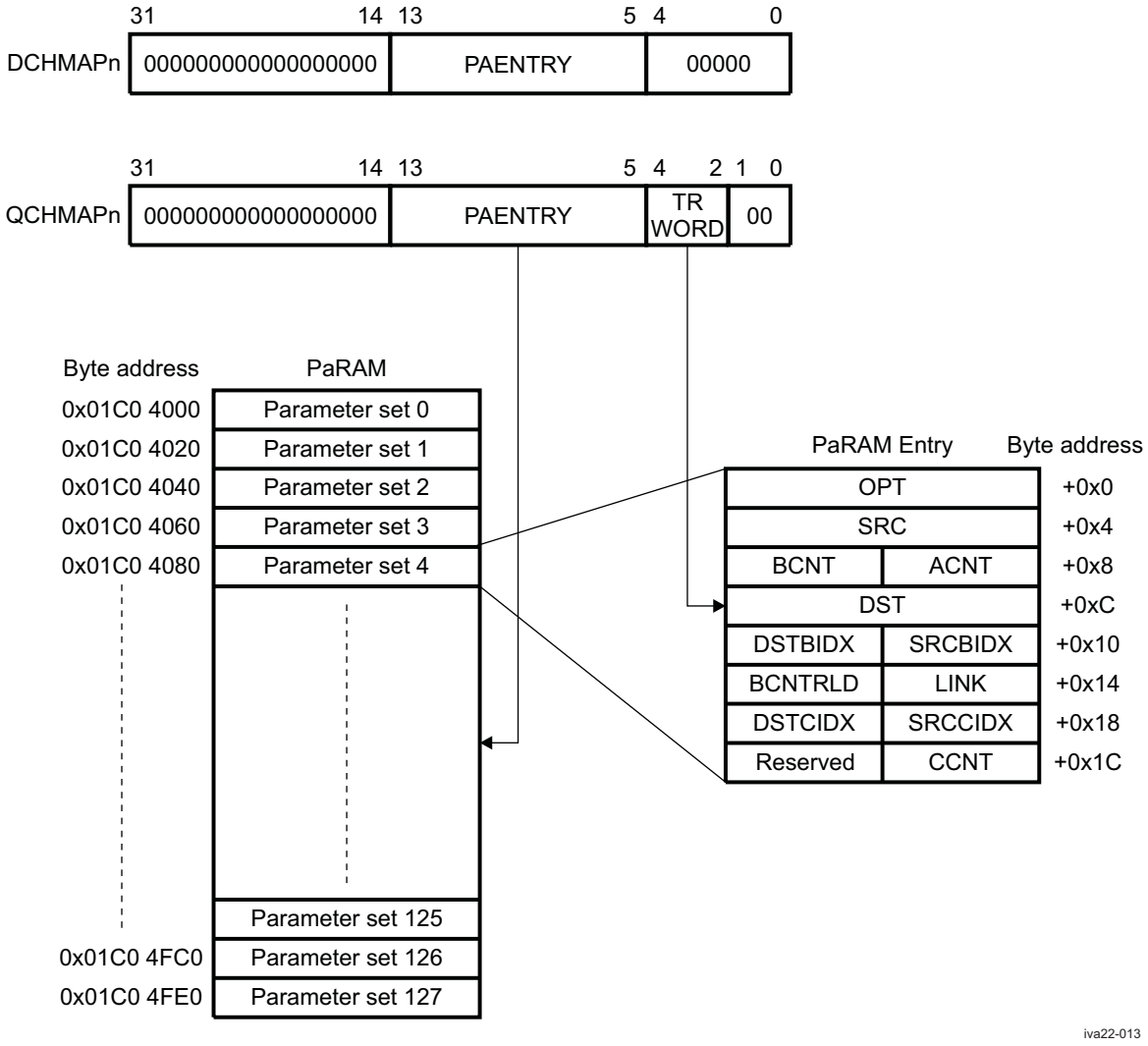
There is a subtle difference between each trigger type and the associated enablers. Event assertion always results in the TPCC_ER[*n*] E*n* bit being set, regardless of the state of the enable (TPCC_EER.E*n*). It is recognized as a TR synchronization only if enabled. Chain triggering always sets the TPCC_CER[*n*] E*n* bit and is always treated as a TR synchronization. Manual triggering always sets the TPCC_ESR[*n*] E*n* bit and is always treated as a TR synchronization. Auto-triggering (QDMA) sets only the event register TPCC_QER[*n*] E*n* bit only if the corresponding event is enabled (TPCC_QEER[*n*] E*n*); when the TPCC_QER[*n*] E*n* bit is set, it is always treated as a TR synchronization.

For more information, see Section 14.4, *IVA2.2 Subsystem Basic Programming Model*.

14.3.2.1.1.3 DMA/QDMA Channel Mapping and PaRAM Entry

The DMA and QDMA channel mapping registers (TPCC_DCHMAPi and TPCC_QCHMAPj) allow each DMA and QDMA channel to be mapped to arbitrary locations in the PaRAM memory map. The entry is designated with a 9-bit PAENTRY (TPCC_DCHMAPi[13:5] PAENTRY and TPCC_QCHMAPj[13:5] PAENTRY bit fields) PaRAM entry number that defines the entry number in a 128-entry maximum PaRAM (see Figure 14-13). TPCC_QCHMAPj[4:2] TRWORD points to the trigger word of the PaRAM entry defined by PAENTRY. A write to the trigger word results in a QDMA event being recognized.

Figure 14-13. DMA/QDMA Channel Mapping and PaRAM Entry



Note: Each parameter entry of PaRAM is organized as eight 32-bit words or 32 bytes, as shown in Figure 14-13.

The location of fields in each entry, as well as bit positions in the options field, must match the TR packet format as closely as possible to the requirements of the TPTC0 and TPTC1 programming. Each PaRAM entry consists of 16-bit and 32-bit parameters that correspond to the transfer geometry. For more information about the parameters, see Section 14.3.2.1.1.2, *DMA Versus QDMA.*

14.3.2.1.1.4 Types of TPCC Transfers

The TPCC streamlines transfers into an orthogonal transfer structure that is always defined by three dimensions. Of the three dimensions, only two synchronization types are supported: 1D synchronized transfers and 2D synchronized transfers; 3D synchronized transfers can be logically achieved by chaining.

The following terms are used:

- 1-dimension (1D) transfer or array: ACNT contiguous bytes transfer. For information about the 1D transfer programming model, see Section 14.3.2.1.1.5, *Transfer Synchronization*.
- 2-dimension (2D) transfer or frame: BCNT arrays of ACNT bytes transfer. Each array transfer in 2D transfer is separated from the other transfers by an index programmed through the TPCC.SBIDX or TPCC.DBIDX registers.
- 3-dimension (3D) or block: CCNT frames of BCNT arrays of ACNT bytes. Each transfer in the third dimension is separated from the previous transfer by an index programmed by the TPCC.SCIDX or TPCC.DCIDX registers. The reference point for the index depends on the synchronization type.

14.3.2.1.1.5 Transfer Synchronization

In a 1D synchronized transfer, each TPCC TR synchronization triggers the transfer of the first dimension of ACNT bytes, or one array of ACNT bytes. In other words, each TR packet consists of transfer information for only one array. Arrays can be separated by SBIDX and DBIDX, as shown in Figure 14-13, where the start address of array N equals the start address of array 1 plus SRC or DST BIDX. Frames can be separated by SCIDX and DCIDX. For 1D synchronized transfers, after the frame is exhausted, the address is updated by adding (S|D)CIDX to the beginning address of the last array in the frame.

Contrast this to the (S|D)CIDX update in 2D synchronized transfers, which are referenced from the beginning address of the first array in the previous frame. For information about parameter updates, see Section 14.3.14, *Error Reporting*.

In a 2D synchronized transfer, each TR synchronization triggers the transfer of two dimensions or one frame. In other words, each TR packet conveys information for one entire frame of BCNT arrays of ACNT bytes. Arrays can be separated by SBIDX and DBIDX, as shown in Figure 14-15. Frames can be separated by SCIDX and DCIDX. For 2D synchronized transfers, after a TR for the frame is submitted, the address is updated by adding (S|D)CIDX to the beginning address of the beginning array in the frame. Contrast this to the (S|D)BIDX update in 1D synchronized transfers. For information about parameter updates, see Section 14.3.14, *Error Reporting*.

14.3.2.1.1.6 DMA Channel Prioritization

If multiple trigger sources (event trigger/chain trigger/manual trigger) for a single channel are active at the same time, the TPCC services events in the following order:

1. Event trigger (through event register [ER])
2. Chain trigger (through chain event register [CER])
3. Manual trigger (through event set register [ESR])

14.3.2.1.1.7 Transfer Completion

Transfer completion can be used for two different mechanisms in the TPCC:

- Generating chained events
- Generating interrupts to an external processor

The underlying mechanism for both features is the same. The user programs the PaRAM Options field with a specific transfer completion code (TPCC_OPTm[17:12] TCC) and indicates whether that completion code is to be used to generate a chained event and/or to generate an interrupt on completion of a transfer (either or both can be done). The user can selectively program whether completion signaling is enabled for the final TR of a PaRAM set, for all except the final TR of a PaRAM set, or for all TRs of a PaRAM set. The specific TCC value (6-bit binary value) programmed by the user dictates which bit of the 64-bit CER and/or interrupt pending register (IPR) is used.

There is no direct correlation between the TCC value used for a specific PaRAM entry and the channel number for that entry. Software controls the allocation of TCC values. There is always a 1:1 relationship between the TCC value and the IPR bit and/or the CER bit set when that transfer completes. For example, completion of the PaRAM entry for channel 0 can chain-trigger the PaRAM entry for channel 1.

14.3.2.1.2 Third-Party Transfer Controller

The TPTC is the DMA transfer engine that generates transfers as programmed in dedicated working registers, using two dedicated master ports: a read-only port and a write-only port.

Note: The port data bus width of the instances of the TPTC is fixed at 64 bits.

14.3.2.1.2.1 TPTC Features

The TPTC features are as follows:

- Pipelined transfers (multiple inflight transfers)
- Background programming
- 2D or 1D transfer
- Increment addressing modes
- Clock-stop handshaking with the SYSC controller
- Programmable tracking of transfer completion
- 2D qualifications of transfers to allow 2D bandwidth optimization

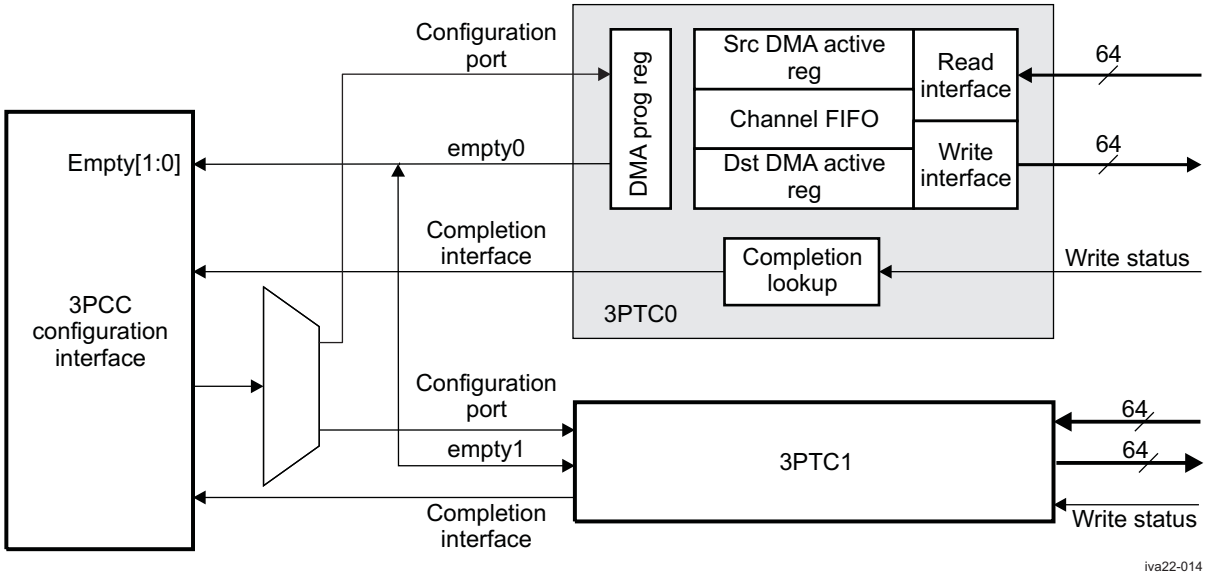
Two instances of the TPTC generate concurrent traffic on the IVA2.2 local interconnect. Each TC controller consists of the following components:

- DMA program register set: Stores the context for the DMA transfer that is loaded into the active register set on completion of the current active register set. The CPU or TPCC programs the program register set, not the active register set. For typical stand-alone operation, the CPU programs the program register while the TC services the active register set. The program register set includes ownership control that synchronizes the CPU software and the DMA.
- Source DMA active register set: Stores the context (src/dst/cnt/etc) for the DMA transfer request in progress in the read controller. The active register set is split into independent source and distant register halves, because the source local interconnect controller and distant local interconnect controller operate independently of each other.
- Distant DMA register set: Stores the context (src/dst/cnt/etc) for the DMA transfer request in progress, or pending, in the write controller. The pending register sets are required to allow the source controller to begin processing a new TR while the distant register set is still processing the previous TR.
- Channel FIFO: Temporary holding buffer for inflight data. The read return data of the source peripheral is stored in the data FIFO and then written to the destination peripheral by the write command/data bus.
- Read controller/local interconnect read interface: The local interconnect read interface issues optimally sized read commands to the source peripheral, based on a burst size of 64 bytes and the available landing space in the channel FIFO.
- Write controller/local interconnect write interface: The local interconnect write interface issues optimally sized write commands to the destination peripheral, based on a burst size of 64 bytes and available data in the channel FIFO.
- Completion interface: The completion interface sends completion information to the TPCC for posting interrupts in the TPCC.
- Configuration port: Slave interface that provides read/write access to program registers, and provides read access to all memory-mapped TC registers. For information about TPTC registers, see Section 14.5, *IVA2.2 Subsystem Register Manual*.

Table 14-4 lists the hardware settings.

Figure 14-14 shows the internal structure of the TPTC and its connection to the TPCC.

Figure 14-14. TPTC Block Diagram



The primary responsibility of the TPTC is to perform read and write transfers through the local interconnect to the slave peripherals, as programmed in the active and program register sets:

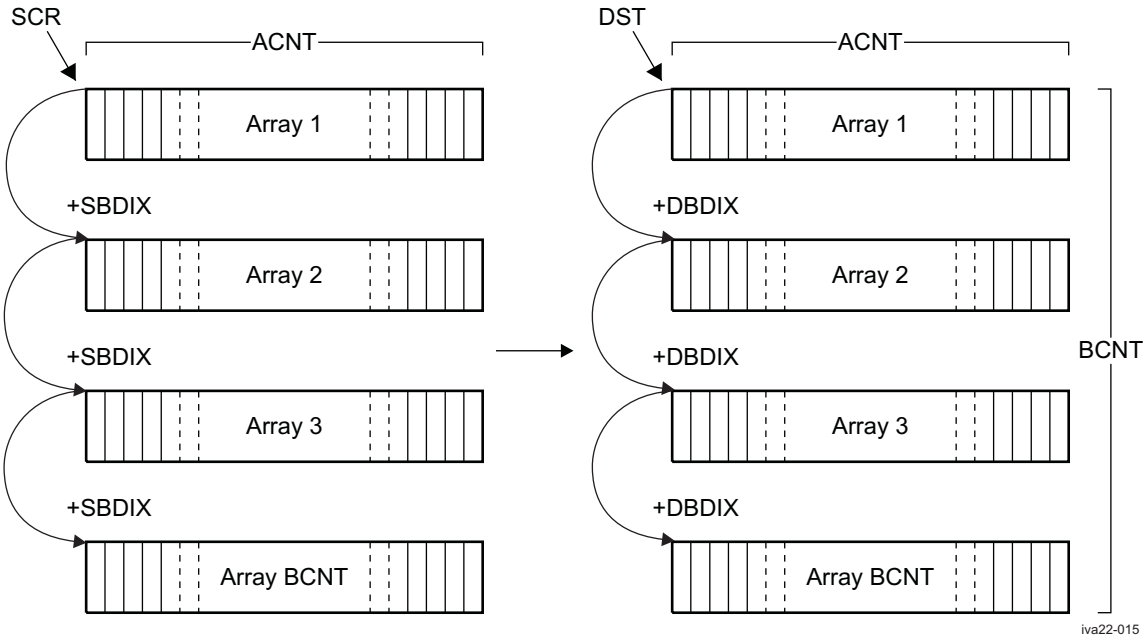
- In the TC stand-alone use model, the user directly programs the program and active register sets for a given channel (TPTC0 or TPTC1). Because of the TPCC, this mode is not the use case targeted for device applications, so it is not emphasized in this chapter. For information about its programming model, see Section 14.4.4.6.6, *Direct Configuration to Transfer Channel (Not Recommended)*.
- In the TC external-control use model, the user does not directly program the active and program register sets. Instead, the TPCC is the user interface to the EDMA system, and the TPTCs (TPTC0 and TPTC1) are slaves to the TPCC. This is the use case of the EDMA in device applications.

14.3.2.1.2.2 Transfer Geometry

The TPTC supports a transfer geometry fully defined by the registers summarized here (see Figure 14-15).

- OPTx = options, (where x = 0, 1)
  - SAM = Source address mode, controls whether the source array is from incrementing addresses or from a single FIFO address
  - DAM = Distant address mode, controls whether the destination array is to an incrementing address or to a single FIFO address
  - FWID = Controls the width of the FIFO
- SRC = Source address
- DST = Distant address
- CNT = BCNT, ACNT
  - ACNT = Number of bytes in each array
  - BCNT = Number of arrays in each TR
- BIDX = DBIDX, SBIDX
  - SBIDX = Source B-dimension index, defines the address offset between starting addresses of each source array
  - DBIDX = Distant B-dimension index, defines the address offset between starting addresses of each destination array

Figure 14-15. Transfer Geometry



Note: Many Texas Instruments DMA controllers employ a concept of element size and element indexing. To maintain orthogonality, the concept of element size has been dropped. An element-indexed transfer is logically achieved by programming ACNT to the size of the element and BCNT to the number of elements that must be transferred.

14.3.2.1.2.3 Tracking Transfers

Each TPTC channel consists of three register sets: a program register set, a source-active register set, and a distant FIFO register set. The status of each of these register sets is indicated by the PROGBUSY, SRCACTV, DSTACTV, and WSACTV status bits, which are user-readable in the TPTCj_TCSTAT register.

The program register set (TPTCj_POPT, TPTCj_PSRC, TPTCj_PCNT, TPTCj_PDST, TPTCj_PBIDX, and TPTCj_PMPPRXY) is never modified by the TPTC, because they are simply holding registers processed by the active register sets. The source active register set (TPTCj_SAOPT, TPTCj_SASRC, TPTCj_SACNT, TPTCj_SADST, TPTCj_SABIDX, TPTCj_SAMPPRXY, TPTCj_SACNTRLD, TPTCj_SASRCBREF, and TPTCj_SADSTBREF) tracks commands for the source side of the transfer, and the distant FIFO register set (TPTCj_DFOPTi, TPTCj_DFSRCi, TPTCj_DFCNTi, TPTCj_DFDSTi, TPTCj_DFBIDXi, and TPTCj_DFMPPRXYi) tracks commands for the distant side of the transfer.

As the source and distant controllers process a TR, the SRC, DST, and CNT fields must be continuously updated as commands are issued. A reference value for these registers is also maintained. The active register set uses the reference address (SRCBREF and/or DSTBREF) to calculate the addresses of the subsequent arrays in a transfer. This is required because subsequent arrays are defined as an offset from the starting address of the current array. Similarly, when a new array begins, the CNT must be restored (from CNTRLD) to the originally programmed value and decremented as the new array is processed.

The following summarizes register use for the active register sets:

- Static fields
  - OPT
  - BIDX = DBIDX/SBIDX
- Dynamic fields
  - SRC = Source address of the next read command to be issued
    - Tracked by source active register set
  - DST = Distant address of the next write command to be issued

    - Tracked by distant FIFO register set
  - CNT = BCNT/ACNT
    - BCNT = Number of arrays remaining to be transferred. This includes the current array; that is, BCNT is decremented after all commands for an array are issued (or at least after all read or write commands are scheduled).
    - ACNT = Number of bytes remaining to be transferred
    - Tracked independently in source and distant FIFO register sets
- Reference fields
  - CNTRLD = ACNTRLD only
    - CNTRLD.ACNTRLD is set with the initial CNT.ACNT value. The reload value is copied into CNT.ACNT when CNT.ACNT decrements to 0.
    - Tracked independently in source and distant FIFO register sets

    Because a command is complete when BCNT decrements to 0, there is no BCNT reload value.
  - SRCBREF = Source address reference points to the starting address of the array being read. The starting address of the next array is calculated as SRCBREF + SBIDX.
    - Tracked by source active register set
  - DSTBREF = Distant address reference points to the starting address of the array being written. The starting address of the next array is calculated as DSTBREF + DBIDX.
    - Tracked by distant FIFO register set

14.3.2.1.2.4 Completion Interface to TPCC

The TPCC can request that the TPTC send completion information to the TPCC when a TR completes. Completion status is requested in the TR options registers (TPTCj_DFOPTi[20] TCINTEN and TPTCj_DFOPTi[22] TCCHEN bits, and TPTCj_DFOPTi[17:12] TCC where i = 0 to 3 when j = 0 and i = 0 or 1 when j = 1).

If TCINTEN or TCCHEN is set to 1, the TPTC must return completion information on completion of the entire TR. The TPCC uses completion information for chaining (enabled by TCCHEN) or for posting interrupts (enabled by TCINTEN).

The TPTC generates status conditions based on completion of a transfer (TPTCj_INTSTAT[1] TRDONE bit) and based on the program register set transitioning to the downstream registers (TPTCj_INTSTAT[0] PROGEMPTY bit).

These two conditions can be enabled by the corresponding bits in the TPTCj_INTEN register to generate an interrupt to the DSP CPU through the TCERRINTx interrupt line (TCERRINT0 for TPTC0 and TCERRINT1 for TPTC1). For the event mapping of these interrupt lines, see Table 14-2.

Note: Status bits TRDONE and PROGEMPTY are always available and are stored in the TPTCj_INTSTAT register regardless of use model and regardless of whether the corresponding bits are enabled. Typically, these bits are not used in the TPCC use model. They are used in a stand-alone use model in which the user directly programs the TC.

For more information about EDMA completion and its programming model, see Section 14.4.4.1, *Transfers From/to Device Memories/Peripherals (EDMA)*.

14.3.2.1.3 EDMA Hardware Parameters

Table 14-4 lists the hardware parameter settings for the IVA2.2 subsystem.

Table 14-4. IVA2.2 EDMA Hardware Parameters

Module		Parameter	IVA2.2
EDMA	TC	FIFO size for TC0	256 bytes
		FIFO size for TC1	128 bytes
		Data bus width	64 bits

Table 14-4. IVA2.2 EDMA Hardware Parameters (continued)

Module		Parameter	IVA2.2
		TR pipelining depth TC0	4
		TR pipelining depth TC1	2
		Endianness	Little
		Burst size	64
	CC	Number of DMA channels	64
		Number of QDMA channels	8
		Number of PaRAM entries	128
		Number of Event Qs	2
		Number of TPTCs	2

Note: The TR pipelining depth controls the number of read TRs for a given transfer channel that can be serviced by the source transfer controller without being serviced by the distant transfer controller. This dictates the amount of storage required in the distant queue channel registers for inflight TRs

The user cannot directly program burst size, as with the sDMA, but must rely on the ACNT/BCNT and issue 1D synchronization transfers to do so.

14.3.2.2 EDMA Access to Video Accelerator/Sequencer

To improve performance and to offload themselves, the DSP megamodule and sequencer can both program the EDMA in IVA2.2 to transfer memory to/from the video accelerator/sequencer (sequencer, iLF, iME) to/from external memories. The DSP megamodule programs the EDMA through the IVA2.2 local interconnect, whereas the sequencer programs the EDMA through the video and sequencer local interconnect. The EDMA can access the following module registers and memories in the video accelerator/sequencer:

- iME configuration registers
- iLF configuration registers
- Sequencer memories
- SL2 memory (access through the IVA2.2 local interconnect)

The DMA accesses the video and sequencer modules with the memory mappings listed in Table 14-5.

Table 14-5. EDMA Memory Mapping for the Video Accelerator/Sequencer

Device Name	Start Address (Hex)	End Address (Hex)	Size (in Kbytes)
Reserved	0x0000 0000	0x0008 FFFF	576
SEQ CFG	0x0009 0000	0x0009 07FF	2
Reserved	0x0009 0800	0x0009 3FFF	14
SEQ DMEM	0x0009 4000	0x0009 4FFF	4
Reserved	0x0009 5000	0x0009 7FFF	12
SEQ IMEM	0x0009 8000	0x0009 9FFF	8
Reserved	0x0009 A000	0x0009 BFFF	8
SYSC - CFG	0x0009 C000	0x0009 CFFF	4
Reserved	0x0009 D000	0x0009 FFFF	12
iME CFG	0x000A 0000	0x000A 0FFF	4
iLF CFG	0x000A 1000	0x000A 1FFF	4
Reserved	0x000A 2000	0x000F 7FFF	344
Video interconnect CFG	0x000F 8000	0x000F BFFF	16

Table 14-5. EDMA Memory Mapping for the Video Accelerator/Sequencer (continued)

Device Name	Start Address (Hex)	End Address (Hex)	Size (in Kbytes)
Reserved	0x000F C000	0x000F FFFF	16
Other MEMS and PERIPHS	0x0010 0000	0xFFFF FFFF	4,193,280

14.3.2.3 IDMA

The IDMA is a simple DMA engine that performs block transfers between any two memory locations local to the DSP megamodule. A local memory has a controller that is included in the DSP megamodule. Local memory can be L1P, L1D, or L2 memories or DSP megamodule port configuration for an offloaded configuration. The IDMA controller consists of two DMA channels (channel 0 and channel 1) that can be independently programmed (a dedicated set of registers) to perform block moves between internal DSP megamodule resources.

IDMA channel 0 is used only to configure registers of IVA2.2 modules connected to the DSP megamodule configuration port, and it is useful for the DMA PaRAM entries.

To allow an easy configuration of IVA2.2 modules, the IDMA channel contains five registers: status, mask, source address, destination address, and window count.

For information about the use of channel 0 for an offloaded configuration, see Section 14.4.4.6.5, *Offloaded Configuration (Using IDMA).*

IDMA channel 1 allows transfers between any two DSP megamodule local internal memories (L2, L1P, L1D). Channel 1 is intended for background transfers in internal memory, such as block moves between the relatively high-latency L2 and the zero-latency L1D SRAM. This lets the CPU process data directly within L1D with minimal CPU impact.

For more information, see Section 14.4.4.2, *Internal Memory-to-Memory Transfer (IDMA).*

14.3.3 MMU

The IVA2.2 MMU communicates accesses from the DSP core of the IVA2.2 subsystem to the L3 interconnect, mapping the 4G bytes of the DSP virtual addresses to any place in the 4G-byte address space of the device.

At reset, the MMU is disabled, and the IVA2.2 DSP CPU can access device global memory mapping from the 0x1100 0000 address. The range of addresses 0x00000000 to 0x10FF FFFF is reachable only by the DSP CPU, because it performs its own internal memory-mapping function. For more information, see Chapter 2, *Memory Mapping*.

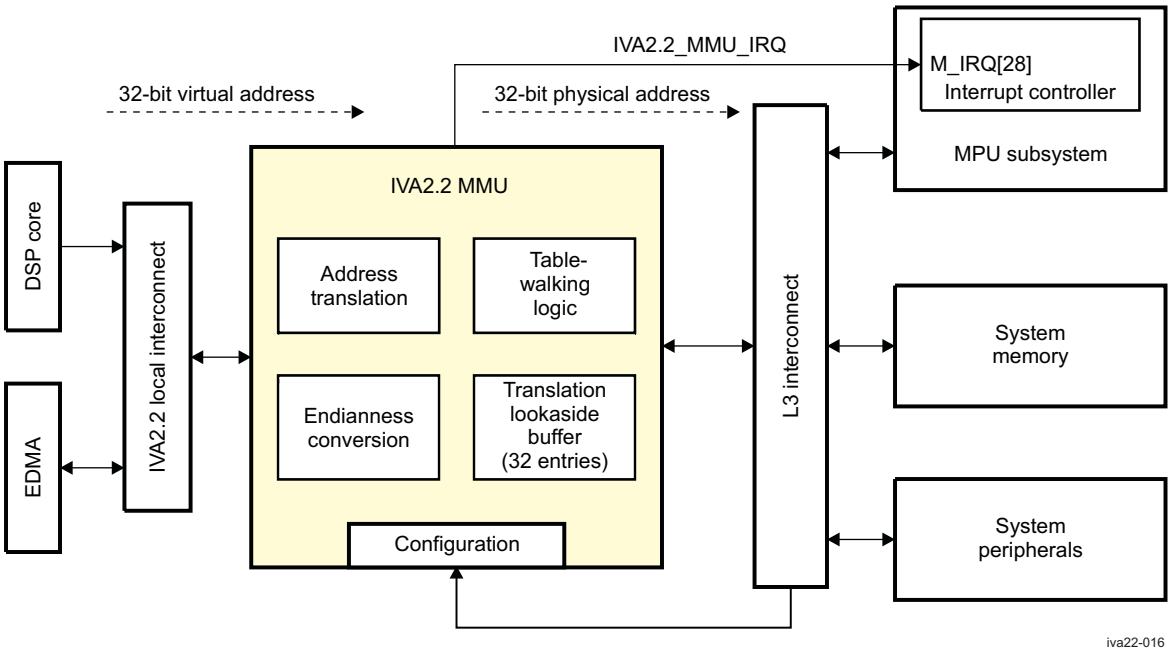
The IVA2.2 MMU main features are:

- 32 entries/fully associative translation lookaside buffer (TLB)
- One interrupt line to the MPU
- 32-bit virtual addresses, 32-bit physical addresses
- 4-KB and 64-KB pages, 1-MB section, 16-MB supersection
- Predefined (static), software-driven (interrupt-based) or table-driven (hardware table-walker) software translation strategies

Note: The endianness conversion capability of the MMU module is not used in the IVA2.2 subsystem (the DSP is configured as a little-endian processor in the device). With this configuration, the MMU2.MMU_RAM[9] ENDIANNESS bit setting is ignored, even if the user changes the value.

Figure 14-16 shows the integration and functionality of the IVA2.2 MMU module.

Figure 14-16. IVA2.2 MMU Block Diagram



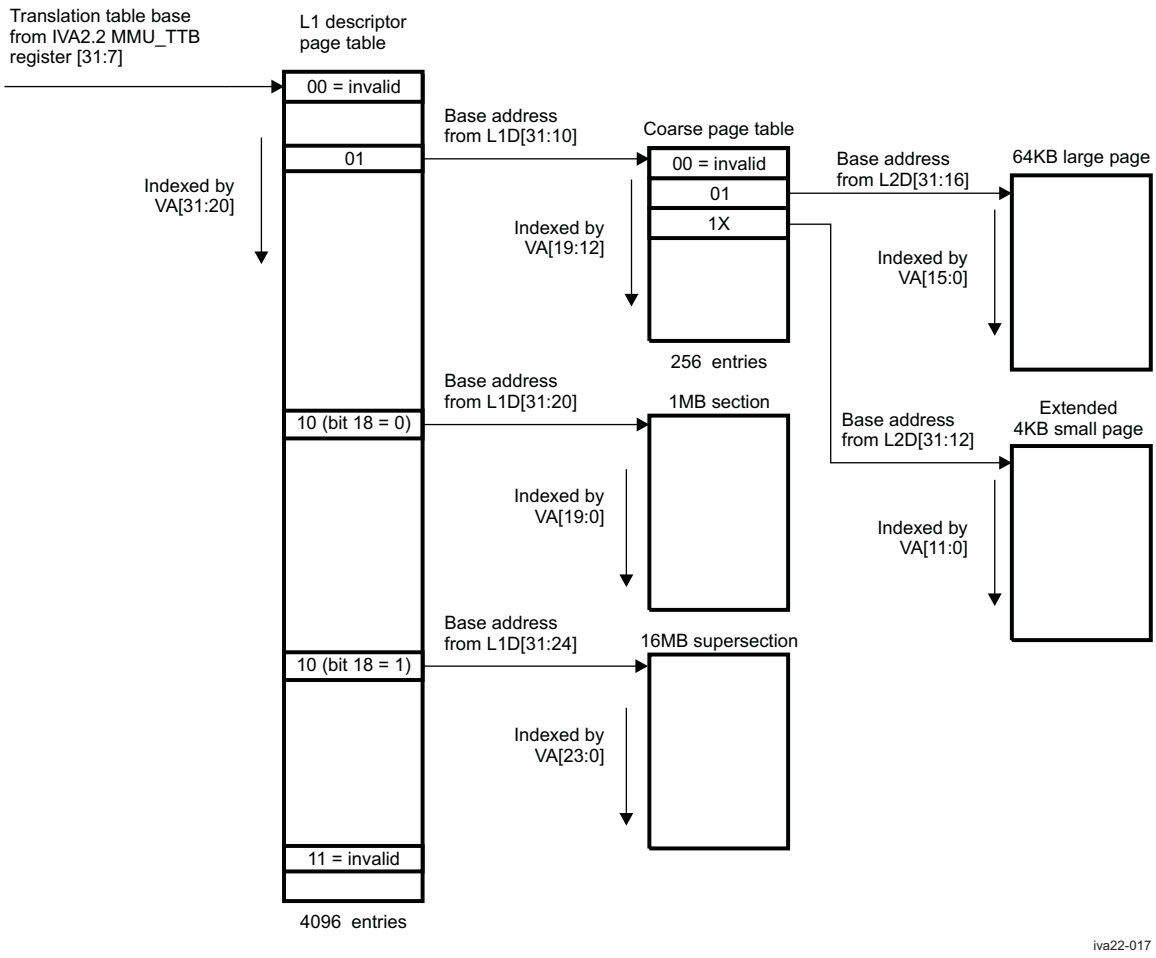
14.3.3.1 MMU VA-to-PA Translation

The IVA2.2 MMU translates the 32-bit DSP external addresses to physical addresses in the 32-bit MPU address space. Address translation is performed by a translation table structure (TTB) that maps the most-significant bits (MSBs) of the DSP byte address to another set of MSBs of a 32-bit MPU byte address. The least-significant bits (LSBs) of the DSP-generated byte address are used as page/section indices in the address translations and are not altered when forming the new 32-bit physical address. The TTB translations are expedited by a TLB that serves as a cache of recently used page translations. The address mapping can be programmed at the TTB level or by writing the TLB entries directly. The IVA2.2 MMU contains 32 TLB entries that can be configured to remap 4-KB, 64-KB, 1-MB, or 16-MB segments of memory.

The TLB can be managed statically or dynamically (through the use of interrupts) by the MPU OS, but the MMU also includes hardware table-walking logic that lets the MMU autonomously traverse the page table on a TLB miss. On a TLB miss, the translation table-walking logic automatically retrieves the information from the translation table stored in physical memory and updates the TLB cache.

Figure 14-17 shows the TTB structure in detail.

Figure 14-17. IVA2.2 MMU Translation Table Hierarchy



Note: The MMU passes the lower bits of the virtual address unchanged.

14.3.3.2 MMU Configuration

If the IVA2.2 MMU requires software intervention, the MPU services the event; IVA2.2 MMU service requests are signaled to the MPU with a dedicated interrupt, M_IRQ[28].

Generally, the MMU is initialized at boot time, but it can also be dynamically reprogrammed. Typically, the MMU is programmed by the MPU through the IVA2.2 slave port on the L3 interconnect when a new task is created on the IVA2.2 subsystem. But the DSP also has access to the MMU configuration registers for the save and restore process. At reset, MMU is disabled and DSP virtual addresses are passed directly through as physical addresses (not translated). IVA2.2 MMU configuration registers are at system base address 0x5D00 0000.

For more information about the functionality of the IVA2.2 subsystem MMU module, including interrupts, register descriptions, and programming model, see Chapter 8, *Memory Management Units*.

14.3.4 Video Sequencer

The sequencer module, referred to as sequencer or video sequencer, is an ARM968E-S based microcontroller that handles maintenance functions for the video hardware accelerators implemented in IVA2.2.

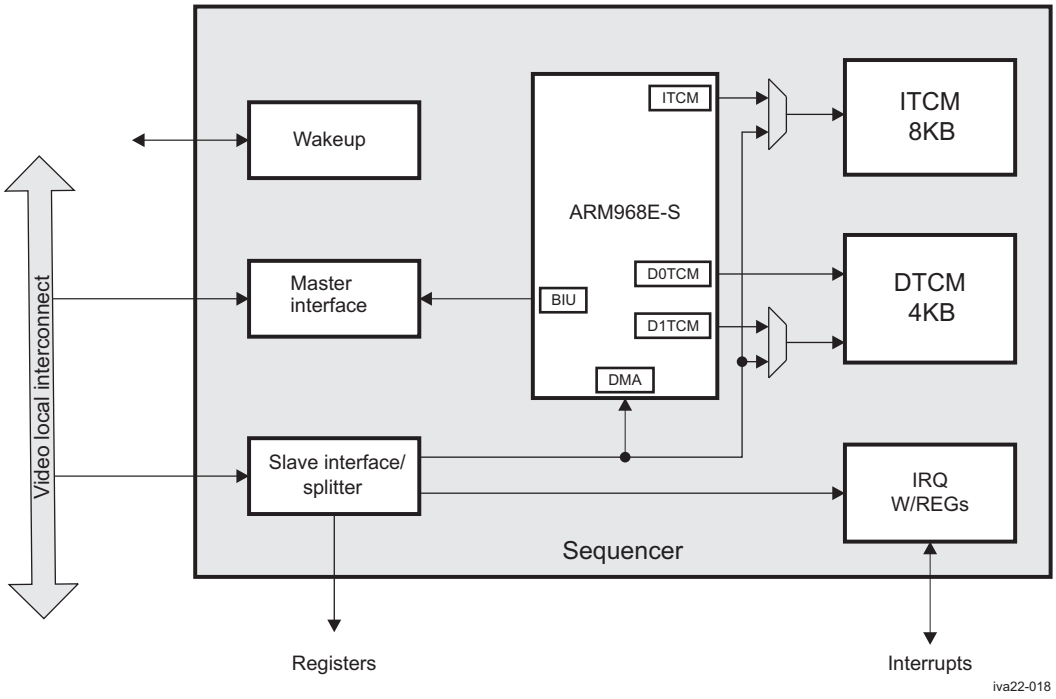
The sequencer is connected to the video accelerator and local interconnect in an accelerator design. The sequencer provides the following functions:

- Accelerator configuration
- Interrupt handling
- Local register access
- Host communication

Figure 14-18 is a block diagram of the sequencer.

The sequencer uses the little-endian convention.

Figure 14-18. Sequencer Block Diagram



The sequencer include two types of memory:

- Instruction tightly coupled memory (ITCM), an 8K-byte instruction memory, used by the ARM968E-S to fetch its instruction. This memory must be filled by DMA.
- Data tightly coupled memory (DTCM), a 4K-byte data memory, used to store data
  These two memories can be reached from the ARM968E-S and from the local interconnect.

14.3.4.1 DMA Access

The sequencer can fully control the EDMA included in the IVA2.2 subsystem. The sequencer accesses the same EDMA registers as the DSP megamodule, but the base address is 0x2 0000. The sequencer can use the EDMA to initialize the ITCM, DTCM.

14.3.4.2 Sequencer Interrupts

The sequencer module has an interrupt controller, IRQ, to capture external interrupts. External interrupts typically come from the EDMA, local interconnect, DSP megamodule, or one of the video hardware accelerators. For a complete list of externally generated interrupts, see Table 14-6. All interrupts are a one-clock-wide active-high pulse. An interrupt causes the appropriate bit in the IVA22.SEQ_IRQSTATE register to be set. Before a second interrupt on the same input can occur, the interrupt must be cleared at the source. Any interrupt can be enabled (disabled) by clearing (setting) the appropriate bit in the IVA.SEQ_IRQMASK register. If at least one enabled interrupt is captured, an FIQ interrupt is signaled to the sequencer.

The FIQ interrupt to the ARM remains active until all enabled interrupts are cleared in the IRQ when 1 is written to the appropriate bits of the IVA.SEQ_IRQSTATE register. The externally generated interrupts must be cleared at the source before being cleared in the IRQ.

The interrupt is captured, regardless of whether the sequencer is in IDLE state (standby). If an unmasked interrupt occurs while the sequencer is idle, the sequencer initiates its wake-up process.

Table 14-6. Sequencer External Interrupt

IRQ#	INT NAME	INT SOURCE	INT DESCRIPTION
0	iME interrupt	iME	iME interrupt notification
1	iLF interrupt	iLF	iLF interrupt notification
2	Reserved	N/A	N/A
3	Reserved	N/A	N/A
4	EDMA access error	Local interconnect	Video accelerator/sequencer local interconnect notification of a DMA access error
5	Sequencer access error	Local interconnect	Video accelerator/sequencer local interconnect notification of a sequencer access error
6	Reserved	N/A	N/A
7	Reserved	N/A	N/A
8	HOST interrupt	DSP megamodule	Software interrupt for DSP megamodule communication
9	DMA interrupt 0	DMA	Completion interrupt from DMA 0
10	DMA interrupt 1	DMA	Completion interrupt from DMA 1
11	DMA interrupt 2	DMA	Completion interrupt from DMA 2
12	DMA interrupt 3	DMA	Completion interrupt from DMA 3
13	DMA interrupt 4	DMA	Completion interrupt from DMA 4
14	DMA interrupt 5	DMA	Completion interrupt from DMA 5
15	DMA interrupt 6	DMA	Completion interrupt from DMA 6
16	DMA interrupt 7	DMA	Completion interrupt from DMA 7
17	DMA interrupt 8	DMA	Completion interrupt from DMA 8
18	DMA interrupt 9	DMA	Completion interrupt from DMA 9
19	DMA interrupt 10	DMA	Completion interrupt from DMA 10
20	DMA interrupt 11	DMA	Completion interrupt from DMA 11
21	DMA interrupt 12	DMA	Completion interrupt from DMA 12
22	DMA interrupt 13	DMA	Completion interrupt from DMA 13
23-31	Reserved	N/A	N/A

14.3.4.3 Sequencer to DSP Megamodule Interrupts

To support a mailbox function between the sequencer and the DSP megamodule CPU, the SEQ_MBX signal interrupts the DSP megamodule when mail is available. The sequencer can set this line high through a write of 1 to the IVA22.SEQ_SWISET register. The host can clear this interrupt through a write of 1 to the IVA22.SEQ_SWICLR register.

The sequencer DTCM memory can be used as shared memory between DSP and sequencer.

14.3.4.4 I/O Interrupts

Table 14-7 lists interrupt signals at the sequencer boundary. Table 14-99 summarizes the interrupt register address mappings.

Table 14-7. Sequencer Out Interrupt Interface

Interrupt Name	Direction	Description	Synchronous to
SEQ_MBX	Out	Software interrupt (from mailbox register)	SEQ_CLK

14.3.4.5 IRQ Generic

The sequencer is connected to the FIQ interrupt. The IRQ interrupt is not used and is tied off to inactive in the sequencer.

Two registers are defined for operations on IRQs:

- The IVA.SEQ_IRQSTATE register keeps track of input events. Each event has its own bit in IVA.SEQ_IRQSTATE. When the bit is set after an active pulse on the associated event line, this bit is kept active (sticky) until the user explicitly writes 1 to it.
- The IVA.SEQ_IRQMASK register controls the sensitivity of the interrupt line to input events. Each event has its own bit in the IVA.SEQ_IRQMASK register. When the user writes 1 to a bit, the associated active event is not allowed to trigger an interrupt. When the user writes 0, the associated active event triggers an interrupt.

For more information about these registers, see Section 14.5*, IVA2.2 Subsystem Register Manual*.

14.3.4.6 Sequencer Core

The ARM968E-S processor is used in the core. For more information, see the *ARM968E-S ARM Technical Reference Manual*.

14.3.4.7 Exception Handling

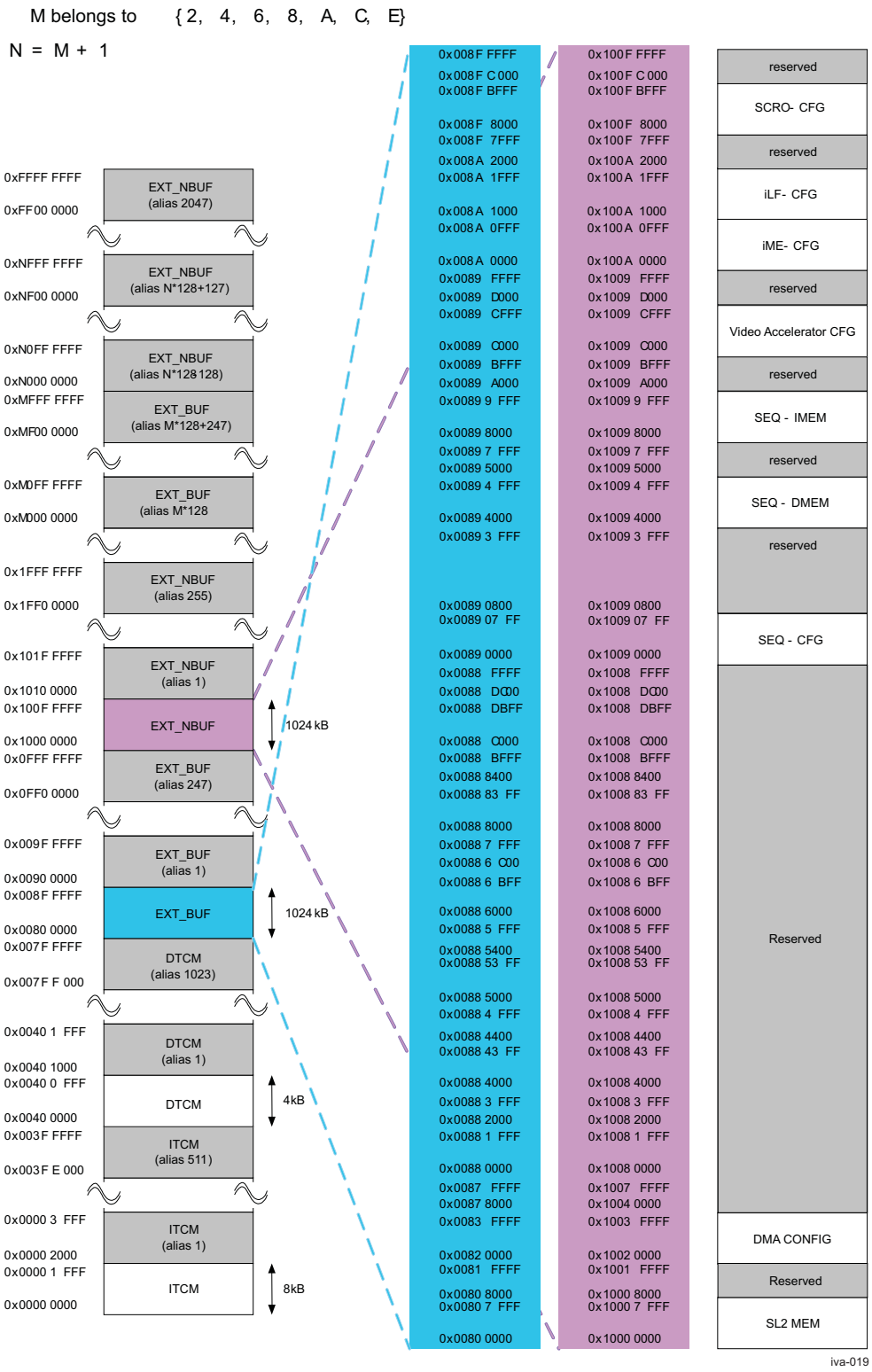
The sequencer slave port responds with an in-band error (SResp = ERR) in the following condition:

- Nonmapped address (reserved space)

14.3.4.8 Sequencer Memory Mapping

Figure 14-19 shows the sequencer memory mapping.

Figure 14-19. Sequencer Memory Mapping



14.3.5 iLF and iME Common Part

iLF and iME are video accelerator modules that share the following properties:

- Commands
- State-machine
- Instruction set (except data-operator-oriented instruction)
- Debug mode

14.3.5.1 iLF/iME Control

The iLF/iME coprocessors are controlled in two ways:

- Direct-write mode: For simple code with few instructions. The processor (DSP megamodule or video sequencer) initializes the iLF/iME coprocessor, and then writes a small instruction sequence. For instructions to be performed, all iLF/iME registers must be initialized individually, and the processor (sequencer or DSP megamodule) must tightly control the iLF/iME coprocessor.
- Normal mode: For application code with many instructions. The processor configures a small set of iLF/iME registers, and then creates a memory image of the iLF/iME parameters and an instruction sequence in the last 32K bytes of the L2 flat memory. A single instruction is written in the iLF/iME command register to specify where the iLF/iME coprocessor must fetch its instruction and retrieve its parameters. Normal mode allows parallel processing and can be highly optimized by using IDMA and different cache levels.

Both modes require an instruction set and commands to control the execution of the instructions. Section 14.3.5.1.1, *Commands*, and Section 14.3.5.2, *Instruction Set*, describe these tools.

14.3.5.1.1 Commands

Commands for controlling the iLF/iME EXECUTION states must be written in the IVA.iLF_COMMANDREG/IVA.iME_COMMANDREG register at address offset 0xFFC from the base address of iME/iLF. Table 14-8 describes the iLF/iME execution commands.

Table 14-8. iLF/iME Common Commands

Command	Description	Code
RRead()	Read register value.	-
RWrite()	Write register value.	-
StartSeq()	Forces the start of execution at the first instruction of the current sequence (PC reset with a pipeline flush engaged, regardless of any memory-access stall). CPU status goes to EXECUTING state (all other flags are cleared), cycle counter is reset and starts counting IRQ. Log content is reset.	0x1
StopSeq()	Forces the stop of execution. CPU state goes to INITIALIZED. No status information is touched. DSP engine pipeline flush is triggered (regardless of any memory-access stall). The DetectedStopSeq flag is set when this command is received and cleared when INITIALIZED state is reached.	0x2

The commands listed in Table 14-8 are mapped through the video and sequencer interconnect memory map, at offset 0xFFC from the base address in the iLF/iME address space associated with a write operation. Register read and write commands are direct reads and writes to the corresponding register address.

14.3.5.1.2 Execution States

Table 14-9 lists the execution states of the iLF/iME modules. The IVA.iLF_CPUSTATUSREG[24:25] EXECSTATE/IVA.iME_CPUSTATUSREG[24:25] EXECSTATE field shows the current execution state of the iLF/iME coprocessor.

Table 14-9. iLF/iME Execution States

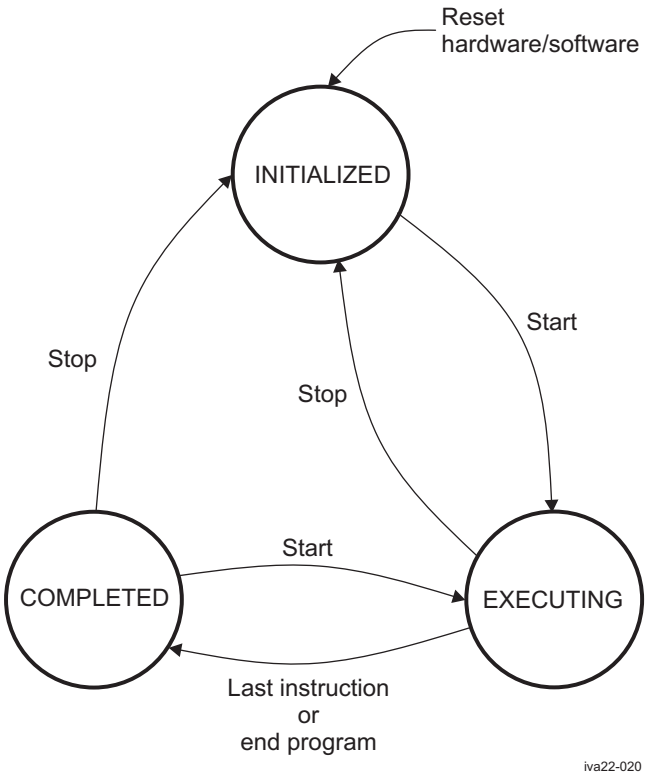
State	Code in EXECSTATE Field	Description
INITIALIZED	0x00	State after reset and after a stop command
EXECUTING	0x10	iLF/iME is computing data; state after a start command
HALTED	0x01	In debug mode only, state reached after EXECUTING state

Table 14-9. iLF/iME Execution States (continued)

State	Code in EXECSTATE Field	Description
COMPLETED	0x11	State reached when EndPgm instruction is executed or after the last instruction

The simple state-machine in Figure 14-20 shows how the EXECUTING state works in the iLF/iME coprocessor.

Figure 14-20. iLF/iME Execution State-Machine



When the coprocessor is in EXECUTING state, modifications must not be made in the iLF/iME register. The register can be read, but not written. If a write is attempted during execution, an error is logged in the IVA.iLF_CPUSTATUSREG[26]/IVA.iME_CPUSTATUSREG[26] WRITEREGERROR bits and the write is not performed.

Note: A write can be issued on the IVA.iLF_COMMANDREG/IVA.iME_COMMANDREG register at address 0xFFC, even if the coprocessor is in EXECUTING state.

14.3.5.2 Instruction Set

The iLF/iME coprocessors provide a reduced instruction set to load, prepare, and compute data. iLF/iME instructions are multicycle instructions. Table 14-10 lists the instructions available in the iLF and iME coprocessors.

Table 14-10. iLF/iME Common Instruction Set List

Instruction	Parameters	Parameter Type	OPcode (hexa)
LoadPStack() iLF only	Stack index where SL2 offset is stored Parameter to specify whether SL2 offset is stored on stack or on instruction Stack offset in SL2 (in L2. Loads 128 bytes starting at 128 bits aligned location).	@field0 P0 P1P5	0x0a

Table 14-10. iLF/iME Common Instruction Set List (continued)

Instruction	Parameters	Parameter Type	OPcode (hexa)
LoadPStack() iME only	Stack offset in SL2 (in L2. Loads 128 bytes starting at 128 bits aligned location).	P1P5	0x0a
LoadInstBuf()	Program start offset in SL2 (16-bit, aligned on a 16-byte boundary). Number of instructions (instructions must be aligned in 64-bit bins, padded with zeroes towards the MSBs)	P1P5 P0	0x10
GenerateIT()	None (triggers interrupt from inside the program, if the interrupt is enabled)		0x1e
Endpgm()	None (indicates program execution complete)		0x1f

Each instruction is stored in a set of two 32-bit registers. There are 128 sets of two instruction registers in the iLF and 256 sets of two instruction registers in the iME. The iLF can buffer 128 instructions at a time, and the iME can buffer 256 instructions at a time.

There are two ways to write instructions in the iLF/iME registers:

- Write the instruction registers individually (for more information about register addresses, see Section 14.5, *IVA2.2 Subsystem Register Manual*). This method corresponds to the direct-write/tightly coupled control mode.
- Write the instructions on the last 32K bytes of the L2 flat RAM. Then write the load instruction buffer with the correct parameters. Finally, start the iLF/iME coprocessor using the appropriate command.

It is possible to write code with more than 128/256 instructions if the final instruction is a LoadInstBuf instruction with the parameters set toward another 128/256 instruction sequence.

Note: The final instruction of a sequence must always be the Endpgm() instruction; otherwise, an error is generated. The only exception is the LoadInstBuf(), which loads an additional instruction set in the program buffer.

Instructions written to a register or to memory must follow the format in Table 14-11 and Table 14-12. The Reserved bit must be set to 0, to pad the MSB to 32 bits.

Table 14-11. iLF/iME PROGRAMBUFFERLINENMSB Register Detail

31-23	22	21-17	16-12	11-7	6-2	1-0
Reserved	SW Breakpoint	OPcode	@field0	@field1	@field2	@field3

Table 14-12. iLF/iME PROGRAMBUFFERLINENLSB Register Detail

31-29	28-24	23-16	15-10	9-4	3	2	1-0
@field3	@field4	P0	P1	P2	P3	P4	P5

@fieldN indexes a data on the parameter stack. The number in @fieldN corresponds to the position of the corresponding data in the parameter stack.

PN parameters are passed directly in the OPcode.

The field 21:17 in the MSB word corresponds to the instruction OPcode.

Bit 22 in the MSB word indicates whether a break point is set for this instruction. If this bit is set to 1, the module reaches HALTED state at the end of this instruction.

Section 14.3.5.2.2, *LoadPStack(), iME Only*, through Section 14.3.5.2.5, *EndPgm()*, describe the instructions common to iLF/iME. For descriptions of unique iLF/iME instructions, see Section 14.3.6, *iLF Module*, and Section 14.3.7, *iME Module*.

Instructions can be written either directly in the iLF_PROGRAMBUFFERLINENLSBi or iME_PROGRAMBUFFERLINENLSBi and iLF_PROGRAMBUFFERLINENMSBi or iME_PROGRAMBUFFERLINENMSBi registers or in the last 32K bytes of L2 memory and loaded into the iLF/iME register with the LoadInstBuf() instruction. Function parameters are passed directly in the OPcode of the instruction.

14.3.5.2.1 LoadPStack(), iLF Only

Function: Loads the parameter stack into the iLF local buffer. This is faster than writing directly to the iLF registers. This instruction supports two addressing modes: direct and indirect mode. In direct mode the SL2 offset is passed in the instruction. In indirect mode, the SL2 address is stored in the stack and the instruction includes an index to the stack word where the offset is stored.

Parameters:

- **SPAddr:** Index in the parameter stack pointing to the offset address from the SL2 start used to download the stack parameters from SL2 to the iLF internal register. Its value range is from 0 to 7.
- **AddressingMode:** This parameter can take two values:
  - 0 - Immediate addressing mode (address defined on P1...P5)
  - 1 - Indirect (address on stack pointed on @field0)
- **SL2 Address:** Offset in the SL2 start address where the array used to download the stack parameters from SL2 to the iLF internal registers is stored. Its value range is from 0 to 0xFFFF. Requirements: The address of the array must be 128-bit aligned. Used in direct addressing mode (AddressingMode = 1).

Table 14-13 shows the iLF LoadPStack() MSB format.

Table 14-13. iLF LoadPStack() MSB Format

63	62	61	60	59	58	57	56	55	54	53	52	51	50	49	48	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32
Unused (pad bit)									SWBP	OPcode					@field0					@field1					@field2					@field3		
-	-	-	-	-	-	-	-	-	0	0	1	0	1	0	SPAddr					-	-	-	-	-	-	-	-	-	-	-	-

Table 14-14 shows the iLF LoadPStack() LSB format.

Table 14-14. iLF LoadPStack() LSB Format

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
@field3			@field4					P0							P1							P2						P3	P4	P5		
-	-	-	-	-	-	-	-	AddressingMode								SL2 Address																

14.3.5.2.2 LoadPStack(), iME Only

Function: Loads the parameter stack into the iME local buffer. This is faster than writing directly to the iME registers. The parameter stack must be in the last 32K bytes of the L2 memory. This instruction is similar to the LoadPStack() used in iLF (see Section 14.3.5.2.1).

Parameters:

- **SL2 Address:** Offset in the SL2 start address where the array used to download the stack parameters from SL2 to the iME internal registers is stored. Its value range is from 0 to 0xFFFF. Requirements: The address of the array must be 128-bit aligned.

Table 14-15 shows the iME LoadPStack() MSB format.

Table 14-15. iME LoadPStack() MSB Format

63	62	61	60	59	58	57	56	55	54	53	52	51	50	49	48	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32
Unused (pad bit)									SWBP	OPcode					@field0					@field1					@field2					@field3		
-	-	-	-	-	-	-	-	-	0	0	1	0	1	0	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-

Table 14-16 shows the iME LoadPStack() LSB format.

Table 14-16. iME LoadPStack() LSB Format

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
@field3			@field4					P0								P1						P2						P3	P4	P5	
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	SL2 Address															

14.3.5.2.3 LoadInstBuf()

Function: Loads the instruction buffer stored in the last 32K bytes of L2 memory into the iLF/iME local instruction buffer. Up to 128 instructions can be loaded with this instruction for the iLF module, and up to 256 instructions can be loaded for the iME module. For code containing more than 128 (iLF) or 256 (iME) instructions, the final instruction must be a LoadInstBuf() instruction with the parameters set toward another 128/256 instruction sequence.

Note: The final instruction of an instruction sequence must be the EndPgm() instruction; otherwise, an error is generated by the iLF/iME module when the sequence executes.

Parameters:

SL2 source program address: Offset from SL2 start address where the array used to download the instruction sequence into iLF/iME internal program memory is stored. Value: 0 to 0xFFFF. Requirements: The address of the array must be 128 bit-aligned.

Number of instructions: Number of instructions, minus one, to download

Note: The true number of instructions downloaded is a multiple of 4, so is equal to:

nb_instr_dl = 4* Int[(nb_instr+4)/4]

where Int[x] denotes the integer part of x.

Example: For nb_instr = 0, 4 instructions are downloaded.

For nb_instr = 3, 4 instructions are downloaded.

...

For nb_instr = 127, 128 instructions are downloaded for iLF (if nb_instr = 255, 256 instructions are loaded).

Table 14-17 shows iLF/iME LoadInstBuff() MSB format.

Table 14-17. iLF/iME LoadInstBuff() MSB Format

63	62	61	60	59	58	57	56	55	54	53	52	51	50	49	48	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32
Unused (pad bit)									SWBP	OPcode						@field0						@field1						@field2				@field3			
-	-	-	-	-	-	-	-	-	0	1	0	0	0	0	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-

Table 14-18 shows iLF/iME LoadInstBuff() LSB format.

Table 14-18. iLF/iME LoadInstBuff() LSB Format

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
@field3			@field4					P0								P1						P2						P3	P4	P5	
-	-	-	-	-	-	-	-	Number of instructions																							
SL2 source program address																															

14.3.5.2.4 GenerateIT()

Function: Generates an interrupt to the sequencer and/or to the DSP megamodule. For more information about interrupts, see Section 14.3.4.2, *Sequencer Interrupts*, for sequencer interrupts and Section 14.3.11.5.3, *Interrupt Handler*, for interrupts toward the DSP megamodule.

Parameters: This function does not take any parameters.

Table 14-19 shows iLF/iME GenerateIT() MSB format.

Table 14-19. iLF/iME GenerateIT() MSB Format

63	62	61	60	59	58	57	56	55	54	53	52	51	50	49	48	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32
Unused (pad bit)									SWBP	OPcode					@field0					@field1								@field2				@field3	
-	-	-	-	-	-	-	-	-	0	1	1	1	1	0	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-

Table 14-20 shows iLF/iME GenerateIT() LSB format.

Table 14-20. iLF/iME GenerateIT() LSB Format

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
@field3			@field4					P0								P1						P2						P3	P4	P5	
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-

14.3.5.2.5 EndPgm()

Function: Indicates completion of program execution; permits the iLF/iME to go to COMPLETED state without error generation

Parameters: This function does not take any parameter.

Table 14-21 shows iLF/iME EndPgm() MSB format.

Table 14-21. iLF/iME EndPgm() MSB Format

63	62	61	60	59	58	57	56	55	54	53	52	51	50	49	48	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32
Unused (pad bit)									SWBP	OPcode					@field0					@field1								@field2				@field3	
-	-	-	-	-	-	-	-	-	0	1	1	1	1	1	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-

Table 14-22 shows iLF/iME EndPgm() LSB format.

Table 14-22. iLF/iME EndPgm() LSB Format

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
@field3			@field4					P0								P1						P2						P3	P4	P5	
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-

14.3.5.3 Parameters in L2

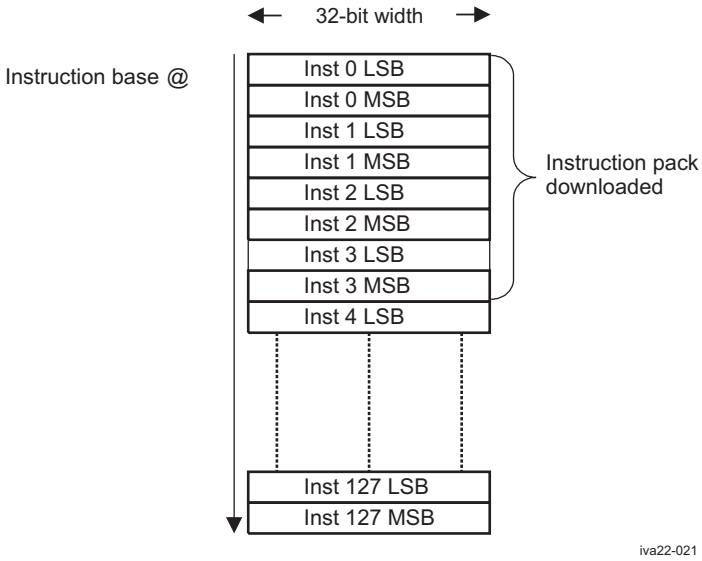
Parameters addressing a block of data in the SL2 space are given as an offset from the SL2 start address. This is true for parameters inside instructions and for parameters stored in the parameter stack.

14.3.5.3.1 Instructions

An instruction sequence can be stored in the last 32K bytes of L2 memory. This sequence is loaded in the iLF/iME with a load instruction followed by an end-program instruction. All iLF/iME instructions are 55 bits wide and padded with 0 on the MSB to reach 64 bits. Then, one 64-bit instruction is split into two 32-bit words to be stored in the DSP megamodule memory.

Figure 14-21 shows the instruction sequence format in L2 memory that must be used with the LoadInstBuf() iLF/iME instruction.

Figure 14-21. iLF/iME Instruction Sequence Format in L2 Memory



When the LoadInstBuf() command is performed from the iLF/iME side, the memory interface reads data in the L2 by 256-bit access.

Thus, the iLF/iME reads instructions by a burst of four instructions. The LoadInstBuf() instruction can load up to 128 instructions for the iLF coprocessor and up to 256 instructions for the iME coprocessor, corresponding to the sizes of the instruction buffers of the iLF and iME modules.

When instructions are loaded in the iLF/iME, they can be read or written (written only if iLF/iME is not in EXECUTING state) in the registers. These instructions are mapped on the iLF_PROGRAMBUFFERLINENLSBi or iME_PROGRAMBUFFERLINENLSBi and iLF_PROGRAMBUFFERLINENMSBi or iME_PROGRAMBUFFERLINENMSBi registers, from offset 0x040 to offset 0x43C from an iLF base address, and from offset 0x040 to offset 0x83C from an iME base address. For more information about mapping instruction registers, see Section 14.5, *IVA2.2 Subsystem Register Manual.*

14.3.6 iLF Module

The IVA2.2 subsystem includes an iLF module for performing hardware filtering for video coding and decoding. This module performs loop filtering for video code/decode operations (codecs) without DSP megamodule processing power. This hardware accelerator is considered a programmable coprocessor, as the main DSP megamodule or the sequencer can load an instruction buffer in the module, start the instruction sequence, and then the iLF module is autonomous until the end of the sequence.

The iLF performs loop filtering for four codecs:

- H264
- H263-MPEG4
- WMV9
- REAL8 and 9

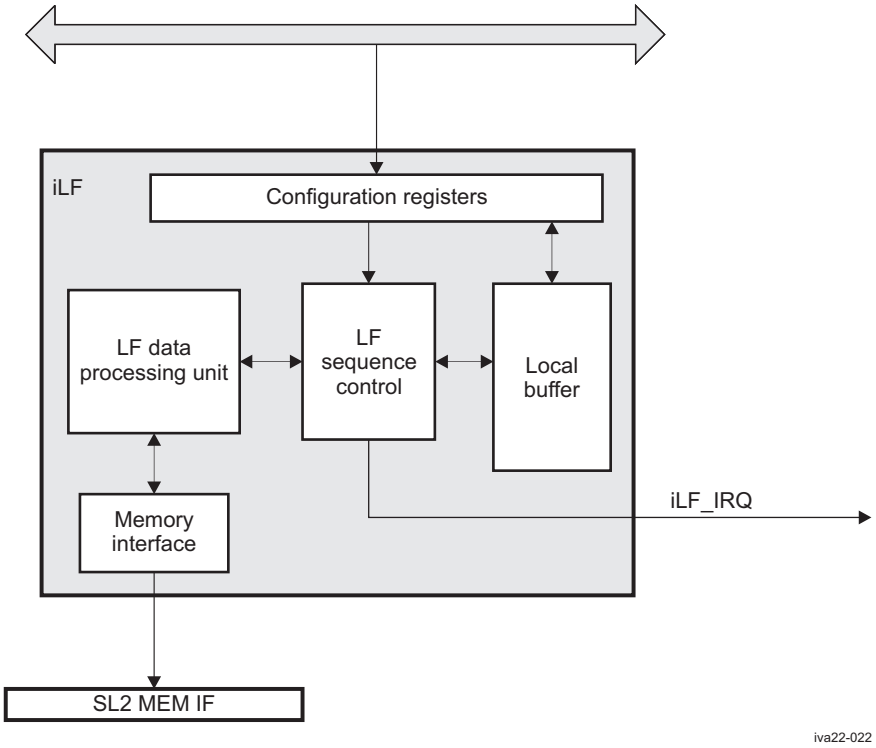
The iLF has two main interfaces:

- Slave on the video and sequencer local interconnect. This interface allows the sequencer and the DSP megamodule to access the iLF register in read/write mode. To access the iLF module registers through the local interconnect, the EFI instructions must be used on the DSP megamodule.
- Master on the SL2 memory interface. This interface allows the iLF to access data stored in the last 32K bytes of the L2 flat memory. The DSP megamodule or the sequencer can exchange configuration/data to/from the iLF through the last 32K bytes of L2 flat memory.

The iLF module can generate an interrupt to the sequencer and the DSP megamodule on completion or on error. For information about sequencer interrupts, see Section 14.3.4.2, *Sequencer Interrupts*. For information about video accelerator and sequencer interrupts toward the DSP megamodule, see Section 14.3.11.5, *Video Accelerator/Sequencer SYSC*.

Figure 14-22 is an overview of the iLF module.

Figure 14-22. iLF Overview



14.3.6.1 iLF Control

iLF control is similar to iME control, and is described in Section 14.3.5, *iLF and iME Common Part*.

14.3.6.1.1 Instruction Set

The iLF module provides a reduced instruction set to load, prepare, and compute data. Table 14-23 lists the instructions available in the iLF.

Table 14-23. iLF Instruction Set List

Instruction	Parameters	Parameter Type	OPcode (hexa)
ParseEdge()	Configuration data (see Section 14.3.6.3.2.2, *Parse-Edge Parameter Stack Data*) MBClips1 (content of parameter set to 0 if not used) MBClips2 (content of parameter set to 0 if not used) Edge index (from 0 to 15) Block number (from 0 to 15) Parsing parameters stored in the parameter stack from address 8 to 15	Stack index, 5-bit Stack index, 5-bit Stack index, 5-bit Parameter, 8-bit Parameter, 6-bit	0x01

Table 14-23. iLF Instruction Set List (continued)

Instruction	Parameters	Parameter Type	OPcode (hexa)
LoadEFPT()	Read address (in L2. Offset from the SL2 start address, aligned to 128 bits)	Stack index, 5-bit	0x0b
LoadPStack()	Stack index where SL2 offset is stored Parameter to specify whether direct or indirect load used Stack offset in SL2 (in L2. Loads 128 bytes starting at 128 bits-aligned location)	Stack index, 5-bit Parameter, 8-bit Parameter, 16-bit	0x0a
LoadInstBuf()	Program start @ (in L2. Offset from the SL2 start address, 16-bit, aligned on a 16-byte boundary) Number of instructions (instructions must be aligned in 64-bit bins, padded with zeroes toward the MSBs)	Parameter, 16-bit Parameter, 8-bit	0x10
FilterEdge()	Base address of the data block (in L2. Offset from the SL2 start address), absolute, 16-bit Opcode (0=Filter, 1 = Load&Filter, 2 = Filter&Storeback, 3 = LoadFilter&Storeback)(1) Filter parameters stored in the parameter stack from address 16 to 31	Stack index, 5-bit Parameter, 2-bit	0x04
Endpgm()	None (indicates program execution completion)		0x1f
GenerateIT()	None (triggers interrupt from inside the program, if IT enabled)		0x1e

(1) Instruction format is described in the iLF/iME command control. For more information, see Section 14.3.6.2.2, *FilterEdge()*.

14.3.6.1.2 Commands

For information about iLF commands, see Section 14.3.5.1.1, *Commands*.

14.3.6.1.3 Execution States

For information about iLF execution states, see Section 14.3.5.1.2, *Execution States*.

14.3.6.2 Instructions

The instructions described in Section 14.3.6.2.1 through Section 14.3.6.2.3 can be either directly written in the iLF_PROGRAMBUFFERLINENLSBi and iLF_PROGRAMBUFFERLINENMSBi registers or written in the last 32K bytes of L2 memory and loaded into the iLF register with the LoadInstBuf() instruction. The function parameters are passed directly in the OPcode of the instruction.

For instructions common to the iLF and iME coprocessors-LoadPStack(), LoadInstBuf(), GenerateIT(), EndPgm()-see Section 14.3.5.2, *Instruction Set*.

Each instruction is stored in a set of two 32-bit registers. There are 128 sets of two instruction registers in the iLF. The iLF can buffer 128 instructions at a time. It is possible to write code with more than 128 instructions if the final instruction is a LoadInstBuf instruction with the parameters set toward another 128-instruction sequence in SL2 memory.

14.3.6.2.1 ParseEdge()

Function: Parses the edge between pixel blocks; prepares the filtering pass

Parameters:

- **CFG:** Index in the parameter stack pointing to the configuration. Value: From 0 to 7.
- **MBClips1idx:** Index in the parameter stack pointing to the first set of MBClips (REAL9, 0 when not used). Value: From 0 to 7.
- **MBClips2idx:** Index in the parameter stack pointing to the second set of MBClips (REAL9, 0 when not used). Value: From 0 to 7.
- **Edge Index:** Index of the current edge in the parameter stack. Value: From 0 to 15.
- **Block Number:** Number of the block 4*4 containing the current edge in the macroblock. Value: From 0 to 15.

Table 14-24 shows the iLF ParseEdge() MSB format.

Table 14-24. iLF ParseEdge() MSB Format

63	62	61	60	59	58	57	56	55	54	53	52	51	50	49	48	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32
Unused (pad bit)									SWBP	OPcode					@field0					@field1					@field2					@field3	
-	-	-	-	-	-	-	-	-	0	0	0	0	0	1	-	-	-	-	-	-	-	-	-	-	CFG					MBClips 1	

Table 14-25 shows the iLF ParseEdge() LSB format.

Table 14-25. iLF ParseEdge() LSB Format

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
@field3				@field4					P0							P1						P2						P3	P4	P5	
MBClips 1				MBClips 2					Edge Index							Block Number						-	-	-	-	-	-	-	-	-	-

14.3.6.2.2 FilterEdge()

Function: Operates the filtering processing. This function operates on data stored in L2 and stores the processed data back in L2 memory.

Parameters:

- **SPAddress:** Index in the parameter stack pointing to the offset address from the SL2 start that contains the input/write buffer windows to filter. Value: From 0 to 7.
- **Opcode:** Specifies whether the SL2 area must be loaded into iLF internal memory (iLF input/write buffer) before filtering and/or whether the internal memory must be stored in SL2 after filtering. Value: 0 to 3 with the following meanings:
  - FILTER 0x0
  - LOAD_FILTER 0x1
  - FILTER_STOREBACK 0x2
  - LOAD_FILTER_STOREBACK 0x3

Table 14-26 shows the iLF FilterEdge() MSB format.

Table 14-26. iLF FilterEdge() MSB Format

63	62	61	60	59	58	57	56	55	54	53	52	51	50	49	48	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32
Unused (pad bit)									SWBP	OPcode					@field0					@field1					@field2					@field3		
-	-	-	-	-	-	-	-	-	0	0	0	1	0	0	SPAddress					-	-	-	-	-	-	-	-	-	-	-	-

Table 14-27 shows the iLF FilterEdge() LSB format.

Table 14-27. iLF FilterEdge() LSB Format

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
@field3			@field4					P0								P1						P2						P3	P4	P5	
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	OPcode	

14.3.6.2.3 LoadEFPT()

Function: Loads the edge-level filter parameters table (EFPT) in the iLF local buffer.

Parameters:

- **SPAddress:** Index in the parameter stack pointing to the address of the EFPT in the SL2 space (offset from the SL2 start address). Requirements: The address of the array must be 128-bit-aligned.

Table 14-28 shows the iLF LoadEFPT() MSB format.

Table 14-28. iLF LoadEFPT() MSB Format

63	62	61	60	59	58	57	56	55	54	53	52	51	50	49	48	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32
Unused (pad bit)									SWBP	OPcode					@field0					@field1							@field2				@field3	
-	-	-	-	-	-	-	-	-	0	0	1	0	1	1	SPAddress					-	-	-	-	-	-	-	-	-	-	-	-

Table 14-29 shows the iLF LoadEFPT() LSB format.

Table 14-29. iLF LoadEFPT() LSB Format

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
@field3			@field4					P0								P1						P2						P3	P4	P5	
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-

14.3.6.3 Parameters

14.3.6.3.1 Instructions

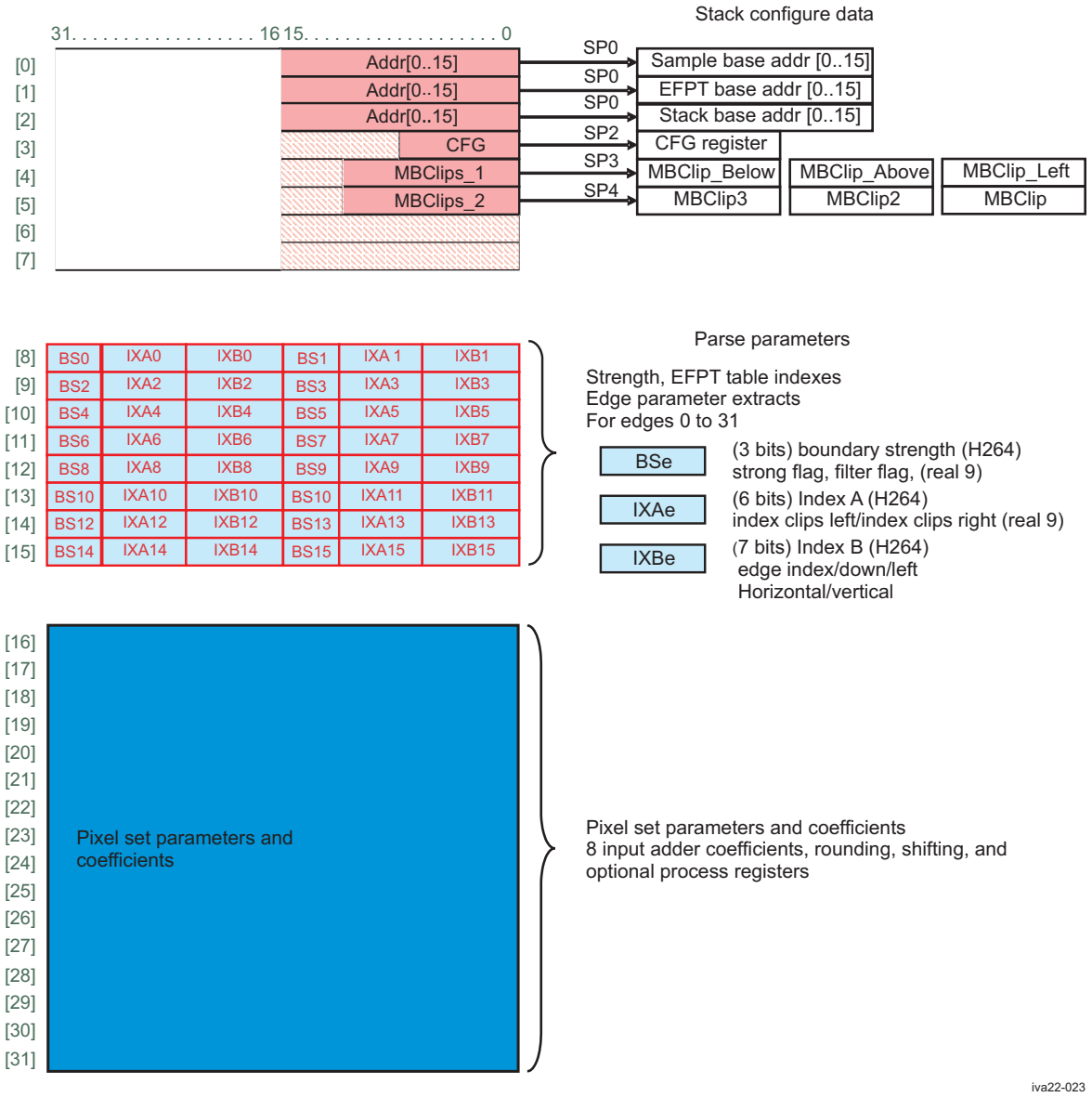
Instruction format in L2 is the same for the iLF and iME modules. For information about formatting instructions in L2, see Section 14.3.5.3.1, *Instructions*.

14.3.6.3.2 Parameter Stack

Figure 14-23 is an overview of the parameter stack stored in L2, consisting of three parts:

- Stack configure data: Includes addresses, MB configure, and the edge-configure section, which is addressed with the @fieldN parameters in the different instructions. Stack configure data are stored from word 0 to word 7. The configuration of this stack section is not fixed; each field is indexed with an instruction parameter when it must be used. These parameters are 32 bits wide in L2 memory, but when they are downloaded to the iLF internal memories, a mask is applied on the 16-MSB section of the word; therefore, only the 16 LSBs must be used.
- Parse parameters: Used by the ParseEdge() instruction. Parse parameters are stored from word 8 to word 15 and are 32-bit words. They allow configuration of the filter for the filtering parts. They are fixed parameters for a macroblock.
- Pixel set parameters and coefficients: Used by the FilterEdge() instruction. They are stored from word 16 to word 31 and are 32-bit words. These parameters allow filtering to be performed. They are fixed parameters

Figure 14-23. iLF Stack Parameter Format



14.3.6.3.2.1 Stack Configure Data

Stack configure data occupy word 0 to word 7 of the parameter stack. In this section of the stack, all words are 32 bits, but only the lower 16 bits are used. When the parameter stack is downloaded to iLF internal memory, a mask is applied on the upper 16 bits of the stack configure data words; consequently, all data stored in the stack configure data section are 16 bits wide, stored in the lower part of a 32-bit word.

Data stored in this section of the parameter stack are not on a fixed word; they are referenced through an index inside the instruction. The value of this index corresponds to the word where the data is stored.

Table 14-30 shows how data are stored in the stack configure data section of the parameter stack.

Table 14-30. iLF Stack Configure Data Format

@	Pointer to @ stack Related @fieldx points to any parameter stack address from range 0 to 7		Description	Std	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	Instruction	@fieldx																		
	FilterEdge()	@field0	Sample base address	ALL	Sample base address															
1	LoadEFPT()	@field0	EFPT base address	ALL	EFPT base address															
	LoadPStack()	@field0	Stack base address	ALL	Parameter stack base address															
2	ParseEdge()	@field2	CFG	H264						Slice A			Slice B				LU	CMR		
				WMV9								PQUANT					MA			
3				H263								PQUANT								
				REAL9								Reserved								
4	ParseEdge()	@field3	Clips1	H264					Reserved											
				WMV9 H263					Reserved											
5				REAL9					MBClip below				MBClip above				MBCLip left			
6	ParseEdge()	@field4	Clips2	H264					Reserved											
				WMV9 H263					Reserved											
7				REAL9					MBClip3				MBClip2				MBClip			

These eight words can store five types of data:

- Sample base address: Used in the FilterEdge() instruction to indicate the offset from the SL2 start of the input/write buffer windows, stored in the last 32K bytes of shared L2 memory, for the filter computation. This address must be 128-bit-aligned. It is coded on 16 bits and must reference the first byte of the first block in the input/write buffer windows. For more information about the input/write buffer, see Section 14.3.6.3.7, *Input/Write Buffer*.
- EFPT base address: Used in the LoadEFPT() instruction, this is an offset (from the SL2 start address) that indicates the start of the EFPT in the last 32K bytes of L2 shared memory. This address must be 128-bit-aligned. The EFPT table is used only for REAL9 and H264 video codecs. The value of this table is given in Section 14.3.6.3.3, *Loop-Filter Parameter Stack and EFPT for H264*, for H264 and in Section 14.3.6.3.6, *Loop-Filter Parameter Stack and EFPT for REAL9*, for REAL9.
- Stack base address: Used in the LoadPStack() instruction, this is an offset (from the SL2 start address) that indicates the start of the parameter stack in the last 32K bytes of the L2 shared memory. This address must be 128-bit-aligned.
- CFG: Configures the iLF for the filtering operation; composed of three parts:
  - CMR (condition mode) detailing the codec currently filtered. For codec codes, see Table 14-31.
  - LUMA detailing if the block filtered is a LUMA block (1 if LUMA, 0 if other)
  - Slice A and Slice B are used to compute index A and index B and take the values alpha and beta in the threshold table. Range value from -6 to 6 (H264 standard).
  - PQUANT contains quantization data, REAL specific value.
- Clips1 and Clips2: REAL9 specific; they contain the MBClip value.

Table 14-31 shows the CMR code corresponding to the codec supported by the iLF coprocessor.

Table 14-31. iLF CMR Code

Codec	Code
H264	0x0
H263	0x1
REAL9	0x2
WMV9	0x3

Note: A data type can have multiple instances in the stack configure data, such as multiple sample base addresses for loading different parts of a macroblock for filtering.

14.3.6.3.2.2 Parse-Edge Parameter Stack Data

Parse-edge parameter data are stored in the parameter stack from word 8 to word 15. All the data blocks are 16 bits wide, so there are 16 parse-edge blocks corresponding to 16 different edges. Each block of data describes the parameter required for an edge. Data stored in these blocks are codec-dependent and are described in the codec specification.

The 16 edge blocks are referenced by the EdgeIndex parameters of the ParseEdge() instruction.

The Edge_HV bit indicates whether the filtered edge is horizontal or vertical; 0 corresponds to a horizontal edge and 1 to a vertical edge.

The BS value for H.264 corresponds to Luma or Chroma first blocks. The BSB value for H.264 corresponds to Chroma second blocks.

After ParseEdge() and before FilterEdge(), the iLF_PARSEDDATAREG1 register is set with the value extracted from Parse-Edge Parameter Stack Data.

Table 14-32 shows the parse-edge parameter data format and the correspondence between the index and parse-edge block data.

Table 14-32. iLF Parse-Edge Parameter Stack Data

@	Edge Index	Std	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
			BSe Even Edges			IXAe Even Edges						IXBe Even Edges							BSe Odd Edges			IXAe Odd Edges						IXBe Odd Edges							
8 9 10 11 12 13 14 15	0 2 4 6 8 10 12 14	H264	BS			BSB			Reserved			Edge_HV	QP																					
		WMV9 H263	Reserved										Reserved																					
		REAL9	reserved	FLT(Filter)	STR(Strong)	ICR index clip right			ICL index clip left				Edge_DNLF	reserved	IXD dither index																			
8 9 10 11 12 13 14 15	1 3 5 7 9 11 13 15	H264																	BS			BSB			Reserved		Edge_HV	QP						
		WMV9 H263																	Reserved									Reserved							
		REAL9																	Reserved	FLT(Filter)	STR(Strong)	ICR index clip right			ICL index clip left			Edge_DNLF	reserved	IXD dither index				

14.3.6.3.3 Loop-Filter Parameter Stack and EFPT for H264

Table 14-33 lists the values of the loop-filter parameters stored in the parameter stack for the H264 video codec.

Table 14-33. iLF Loop Filter Parameter Stack for H264

Word	Value	Word	Value
16	0x00 C8 40 00	24	0x06 84 90 22
17	0x00 13 10 01	25	0x26 8D 1F DF

Table 14-33. iLF Loop Filter Parameter Stack for H264 (continued)

Word	Value	Word	Value
18	0x00 AA 88 02	26	0x46 95 1F BC
19	0x00 12 C8 04	27	0x00 00 0F A0
20	0x00 12 62 00	28	0x37 34 04 04
21	0x00 86 80 00	29	0xC1 20 22 20
22	0x90 28 00 00	30	0x20 20 00 00
23	0x80 38 00 11	31	0x00 86 00 00

Table 14-34 lists the values of the EFPT table stored in L2 memory for the H264 video codec.

Table 14-34. iLF EFPT Table for H264

Word	Value	Word	Value
0	0x04 02 00 00	18	0x14 14 08 44
1	0x04 02 00 01	19	0x14 16 88 64
2	0x04 02 80 01	20	0x16 19 08 64
3	0x06 03 00 01	21	0x16 1C 0C 65
4	0x06 03 80 01	22	0x18 1F 8C 86
5	0x06 04 00 21	23	0x18 23 8C 86
6	0x06 04 80 21	24	0x1A 28 10 A7
7	0x08 05 04 21	25	0x1A 2D 10 A8
8	0x08 06 04 21	26	0x1C 32 90 C9
9	0x08 06 84 21	27	0x1C 38 94 EA
10	0x0C 07 84 21	28	0x1E 3F 99 0B
11	0x0C 08 84 22	29	0x1E 48 19 0D
12	0x0E 0A 04 22	30	0x20 51 1D 4E
13	0x0E 0B 04 22	31	0x20 5B 21 70
14	0x10 0C 84 22	32	0x22 65 A5 92
15	0x10 0E 04 43	33	0x22 71 29 B4
16	0x12 10 04 43	34	0x24 7F AD F7
17	0x12 12 08 43	35	0x24 7F B6 39

14.3.6.3.4 Loop-Filter Parameter Stack for H263

Table 14-35 lists the values of the loop-filter parameters stored in the parameter stack for the H263 video codec.

Table 14-35. iLF Loop Filter Parameter Stack for H263

Word	Value	Word	Value
16	0x01 A8 40 00	24	0x34 C9 31 95
17	0x00 06 A1 04	25	0x24 CD 3F 6E
18	0x68 C3 13 34	26	0x24 CD 3E 7B
19	0x68 C6 A1 00	27	0x34 C9 30 00
20	0x00 00 00 00	28	0x03 04 04 A4
21	0x00 00 00 00	29	0x48 48 48 48
22	0x80 40 20 00	30	0x58 58 58 58
23	0x10 84 20 A2	31	0x00 46 60 01

There is no EFPT table for the MPG4 codec.

14.3.6.3.5 Loop-Filter Parameter Stack for WMV9

Table 14-36 lists the values of the loop-filter parameters stored in the parameter stack for the WMV9 video codec.

Table 14-36. iLF Loop-Filter Parameter Stack for WMV9

Word	Value	Word	Value
16	0x01 A8 40 00	24	0x32 C8 B1 95
17	0x00 06 A1 04	25	0x22 CC BF 6E
18	0x68 C3 13 34	26	0x22 CC BE 7B
19	0x68 C6 A1 00	27	0x32 C8 B0 00
20	0x00 00 00 00	28	0x03 04 04 A4
21	0x00 00 00 00	29	0x48 48 48 48
22	0x80 40 A0 00	30	0x58 58 58 58
23	0x12 84 A0 A2	31	0x00 46 60 01

There is no EFPT table for the WMV9 codec.

14.3.6.3.6 Loop-Filter Parameter Stack and EFPT for REAL9

Table 14-37 lists the values of the loop-filter parameters stored in the parameter stack for the REAL9 video codec.

Table 14-37. iLF Loop-Filter Parameter Stack for REAL9

Word	Value	Word	Value
16	0x0C 17 18 00	24	0x72 42 D1 A4
17	0x01 AA 40 01	25	0x14 8A 4B 3F
18	0x35 A7 18 04	26	0x3C D3 39 A2
19	0x00 A8 88 44	27	0x56 C0 00 0C
20	0x00 15 11 84	28	0x0D 2C 05 34
21	0x00 02 9A 81	29	0x41 F8 49 F8
22	0xB0 30 00 00	30	0x20 20 20 00
23	0xD0 D8 01 91	31	0x00 46 7C A4

Table 14-38 lists the values of the EFPT table stored in L2 memory for the H264 video codec.

Table 14-38. iLF EFPT Table for REAL9

Word	Value
0	0x00 76 48 C9
1	0x00 00 00 00
2	0x00 00 00 00
3	0x00 00 00 00
4	0x00 59 CC AC
5	0x00 8D D7 2B
6	0x00 76 47 65
7	0x00 8F 29 56
8...35	0x00 00 00 00

14.3.6.3.7 Input/Write Buffer

The iLF module contains an input/write buffer that retrieves pixel data stored in the last 32K bytes of the L2 flat memory by the memory interface, and stores them in an iLF local buffer during the pixel filtering pass. There is no dedicated command to load and/or write to the iLF input/write buffer. The action is performed with an option passed in the FilterEdge() instruction. The second parameters in the OPcode instruction contain this option. Table 14-39 shows the different value for the second parameters.

Table 14-39. iLF FilterEdge() Load and Store Parameter

Name	OPcode	Description
FILTER	0x0	Filter data already on input/write buffer.
LOAD_FILTER	0x1	Load input/write buffer from L2 memory, then filter the data in the buffer.
FILTER_STOREBACK	0x2	Filter data already in the input/write buffer, then write the iLF input/write buffer to L2 memory.
LOAD_FILTER_STOREBACK	0x3	Load the input/write buffer from L2 memory, filter the data in the buffer, then write back the iLF input/write buffer to L2 memory.

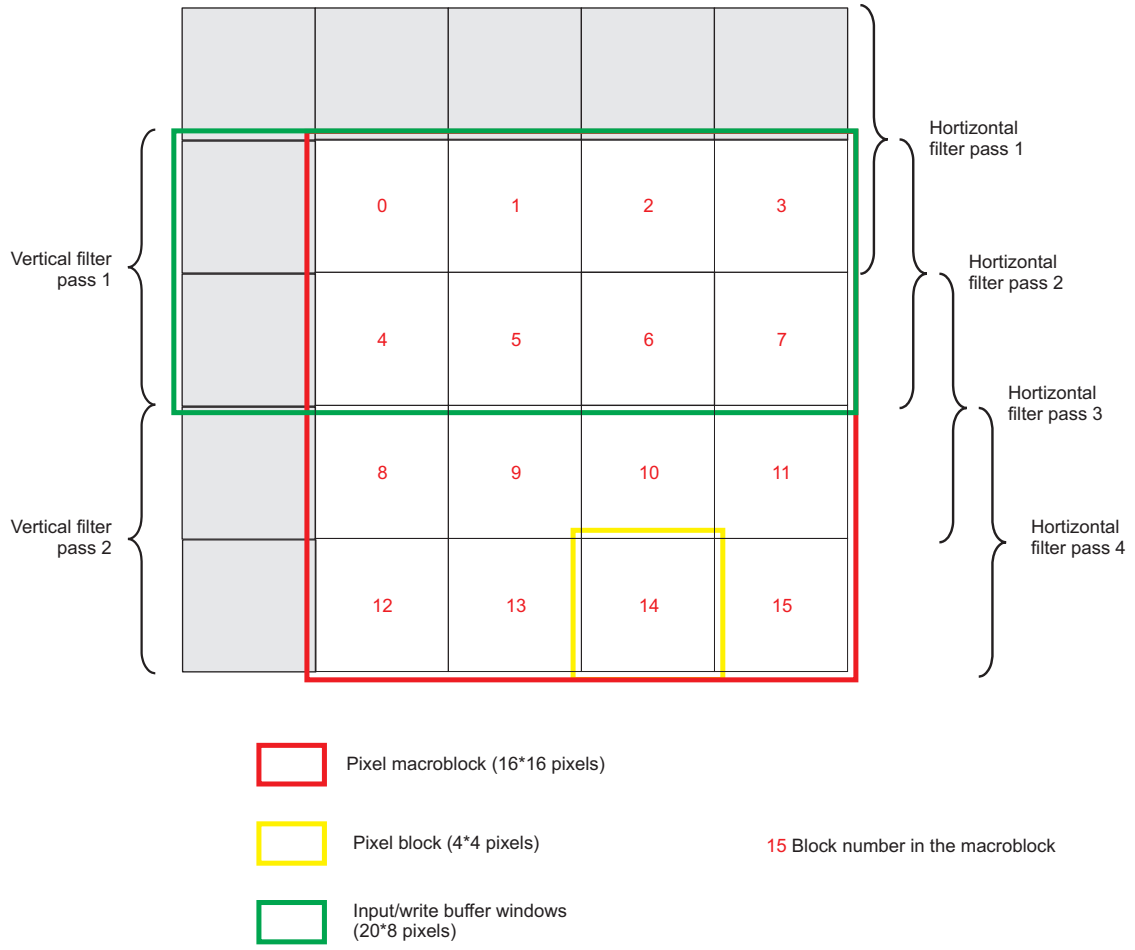
The memory interface of the iLF coprocessor retrieves and/or stores data from/to the last 32K bytes of the L2 memory, depending on the OPcode passed to the FilterEdge() instruction. The input/output (I/O) buffer can store 2 rows of 5 pixel matrices (4 x 4). Ultimately, the iLF can filter 8 vertical edges or 4 horizontal edges with a single load.

The iLF input/write buffer processes a macroblock with 2 windows for a vertical edge and 4 windows for a horizontal edge.

The FilterEdge() instruction contains a stack pointer address (SPAddress) that is a link to a 16-bit word contained in the parameter stack (for more information about the parameter stack, see Section 14.3.6.3.2, *Parameter Stack*). This address is an offset from the SL2 start pointing toward the start of the input/write buffer windows. It must be 128-bit-aligned. It is used when a load or a store is ordered by the FilterEdge() instruction.

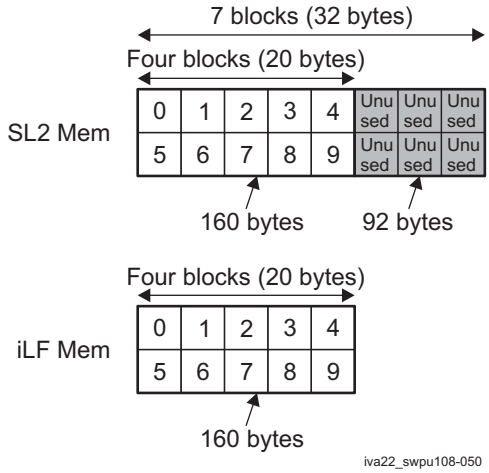
Figure 14-24 shows the pixel data format and order in memory, and the input/write buffer window. There are two numbers for blocks; the upper number corresponds to the location of the block in L2 memory, the lower number corresponds to the index of the block in a macroblock.

Figure 14-24. iLF Pixel Format in L2 Memory



The iLF has an internal 160-bit-wide buffer and can only perform 256-bit accesses to SL2. Each time the FilterEdge() instruction is executed to load memory from SL2 into iLF internal memories, iLF loads 8 lines of 32 bytes (only 20 bytes are useful). Figure xx shows how data is organized in SL2 and iLF internal memory.

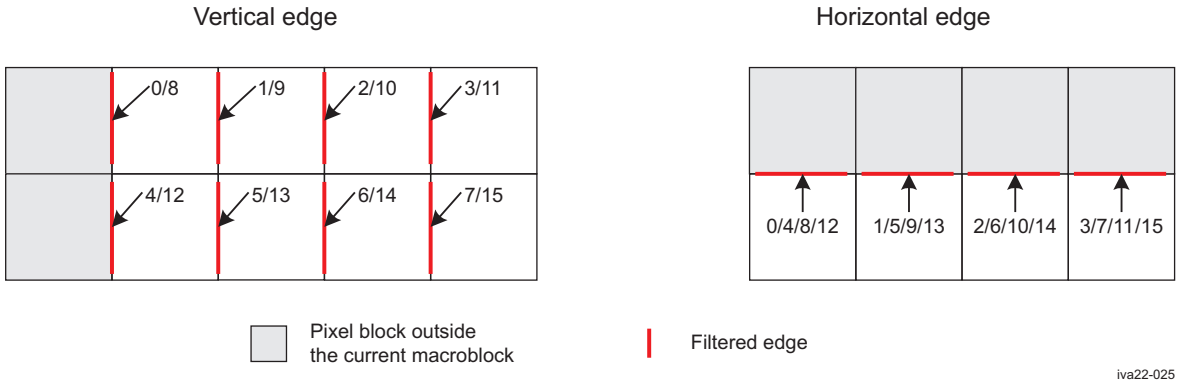
Figure 14-25. iLF Internal Pixel Format



14.3.6.3.8 Parse-Edge Parameters

The block-number parameter in the ParseEdge() instruction indicates which edge must be parsed to be filtered. This number depends on the current windows in the input/write buffer and on the edge type (horizontal or vertical). Figure 14-26 shows the position of the 15 edges in the input/write buffer. For more information about the ParseEdge() parameters, see Section 14.3.6.3.2.2, *Parse-Edge Parameter Stack Data*.

Figure 14-26. iLF Edge Numbering



14.3.7 iME Module

The IVA2.2 subsystem contains an iME module that estimates hardware motion for video coding, according to multiple video codecs, without using DSP megamodule processing power. This hardware accelerator is considered a programmable coprocessor, because the main DSP megamodule or the sequencer can load an instruction buffer in the module, start the instruction sequence, and then the iME module is autonomous until the end of the sequence.

The iME is not codec-dependent; it is not designed for a given motion estimation algorithm (like step-search).

The iME interpolates motion estimates for three codecs:

- H264
- H263-MPEG4
- WMV9

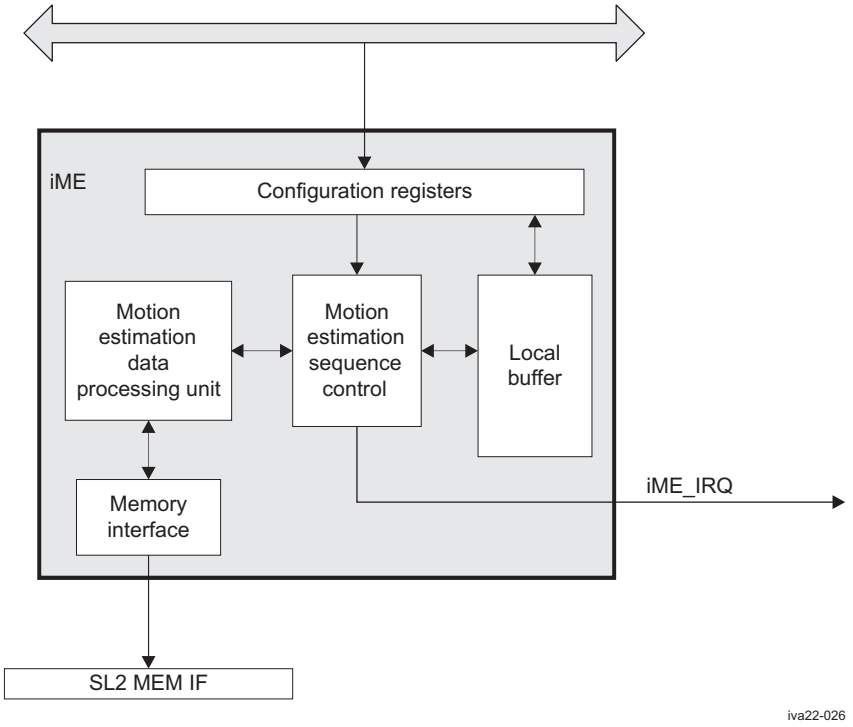
The iME has two interfaces:

- Slave on the local interconnect. This interface gives the sequencer and the DSP megamodule access to the iME register in read/write mode. To access the iME module registers, EFI instructions must be used on the DSP megamodule.
- Master on the SL2 memory interface. This interface gives the iME access to data stored in the last 32K bytes of L2 flat memory. The DSP megamodule or the sequencer can exchange configuration/data to/from the iME through the last 32K bytes of L2 flat memory.

The iME module can generate an interrupt to the sequencer and the DSP megamodule, on completion or on error. For more information about sequencer interrupts, see Section 14.3.4.2, *Sequencer Interrupts*. For more information about video accelerator and sequencer submodule interrupts toward the DSP megamodule, see Section 14.3.11.5.3, *Interrupt Handler*.

Figure 14-27 is an overview of the iME module.

Figure 14-27. iME Overview



14.3.7.1 iME Control

The iME control is similar to the iLF control, and is detailed in Section 14.3.5, *iLF and iME Common Part*.

14.3.7.1.1 Instruction Set

The iME provides a reduced instruction set to load, prepare, and compute data.Table 14-40 lists the instructions available in the iME.

Table 14-40. iME Instruction Set

Instruction	Parameters	Parameter Type	OPcode (hexa)
ErrorCalc()	Search window base @ Search start @ OffsetH (horizontal, local offset) OffsetV (vertical, local offset) Motion vector parameters Search window line length (in bytes) Offset code Error number Use best-match information (Y, N).	@field0 @field1 @field2 @field3 @field4 P0 P1 P2 P4	0x00000
SaveErrs()	Save address, 16-bit (offset from SL2 start address. Saves all errors in the error-storage table in 2 contiguous 256-bit-aligned words).	P1-P5	0x02
SaveStatus()	Save address, 16-bit (offset from SL2 start address. Saves the best-match error in one 128-bit-aligned word).	P1-P5	0x03
RestorErrs()	Restore address (offset from SL2 start address. Restores all errors in the error-storage table, from 2 contiguous 256-bit-aligned words).	P1-P5	0x04
Mcompare () *(supports only comparison of 5 to 16 errors)*	Motion vector parameters (only MVtype is used) Offset Error number start, in error-storage table Number of errors to compare (= 5 errors) Include last best match (Y, N). Compare every N errors (N = 1, 2, 4, 8).	@field4 P0 P1 P2 P4 P5	0x05

Table 14-40. iME Instruction Set (continued)

Instruction	Parameters	Parameter Type	OPcode (hexa)
Mcompare2 () (supports comparison of only 2 errors)	Motion vector parameters (only MVtype is used) Offset Error number start, in error-storage table Include last best match (Y, N). Compare every N errors (N = 1, 2, 4, 8).	@field4 P0 P1 P4 P5	0x06
Mcompare4 () (supports comparison of only 4 errors)	Motion vector parameters (only MVtype is used) Offset Error number start, in error-storage table Include last best match (Y, N). Compare every N errors (N = 1, 2, 4, 8).	@field4 P0 P1 P4 P5	0x07
ClearStatus()			0x08
LoadRefBlk()	Load start @ (offset from SL2 start address, aligned on a 16-byte boundary)	P1-P5	0x09
LoadPStack()	Stack address (offset from SL2 start address, loads 128 bytes starting at a 128-bit-aligned location)	P1-P5	0x0a
LoadRefBlk_Ind()	Load start @	@field0	0x0b
SaveBestMatch()	Parameter stack destination index	@field0	0x0c
RestoreBestMatch()	Parameter stack source index	@field0	0x0d
ExitOnMinReached()	None (acts as EndPgm() if AbsMinReached is set, otherwise act as nop)		0x0e
WaitOnSignal()	None (waits for OCP sync command)		0x0f
LoadInstBuf()	Program start @ (offset from SL2 start address, 16-bit, aligned on a 16-byte boundary) Number of instructions (instructions must be aligned into 64-bit bins, padded with zeroes towards the MSBs)	P1-P5 P0	0x10
Memcopy()	Block source start @ (offset from SL2 start address, 16-bit, aligned to a 16-byte boundary) Block destination start @ (offset from SL2 start address, 16-bit, aligned to a 16-byte boundary) Number of 32 bytes transferred (copy of Nx32 bytes from source to destination, in L2)	@field0 @field1 P0	0x11
LoadMVCT()	Load address (offset from SL2 start address. Loads both X and Y MVC tables, from 2 contiguous 256-bit-aligned words).	P1-P5	0x12
Interpolate()	Search window base @, absolute Filter start @ Store @ (aligned to a 16-byte boundary) Line size Include last best-match info (Y, N) Block size (8x8, 16x16, 24x24)	@field0 @field1 @field3 P0 P4 P5	0x13 ~ 0x17
GenerateIT()	None (triggers interrupt from inside the program, if IT enabled)		0x1e
Endpgm()	None (indicates program execution completion)		0x1f

Instruction format is described in the iLF/iME command control. For more information, see Section 14.3.5.1.1, *Commands*.

14.3.7.1.2 Commands

The iME hardware accelerator uses all the generic commands described in Section 14.3.5.1.1, *Commands*, plus a sync() command. The sync() command is described in Table 14-41.

Table 14-41. iME Specific Command

Command	Description	Code
sync()	This command resets the iME_CONFIGREG[6] WAITING_FOR_SYNC_SIGNAL bit if it is already set; otherwise, the iME_CONFIGREG[5] SYNC_SIGNAL_SET bit is set, and is then reset when the WaitOnsignal() instruction is decoded.	0x7

14.3.7.1.3 Execution States

For information about iME commands, see Section 14.3.5.1.1, *Commands*.

14.3.7.2 Instructions

The instructions described in Section 14.3.7.2.1, *ErrorCalc()*, to Section 14.3.7.2.16, *Interpolate()*, can be either written directly in the iME_PROGRAMBUFFERLINENLSBi and iME_PROGRAMBUFFERLINENMSBi registers or written in the last 32K bytes of L2 memory and loaded into the iME register by the LoadInstBuf() instruction. The function parameters are passed directly in the OPcode of the instruction.

For instructions common to the iLF and iME coprocessors-LoadPStack(), LoadInstBuf(), GenerateIT(), EndPgm()-see Section 14.3.5.2, *Instruction Set*.

Each instruction is stored in two 32-bit registers. There are 256 sets of two instruction registers in the iME. The iME can buffer 256 instructions at a time. Code with more than 256 instructions can be written if the final instruction is a LoadInstBuf with the parameters set toward another 256 instruction sequence in SL2 memory.

14.3.7.2.1 ErrorCalc()

Function: Calculates the error between the reference block and a block in the search area. The result of this instruction is an error value and an address corresponding to the block compared in the search area. These two results are stored in a single 32-bit word in the error table. The index in the error table is determined by the ErrorNum parameter. The MVParams describe how the two blocks are compared. If a parallel motion vector is used, multiple errors and addresses are generated. When two errors are computed in parallel, they are stored contiguously starting at an even location index. When four errors are computed in parallel, errors are stored contiguously starting at a multiple-of-four location index

Parameters:

- **WinStartAddr**: Index in the parameter stack pointing to the offset address from the SL2 start of the upper left pixel of the search window in SL2 memory
- **SearchStartAddr**: Index in the parameter stack pointing to the upper left pixel of the current block in the current search window
- **OffsetH:** Index in the parameter stack pointing to the horizontal offset (expressed in bytes) unsigned
- **OffsetV:** Index in the parameter stack pointing to the vertical offset (expressed in bytes) unsigned
- **MVParams:** Index in the parameter stack pointing to the motion vector parameters
- **SearchWinLineLength:** Line length of the search window expressed in pixels. Value = 0 to 255.
- **OffsetCode**: Code of the offset to determine which neighbor block is to be processed. Value = 0 to 8, defined as follows:

CENTRAL	0x0
UP_LEFT	0x1
UP	0x2
UP_RIGHT	0x3
LEFT	0x4
RIGHT	0x5
DOWN_LEFT	0x6
DOWN	0x7
DOWN_RIGHT	0x8

- **ErrorNum:** Index of the error stored in the error table. Value = 0 to 15.
- **UseBestMatchInfo:** Initial address flag; if it equals 0, the current block address is based on the WindowStartAddress and SearchStartAddress parameters. If it equals 1, the address is based on the address of the block having the best error. It is saved in the iME internal register LatestError.

Table 14-42 shows the iME ErrorCalc() MSB format.

Table 14-42. iME ErrorCalc() MSB Format

63	62	61	60	59	58	57	56	55	54	53	52	51	50	49	48	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32
Unused (pad bit)									SWBP	OPcode					@field0					@field1						@field2				@field3		
-	-	-	-	-	-	-	-	-	0	0	0	0	0	0	WinStartAddr					SearchStartAddr						OffsetH				OffsetV		

Table 14-43 shows the iME ErrorCalc() LSB format.

Table 14-43. iME ErrorCalc() LSB Format

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
@field3			@field4						P0						P1								P2				P3	P4		P5		
OffsetV			MVParams					SearchWinLineLength							OffsetCode							ErrNum					-	UseBestMatchInfo	-	-		

14.3.7.2.2 SaveErrs()

Function: Stores the iME internal error table in shared L2 memory. The L2 address is given through a parameter. iME internal errors are in a 16-entry register file (iME_ERRORTABLEj) containing final 16-bit errors and associated 16-bit addresses.

Parameters:

- **SL2Address**: Offset in the SL2 space (from the SL2 start address) of the array used to store the iME internal error table. Value = 0 to 0xFFFF. The address of the SL2 array must be 128-bit-aligned.

Table 14-44 shows the iME SaveErrs() MSB format.

Table 14-44. iME SaveErrs() MSB Format

63	62	61	60	59	58	57	56	55	54	53	52	51	50	49	48	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32
Unused (pad bit)									SWBP	OPcode					@field0					@field1						@field2				@field3		
-	-	-	-	-	-	-	-	-	0	0	0	0	1	0	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-

Table 14-45 shows the iME SaveErrs() LSB format.

Table 14-45. iME SaveErrs() LSB Format

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
@field3			@field4						P0						P1								P2				P3	P4		P5		
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	SL2Address															

14.3.7.2.3 SaveStatus()

Function: Stores the iME best-match address in the shared L2 memory. The L2 address is given through a parameter.

Parameters:

- **SL2Address:** Offset in the SL2 space (from the SL2 start address) of the array used to store iME status values. Value = 0 to 0xFFFF. The address of the SL2 array must be 128-bit-aligned.

Table 14-46 shows the iME SaveStatus() MSB format.

Table 14-46. iME SaveStatus() MSB Format

63	62	61	60	59	58	57	56	55	54	53	52	51	50	49	48	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32
Unused (pad bit)									SWBP	OPcode					@field0					@field1					@field2					@field3	
-	-	-	-	-	-	-	-	-	0	0	0	0	1	1	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-

Table 14-47 shows the iME SaveStatus() LSB format.

Table 14-47. iME SaveStatus() LSB Format

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
@field3			@field4					P0								P1								P2				P3	P4	P5	
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	SL2Address															

14.3.7.2.4 RestoreErrs()

Function: Restores the iME internal error table from shared L2 memory. The L2 address is given through a parameter. iME internal errors are in a 16-entry register file (iME_ERRORTABLEj) containing final 16-bit errors and associated 16-bit addresses.

Parameters:

- **SL2Address:** Offset in the SL2 space (from the SL2 start address) of the array used to download the error table into iME internal registers. Value = 0 to 0xFFFF. The address of the SL2 array must be 128-bit-aligned.

Table 14-48 shows the iME RestoreErrs() MSB format.

Table 14-48. iME RestoreErrs() MSB Format

63	62	61	60	59	58	57	56	55	54	53	52	51	50	49	48	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32
Unused (pad bit)									SWBP	OPcode					@field0					@field1					@field2					@field3	
-	-	-	-	-	-	-	-	-	0	0	0	1	0	0	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-

Table 14-49 shows the iME RestoreErrs() LSB format.

Table 14-49. iME RestoreErrs() LSB Format

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
@field3			@field4					P0								P1								P2				P3	P4	P5	
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	SL2Address															

14.3.7.2.5 Mcompare ()

Function: Compares 5 to 16 errors stored in the iME error table and stores the best match (the smallest error) in the IVA.iME_LATESTERRORS register with the corresponding address (IVA.iME_LATESTERRORS has the same format as the error table). Before performing the comparison between errors stored in the error table, a motion vector cost can be applied to errors. If the comparison includes the IVA.iME_LATESTERRORS register, the IncludeLastBest parameter must be set to 1. This instruction should be explicitly executed after 16 ErrorCalc() instructions.

Parameters:

- **MVType:** Index in the parameter stack pointing to the motion vector type. Value = 0 to 32.
- **Offset:** Offset for motion vector cost functionality. Value = 0 to 255.
- **ErrorStartNum:** Index of the first error to compare in the iME error table. Value = 0 to 14.
- **NumOfErrs:** Number of errors to compare, excluding the last best error. Value must be at least 5.
- **IncludeLastBest**: If it equals 0, it compares the errors to only the error table values. If it equals 1, it compares the errors to the best error already found. Value = 0 or 1.
- **CompareEvery**: Error table stride code. The stride is derived from the CompareEvery parameter by the relation

  stride = 1 << CompareEvery.

  The values possible for this parameter are: 0, 1, 2, or 3, indicating a comparison to 1, 2, 4, or 8 errors, respectively.

Table 14-50 shows the iME MCompare() MSB format.

Table 14-50. iME MCompare() MSB Format

63	62	61	60	59	58	57	56	55	54	53	52	51	50	49	48	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32
Unused (pad bit)									SWBP	OPcode					@field0					@field1					@field2					@field3	
-	-	-	-	-	-	-	-	-	0	0	0	1	0	1	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	

Table 14-51 shows the iME MCompare() LSB format.

Table 14-51. iME MCompare() LSB Format

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
@field3			@field4							P0						P1						P2							P3	P4	P5	
-	-	-	MVType						Offset						ErrorStartNum					NumOfErrs								-	IncludeLastBest	CompareEvery		

14.3.7.2.6 Mcompare2 ()

Function: Compares two errors stored in the iME error table and stores the best match (the smallest error) in the IVA.iME_LATESTERRORS register with the corresponding address (IVA.iME_LATESTERRORS has the same format as the error table). Before performing the comparison between errors stored in the error table, a motion vector cost can be applied to errors. If the comparison includes the IVA.iME_LATESTERRORS register, the IncludeLastBest parameter must be set to 1.

Parameters:

- **MVType:** Index in the parameter stack pointing to the motion vector type. Value = 0 to 32.
- **Offset**: Offset for motion vector cost functionality. Value = 0 to 255.
- **ErrorStartNum:** Index of the first error to compare in the iME error table. Value = 0 to 14.
- **IncludeLastBest:** If it equals 0, it compares the errors to only the error table values. If it equals 1, it compares the errors to the best error already found. Value = 0 or 1.
- **CompareEvery**: Error table stride code. The stride is derived from the CompareEvery parameter by

the relation

stride = 1 << CompareEvery.

The values possible for this parameter are: 0, 1, 2, or 3, indicating a comparison to 1, 2, 4, or 8 errors, respectively.

Table 14-52 shows the iME MCompare2() MSB format.

Table 14-52. iME MCompare2() MSB Format

63	62	61	60	59	58	57	56	55	54	53	52	51	50	49	48	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32
Unused (pad bit)									SWBP	OPcode					@field0					@field1					@field2					@field3	
-	-	-	-	-	-	-	-	-	0	0	0	1	1	0	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	

Table 14-53 shows the iME MCompare2() LSB format.

Table 14-53. iME MCompare2() LSB Format

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
@field3			@field4							P0						P1						P2						P3	P4	P5	
-	-	-	MVType							Offset						ErrorStartNum						NumOfErrs						-	IncludeLastBest	CompareEvery	

14.3.7.2.7 Mcompare4 ()

Function: Compares four errors stored in the iME error table and stores the best match (the smallest error) in the IVA.iME_LATESTERRORS register with the corresponding address (IVA.iME_LATESTERRORS has the same format as the error table). Before performing the comparison between errors stored in the error table, a motion vector cost can be applied to errors. If the comparison includes the IVA.iME_LATESTERRORS register, the IncludeLastBest parameter must be set to 1.

Parameters:

- **MVType:** Index in the parameter stack pointing to the motion vector type. Value = 0 to 32.
- **Offset**: Offset for motion vector cost functionality. Value = 0 to 255.
- **ErrorStartNum:** Index of first error to compare in iME error table. Value = 0 to 14.
- **IncludeLastBest:** If it equals 0, it compares the errors to only the error table values. If it equals 1, it compares the errors to the best error already found. Value = 0 or 1.
- **CompareEvery**: Error table stride code. The stride is derived from the CompareEvery parameter by the relation

  stride = 1 << CompareEvery.

  The values possible for this parameter are: 0, 1, 2, or 3, indicating a comparison to 1, 2, 4, or 8 errors, respectively.

Table 14-54 shows the iME MCompare4() MSB format.

Table 14-54. iME MCompare4() MSB Format

63	62	61	60	59	58	57	56	55	54	53	52	51	50	49	48	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32
Unused (pad bit)									SWBP	OPcode					@field0					@field1					@field2					@field3	
-	-	-	-	-	-	-	-	-	0	0	0	1	1	1	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	

Table 14-55 shows the iME MCompare4() LSB format.

Table 14-55. iME MCompare4() LSB Format

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
@field3			@field4							P0						P1						P2						P3	P4	P5	
-	-	-	MVType					Offset								ErrorStartNum						NumOfErrs						-	IncludeLastBest	CompareEvery	

14.3.7.2.8 ClearStatus()

Function: Clears the Address field of the best match, the absolute-mean-reached flag, and the iME_XMVCTm and iME_YMVCTm tables. Sets the Error field of the best match to 0xFFFF).

Parameters: This function does not take any parameter.

Table 14-56 shows the iME ClearStatus() MSB format.

Table 14-56. iME ClearStatus() MSB Format

63	62	61	60	59	58	57	56	55	54	53	52	51	50	49	48	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32
Unused (pad bit)									SWBP	OPcode					@field0					@field1					@field2					@field3	
-	-	-	-	-	-	-	-	-	0	0	1	0	0	0	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-

Table 14-57 shows the iME ClearStatus() LSB format.

Table 14-57. iME ClearStatus() LSB Format

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
@field3			@field4							P0						P1						P2						P3	P4	P5	
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-

14.3.7.2.9 LoadRefBlk()

Function: Loads a reference block (16×16 pixels) from L2 memory to be the reference block for computing motion estimates. The L2 address is given through a parameter.

Parameters:

- **SL2Address:** Offset in the SL2 space (from the SL2 start address) of the array used to download the reference block from SL2 to iME internal registers. Value = 0 to 0xFFFF. The address of the SL2 array must be 128-bit-aligned.

Table 14-58 shows the iME LoadRefBlk() MSB format.

Table 14-58. iME LoadRefBlk() MSB Format

63	62	61	60	59	58	57	56	55	54	53	52	51	50	49	48	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32
Unused (pad bit)									SWBP	OPcode					@field0					@field1					@field2					@field3	
-	-	-	-	-	-	-	-	-	0	0	1	0	0	1	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-

Table 14-59 shows the iME LoadRefBlk() LSB format.

Table 14-59. iME LoadRefBlk() LSB Format

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
@field3			@field4					P0								P1						P2						P3	P4	P5	
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	SL2Address															

14.3.7.2.10 LoadRefBlk_Ind()

Function: Loads a reference block (16 × 16 pixels) from L2 memory to be the reference block for computing motion estimates. The L2 address is given through the parameter stack. This instruction is similar to the LoadRefBlk() instruction, but the parameter is passed through the stack instead of being directly hard coded in the OPcode.

Parameters:

- **RefBlockAddr:** Index in the parameter stack pointing to the 16-bit offset address from the SL2 start of the array used to download the reference block from SL2 to the iME internal registers. Value = 0 to 0xFFFF. The address of the SL2 array stored in the stack must be 128-bit-aligned.

Table 14-60 shows the iME LoadRefBlk_Ind() MSB format.

Table 14-60. iME LoadRefBlk_Ind() MSB Format

63	62	61	60	59	58	57	56	55	54	53	52	51	50	49	48	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32
Unused (pad bit)									SWBP	OPcode					@field0					@field1					@field2					@field3		
-	-	-	-	-	-	-	-	-	0	0	1	0	1	1	RefBlockAddr					-	-	-	-	-	-	-	-	-	-	-	

Table 14-61 shows the iME LoadRefBlk_Ind() LSB format.

Table 14-61. iME LoadRefBlk_Ind() LSB Format

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
@field3			@field4					P0								P1						P2						P3	P4	P5	
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-

14.3.7.2.11 SaveBestMatch()

Function: SaveBestMatch() saves the best match into the stack parameter entry, for which the index is @field0. The error and address fields are swapped in the parameter stack so that the address field is in the LSBs, and can be used for address generation.

Parameters:

- **Dst:** Index in the parameter stack to save the best match. Valid destinations are parameter stack entries 16 to 31.

Table 14-62 shows the iME SaveBestMatch() MSB format.

Table 14-62. iME SaveBestMatch() MSB Format

63	62	61	60	59	58	57	56	55	54	53	52	51	50	49	48	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32
Unused (pad bit)									SWBP	OPcode					@field0					@field1					@field2					@field3		
-	-	-	-	-	-	-	-	-	0	0	1	1	0	0	Dst					-	-	-	-	-	-	-	-	-	-	-	

Table 14-63 shows the iME SaveBestMatch() LSB format.

Table 14-63. iME SaveBestMatch() LSB Format

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
@field3			@field4					P0								P1						P2						P3	P4	P5	
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-

14.3.7.2.12 RestoreBestMatch()

Function: LoadBestMatch() loads the best match with the stack parameter entry, for which the index is @field0. The error and address fields are swapped in the process, so that the address field is in the MSBs of the best match register.

Parameters:

- **Src:** Index in the parameter stack to restore the best match. Valid destinations are parameter stack entries 16 to 31.

Table 14-64 shows the iME RestoreBestMatch() MSB format.

Table 14-64. iME RestoreBestMatch() MSB Format

63	62	61	60	59	58	57	56	55	54	53	52	51	50	49	48	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32
Unused (pad bit)									SWBP	OPcode					@field0					@field1					@field2					@field3		
-	-	-	-	-	-	-	-	-	0	0	1	1	0	1	Src					-	-	-	-	-	-	-	-	-	-	-	-

Table 14-65 shows the iME RestoreBestMatch() LSB format.

Table 14-65. iME RestoreBestMatch() LSB Format

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
@field3			@field4					P0								P1						P2						P3	P4	P5	
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-

14.3.7.2.13 ExitOnMinReached()

Function: ExitOnMinReached() terminates the iME program and sets the iME_IRQLOG[0] ENDPGMEVENTLOG bit, if the iME_ABSMINREACHED[0] ABSMINREACHED bit is set.

Parameters: This function does not take any parameter.

Table 14-66 shows the iME LoadRefBlk() MSB format.

Table 14-66. iME ExitOnMinReached() MSB Format

63	62	61	60	59	58	57	56	55	54	53	52	51	50	49	48	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32
Unused (pad bit)									SWBP	OPcode					@field0					@field1					@field2					@field3		
-	-	-	-	-	-	-	-	-	0	0	1	1	1	0	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-

Table 14-67 shows the iME LoadRefBlk() LSB format.

Table 14-67. iME ExitOnMinReached() LSB Format

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
@field3			@field4					P0								P1						P2						P3	P4	P5	
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-

14.3.7.2.14 WaitOnSignal()

Function: If the iME_CONFIGREG[5] SYNC_SIGNAL_SET bit is reset, WaitOnSignal() sets the iME_CONFIGREG[6] WAITING_FOR_SYNC_SIGNAL bit. Then, the iME waits until a sync command is sent through the iME_COMMANDREG register (sync() command is value 0x7). If the iME_CONFIGREG[5] SYNC_SIGNAL_SET bit is set, WaitOnSignal() resets the iME_CONFIGREG[5] SYNC_SIGNAL_SET bit, it does not set the iME_CONFIGREG[6] WAITING_FOR_SYNC_SIGNAL bit of the configuration register, and the next instruction executes normally.

Parameters: This function does not take any parameter.

Table 14-68 shows the iME LoadRefBlk() MSB format.

Table 14-68. iME WaitOnSignal() MSB Format

63	62	61	60	59	58	57	56	55	54	53	52	51	50	49	48	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32
Unused (pad bit)									SWBP	OPcode					@field0					@field1					@field2					@field3		
-	-	-	-	-	-	-	-	-	0	0	1	1	1	1	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-

Table 14-69 shows the iME LoadRefBlk() LSB format.

Table 14-69. iME WaitOnSignal() LSB Format

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
@field3			@field4					P0								P1						P2						P3	P4	P5	
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-

14.3.7.2.15 Memcopy()

Function: Copies up to 256 lines of 256 bits each from shared L2 memory to shared L2 memory. The number of lines is given through a parameter as the source and destination addresses.

Parameters:

- **SourceAddress:** Index in the parameter stack pointing to the offset in the SL2 space (from the SL2 start address) of the SL2 array to be copied. Value = 0 to 0x1F.
- **DestAddress:** Index in the parameter stack pointing to the offset in the SL2 space (from the SL2 start address) of the destination SL2 array. Value = 0 to 0x1F.
- **NbElements:** Number of 256-bit lines to be copied

  NbElements varies from 0 to 255.

  If NbElements = 0, 256 lines are copied

  Else NbElements lines are copied

Table 14-70 shows the iME Memcopy() MSB format.

Table 14-70. iME Memcopy() MSB Format

63	62	61	60	59	58	57	56	55	54	53	52	51	50	49	48	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32
Unused (pad bit)									SWBP	OPcode					@field0					@field1					@field2					@field3		
-	-	-	-	-	-	-	-	-	0	1	0	0	0	1	SourceAddress					DestAddress					-	-	-	-	-	-		

Table 14-71 shows the iME Memcopy() LSB format.

Table 14-71. iME Memcopy() LSB Format

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
@field3			@field4					P0								P1						P2						P3	P4	P5	
-	-	-	-	-	-	-	-	NbElements								-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-

Table 14-72. iME LoadMVCT() LSB Format

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
@field3			@field4					P0								P1						P2						P3	P4	P5	
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	SL2Address															

14.3.7.2.16 Interpolate()

Function: The interpolate function

Parameters:

- **WinStartAddr:** Index in the parameter stack pointing to the offset of the upper-left pixel of the search window in the SL2 space (offset from the SL2 start address)
- **FilterStartAddr:** Index in the parameter stack pointing to the upper-left pixel of the block to interpolate in the search window
- **StoreAddress**: Index in the parameter stack pointing to the upper-left pixel in plane 0 of the interpolated block. It is 128-bit-aligned.
- **LineSize**: Line length, in pixels, of the search window. Value = 8 to 255.
- **IncludeLastBest**: If it equals 0, addressing is based on parameter stack entry. If it equals 1, addressing is based on the best match register. Value = 0 or 1..
- **BlockType:** Size of the block:

  Values:

  – 0x1 for an 8*8 block
  – 0x2 for a 16*16 block
  – 0x3 for a 24*24 block

Table 14-73 shows the iME Interpolate() MSB format.

Table 14-73. iME Interpolate() MSB Format

63	62	61	60	59	58	57	56	55	54	53	52	51	50	49	48	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32
Unused (pad bit)									SWBP	OPcode						@field0				@field1					@field2					@field3	
-	-	-	-	-	-	-	-	-	0	1	0	0	1	1	WinStartAddr					FilterStartAddr				-	-	-	-	-	-		

Table 14-74 shows the iME Interpolate() LSB format.

Table 14-74. iME Interpolate() LSB Format

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
@field3			@field4					P0								P1						P2						P3	P4	P5	
StoreAddress	-	-	-	-	-	LineSize										-	-	-	-	-	-	-	-	-	-	-	-	-	IncludeLastBest	BlockType	

14.3.7.3 Parameters

14.3.7.3.1 Instruction Buffer in L2

Instruction format in L2 is common to iLF and iME. For information about formatting instructions in L2, see Section 14.3.5.3.1, *Instructions*.

14.3.7.3.2 Generic Parameter Stack in L2

The parameter stack is composed of 16 words of 16 bits and 16 words of 32 bits. When the parameter stack is downloaded from L2 memory to the iME internal registers, a 16-bit mask is applied on the 16 MSB of the first 16 words. The 32 bits of the last 16 words are downloaded in the stack.

The interpolate() instruction use the last 16 words (which are 32-bit words) as interpolate parameters. For more information about interpolate parameters, see Section 14.3.7.3.3, *Interpolate Parameters*.

SaveBestMatch() and RestoreBestMatch() use one of the last 16 words to store or to restore the best match entry. The word used is defined in the instruction and should be in the last 16 words because the stored value is a 32-bit value.

Data in the generic parameter stack have no order. When required by an instruction, these data are accessed with the help of an index passed as a parameter inside the instruction. This index references the position of the word in the generic parameter stack. The entire parameter stack is addressed through index parameters (0-31).

The generic parameter stack contains three types of parameters:

- Motion vector parameters
- L2 addresses
- Offset

The motion vector parameter stored in the generic parameter stack is 16 bits wide and is composed of 4 fields. Table 14-75 shows the motion vector parameter format.

Table 14-75. iME Motion Vector Parameter Format

31:16	15:12	11:8	7:4	3:0
Reserved	Iteration count	Line offset	Lines per iteration	Motion vector type

In the parallel motion vector type, the number of lines used for computation is defined by a variable parameter. Another parameter defines the offset, in number of lines, to read in the reference block. An iteration parameter defines how many times parallel processing is executed (1, 2, or 4 times; new errors are initialized each iteration). The following parameters are used only with the parallel motion vector type. When other motion vector types are used, the field must be filled with 0s:

- Iteration count: Parallel processing iteration (1, 2, or 4)
- Line offset: Line start, from line 0, into the reference block (0, 4, 8, or 12). This field is also used in 4MV and 16 MV.
- Lines per iteration: Number of lines processed per iteration of parallel processing (4, 8, or 16)

The motion vector type parameter determines which motion vector and parameters are used for computation. Table 14-76 shows the different motion vector types with a short description of each and the corresponding encoding in the motion vector type field.

Table 14-76. iME Motion Vector Type

MV_type encoding	MV_type name	Description
0x0	1MV_16	16-pixel-wide image
0x1	1MV_8	Downsampled 8-pixel-wide image
0x2	1MV_4	Downsampled 4-pixel-wide image
0x3	Reserved	-

Table 14-76. iME Motion Vector Type (continued)

MV_type encoding	MV_type name	Description
0x4	4MV_L	8-pixel-wide image, left side
0x5	4MV_R	8-pixel-wide image, right side
0x6	Reserved	-
0x7	Reserved	-
0x8	16MV_0	4-pixel-wide image, first quarter
0x9	16MV_1	4-pixel-wide image, second quarter
0xA	16MV_2	4-pixel-wide image, third quarter
0xB	16MV_3	4-pixel-wide image, fourth quarter
0xC	pMV_8	2 // 8-pixel-wide images
0xD	pMV_4	4 // 4-pixel-wide images
0xE	Reserved	-
0xF	Reserved	-

14.3.7.3.3 Interpolate Parameters

When the Interpolate() instruction is used, the interpolation parameters are extracted from the parameter stack. Interpolate parameters are 32-bit-wide words stored in the last 16 words of the parameter stack. For information about how to fill the parameter stack with generic parameters, see Section 14.3.7.3.2, *Generic Parameter Stack in L2*.

Interpolate instructions also use eight coefficients stored in the IVA.iME_COEFFREGBANKl registers. These registers must be loaded through the local interconnect, because there is no dedicated instruction to load them from L2 memory. These eight coefficients are stored on 4*32-bit registers, and their values for the different supported codecs are listed in Table 14-77.

Table 14-77. iME Interpolate Coefficient Register Bank

Register Address Offset	MPEG4/H263	H264	WMV9
0x0980	0x00 00 00 00	0x00 FB 00 01	0x00 09 00 FF
0x0984	0x00 00 00 00	0x00 00 00 14	0x00 35 00 FC
0x0988	0x00 00 00 00	0x00 00 00 00	0x00 FD 00 12
0x098C	0x00 00 00 00	0x00 00 00 00	0x00 00 00 24

Values for the interpolate parameter stack for MEPG4, H264, and WMV9 are listed in Table 14-78, Table 14-79, and Table 14-80, respectively. These values are defined by the different video codecs.

H263 and WMV9 use two different sets of parameters, depending on the global rounding frame control parameter (application-dependent). These two sets differ only in their rounding values.

Table 14-78. iME Interpolate Parameter Stack for MPEG4/H263

GDP Parameter Stack Value	Value	
	Rounding = 0	Rounding = 1
0	0x20 00 00 00	0x20 00 00 00
1	0x00 00 00 00	0x00 00 00 00
2	0x00 00 10 00	0x00 00 10 00
3	0x00 00 00 00	0x00 00 00 00
4	0x00 00 00 00	0x00 00 00 00
5	0x00 00 00 00	0x00 00 00 00
6	0x00 00 00 00	0x00 00 00 00
7	0x00 01 01 00	0x00 00 00 80
8	0x00 00 00 00	0x00 00 00 00
9	0x91 49 00 00	0x91 49 00 00

Table 14-78. iME Interpolate Parameter Stack for MPEG4/H263 (continued)

GDP Parameter Stack Value	Value	
	Rounding = 0	Rounding = 1
10	0x00 02 82 00	0x00 02 82 00
11	0x00 00 04 00	0x00 00 04 00
12	0x00 00 00 00	0x00 00 00 00
13	0x00 00 00 00	0x00 00 00 00
14	0x00 00 00 00	0x00 00 00 00
15	0x00 00 00 00	0x00 00 00 00

Table 14-79. iME Interpolate Parameter Stack for H264

GDP Parameter Stack Value	Value
0	0x37 00 00 00
1	0x00 00 00 00
2	0x00 00 10 00
3	0x00 00 00 00
4	0x00 01 22 10
5	0x00 00 00 00
6	0x00 00 00 00
7	0x00 10 08 50
8	0x00 00 00 00
9	0x90 09 00 00
10	0x00 00 0a 00
11	0x00 00 14 00
12	0x00 00 00 00
13	0x00 00 00 00
14	0x00 00 00 00
15	0x00 00 00 00

Table 14-80. iME Interpolate Parameter Stack for WMV9

GDP Parameter Stack value	Value	
	Rounding = 0	Rounding = 1
0	0X0D 30 40 50	0X0D 30 40 50
1	0XC0 60 70 80	0XC0 60 70 80
2	0XD1 90 A0 B0	0XD1 90 A0 B0
3	0XC2 60 70 80	0XC2 60 70 80
4	0X08 82 34 58	0X08 82 34 58
5	0X43 28 80 11	0X43 28 80 11
6	0X82 66 28 85	0X82 66 28 85
7	0X00 20 10 08	0X00 1F 0E 0F
8	0X34 12 04 F8	0XEF EF F3 00
9	0X00 00 00 E0	0X00 00 01 00
10	0X88 00 8C 00	0X88 00 8C 00
11	0XD1 01 10 1B	0XD1 01 10 23
12	0XC9 12 14 7D	0XC9 12 14 85
13	0X8C 68 58 01	0X9A B1 1A 9C
14	0X9A B1 1A 9C	0X00 00 00 00
15	0XDC 93 11 46	0XDC 93 11 46

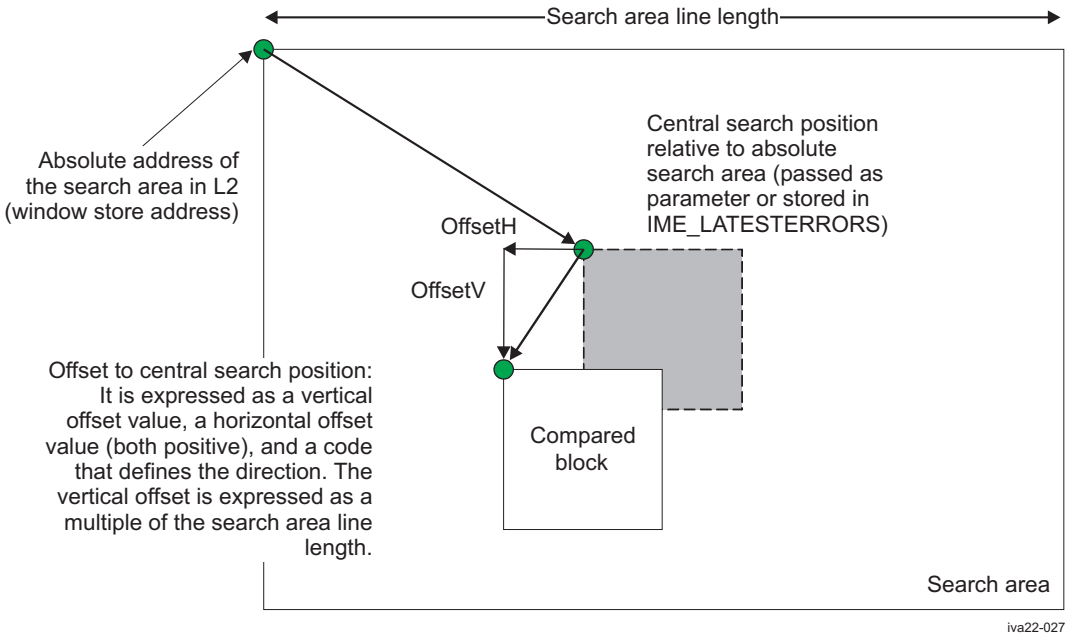
14.3.7.3.4 Pixel Format in L2

Pixels are stored in L2 memory before being downloaded to iME internal memory. A set of three parameters is used to indicate which block is compared to the reference block stored in the iME internal memory:

- A search area must be given as a parameter to the ErrorCalc() instruction. This search area is composed of two parameters:
  - Windows store address parameter, referencing the start address of the search area
  - Search area line length parameter, indicating the line length of the search area in L2, which allows the iME to calculate all addresses in the search area
- The current central search position is also passed in parameter. This address references the upper-left pixel of the current block, considered the central block for the current motion estimate. This parameter can be ignored if the UseBestMatchInfo bit is set. When UseBestMatchInfo bit is set, the central search position address corresponds to the address stored in the IVA.iME_LATESTERRORS BESTMATCHADDRESS field.
- The offset from the central search is given through parameters OffsetH and OffsetV. These values are unsigned, and direction is given with the OffsetCode.

Figure 14-28 describes these parameters.

Figure 14-28. iME Pixel Format in L2



14.3.7.3.5 Motion Vector Cost Table Format in L2

The iME coprocessor includes two MVC tables, iME_XMVCTm and iME_YMVCTm, each composed of 16 entries of 16 bits. These tables contain cost-correction values used by the MC instruction. Each table is addressed by its two 4-bit indexes, to output one value per parallel comparison. These indexes are generated from an 8-bit OFFSET parameter, the MV type, and the error indexes. Table 14-81 describes how the indexes are calculated. The two selected MVC entries are added together and pipelined to be added to the error value before the comparison.

Table 14-81. iME Motion Vector Offset Index

MVType	Error Index	XMVC_index (4 bits)	YMVC_index (4 bits)
pMV4	0	OFFSET & 0xF	OFFSET >> 4
	1	(OFFSET+4) &0xF	(OFFSET+4) >> 4
	2	(OFFSET+8) &0xF	(OFFSET+8) >> 4
	3	(OFFSET+12) &0xF	(OFFSET+12) >> 4
	4	(OFFSET+64) &0xF	(OFFSET+64) >> 4
	5	(OFFSET+68) &0xF	(OFFSET+68) >> 4
	6	(OFFSET+72) &0xF	(OFFSET+72) >> 4
	7	(OFFSET+76) &0xF	(OFFSET+76) >> 4
	8	(OFFSET+128) &0xF	(OFFSET+128) >> 4
	9	(OFFSET+132) &0xF	(OFFSET+132) >> 4
	10	(OFFSET+136) &0xF	(OFFSET+136) >> 4
	11	(OFFSET+140) &0xF	(OFFSET+140) >> 4
	12	(OFFSET+192) &0xF	(OFFSET+192) >> 4
	12	(OFFSET+196) &0xF	(OFFSET+196) >> 4
	14	(OFFSET+200) &0xF	(OFFSET+200) >> 4
	15	(OFFSET+204) &0xF	(OFFSET+204) >> 4
pMV8	i%4=0	OFFSET & 0xF	OFFSET >> 4
	i%4=1	(OFFSET+8) & 0xF	(OFFSET+8) >> 4
	i%4=2	(OFFSET+128) & 0xF	(OFFSET+128) >> 4
	i%4=3	(OFFSET+136) & 0xF	(OFFSET+136) >> 4
Others	i	(OFFSET + i) & 0xF	(OFFSET + i) & 0xF

When the sum of the incoming error and the MV cost correction overflows the 16 bits, the local error is not updated. The comparator behaves as if the incoming data is clipped to the maximum positive integer, which by definition is greater than or equal to the current local error.

The MV cost tables can be loaded from memory with the loadMVCT() instruction and then reset by the clearStatus() instruction. The 16 entries of each table are packed so that one 256-bit read fills up one complete table.

14.3.7.3.6 Error Table

The iME coprocessor includes an error table that stores the result of ErrorCalc() to compute the minimum error reached. This table can be accessed through the local interconnect. It is composed of 16 32-bit registers, each composed of 16 bits coding the errors and 16 bits coding the address corresponding to the error. For more information, see Table 14-650, the IVA error table register description. This table is used by the MCompare(), MCompare2(), and MCompare4() instructions as input data. These functions output an error and an address stored in the IVA.iME_LATESTERRORS register. This register has the same format as the error table registers and can be used as input data during MCompare(), MCompare2(), and MCompare4() instructions, depending on the value of the IncludeLastBest parameter.

Table 14-82 shows the error table format.

Table 14-82. Error Table Format

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ET_ADDRESSN																ET_ERRORN															

The 16 MSBs (bits 16 to 31) denote the offset from the SL2 start address. The 16 LSBs (bits 0 to 15) denote the error field, that is, the absolute SAD value.

14.3.8 Video Accelerator/Sequencer Local Interconnect

The video accelerator/sequencer local interconnect allows the DSP megamodule and the sequencer to access the coprocessor, the sequence and video accelerator system control, the shared L2 memory interface, and the IVA2.2 EDMA.

All sequencer reads and writes are performed through the video accelerator/sequencer local interconnect. The DSP can access the interconnect with EFI instructions (EFSDW, EFSW, EFRW, and EFRDW). A write is always composed of an EFSDW (32b). Depending on the destination value associated with the instruction, the pair of DSP megamodule registers contains either the address (32b; only the lowest 20 bits are significant) and the 32b data, or only the 64b data (auto-incrementing, constant addressing mode). A read is composed of a pair of EFSW and EFRW or EFRDW, depending on the size of the read data. For consistency, reads have the same addressing modes as writes.

Note: The DSP accesses the shared L2 memory and the IVA2.2 EDMA through the IVA interconnect.

Table 14-83 shows the video accelerator/sequencer interconnect memory mapping. Addresses correspond to the DSP megamodule address using the EFI instruction. For sequencer access, all addresses must have an offset, described in Section 14.3.4.8, *Sequencer Memory Mapping*.

For more information about EFI instruction in IVA2.2, see Section 14.4.5, *IVA2.2 Extended Function Interface*.

Table 14-83. Video Accelerator/Sequencer Memory Mapping

Device Name	Start Address (Hex)	End Address (Hex)	Size	Description
L2 memory	0x0 0000	0x0 7FFF	32KB	L2 shared SRAM
Reserved	0x0 8000	0x1 FFFF	96KB	Reserved
DMA registers	0x2 0000	0x3 FFFF	128KB	Configuration registers
Reserved	0x4 0000	0x8 FFFF	320KB	Reserved
SEQ registers	0x9 0000	0x9 07FF	2KB	Configuration registers
Reserved	0x9 0800	0x9 0FFF	2KB	Reserved
Reserved	0x9 1000	0x9 3FFF	12KB	Reserved
SEQ DMEM	0x9 4000	0x9 4FFF	4KB	Sequencer data memory
Reserved	0x9 5000	0x9 7FFF	12KB	Reserved
SEQIMEM	0x9 8000	0x9 9FFF	8KB	Sequencer instruction memory
Reserved	0x9 A000	0x9 BFFF	8KB	Reserved
Video SYSC registers	0x9 C000	0x9 CFFF	4KB	Configuration registers
Reserved	0x9 D000	0x9 FFFF	12KB	Reserved
iME registers	0xA 0000	0xA 0FFF	4KB	Configuration registers
iLF registers	0xA 1000	0xA 1FFF	4KB	Configuration registers
Reserved	0xA 2000	0xF 7FFF	344KB	Reserved
Interconnect registers	0xF 8000	0xF BFFF	16KB	Configuration registers
Reserved	0xF C000	0xF FFFF	16KB	Reserved

14.3.9 SL2 Interface

The SL2 memory interface (SL2IF) provides access to L2 memory for the following units:

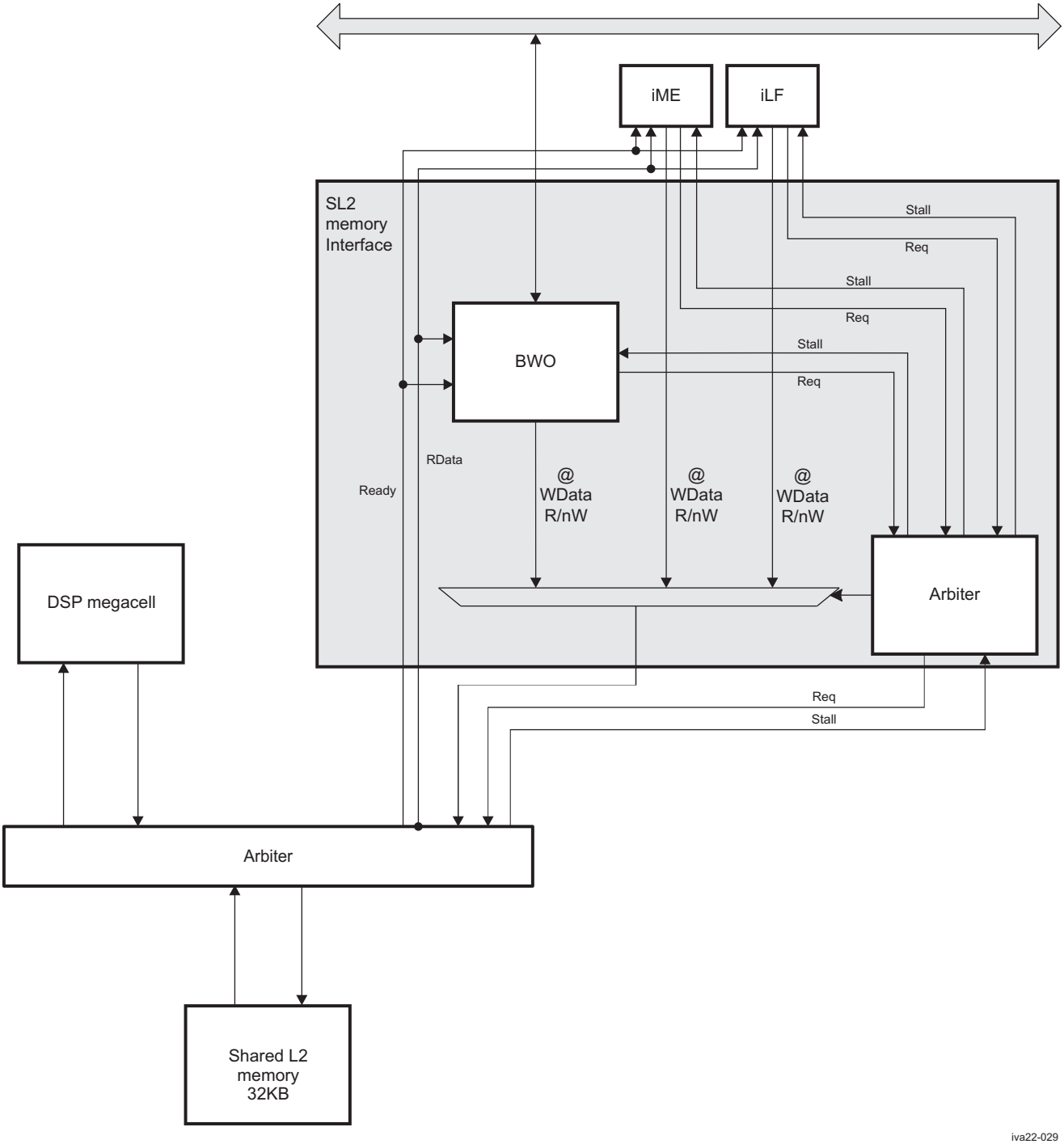
- iME
- iLF
- Video accelerator/sequencer interconnect

The SL2 memory interface is composed of two modules:

- Bandwidth optimizer (BWO) for video accelerator/sequencer local interface interconnect
- Arbiter

Figure 14-29 is a block diagram of the SL2 memory interface.

Figure 14-29. SL2 Memory Interface Block Diagram



14.3.9.1 BWO

The BWO optimizes write bandwidth and read latency for connections from/to the local interconnect from/to the shared L2 memory. It has two components:

- 256b write buffer to gather bursted write data before sending them to the shared L2
- 256b read buffer to prefetch a read line from which the burst is read

These buffers are completely decoupled; writes use the write buffer and reads use the read buffer. This is not a data cache, as there is no load-forwarding from the write buffer, and no prefetch buffer to be updated by an incoming write.

Note: There is no concept of dirty bit and write-back policy.

14.3.9.2 Arbiter

The arbiter allows the three modules (iME, iLF, BWO) to access L2 memory without taking care of concurrent access themselves. No bubble is introduced by the arbitration, which means that when a request is initiated by the iME, the iLF, or the BWO, a request is transmitted to the SL2. This arbitration ensures 100 percent bandwidth use. The arbitration is fair and does not starve an initiator for more than three cycles.

To return data to the correct initiator, the arbiter keeps track of the ordering of memory requests committed to the SL2 memory interface.

The SL2 memory interface can sustain a 2x16-byte request per SL2 memory interface clock cycle, whether the request is a read or write or interleaved reads and writes, and regardless of which module generates the request (iME, iLF, or local interconnect port initiators), if the initiator is not preempted by the others (no competition) and provides an equivalent input throughput.

14.3.9.3 Restrictions on SL2 Memory Usage

Because accesses to SL2 (from iME, iLF, or any other initiator with access to the SL2IF interface) are not checked for access permission, the SL2 memory must not be used for memory-protected or sensitive data.

SL2 memory must not be used to store sequencer instructions to be fetched directly by the sequencer; the SL2IF is not designed to efficiently serve sequencer instruction fetches. SL2 memory can hold sequencer instructions, but they must be transferred by the EDMA module into the ITCM before being executed.

14.3.9.4 Error Management

The SL2 memory interface responds to the following conditions with an in-band error (SResp = ERR):

- Nonaligned address (MAddr[1:0] <>0)
- Unsupported command (MCmd not belonging to READ/WR/WRNP)

All valid accesses are propagated to the SL2 memory interface; the address must always be mapped in SL2.

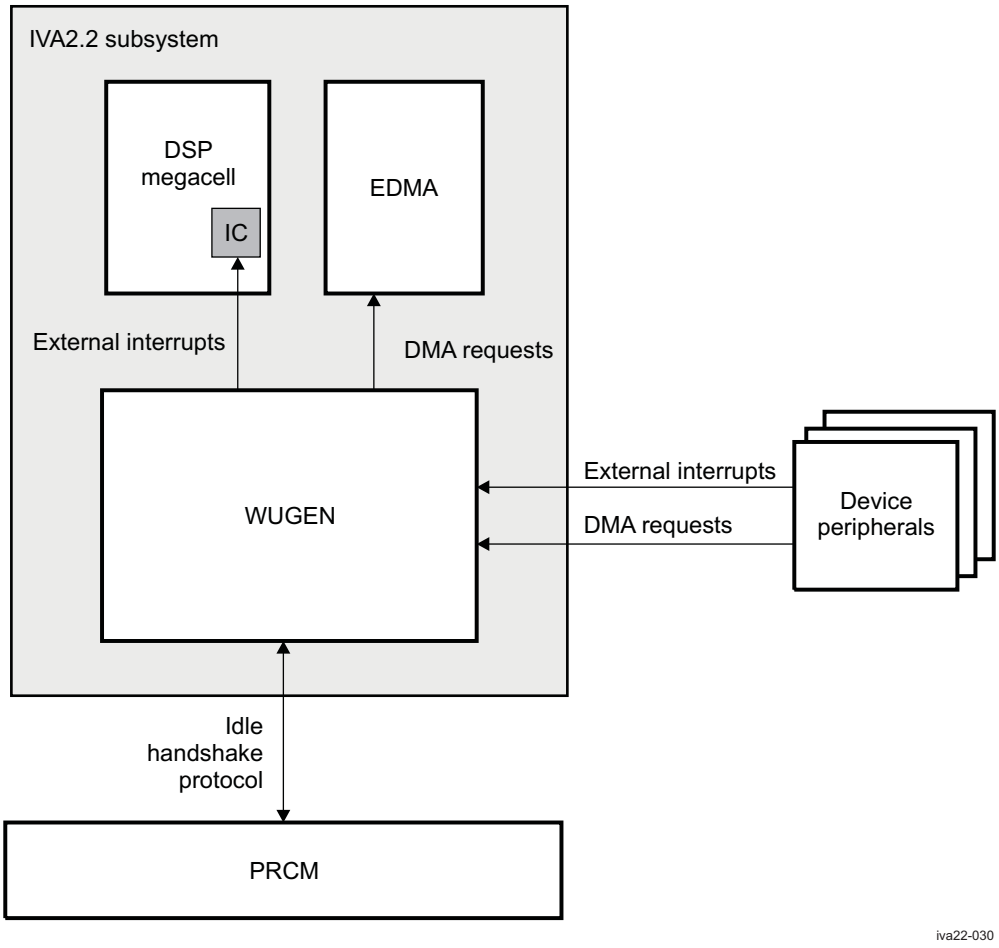
14.3.10 Wake-Up Generator

The wake-up generator (WUGEN) controls the following:

- Implementing the IVA2.2 idle handshake protocol with the PRCM
- Resynchronizing interrupts and DMA requests from device peripherals external to the IVA2.2 subsystem
- Blocking the interrupts and DMA requests to the IVA2.2 on clean boundaries, when the WUGEN is asked to go into IDLE state
- Detecting the enabled wake-up interrupts and DMA requests and generating a wake-up event to the PRCM
- Formatting interrupts to DSP megamodule interrupt format

Figure 14-30 shows the WUGEN in the IVA2.2 subsystem and its interactions with other submodules.

Figure 14-30. IVA2.2 WUGEN Description



14.3.10.1 Interrupts, DMA Requests, and Event Management

Interrupts and DMA requests are generated the same way in the WUGEN module. In this section, both are called events.

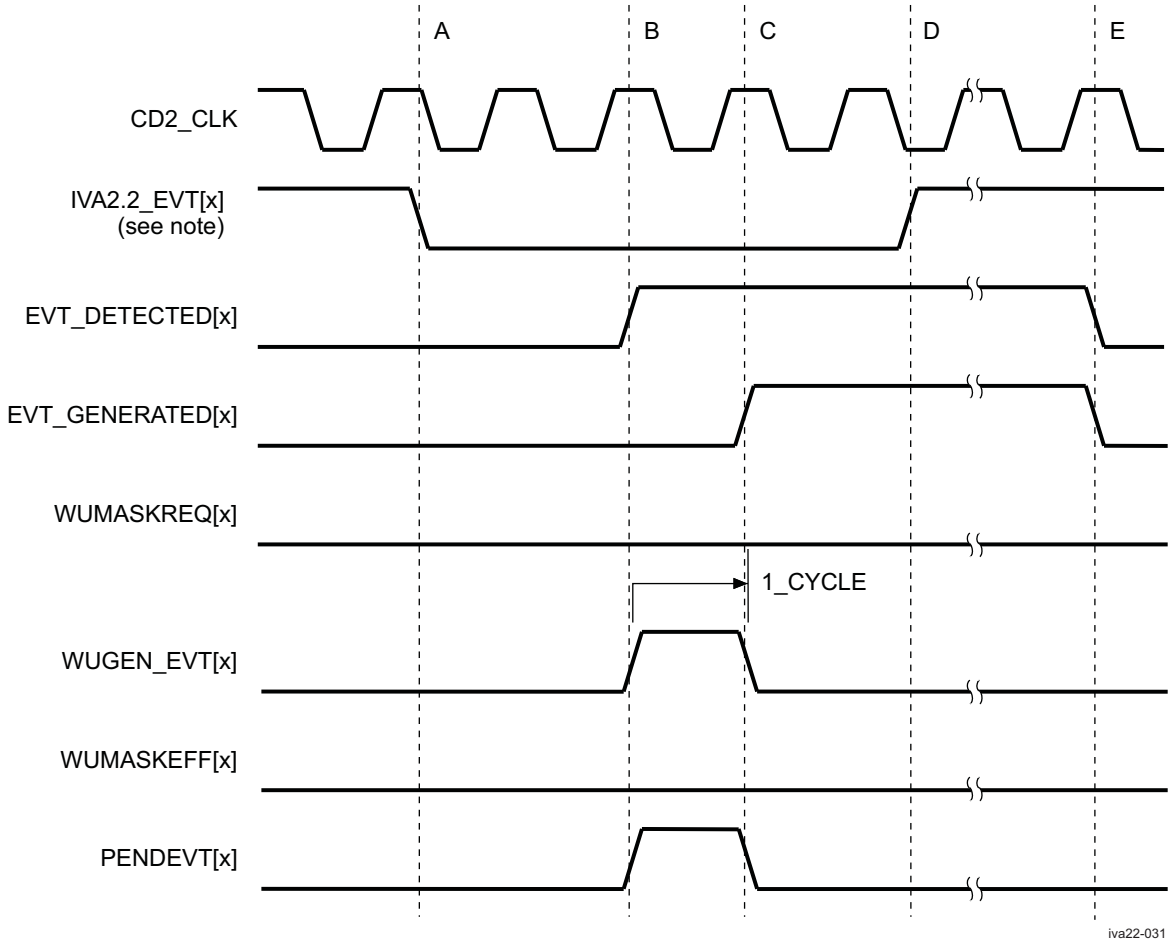
To manage interrupts, DMA requests, and slave port accesses from external modules, the WUGEN module uses several sets of user-programmable registers:

- The WUGEN_MEVT0, WUGEN_MEVT1, and WUGEN_MEVT2 registers define individual mask bits for external asynchronous events (interrupts, DMA requests, slave port access). This register is read-only, only by the DSP. To modify the value of the individual mask, write 1 to the associated bits of WUGEN_MEVTCLR0, WUGEN_MEVTCLR1, and WUGEN_MEVTCLR2 (to clear) and WUGEN_MEVTSET0, WUGEN_MEVTSET1, and WUGEN_MEVTSET2 (to set).
- The WUGEN_PENDEVT0, WUGEN_PENDEVT1, and WUGEN_PENDEVT2 registers are read-only registers that track which external events are pending in the WUGEN module and are not yet generated to the DSP megamodule interrupt controller (INTC) or to the EDMA. If an event is masked, this information is stable until the corresponding mask bit is cleared. If an event is unmasked, this information may not be stable, and therefore corresponds to a transitive period of processing the event in the WUGEN module.

14.3.10.1.1 Event Generation

Figure 14-31 shows event-generation steps in the WUGEN.

Figure 14-31. WUGEN Event Generation



Note: IVA2.2_EVT[x] refers to either IVA2.2nIRQ[x] or IVA2.2_nDMAREQ[x].

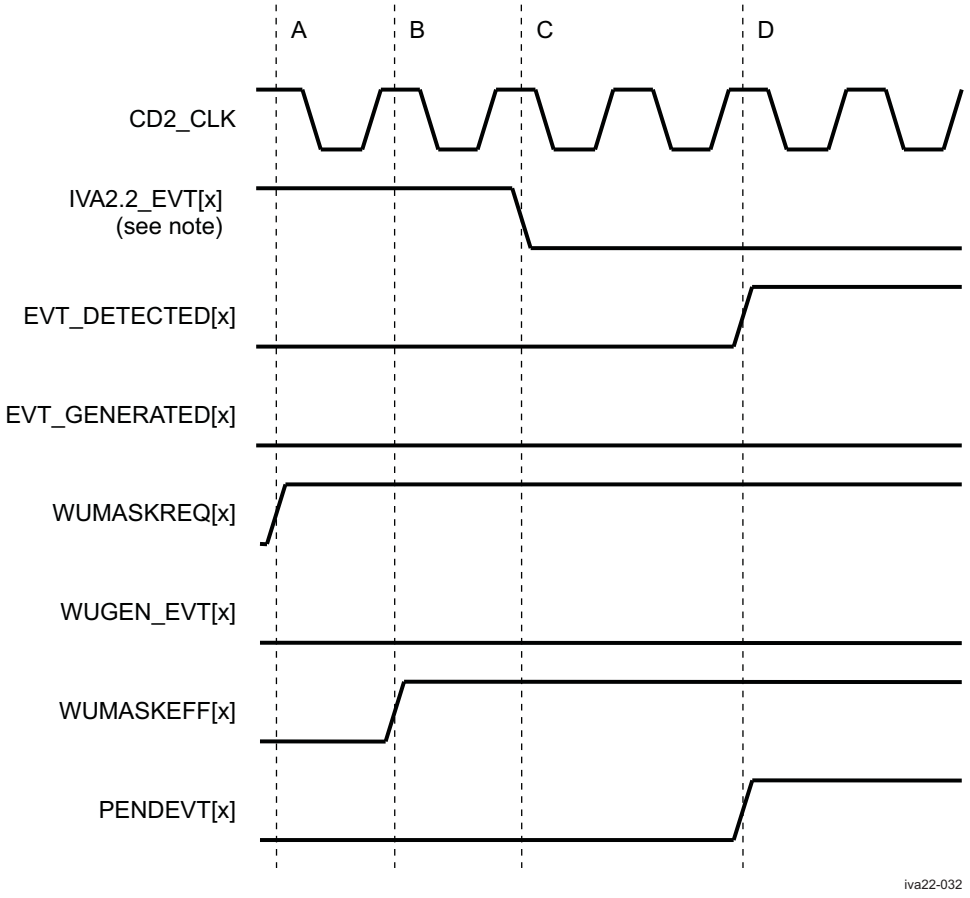
A. An asynchronous event is asserted.
B. The event is resynchronized and detected. Because the event is not masked, it is propagated to DSP megamodule/EDMA.
C. The event-pending flag is cleared because the event was generated. The event to DSP megamodule/EDMA is deasserted after one WUGEN clock cycle.
D. After some time, the source of the event is released (probably because it cleared in the corresponding interrupt status register).
E. The release of the event is detected and clears the internal EVT_GENERATED flag for that interrupt.

14.3.10.1.2 Individual Event Masking

To mask individual events, write 1 in the WUGEN_MEVTSET0, WUGEN_MEVTSET1, or WUGEN_MEVTSET2 register of the WUGEN module. These registers are used to mask interrupts and DMA requests.

Figure 14-32 shows the mechanism of event masking in the WUGEN module.

Figure 14-32. WUGEN Event Masking



Note: IVA2.2_EVT[x] refers to either IVA2.2_nIRQ[x] or IVA2.2_nDMAREQ[x].

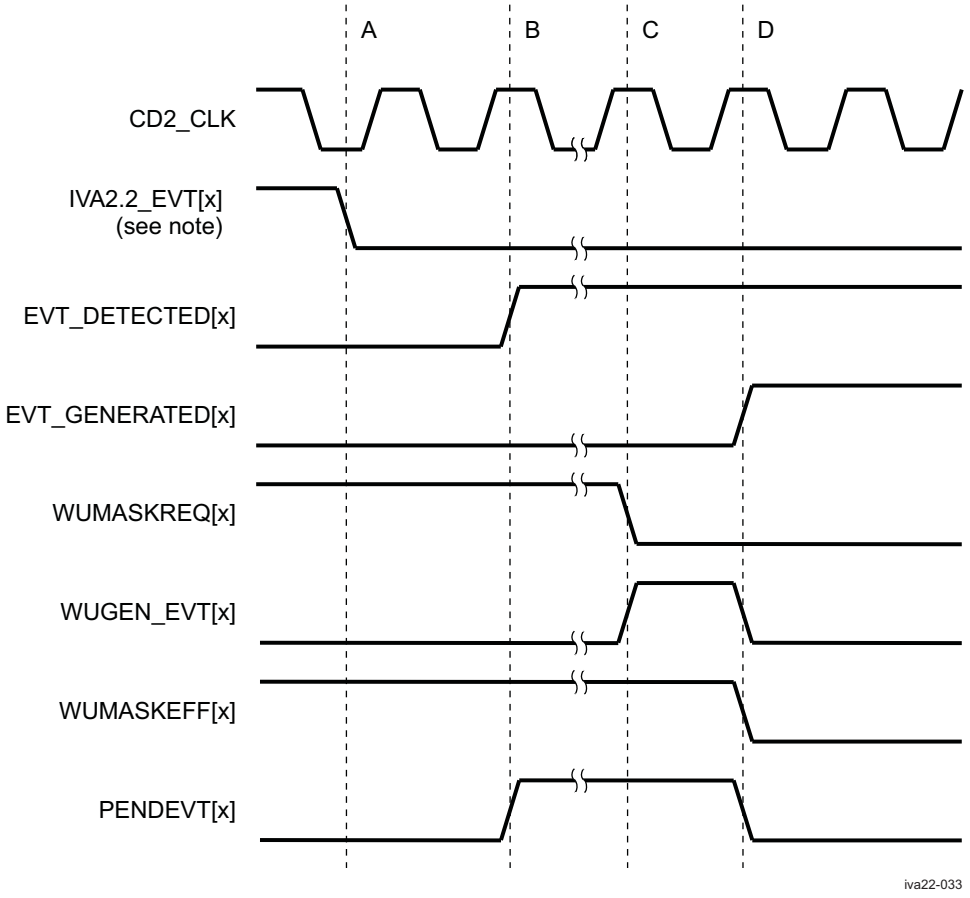
A. The user sets the corresponding bit in the mask. Event generation is disabled.
B. Event mask completion is reflected in the IVA2.2.WUGEN_MEVT0 to IVA2.2.WUGEN_MEVT2 registers (where i = 0 to 2) for reads.
C. An asynchronous event is asserted.
D. The event is resynchronized and detected. Because the event is masked, it is not propagated to DSP megamodule/EDMA. The event-pending flag is kept active (sticky) until the mask is removed.

14.3.10.1.3 Individual Event Mask Clear

To unmask individual events, write 1 in the WUGEN_MEVTCLR0, WUGEN_MEVTCLR1, or WUGEN_MEVTCLR2 register of the WUGEN module.

Figure 14-33 shows the event-mask-clear mechanism in the WUGEN module.

Figure 14-33. WUGEN Event Mask Clear



Note: IVA2.2_EVT[x] refers to either IVA2.2_nIRQ[x] or IVA2.2_nDMAREQ[x].

A. An asynchronous event is asserted.
B. The event is resynchronized and detected. Because the event is masked, it is not propagated to DSP megamodule/EDMA. The event-pending flag is kept active (sticky) until the mask is removed.
C. After some time, clear the corresponding bit in the WUGEN mask. The event is automatically generated to DSP megamodule if the event is still seen as active. If the event is not seen as active, nothing happens.
D. In both cases, the event-pending flag is cleared.

For more information about interrupts and the EDMA programming model, see Section 14.4.7, *Interrupt Management*, and Section 14.4.4.1, *Transfers From/to Device Memories/Peripherals (EDMA)*.

14.3.10.2 Idle Handshake

After reset, the WUGEN waits for a request. If the user executes the DSP IDLE instruction to put the DSP in standby state, the SYSC initiates a clock-off handshake with the WUGEN.

14.3.11 SYSC Module

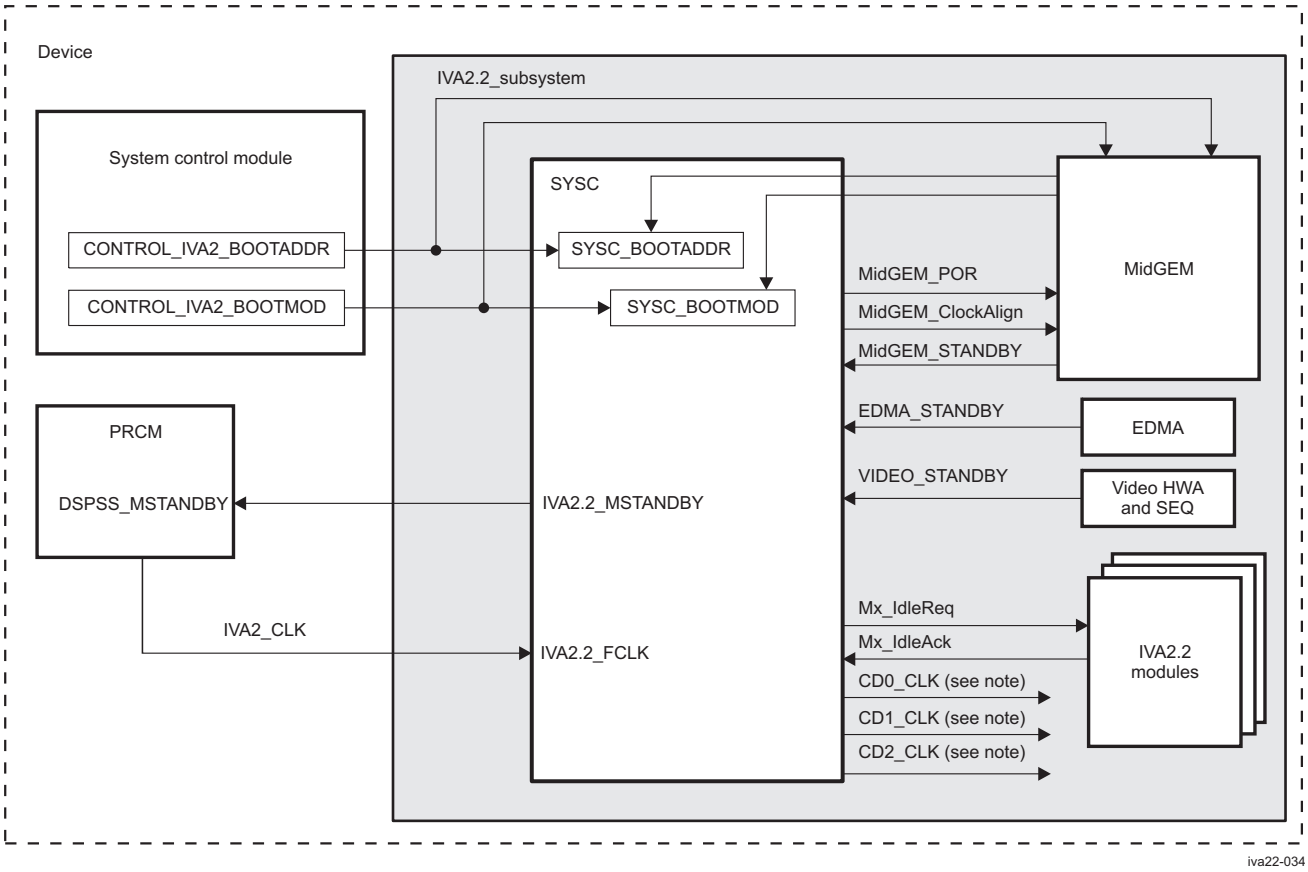
The SYSC module of the IVA2.2 subsystem controls the following functions:

- Generation of the divided clocks to all components of the IVA2.2 subsystem
- Synchronization of the IVA2.2 divided clocks and the DSP megamodule internal-divided clocks
- PRCM power handshaking

- Sequencing of clock-to-off transition for the IVA2.2 modules
- Reset input resynchronization of the active-to-nonactive transition to the CD2_CLK clock
- IVA2.2 boot configuration and its access from the DSP

Figure 14-34 is the block diagram of the SYSC in the IVA2.2 subsystem.

Figure 14-34. SYSC Block Diagram



Note: For more information, see Section 14.2.1, *Clocking, Reset, and Power-Management Scheme*.

14.3.11.1 Divided Clock Generation

The SYSC module generates the divided clocks used by the modules in the IVA2.2 subsystem. The SYSC generates three clocks:

- CD0_CLK
- CD1_CLK
- CD2_CLK

Based on the IVA2.2_FCLK clock, these three clocks are configured from the PRCM. For details, see Section 14.2.1, *Clocking, Reset, and Power-Management Scheme.*

14.3.11.2 Clock Management, Power-Down, and Wake-Up

The SYSC ensures correct generation of the clocks described in Figure 14-34. The SYSC also allows management of the clock gating of the EDMA, video accelerator, sequencer, and DSP megamodule if the user wants to lead some dynamic power aspects.

The SYSC module controls the transition to IVA2.2 subsystem standby state and standby signal generation (IVA2.2_MSTANDBY signal in Figure 14-34) to the PCRM. Using the signals DSP_MEGACELL_STANDBY, EDMA_STANDBY, VIDEO_STANDBY, Mx_IdleReq, and Mx_IdleAck, the SYSC manages and controls the correct power-down transition of the IVA2.2 subsystem and its submodules to ensure that the IVA2.2 internal clocks can be cut. For information about the IVA2.2 power-down transition and its programming model, see Section 14.4.9.3, *Power-Down and Wake-Up Management*.

External events can also occur at the IVA2.2 boundary (access to IVA2.2 using the slave access port or external nonmasked events), requiring the restart of the IVA2.2 clocks (in case of IVA2.2 standby state). The WUGEN module controls the asynchronous generation of a wake-up signal to the PRCM; this signal restarts IVA2.2 PLL clock generation, allowing the SYSC to regenerate the IVA2.2 internal clocks.

14.3.11.3 Boot Configuration

The IVA_SYSC.SYSC_BOOTADDR and IVA_SYSC.SYSC_BOOTMOD registers are the boot-time configuration registers. These read-only registers are accessible only by the DSP, and their values are determined when the IVA2.2 subsystem is released from reset by the PRCM.

The values of these registers are driven externally by two system control module registers, SYSC_GENERAL1.CONTROL_IVA2_BOOTADDR and SYSC_GENERAL1.CONTROL_IVA2_BOOTMOD, which are read-write accessible by the MPU subsystem for MPU-driven IVA2.2 boot sequence and/or by the IVA2.2 DSP for autonomous boot. For more information, see Section 14.4.1*, IVA2.2 Boot*.

Note: If the values of the SYSC_GENERAL1.CONTROL_ IVA2_BOOTADDR and SYSC_GENERAL1.CONTROL_IVA2_BOOTMOD registers change, the new IVA2.2.SYSC_BOOTADDR and IVA2.2.SYSC_BOOTMOD register values are not updated until the next reset.

For more information, see Section 14.4.1, *IVA2.2 Boot*.

14.3.11.4 Interconnect Optimization

The IVA2.2.SYSC_LICFG0 and IVA2.2.SYSC_LICFG1 registers are local interconnect configuration registers. They are read-write accessible by the DSP only. Setting these registers enables or disables interconnect optimization.

14.3.11.5 Video Accelerator/Sequencer SYSC

The video and sequencer system control module (called VIDEOSYSC in Section 14.5.10, *Video System Controller Registers*) controls module reset, power management, and interrupt handling for the video accelerators/sequencer modules.

14.3.11.5.1 Reset

The reset module resynchronizes asynchronous reset for the following modules:

- iME
- iLF
- SL2IF
- Video and sequencer interconnect
- Sequencer

For details, see Section 14.2, *IVA2.2 Subsystem Integration*.

14.3.11.5.2 Power Management

The video accelerator/sequencer SYSC controls clock gating for the iLF, iME, and sequencer modules and controls clock division for the sequencer module.

The VIDEOSYSC_CLKCTL register is used to stop or restart the module clocks. A write of 1 requests the module logic to go idle and to stop the clock. A write of 0 requests a restart of logic and clock. When a new request or command occurs, a module automatically exits the IDLE state and the clock starts. Clock gating can be controlled for the following modules:

- SL2 memory interface (controlled by the VIDEOSYSC_CLKCTL[5] SL2IFCLKEN bit)
- Sequencer memory and slave port (controlled by the VIDEOSYSC_CLKCTL[4] SEQMEMCLKEN bit)
- iME module (controlled by the VIDEOSYSC_CLKCTL[1] IMECLKEN bit)
- iLF module (controlled by the VIDEOSYSC_CLKCTL[0] ILFCLKEN bit)

For details, see Section 14.5.10 *Video System Controller Registers*.

The VIDEOSYSC_CLKDIV[1:0] SEQCLKDIV bit field is used to set a clock divider for the sequencer. A divider of 1, 2, 3, or 4 is available. For details, see Section 14.5.10, *Video System Controller Registers*.

14.3.11.5.3 Interrupt Handler

The video accelerator/sequencer SYSC has a single interrupt output, VIDEO_INT, which is an active low-level interrupt connected to the DSP megamodule. Once asserted, it can be cleared explicitly only by software. It can be enabled or disabled by software. It is asserted when one of the enabled event lines gets asserted. Table 14-84 lists these interrupts.

Table 14-84. LSYS Input Interrupts

Nb	Name	Description
7	Reserved	Reserved
6	SEQ_MBX	Sequencer mailbox IRQ
5	DMA_ERROR	DMA error IRQ
4	HOST_ERROR	HOST error IRQ
3	Reserved	Reserved
2	Reserved	Reserved
1	iLF	iLF IRQ
0	iME	iME IRQ

Four registers are defined for IRQ operation:

- The IVA.VIDEOSYSC_IRQSTATE register tracks input events. Each event is associated with a bit in this register. When the bit is set after an active pulse on the associated event line, this bit is kept active (sticky) until the user explicitly writes 1 to that bit.
- The IVA.VIDEOSYSC_IRQMASK register controls the sensitivity of the interrupt line to input events. Each event is associated with a bit in this register. When the user writes 1 to a bit, the associated active event is not allowed to trigger an interrupt. When the user writes 0, the associated active event triggers an interrupt.
- The IVA.VIDEOSYSC_IRQSET register is used to set the interrupt bits (used to test interrupt). Each event is associated with a bit in this register. When the user writes 1 to a bit, the associated event is set in the IVA.VIDEOSYSC_IRQSTATE register. When the user writes 0, there is no effect.
- The IVA.VIDEOSYSC_IRQCLR register is used to clear the interrupt bits in the IVA.VIDEOSYSC_IRQSTATE register. Each event is associated with a bit in this register. When the user writes 1 to a bit, the associated event is cleared in the IVA.VIDEOSYSC_IRQSTATE register. When the user writes 0, there is no effect.

14.3.12 Local Memories

The IVA2.2 subsystem integrates three memory controllers under the control of the DSP megamodule:

- DMC

- PMC
- UMC

Depending on the software configuration of these memory controllers, the IVA2.2 subsystem local memories (ROM and RAMs) can be used as cache or memory-mapped memories.

Figure 14-35 shows the memory hierarchy of the IVA2.2 subsystem.

Figure 14-35. IVA2.2 Local Memories Hierarchy

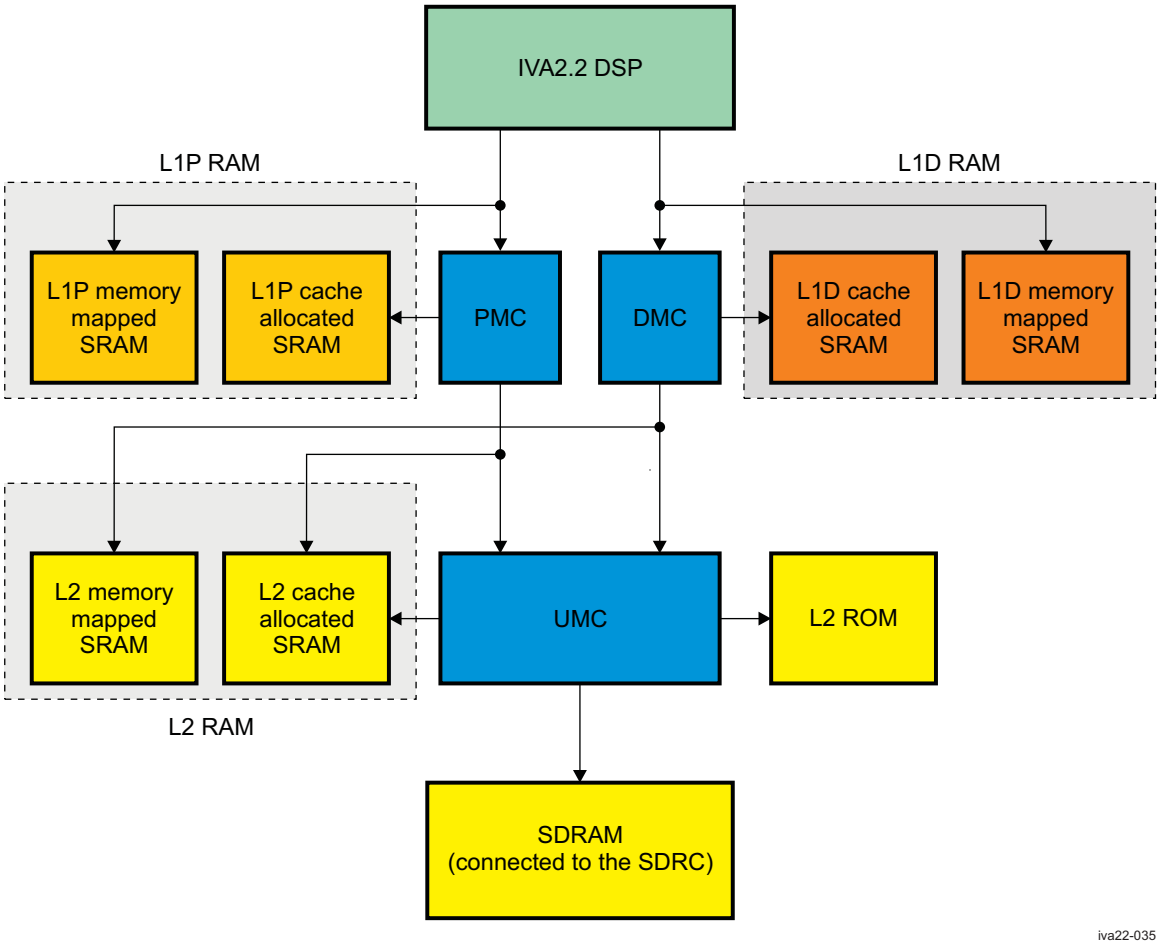


Table 14-85 lists the IVA2.2 DSP megamodule cache controller features.

Table 14-85. IVA2.2 DSP Megamodule Cache Controller Features

Cache	L1 Program Cache	L1 Data Cache	L2 Unified Cache
SIZE	Programmable to 0KB, 4KB, 8KB, 16KB, or 32KB	Programmable to 0KB, 4KB, 8KB, 16KB, or 32KB	Programmable to 0KB, 32KB, or 64KB
ASSOCIATIVITY	Direct-mapped	Two-way set associative	Four-way set associative
SRAM MODE [memory-mapped SRAM]	Programmable to 0KB, 16KB, 24KB, 28KB, 32KB [32KB - cache size]	Programmable to 48KB, 64KB, 72KB, 76KB, 80KB [80KB - cache size]	Programmable to 32KB, 64KB, or 96KB [96KB - cache size]
WRITE BUFFER	NR	Yes	Yes
HIT UNDER MISS	Yes	Yes	No
MISS PIPELINING	Yes	Yes	No
REPLACEMENT POLICY	NR	True LRU	True LRU (pseudo-LRU)
CACHEABILITY	Always cached	Programmable	Programmable
BUFFERABILITY	NR	Always buffered	Always buffered

Table 14-85. IVA2.2 DSP Megamodule Cache Controller Features (continued)

Cache	L1 Program Cache	L1 Data Cache	L2 Unified Cache
WRITE POLICY	NR	Always write-back	Always write-back
ALLOCATION POLICY	Read-allocate	Read-allocate	Read-write-allocate
CRITICAL WORD FIRST	No	No	No
CRITICAL LINE FIRST	NR	NR	Yes
ADVANCE FETCH	Yes, because of DSP advanced fetch-and-miss pipelining	NR	NR

14.3.12.1 ROM Overview

The IVA2.2 subsystem contains 16KB L2 ROM. The content of this ROM includes boot code.

For information about boot code and the IVA2.2 boot mechanism, see Section 14.4.1, *IVA2.2 Boot*.

When the L1P SRAM is configured to be inactive (that is, fully memory-mapped SRAM), DSP program fetch accesses to the L2 ROM are realized directly, and thus suffer the L2 latency.

14.3.12.2 RAM Overview

- 32KB L1 program RAM (L1P)
- 80KB L1 data RAM (L1D)
- 96KB L2 unified RAM (L2)

Note: The maximum cache size for the PMC is 32KB. L1P memory-mapped RAM can be programmed to allocate 0KB (full-cache), 16KB, 24KB, 28KB, or 32KB (no cache, default) using the PMC registers. The remaining memory is allocated to the cache controller.

The maximum cache size for the DMC is the default 32K bytes. L1D memory-mapped RAM can be programmed to allocate 48KB (full-cache), 64KB, 72KB, 76KB, or 80KB (no cache, default). The remaining memory is allocated to the cache controller (DMC).

L1P and L1D RAM operate at the DSP frequency (CD0 clock domain).

L2 RAM operates at half the DSP frequency (CD1 clock domain).

In the IVA2.2 subsystem, L2 can be configured so that the memory-mapped RAM allocates 32KB, 64KB, or 96KB (no cache) of L2 memory. The remaining memory is allocated to the cache controller

By default, L2 memory is used as 96-KB local memory-mapped RAM. However, L2 RAM is typically used as 64KB allocated to the cache RAM. This is programmable in the DSP megamodule UMC.

The L2 memory-mapped SRAM is shared by the DSP megamodule and the SL2 interface. An arbitration mechanism is implemented for concurrent access, with the DSP megamodule having priority. These last 32KB cannot be used as cache RAM, but only as memory-mapped RAM. Because of address restrictions, the iME and iLF modules can access only the last 32KB of shared L2.

For information about the software programming model for cache management of the local RAM memories of the IVA2.2 subsystem, see Section 14.4.3, *Cache Management*.

14.3.13 Local Interconnect Network

The local interconnect network is the IVA2.2 internal logic that allows internal communication between the modules of the subsystem (for example, the DSP megamodule can communicate with the EDMA or with the WUGEN module). Communication with the rest of the system is through the MMU as master port (accesses from the IVA2.2 subsystem to the L3 interconnect), and through a slave port for accesses from the L3 interconnect (accesses from the L3 interconnect to the IVA2.2 subsystem).

For information about global memory mapping of the IVA2.2 subsystem (DSP CPU view or MPU view through the L3 interconnect), see Chapter 2, *Memory Mapping*.

14.3.13.1 Endianness

The IVA2.2 local interconnect network does not support the big-endian convention or any type of endianness conversion. Only little-endian is supported by the local interconnect network.

14.3.14 Error Reporting

Several mechanisms are available in the IVA2.2 subsystem to report errors at different levels.

The INTC of the DSP megamodule allows reporting of dropped interrupts, through the INTERR signal.

L3 out-of-band errors are also reported through the external L3 interrupt signal. For details about interrupt mapping, see Table 14-3. For information about L3 interconnect error reporting, see Chapter 5, *Interconnect*.

The IDMA and EDMA have their own error-reporting mechanism. Error-status registers and error interrupts inform the user of problems during IVA2.2 internal operation (data error, bus activity error, etc.).

For information about error management and the register set that allows error tracing, see Section 14.4.10, *Error Identification Process*.

14.4 IVA2.2 Subsystem Basic Programming Model

14.4.1 IVA2.2 Boot

The boot-time configuration registers are IVA_SYSC.SYSC_BOOTADDR and IVA_SYSC.SYSC_BOOTMOD. These read-only registers are accessible only by the DSP CPU, and their values are determined when the IVA2.2 is released from reset, using the CONTROL.CONTROL_IVA2_BOOTADDR and CONTROL.CONTROL_IVA2_ BOOTMOD registers of the system control module. For more information, see Chapter 7, *System Control Module*. A change to those registers is not seen (not sampled) by the IVA2.2 until the next reset.

The IVA2.2 boots two ways:

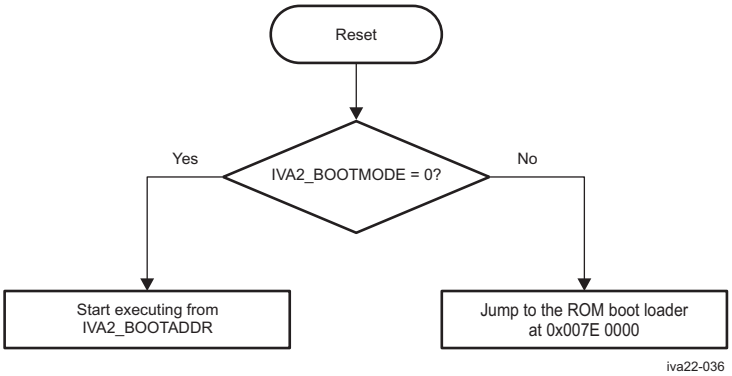
- Boot under MPU control, described in Section 14.4.1.2.1, *Boot Under MPU Control*.
- Autonomous boot, described in Section 14.4.1.2.2, *Autonomous Boot*.

A boot under MPU control is typically used after the device cold reset as a first-time configuration of the IVA2.2 subsystem. Subsequent boots of the IVA2.2, resulting from a wake-up transition from OFF state, for example, use the autonomous boot.

14.4.1.1 IVA2.2 Boot Configuration

Figure 14-36 shows the boot sequence followed by IVA2 on release from reset.

Figure 14-36. IVA2 Boot Mode Configuration



When the IVA2.2 subsystem is released from reset and CONTROL.CONTROL_IVA2_BOOTMOD[3:0] BootMode equals 0x0, the first fetch address of the C64x equals the address defined in the CONTROL.CONTROL_IVA2_BOOTADDR[31:10] BOOTLOADADDR bit field. This address can be aligned on any 1-K byte boundary, in ROM, in IVA2.2 local RAM, in on-chip memory (OCM-RAM), or directly in external memory (for example, SDRAM through the SDRC).

When the IVA2.2 subsystem is released from reset and CONTROL.CONTROL_IVA2_BOOTMOD[3:0] BOOTMODE is different from 0x0, the first fetch address of the C64x is fixed to 0x007E0000 (in IVA2.2 local ROM). ROM code boot loader configuration is detailed in Table 14-86.

Table 14-86. Boot Loader Configuration

Value of SYSC_IVA2_BOOTMOD	Description
0x01	IDLE Boot: Boot loader configures the PDCCMD and then executes the IDLE instruction. See Section 14.4.1.1.1.
0x02	Wait in self-loop mode: Boot loader puts IVA2.2 in a self-loop. See Section 14.4.1.1.2.
0x03	Cache Config Mode: Boot loader configures the L1P, L1D, L2 caches and the MAR register. See Section 14.4.1.1.3.
0x04	User defined bootstrap mode: Boot loader copies the boot strap into L2 memory and branches to it. See Section 14.4.1.1.4.
0x05	Reserved.

14.4.1.1.1 IDLE Boot Mode

In this boot mode the PDCCMD register is configured before the IDLE instruction is executed. When the IDLE instruction is executed, if there are no pending requests from either the DMA or an interrupt, IVA goes into sleep mode.

The PDCCMD register is programmed as shown in Table 14-87.

Table 14-87. PDCCMD Programmed Value in IDLE Boot Mode

BitField	Value	Description
GEMPD	1	Sleep mode. Power-down DSP CPU and megamodule when DSP CPU enters IDLE state.
EMCMEM	11	Internal RAMs sleep with retention.
EMCLOG	11	Maximum dynamic clock gating of module regions, with potential : latencies/penalties for wake when megamodule is active and Static clock gating to the EMC when megamodule is in standby.
UMCMEM	11	Sleep mode 3. Internal RAMs sleep with retention, L2 defined at chip-level.
UMCLOG	11	Maximum dynamic clock gating of module regions, with potential latencies/penalties for wake when megamodule is active and Static clock gating to the UMC when megamodule is in standby.
DMCMEM	11	Sleep mode 3. Internal RAMs sleep with retention, L1D defined at chip-level.
DMCLOG	11	Maximum dynamic clock gating of module regions, with potential latencies/penalties for wake when megamodule is active and Static clock gating to the DMC when megamodule is in standby
PMCMEM	11	Sleep mode 3. Internal RAMs sleep with retention, L1P defined at chip-level.
PMCLOG	11	Maximum dynamic clock gating of module regions, with potential latencies/penalties for wake when megamodule is active and Static clock gating to the PMC when megamodule is in standby

14.4.1.1.2 Wait in Self Loop Mode

In this mode boot loader puts the IVA2 in a self-loop. The MPU then has the option to download the bootstrap code from the MPU side directly into IVA2 internal memory through Host Port Interface (L3 slave interface). MPU then sets up CONTROL.CONTROL_IVA2_BOOTMOD[3:0] BOOTMODE to '0' and CONTROL.CONTROL_IVA2_BOOTADDR[31:10] BOOTLOADADDR to the internal memory address where it copied the bootstrap, IVA2.2 is then released from reset. Upon completion of the reset sequence IVA2.2 jumps to the bootstrap code loaded by the user in its internal memory and starts executing it.

14.4.1.1.3 Default Config Cache Mode

In this mode the boot loader configures the L1P, L1D, L2 caches and the MAR registers of DSP megamodule. After performing this configuration the boot loader branches to an external memory address. The values with which these settings are done and the external memory address to which the boot loader jumps is specified using a header.

Table 14-88 specifies the format of the header.

Table 14-88. Header Format Used in Defautl Config Cache Mode

Offset from Base Address of Header (in bytes)	Fields Description
0x00	Value of L1PCFG register to be loaded.
0x04	Value of L1DCFG register to be loaded.
0x08	Value of L2CFG register to be loaded.
0x0C	1 -> Set PC bit of all MARi registers to 1 0 -> No action will be taken as by default PC bit of all MARi registers is set to 0
0x10	Address of external memory to which boot loader should branch to.

The address of this table (header) should be specified in the CONTROL.CONTROL_IVA2_BOOTADDR[31:10] BOOTLOADADDR register before IVA2.2 is released from reset.

Setting of the L1PCFG, L1DCFG, and L2CFG is done as follows:

- The desired cache mode is written to the L1PCFG, L1DCFG, and L2CFG.
- L1PCFG, L1DCFG, and L2CFG are read back. This stalls the CPU until the mode change completes

Setting of the MAR registers is done as follows:

- The desired value (in our case 1) is written to the MAR
- The final value written to the MAR address is read back. This stalls the CPU until all the MAR writes are completed.

14.4.1.1.4 User Defined Bootstrap Mode

In this mode the boot loader downloads a user bootstrap code, which is kept in the external memory, into the L2 memory of IVA2.2. After transferring, the boot loader branches to the user bootstrap code. Please note that all sections of the user bootstrap code should fit into L2 memory and no section of the bootstrap code should be mapped to either L1P or L1D memory.

Various parameters that the boot loader needs to transfer the bootstrap code are provided using a user bootstrap header present in the external memory. The CONTROL.CONTROL_IVA2_BOOTADDR[31:10] BOOTLOADADDR is setup with the address of this user bootstrap header before IVA2.2 is released from reset.

Table 14-89 specifies the format of the bootstrap header.

Table 14-89. Header Format Used in User Defined Bootstrap Mode

Offset from Base Address of Header (in bytes)	Fields Description
0x00	Size of the boot strap code in bytes that will be downloaded into internal memory.
0x04	0 -> Use DMA for transferring the bootstrap code 1 -> Use CPU copy for transferring the bootstrap code.
0x08	Value of L2CFG register to be loaded.
0x0C	Absolute address of L2 memory where the boot strap code is to be copied.
0x10	Offset in bytes from the beginning of the bootstrap code where the first executable instruction of the bootstrap is present.
0x14	Absolute address in external memory from where the bootstrap code will be copied.

Note: It is mandatory for the user bootstrap code to be a multiple of 4 words on IVA2.2

14.4.1.2 Example of IVA2.2 Boot

14.4.1.2.1 Boot Under MPU Control

Before waking up the IVA2.2 subsystem, the MPU performs the following sequence (also shown in Figure 14-37):

1. The MPU prepares a translation table hierarchy (TTH) in SDRAM at address <TTH Physical Address>. This TTH must contain at least an address translation for the IVA2.2 MMU physical address range.

   Note: The DSP CPU does not require an address translation for the TTH, as the TTH is under MPU control only. The IVA2.2 DSP CPU is responsible only for saving and restoring an IVA2.2 MMU context that has been programmed by the MPU.

2. The MPU writes a bootstrap sequence in SDRAM at address <BootLoader Physical Address>. This

sequence is executable by the DSP CPU and contains only relative address references, so that it is relocatable without code modification. This sequence must contain at least (in order):

a. Set the size of the bootstrap code.
b. Program the IVA2.2 MMU using physical addresses, as described in the following steps.

Note: The device has two instances of MMU: camera MMU or MMU1 (used for the camera subsystem) and IVA2.2 MMU or MMU2 (used for the IVA2.2 subsystem). These two instances are programmed by two identical sets of configuration registers. For more information, see Chapter 8, *Memory Management Units*.

3. Programming of the MMU2 is functionally restricted to setting the MMU2.MMU_CNTL[1] MMUENABLE bit, and the TLB misses are served by the MPU-dedicated interrupt service routine (ISR). However, MMU table-walking logic is preferred because the IVA2.2 must be autonomous after the preliminary settings.

   Moreover, it is critical that the duration of the MMU save-and-restore process use a minimum number of cycles, so that address translation for the MMU configuration registers uses a TLB locked entry. For those reasons, it is recommended that the bootstrap follow the MMU initialization sequence:

   a. Write <TTH physical address>[31:7] to the MMU2.MMU_TTB[31:17] TTBADDRESS bit field.
   b. Lock the address translation for the MMU configuration registers.
      i. Write 0x0 to the MMU2.MMU_LOCK[8:4] CURRENTVICTIM bit field.
      ii. Write <MMU virtual address>[31:12] to the MMU2.MMU_CAM[31:12] VATAG bit field.
      iii. Write 1 to the MMU2.MMU_CAM[2] V bit.
      iv. Write 0x2 to the MMU2.MMU_CAM[1:0] PAGESIZE bit field.
      v. Write <MMU physical address>[31:12] to the MMU2.MMU_RAM[31:12] PHYSICALADDRESS bit field.
      vi. Write 1 to the MMU2.MMU_LD_TLB[0] LDTLBITEM bit.
      vii. Write 0x1 to the MMU2.MMU_LOCK[8:4] CURRENTVICTIM bit field.
      viii. Write 0x1 to the MMU2.MMU_LOCK[14:10] BASEVALUE bit field.
   c. Write 1 to the MMU2.MMU_CNTL[2] TWLENABLE bit.
   d. Write 1 to the MMU2.MMU_CNTL[1] MMUENABLE bit.
   e. Read back the MMU2.MMU_CNTL register to ensure write completion.

   Note: This read does not generate a TLB miss, because the address translation for MMU configuration registers is locked in the TLB.

4. The MPU can lock some other TLB entries, define some power-management MMU settings, and/or enable some interrupts.
5. The MPU correctly programs the associated L3 firewall, ensuring that the DSP CPU has read access to the memory area containing the IVA2.2 bootstrap sequence.
6. The MPU programs the associated L3 firewall, ensuring that the DSP CPU has read and write access to the configuration registers of the MMU2.
7. The MPU writes <bootloader physical address> to the CONTROL.CONTROL_IVA2_BOOTADDR[31:10] BOOTADDR bit field and configures the correct CONTROL.CONTROL_IVA2_BOOTMOD[3:0] BOOTMOD bit field nonzero value.

   Note: For a detailed description of the system boot registers, see Chapter 7, *System Control Module*.

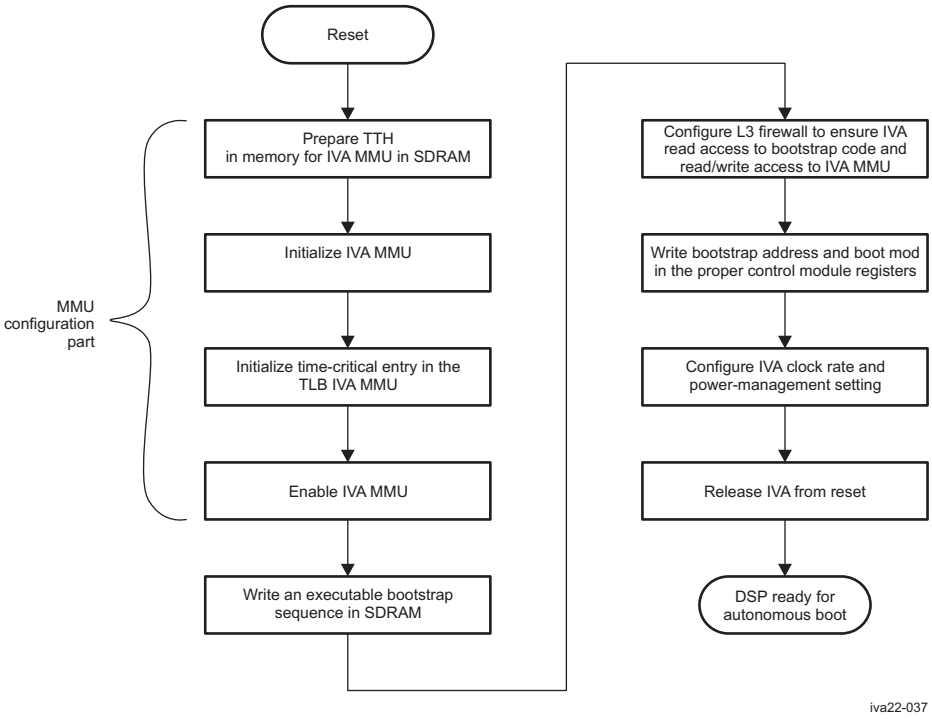
8. The MPU configures the IVA2.2 clock rate and IVA2.2-related power-management settings in the PRCM.

   After initialization completes, the MPU allows the IVA2.2 to boot:

   a. The MPU releases the IVA2.2 from the OFF state by programming the PRCM.
   b. The MPU sets the clocks back to the IVA2.2.
   c. The MPU releases the IVA2.2 from reset.

Note: Following Steps 1 through 8, the IVA2.2 boot sequence is identical to the autonomous boot sequence. For information about the IVA2.2 boot sequence, see Section 14.4.1, *IVA2.2 Boot*.

Figure 14-37. IVA2 Boot Basic Programming Model



14.4.1.2.2 Autonomous Boot

On an OFF-to-ACTIVE transition (under MPU control or upon interrupt wakeup), the IVA2.2 subsystem is released from reset.

After hardware configuration of values for DSP megamodule generic parameters, the IVA2.2 starts fetching from the address 0x00000000 in ROM. The IVA2.2 must follow the (nonexhaustive) boot process:

1. Configure memory protection:
   a. Specify cache RAMs versus IDMA and DMA accesses.
   b. Define accessible L2 space (static shared L2 with the MPU/LCD).
2. Load the bootstrap code:
   a. Read the IVA_SYSC.SYSC_BOOTADDR[31:12] BOOTLOADADDR bit field.
   b. Read the number of bootstrap words <NbOfWords> to transfer in the first word at the address pointed to by BOOTLOADADDR.
   c. Transfer as many words as defined in <NbOfWords>.
3. Execute the bootstrap code.

Note: This prepares the MMU for address translation of external accesses. In the case of a boot upon interrupt wakeup, the MMU is restored to its exact context before the IVA2.2 is shut off, if this context was saved correctly.

Only from this point can the IVA2.2 subsystem work with virtual addresses.

14.4.2 Sequencer Boot/Reset

After the reset input signal is applied to the video accelerator/sequencer, all modules are operational except the sequencer CPU, which is maintained under reset so that the host can initialize some configuration registers and upload some boot code in the ITCM and DTCM memories of the sequencer.

The sequencer reset is connected on IVA2_RST3, and this reset is controlled by the PRCM. The DSP megamodule or the MPU can release the IVA2_RST3 by clearing the PRCM.RM_RSTCTRL_IVA2[2] RST3_IVA2 bit.

When the sequencer reset (IVA2_RST3) is released, the sequencer starts fetching instructions from ITCM memory, which is initialized by the DSP megamodule or the MPU before the sequencer reset is released. Thus, a classic boot/reset sequence follows the sequence (only the sequencer is under reset; other modules like iME and iLF are already out of reset):

1. The DSP initializes the ITCM sequencer memory with sequencer code.
2. The DSP initializes the DTCM sequencer memory with sequencer data.
3. The DSP sets the clock divider for the sequencer module in the IVA.VIDEOSYSC_CLKDIV register.
4. The DSP releases the sequencer from reset by clearing the PRCM.RM_RSTCTRL_IVA2[2] RST3_IVA2 bit.

The sequencer is autonomous (no longer under DSP control).

The DSP and the sequencer processor communicate through a message box; two interrupts are provided for this purpose.

Note: For more information about sequencer boot/reset, see *ARM968E-S Technical Reference Manual ARM*.

14.4.3 Cache Management

The IVA2.2 subsystem has a 2-level cache-based architecture. Level 1 data memory/cache (L1D) consists of an 80-KB memory space dedicated to data. L1D memory can be configured as mapped memory, cache, or a combination of the two. The level 1 program memory/cache (L1P) consists of a 32-KB memory space dedicated to program instructions. L1P memory can be configured as mapped memory, cache, or a combination of the two. Level 2 memory/cache (L2) consists of a 96-KB memory space shared by program and data space. L2 memory can be configured as mapped memory, cache, or a combination of the two.

Configuration of the allocation of the L1D, L1P, and L2 memories to mapped memory and/or cache is described in Section 14.4.3.1, *Cache-Size Configuration*.

The virtual address space is split into contiguous chunks to configure which parts of the memory are cacheable and not cacheable. Configuration of cacheability is described in Section 14.4.3.3, *Cacheability Settings*.

14.4.3.1 Cache-Size Configuration

Cache-size configuration is controlled by a set of registers: IVA_XMC.L1PCFG, IVA_XMC.L1DCFG, and IVA_XMC.L2CFG, that correspond to the L1P, L1D, and L2 memories, respectively.

The IVA_XMC.L1PCFG register allows the selection of the size of the L1P cache. The user selects the size of the L1P cache by writing the requested mode to the L1PCFG[2:0] L1PMODE bit field.

Table 14-90 lists the valid settings of L1PMODE.

Table 14-90. Cache Size Specified by L1PMODE

L1PCFG[2:0] L1PMODE	Amount of L1P
Setting	Cache
000b	0 KB (default)
001b	4KB

Table 14-90. Cache Size Specified by L1PMODE (continued)

L1PCFG[2:0] L1PMODE	Amount of L1P
010b	8KB
011b	16KB
100b	32KB
101b	Not used
110b	Not used
111b	Maximum cache (maps to 32KB)

The IVA_XMC.L1DCFG register allows the selection of the size of the L1D cache. The user selects the size of the L1D cache by writing the requested mode to the L1DCFG[2:0] L1DMODE bit field.

Table 14-91 lists the valid settings of L1DMODE.

Table 14-91. Cache Size Specified by L1DMODE

L1DCFG[2:0] L1DMODE	Amount of L1D
Setting	Cache
000b	0KB (default)
001b	4KB
010b	8KB
011b	16KB
100b	32KB
101b	Not used
110b	Not used
111b	Maximum cache (maps to 32KB)

In the same way, the L2CFG[2:0] L2MODE bit field controls the size of the L2 cache. The user sets the amount of L2 memory mapped as L2 cache by writing the desired cache mode to this bit field.

Table 14-92 shows the valid settings for L2MODE.

Table 14-92. Cache Size Specified by L2MODE

L2CFG[2:0] L2MODE	Amount of L2
Setting	Cache
000b	0KB (default)
001b	32KB
010b	64KB

When programs initiate a cache mode change, the L1D cache must write back and invalidate its current contents without losing data. This ensures that all updated data held in cache is written back, and that no false hits occur because of a change in the interpretation of cache tags. It also ensures that the L1D snoop tag RAM in the UMC stays in sync with L1D.

While the write-back-invalidate is required to ensure correct cache behavior and to ensure that no cached data is lost, it is not sufficient to prevent data loss as a result of portions of L1D RAM becoming cache. To safely change L1D cache modes, applications must adhere to the procedure described in Table 14-93.

Table 14-93. Switching Cache Modes

To switch from	To	The program must perform the following steps:
A mode with no or some cache	A mode with more cache	1: EDMA, IDMA, or copy necessary data out of the affected range of RAM. 2: Write the desired cache mode to the bit field in the configuration cache register (L1DCFG, L1PCFG, and L2CFG). 3: Read back the register to stall the DSP CPU until the mode change completes.

Table 14-93. Switching Cache Modes (continued)

To switch from	To	The program must perform the following steps:
A mode with some cache	A mode with less or no cache	1: Write the desired cache mode to the bit field in the cache configuration register concerned. 2. Read back the register to stall the DSP CPU until the mode change completes.

L1P RAM is typically used as cache RAM, with no local flat RAM (not the default; must be configured by the user).

L1D RAM is typically used as 32KB allocated to the cache RAM and 48KB has local memory-mapped RAM. By default, L2 memory is configured as 0KB allocated to the cache RAM and 96KB as local memory-mapped RAM (see Table 14-94).

Table 14-94. Default Cache Configuration

Memory Type	Memory Size	Default Cache Setting
L1P RAM	32KB	0KB cache
L1D RAM	80KB	0KB cache
L2 RAM	96KB	0KB cache

14.4.3.2 Cache Mode Configuration

The memory cache controllers of the DSP megamodule (L1D, L1P, and L2) offer two additional operating modes: freeze and/or bypass. These modes are set by the L1DCC[2:0] OPER bit field for L1D, the L1PCC[2:0] OPER bit field for L1P, and the L2CFG[4:3] L2CC bit field for L2.

The freeze and bypass modes affect the operation of only the cache section (no impact on the memory-mapped section of the memory) of each memory controller.

This feature allows real-time applications to limit the amount of data evicted from cache controllers, such as interrupt handlers, during various sections of code.

Table 14-95 summarizes the freeze and bypass modes for each cache controller (set through the OPER field in the L1PCC register for the L1P cache controller, the OPER field in the L1DCC register, and the L2CC field of the L2CFG register for the L2 cache controller).

Table 14-95. Cache Mode Configuration

Cache Controller	Operation Mode		
	Normal	Freeze	Bypass
L1P	Cache operates normally. Read hits return data from the cache. Write hits update the cached data for the cache line.	The L1P cache does not allocate new cache lines on read misses, nor does it cause existing cache contents to be marked invalid. Write misses are dropped. The L1P cache responds normally to program-initiated cache controls (invalidate, mode change).	N/A
L1D	Cache operates normally.	The L1D cache does not allocate new cache lines on read misses, nor does it evict existing cache contents. The L1D cache responds normally to program-initiated cache commands (invalidate, write-back-invalidate mode change).	N/A

Table 14-95. Cache Mode Configuration (continued)

Cache Controller	Operation Mode		
	Normal	**Freeze**	**Bypass**
L2	Cache operates normally.	Cache frozen. Hits proceed normally. L2 sends read and write misses directly to external memory, as if the L2 cache is not present. The L2 does not allocate a new cache line while frozen. Lines can be evicted from L2 only during freeze mode by program-initiated cache coherence operations.	Cache disabled, although internal cache state is retained. When in bypass mode, the L2 cache responds to neither reads nor writes. All requests for external addresses are sent externally. L2 does not update its contents in this mode. As with freeze mode, L2 evicts lines from L2 only during bypass mode as a result of program-initiated cache coherence operations.

14.4.3.3 Cacheability Settings

The address space of the IVA2.2 subsystem is split into contiguous regions of 16M bytes each. Each region has a cacheable attribute that defines whether a reference to a location belonging to the associated memory region must be sent directly to the memory (with exact size and occurrence as requested by the CPU) or must induce a cache line refill and potentially an eviction of another cache line.

The UMC contains registers that control whether certain ranges of memory are cacheable, and whether one or more requestors is allowed access to these ranges.

14.4.3.4 Coherence Maintenance

14.4.3.4.1 Memory-Mapped L1P and L1D Coherence

L1D and L1P are never cached, so there is no coherence maintenance for those memories.

14.4.3.4.2 Memory-Mapped L2 Coherence

Coherence is maintained by hardware between L1D cache content and the L2 memory-mapped memory region.

An L2 reference from the DSP CPU updating an L1D cache location is automatically made visible to the DMA and any master processor on the device with access to the DSP megamodule memory-mapped L2 through the IVA2.2 slave port.

An L2 reference from the DMA and any other master processor on the device with access to the DSP megamodule memory-mapped L2 through the IVA2.2 subsystem slave port is automatically made visible to the DSP CPU through the L1D cache, if it is holding the associated cache line.

Note: To reduce the complexity of the L1P cache controller to the L2 controller interface, the L1P cache coherency protocol is removed. This means that coherency between L2 cache and L1P cache is not maintained. On the C64x+ CPU, writes to L2 do not invalidate the corresponding region in L1P. This must be done manually.

14.4.3.4.3 Device Memory Coherence

Coherence is not maintained by hardware between the L2 cache (and L1D cache/L1P cache) and device memories (on-chip memories external to the IVA2.2 subsystem and off-chip memories connected to the SDRAM controller and/or general-purpose memory controller [GPMC]). Coherence in this case must be handled by software. The DSP megamodule offers two sets of registers for user-initiated coherence maintenance between DSP local memories and device memories.

Software coherence maintenance must be synchronized in the system (for example, through message passing; for information, see Chapter 6, *Interprocessor Communication*). In the producer/consumer model,

the producer sends a completion message to the consumer only after the write is complete in end memory. If the producer has a cache-based architecture, the producer must initiate a write-back and track for the completion of the write-back sequence in end memory. For a description of how the write-back sequence occurs in the IVA2.2 subsystem, see Section 14.4.3.4.6, *Write-Back Completion*. When the consumer receives the message, if the consumer has a cache-based architecture, updates by the producer must be effectively seen (not a local nonupdated copy). For a description of how the invalidate sequence occurs in the IVA2.2 subsystem, see Section 14.4.3.4.4, *Global Cache Management*.

Two types of cache coherence management are possible:

- Global cache coherence allows the DSP CPU to ensure coherence of the entire cache at once. See Section 14.4.3.4.4, *Global Cache Management*.
- Block cache coherence allows the DSP CPU to ensure coherence of a contiguous region of the virtual address map, restricted in size to only what is needed. See Section 14.4.3.4.5, *Block Cache Management*.

Each management type provides three sets of actions:

- Invalidate ensures that all required lines are made invalid in the cache. After that operation, any update (before invalidate operation) in end memory by an alternate processor/DMA is seen by the DSP CPU, forcing a cache line refill.
- Write-back ensures that all required cache lines modified by the DSP CPU (also called dirty lines) are written back to end memory, so that any local update by the DSP CPU is made visible by an alternate processor/DMA. This applies only to L1D cache and L2 cache, not to L1P cache.
- Write-back and invalidate ensure that all required cache lines modified by the DSP CPU (also called dirty lines) are written back to end memory so that any local update by the DSP CPU is made visible by an alternate processor/DMA, and they ensure that all required lines are made invalid in the cache. After that operation, any update in end memory by an alternate processor/DMA is seen by the DSP CPU, forcing a cache line refill. This applies only to L1D cache and L2 cache, not to L1P cache.

14.4.3.4.4 Global Cache Management

This section describes how to invalidate and write back the cache memory

- Global invalidate

  Global invalidate is controlled by the following registers: IVA_XMC.L2INV, IVA_XMC.L1DINV, and IVA_XMC.L1PINV.

  Example of global invalidate:

```
/* ---------------------------------------------------
            */
/* Invalidate anything held in cache. */
/* ---------------------------------------------------
            */
L2INV = 1;
/* ---------------------------------------------------
            */
/* Now, spin waiting for operation to complete. */
/* ---------------------------------------------------
            */
while ((L2INV & 1) != 0)
;
```

  L2INV = 0 ensures that L1D cache and L1P cache are also globally invalidated before the L2 cache invalidate process.

- Global write-back

  Global write-back is controlled by the following registers: IVA_XMC.L2WB and IVA_XMC.L1DWB.

  Example of global write-back:

```
/* ---------------------------------------------------
            */
/* Write back anything held in cache. */
/* ---------------------------------------------------
```

```
                */
L2WB = 1;
/* ----------------------------------------------------
                */
/* Now, spin waiting for operation to complete. */
/* ----------------------------------------------------
                */
while ((L2WB & 1) != 0)
;
```

To ensure write-back completion of a specific buffer (contiguous address range), see Section 14.4.3.4.6, *Write-Back Completion*, for additional programming steps and an example.

- Global write-back and block invalidate

  Global write-back and block invalidate is controlled by the following registers: IVA_XMC.L2WBINV and IVA_XMC.L1DWBINV.

  Example of global write-back and block invalidate:

```
/* ----------------------------------------------------
                */
/* Write back and invalidate anything held in cache.
                */
/* ----------------------------------------------------
                */
L2WBINV = 1;
/* ----------------------------------------------------
                */
/* Now, spin waiting for operation to complete. */
/* ----------------------------------------------------
                */
while ((L2WBINV & 1) != 0)
;
```

To ensure write-back completion of a specific buffer (contiguous address range), see Section 14.4.3.4.6, *Write-Back Completion*, for additional programming steps and an example.

14.4.3.4.5 Block Cache Management

- Block invalidate

  Block invalidate is controlled by the following registers: IVA_XMC.L2BAR, IVA_XMC.L2IWC, IVA_XMC.L1DIBAR, IVA_XMC.L1DIWC, IVA_XMC.L1PIBAR, and IVA_XMC.L1PIWC.

  Example of block invalidate:

```
/* ----------------------------------------------------
                */
/* Write base address of array to Base Address Register.
                /
/* Then write length of the array, in words, to the
                Word*/
/* Count register. */
/* ----------------------------------------------------
                */
L2IBAR = &array[0];
L2IWC = = sizeof(array) / sizeof(int);
/* . . . */
/* ----------------------------------------------------
                */
/* The CPU can execute other code here. Block cache
                operations proceed in parallel with CPU execution, stalling the
                CPU minimally. */
/* ----------------------- -------------------------
                */
/* . . . */
```

```
/* ------------------------------------------------------
                */
/* Now, spin waiting for operation to complete. */
/* ------------------------------------------------------
                */
while (L2IWC != 0)
;
```

L2IWC = 0 ensures that L1D cache and L1P cache are also block invalidated before the L2 cache invalidate process.

- Block write-back

  Block write-back is controlled by the following registers: IVA_XMC.L2WBAR, IVA_XMC.L2WWC, IVA_XMC.L1DWBAR, and IVA_XMC.L1DWWC.

  Example of block write-back:

```
/* ------------------------------------------------------
                */
/* Write base address of array to Base Address
                Register.*/
/* Then write length of array, in words, to the Word
                */
/* Count register. */
/* ------------------------------------------------------
                */
L2WBAR = &array[0];
L2WWC = sizeof(array) / sizeof(int);
/* . . . */
/* ------------------------------------------------------
                */
/* CPU can execute other code here. Block cache operations
                proceed in parallel with CPU execution, stalling CPU minimally.
                */
/* ------------------------------------------------------
                */
/* . . . */
/* ------------------------------------------------------
                */
/* Now, spin waiting for operation to complete. */
/* ------------------------------------------------------
                */
while (L2WWC != 0)
;
```

To ensure write-back completion of a specific buffer (contiguous address range), see Section 14.4.3.4.6, *Write-Back Completion*, for additional programming steps and an example.

- Block write-back and invalidate

  Block write-back and invalidate is controlled by the following registers: IVA_XMC.L2WIBAR, IVA_XMC.L2WIWC, IVA_XMC.L1DWIBAR, and IVA_XMC.L1DWIWC.

  Example of block write-back and invalidate:

```
/* ------------------------------------------------------
                */
/* Write base address of array to Base Address
                Register.*/
/* Then write length of array, in words, to the Word
                */
/* Count register. */
/* ------------------------------------------------------
                */
L2WIBAR = &array[0];
L2WIWC = (array) / sizeof(int);
```

```
/* . . . */
/* ------------------------------------------------------
              */
/*The CPU can execute other code here. Block cache
              operations proceed in parallel with CPU execution, stalling the
              CPU minimally. */
/* ------------------------------------------------------
              */
/* . . . */
/* ------------------------------------------------------
              */
/* Now, spin waiting for operation to complete. */
/* ------------------------------------------------------
              */
while (L2WIWC != 0)
;
```

To ensure write-back completion of a specific buffer (contiguous address range), see Section 14.4.3.4.6 for additional programming steps and an example.

14.4.3.4.6 Write-Back Completion

The SYSC_LICFG0[15] GEMTRUECOMPEN bit must be set to 1 before any DSP CPU C64x+ write for which completion must be ensured. This applies to writes to noncacheable regions and to cache-line write-back completions. By default, SYSC_LICFG0[15] GEMTRUECOMPEN = 0. This default is recommended, to statically set that bit (unless the user wants to locally send a large number of writes to a noncacheable memory region):

- SYSC_LICFG0.GEMTRUECOMPEN = 1;

  Polling on the L*WWC (resp. L*WIWC) ensures that the block write-back operation (and associated invalidate, when applicable) was completely issued by the associated cache controller, but does not ensure that the write-back is effective in end memory.

  To get completion of the write-back of a specific buffer after block or global cache write-back, the user must read back from a noncacheable region of the same L3 or L4 target as the buffer (if data is written in external DDR memory, the read access can be on an SDRC register). The C64x + is uninstalled only after the read following the write-back is complete.

  In the producer/consumer model, the cache-based producer typically ensures that the produced buffer is visible to the consumer of that buffer by writing back the address range of that buffer and checking for completion of the write-back sequence before sending a completion message to the consumer.

  Example:

```
/* ------------------------------------------------------
              */
/* nonCachedArea to be linked to non-cached SDRAM
              region*/
/* ------------------------------------------------------
              */
#pragma DATA_SECTION(nonCachedDummyVar,
              ".nonCachedArea")
volatile int nonCachedDummyVar;
/* ------------------------------------------------------
              */
/* outBuffer is produced buffer in SDRAM to be visible to
              the consumer */
/* ------------------------------------------------------
              */
L2WBAR = &outBuffer[0];
L2WWC = sizeof(outBuffer) / sizeof(int);
/* ------------------------------------------------------
              */
/* L2WWC ensures the sequence is completed by the cache
```

```
                controller but not completed in end memory */
/* ---------------------------------------------------
                */
while (L2WWC != 0)
;
/* ---------------------------------------------------
                */
/*C64x+ is stalled until dummy memory read has completed
                which the hardware ensures happens after write-back of outBuffer
                completed in SDRAM. */
/* ---------------------------------------------------
                */
int dummyRead = nonCachedDummyVar;
/* ---------------------------------------------------
                */
/* Then C64x SW (producer) can send message to consumer
                */
/* ---------------------------------------------------
                */
```

- sendCompletionMsgToConsumer();

Note: To ensure completion, the nonCachedDummyVar must be in the same target as the written-back buffer.

In the case of the C64x writing in the noncache area: To ensure the completion of the C64x write in physical end memory, set the SYSC_LICFG0 [15] GEMTRUECOMPEN bit to 1 and also read back after the last C64x write.

1. DSP write and DMA read
   The user writes to some noncache region with DSP and then reads from the same area with DMA. To ensure completion of the DSP write in physical end memory, set the SYSC.LICFG0[15] GEMTRUECOMPEN bit to 1 and also read back after the last DSP write. The sequence is:
   a. Set the SYSC_LICFG0[15] GEMTRUECOMPEN bit to 1.
   b. DSP writes data in external memory, noncache area.
   c. DSP reads data in external memory, noncache area: last write data for instance.
   d. DMA reads data from external memory.
2. DMA write and DSP read
   The opposite of 1. To ensure the completion of DMA write in physical end memory, set the SYSC_LICFG0 DMATRUECOMPEN bit to 1 and PARAM [LCHi].OPT.TCCMODE to 0 (no early completion). The sequence is:
   a. Set the SYSC_LICFG0.DMATRUECOMPEN bit to 1.
   b. Set PARAM[LCHi].OPT.TCCMODE to 0.
   c. Set PARAM[LCHi].OPT.TCCMODE to 0.
   d. DSP waits for end of DMA transferred:
      - IPR/IPRH bit update (for polling-scheme)
      - Interrupt generation (for interrupt-scheme)
      - CER/CERH bit update (for chaining)
      - DSP reads data from external memory.

14.4.3.4.7 Performance Consideration Timing

Long burst is key to improving the efficiency of the SDRAM and reducing cache-line refill and cache-line write-back latencies. The IVA2.2 subsystem allows for improving the burst generation over the device interconnect. This is configurable with the SYSC_LICFG0[16].GEMBURSTOPTEN bit. By default, SYSC_LICFG0[16] GEMBURSTOPTEN = 0, and the burst optimization is disabled. This default is recommended to statically set the bit:

SYSC_LICFG0.GEMBURSTOPTEN = 1

14.4.4 DMA Management

14.4.4.1 Transfers From/to Device Memories/Peripherals (EDMA)

The EDMA optimizes transfers from/to the DSP megamodule (memory-mapped) memories to/from device on-chip and off-chip memories.

The EDMA can perform transfers from IVA2.2 internal memories to IVA2.2 internal memories, but it is not designed for that purpose. The IDMA is recommended in that case (unless a 2-dimensional transfer is required).

The EDMA can perform transfers from device memories/peripherals to device memories/peripherals, but it is not designed for that purpose. The sDMA is recommended in that case. For more information, see Chapter 9, *DMA*.

14.4.4.2 Internal Memory-to-Memory Transfer (IDMA)

The user can quickly page memory regions or fill a memory region of the DSP megamodule memories by using channel 1 of the IDMA module (internal to the DSP megamodule).

The IDMA.IDMA1_COUNT[16] FILL bit defines whether this is a transfer from memory to memory or if this is a solid-color fill.

An IDMA1 transfer where iDMA.IDMA1_COUNT[16] FILL = 0 copies IDMA.IDMA1_COUNT[15:2] COUNT bytes from the address defined in the IDMA.IDMA1_SOURCE register to the address defined in the IDMA.IDMA1_DEST register. The addresses must be aligned on word (4-byte) boundaries. The byte count must be a multiple of 4 bytes.

The IDMA.IDMA1_COUNT[28] INT register bit enables the IDMA_INT1 interrupt (Evt 14; for information about generation on completion of the IDMA1 transfer, see Table 14-3). By default, no interrupt is generated.

When conflicts occur, the IDMA.IDMA1_COUNT[31:29] PRI bit field defines the priority of the IDMA transfer with respect to DSP and DMA/HOST accesses.

The memory paging programming example follows:

- IDMA1_SOURCE = &mySrcTable[0]; // mySrcTable aligned on word boundary
- IDMA1_DEST = &myDstTable[0]; // myDstTable aligned on word boundary
- IDMA1_COUNT = (IDMA1_COUNT & ~(0xFFFC)) | size of (mySrcTable);
- IDMA1_COUNT = (IDMA1_COUNT & ~(116)) | 016; // copy mode
- IDMA1_COUNT = (IDMA1_COUNT & ~(128)) | 028; // no interrupt
- IDMA1_COUNT = (IDMA1_COUNT & ~(0x729)) | 0x729; // low priority

An IDMA1 transfer where iDMA.IDMA1_COUNT[16] FILL = 1 replicates the IDMA.IDMA1_SOURCE word as many times as defined in IDMA. An IDMA1_COUNT.COUNT (count divided by 4) to the address defined in the IDMA.IDMA1_DEST destination address must be aligned on word (4-byte) boundaries. The byte count must be a multiple of 4 bytes.

The solid-color copy (SSC) programming example follows:

- IDMA1_SOURCE = my32bPattern; // 32b pattern to be replicated
- IDMA1_DEST = &myDstTable[0]; // myDstTable aligned on word boundary
- IDMA1_COUNT = (IDMA1_COUNT & ~(0xFFFC)) | size of (mySrcTable);
- IDMA1_COUNT = (IDMA1_COUNT & ~(116)) | 116; // SCC mode
- IDMA1_COUNT = (IDMA1_COUNT & ~(128)) | 028; // no interrupt
- IDMA1_COUNT = (IDMA1_COUNT & ~(0x729)) | 0x729; // low priority

Note: Because the IVA2.2 subsystem is typically configured (recommended for video applications) so that L1D has memory-mapped SRAM, IDMA1 is normally used only for L1D>L1D fast copy.

14.4.4.3 Programming an EDMA Transfer

Programming a complete EDMA transfer requires the following steps:

1. Define the logical channel(s).
2. Prioritize the defined transfer (with respect to other defined transfers).
3. Start the transfer.
4. Review the progression and completion of the transfer.

14.4.4.4 Defining a Logical Channel

14.4.4.4.1 Single Logical Channel Definition

A complete EDMA transfer can be defined by one or several chained and/or linked logical channels.

Up to 128 independent contexts, each fully defining a logical channel, can be defined. These 128 contexts correspond to the 128 PaRAM entries available in the IVA2.2 subsystem. For more information about these PaRAM entries, see Section 14.3.2.1.1.3, *DMA/QDMA Channel Mapping and PaRAM Entry*, and the corresponding Figure 14-13.

Logical channel definition relies on the following:

- Base addresses:
  - PARAM[LCH#].SRCi: 32-bit source address
  - PARAM[LCH#].DSTi: 32-bit destination address
- Transfer sizes:

  Transfer size is common to source and destination. A transfer can be constituted on a 3-dimensional array; C is an array of CCNT arrays, each composed of BCNT arrays, each composed of ACNT bytes:
  - PARAM[LCH#].ACNT: Number of bytes in the A array (from 0 to 65,535)
  - PARAM[LCH#].BCNT: Number of A arrays in the B array (from 0 to 65,535)
  - PARAM[LCH#].CCNT: Number of B arrays in the C array (from 0 to 65,535)

    Note: Setting one of the ACNT, BCNT, or CCNT arrays to 0x0 prevents a transfer from being submitted to one of the physical channels (assuming that the compatibility mode is not set; see Kelvin DMA compatibility mode).
  - PARAM[LCH#].BCNTRLD: BCNT reload value when BCNT reaches 0 (from 0 to 65,535)

    Note: When programming CCNT>1, programming PARAM[LCH#].BCNTRLD to be equal to PARAM[LCH#].BCNT is recommended.
- Indexes between dimensions:
  - PARAM[LCH#].SRCBIDX: Index between A arrays at source (from -32,768 to 32,767)
  - PARAM[LCH#].SCRCIDX: Index between B arrays at source (from -32,768 to 32,767)
  - PARAM[LCH#].DSTBIDX: Index between A arrays at destination (from -32,768 to 32,767)
  - PARAM[LCH#].DSTCIDX: Index between B arrays at destination (from -32,768 to 32,767)

    Note: Programming the logical channel does not automatically start it. See Section 14.4.4.6.1, *Assigning a Logical Channel to a Trigger Event.*
- Addressing modes:
  - PARAM[LCH#].OPT[0] SAM: source addressing mode (0: post-incremented; 1: constant)
  - PARAM[LCH#].OPT[1] DAM: destination addressing mode (0: post-incremented; 1: constant)

Note: Constant addressing mode is supported only from/to IVA2.2 DSP megamodule memories, not from/to device memories and peripherals. This can be replaced by using a post-increment addressing mode and an index equal to 0.

Example:

```
/* ---------------------------------------------------
           */
/*fills dstArray with cstValue values (w/o constant AM)
           */
/* ---------------------------------------------------
           */
```

PARAM[LCH#].SRC = &cstValue;

PARAM[LCH#].DST = &dstArray[0];

PARAM[LCH#].ACNT = sizeof(int);

PARAM[LCH#].BCNT = sizeof(dstArray) / sizeof(int);

PARAM[LCH#].CCNT = 1;

PARAM[LCH#].SRCBIDX = 0;

PARAM[LCH#].DSTBIDX = sizeof(int);

PARAM[LCH#].OPT.SAM = 0;

PARAM[LCH#].OPT.DAM = 0;

14.4.4.4.2 Controlling Submission Granularity

The logical channel (when triggered) can split a transfer into several requests submitted to one of the physical channels. The physical channel supports submitted requests of up to two dimensions, meaning that a 3-dimensional transfer is always split into (at least) 2-dimensional transfers. The user can also program the logical channel so that submitted requests are 1-dimensional transfers; for example:

- PARAM[LCH#].SYNCDIM = 0; // submitted transfers are maximum 1D.
- PARAM[LCH#].SYNCDIM = 1; // submitted transfers are maximum 2D.

14.4.4.4.3 Linking to Another Logical Channel

A logical channel can be programmed so that on its completion another context is copied from another logical channel. This is useful because the logical channel is used as a working set during the DMA transfer, meaning that initial context configuration is lost after the associated transfer completes.

- PARAM[LCH#].LINK = linkLCH# << 5

The LINK value is the base address of the linked logical channel context.

When the LINK value is set to 0xFFFF, no context is loaded for that logical channel:

- PARAM[LCH#].LINK = -1

Note: Only the context is copied, and the logical channel is not automatically restarted on link. Restarting a logical channel that just received its new context occurs only after a trigger event associated with that logical channel is detected. See Section 14.4.4.6.1, *Assigning a Logical Channel to a Trigger Event*.

14.4.4.4.4 Chaining Logical Channel

A logical channel can be programmed so that on its total or partial completion, another logical channel is started. This is useful for defining a series of transfers with different contexts as a complete DMA transfer. The main advantage is that the overhead to program all the chained transfers is shared among all channels.

- Partial completion chaining

  After the submitted section (see Section 14.4.4.4.2, *Controlling Submission Granularity*) of a logical channel is complete, a programmable completion code is returned. If partial completion chaining is enabled in the context of the logical channel, this completion code defines which trigger event is set. The logical channel associated with that trigger event is automatically submitted. See Section 14.4.4.6.1, *Assigning a Logical Channel to a Trigger Event*.

  To chain a logical channel LCHi to LCHj, so that LCHj is automatically started after each LCHi submission to a physical channel has completed:

  – PARAM[LCHi].OPT.TCC = trigEvtx; // trigEvtx: trigger event number
  – PARAM[LCHi].OPT.ITCCHEN = 1
  – DCHMAP[trigEvtx] = LCHj

- Total completion chaining

  After all submitted parts of a logical channel are complete, programmable completion code is returned. If total completion chaining is enabled in the context of the logical channel, this completion code defines which trigger event is set. The logical channel associated with that trigger event is automatically submitted. See Section 14.4.4.6.1, *Assigning a Logical Channel to a Trigger Event*.

  For example, to chain a logical channel LCHi to LCHj so that LCHj is automatically started after LCHi has completed:

  – PARAM[LCHi].OPT.TCC = trigEvtx; // trigEvtx: trigger event number
  – PARAM[LCHi].OPT.TCCHEN = 1
  – DCHMAP[trigEvtx] = LCHj

14.4.4.5 Prioritizing Defined Transfers

14.4.4.5.1 Mapping Between DMA/QDMA Events and Event Queues

The assignment of 64 DMA and 8 QDMA channels to two event queues of the channel controller is achieved by configuring the DMA queue number registers (TPCC_DMAQNUM0 and TPCC_DMAQNUM1) and the QDMA queue number register (IVA_TPCC.QDMAQNUM). When an event is selected for submission, it is queued in the user-defined event queue. This typically defines a 2-level priority preemption scheme, as queues are usually mapped to different transfer controllers.

14.4.4.5.2 Mapping a Queue to a Transfer Controller

Events at the head of an event queue define which logical channel (Param Entry) is selected for submission to the associated physical channel (transfer controller). The association between an event queue and a physical channel can be defined in the TPCC_QUETCMAP register. By default, TPTC0 is associated with event queue 0, and TPTC1 is associated with event queue 1. This mapping is a static decision; it does not change during DMA operation.

Note: TPTC0 and TPTC1 are not symmetrical; their numbers are just inverted, compared to the Davinci device. To improve compatibility with Davinci devices, invert the mapping of the event queue to the transfer controller, as shown below:

QUETCMAP = (QUETCMAP & ~0xFF) | 0x10;

Typically, this is used to allow submission-time preemption, so that by example:

- Event queue 0 is used for background, potentially long, not very latency-sensitive, not very critical DMA transfers.
- Event queue 1 is used for short, latency-sensitive, or critical (for example, hardware synchronized) DMA transfers.

14.4.4.5.3 Handling Priority

In IVA2.2 local interconnect arbitration, priority of individual bus requests for a DMA transfer over other DMA- or CPU-initiated bus requests is defined by the event queue to which the event associated with the transfer is submitted. This can be configured per event queue in the TPCC_QUEPRI[2:0] PRIQ0 and TPCC_QUEPRI[6:4] PRIQ1 bit fields. By default, event queues 0 and 1 have the same priority (highest possible priority is 0x0).

CPU bus-requests priority is defined in the IDMA.MDMAARBE[18:16] PRI bit field. By default, the CPU has the lowest possible priority (0x7).

Typically, this is used to allow transfer-time preemption, so that, for example, bus requests associated with event queue 1 are served ahead of event queue 1 or CPU requests.

Example:

- // DMA#0->0x7 (lowest), DMA#1->0x0 (highest), CPU->0x4 (mid)
- QUEPRI = (QUEPRI & ~0xFF) | 0x07;
- MDMAARBE = (QUEPRI & ~(0xF16)) | 0x4 16;

14.4.4.5.4 Aged Priority

To prevent having a DMA request stalled for a very long time, the IVA2.2 implements an aged priority scheme (also referred to as an inversion priority scheme) on the DMA ports to change the priority defined in the TPCC_QUEPRI register. This is done by regularly decreasing the priority level (increasing priority in arbitration) of a stalled request. The interval between two consecutive updates of the priority level is defined in the SYSC_LICFG1 register. By default, the aged priority scheme is disabled (SYSC_LICFG1=0x0), and arbitration priority is dictated by programmed values in QUEPRI and MDMAARBE.

14.4.4.5.5 Optimizing 2D Transfers

IVA2.2 EDMA can be configured so that DMA 2D transfers are optimized, allowing for large bursts to be generated to the SDRAM. This optimization has no effect on transfers issued as 1D transfers to the physical channels. This is recommended to enable that feature when the sources or destinations of the 2D transfers are the VRFB (SDRAM tiling structure).

To fully benefit from the optimization and disable MMU page-crossing checks:

- Large MMU page(s) are defined for the VRFB view(s) (typically 16MB supersection).
- The user software ensures that a 2D transfer does not span MMU large pages.

With the preceding conditions, use the following settings to use the IVA2.2 2D burst optimization:

- IVA_SYSC.SYSC_LICFG0.DMA2DOPTEN = 1
- IVA_SYSC.SYSC_LICFG0.PAGEXINGEN = 1

Note: The user software must ensure that a 2D transfer never spans MMU page boundaries. The reason for IVA_SYSC.SYSC_LICFG0.PAGEXINGEN = 1 is to remove the hardware check mechanism of 2D bursts crossing MMU pages. A 2D burst that spans these boundaries can lead to undefined behavior. PAGEXINGEN = 0 prevents such situations through hardware.

14.4.4.6 Starting the Transfer

Before starting the transfer, a trigger event must be associated with the logical channel. Three modes trigger a DMA transfer:

- Manual trigger (software-synchronized transfers)
- Hardware trigger (hardware-synchronized transfers)
- Automatic trigger (automatic on-submission transfer start)

14.4.4.6.1 Assigning a Logical Channel to a Trigger Event

The 64 DMA channels and the 8 QDMA channels can be flexibly mapped to any of the 128 available PaRAM entries (see Figure 14-13).

Any of the 64 DMA channels can be mapped to any of the 128 PaRAM entries through DMA channel-mapping registers TPCC_DCHMAPi (i = 0 to 63).

Any of the 8 QDMA channels can be mapped to any of the 128 PaRAM entries through QDMA channel-mapping registers TPCC_QCHMAPj (j = 0 to 7).

14.4.4.6.2 Manual Trigger (Software-Synchronized Transfers)

When a logical channel is defined and prioritized, the user can assign the logical channel (or the first in the chained list) to a trigger event (from 0 to 63) by writing the number of the logical channel (PaRAMEntry #) to one of the DMA channel-mapping registers TPCC_DCHMAPi (i = 0 to 63). Then, the user can manually start the transfer (one logical channel or a chained list of logical channels) by writing 1 to the bit in the TPCC_ESR or TPCC_ESRH register associated with the trigger event of the logical channel (or the first logical channel in the chained list).

Note: The event does not need to be enabled in the TPCC_EER register to be manually triggered.

Example:

```
/* ---------------------------------------------- */
/*To manually start defined logical channel #0x3, uses event
            #20 */
/* ---------------------------------------------------
            */
DCHMAP[20] = (DCHMAP[20] & ~(0x1FF<<5)) |
            0x3<<5;
ESR = 1 << 20;
```

14.4.4.6.3 Hardware Trigger (Hardware-Synchronized Transfers)

When a logical channel is defined and prioritized, the user can assign the logical channel (or the first in the chained list) to a trigger event (from 0 to 19) by writing the number of the logical channel (PaRAMEntry #) to one of the DMA channel-mapping registers TPCC_DCHMAPi (i = 0 to 19). The user can allow this logical channel to be triggered by an associated hardware DMA request by writing 1 in the associated bit of the EER register. The mapping of a hardware DMA request to DMA events is fixed. The mapping of DMA requests to device peripheral sources is listed in Table 14-2.

Example:

```
/* ---------------------------------------------------
            */
/* Associate defined logical channel #0x5 to
            UART3_DMA_TX*/
/* UART3_DMA_TX is DMA request #10 and associated to evt #10
            */
/* ---------------------------------------------------
            */
```

DCHMAP[10] = (DCHMAP[10] & ~(0x1FF<<5)) | 0x5<<5;

14.4.4.6.4 Automatic Trigger (QDMA)

The user can specify a trigger word from among any of the eight 32-bit words of the logical channel context (PaRAM entry) for QDMA. Writing to the trigger word triggers the channel controller of QDMA to issue a transfer request. The trigger word field of the QDMA channel mapping register TPCC_QCHMAPj (where j = {0 to 7}) defines the trigger word for a particular QDMA channel, as shown in Figure 14-13.

This flexibility enables the CPU to selectively modify only the PaRAM entry that requires modification, and thereby trigger the transfer. For example, after a transfer, if only count must change, QCHMAP can be configured so that count is the trigger word, and a write to it automatically triggers the transfer.

Example:

```
/* ---------------------------------------------------
            */
/* Associate defined logical channel #0x5 to QDMA #1
            */
/* ---------------------------------------------------
            */
```

QCHMAP[1] = (QCHMAP[1] & ~(0x1FF<<5)) | 0x5<<5;

```
/* ---------------------------------------------------
            */
/* Define DST parameter (0x3) to be trigger word of LCH
            */
/* ---------------------------------------------------
            */
```

QCHMAP[1] = (QCHMAP[1] & ~(0x7<<2)) | 0x3<<2;

In addition, the IDMA can be used to offload the CPU of the DMA configuration. See Section 14.4.4.6.5, *Offloaded Configuration (Using IDMA)*.

14.4.4.6.5 Offloaded Configuration (Using IDMA)

The IVA2.2 allows quick programming of DMA transfers by offloading the CPU of most of the DMA transfer issue time. To do so, the user typically maintains a copy of the logical channel contexts (PaRAM entries) in L1D SRAM. A CPU update of a logical channel context is very fast in L1D SRAM. After completing a logical channel context update, the CPU can page the context from L1D to DMA PaRAM entries using another simple DMA, internal-to-DSP megamodule (IDMA). For example:

disable_interrupts();

while(IDMA0_STATUS & 0x3);// previous IDMA completion/*

```
---------------------------------------------------
            */
/* Update of logical channels definition table in L1D
            */
/* ---------------------------------------------------
            */
```

LCTable->OPT = opt;

LCTable->SRC = src;

LCTable->ACNT = num_bytes;

LCTable->BCNT = num_arrays;

LCTable->DST = dst;

LCTable->DSTBIDX = dbidx;

LCTable->SRCBIDX = sbidx;

LCTable->LINK = 0xFFFF;

LCTable->BCNTRLD = bcntrld;

LCTable->DSTCIDX = dcidx;

LCTable->SRCCIDX = scidx;

LCTable->CCNT = num_frames;

```
/* ---------------------------------------------------
            */
```

```
/* initiate IDMA transfer */
/* ------------------------------------------------
          */
IDMA0_SOURCE = &LCTable[0];
IDMA0_DEST = &PaRAM[0];
IDMA0_MASK = 0xFFFFFF00;
IDMA0_COUNT = 0x0;
enable_interrupts();
```

14.4.4.6.6 Direct Configuration to Transfer Channel (Not Recommended)

The registers of the physical channels are memory-mapped, primarily to enable, clear, and read status for error interrupts generated by the physical channel. For more information, see Section 14.4.10, *Error Identification Process*.

It is possible to write directly to other control registers of the physical channels and issue a transfer; however, this is not recommended and therefore is not described here.

Note: This can work safely only if the DMA controller is not used.

14.4.4.6.7 DMA Completion Mode

The IVA2.2 EDMA defines when a DMA transfer is complete:

- Early completion: All associated transfers were submitted to the physical channel. Early completion does not ensure that transfer is complete in end memory.
  - PARAM[LCHi].OPT.TCCMODE = 1
- True completion: All associated transfers were submitted to the physical channel, and all those transfers are complete, from the physical channel standpoint. True completion ensures that transfer is complete in end memory.
  - SYSC.SYSC_LICFG0.DMATRUECOMPEN = 1; // is statically set.
  - PARAM[LCHi].OPT.TCCMODE = 0

Note: TCCMODE = 0 does not ensure that transfer is complete in end memory if DMATRUECOMPEN = 0.

By default, DMATRUECOMPEN = 0. This is recommended to statically set DMATRUECOMPEN.

True completion is typically used when the IVA2.2 DMA is the producer of a buffer shared with another master processor or a DMA (consumer). A completion message is sent to the consumer only after the DMA transfer is complete.

Completion mode affects timing for the following actions:

- IPR/IPRH bit update (for polling scheme)
- Interrupt generation (for interrupt scheme)
- CER/CERH bit update (for chaining)

14.4.4.6.8 Partial Versus Total Completion

DMA can be programmed so that IPR bit update and interrupt generation occur:

- After each submission to the physical channel is complete
- After the last submission to the physical channel is complete

Partial completion interrupt

After the submitted section (see Section 14.4.4.4.2, *Controlling Submission Granularity*) of a logical channel is complete, a programmable completion code is returned. If a partial completion interrupt is enabled in the context of the logical channel, this completion code defines which IPR bit is set.

For example, to allow IPR to be updated (and possibly an interrupt to be generated) after each LCHi submission to a physical channel is complete:

- PARAM[LCHi].OPT.TCC = intEvtx; // intEvtx: IPR bit to be updated
- PARAM[LCHi].OPT.ITCINTEN = 1

Total completion interrupt

After all submitted parts of a logical channel are complete, a programmable completion code is returned. If a total completion interrupt is enabled in the context of the logical channel, this completion code defines which IPR bit is to be set.

For example, to allow IPR to be updated (and possibly an interrupt to be generated) after all LCHi submissions to a physical channel are complete:

- PARAM[LCHi].OPT.TCC = intEvtx; // intEvtx: IPR bit to be updated
- PARAM[LCHi].OPT.TCINTEN = 1
  - By polling
  - By interrupt

14.4.4.6.9 Tracking DMA Completion

There are two ways to track DMA completion:

- Polling the completion register when the estimated completion time has elapsed
- Enabling completion interrupts

Polling example (total completion)

- PARAM[myLCH].OPT.TCINTEN = 1; // total interrupt completion bit update
- PARAM[myLCH].OPT.ITCINTEN = 0; // no partial interrupt completion bit update
- PARAM[myLCH].OPT.TCC = myTCC
- // myTCC does not contribute to interrupt generation (polling mode)
- IER = (IER & ~(1<<myTCC)) | 0<<myTCC
- // start transfer
- DCHMAP[myEvt] = (DCHMAP[myEvt] & ~(0x1FF<<5)) | myLCH<<5
- ESR = 1 << myEvt
- // do something useful (that does not depend on DMA completion)
- // poll IPR bit
- while(!(IPR & (1<<myTCC))); // polls for completion

Interrupt example (total completion)

- disable_interrupts()
- PARAM[myLCH].OPT.TCINTEN = 1; // total interrupt completion bit update
- PARAM[myLCH].OPT.ITCINTEN = 0; // no partial interrupt completion bit update
- PARAM[myLCH].OPT.TCC = myTCC
- // myTCC does contribute to interrupt generation (polling mode)
- IER = (IER & ~(1<<myTCC)) | 1<<myTCC
- INTMUX[0] = (INTMUX[0] & ~(0x7F)) | 0x1D; // map CPU it #4
- CPU.IER = (CPU.IER & (1<<4)) | 1<<4; // unmask CPU it #4
- enable_interrupts()
- // start transfer
- DCHMAP[myEvt] = (DCHMAP[myEvt] & ~(0x1FF<<5)) | myLCH<<5
- ESR = 1 << myEvt
- // do something useful (that does not depend on DMA completion)

- // this code is interrupted when DMA completes

14.4.4.6.10 DMA Interrupt Service Routine

The channel controller does not generate a new interrupt signal for new pending interrupts if the user did not clear previous pending interrupts. There are two options for constructing an ISR for DMA. The first is to poll all the bits during execution of the ISR, and to clear all enabled bits in an interrupt-pending register by writing to the interrupt-pending clear (ICR/ICRH) register before exiting any ISR. Pseudo-code for this option follows:

1. Enter ISR.
2. Read TPCC_IPR.
3. For the condition set in TPCC_IPR,
   a. Perform operation as needed.
   b. Clear bit for serviced INT.
4. Read IPR.
   a. If TPCC_IPR = 0, exit ISR.
   b. If TPCC_IPR = 1, go to Step 3.

The second option is to service the ISR and then, before exiting the ISR, to write to the EVAL bit of the IEVAL register. This forces the EDMA channel controller to generate a new interrupt signal if there are still pending interrupts in the interrupt-pending register (IPR/IPRH). Pseudo-code for this ISR option follows:

1. Enter ISR.
2. Read TPCC_IPR.
3. For the condition set in TPCC_IPR,
   a. Perform operation as needed.
   b. Clear bit for serviced INT.
4. Read IPR.
   a. If TPCC_IPR=0, exit ISR.
   b. If TPCC_IPR=1, set the TPCC_IEVAL.EVAL bit to force triggering a new interrupt.

14.4.4.6.11 Benchmarking

The DMA channel controller monitors the number of event entries in an event queue to determine whether they exceed the user-programmed threshold. The queue threshold for each event queue can be programmed in the TPCC_QWMTHRA and TPCC_QWMTHRB registers. When the number of event entries in an event queue exceeds the user-programmed threshold, a queue threshold error occurs. This error is captured in the THRXCD field of event-queue status register TPCC_QSTATl (l = 0 or 1) and in the QTHRXCDn field in the TPCC_CCERR register.

14.4.5 IVA2.2 Extended Function Interface

This section describes the extended function interface (EFI) signals, timing, and associated instructions. The EFI runs at the CPU clock that allows the transfer of data between the CPU register file and registers at the CPU boundary that are attached to devices outside the CPU. The C64x CPU does not support an EFI. This section applies only to the C64x+ CPU. For information about other C64x+ instructions, see the *TMS320C64x/C64x+ DSP CPU and Instruction Set Reference Guide* (TI literature number SPRU732).

14.4.5.1 Overview

The EFI provides hardware support for data and command transfers to registers outside the CPU in multi-CPU devices.

14.4.5.2 C64x+ EFI Instructions

The following instruction can be used to transfer commands by using the EFI:

- EFCMD: Send 10-bit command code

The following instructions can be used to receive data by using the EFI:

- EFRW: Receive word
- EFRDW: Receive double word

The following instructions can be used to transfer data by using the EFI:

- EFSW: Send word
- EFSDW: Send double word

EFCMD *Send Command to EFI*

Syntax: EFCMD (.unit) ucst10

.unit = .S1, .S2, .M1, .M2

Compatibility: C64x+ CPU only

Opcode: .S Unit

31		29	28	27				23	22				18	17				13	12	11					6	5	4	3	2	1	0
0	0	0	0	dst					src2					0	1	1	0	1	0	1	1	1	1	0	0	1	0	0	0	s	p

Opcode map field used…	For operand type…	Unit
src2	ucst5	.S1,.S2
dst	ucst5	

Opcode: .M Unit

31		29	28	27				23	22				18	17				13	12	11					6	5	4	3	2	1	0
0	0	0	0	dst					src2					1	1	0	1	1	0	0	0	0	0	1	1	1	1	0	0	s	p

Opcode map field used…	For operand type…	Unit
src2	ucst5	.M1,.M2
dst	ucst5	

Description: A 10-bit data value is sent to the EFI. The five most-significant bits (MSBs) of the 10-bit value are taken from dst; the five least-significant bits (LSBs) are taken from src2. There is an A-side EFI and a B-side EFI. The side used is determined by the S unit that is selected.

This instruction executes unconditionally; that is, it cannot be predicated. The EFCMD instruction does not use any data path resources. It is available for scheduling purposes on the S or M units. Only one EFCMD instruction can be scheduled per side; the S or M slot on a side can be used, but not both in the same execute packet. If used in an SPLOOP body, the EFCMD instruction occupies an execution slot corresponding to the unit specified.

The 10-bit command is output at the CPU boundary in E2. The ready signal is not sampled in conjunction with the EFCMD instruction.

Execution:

if(.S1)	ucst10 → A_EFI_Cmd[9:0]
else if(.S2)	ucst10 → B_EFI_Cmd[9:0]
else if(.M1)	ucst10 → A_EFI_Cmd[9:0]
else if(.M2)	ucst10 → B_EFI_Cmd[9:0]

Instruction type: Single-cycle

Delay slots: 0

EFRDW *Receive Double Word From EFI*

Syntax: EFRDW (.unit) dst_o:dst_e

.unit = .S1, .S2

Compatibility: C64x+ CPU only

Opcode:

31		29	28	27				23	22				18	17				13	12	11					6	5	4	3	2	1	0
0	0	0	0	dst					0	0	0	0	0	0	1	1	1	1	0	1	1	1	1	0	0	1	0	0	0	s	p

Opcode map field used…	For operand type…	Unit
dst	dint	.S1,.S2

Description: A 64-bit data value is read from the EFI and is written to the register pair dst_o:dst_e. There is an A-side EFI and a B-side EFI. The side used is determined by the S unit that is selected. There is no cross-path capability; A-side receives must go to an A-file register pair, and B-side receives must go to a B-file register pair. This instruction executes unconditionally; that is, it cannot be predicated.

The ready signal is sampled in the DS phase of the EFRDW instruction and will cause the CPU to stall before the EFRDW instruction reaches E1 if the data to be read has not yet been registered in the CPU.

Execution:

if(.S1)	A_EFI_Din[63:0] → dst_o:dst_e (A1:A0 – A31:A30)
else // .2	B_EFI_Din[63:0] → dst_o:dst_e (B1:B0 – B31:B30)

Instruction type: Single-cycle

Delay slots: 0

See also: EFRW

EFRW *Receive Word From EFI*

Syntax: EFRW (.unit) dst

.unit = .S1, .S2

Compatibility: C64x+ CPU only

Opcode:

31		29	28	27				23	22				18	17				13	12	11					6	5	4	3	2	1	0
0	0	0	0	dst				0	0	0	0	0	0	1	1	1	0	0	1	1	1	1	0	0	1	0	0	0	s	p	

Opcode map field used…	For operand type…	Unit
dst	int	.S1,.S2

Description: A 32-bit data value is read from the EFI and is written to dst. There is an A-side EFI and a B-side EFI. The side used is determined by the S unit that is selected. There is no cross-path capability; A-side receives must go to an A-file register pair, and B-side receives must go to a B-file register pair. This instruction executes unconditionally; that is, it cannot be predicated.

The ready signal is sampled in the DS phase of the EFRW instruction and will cause the CPU to stall before the EFRDW instruction reaches E1 if the data to be read has not yet been registered in the CPU.

Execution:

```
if(.S1)          A_EFI_Din[31:0] → dst (A0 – A31)
else // .2       B_EFI_Din[31:0] → dst (B0 – B31)
```

Instruction type: Single-cycle

Delay slots: 0

See also: EFRDW

EFSDW *Send Double Word to EFI*

Syntax: EFSDW (.unit) src1, src2, dst

.unit = .L1, .L2

Compatibility: C64x+ CPU only

Opcode:

31		29	28	27				23	22				18	17				13	12	11						5	4	3	2	1	0
creg			z	dst				src2					src1					x	0	1	1	1	0	1	0	1	1	0	s	p	

Opcode map field used…	For operand type…	Unit
src1	int	.L1, .L2
src2	xint	
dst	ucst5	

Description: The registers src1 and src2 are read from the register file and combined to form a 64-bit value that is sent to the EFI with a 5-bit constant taken from dst. The src2 register provides the 32 LSBs, and the src1 register provides the 32 MSBs. There is an A-side EFI and a B-side EFI. The side used is determined by the L unit that is selected.

The data and constant are output at the CPU boundary in E2. The EFI ready signal is sampled in E2 of the EFSDW instruction and will cause the CPU to stall if the signal indicates the buffering external to the CPU is full.

Execution:

```
if(cond)
if(.L1)        src1:src2 → A_EFI_Dout[63:0]
               ucst5 → B_EFI_Scmd[4:0]
else // .L2    src1:src2 → B_EFI_Dout[63:0]
               ucst5 → B_EFI_Scmd[4:0]
else           nop
```

Instruction type: Single-cycle

Delay slots: 0

See also: EFSW

EFSW *Send Word to EFI*

Syntax: EFSW (.unit) src1, dst

.unit = .L1, .L2

Compatibility: C64x+ CPU only

Opcode:

31		29	28	27				23	22				18	17				13	12	11						5	4	3	2	1	0
creg			z	dst				src2					0	1	1	1	1	x	0	0	1	1	0	1	0	1	1	0	s	p

Opcode map field used…	For operand type…	Unit
src2	xint	.L1, .L2
dst	ucst5	

Description: The src2 register is read from the register file and sent to the EFI along with a 5-bit constant taken from dst. There is an A-side EFI and a B-side EFI. The side used is determined by the L unit that is selected.

The data and constant are output at the CPU boundary in E2. The EFI ready signal is sampled in E2 of the EFSW instruction and will cause the CPU to stall if the signal indicates the buffering external to the CPU is full.

Execution:

```
if(cond)
if(.L1)        src2 → A_EFI_Dout[31:0]
               ucst5 → A_EFI_Scmd[4:0]
else // .L2    src2 → B_EFI_Dout[31:0]
               ucst5 → B_EFI_Scmd[4:0]
else           nop
```

Instruction type: Single-cycle

Delay slots: 0

See also: EFSDW

14.4.5.3 C64x+ EFI Use in IVA2.2

The following programming model describes how to write and read a register using the EFI. In the context of IVA2.2, only the EFSW, EFRW,and EFSDW instructions are used.

14.4.5.3.1 Read Registers Using the EFI Programming Model

To read a 32-bit value in a video accelerator register using the EFI, the following sequence must be used:

```
EFSW address_32b, opcode_5b
NOP N
EFRW cpu_reg
```

With:

- address_32b is the video accelerator address of the register from which the value must be read.
- opcode_5b is used to code the read operation: 0x02.
- N is the number of cycles to wait data is written in the EFI FIFO. In the context of IVA2.2 N = 0x22.
- Cpu_reg is the 32-bit CPU register where the read value is stored.

Note: To avoid handling specific cases of successive read requests, interrupts are disabled between the time the EFSW is issued and the data is retrieved in the DSP register.

Note: When it is necessary to read a large number of registers, the two macros that correspond to the two instructions given previously can be used to allow issuing a read request while reading data:

- EFSW address_32b, opcode_5b: IME_SendReadReq(unsigned ime_addr_reg)
- EFRW cpu_reg: IME_ReceiveData()

In that case, there is no need to insert a NOPs instruction between the two EFI commands. Read latency is under CPU control.

14.4.5.3.2 Write Registers Using the EFI Programming Model

To write a 32 bit-value in a video accelerator register using the EFI, the following instruction must be used:

```
EFSDW value_32b, address_32b, opcode_5b
```

With:

- The value_32b register contains the 32-bit word to write.
- address_32b is the video accelerator address of the register in which value must be written.
- opcode_5b is used to code the write operation: 0x01.

Note: The address is stored in the even 32-bit register of the 64-bit register pair, while value_32b is stored in the odd register (that is, the MSB part).

14.4.6 iME and iLF Basic Programming Model

14.4.6.1 Typical Use

The iME and iLF coprocessors are parallel coprocessors, so they are partially autonomous and provide highly parallel video processing. To use the iME/iLF coprocessor in a video accelerator design, perform the following steps:

Step 1. Initialize the module.

Step 2. Prepare the data to be processed in the shared L2 memory.

Step 3. Prepare the instructions for processing in the L2 memory.

Step 4. Prepare the parameter stack in the L2 memory.

Step 5. Configure the device to generate an interrupt on completion.

Step 6. Write a LoadInstBuf() instruction in the program buffer register (iME_PROGRAMBUFFERLINENLSBi and iME_PROGRAMBUFFERLINENMSBi or iLF_PROGRAMBUFFERLINENLSBi and iLF_PROGRAMBUFFERLINENMSBi) with the address parameter pointing to the instruction in L2 shared memory.

Step 7. Start the coprocessor by writing to START in the iME_COMMANDREG or iLF_COMMANDREG register.

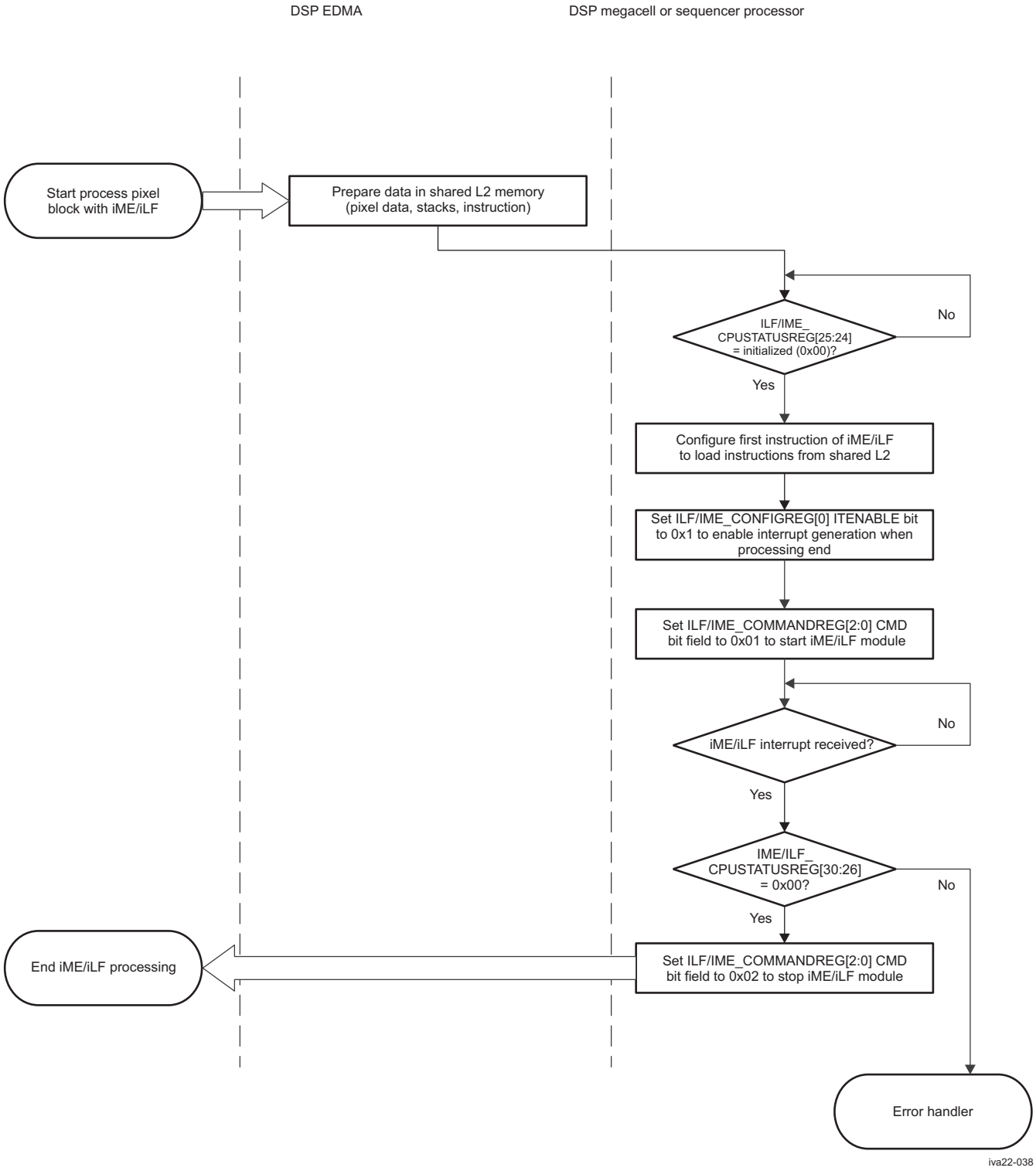
Step 8. Wait for the coprocessor interrupt (the DSP or sequencer is free to perform other tasks during coprocessor processing).

Steps 2, 3, and 4 can be performed with the help of the DSP EDMA, which can be configured by both the sequencer and the DSP megamodule.

Steps 5, 6, and 7 can be performed by both the sequencer and the DSP megamodule, but using the DSP megamodule for this task causes long DSP stalls because of high latency between the DSP megamodule and the iLF/iME module.

The DSP megamodule or the sequencer can prepare multiple sets of data in shared L2 memory (Steps 2, 3, and 4), then the sequencer can launch multiple processing with the coprocessor (Steps 6 and 7) until all prepared data sets in shared L2 memory are used. This way, the DSP and/or DSP EDMA is offloaded for a longer time. Figure 14-38 is a flowchart of a typical use of iLF/iME in video processing. In the flowchart, the four units (DSP, EDMA, sequencer, and coprocessor) work in parallel.

Figure 14-38. iME/iLF Typical Use Flowchart



14.4.7 Interrupt Management

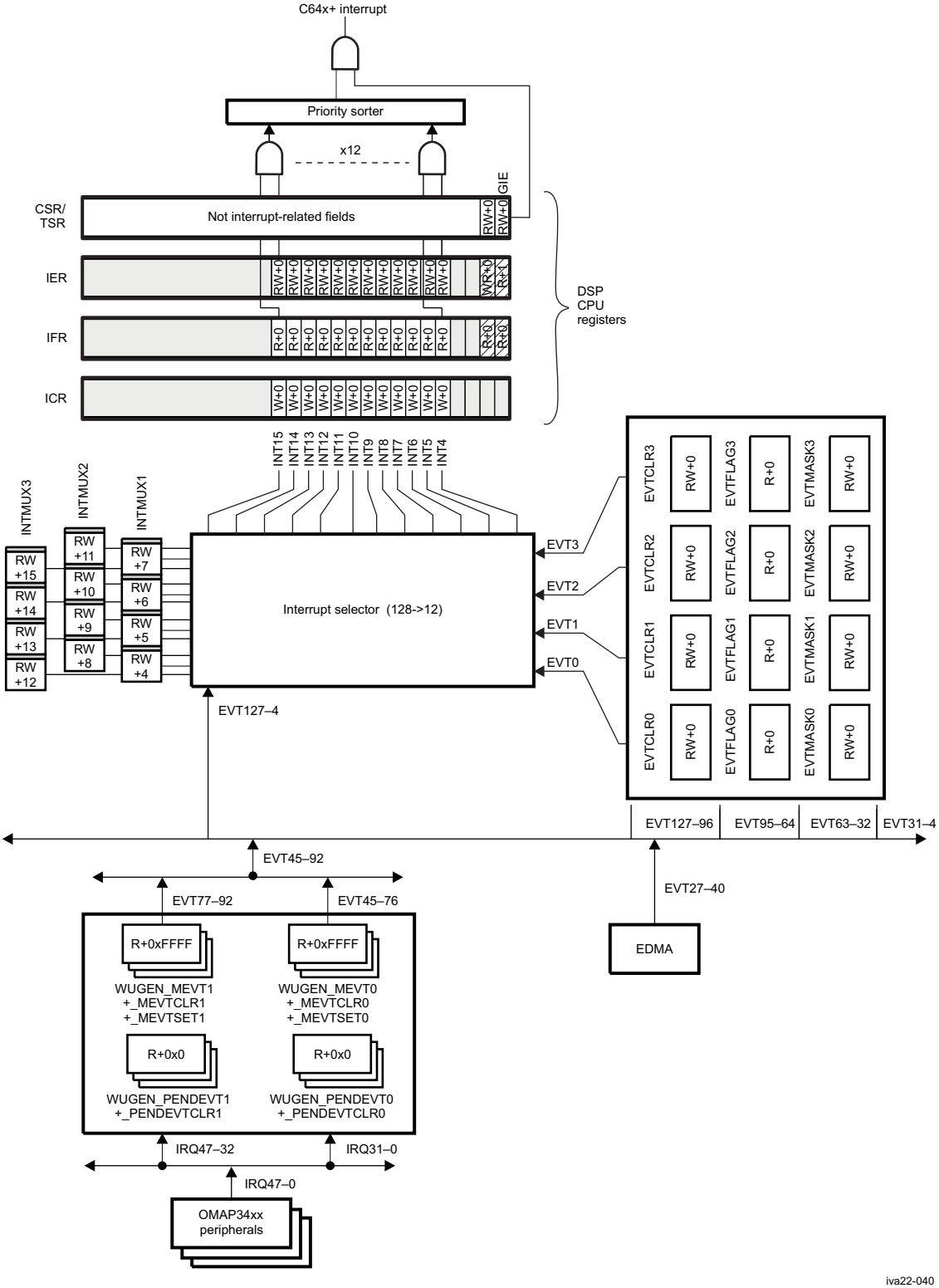
14.4.7.1 Interrupt Flow in IVA2.2 Subsystem

The DSP megamodule interrupt controller (IC) detects, combines, and routes up to 128 system events (internal and external) to the 12 DSP CPU interrupt lines. For more information about interrupt mapping of the IVA2.2 subsystem (internal and external interrupts), see Table 14-3.

In addition to performing handshaking for chip-level wake-up sequencing for waking from static power-down, the DSP megamodule WUGEN module internally routes the interrupt request to the IC module. This occurs because the WUGEN module is the only DSP megamodule module in the CORE power domain to enable the powered-off DSP subsystem to be awakened after the DSP receives an interrupt from any device peripherals.

Figure 14-39 shows the IVA2.2 subsystem interrupt flow with the main WUGEN and IC registers used for event generation.

Figure 14-39. IVA2.2 Interrupt Flow



Note: The CSR/TCR, IER, IFR, and ICR registers belong to the DSP CPU block (C64x+). For more information, see the C64x+ documentation (Section 14.3.1.8).

14.4.7.2 Event Combined Programming Sequence

In addition to generating a combined interrupt based on programmable event combinations, the event combiner provides a masked view of the event flag registers. By reading the masked event flag (IVA_IC.MEVTFLAGi where i = {0 to 3}) registers, the DSP CPU sees only the event flags pertaining to the corresponding combined event (EVTx with x = {0, 1, 2, 3}).

When servicing a combined interrupt, perform the following steps:

1. Read the IVA_IC.MEVTFLAGi register corresponding to the combined event EVTx (where i = {0 to 3}).
2. Check the pending events.
3. Write the value of the IVA_IC.MEVTFLAGi register into the IVA_IC.EVTCLRx register (where i = {0 to 3}).
4. Service the received interrupts.
5. Repeat steps 1 through 4 until IVA_IC.MEVTFLAGi = 0x0 (where i = {0 to 3}).

This clears only those events combined on EVTx. Any events masked in the IVA_IC.EVTMASKi register (where i = {0 to 3}) do not need to be cleared if set in the IVA_IC.EVTFLAGi register (where i = {0 to 3}) (they can generate an exception or are used as event outputs).

Before returning, the DSP CPU should repeat steps 1 through 4 until no pending events are found. This ensures that any events received during the ISR are captured. If an event EVTx is received at the same time that its flag is being cleared in the IVA_IC.EVTCLRi register (where i = {0 to 3}), it does not clear. Repeating steps 1 through 4 ensures that no events are missed.

14.4.7.3 Event <-> Interrupt Mapping Programming Sequence

The INTC allows programming independently which of the 128 inputs events is mapped to each DSP CPU interrupt by writing the event number in the bit field corresponding to the CPU interrupt in the IC.INTMUXj registers (where j = {1 to 3}).

Example:

```
/* ---------------------------------------------------
          */
/*evtTable has the 12 evt <-> CPU interrupt mapping
          */
/* ---------------------------------------------------
          */
evtTable[0] = 55; // Mailbox event highest priority
evtTable[1] = 61; // McBSP1TX event
[...]
evtTable[11] = 29; // EDMA3 gbl completion event lowest
          priority
for(I=0; i<12; I++) {// for each CPU maskeable
          interrupt
INTMUX(I >> 2 + 1) |= (evtTable[i] & 0x7F) <<
          ((I & 0x3) << 3);
}
```

14.4.7.4 Interrupt Exception Programming Sequence

The INTC can generate a system event (INTERR) that is internally routed to system event input EVT96. This event is generated when a DSP CPU interrupt is dropped (that is, when a DSP CPU interrupt is received while the interrupt flag is already set in the DSP CPU).

This can inform the user of possible problems in the software code, such as whether interrupts were disabled for an extended period of time, or whether pipelined (noninterruptible) code sections were too long.

Note: Because the interrupt drop detection logic is in the CPU, only interrupts sourced from a single system event can be detected. The dropping of individual combined events is not possible, although dropping the output of an event combiner is possible. Only the first dropped interrupt detected is reported by the INTERR event.

The IVA_IC.INTXSTAT register holds the ID of the CPU interrupt and the system event number of the dropped event. By setting the IVA_IC.INTXCLR[0] CLEAR bit to 1, the software user resets the IVA_IC.INTXSTAT register to 0.

When servicing the interrupt exception, the CPU performs three steps:

1. Reads the IVA_IC.INTXSTAT register
2. Checks the error condition
3. Clears the error through the IVA_IC.INTXCLR register

The dropped events that generate the INTERR event (EVT96) can be qualified by a mask register. CPU interrupts that are to be ignored by the drop detection hardware can be masked in the IVA_IC.INTDMASK register.

14.4.7.5 Interrupt Controller Basic Programming Model for Power Down of IVA2.2 Subsystem

When going to a low-power state not involving IVA2.2 logic power off, there are no special software settings to perform. In particular, the noncombined interrupts can be used without any specific handling for that power transition. To run the procedure for a correct transition to power-down state not involving logic-off, the user must perform the following sequence:

1. Ensure that all wake-up events are correctly mapped to enabled DSP CPU interrupts, and are unmasked in combined event registers (if combined events are used).
2. Clear associated bits in the WUGEN_MEVT0 and WUGEN_MEVT1 register to unmask wake-up interrupts.

Note: Software must not unmask in the WUGEN module an event that is not correctly mapped to an enabled DSP CPU interrupt. If that event is triggered, the IVA2.2 subsystem exits the power-down state but does not wake up.

The user must program the following procedure when transitioning to the (logic) power-off state (OFF state for device):

1. Ensure that all wake-up events are correctly mapped to enabled DSP CPU interrupts, and are unmasked in combined event registers (if combined events are used).
2. Clear correct bits in the WUGEN_MEVT0 and WUGEN_MEVT1 registers to unmask wake-up interrupts.
3. Save necessary context (all except interrupt-related registers).

Note: Steps 3 through 5 are automatic, but interruptible. If a wake-up interrupt occurs at this stage, the transition to power down is canceled and the next transition restarts from the beginning. Software tip: A software variable (semaphore) can be set in Step 3, modified in all calls of the wake-up interrupt ISR routine, and checked before executing the IDLE instruction in Step 5. For a complete description of the DSP CPU ISR register, see the C64x+ documentation (Section 14.3.1.8).

4. Save the interrupt configuration in:
   a. IVA_IC.INTMUXj register (where j = {1 to 3})
   b. IVA_IC.EVTMASKi register (optional, where i = {0 to 3})
   c. IVA_IC.INTDMASK register (optional)
   d. DSP CPU IER register. See the C64x+ documentation for IER register description (Section 14.3.1.8).
5. Execute the DSP IDLE instruction.

Note: Software must not unmask in the WUGEN module an event that is not correctly mapped to an enabled DSP CPU interrupt. If that event is triggered, the IVA2.2 subsystem exits the power-down state but does not wake up.

14.4.7.6 Interrupt Controller Basic Programming Model for Power On of IVA2.2 Subsystem

Noncombined events can be lost after power on wakeup of the DSP megamodule module. To avoid this, one solution is to replay the DSP CPU interrupt after restoring the interrupt selector and event combiner context.

- Programming model at boot time

There is no specific software setting needed to use combined/noncombined events during normal operation. However, at boot time, the event can be captured in the event-combiner event flag and not propagated to DSP. For this purpose, the following sequence must be performed:

1. The IVA2.2 logic ensures that interrupts are only presented to DSP megamodule after:
   a. The DSP megamodule module has its clock.
   b. The DSP megamodule module is correctly reset.

   Note: Steps a and b ensure that the DSP megamodule module cannot miss a dropped event.
2. From the time the pre-idle sequence is started to the time when the IVA2.2 context is fully restored and operational after wakeup, it is assumed that only external (from device peripherals) interrupts can occur. DSP megamodule internal interrupts can result only from an application software side-effect; the application is stopped when entering the pre-idle sequence and is restarted only after IVA2.2 context is fully restored and operational (with special attention to memory protection). EDMA interrupts are inactive at this stage, as the EDMA module is completely reset after power up, and a software action is required to make the EDMA interrupts active again.
3. All device peripheral interrupts are level and are kept asserted until the software acknowledges the interrupt by writing 1 to the interrupt status register (DSP CPU ISR) bit corresponding to the enabled asserted event(s). The device peripheral must keep an event flag to track missed interrupts. For a complete description of the DSP CPU ISR register, see the C64x+ documentation (Section 14.3.1.8).
4. There is no specific requirement to configure or restore interrupt mapping, except having the interrupts globally disabled during the sequence:
   a. Set the DSP CPU TSR[0] or CSR[0] GIE bit to 0 to disable all interrupts except the reset interrupt and NMI (nonmaskable interrupt). The GIE bit is the same physical bit in the TSR and CSR registers. For a complete description of these registers, see the C64x + documentation (Section 14.3.1.8).
   b. Remap interrupts by configuring the IVA_IC.INTMUXj registers (where j = {1 to 3}).
   c. Set the DSP CPU TSR[0] or CSR[0] GIE bit to 1 to enable all DSP CPU interrupts.

The DSP CPU software recognizes which degree of power state the C64x reaches when executing the IDLE instruction, because the PRCM module is under C64x software control. Therefore, the DSP CPU software can correctly skip some unnecessary parts of the pre-idle routines.

- State after reset for DSP CPU and IVA2.2 IC registers:
  - The DSP CPU IER register value is 0x0.
  - The DSP CPU IFR register value is 0x0 because the interrupt selector default mapping is routing only emulation and unmapped events.
  - The IVA_IC.INTMUXj registers (where j = {1 to 3}): The default mapping is to route EVTx (with x = {0 to 3}) on INTy (with y = {4 .. 15}), that is, emulation and nonmapped events.
  - The IVA_IC.EVTMASKi registers (where i = {0 to 3}) value is 0x0 (all events are unmasked). These registers values are don't care values because of the default interrupt selector mapping (where iVA_IC.INTMUXj registers where j = {1 to 3}).
  - The IVA_IC.EVTFLAGi registers (where i = {0, 1, 2, 3}): These registers log the possible wake-up interrupt(s) (if more than one). At this stage, a new wake-up interrupt from device peripherals cannot be missed, because interrupts are level in the device, waiting for a software acknowledge (the user must clear the interrupt in the module to deassert the interrupt line).

For a complete description of the DSP CPU IER and IFR registers, see the C64x+ documentation (Section 14.3.1.8).

- State after reset for WUGEN registers: two cases depending on the reset type:
  - In case of cold power-on reset (the CORE power domain was in OFF state), the IVA_WUGEN.WUGEN_MEVT0 register value is 0xFFFFFFFF and the IVA_WUGEN.WUGEN_MEVT1 register value is 0xFFFF, meaning that all interrupts are masked.
  - In case of warm power-on reset (the CORE power domain was in ACTIVE state), the WUGEN_MEVT0 and WUGEN_MEVT1 registers value is the previous programmed one, enabling only wake-up interrupts.
- Programming model for a correct warm-reset:

The software user must replay the noncombined events required for wakeup (if any):

1. Globally mask interrupts by setting the DSP CPU TSR[0] or CSR[0] GIE bit to 0 (already done at boot time).
2. Restore the interrupt selector configuration:
   a. IVA_IC.INTMUXj register (where j = {1 to 3})
   b. IVA_IC.EVTMASKi register (optional, where i = {0 to 3})
   c. IVA_IC.INTDMASK register (optional)
3. Replay the noncombined event captured in IVA_IC.EVTFLAGi (where i = {0 to 3}):
   For each DSP CPU interrupt I = (4 to 15)
   {
   a. Grab mapped event by setting the IVA_IC.INTMUXj INTSELi[6:0] field (where j = {1 to 3}).
   b. Check if the event is combined (EVT0...3) or not combined (EVT4...127).
   c. If combined, exit the loop and go to next loop iteration.
   d. If noncombined:
      i. Check whether the event is pending in the IVA_IC.EVTFLAGi EFy bit (where i = {0 to 3} and y = {0 to 127}).
      ii. If not pending, exit the loop and go to the next loop iteration.
      iii. If pending, set the associated IFi bit in the DSP CPU IFR register.
   }

Note: Software must ensure that an event is enabled only once in the DSP megamodule interrupt selector/combiner: If a noncombined event is mapped to an enabled DSP CPU interrupt, the associated combined event is masked in the associated IVA_IC.EVTMASKi (where i = {0 to 3}) register (and/or the combined event is not mapped to the DSP CPU interrupt). Reciprocally, if a combined event is mapped to a DSP CPU interrupt, all the unmasked events in the associated IVA_IC.EVTMASKi (with i = {0, 1 to 3}) register are not mapped as noncombined events to an enabled DSP CPU interrupt (through the DSP CPU IER register).

Software program example for Step 3:

```
/* ---------------------------------------------------
            */
/* Replays IFR bits associated to noncombined events
            */
/* Assumes interrupts globally disabled (GIE bit set to
            0)*/
/* Assumes IVA_IC.INTMUXj() and IVA_IC.EVTFLAGi() are access
            macros to DSP megamodule IC registers */
/* ---------------------------------------------------
            */
myIPR = IPR; // save IPR
for(I=0; i<12; I++) { // for each CPU maskable
            interrupt
myEvt = ( INTMUX( I >> 2 + 1) >> ( (I & 0x3)
            << 3 ) ) & 0x7F;
```

```
if(myEvt >= 4) { // noncombined event
if( (EVTFLAG(myEvt >> 5 ) >> (myEvt & 0x1F )
            ) & 0x1 ) {
myIPR |= (1<<(I+4));
}
}
}
IPR=myIPR; // update IPR register
/* ------------------------------------------------
            */
/* Interrupts can be globally re-enabled from that
            point*/
/* ------------------------------------------------
            */
```

4. Restore the CPU interrupt configuration by setting the DSP CPU IER register accordingly.
5. Restore the IVA2.2 context (except saved return-PC).
6. Globally enable the interrupts by setting the DSP CPU TSR[0] or CSR[0] GIE bit to 1.
7. Branch to saved return-PC.

For a complete description of the DSP CPU TSR, CSR, IER, or IFR registers, see the C64x+ documentation (Section 14.3.1.8).

- Procedure for a correct cold reset

1. Globally mask interrupts by setting the DSP CPU TSR[0] or CSR[0] GIE bit to 0 (already done at boot time).
2. Restore the interrupt selector configuration:
   a. IVA_IC.INTMUXj register (where j = {1 to 3})
   b. IVA_IC.EVTMASKi register (optional, with i = {0 to 3})
   c. IVA_IC.INTDMASK register (optional)
3. It is unnecessary to replay events, because they are masked in WUGEN_MEVT0 and WUGEN_MEVT1 registers (default WUGEN module configuration). This can be done to simplify the boot sequence without discriminating between cold and warm reset.
4. Restore the CPU interrupt configuration by setting the DSP CPU IER REGISTER accordingly.
5. Restore the WUGEN module context. This can be forced even in warm reset to simplify the boot sequence without discriminating between cold and warm reset.

Note: If Steps 3 and 5 are performed systematically, the boot code can be shared between warm and cold reset boot without programming two distinct software boot codes.

6. Restore the rest of the IVA2.2 subsystem context (except saved return-PC).
7. Globally enable the interrupts by setting the DSP CPU TSR[0] or CSR[0] GIE bit to 1.
8. Branch to saved return-PC.

14.4.7.7 Video and Sequencer Module interrupt Handling

14.4.7.7.1 Sequencer Interrupt

The sequencer uses a single FIQ interrupt. This single interrupt is driven by all the video accelerator interrupts, DSP EDMA interrupts, DSP megamodule soft interrupts, and error interrupts. Registers are available to manage these interrupts in the sequencer.

Before servicing an interrupt, the sequencer ISR must:

1. Identify the source event(s) for the interrupt line assertion
2. For each source event:
   a. Clear the interrupt at the source (in the module interrupt clear register)

b. Clear the interrupt at the second-level (sequencer) interrupt controller
c. Service the interrupt event

For more information about sequencer interrupts, see Table 14-6.

14.4.7.7.2 DSP Megamodule Interrupt

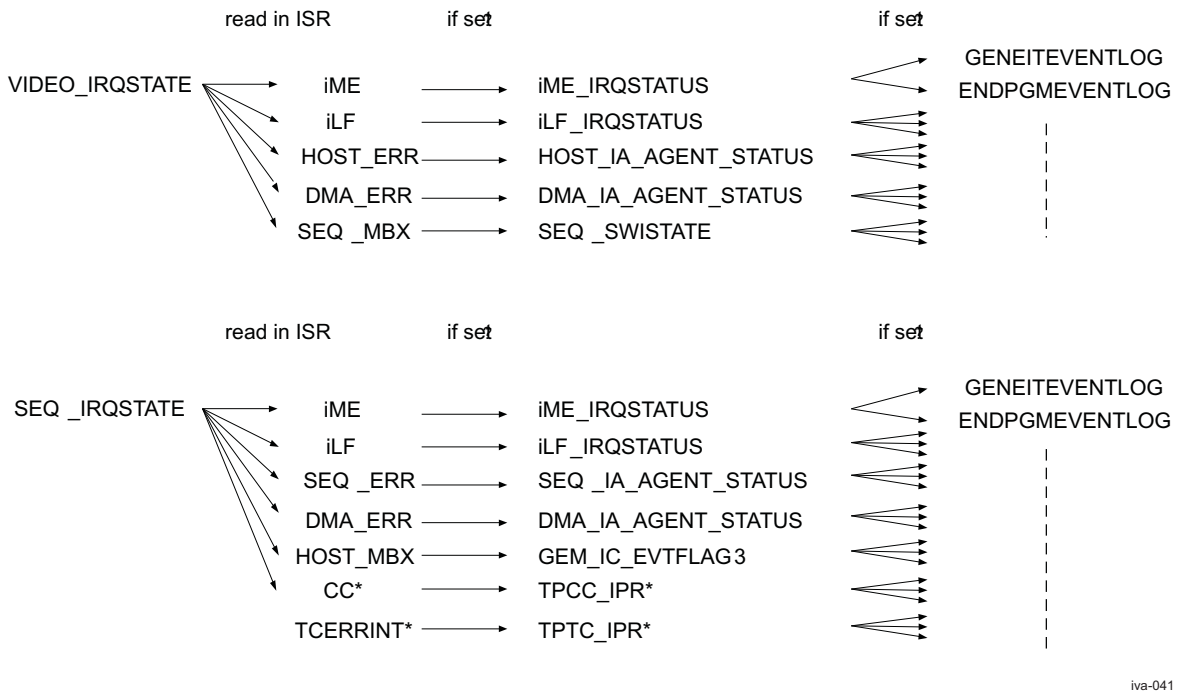
A single interrupt line is used per processor to gather all events that might require an interrupt of the DSP megacell. This interrupt line is shared by all the modules responsible for generating an interrupt request to the associated processor (iME, iLF, DSP megacell initiator error, DMA initiator error, and sequencer soft interrupt). In turn, each of these modules has a single interrupt request output that is shared by all events internal to the module.

Before servicing an interrupt, the processor (DSP megacell) ISR must:

1. Identify the source event(s) for the interrupt line assertion
2. For each source event:
   a. Clear the interrupt at the source (in the module interrupt clear register)
   b. Clear the interrupt at the second-level interrupt controller
   c. Clear the interrupt at the DSP interrupt controller (if combined interrupt is used, see Section 14.4.7.2, *Event Combined Programming Sequence*, for details)
   d. Service the interrupt event
3. Acknowledge the interrupt to the processor

Step 1 is done hierarchically through reading the IVA.VIDEOSYSC_IRQSTATE register and, depending on which bit is set, through reading of the IVA.module>_IRQSTATUS registers (or equivalent) (see Figure 14-40). Step 2 is done hierarchically, as well, but in the reverse order, using IVA.module>_IRQCLEAR (or equivalent) first and then IVA.VIDEOSYSC_IRQCLR (IVA.SEQ_IRQCLR, respectively).

Figure 14-40. Process of Identifying Source Event of an Interrupt



14.4.8 Memory Management

14.4.8.1 External Memory

14.4.8.1.1 Cacheability

The L1P, L1D, and L2 cache sizes are all 0K byte after reset.

- The user enables the L2 cache by writing 0x2 (maximum cache is 64K bytes) to the IVA_XMC.L2CFG register.
- However, this is not sufficient, because the region external to the IVA2.2 is configured as a noncached area after reset. The user must configure the external area to be cached by setting the IVA_XMC.MARi[0] PC bit to 1, i referring to the index of the related 16-MB page (starting from i = 16 for the first external page at address 0x1000 0000).

The MAR settings only control the L2 and L1D cache's response to CPU data and program fetches. L1P caches the memory range regardless of the values of the MAR bits.

14.4.8.1.2 Virtual Addressing

The device embeds two instances of MMU: one instance is camera MMU (also named MMU1) dedicated to the camera subsystem; the other instance is IVA2.2 MMU (also named MMU2) used by the IVA2.2 subsystem. For more information about MMU2 software settings at IVA2.2 boot, see Section 14.4.1.2, *Example of IVA2.2 Boot*.

For a complete programming guide of IVA2.2 MMU, see Chapter 8, *Memory Management Units*.

14.4.8.2 Internal Memory

14.4.8.2.1 Memory Protection

The DSP megacell memory protection divides the memory map into pages and defines a per-page permission structure. This permission structure is in the MPPA registers: IVA_XMC.L1PMPPAk, IVA_XMC.L1DMPPAk, IVA_XMC.L2MPPAj for L1P, L1D, L2 memory protection, and TPCC_MPPAG, TPCC_MPPAj for EDMA memory protection. Figure 14-41, Figure 14-42, and Figure 14-43 show which register corresponds to each page permission structure.

Figure 14-41. L1P Memory Protection Registers

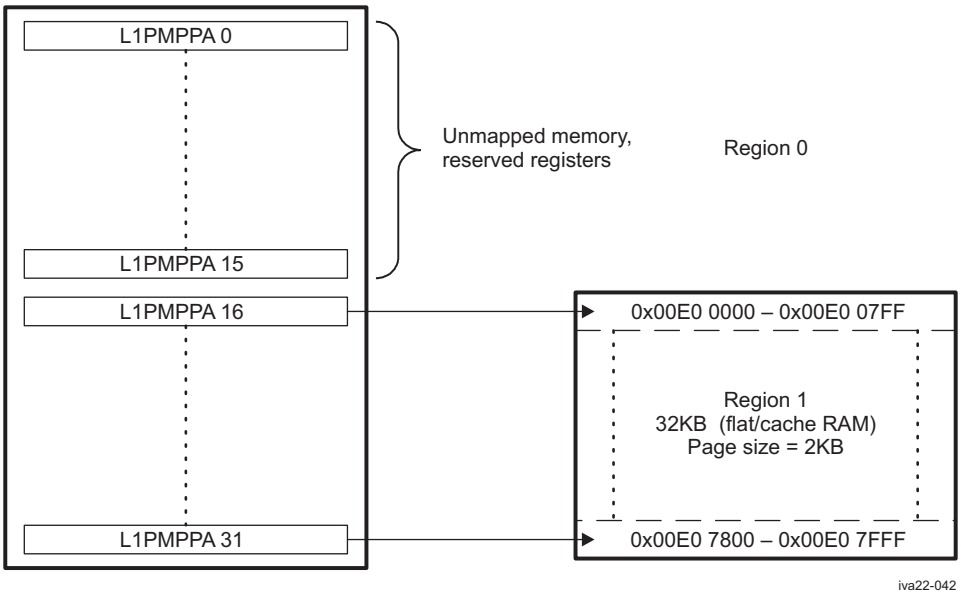


Figure 14-42. L1D Memory Protection Registers

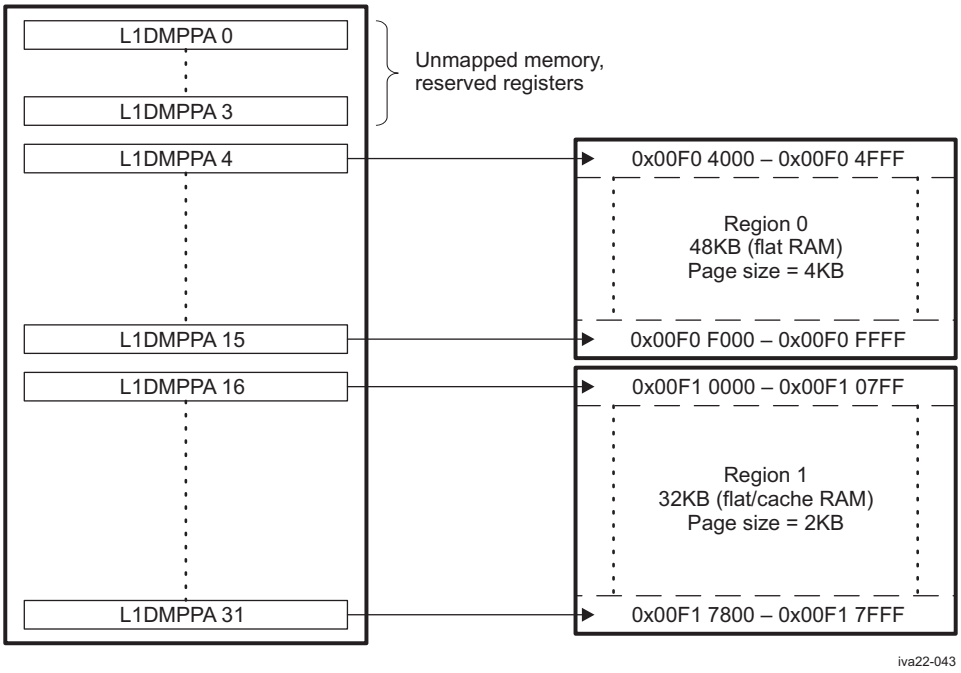
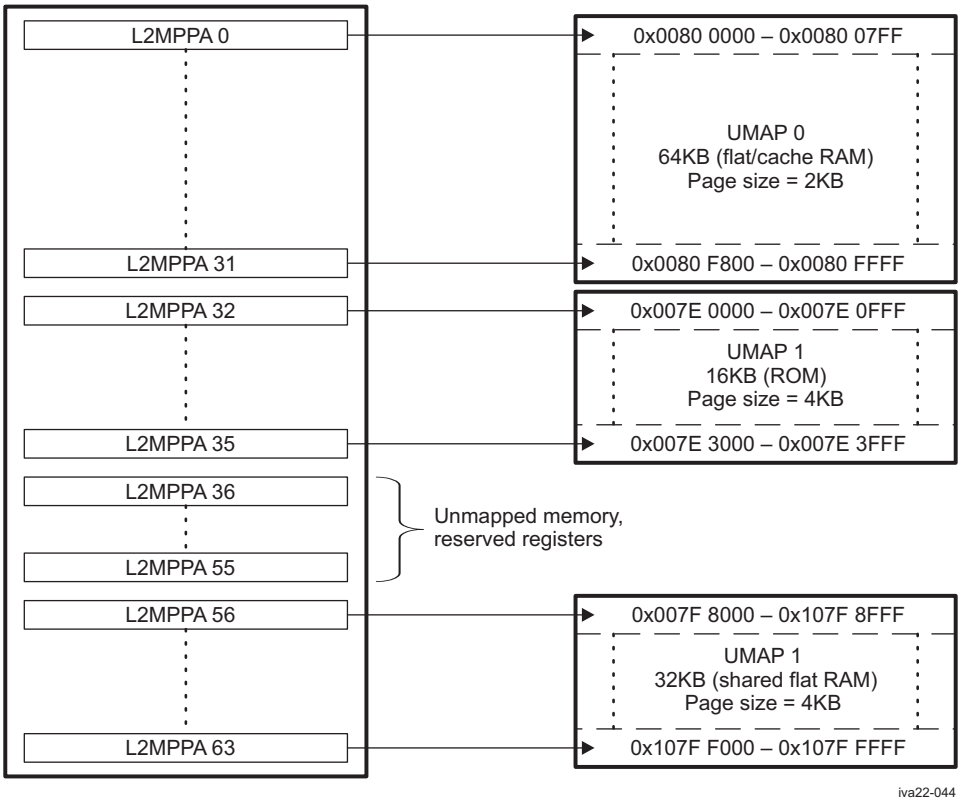


Figure 14-43. L2 Memory Protection Registers



Note: When memory is used as cache, all its corresponding MPPA registers should be set to 0x0000.

Table 14-96 shows which register corresponds to each memory address for megacell internal memory.

Table 14-96. IVA2.2 Megamodule Memory Protection Page Registers

Register Number	L1PMPPAk, Address Range	L1DMPPAk, Address Range	L2MPPAj, Address Range
0	N/A(1)	N/A	0x0080 0000 - 0x0080 07FF
1	N/A	N/A	0x0080 0800 - 0x0080 0FFF
2	N/A	N/A	0x0080 1000 - 0x0080 17FF
3	N/A	N/A	0x0080 1800 - 0x0080 1FFF
4	N/A	0x00F0 4000 - 0x00F0 4FFF	0x0080 2000 - 0x0080 27FF
5	N/A	0x00F0 5000 - 0x00F0 5FFF	0x0080 2800 - 0x0080 2FFF
6	N/A	0x00F0 6000 - 0x00F0 6FFF	0x0080 3000 - 0x0080 37FF
7	N/A	0x00F0 7000 - 0x00F0 7FFF	0x0080 3800 - 0x0080 3FFF
8	N/A	0x00F0 8000 - 0x00F0 8FFF	0x0080 4000 - 0x0080 47FF
9	N/A	0x00F0 9000 - 0x00F0 9FFF	0x0080 4800 - 0x0080 4FFF
10	N/A	0x00F0 A000 - 0x00F0 AFFF	0x0080 5000 - 0x0080 57FF
11	N/A	0x00F0 B000 - 0x00F0 BFFF	0x0080 5800 - 0x0080 5FFF
12	N/A	0x00F0 C000 - 0x00F0 CFFF	0x0080 6000 - 0x0080 67FF
13	N/A	0x00F0 D000 - 0x00F0 DFFF	0x0080 6800 - 0x0080 6FFF
14	N/A	0x00F0 E000 - 0x00F0 EFFF	0x0080 7000 - 0x0080 77FF
15	N/A	0x00F0 F000 - 0x00F0 FFFF	0x0080 7800 - 0x0080 7FFF
16	0x00E0 0000 - 0x00E0 07FF	0x00F1 0000 - 0x00F1 07FF	0x0080 8000 - 0x0080 87FF
17	0x00E0 0800 - 0x00E0 0FFF	0x00F1 0800 - 0x00F1 0FFF	0x0080 8800 - 0x0080 8FFF
18	0x00E0 1000 - 0x00E0 17FF	0x00F1 1000 - 0x00F1 17FF	0x0080 9000 - 0x0080 97FF
19	0x00E0 1800 - 0x00E0 1FFF	0x00F1 1800 - 0x00F1 1FFF	0x0080 9800 - 0x0080 9FFF
20	0x00E0 2000 - 0x00E0 27FF	0x00F1 2000 - 0x00F1 27FF	0x0080 A000 - 0x0080 A7FF
21	0x00E0 2800 - 0x00E0 2FFF	0x00F1 2800 - 0x00F1 2FFF	0x0080 A800 - 0x0080 AFFF
22	0x00E0 3000 - 0x00E0 37FF	0x00F1 3000 - 0x00F1 37FF	0x0080 B000 - 0x0080 B7FF
23	0x00E0 3800 - 0x00E0 3FFF	0x00F1 3800 - 0x00F1 3FFF	0x0080 B800 - 0x0080 BFFF
24	0x00E0 4000 - 0x00E0 47FF	0x00F1 4000 - 0x00F1 47FF	0x0080 C000 - 0x0080 C7FF
25	0x00E0 4800 - 0x00E0 4FFF	0x00F1 4800 - 0x00F1 4FFF	0x0080 C800 - 0x0080 CFFF
26	0x00E0 5000 - 0x00E0 57FF	0x00F1 5000 - 0x00F1 57FF	0x0080 D000 - 0x0080 D7FF
27	0x00E0 5800 - 0x00E0 5FFF	0x00F1 5800 - 0x00F1 5FFF	0x0080 D800 - 0x0080 DFFF
28	0x00E0 6000 - 0x00E0 67FF	0x00F1 6000 - 0x00F1 67FF	0x0080 E000 - 0x0080 E7FF
29	0x00E0 6800 - 0x00E0 6FFF	0x00F1 6800 - 0x00F1 6FFF	0x0080 E800 - 0x0080 EFFF
30	0x00E0 7000 - 0x00E0 77FF	0x00F1 7000 - 0x00F1 77FF	0x0080 F000 - 0x0080 F7FF
31	0x00E0 7800 - 0x00E0 7FFF	0x00F1 7800 - 0x00F1 7FFF	0x0080 F800 - 0x0080 FFFF
32	-	-	0x007E 0000 - 0x007E 0FFF
33	-	-	0x007E 1000 - 0x007E 1FFF
34	-	-	0x007E 2000 - 0x007E 2FFF
35	-	-	0x007E 3000 - 0x007E 3FFF
36 - 55	-	-	N/A
56	-	-	0x007F 8000 - 0x007F 8FFF
57	-	-	0x007F 9000 - 0x007F 9FFF
58	-	-	0x007F A000 - 0x007F AFFF
59	-	-	0x007F B000 - 0x007F BFFF
60	-	-	0x007F C000 - 0x007F CFFF
61	-	-	0x007F D000 - 0x007F DFFF
62	-	-	0x007F E000 - 0x007F EFFF
63	-	-	0x007F F000 - 0x007F FFFF

(1) N/A = nonmapped memory. All corresponding registers should be considered reserved.

All these registers include three permission fields in a 16-bit permission entry in the memory protection page attribute MPPA register:

- Allowed ID field:

  Each requestor on the device has an N-bit code associated where it that identifies it for privilege purposes. This code, referred to as the VBUS PrivID, accompanies all memory accesses and DMAs/IDMAs on behalf of that requestor. That is, when a requestor triggers a DMA/IDMA transfer directly, either by writing to IDMA registers or by triggering the execution of a set of EDMA parameters, the corresponding DMA engine captures the PrivID of the requestor and provides that PrivID with the transfer.

  Each memory protection entry is associated with an 8-bit allowed-IDs field that indicates which requestors can access the given page. The memory protection hardware maps the PrivIDs of all possible requestors to bits in the allowed-IDs field in the memory protection entries. The allowed-IDs field discriminates among the various CPUs, non-CPU requestors, and the accesses of a given CPU to its own local memories:

  – AID0 through AID5 map small-numbered VBUS PrivIDs to allowed ID bits.
  – An additional allowed-ID bit, AIDX, captures accesses by higher-numbered PrivIDs.
  – The LOCAL bit treats CPU accesses to its local L1s and L2 specially. Only the L1 and L2 memory controllers implement the LOCAL bit.

  When set to 1, the AID bit grants access to the corresponding PrivID. When set to 0, the AID bit denies access to the corresponding requestor.

  PrivID assignments for bits AID0 through AIDX apply to:

  – CPU memory and IDMA memory: IVA_XMC.L1PMPPAk[15:9] for L1P memory, IVA_XMC.L1DMPPAk[15:9] for L1D memory, and IVA_XMC.L2MPPAj[15:9] for L2 memory
  – EDMA memory: TPCC_MPPAG[15:9] and TPCC_MPPAj[15:9] for EDMA block

  The AIDX bit maps to PrivIDs that do not have dedicated AID bits associated with them. This bit refers to external mastering peripherals, especially on devices with a large number of CPUs. If a given device must discriminate among external mastering peripherals, it can assign lower-numbered PrivIDs to these peripherals.

  For the EDMA module, the AIDX bit is the TPCC_MPPAj[9] and TPCC_MPPAj[9] EXT bits.

  The LOCAL bit governs DSP CPU accesses to its own local L1 and L2 memories. It is defined in IVA_XMC.L1PMPPAk[8] for L1P memory, in IVA_XMC.L1DMPPAk[8] for L1D memory, and in IVA_XMC.L2MPPAj[8] for L2 memory. It is not defined for EDMA.

  With respect to the allowed ID field, the DSP megamodule treats all remote CPU accesses to DSP megamodule memories identically to DMA accesses from that remote CPU. Therefore, DSP megamodule uses the PrivID to consult the corresponding AID bit when considering remote accesses.

  However, the DPS megamodule module never compares local accesses to L1 and L2 memory against the AID bits in the allowed ID field. Instead, it consults the LOCAL bit to determine whether the CPU can access its local memory.

  With this scheme, a software program can designate a page as direct CPU access only by setting the LOCAL bit to 1 and setting all other allowed IDs to 0. Conversely, a software program can designate a page as DMAs issued from this CPU only by setting its AID bit and clearing the LOCAL bit. Such a setting is limited, but it can be useful in a sensitive environment, or in the context of a device driver.

- Request-Type Based Permissions field:

  The DPS megamodule memory protection model defines three fundamental functional access types: read, write, and execute. Read and write refer to data accesses originating through the load/store units on the DSP CPU or through the DMA/IDMA engines. Execute refers to accesses associated with program fetch.

  The DSP megamodule memory protection model allows control of read, write, and execute permissions independently for both user and supervisor mode. This results in the 6-bit permission field shown in Table 14-97.

Table 14-97. Request-Type Access Controls

Bit Name	Description
SR	Supervisor can read
SW	Supervisor can write

Table 14-97. Request-Type Access Controls (continued)

Bit Name	Description
SX	Supervisor can execute (1)
UR	User can read
UW	User can write
UX	User can execute (1)

(1) IVA_XMC.L1DMPPAXX do not implement these bits.

For each bit, writing 1 permits the access type, and writing 0 denies it. For instance, setting the UX bit to 1 means that user mode can execute from the given page. The DSP megamodule allows specifying all six of these bits separately. For L1P, L1D, and L2 memories, and for EDMA, these bits are defined in IVA_XMC.L1PMPPAk[5:0], IVA_XMC.L1DMPPAk[5:0], IVA_XMC.L2MPPAj[5:0], and TPCC_MPPAG[5:0] and TPCC_MPPAj[5:0], respectively.

Note: IVA_XMC.L1DMPPAk do not implement the SX and UX bit, because execution is not possible from L1D memory. Therefore, these bits are reserved and should not be used in these registers.

- Other registers

  The memory protection block records the address of the fault in the peripheral memory protection fault address register (MPFAR). It records the rest of the information about the fault in the peripheral memory protection fault status register (MPFSR); this register is formatted similarly to the MPPA. Software can write to the memory protection fault command register (MPFCR).

  The MPFAR corresponds to the following registers: IVA_XMC.L1PMPFAR for L1P memory, IVA_XMC.L1DMPFAR for L1D memory, IVA_XMC.L2MPFAR for L2 memory, IVA_IDMA.ICFGMPFAR for IDMA, and TPCC_MPFAR for EDMA.

  The MPFSR corresponds to the following registers: IVA_XMC.L1PMPFSR for L1P memory, IVA_XMC.L1DMPFSR for L1D memory, IVA_XMC.L2MPFSR for L2 memory, IVA_IDMA.ICFGMPFSR for IDMA, and TPCC_MPFSR for EDMA.

  Using the MPFSR register, a memory protection fault can be decoded as follows by software:

  – If the LOCAL status bit is set to 1, the request was a local DSP CPU request to its own memories. Otherwise (if the LOCAL status bit is set to 0), the VBUS ID of the faulting requestor is in the MPFSR[15:9] field.
  – The value of the access type field (SR, SW, SX, UR, UW, UX) indicates the type of access that was at fault.

  The MPFAR and MPFSR store information for only one fault. The hardware block holds the fault information until the software clears it by writing to MPFCR.

  The user clears the recorded fault by setting the MPFCR[0] MPFCLR bit to 1. Writing 1 to this bit clears both MPFAR and MPFSR registers. The MPFAR and MPFSR read-only registers do not respond to writes. Once the user clears the fault, the hardware records the next protection violation, and signals an exception when it occurs. Writing 1 to any other bit of the MPFCR has no effect on the memory protection registers. Writing 0 to the MPFCR[0] MPFCLR bit also has no effect.

  The MPFCR corresponds to the following registers: IVA_XMC.L1PMPFCR for L1P memory, IVA_XMC.L1DMPFCR for L1D memory, IVA_XMC.L2MPFCR for L2 memory, IVA_IDMA.ICFGMPFCR for IDMA, and TPCC_MPFCR for EDMA.

- Events generated to DSP megamodule IC block

  Internal events are generated to inform the DSP megamodule module for memory protection:

  – CCMPINT (EVT28) is generated for a TPCC memory protection interrupt.
  – SYS_CMPA (EVT 119) is generated in case of a SYS CPU memory protection fault.
  – PMC_CMPA (EVT120) and PMC_DMPA (EVT121) are generated in case of CPU and DMA memory protection faults, respectively, on PMC.
  – DMC_CMPA (EVT122) and DMC_DMPA (EVT123) are generated in case of CPU and DMA memory protection faults, respectively, on DMC.
  – UMC_CMPA (EVT124) and UMC_DMPA (EVT125) are generated in case of CPU and DMA

memory protection faults, respectively, on UMC.
– PMC_CMPA (EVT126) is generated in case of a CPU memory protection fault on EMC.

The AID of the DSP appears in the DNUM control register in the CPU. DNUM is the DSP core number register; it is used to identify which DSP subsystem accessed the shared resources. Because there is only one C64x+:

- GEM_DNUM[7:4] always equals 0x0.
- GEM_DNUM[3:0] equals 0x0 because DSP is labeled as CPU0 in the single core system.

Thus, the possible requestors are:

- C64x
- IDMA
- EDMA
- Any device L3 initiator (through the IVA2.2 slave port)

The PrivID (relative to the DNUM register) equals 0. The ID of IVA DMA accesses is always 0x0, because the IVA DMA inherits ID from DSP megamodule. All accesses on the IVA2 slave port to local memories are seen by DSP megamodule as accesses from an external (non-DSP megamodule) initiator.

Thus, the hard-mapped memory protection IDs are:

- AID0, used for IVA DMA
- AID1 to AID5 (not used)
- AIDX, used by the slave port (L3 initiators)
- LOCAL bit, used by DSP loads, stores, and program fetches to memory attached directly to DSP megamodule

14.4.8.2.2 Bandwidth Management

The bandwidth management scheme can be summarized as weighted-priority-driven bandwidth allocation. Each requestor (EDMA, IDMA, CPU, etc.) is assigned a priority level on a per-transfer basis. The programmable priority level has a single meaning throughout the system: there is a total of nine priority levels, where priority 0 is highest and priority 8 is lowest priority. When requests for a single resource contend, access is granted to the highest priority requestor. When contention occurs for multiple successive cycles, a contention counter ensures that the lower priority requestor gets access to the resource every one out of *n* arbitration cycles, where *n* is programmable through the MAXWAIT field. For each identified requestor, a corresponding MAXWAIT field controls the maximum number of cycles that a request can be blocked.

- CPUARB registers: CPU priority and MAXWAIT programming model

The DSP CPU-initiated transfers consist of two components:

1. Program fetch transfers are issued by the CPU to the PMC, and the resulting L1P cache coherency operations (such as alloc/evict) are then issued to the UMC.
2. Data load/store transfers are issued by the CPU to the DMC, and the resulting L1D cache coherency operations (such as alloc/evict/long distance accesses) are then issued to the UMC.

Both program and data requests use the CPUARB values to define the maximum wait time (CPUARB[5:0] MAXWAIT) and priority (CPUARB[18:16] PRI). The effect of the CPUARB values is not just local to PMC or DMC. Priority/maxwait time applies to DMC/PMC cache transactions throughout the DSP megamodule and controls arbitration at each relevant subblock within DSP megamodule. The priority (PRI) and MAXWAIT field is programmed locally in the UMC, DMC, and EMC blocks, through the IVA_UMC.CPUARBU, IVA_DMC.CPUARBD, and IVA_IDMA.CPUARBE PRI and MAXWAIT fields, respectively.

The default value of the CPUARB[18:16] PRI field is set so that CPU transactions are second to highest in the system. This should be a relatively typical value used in most systems, which results in the DSP CPU being granted highest priority most of the time, but a McBSP-type peripheral (typically programmed as the highest priority transfer for sDMA requests) can interrupt the DSP CPU transfers on a nearly immediate basis.

Note: CPU priority is software-programmable, but it is considered static (1-time-programmable) for bandwidth management. That is, during normal operation, all DSP CPU transactions are of the same priority. There is no concept of priority inheritance for CPU-initiated transfers as a result of CPU transfer A set at priority X and CPU transfer B set at priority Y. Because CPUARB must be programmed by the CPU, and is therefore a run-time programmation, the effect of the new CPUARB value must take effect for future CPU transfers.

- UCARB registers - User coherence operations

  User coherence operations are broken into two types, block-oriented coherence and global coherence operations. The priority of these requests relative to other requests in the system varies as follows:

  – Global user coherence: Always highest priority
  – Block-oriented coherence: Always lowest priority

Because user coherence priority is fixed, the UCARB registers do not include a priority field. And because global user coherence operations are inherently highest priority, the MAXWAIT field programming applies only to block-oriented user coherence operations, not to global cache operations.

The UCARB[5:0] MAXWAIT field affects only the UMC (IVA_UMC.UCARBU register) and the DMC (IVA_DMC.UCARBD register). The priority values (being fixed) are assumed to be known at both the UMC and the DMC.

Note: The UCARB[5:0] MAXWAIT field (and the implied priorities) does not control the priority of coherence operations that result from DMA transactions or CPU transactions.

- IDMAARB registers - IDMA priority programming model

  IDMA supports two active transfers at any time through IDMA channel 0 (used for memory to/from configuration bus transfers) and IDMA channel 1 (used for memory-to-memory transfers). The IDMAARB[5:0]MAXWAIT field is used to determine the maximum wait time for IDMA transactions. The priority level is not programmed through the IDMAARB register. Instead, the priority level is programmed directly through the IDMA control registers. In summary, IDMA transfer priority is:

  – IDMA channel 0: Always highest priority
  – IDMA channel 1: Programmable priority: the IVA_IDMA.IDMA1_COUNT[31:29] PRI field

  For this reason, the IDMAARB registers do not include a priority field. The IDMAARB[5:0] MAXWAIT field affects not only the EMC (IVA_IDMA.IDMAARBE register) but also the UMC (IVA_UMC.IDMAARBU register) and the DMC (IVA_DMC.IDMAARBD register).

Note: IDMA channel 0 has no need for priority inheritance, because it is always the highest priority transfer in the system.

However, if an IDMA channel 1 transfer and an IDMA channel 0 transfer are pending at the same instant, but the IDMA channel 1 queue is blocked because of contention with another requestor, the IDMA channel 1 transfer should briefly inherit the priority of the IDMA channel 0 transfer (which is always 0x0) so that the pending dataphase on IDMA channel 1 can complete and the normal priority mechanism can take control.

- SDMAARB registers - Slave DMA priority programming model

The DSP megamodule slave DMA (SDMA) interface can support multiple active transfers at a time. The SDMAARB[5:0] MAXWAIT field controls the maximum wait time for all slave DMA transactions. The priority level is not programmed by the SDMAARB register. Instead, the priority level is dictated by the priority fields of the IVA_TPCC.QUEPRI register.

The SDMAARB[5:0] MAXWAIT field does not affect only the EMC (IVA_IDMA.SDMAARBE register), but also the UMC (IVA_UMC.SDMAARBU register) and the DMC (IVA_DMC.SDMAARBD register).

Note: Because of the pipelined nature of the VBUS interface, the DSP megamodule module can have multiple DMA transfers in flight at a time. To avoid priority inversion, the DSP megamodule must not only evaluate the priority of the transfer at the head of the EMC queue(s), but the priority of all transfers that have been accepted/queued on the VBUS interface(s), and of higher priority transfers that are queued in the device-level DMA infrastructure.

- MDMAARBE register - Master DMA priority programming model

  The IVA_IDMA.MDMAARBE[18:16] PRI field controls the submission priority for master DMA transactions (which are a result of cache misses or long-distance accesses to nonconfiguration space) and configuration bus transactions (long-distance accesses to configuration space or IDMA transfers to configuration space).

  The IVA_IDMA.MDMAARBE priority is different from other programmable priorities in the system, in that the priority value does not affect internal arbitration for resources. This PRI[3:0] value is simply used as the VBUS priority value for all transactions initiated by the DMA master interface or memory-mapped register configuration interface.

  Arbitration for the internal half of the transfer depends on the initiator (which could be DSP CPU, PMC, or DMC) or user coherence (UMC), or IDMA, etc. Arbitration for the external half of the transfer is DMA-dependent, and should rely on the Vbus priority, which is copied from the IVA_IDMA.MDMAARBE[18:16] PRI field.

  Note: Because no internal arbitration results from the IVA_IDMA.MDMAARBE register, there is no need for the MAXWAIT field in this register.

14.4.8.3 SL2 Memory Management

14.4.8.3.1 SL2 Performance Optimizations

To limit the number of accesses through the SL2 interface and to optimize bandwidth, it is recommended that the user access SL2 using, as much as possible, bursts and aligned on burst boundaries. Optimum bandwidth savings can be obtained by doing bursts of 8x32 aligned on 32-byte boundaries.

This can be done using ARM968 (sequencer module) LDM or STM instructions. For details, see the ARM reference manual.

14.4.8.3.2 SL2 Performance Limitations

Accesses through the SL2IF OCP interface to SL2 memory are not intended to be optimized for single reads, but only for bursts. Maximum bandwidth can be obtained using bursts of 8x32b bursts or larger (bursts larger than 8x32 are divided into 8x32 bursts). Generating single requests to the SL2IF interface has two disadvantages:

- Uses a large portion of the SL2 bandwidth, resulting in iME and iLF performance degradation
- Increases dynamic power consumption

In particular, ARM968 instruction fetches generate single requests on the local interconnect. SL2 is not optimized for ARM968 instruction direct fetching. If the ITCM is too small for an ARM968 program, an ARM968 program must be DMAed in from memory (possibly SL2, as DMA accesses are bursted).

14.4.8.3.3 SL2 Illegal Accesses

It is illegal for the user to generate unaligned bursts that span SL2 address range boundaries.

14.4.9 IVA2.2 Power Management

14.4.9.1 Clock Management

14.4.9.1.1 Clock Configuration

The IVA2.2 subsystem receives one single-clock signal from the PRCM, the DPLL2_ALWON.FCLK.

From the DPLL2_ALWON.FCLK functional clock provided by the PRCM, three internal clocks (CD0_CLK, CD1_CLK, and CD2_CLK) are generated by the IVA2.2 DPLL and SYSC modules:

- The frequency of the CD0_CLK clock can be software-tuned with a PRCM register by setting the PRCM.CM_CLKSEL1_PLL_IVA2 and PRCM.CM_CLKSEL2_PLL_IVA2 registers.
- The CD1_CLK clock is always a divide-by-two of the CD0_CLK clock and its frequency cannot be changed by software.
- The CD2_CLK clock is always a divide-by-two of the CD0_CLK clock and its frequency cannot be changed by software.

For a complete description of the PRCM registers discussed in this section, see Chapter 4, *Power, Reset, and Clock Management*.

14.4.9.1.2 Clock Gating

- IVA2.2 subsystem

  The IVA2.2 internal clocks can be hardware-disabled by the PRCM when the MSTANDBY/WAIT handshake protocol occurs. To configure the PRCM so that IVA2.2 internal clocks are hardware-supervised, set the PRCM.CM_AUTOIDLE_PLL_IVA2[2:0] AUTO_IVA2_DPLL field to 0x1.

  When the IVA2.2 subsystem does not require its internal clocks, they can be disabled at the PRCM level by setting the PRCM.CM_FCLKEN_IVA2[0] EN_IVA2 bit to 0.

  For a complete description of these PRCM registers, see Chapter 4, *Power, Reset, and Clock Management*.
- SYSC module

  The IVA2.2 SYSC module implements automatic clock gating on internal hardware detection of the absence of activity. The transition from clock-gated to clock-nongated state is operated with no cycle latency penalty.

  The automatic clock-gating feature is enabled by setting the IVA_SYSC.SYSC_SYSCONFIG[0] AUTOIDLE bit to 1 (default value). This feature can be disabled by setting the IVA_SYSC.SYSC_SYSCONFIG[0] AUTOIDLE bit to 0 and making the clock free-running.
- WUGEN module

  The IVA2.2 WUGEN module implements automatic clock gating on internal hardware detection of the absence of activity. The transition from clock-gated to clock-nongated state operates with no cycle latency penalty.

  Automatic clock gating is enabled by setting the IVA_WUGEN.WUGEN_SYSCONFIG[0] AUTOIDLE bit to 1 (default mode). The clock can be made free-running by setting the IVA_WUGEN.WUGEN_SYSCONFIG[0] AUTOIDLE bit to 0.
- TPCC from EDMA module

  The EDMA (or TPCC) module implements automatic clock gating on internal hardware detection of the absence of activity. The transition from clock-nongated to clock-gated state operates with no cycle latency penalty.

14.4.9.2 Reset Management

In addition to hardware reset signals generated by the PRCM, the IVA2.2 subsystem can be reset by software control.

The three DSP power domain software resets (DSP_RST1, DSP_RST2, and DSP_RST3) are partial warm-reset sources. These software resets map to the PRCM.RM_RSTCTL_IVA2[0] RST1_IVA2 bit, the RM_RSTCTRL_IVA2[1] RST2_IVA2 bit, and the PRCM.RM_RSTCTL_IVA2[2] RST3_IVA2 bit, respectively, in the PRCM register RM_RSTCTRL_IVA2.

Setting these 3 bits to 1 performs a software reset to the IVA2.2 subsystem.

The reset status is logged in the PRCM.RM_RSTST_IVA2[8] IVA2_SW_RST1 read-only bit for IVA2_RST1 software reset, the PRCM.RM_RSTST_IVA2[9] IVA2_SW_RST2 read-only bit for IVA2_RST2 software reset, and the PRCM.RM_RSTST_IVA2[10] IVA2_SW_RST3 read-only bit for IVA2_RST3 software reset.

For a complete description of these PRCM registers, see Chapter 4, *Power, Reset, and Clock Management*.

Note: Software reset can be applied only while the IVA2.2 subsystem is in clock-off mode.

14.4.9.3 Power-Down and Wake-Up Management

- DSP megamodule power-down controller (PDC)

  The DSP megamodule embeds a PDC block that allows power-down management by software. Individual DSP megamodule blocks such as DSP CPU, PMC, DMC, EMC, and UMC can be powered off by the PDC.
- This software control is ensured by the IVA_SYS.PDCCMD register:
  - By setting the IVA_SYS.PDCCMD[16] GEMPD bit to 1, the user enables power-down management during idle mode.
  - By setting the IVA_SYS.PDCCMD xMCLOG[1:0] and IVA_SYS.PDCCMD xMCMEM[1:0] fields (x = {P, D, U}), the user controls the XMC clock-gating and standby memory modes. For example, DMC is controlled with the IVA_SYS.PDCCMD[5:4] DMCLOG and IVA_SYS.PDCCMD[7:6] xMCMEM fields.
  - The programming sequence for transition to clock-off state is as follows:

    Before executing the IDLE instruction, the user must perform the following sequence:

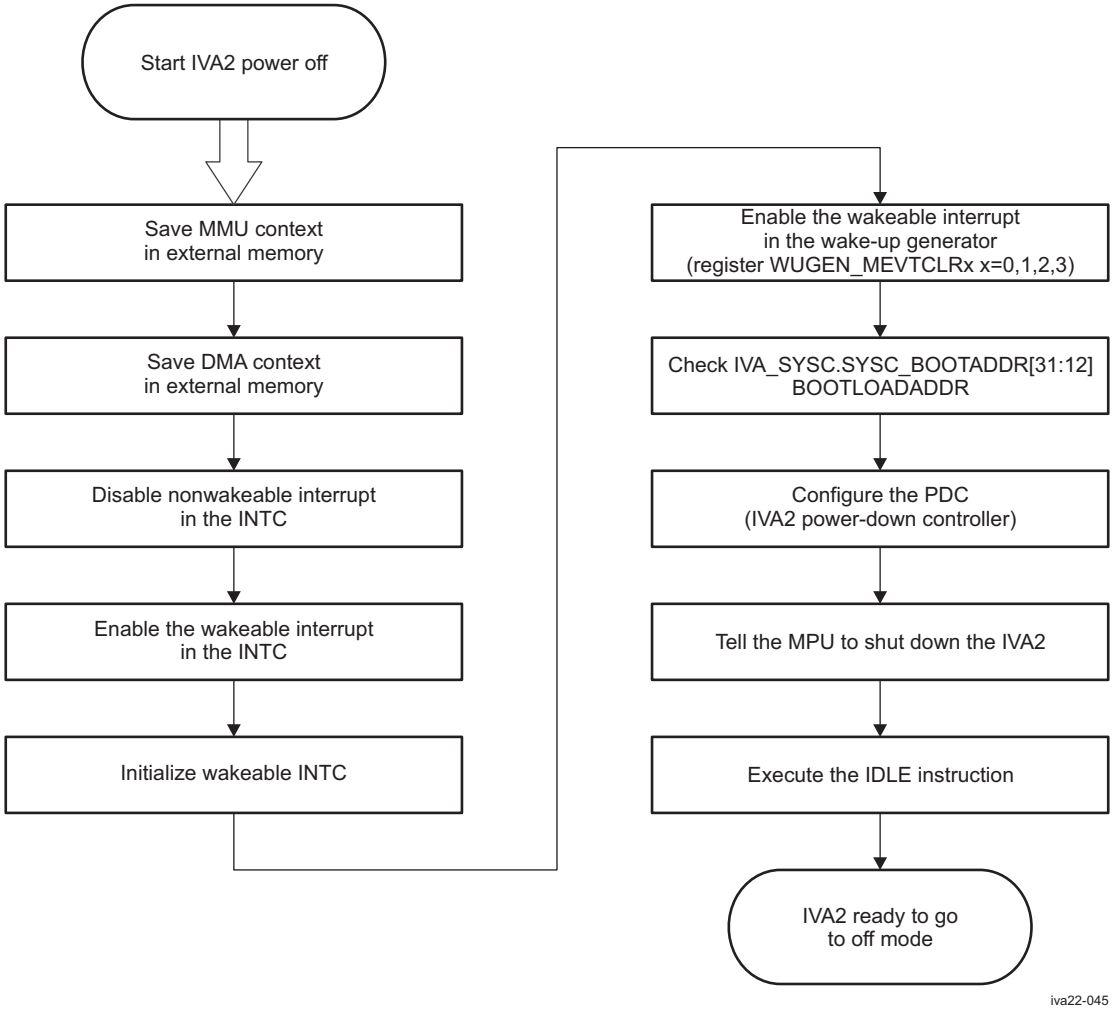
    1. Write 1 to the IVA_SYS.PDCCMD[16] GEMPD bit (standby state); by default, the IVA_SYS.PDCCMD xMCLOG[1:0] and IVA_SYS.PDCCMD xMCMEM[1:0] (x = {P, D, U}) field values are all 0x1, so that module clock gating is enabled and standby mode of memories is statically activated after the IDLE instruction executes.
    2. Mask all interrupts not intended to wake up the IVA2.2 subsystem.
    3. Program the PRCM so that IVA2.2 clocks are cut on IVA2.2 standby. For more information, see Chapter 4, *Power, Reset, and Clock Management*.
    4. Read back all the written registers to ensure write completion.

    The user must also ensure that no other instruction is executed parallel to the IDLE instruction.

    When a clock stop request is asserted, the PDC_INT event (EVT118, see Table 14-3) is generated to the DSP megamodule IC module to inform the DSP CPU to initiate a power-down sequence. For more information about interrupt management, see Section 14.4.7, *Interrupt Management*.
- Programming sequence for transition to power-off state:

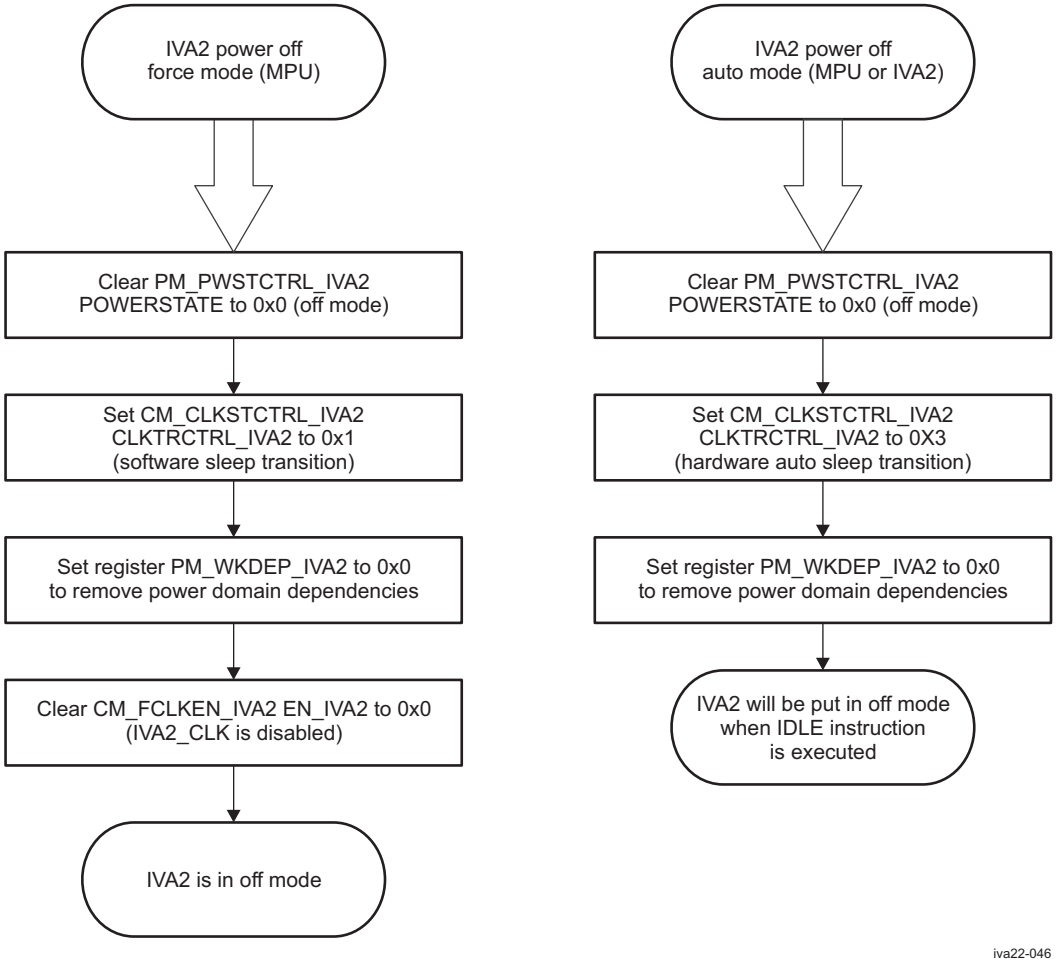
  Before executing the IDLE instruction, the user must perform the sequence shown in Figure 14-44.

Figure 14-44. IVA2 Power Off



The user must also ensure that no other instruction is executed parallel to the IDLE instruction. When IVA2 is ready to go to off mode, there are two ways to completely shut down IVA2 subsystem (see Figure 14-45):

Figure 14-45. IVA2 Power Down

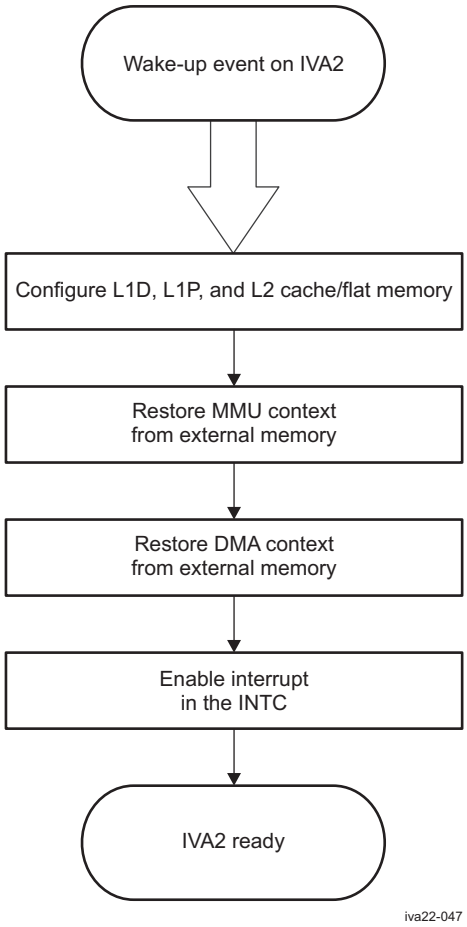


CAUTION

The IVA2 clock must not be stopped manually; otherwise, the WUGEN module inside the IVA2 has no functional clock and cannot wake up the IVA subsystem.

When the IVA2 is shut down, it can be waked up by an external event. The boot code executed after wakeup must include at least the following steps (see Figure 14-46):

Figure 14-46. IVA2 Wake Up



14.4.9.4 Powering Down L2$ Memory While IVA2 is Active

If the L2$ is unused while the DSP is active, the IVA2.2 can completely switch off the 96-KB SRAM attached to the L2$ controller. This typically happens when the DSP is running at a low frequency (for instance, at the same frequency as the SDRAM) where L2$ is not justified. The alternative is to switch off L2$ SRAM completely, losing all L2-cache content and memory-mapped SRAM (from 0x10800000 to 0x1080FFFF). It is impossible to put the SRAM in low-leakage mode and retain data/program while the DSP is active. Also, L2 cache SRAM is the only memory/cache that supports this off mode while DSP is active. L1D and L1P cache SRAM off modes are not supported while DSP is active.

The sequence to enter L2$ off mode while DSP is active follows:

1. Save L2CFG and disable the L2$ by converting L2-cache SRAM to memory-mapped SRAM only (96KB): L2CFG.L2MODE = 0x0.
2. Read back L2CFG to ensure that Step 1 is complete.
3. Save L2MPPAj (j = 0 to 31) and write 0x0 to the L2MPPAj (j = 0 to 31) registers to report access to L2 memory (for debug) and enable associated permission check interrupt and/or exception.
4. Configure the PRCM to switch off L2: PRCM.PM_PWSTCTRL_IVA2.SHAREDL2CACHEFLATONSTATE = 0x0.
5. Read back PRCM.PM_PWSTCTRL_IVA2 to ensure that Step 4 is complete.

After this sequence, the user must not access the L2 memory. If this happens (typically in code under debug), a memory-protection interrupt and/or exception is taken.

The sequence to exit L2$ off mode (while DSP is active) follows:

1. Configure the PRCM to switch on L2: PRCM.PM_PWSTCTRL_IVA2.SHAREDL2CACHEFLATONSTATE = 0x3.
2. Read back PRCM.PM_PWSTCTRL_IVA2 to ensure that Step 1 is complete.
3. Restore L2MPPAj (j = 0 to 31).
4. Restore L2CFG.
5. Read back PRCM.PM_PWSTCTRL_IVA2 to ensure that Step 4 is complete.

Note: After this sequence, L2$ is empty and content in L2$-SRAM configured as memory-mapped is lost.

Note: This mode does not provide large power savings, but is provided in IVA2.2 as an enabler for power-management software development, anticipating that leakage is a major contributor to power consumption, even in active mode.

14.4.9.5 Video and Sequencer Module Management

14.4.9.5.1 Module Dynamic Power Savings

The video accelerator/sequencer modules implement auto-clock gating to save dynamic power consumption when no activity is locally detected. For some modules (iME, iLF, sequencer, video system controller), it is also possible to disable auto-gating by software. Auto-clock gating has no performance impact.

To optimize local power management, configure the following registers (default values):

- IVA.iME_SYSCONFIG AutoIdle = 1
- IVA.iLF_SYSCONFIG AutoIdle = 1
- IVA.SEQ_SYSCONFIG AutoIdle = 1
- IVA.VIDEOSYSC_SYSCONFIG AutoIdle = 1

14.4.9.5.2 System Dynamic Power Savings

For further system dynamic power savings, a module root clock can be independently stopped when the module has no activity.

The root clock of a module can be stopped by writing 0 in the bit associated with that module in the IVA.VIDEOSYSC_CLKCTL register. The clock can be stopped, depending on whether the module is active when it is asked to be stopped. Reading the bit associated with the module in the IVA.VIDEOSYSC_CLKST register lets the user check whether the module clock has effectively been stopped.

Writing 1 in the bit associated with that module immediately restarts clocks if they are stopped, and makes the module operational

When stopping the module root clock, take the following precautions:

- Synchronize with other CPUs/DMAs to determine whether another part of the system requires access to the module.
- Check whether the module has dependencies with other modules.
- Poll the status register in the module to determine whether the module is active.

CAUTION

Invalid configuration of VIDEOSYSC_CLKCTL causes unpredictable results.

For further system dynamic power savings, the sequencer CPU root clock can be independently divided when sequencer activity is reduced.

The root clock of the sequencer CPU can be divided by writing a value different from the value of the NOCLKDIV bit in the VIDEOSYSC_CLKDIV register. There is no software precaution to divide the sequencer CPU clock internally, in terms of traffic or idle conditions. However, this setting is static; that is, it is defined based on whether the sequencer MHz budget must run at full speed, based on a particular task or scenario. The VIDEOSYSC_CLKDIV value must not be changed twice within 16 sequencer clock cycles, or the second request can be ignored. This has no other side effect. Because of the requirement that the setting be static, it is acceptable to make this sequence uninterruptible, pad it with NOPs (or useful operations), or check for change effect, after changing the sequencer clock ratio.

The sequencer CPU clock can be independently divided by 2, 3, or 4, or not divided at all. By default, this clock is not divided.

14.4.10 Error Identification Process

Several mechanisms are available in the IVA2.2 subsystem to report errors at different levels.

14.4.10.1 Error Reporting for IDMA Module

The IDMA has its own error-reporting mechanism. The DSP megamodule implements an error register in EMC (IVA_IDMA.IBUSERR) that latches errors for external invalid transactions on either the DMA (MDMA) bus or the configuration (CFG) bus. Errors are detected on the CFG and MDMA write status or read status interfaces. If an error is received and the corresponding command is an invalid transaction, the IVA_IDMA.IBUSERR[31:29] ERR, IVA_IDMA.IBUSERR[10:8] XID, and IVA_IDMA.IBUSERR[2:0] STAT fields are updated accordingly. An error signaled through a non-0 read status is detected on the read data/status interface or a non-0 write status on the write status interface. Alternately, the EMC records an error if a read or write status response is detected for an unrecognized write ID.

Note: The user must ensure that the EMC detects/stores error information only for the first detected error on either the MDMA or CFG bus, regardless of whether it is a read or write status. In other words, a single error register (IVA_IDMA.IBUSERR) is shared by read and write errors and by the MDMA bus and the CFG bus, and only the first error is stored. If a read error and a write error are detected at the same time, the write status error is given higher priority and is latched into the IVA_IDMA.IBUSERR register.

When an error is detected and stored in the IVA_IDMA.IBUSERR register, the EMC_BUSERR event (EVT127, see Table 14-3) is available as a system event that can be selected as either a DSP CPU interrupt or an exception event.

The user can clear latched errors by writing 1 to the IVA_IDMA.IBUSERRCLR[0] CLR bit.

14.4.10.2 Error Reporting for EDMA Module

The TPTC and TPCC blocks of the EDMA module also contain registers to inform the user of a problem during IVA2.2 external DMA communication.

TPCC Block

The TPCC provides a single error interrupt output CCERRINT (EVT38, see Table 14-3) that consolidates the error conditions:

- QDMA missed events (stored in the TPCC_QEMR register)
- DMA missed events (stored in the TPCC_EMR register)
- Transfer completion code error (stored in the TPCC_CCERR[16] TCCERR bit)
- Queue threshold error events (stored in the TPCC_CCERR[n] QTHRXCDn bit, n = {0,1}]

When an error is detected, the CCERRINT event is asserted, because error events do not have enables.

Note: The CCERRINT event follows the same conventions as other TPCC interrupt outputs. When the error interrupt condition transitions from a no-errors-are-set state to at least one error set, the error output (CCERRINT) pulses high for one TPCC clock cycle. If additional errors are latched before the original error is cleared by the user, the TPCC does not generate additional interrupt pulses. The software must poll all bits during execution of the ISR, and clear all error conditions, so that subsequent error pulses can be generated by the TPCC block.

Error interrupts can be set and/or reevaluated by writes to the TPCC_EEVAL register.

TPTC Block

The TPTC can also detect several error conditions:

- Read status or write status errors in the TPTCj_ERRSTAT[0] BUSERR status bit
- TR error in the TPTCj_ERRSTAT[2] TRERR status bit
- MMR address error in the TPTCj_ERRSTAT[3] MMRAERR status bit

Errors are recorded in the TPTCj_ERRSTAT register, regardless of whether they are enabled. They can be cleared from the TPTCj_ERRSTAT register only by writing 1 to the corresponding bit of the IVA_TPTCj_ERRCLR register.

The error details register (IVA_TPTCj_ERRDET) contains additional information about the first read status error or write status error detected. Future errors are bit-recorded until the TPTCj_ERRDET register is cleared.

If read status and write status are returned to the TPTC in the same cycle, the read or write status value that has an error (nonzero) is latched in the TPTCj_ERRDET register and the TPTCj_ERRSTAT[0] BUSERR bit is set. If both read and write status are nonzero, the write status is given priority for setting the TPTCj_ERRDET register. The TPTCj_ERRDET register is cleared by writing 1 to the TPTCj_ERRCLR[0] BUSERR bit.

If an error is enabled (by the TPTCj_ERREN register bits), the first occurrence of an enabled error generates a pulsed interrupt to the CPU by the TCERRINT event output (TCERRINT0 with EVT39 for TPTG0 and TCERRINT1 with EVT40 for TPTG1, respectively; see Table 14-3). Subsequent errors do not generate a new pulse until all accumulated errors are cleared by the CPU. The CPU clears bits in the TPTCj_INTSTAT register by writing 1 to the corresponding bit(s) of the TPTCj_NTCLR register.

14.4.10.3 Error Reporting for the L3 Interconnect

L3 interconnect out-of-band errors are also reported by the external L3 interrupt signal (l3_ia_iva2_initSError_o). This signal corresponds to the EVT84 event and is directly connected to the IVA2.2_nIRQ[39] DSP CPU interrupt line. For more information about L3 interconnect error reporting, see Chapter 5, *Interconnect*.

14.4.11 Recommendations for Static Settings

The following static settings are recommended:

- SYSC.SYSC_LICFG0.DMATRUECOMPEN = 1; // DMA last write is nonposted.
- SYSC.SYSC_LICFG0.GEMTRUECOMPEN = 1; // DSP last write is nonposted.
- SYSC.SYSC_LICFG0.GEMBURSTOPTEN = 1; // DSP burst optimization

Ensure that the following setting is carefully set:

- All 2D DMA transfers source and/or destination are to a VRFB view (tiling structure):
  - // DMA 2D burst optimization
  - SYSC.SYSC_LICFG0.DMA2DOPTEN = 1

Note: If the preceding condition is not set accurately, setting the DMA2DOPTEN optimization can degrade performance.

Notes:

1. A supersection type MMU page (16MB) must be defined for that VRFB view.
2. When the SYSC.SYSC_LICFG0.PAGEXINGEN is set to 1, a 2D DMA should not cross a MMU page boundary. When this bit is set, it disables the hardware check mechanism for better performance (but the user should ensure that 2D DMA does not cross MMU pages).

   Therefore, ensure that the following setting is carefully set:
   SYSC.SYSC_LICFG0.PAGEXINGEN = 1.

Note: If the preceding condition is not set accurately, setting the PAGEXINGEN optimization can cause undefined results, including deadlock situations.

14.5 IVA2.2 Subsystem Register Manual

Table 14-98 lists the IVA2.2 memory space mapping as seen by the IVA2.2 DSP megamodule.

Table 14-98. Instance Summary

Module Name	Base Address	Size
IC	0x0180 0000	64K bytes
SYS	0x0181 0000	64K bytes
IDMA	0x0182 0000	64K bytes
XMC	0x0184 0000	64K bytes
TPCC	0x01C0 0000	64K bytes
TPTC0	0x01C1 0000	1K byte
TPTC1	0x01C1 0400	1K byte
SYSC	0x01C2 0000	4K bytes
WUGEN	0x01C2 1000	4K bytes
SEQ(1)	0x0009 0000	2K bytes
VIDEOSYSC(1)	0x0009 C000	4K bytes
iME(1)	0x000A 0000	4K bytes
iLF(1)	0x000A 1000	4K bytes
IA_GEM	0x000F 8800	1K bytes
IA_EDMA	0x000F 8C00	1K bytes
IA_SEQ	0x000F 9000	1K bytes

(1) These modules are accessible through DSP EFI port, with EFI instruction only. See Section 14.3.8, *Video Accelerator/Sequencer Local Interconnect*.

For more information on the IVA2.2 memory mapping, see Chapter 2, *Memory Mapping*.

Note: The MMU2 (IVA2.2 MMU) registers are descibed in Chapter 8, *Memory Management Units*. (The MMU base address is 0x5D00 0000 on the L3 interconnect.)

CAUTION

The IVA2.2 registers are limited to 32-bit data accesses. 16-bit and 8-bit are not allowed and can corrupt register content.

14.5.1 IC Registers

This section provides information about the IC Module. Each register in the module is described in Table 14-100 through Table 14-122.

14.5.1.1 IC Register Mapping Summary

Table 14-99. IC Register Summary

Register Name	Type	Register Width (Bits)	Address Offset	Physical Address
EVTFLAGi[1]	R	32	0x0000 0000	0x0180 0000 + (0x4 * i)
EVTSETi[1]	W	32	0x0000 0020	0x0180 0020 + (0x4 * i)
EVTCLRi[1]	W	32	0x0000 0040	0x0180 0040 + (0x4 * i)
EVTMASKi[1]	RW	32	0x0000 0080	0x0180 0080 + (0x4 * i)
MEVTFLAGi[1]	R	32	0x0000 00A0	0x0180 00A0 + (0x4 * i)
EXPMASKi[1]	RW	32	0x0000 00C0	0x0180 00C0 + (0x4 * i)
MEXPFLAGi[1]	R	32	0x0000 00E0	0x0180 00E0 + (0x4 * i)
INTMUXj[2]	RW	32	0x0000 0104	0x0180 0104 + (0x4 * j)
INTXSTAT	R	32	0x0000 0180	0x0180 0180
INTXCLR	W	32	0x0000 0184	0x0180 0184
INTDMASK	RW	32	0x0000 0188	0x0180 0188
EVTASRT	W	32	0x0000 01C0	0x0180 01C0

(1) i = 0 to 3
(2) j = 1 to 3

14.5.1.2 IC Register Descriptions

Table 14-100. EVTFLAGi

Address Offset	0x0000 + (0x4*i)		
Physical address	0x0180 0000 + (0x4*i)	**Instance**	IVA2.2 GEMIC
Description	Event Flag Register		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EF																															

Bits	Field Name	Description	Type	Reset
31:0	EF	Event Flag status 0: No event occurred 1: An event occurred	R	0

Table 14-101. Register Call Summary for Register EVTFLAGi

IVA2.2 Subsystem Functional Description
- INTC: [0] [1] [2] [3] [4] [5] [6] [7] [8] [9] [10]

IVA2.2 Subsystem Basic Programming Model
- Event Combined Programming Sequence: [11]
- Interrupt Controller Basic Programming Model for Power On of IVA2.2 Subsystem: [12] [13] [14]

IVA2.2 Subsystem Register Manual
- IC Register Mapping Summary: [15]

Table 14-102. EVTSETi

Address Offset	0x0020 + (0x4*i)		
Physical address	0x0180 0020 + (0x4*i)	**Instance**	IVA2.2 GEMIC
Description	Event Set Register		
Type	W		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ES																															

Bits	Field Name	Description	Type	Reset
31:0	ES	Event Set Write 0: No action Write 1: Set corresponding event flag	W	0

Table 14-103. Register Call Summary for Register EVTSETi

IVA2.2 Subsystem Functional Description
- INTC: [0]

IVA2.2 Subsystem Register Manual
- IC Register Mapping Summary: [1]

Table 14-104. EVTCLRi

Address Offset	0x0040 + (0x4*i)		
Physical address	0x0180 0040 + (0x4*i)	**Instance**	IVA2.2 GEMIC
Description	Event Clear Register		
Type	W		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EC																															

Bits	Field Name	Description	Type	Reset
31:0	EC	Event Clear Write 0: No action Write 1: Clear corresponding event flag	W	0

Table 14-105. Register Call Summary for Register EVTCLRi

IVA2.2 Subsystem Functional Description
- INTC: [0] [1]

IVA2.2 Subsystem Basic Programming Model
- Event Combined Programming Sequence: [2]

IVA2.2 Subsystem Register Manual
- IC Register Mapping Summary: [3]

Table 14-106. EVTMASKi

Address Offset	0x0080 + (0x4*i)		
Physical address	0x0180 0080 + (0x4*i)	**Instance**	IVA2.2 GEMIC
Description	Event Mask Register		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EM																															

Bits	Field Name	Description	Type	Reset
31:0	EM	Disables event from being used as input to the event combiner: 0: Will be combined 1: Is disabled from being combined.	RW	0

Table 14-107. Register Call Summary for Register EVTMASKi

IVA2.2 Subsystem Functional Description
- INTC: [0] [1] [2] [3] [4] [5] [6] [7] [8]

IVA2.2 Subsystem Basic Programming Model
- Event Combined Programming Sequence: [9]
- Interrupt Controller Basic Programming Model for Power Down of IVA2.2 Subsystem: [10]
- Interrupt Controller Basic Programming Model for Power On of IVA2.2 Subsystem: [11] [12] [13] [14] [15]

IVA2.2 Subsystem Register Manual
- IC Register Mapping Summary: [16]

Table 14-108. MEVTFLAGi

Address Offset	0x00A0 + (0x4*i)		
Physical address	0x0180 00A0 + (0x4*i)	**Instance**	IVA2.2 GEMIC
Description	Masked Event Flag Register		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MEF																															

Bits	Field Name	Description	Type	Reset
31:0	MEF	Masked Event Flag 0: No unmasked event occurred 1: An unmasked event occurred	R	0

Table 14-109. Register Call Summary for Register MEVTFLAGi

IVA2.2 Subsystem Basic Programming Model
- Event Combined Programming Sequence: [0] [1] [2] [3]

IVA2.2 Subsystem Register Manual
- IC Register Mapping Summary: [4]

Table 14-110. EXPMASKi

Address Offset	0x00C0 + (0x4*i)		
Physical address	0x0180 00C0 + (0x4*i)	**Instance**	IVA2.2 GEMIC
Description	Exception Mask Register 0		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
XM																															

Bits	Field Name	Description	Type	Reset
31:0	XM	Enables event from being used in the exception combiner: 0: Will be combined 1: Is disabled from being combined	RW	0xFFFF FFFF

Table 14-111. Register Call Summary for Register EXPMASKi

IVA2.2 Subsystem Functional Description

- INTC: [0]

IVA2.2 Subsystem Register Manual

- IC Register Mapping Summary: [1]

Table 14-112. MEXPFLAGi

Address Offset	0x00E0 + (0x4*i)		
Physical address	0x0180 00E0 + (0x4*i)	**Instance**	IVA2.2 GEMIC
Description	Masked Exception Flag Register 0		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MXF																															

Bits	Field Name	Description	Type	Reset
31:0	MXF	Masked Exception Flag 0: No unmasked exception occurred 1: An unmasked exception occurred	R	0

Table 14-113. Register Call Summary for Register MEXPFLAGi

IVA2.2 Subsystem Register Manual

- IC Register Mapping Summary: [0]

Table 14-114. INTMUXj

Address Offset	0x0104 + (0x4*j), j= 1 to 3		
Physical address	0x0180 0104 + (0x4*i)	**Instance**	IVA2.2 GEMIC
Description	Interrupt Mux Register		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved	INTSEL(i*4+3)							Reserved	INTSEL(i*4+2)							Reserved	INTSEL(i*4+1)							Reserved	INTSEL(i*4)						

Bits	Field Name	Description	Type	Reset
31	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0x0
30:24	INTSEL(j*4+3)	Source event number of the CPU interrupt #j*4+3	RW	i*4+3
23	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0x0
22:16	INTSEL(j*4+2)	Source event number of the CPU interrupt #j*4+2	RW	i*4+2
15	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0x0
14:8	INTSEL(j*4+1)	Source event number of the CPU interrupt #j*4+1	RW	i*4+1
7	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0x0
6:0	INTSEL(j*4)	Source event number of the CPU interrupt #j*4	RW	i*4

Table 14-115. Register Call Summary for Register INTMUXj

IVA2.2 Subsystem Functional Description
- INTC: [0] [1]

IVA2.2 Subsystem Basic Programming Model
- Event <-> Interrupt Mapping Programming Sequence: [2]
- Interrupt Controller Basic Programming Model for Power Down of IVA2.2 Subsystem: [3]
- Interrupt Controller Basic Programming Model for Power On of IVA2.2 Subsystem: [4] [5] [6] [7] [8] [9]

IVA2.2 Subsystem Register Manual
- IC Register Mapping Summary: [10]

Table 14-116. INTXSTAT

Address Offset	0x0000 0180		
Physical address	0x0180 0180	**Instance**	IVA2.2 GEMIC
Description	Interrupt Exception Status Register		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SYSINT								CPUINT								RESERVED															DROP

Bits	Field Name	Description	Type	Reset
31:24	SYSINT	System Event number 00000000: EVT0 ..... 01111111: EVT127 Others: Reserved	R	0x00
23:16	CPUINT	CPU interrupt number 00000000: CPUINT0 ..... 00001111: CPUINT15 Others: Reserved	R	0x00
15:1	RESERVED	Write 0s for future compatibility. Read returns 0.	R	0x0000
0	DROP	Dropped event flag 0: No events dropped 1: Event was dropped by the CPU	R	0

Table 14-117. Register Call Summary for Register INTXSTAT

IVA2.2 Subsystem Basic Programming Model

- Interrupt Exception Programming Sequence: [0] [1] [2]

IVA2.2 Subsystem Register Manual

- IC Register Mapping Summary: [3]

Table 14-118. INTXCLR

Address Offset	0x0000 0184		
Physical address	0x0180 0184	**Instance**	IVA2.2 GEMIC
Description	Interrupt Exception Clear Register		
Type	W		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																															CLEAR

Bits	Field Name	Description	Type	Reset
31:1	Reserved	Write 0s for future compatibility. Read returns 0.	W	0x00000000
0	CLEAR	Interrupt exception status clear register: Write 0: No effect Write 1: Clears the Interrupt Exception Status register	W	0

Table 14-119. Register Call Summary for Register INTXCLR

IVA2.2 Subsystem Basic Programming Model

- Interrupt Exception Programming Sequence: [0] [1]

IVA2.2 Subsystem Register Manual

- IC Register Mapping Summary: [2]

Table 14-120. INTDMASK

Address Offset	0x0000 0188		
Physical address	0x0180 0188	**Instance**	IVA2.2 GEMIC
Description	Dropped Interrupt Mask Register		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																IDM15	IDM14	IDM13	IDM12	IDM11	IDM10	IDM9	IDM8	IDM7	IDM6	IDM5	IDM4	Reserved			

Bits	Field Name	Description	Type	Reset
31:16	Reserved	Write 0 for future compatibility Read returns 0	RW	0
15	IDM15	Dropped event mask for CPU interrupt #15	RW	0
14	IDM14	Dropped event mask for CPU interrupt #14	RW	0
13	IDM13	Dropped event mask for CPU interrupt #13	RW	0
12	IDM12	Dropped event mask for CPU interrupt #12	RW	0
11	IDM11	Dropped event mask for CPU interrupt #11	RW	0
10	IDM10	Dropped event mask for CPU interrupt #10	RW	0
9	IDM9	Dropped event mask for CPU interrupt #9	RW	0
8	IDM8	Dropped event mask for CPU interrupt #8	RW	0
7	IDM7	Dropped event mask for CPU interrupt #7	RW	0

Bits	Field Name	Description	Type	Reset
6	IDM6	Dropped event mask for CPU interrupt #6	RW	0
5	IDM5	Dropped event mask for CPU interrupt #5	RW	0
4	IDM4	Dropped event mask for CPU interrupt #4	RW	0
3:0	Reserved	Write 0 for future compatibility Read returns 0	RW	0

Table 14-121. Register Call Summary for Register INTDMASK

IVA2.2 Subsystem Functional Description

- INTC: [0]

IVA2.2 Subsystem Basic Programming Model

- Interrupt Exception Programming Sequence: [1]
- Interrupt Controller Basic Programming Model for Power Down of IVA2.2 Subsystem: [2]
- Interrupt Controller Basic Programming Model for Power On of IVA2.2 Subsystem: [3] [4]

IVA2.2 Subsystem Register Manual

- IC Register Mapping Summary: [5]

Table 14-122. EVTASRT

Address Offset	0x0000 01C0		
Physical address	0x0180 01C0	**Instance**	IVA2.2 GEMIC
Description	Event Assert Register		
Type	W		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																								MXF7	MXF6	MXF5	MXF4	MXF3	MXF2	MXF1	MXF0

Bits	Field Name	Description	Type	Reset
31:8	Reserved	write 0 for future compatibility	W	0
7	MXF7	Event Assert output #7 EA7 = 0: No effect EA7 = 1: EVTOUT7 pulsed high for 4 clk1 cycles, then low.	W	0
6	MXF6	Event Assert output #6 EA6 = 0: No effect EA6 = 1: EVTOUT6 pulsed high for 4 clk1 cycles, then low.	W	0
5	MXF5	Event Assert output #5 EA5 = 0: No effect EA5 = 1: EVTOUT5 pulsed high for 4 clk1 cycles, then low.	W	0
4	MXF4	Event Assert output #4 EA4 = 0: No effect EA4 = 1: EVTOUT4 pulsed high for 4 clk1 cycles, then low.	W	0
3	MXF3	Event Assert output #3 EA3 = 0: No effect EA3 = 1: EVTOUT3 pulsed high for 4 clk1 cycles, then low.	W	0
2	MXF2	Event Assert output #2 EA2 = 0: No effect EA2 = 1: EVTOUT2 pulsed high for 4 clk1 cycles, then low.	W	0
1	MXF1	Event Assert output #1 EA1 = 0: No effect EA1 = 1: EVTOUT1 pulsed high for 4 clk1 cycles, then low.	W	0
0	MXF0	Event Assert output #0 EA0 = 0: No effect EA0 = 1: EVTOUT0 pulsed high for 4 clk1 cycles, then low.	W	0

Table 14-123. Register Call Summary for Register EVTASRT

IVA2.2 Subsystem Register Manual • IC Register Mapping Summary: [0]

14.5.2 SYS Registers

This section provides information about the SYS Module. Each register in the module is described separately below.

14.5.2.1 SYS Register Mapping Summary

Table 14-124. SYS Register Summary

Register Name	Type	Register Width (Bits)	Address Offset	Physical Address
PDCCMD	RW	32	0x0000 0000	0x0181 0000
REVID	R	32	0x0000 2000	0x0181 2000

14.5.2.2 SYS Register Descriptions

Table 14-125. PDCCMD

Address Offset	0x0000 0000		
Physical address	0x0181 0000	**Instance**	IVA2.2 GEMSYS
Description	Power-Down Command Register		
Type	RW		

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14	13 12	11 10	9 8	7 6	5 4	3 2	1 0
Reserved	GEMPD	EMCMEM	EMCLOG	UMCMEM	UMCLOG	DMCMEM	DMCLOG	PMCMEM	PMCLOG

Bits	Field Name	Description	Type	Reset
31:17	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0x0000
16	GEMPD	Power-down during IDLE: GEMPD = 0: Normal operation. Do not power-down CPU or DSP megamodule when CPU is IDLE. GEMPD = 1: Sleep mode. Power-down CPU and DSP megamodule when CPU enters IDLE state	RW	0
15:14	EMCMEM	SRAM Sleep Modes Determines the RAM sleep modes used by the EMC for powering-down internal memories. 0x0: No sleep mode supported 0x1: Sleep mode 1 Write 0x2: Sleep mode 2 - equivalent to sleep mode 1 Write 0x3: Sleep mode 3 - equivalent to sleep mode 1	RW	0x1
13:12	EMCLOG	Logic Clock Gating Modes. Determines to what degree the EMC gates its clockinternally. 0x0: No clock gating supported beyond leaf clock gating. 0x1: Static clock gating of unused modulesregions when DSP megamodule is active(pmc_pd_pdstat[1:0] = 00) and Static clock gating when DSP megamodule is in standby (pmc_pd_pdstat[1:0] = 11)	RW	0x1
11:10	UMCMEM	SRAM Sleep Modes Determines the RAM sleep modes used by the UMC for powering-down L2 pages. 0x0: No sleep mode supported 0x1: Sleep mode 1 Write 0x2: Sleep mode 2 - equivalent to sleep mode 1 Write 0x3: Sleep mode 3 - equivalent to sleep mode 1	RW	0x1

Bits	Field Name	Description	Type	Reset
9:8	UMCLOG	Logic Clock Gating Modes. Determines to what degree the UMC gates its clockinternally. 0x0: No clock gating supported beyond leaf clock gating. 0x1: Static clock gating of unused modulesregions when DSP megamodule is active(pmc_pd_pdstat[1:0] = 00) and Static clock gating when DSP megamodule is in standby (pmc_pd_pdstat[1:0] = 11)	RW	0x1
7:6	DMCMEM	SRAM Sleep Modes Determines the RAM sleep modes used by the DMC for powering-down L1D pages. 0x0: No sleep mode supported 0x1: Sleep mode 1 Write 0x2: Sleep mode 2 - equivalent to sleep mode 1 Write 0x3: Sleep mode 3 - equivalent to sleep mode 1	RW	0x1
5:4	DMCLOG	Logic Clock Gating Modes. Determines to what degree the DMC gates its clockinternally. 0x0: No clock gating supported beyond leaf clock gating. 0x1: Static clock gating of unused modulesregions when DSP megamodule is active (pmc_pd_pdstat[1:0] = 00) and Static clock gating when DSP megamodule is in standby (pmc_pd_pdstat[1:0] = 11)	RW	0x1
3:2	PMCMEM	SRAM Sleep Modes Determines the RAM sleep modes used by the PMC for powering-down L1P pages. 0x0: No sleep mode supported 0x1: Sleep mode 1 Write 0x2: Sleep mode 2 - equivalent to sleep mode 1 Write 0x3: Sleep mode 3 - equivalent to sleep mode 1	RW	0x1
1:0	PMCLOG	Logic Clock Gating Modes. Determines to what degree the PMC gates its clockinternally. 0x0: No clock gating supported beyond leaf clock gating. 0x1: Static clock gating of unused modulesregions when DSP megamodule is active (pmc_pd_pdstat[1:0] = 00) and Static clock gating when DSP megamodule is in standby (pmc_pd_pdstat[1:0] = 11)	RW	0x1

Table 14-126. Register Call Summary for Register PDCCMD

IVA2.2 Subsystem Functional Description
- INTC: [0]

IVA2.2 Subsystem Basic Programming Model
- IVA2.2 Boot Configuration: [1] [2] [3]
- Power-Down and Wake-Up Management: [4] [5] [6] [7] [8] [9] [10] [11] [12]

IVA2.2 Subsystem Register Manual
- SYS Register Mapping Summary: [13]

Table 14-127. REVID

Address Offset	0x0000 2000		
Physical address	0x0181 2000	**Instance**	IVA2.2 GEMSYS
Description	DSP megamodule Revision ID Register		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
VERSION																REVISION															

Bits	Field Name	Description	Type	Reset
31:16	VERSION	Functional implementation of DSP megamodule 0x0002 : MidGEM	R	0x0002
15:0	REVISION	Physical implementation of DSP megamodule version	R	0x0000

Table 14-128. Register Call Summary for Register REVID

IVA2.2 Subsystem Register Manual

- SYS Register Mapping Summary: [0]

14.5.3 IDMA Registers

This section provides information about the IDMA Module. Each register in the module is described separately below.

14.5.3.1 IDMA Register Mapping Summary

Table 14-129. IDMA Register Summary

Register Name	Type	Register Width (Bits)	Address Offset	Physical Address
IDMA0_STAT	R	32	0x0000 0000	0x0182 0000
IDMA0_MASK	RW	32	0x0000 0004	0x0182 0004
IDMA0_SOURCE	RW	32	0x0000 0008	0x0182 0008
IDMA0_DEST	RW	32	0x0000 000C	0x0182 000C
IDMA0_COUNT	RW	32	0x0000 0010	0x0182 0010
IDMA1_STAT	R	32	0x0000 0100	0x0182 0100
IDMA1_SOURCE	RW	32	0x0000 0108	0x0182 0108
IDMA1_DEST	RW	32	0x0000 010C	0x0182 010C
IDMA1_COUNT	RW	32	0x0000 0110	0x0182 01100
CPUARBE	RW	32	0x0000 0200	0x0182 0200
IDMAARBE	RW	32	0x0000 0204	0x0182 0204
SDMAARBE	RW	32	0x0000 0208	0x0182 0208
MDMAARBE	RW	32	0x0000 020C	0x0182 020C
ICFGMPFAR	R	32	0x0000 0300	0x0182 0300
ICFGMPFSR	R	32	0x0000 0304	0x0182 0304
ICFGMPFCR	W	32	0x0000 0308	0x0182 0308
IBUSERR	R	32	0x0000 0400	0x0182 0400
IBUSERRCLR	W	32	0x0000 0404	0x0182 0404

14.5.3.2 IDMA Register Descriptions

Table 14-130. IDMA0_STAT

Address Offset	0x0000 0000		
Physical address	0x0182 0000	**Instance**	IVA2.2 GEMIDMA
Description	IDMA Channel 0 Status Register		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																														PEND	ACTV

Bits	Field Name	Description	Type	Reset
31:2	Reserved	Reads return 0s	R	0
1	PEND	Pending transfer: Set when control registers are written to by the CPU and there is already an active transfer in progress (ACTV = 1) and cleared when the transfer becomes active. PEND = 1: Transfer is pending PEND = 0: No pending transfer	R	0
0	ACTV	Active transfer: Set when channel 0 begins reading data from the source address and cleared following the last write to the destination address. ACTV = 1: Active transfer ACTV = 0: No active transfer	R	0

Table 14-131. Register Call Summary for Register IDMA0_STAT

IVA2.2 Subsystem Register Manual

- IDMA Register Mapping Summary: [0]

Table 14-132. IDMA0_MASK

Address Offset	0x0000 0004		
Physical address	0x0182 0004	**Instance**	IVA2.2 GEMIDMA
Description	IDMA Channel 0 Mask Register		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
M31	M30	M29	M28	M27	M26	M25	M24	M23	M22	M21	M20	M19	M18	M17	M16	M15	M14	M13	M12	M11	M10	M9	M8	M7	M6	M5	M4	M3	M2	M1	M0

Bits	Field Name	Description	Type	Reset
31	M31	Register Mask bit: M31 = 1: Register access blocked (masked) M31 = 0: Register access permitted (not masked)	RW	0
30	M30	Register Mask bit: M30 = 1: Register access blocked (masked) M30 = 0: Register access permitted (not masked)	RW	0
29	M29	Register Mask bit: M29 = 1: Register access blocked (masked) M29 = 0: Register access permitted (not masked)	RW	0
28	M28	Register Mask bit: M28 = 1: Register access blocked (masked) M28 = 0: Register access permitted (not masked)	RW	0
27	M27	Register Mask bit: M27 = 1: Register access blocked (masked) M27 = 0: Register access permitted (not masked)	RW	0
26	M26	Register Mask bit: M26 = 1: Register access blocked (masked) M26 = 0: Register access permitted (not masked)	RW	0
25	M25	Register Mask bit: M25 = 1: Register access blocked (masked) M25 = 0: Register access permitted (not masked)	RW	0
24	M24	Register Mask bit: M24 = 1: Register access blocked (masked) M24 = 0: Register access permitted (not masked)	RW	0
23	M23	Register Mask bit: M23 = 1: Register access blocked (masked) M23 = 0: Register access permitted (not masked)	RW	0
22	M22	Register Mask bit: M22 = 1: Register access blocked (masked) M22 = 0: Register access permitted (not masked)	RW	0
21	M21	Register Mask bit: M21 = 1: Register access blocked (masked) M21 = 0: Register access permitted (not masked)	RW	0
20	M20	Register Mask bit: M20 = 1: Register access blocked (masked) M20 = 0: Register access permitted (not masked)	RW	0
19	M19	Register Mask bit: M19 = 1: Register access blocked (masked) M19 = 0: Register access permitted (not masked)	RW	0
18	M18	Register Mask bit: M18 = 1: Register access blocked (masked) M18 = 0: Register access permitted (not masked)	RW	0

Bits	Field Name	Description	Type	Reset
17	M17	Register Mask bit: M17 = 1: Register access blocked (masked) M17 = 0: Register access permitted (not masked)	RW	0
16	M16	Register Mask bit: M16 = 1: Register access blocked (masked) M16 = 0: Register access permitted (not masked)	RW	0
15	M15	Register Mask bit: M15 = 1: Register access blocked (masked) M15 = 0: Register access permitted (not masked)	RW	0
14	M14	Register Mask bit: M14 = 1: Register access blocked (masked) M14 = 0: Register access permitted (not masked)	RW	0
13	M13	Register Mask bit: M13 = 1: Register access blocked (masked) M13 = 0: Register access permitted (not masked)	RW	0
12	M12	Register Mask bit: M12 = 1: Register access blocked (masked) M12 = 0: Register access permitted (not masked)	RW	0
11	M11	Register Mask bit: M11 = 1: Register access blocked (masked) M11 = 0: Register access permitted (not masked)	RW	0
10	M10	Register Mask bit: M10 = 1: Register access blocked (masked) M10 = 0: Register access permitted (not masked)	RW	0
9	M9	Register Mask bit: M9 = 1: Register access blocked (masked) M9 = 0: Register access permitted (not masked)	RW	0
8	M8	Register Mask bit: M8 = 1: Register access blocked (masked) M8 = 0: Register access permitted (not masked)	RW	0
7	M7	Register Mask bit: M7 = 1: Register access blocked (masked) M7 = 0: Register access permitted (not masked)	RW	0
6	M6	Register Mask bit: M6 = 1: Register access blocked (masked) M6 = 0: Register access permitted (not masked)	RW	0
5	M5	Register Mask bit: M5 = 1: Register access blocked (masked) M5 = 0: Register access permitted (not masked)	RW	0
4	M4	Register Mask bit: M4 = 1: Register access blocked (masked) M4 = 0: Register access permitted (not masked)	RW	0
3	M3	Register Mask bit: M3 = 1: Register access blocked (masked) M3 = 0: Register access permitted (not masked)	RW	0
2	M2	Register Mask bit: M2 = 1: Register access blocked (masked) M2 = 0: Register access permitted (not masked)	RW	0
1	M1	Register Mask bit: M1 = 1: Register access blocked (masked) M1 = 0: Register access permitted (not masked)	RW	0
0	M0	Register Mask bit: M0 = 1: Register access blocked (masked) M0 = 0: Register access permitted (not masked)	RW	0

Table 14-133. Register Call Summary for Register IDMA0_MASK

IVA2.2 Subsystem Basic Programming Model

- Starting the Transfer: [0]

IVA2.2 Subsystem Register Manual

- IDMA Register Mapping Summary: [1]

Table 14-134. IDMA0_SOURCE

Address Offset	0x0000 0008		
Physical address	0x0182 0008	**Instance**	IVA2.2 GEMIDMA
Description	IDMA Channel 0 Source Address Register		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SOURCEADDR																											Reserved					

Bits	Field Name	Description	Type	Reset
31:5	SOURCEADDR	Source Address: Must point to a 32-byte-aligned (e.g. window-aligned) memory location local to DSP megamodule or to a valid configuration register space.	RW	0x0000000
4:0	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0x00

Table 14-135. Register Call Summary for Register IDMA0_SOURCE

IVA2.2 Subsystem Basic Programming Model

- Starting the Transfer: [0]

IVA2.2 Subsystem Register Manual

- IDMA Register Mapping Summary: [1]

Table 14-136. IDMA0_DEST

Address Offset	0x0000 000C		
Physical address	0x0182 000C	**Instance**	IVA2.2 GEMIDMA
Description	IDMA Channel 0 Destination Address Register		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DESTADDR																											Reserved					

Bits	Field Name	Description	Type	Reset
31:5	DESTADDR	Destination Address: Must point to a 32-byte-aligned (e.g. windowaligned) memory location local to DSP megamodule or to a valid configuration register space.	RW	0x0000000
4:0	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0x00

Table 14-137. Register Call Summary for Register IDMA0_DEST

IVA2.2 Subsystem Basic Programming Model

- Starting the Transfer: [0]

IVA2.2 Subsystem Register Manual

- IDMA Register Mapping Summary: [1]

Table 14-138. IDMA0_COUNT

Address Offset	0x0000 0010		
Physical address	0x0182 0010	**Instance**	IVA2.2 GEMIDMA
Description	IDMA Channel 0 Count Register		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved			INT	Reserved																				COUNT							

Bits	Field Name	Description	Type	Reset
31:29	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0x0
28	INT	CPU interrupt enable INT = 1: Interrupt CPU (IDMA_INT0) on completion INT = 0: Do not interrupt CPU on completion	RW	0
27:4	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0x000000
3:0	COUNT	Window count COUNT = 0000: Transfer to/from one 32-word window COUNT = n: Transfer to/from n+1 32-word windows	RW	0x0

Table 14-139. Register Call Summary for Register IDMA0_COUNT

IVA2.2 Subsystem Basic Programming Model

- Starting the Transfer: [0]

IVA2.2 Subsystem Register Manual

- IDMA Register Mapping Summary: [1]

Table 14-140. IDMA1_STAT

Address Offset	0x0000 0100		
Physical address	0x0182 0100	**Instance**	IVA2.2 GEMIDMA
Description	IDMA Channel 1 Status Register		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																														PEND	ACTV

Bits	Field Name	Description	Type	Reset
31:2	Reserved	Write 0s for future compatibility. Read returns 0.	R	0x00000000
1	PEND	Pending transfer: Set when control registers are written to by the CPU and there is already an active transfer in progress (ACTV = 1) and cleared when the transfer becomes active. PEND = 1: Transfer is pending PEND = 0: No pending transfer	R	0
0	ACTV	Active transfer: Set when channel 1 begins reading data from the source address and cleared following the last write to the destination address. ACTV = 1: Active transfer ACTV = 0: No active transfer	R	0

Table 14-141. Register Call Summary for Register IDMA1_STAT

IVA2.2 Subsystem Register Manual

- IDMA Register Mapping Summary: [0]

Table 14-142. IDMA1_SOURCE

Address Offset	0x0000 0108		
Physical address	0x0182 0108	**Instance**	IVA2.2 GEMIDMA
Description	IDMA Channel 1 Source Address Register		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SOURCEADDR																															

Bits	Field Name	Description	Type	Reset
31:0	SOURCEADDR	Source Address: Must point to a word-aligned memory location local to GEM. When performing a block fill (IDMA1_COUNT.FILL = 1) the source address is the fill value. Note that when performing a Fill Mode transfer all 32-bits of the SOURCEADDR are writeable and when performing a linear transfer the two LSBs are implemented as 00b.	RW	0x00000000

Table 14-143. Register Call Summary for Register IDMA1_SOURCE

IVA2.2 Subsystem Basic Programming Model
- Internal Memory-to-Memory Transfer (IDMA): [0] [1] [2] [3]

IVA2.2 Subsystem Register Manual
- IDMA Register Mapping Summary: [4]

Table 14-144. IDMA1_DEST

Address Offset	0x0000 010C		
Physical address	0x0182 010C	**Instance**	IVA2.2 GEMIDMA
Description	IDMA Channel 1 Destination Address Register		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DESTADDR																														Reserved	

Bits	Field Name	Description	Type	Reset
31:2	DESTADDR	Destination Address: Must point to a word-aligned memory location local to DSP megamodule.	RW	0x00000000
1:0	Reserved	Reads return 0s Write 0 for further compaibility	R	0x00

Table 14-145. Register Call Summary for Register IDMA1_DEST

IVA2.2 Subsystem Basic Programming Model
- Internal Memory-to-Memory Transfer (IDMA): [0] [1] [2] [3]

IVA2.2 Subsystem Register Manual
- IDMA Register Mapping Summary: [4]

Table 14-146. IDMA1_COUNT

Address Offset	0x0000 0110		
Physical address	0x0182 0110	**Instance**	IVA2.2 GEMIDMA
Description	IDMA Channel 0 Count Register		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PRI			INT	Reserved											FILL	COUNT														Reserved	

Bits	Field Name	Description	Type	Reset
31:29	PRI	Transfer priority. Used for arbitration between CPU and DMA accesses when there are conflicts. PRI = 111b: Low priority PRI = 000b: High priority	RW	0x0
28	INT	CPU interrupt enable INT = 1: Interrupt CPU (IDMA_INT1) on completion INT = 0: Do not interrupt CPU on completion	RW	0
27:17	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0x000
16	FILL	Block fill: FILL = 1: Perform a block fill using the Source address field as the fill value to the memory buffer pointed to by the Destination address field. FILL = 0: Block transfer from the source address to the destination address.	RW	0
15:2	COUNT	16-bit byte count. Must be a multiple of 4 bytes. A transfer count of zero will not transfer any data, but will generate an interrupt if requested in the INT field.	RW	0x0000
1:0	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0x0

Table 14-147. Register Call Summary for Register IDMA1_COUNT

IVA2.2 Subsystem Basic Programming Model

- Internal Memory-to-Memory Transfer (IDMA): [0] [1] [2] [3] [4] [5] [6] [7] [8] [9] [10] [11] [12] [13] [14] [15] [16] [17] [18] [19] [20] [21] [22]
- Internal Memory: [23]

IVA2.2 Subsystem Register Manual

- IDMA Register Mapping Summary: [24]
- IDMA Register Descriptions: [25]

Table 14-148. CPUARBE

Address Offset	**0x0000 0200**		
Physical address	0x0182 0200	**Instance**	IVA2.2 GEMIDMA
Description	CPU Arbitration Control		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved													PRI			Reserved								MAXWAIT							

Bits	Field Name	Description	Type	Reset
31:19	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0x000
18:16	PRI	Priority 0x0: Highest priority	RW	0x1

Bits	Field Name	Description	Type	Reset
		0x1: 2nd highest priority		
		0x2: 3rd highest priority		
		0x3: 4th highest priority		
		0x4: 5th highest priority		
		0x5: 6th highest priority		
		0x6: 7th highest priority		
		0x7: Lowest priority		
15:6	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0x000
5:0	MAXWAIT	Maximum Wait time (in UMC/EMC cycles)	RW	0x10
		0x0: Always stalls due to higher priority requestor		
		0x1: Maximum wait of 1 cycles (1/2 = 50% access)		
		0x2: Maximum wait of 2 cycles (1/3 = 33% access)		
		0x4: Maximum wait of 4 cycles (1/5 = 20% access)		
		0x8: Maximum wait of 8 cycles (1/9 = 11% access)		
		0x10: Maximum wait of 16 cycles (1/17 = 6% access)		
		0x20: Maximum wait of 32 cycles (1/33 = 3% access)		

Table 14-149. Register Call Summary for Register CPUARBE

IVA2.2 Subsystem Basic Programming Model
- Internal Memory: [0]

IVA2.2 Subsystem Register Manual
- IDMA Register Mapping Summary: [1]

Table 14-150. IDMAARBE

Address Offset	0x0000 0204		
Physical address	0x0182 0204	**Instance**	IVA2.2 GEMIDMA
Description	IDMA Arbitration control		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																								MAXWAIT							

Bits	Field Name	Description	Type	Reset
31:6	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0x0000000
5:0	MAXWAIT	Maximum Wait time (in UMC/EMC cycles)	RW	0x10
		0x0: Always stalls due to higher priority requestor		
		0x1: Maximum wait of 1 cycles (1/2 = 50% access)		
		0x2: Maximum wait of 2 cycles (1/3 = 33% access)		
		0x4: Maximum wait of 4 cycles (1/5 = 20% access)		
		0x8: Maximum wait of 8 cycles (1/9 = 11% access)		
		0x10: Maximum wait of 16 cycles (1/17 = 6% access)		
		0x20: Maximum wait of 32 cycles (1/33 = 3% access)		

Table 14-151. Register Call Summary for Register IDMAARBE

IVA2.2 Subsystem Basic Programming Model
- Internal Memory: [0]

Table 14-151. Register Call Summary for Register IDMAARBE (continued)

IVA2.2 Subsystem Register Manual • IDMA Register Mapping Summary: [1]

Table 14-152. SDMAARBE

Address Offset	0x0000 0208		
Physical address	0x0182 0208	**Instance**	IVA2.2 GEMIDMA
Description			
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																								MAXWAIT							

Bits	Field Name	Description	Type	Reset
31:6	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0x000
5:0	MAXWAIT	Maximum Wait time (in UMC/EMC cycles)	RW	0x01
		0x0: Always stalls due to higher priority requestor		
		0x1: Maximum wait of 1 cycles (1/2 = 50% access)		
		0x2: Maximum wait of 2 cycles (1/3 = 33% access)		
		0x4: Maximum wait of 4 cycles (1/5 = 20% access)		
		0x8: Maximum wait of 8 cycles (1/9 = 11% access)		
		0x10: Maximum wait of 16 cycles (1/17 = 6% access)		
		0x20: Maximum wait of 32 cycles (1/33 = 3% access)		

Table 14-153. Register Call Summary for Register SDMAARBE

IVA2.2 Subsystem Basic Programming Model • Internal Memory: [0]
IVA2.2 Subsystem Register Manual • IDMA Register Mapping Summary: [1]

Table 14-154. MDMAARBE

Address Offset	0x0000 020C		
Physical address	0x0182 020C	**Instance**	IVA2.2 GEMIDMA
Description			
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved													PRI			Reserved															

Bits	Field Name	Description	Type	Reset
31:19	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0x0000
18:16	PRI	Priority	RW	0x7
		0x0: Highest priority		
		0x1: 2nd highest priority		
		0x2: 3rd highest priority		
		0x3: 4th highest priority		
		0x4: 5th highest priority		
		0x5: 6th highest priority		

Bits	Field Name	Description	Type	Reset
		0x6: 7th highest priority 0x7: Lowest priority		
15:0	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0x00000

Table 14-155. Register Call Summary for Register MDMAARBE

IVA2.2 Subsystem Basic Programming Model
- Prioritizing Defined Transfers: [0] [1] [2]
- Internal Memory: [3] [4] [5] [6] [7]

IVA2.2 Subsystem Register Manual
- IDMA Register Mapping Summary: [8]

Table 14-156. ICFGMPFAR

Address Offset	0x0000 0300		
Physical address	0x0182 0300	**Instance**	IVA2.2 GEMIDMA
Description	ICFG Memory Protection Fault Address Register		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDR																															

Bits	Field Name	Description	Type	Reset
31:0	ADDR	Fault Address	R	0x00000000

Table 14-157. Register Call Summary for Register ICFGMPFAR

IVA2.2 Subsystem Basic Programming Model
- Internal Memory: [0]

IVA2.2 Subsystem Register Manual
- IDMA Register Mapping Summary: [1]

Table 14-158. ICFGMPFSR

Address Offset	0x0000 0304		
Physical address	0x0182 0304	**Instance**	IVA2.2 GEMIDMA
Description	ICFG Memory Protection Fault Status Register		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																FLTID								Reserved						ATYP	

Bits	Field Name	Description	Type	Reset
31:16	Reserved	Write 0s for future compatibility. Read returns 0.	R	0x0000
15:8	FLTID	Faulted ID: VBUS PrivID of faulting requestor. This field is valid only if LE is zero.	R	0x00
7:6	Reserved	Write 0s for future compatibility. Read returns 0.	R	0x0
5:0	ATYP	Access type	R	0x00

Table 14-159. Register Call Summary for Register ICFGMPFSR

IVA2.2 Subsystem Basic Programming Model

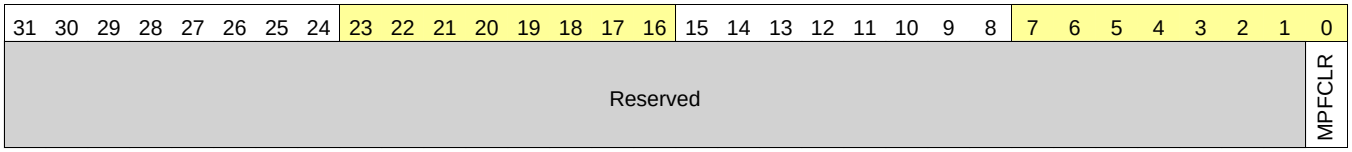
- Internal Memory: [0]

IVA2.2 Subsystem Register Manual

- IDMA Register Mapping Summary: [1]

Table 14-160. ICFGMPFCR

Address Offset	0x0000 0308		
Physical address	0x0182 0308	**Instance**	IVA2.2 GEMIDMA
Description	ICFG Memory Protection Fault Command Register		
Type	W		



Bits	Field Name	Description	Type	Reset
31:1	Reserved	Write 0s for future compatibility. Read returns 0.	W	
0	MPFCLR	Write 0: No effect Write 1: Clear fault logged information	W	0

Table 14-161. Register Call Summary for Register ICFGMPFCR

IVA2.2 Subsystem Basic Programming Model

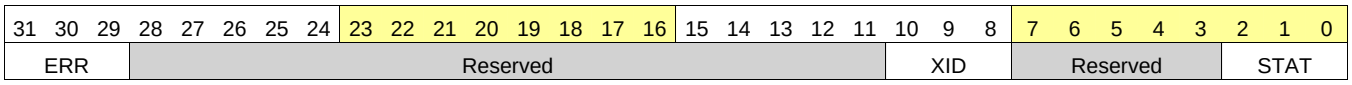
- Internal Memory: [0]

IVA2.2 Subsystem Register Manual

- IDMA Register Mapping Summary: [1]

Table 14-162. IBUSERR

Address Offset	**0x0000 0400**		
Physical address	0x0182 0400	Instance	IVA2.2 GEMIDMA
Description	Bus Access Error Register		
Type	R		



Bits	Field Name	Description	Type	Reset
31:29	ERR	Error detected 0x0: No error 0x1: MDMA Read Status Error detected 0x2: MDMA Write Status Error detected 0x3: CFG Read Status Error detected 0x4: CFG Write Status Error detected	R	0x00000
28:11	Reserved	Write 0s for future compatibility. Read returns 0.	R	0x00
10:8	XID	Transaction ID Stores the Transaction ID when an error is detected on the response. Value should match the command id for the access that resulted in an error.	R	0x0

Bits	Field Name	Description	Type	Reset
7:3	Reserved	Write 0s for future compatibility. Read returns 0.	R	0x00
2:0	STAT	Transaction Status Stores the non-zero status/error code that wasdetected on the response to an access 0x0: Success (should not cause error to be latched), or unrecognized RID/WID (should cause error to be latched) 0x1: Addressing error 0x2: Privilege error 0x3: Timeout error 0x4: Data error 0x7: Exclusive-operation failure	R	0x0

Table 14-163. Register Call Summary for Register IBUSERR

IVA2.2 Subsystem Basic Programming Model • Error Reporting for IDMA Module: [0] [1] [2] [3] [4] [5] [6]
IVA2.2 Subsystem Register Manual • IDMA Register Mapping Summary: [7]

Table 14-164. IBUSERRCLR

Address Offset	0x0000 0404		
Physical address	0x0182 0404	**Instance**	IVA2.2 GEMIDMA
Description	Bus Access Error Clear		
Type	W		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																															CLR

Bits	Field Name	Description	Type	Reset
31:1	Reserved	Write 0s for future compatibility. Read returns 0.	W	0x00000000
0	CLR	Clear Error CLR = 0: Writes of 0 have no effect. CLR = 1: Write of 1 clears all bits in the IBUSERRregister. Once an error is detected, the MDMA Error register must be cleared before additional errors canbe detected/stored.	W	0

Table 14-165. Register Call Summary for Register IBUSERRCLR

IVA2.2 Subsystem Basic Programming Model • Error Reporting for IDMA Module: [0]
IVA2.2 Subsystem Register Manual • IDMA Register Mapping Summary: [1]

14.5.4 XMC Registers

This section provides information about the XMC module, which contains the registers related to the three memory controllers of DSP megamodule: UMC, PMC, and DMC. Each register in the module is described separately below.

14.5.4.1 XMC Register Mapping Summary

Table 14-166. XMC Register Summary

Register Name	Type	Register Width (Bits)	Address Offset	Physical Address
L2CFG	RW	32	0x0000 0000	0x0184 0000
L1PCFG	RW	32	0x0000 0020	0x0184 0020
L1PCC	RW	32	0x0000 0024	0x0184 0024
L1DCFG	RW	32	0x0000 0040	0x0184 0040
L1DCC	RW	32	0x0000 0044	0x0184 0044
CPUARBU	RW	32	0x0000 1000	0x0184 1000
IDMAARBU	RW	32	0x0000 1004	0x0184 1004
SDMAARBU	RW	32	0x0000 1008	0x0184 1008
UCARBU	RW	32	0x0000 100C	0x0184 100C
CPUARBD	RW	32	0x0000 1040	0x0184 1040
IDMAARBD	RW	32	0x0000 1044	0x0184 1044
SDMAARBD	RW	32	0x0000 1048	0x0184 1048
UCARBD	RW	32	0x0000 104C	0x0184 104C
L2WBAR	W	32	0x0000 4000	0x0184 4000
L2WWC	RW	32	0x0000 4004	0x0184 4004
L2WIBAR	W	32	0x0000 4010	0x0184 4010
L2WIWC	RW	32	0x0000 4014	0x0184 4014
L2IBAR	W	32	0x0000 4018	0x0184 4018
L2IWC	RW	32	0x0000 401C	0x0184 401C
L1PIBAR	W	32	0x0000 4020	0x0184 4020
L1PIWC	RW	32	0x0000 4024	0x0184 4024
L1DWIBAR	W	32	0x0000 4030	0x0184 4030
L1DWIWC	RW	32	0x0000 4034	0x0184 4034
L1DWBAR	W	32	0x0000 4040	0x0184 4040
L1DWWC	RW	32	0x0000 4044	0x0184 4044
L1DIBAR	W	32	0x0000 4048	0x0184 4048
L1DIWC	RW	32	0x0000 404C	0x0184 404C
L2WB	RW	32	0x0000 5000	0x0184 5000
L2WBINV	RW	32	0x0000 5004	0x0184 5004
L2INV	RW	32	0x0000 5008	0x0184 5008
L1PINV	RW	32	0x0000 5028	0x0184 5028
L1DWB	RW	32	0x0000 5040	0x0184 5040
L1DWBINV	RW	32	0x0000 5044	0x0184 5044
L1DINV	RW	32	0x0000 5048	0x0184 5048
MARi[1]	RW (RO if i = 0...15)	32	0x0000 8000 + (0x4*i)	0x0184 8000 + (0x4*i)
L2MPFAR	R	32	0x0000 A000	0x0184 A000
L2MPFSR	R	32	0x0000 A004	0x0184 A004
L2MPFCR	W	32	0x0000 A008	0x0184 A008
L2MPPAj[2]	RW	32	0x0000 A200 + (0x4*j)	0x0184 A200 + (0x4*j)
L1PMPFAR	R	32	0x0000 A400	0x0184 A400
L1PMPFSR	R	32	0x0000 A404	0x0184 A404
L1PMPFCR	W	32	0x0000 A408	0x0184 A408
L1PMPPAk[3]	RW	32	0x0000 A600 + (0x4*k)	0x0184 A600 + (0x4*k)

(1) i = 0 to 255
(2) j = 0 to 64
(3) k = 0 to 32

Table 14-166. XMC Register Summary (continued)

Register Name	Type	Register Width (Bits)	Address Offset	Physical Address
L1DMPFAR	R	32	0x0000 AC00	0x0184 AC00
L1DMPFSR	R	32	0x0000 AC04	0x0184 AC04
L1DMPFCR	W	32	0x0000 AC08	0x0184 AC08
L1DMPPAk[3]	RW	32	0x0000 AE00 + (0x4*k)	0x0184 AE00 + (0x4*k)

14.5.4.2 XMC Register Descriptions

Table 14-167. L2CFG

Address Offset	0x0000 0000		
Physical address	0x0184 0000	**Instance**	IVA2.2 GEMXMC
Description	L2 cache configuration register		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved				NUM_MM				Reserved				MMID				Reserved					NOINIT	IP	ID	Reserved			L2CC		L2MODE		

Bits	Field Name	Description	Type	Reset
31:28	Reserved	Write 0s for future compatibility. Read returns 0.	R	0x0
27:24	NUM_MM	Number of megamodules -1 (always 0 for IVA2)	R	0x0
23:20	Reserved	Write 0s for future compatibility. Read returns 0.	R	0x0
19:16	MMID	Megamodule ID (always 0x0 for IVA2)	R	0x0
15:11	Reserved	Write 0s for future compatibility. Read returns 0.	W	0x00
10	NOINIT	No init upon cache config: when written '1', cache config is restored without re-initializing cache context (tags, validity bits) (this is assumed here that the restored cache settings are the same as prior to the execution of the IDLE instruction) when written '0', cache context is re-initialized (cache content is indirectly lost) this bit is always read as 0	W	0
9	IP	Global L1P invalidate (for backward compatibility,deprecated)	W	0
8	ID	Global L1D invalidate (for backward compatibility,deprecated)	W	0
7:5	Reserved	Write 0s for future compatibility. Read returns 0.	W	0x0
4:3	L2CC	L2 cache control 0x0: L2 cache operates normally 0x1: L2 Cache is frozen 0x2: L2 cache is bypassed	RW	0x0
2:0	L2MODE	L2 Configuration Register 0x0: 0KB of L2 Cache 0x1: 32KB of L2 Cache 0x2: 64KB of L2 Cache	RW	0x0

Table 14-168. Register Call Summary for Register L2CFG

IVA2.2 Subsystem Basic Programming Model

- IVA2.2 Boot Configuration: [0] [1] [2] [3] [4]
- Cache-Size Configuration: [5] [6] [7] [8]
- Cache Mode Configuration: [9] [10]
- External Memory: [11]
- Powering Down L2$ Memory While IVA2 is Active: [12] [13] [14] [15]

Table 14-168. Register Call Summary for Register L2CFG (continued)

IVA2.2 Subsystem Register Manual • XMC Register Mapping Summary: [16]

Table 14-169. L1PCFG

Address Offset	0x0000 0020		
Physical address	0x0184 0020	**Instance**	IVA2.2 GEMXMC
Description			
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																														L1PMODE	

Bits	Field Name	Description	Type	Reset
31:3	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0x00000000
2:0	L1PMODE	L1P Configuration Register	RW	0x0
		0x0: 0KB of L1P Cache		
		0x1: 4KB of L1P Cache		
		0x2: 8KB of L1P Cache		
		0x3: 16KB of L1P Cache		
		0x4: Maximum cache (32KB of L1P Cache)		

Table 14-170. Register Call Summary for Register L1PCFG

IVA2.2 Subsystem Basic Programming Model • IVA2.2 Boot Configuration: [0] [1] [2] [3] • Cache-Size Configuration: [4] [5] [6] [7] [8]
IVA2.2 Subsystem Register Manual • XMC Register Mapping Summary: [9]

Table 14-171. L1PCC

Address Offset	**0x0000 0024**		
Physical address	0x0184 0024	**Instance**	IVA2.2 GEMXMC
Description	L1P Configuration Register		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved													POPER			Reserved														OPER	

Bits	Field Name	Description	Type	Reset
31:19	Reserved	Write 0s for future compatibility. Read returns 0.	R	0x0000
18:16	POPER	Previous value of the OPER field	R	0x0
		Read 0x0: L1P cache operates normally		
		Read 0x1: L1P Cache is frozen		
15:3	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0x0000
2:0	OPER	DMC operation control	RW	0x0
		0x0: L1P cache operates normally		
		0x1: L1P Cache is frozen		

Table 14-172. Register Call Summary for Register L1PCC

IVA2.2 Subsystem Basic Programming Model
- Cache Mode Configuration: [0] [1]

IVA2.2 Subsystem Register Manual
- XMC Register Mapping Summary: [2]

Table 14-173. L1DCFG

Address Offset	0x0000 0040		
Physical address	0x0184 0040	**Instance**	IVA2.2 GEMXMC
Description	L1D resets to Maximal cache mode using dmc_default_cachemode tie-off input.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																														L1DMODE	

Bits	Field Name	Description	Type	Reset
31:3	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0x00000000
2:0	L1DMODE	L1D Configuration Register	RW	0x0
		0x0: 0KB of L1D Cache		
		0x1: 4KB of L1D Cache		
		0x2: 8KB of L1D Cache		
		0x3: 16KB of L1D Cache		
		0x4: 32KB of L1D Cache		
		0x5: Not used		
		0x6: Not used		
		0x7: Maximum cache (maps to 32KB of L1D Cache)		

Table 14-174. Register Call Summary for Register L1DCFG

IVA2.2 Subsystem Basic Programming Model
- IVA2.2 Boot Configuration: [0] [1] [2] [3]
- Cache-Size Configuration: [4] [5] [6] [7] [8]

IVA2.2 Subsystem Register Manual
- XMC Register Mapping Summary: [9]

Table 14-175. L1DCC

Address Offset	0x0000 0044		
Physical address	0x0184 0044	**Instance**	IVA2.2 GEMXMC
Description	L1D Configuration Register		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved													POPER			Reserved														OPER	

Bits	Field Name	Description	Type	Reset
31:19	Reserved	Write 0s for future compatibility. Read returns 0.	R	0x0000
18:16	POPER	Previous value of the OPER field	R	0x0
		Read 0x0: L1D cache operates normally		
		Read 0x1: L1D Cache is frozen		
15:3	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0x0000
2:0	OPER	DMC operation control	RW	0x0

Bits	Field Name	Description		Type	Reset
		0x0:	L1D cache operates normally		
		0x1:	L1D Cache is frozen		

Table 14-176. Register Call Summary for Register L1DCC

IVA2.2 Subsystem Basic Programming Model
- Cache Mode Configuration: [0] [1]

IVA2.2 Subsystem Register Manual
- XMC Register Mapping Summary: [2]

Table 14-177. CPUARBU

Address Offset	0x0000 1000		
Physical address	0x0184 1000	**Instance**	IVA2.2 GEMXMC
Description			
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved													PRI			Reserved								MAXWAIT							

Bits	Field Name	Description		Type	Reset
31:19	Reserved	Write 0s for future compatibility. Read returns 0.		RW	0x0000
18:16	PRI	Priority		RW	0x1
		0x0:	Highest priority		
		0x1:	2nd highest priority		
		0x2:	3rd highest priority		
		0x3:	4th highest priority		
		0x4:	5th highest priority		
		0x5:	6th highest priority		
		0x6:	7th highest priority		
		0x7:	Lowest priority		
15:6	Reserved	Write 0s for future compatibility. Read returns 0.		RW	0x000
5:0	MAXWAIT	Maximum Wait time (in UMC/EMC cycles)		RW	0x10
		0x0:	Always stalls due to higher priority requestor		
		0x1:	Maximum wait of 1 cycles (1/2 = 50% access)		
		0x2:	Maximum wait of 2 cycles (1/3 = 33% access)		
		0x4:	Maximum wait of 4 cycles (1/5 = 20% access)		
		0x8:	Maximum wait of 8 cycles (1/9 = 11% access)		
		0x10:	Maximum wait of 16 cycles (1/17 = 6% access)		
		0x20:	Maximum wait of 32 cycles (1/33 = 3% access)		

Table 14-178. Register Call Summary for Register CPUARBU

IVA2.2 Subsystem Basic Programming Model
- Internal Memory: [0]

IVA2.2 Subsystem Register Manual
- XMC Register Mapping Summary: [1]

Table 14-179. IDMAARBU

Address Offset	0x0000 1004		
Physical address	0x0184 1004	**Instance**	IVA2.2 GEMXMC
Description			
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																								MAXWAIT							

Bits	Field Name	Description	Type	Reset
31:6	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0x0000000
5:0	MAXWAIT	Maximum Wait time (in UMC/EMC cycles)	RW	0x10
		0x0: Always stalls due to higher priority requestor		
		0x1: Maximum wait of 1 cycles (1/2 = 50% access)		
		0x2: Maximum wait of 2 cycles (1/3 = 33% access)		
		0x4: Maximum wait of 4 cycles (1/5 = 20% access)		
		0x8: Maximum wait of 8 cycles (1/9 = 11% access)		
		0x10: Maximum wait of 16 cycles (1/17 = 6% access)		
		0x20: Maximum wait of 32 cycles (1/33 = 3% access)		

Table 14-180. Register Call Summary for Register IDMAARBU

IVA2.2 Subsystem Basic Programming Model
- Internal Memory: [0]

IVA2.2 Subsystem Register Manual
- XMC Register Mapping Summary: [1]

Table 14-181. SDMAARBU

Address Offset	0x0000 1008		
Physical address	0x0184 1008	Instance	IVA2.2 GEMXMC
Description			
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																								MAXWAIT							

Bits	Field Name	Description	Type	Reset
31:6	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0x0000000
5:0	MAXWAIT	Maximum Wait time (in UMC/EMC cycles)	RW	0x01
		0x0: Always stalls due to higher priority requestor		
		0x1: Maximum wait of 1 cycles (1/2 = 50% access)		
		0x2: Maximum wait of 2 cycles (1/3 = 33% access)		
		0x4: Maximum wait of 4 cycles (1/5 = 20% access)		
		0x8: Maximum wait of 8 cycles (1/9 = 11% access)		
		0x10: Maximum wait of 16 cycles (1/17 = 6% access)		
		0x20: Maximum wait of 32 cycles (1/33 = 3% access)		

Table 14-182. Register Call Summary for Register SDMAARBU

IVA2.2 Subsystem Basic Programming Model
- Internal Memory: [0]

Table 14-182. Register Call Summary for Register SDMAARBU (continued)

IVA2.2 Subsystem Register Manual • XMC Register Mapping Summary: [1]

Table 14-183. UCARBU

Address Offset	0x0000 100C		
Physical address	0x0184 100C	**Instance**	IVA2.2 GEMXMC
Description			
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																								MAXWAIT							

Bits	Field Name	Description	Type	Reset
31:6	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0x0000000
5:0	MAXWAIT	Maximum Wait time (in UMC/EMC cycles) 0x0: Always stalls due to higher priority requestor 0x1: Maximum wait of 1 cycles (1/2 = 50% access) 0x2: Maximum wait of 2 cycles (1/3 = 33% access) 0x4: Maximum wait of 4 cycles (1/5 = 20% access) 0x8: Maximum wait of 8 cycles (1/9 = 11% access) 0x10: Maximum wait of 16 cycles (1/17 = 6% access) 0x20: Maximum wait of 32 cycles (1/33 = 3% access)	RW	0x20

Table 14-184. Register Call Summary for Register UCARBU

IVA2.2 Subsystem Basic Programming Model • Internal Memory: [0]
IVA2.2 Subsystem Register Manual • XMC Register Mapping Summary: [1]

Table 14-185. CPUARBD

Address Offset	0x0000 1040		
Physical address	0x0184 1040	**Instance**	IVA2.2 GEMXMC
Description	CPU Arb Control		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved													PRI			Reserved								MAXWAIT							

Bits	Field Name	Description	Type	Reset
31:19	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0x0000
18:16	PRI	Priority 0x0: Highest priority 0x1: 2nd highest priority 0x2: 3rd highest priority 0x3: 4th highest priority 0x4: 5th highest priority 0x5: 6th highest priority 0x6: 7th highest priority 0x7: Lowest priority	RW	0x1

Bits	Field Name	Description	Type	Reset
15:6	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0x000
5:0	MAXWAIT	Maximum Wait time (in UMC/EMC cycles)	RW	0x10
		0x0: Always stalls due to higher priority requestor		
		0x1: Maximum wait of 1 cycles (1/2 = 50% access)		
		0x2: Maximum wait of 2 cycles (1/3 = 33% access)		
		0x4: Maximum wait of 4 cycles (1/5 = 20% access)		
		0x8: Maximum wait of 8 cycles (1/9 = 11% access)		
		0x10: Maximum wait of 16 cycles (1/17 = 6% access)		
		0x20: Maximum wait of 32 cycles (1/33 = 3% access)		

Table 14-186. Register Call Summary for Register CPUARBD

IVA2.2 Subsystem Basic Programming Model • Internal Memory: [0]
IVA2.2 Subsystem Register Manual • XMC Register Mapping Summary: [1]

Table 14-187. IDMAARBD

Address Offset	0x0000 1044		
Physical address	0x0184 1044	**Instance**	IVA2.2 GEMXMC
Description			
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																								MAXWAIT							

Bits	Field Name	Description	Type	Reset
31:6	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0x0000000
5:0	MAXWAIT	Maximum Wait time (in UMC/EMC cycles)	RW	0x10
		0x0: Always stalls due to higher priority requestor		
		0x1: Maximum wait of 1 cycles (1/2 = 50% access)		
		0x2: Maximum wait of 2 cycles (1/3 = 33% access)		
		0x4: Maximum wait of 4 cycles (1/5 = 20% access)		
		0x8: Maximum wait of 8 cycles (1/9 = 11% access)		
		0x10: Maximum wait of 16 cycles (1/17 = 6% access)		
		0x20: Maximum wait of 32 cycles (1/33 = 3% access)		

Table 14-188. Register Call Summary for Register IDMAARBD

IVA2.2 Subsystem Basic Programming Model • Internal Memory: [0]
IVA2.2 Subsystem Register Manual • XMC Register Mapping Summary: [1]

Table 14-189. SDMAARBD

Address Offset	0x0000 1048		
Physical address	0x0184 1048	Instance	IVA2.2 GEMXMC
Description			
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																								MAXWAIT							

Bits	Field Name	Description	Type	Reset
31:6	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0x0000000
5:0	MAXWAIT	Maximum Wait time (in UMC/EMC cycles)	RW	0x01
		0x0: Always stalls due to higher priority requestor		
		0x1: Maximum wait of 1 cycles (1/2 = 50% access)		
		0x2: Maximum wait of 2 cycles (1/3 = 33% access)		
		0x4: Maximum wait of 4 cycles (1/5 = 20% access)		
		0x8: Maximum wait of 8 cycles (1/9 = 11% access)		
		0x10: Maximum wait of 16 cycles (1/17 = 6% access)		
		0x20: Maximum wait of 32 cycles (1/33 = 3% access)		

Table 14-190. Register Call Summary for Register SDMAARBD

IVA2.2 Subsystem Basic Programming Model
- Internal Memory: [0]

IVA2.2 Subsystem Register Manual
- XMC Register Mapping Summary: [1]

Table 14-191. UCARBD

Address Offset	0x0000 104C		
Physical address	0x0184 104C	**Instance**	IVA2.2 GEMXMC
Description			
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																								MAXWAIT							

Bits	Field Name	Description	Type	Reset
31:6	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0x0000000
5:0	MAXWAIT	Maximum Wait time (in UMC/EMC cycles)	RW	0x20
		0x0: Always stalls due to higher priority requestor		
		0x1: Maximum wait of 1 cycles (1/2 = 50% access)		
		0x2: max wait of 2 cycles (1/3 = 33% access)		
		0x4: Maximum wait of 4 cycles (1/5 = 20% access)		
		0x8: Maximum wait of 8 cycles (1/9 = 11% access)		
		0x10: Maximum wait of 16 cycles (1/17 = 6% access)		
		0x20: Maximum wait of 32 cycles (1/33 = 3% access)		

Table 14-192. Register Call Summary for Register UCARBD

IVA2.2 Subsystem Basic Programming Model
- Internal Memory: [0]

Table 14-192. Register Call Summary for Register UCARBD (continued)

IVA2.2 Subsystem Register Manual • XMC Register Mapping Summary: [1]

Table 14-193. L2WBAR

Address Offset	0x0000 4000		
Physical address	0x0184 4000	**Instance**	IVA2.2 GEMXMC
Description	L2 block writeback base address		
Type	W		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDR																															

Bits	Field Name	Description	Type	Reset
31:0	ADDR	Block base address	W	0x--------

Table 14-194. Register Call Summary for Register L2WBAR

IVA2.2 Subsystem Basic Programming Model • Coherence Maintenance: [0]
IVA2.2 Subsystem Register Manual • XMC Register Mapping Summary: [1]

Table 14-195. L2WWC

Address Offset	0x0000 4004		
Physical address	0x0184 4004	**Instance**	IVA2.2 GEMXMC
Description			
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																WC															

Bits	Field Name	Description	Type	Reset
31:16	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0x0000
15:0	WC	Number of 32-bit words in the block	RW	0x0000

Table 14-196. Register Call Summary for Register L2WWC

IVA2.2 Subsystem Basic Programming Model • Coherence Maintenance: [0]
IVA2.2 Subsystem Register Manual • XMC Register Mapping Summary: [1]

Table 14-197. L2WIBAR

Address Offset	0x0000 4010		
Physical address	0x0184 4010	Instance	IVA2.2 GEMXMC
Description	L2 block wbinv base address		
Type	W		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDR																															

Bits	Field Name	Description	Type	Reset
31:0	ADDR	Block base address	W	0x--------

Table 14-198. Register Call Summary for Register L2WIBAR

IVA2.2 Subsystem Basic Programming Model
- Coherence Maintenance: [0]

IVA2.2 Subsystem Register Manual
- XMC Register Mapping Summary: [1]

Table 14-199. L2WIWC

Address Offset	0x0000 4014		
Physical address	0x0184 4014	**Instance**	IVA2.2 GEMXMC
Description			
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																WC															

Bits	Field Name	Description	Type	Reset
31:16	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0x0000
15:0	WC	Number of 32-bit words in the block	RW	0x0000

Table 14-200. Register Call Summary for Register L2WIWC

IVA2.2 Subsystem Basic Programming Model
- Coherence Maintenance: [0]

IVA2.2 Subsystem Register Manual
- XMC Register Mapping Summary: [1]

Table 14-201. L2IBAR

Address Offset	0x0000 4018		
Physical address	0x0184 4018	**Instance**	IVA2.2 GEMXMC
Description			
Type	W		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDR																															

Bits	Field Name	Description	Type	Reset
31:0	ADDR		W	0x--------

Table 14-202. Register Call Summary for Register L2IBAR

IVA2.2 Subsystem Register Manual

- XMC Register Mapping Summary: [0]

Table 14-203. L2IWC

Address Offset	0x0000 401C		
Physical address	0x0184 401C	**Instance**	IVA2.2 GEMXMC
Description			
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																WC															

Bits	Field Name	Description	Type	Reset
31:16	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0x0000
15:0	WC	Number of 32-bit words in the block	RW	0x0000

Table 14-204. Register Call Summary for Register L2IWC

IVA2.2 Subsystem Basic Programming Model

- Coherence Maintenance: [0] [1]

IVA2.2 Subsystem Register Manual

- XMC Register Mapping Summary: [2]

Table 14-205. L1PIBAR

Address Offset	0x0000 4020		
Physical address	0x0184 4020	**Instance**	IVA2.2 GEMXMC
Description	L1P Block Invalidate Base Address Register		
Type	W		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDR																															

Bits	Field Name	Description	Type	Reset
31:0	ADDR	Block base address	W	0x--------

Table 14-206. Register Call Summary for Register L1PIBAR

IVA2.2 Subsystem Basic Programming Model

- Coherence Maintenance: [0]

IVA2.2 Subsystem Register Manual

- XMC Register Mapping Summary: [1]

Table 14-207. L1PIWC

Address Offset	0x0000 4024		
Physical address	0x0184 4024	**Instance**	IVA2.2 GEMXMC
Description	L1P Block Invalidate Word Count		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																WC															

Bits	Field Name	Description	Type	Reset
31:16	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0x0000
15:0	WC	Number of 32-bit words in the block	RW	0x0000

Table 14-208. Register Call Summary for Register L1PIWC

IVA2.2 Subsystem Basic Programming Model
- Coherence Maintenance: [0]

IVA2.2 Subsystem Register Manual
- XMC Register Mapping Summary: [1]

Table 14-209. L1DWIBAR

Address Offset	0x0000 4030		
Physical address	0x0184 4030	**Instance**	IVA2.2 GEMXMC
Description			
Type	W		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDR																															

Bits	Field Name	Description	Type	Reset
31:0	ADDR	Block base address	W	0x--------

Table 14-210. Register Call Summary for Register L1DWIBAR

IVA2.2 Subsystem Basic Programming Model
- Coherence Maintenance: [0]

IVA2.2 Subsystem Register Manual
- XMC Register Mapping Summary: [1]

Table 14-211. L1DWIWC

Address Offset	0x0000 4034		
Physical address	0x0184 4034	**Instance**	IVA2.2 GEMXMC
Description	L1D Block Wb-Inv Word Count		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																WC															

Bits	Field Name	Description	Type	Reset
31:16	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0x0000
15:0	WC	Number of 32-bit words in the block	RW	0x0000

Table 14-212. Register Call Summary for Register L1DWIWC

IVA2.2 Subsystem Basic Programming Model
- Coherence Maintenance: [0]

IVA2.2 Subsystem Register Manual
- XMC Register Mapping Summary: [1]

Table 14-213. L1DWBAR

Address Offset	0x0000 4040		
Physical address	0x0184 4040	**Instance**	IVA2.2 GEMXMC
Description	L1D Block Writeback Base Address Register		
Type	W		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDR																															

Bits	Field Name	Description	Type	Reset
31:0	ADDR	Block base address	W	0x--------

Table 14-214. Register Call Summary for Register L1DWBAR

IVA2.2 Subsystem Basic Programming Model
- Coherence Maintenance: [0]

IVA2.2 Subsystem Register Manual
- XMC Register Mapping Summary: [1]

Table 14-215. L1DWWC

Address Offset	0x0000 4044		
Physical address	0x0184 4044	**Instance**	IVA2.2 GEMXMC
Description	L1D Block Writeback Word Count		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																WC															

Bits	Field Name	Description	Type	Reset
31:16	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0x0000
15:0	WC	Number of 32-bit words in the block	RW	0x0000

Table 14-216. Register Call Summary for Register L1DWWC

IVA2.2 Subsystem Basic Programming Model
- Coherence Maintenance: [0]

IVA2.2 Subsystem Register Manual
- XMC Register Mapping Summary: [1]

Table 14-217. L1DIBAR

Address Offset	0x0000 4048		
Physical address	0x0184 4048	**Instance**	IVA2.2 GEMXMC
Description	L1D Block Invalidate Base Address Register		
Type	W		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDR																															

Bits	Field Name	Description	Type	Reset
31:0	ADDR	Block base address	W	0x--------

Table 14-218. Register Call Summary for Register L1DIBAR

IVA2.2 Subsystem Basic Programming Model

- Coherence Maintenance: [0]

IVA2.2 Subsystem Register Manual

- XMC Register Mapping Summary: [1]

Table 14-219. L1DIWC

Address Offset	0x0000 404C		
Physical address	0x0184 404C	**Instance**	IVA2.2 GEMXMC
Description	L1D Block Invalidate Word Count		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																WC															

Bits	Field Name	Description	Type	Reset
31:16	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0x0000
15:0	WC	Number of 32-bit words in the block	RW	0x0000

Table 14-220. Register Call Summary for Register L1DIWC

IVA2.2 Subsystem Basic Programming Model

- Coherence Maintenance: [0]

IVA2.2 Subsystem Register Manual

- XMC Register Mapping Summary: [1]

Table 14-221. L2WB

Address Offset	0x0000 5000		
Physical address	0x0184 5000	Instance	IVA2.2 GEMXMC
Description	L2 global writeback		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																															C

Bits	Field Name	Description	Type	Reset
31:1	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0x--------
0	C	L2 global write-back control	RW	0
		Read 0x0: Read 0: Previous L2 global write-back has completed		

Bits	Field Name	Description		Type	Reset
		Write 0x0:	Write 0: No effect		
		Read 0x1:	Read 1: Previous L2 global write-back has not completed		
		Write 0x1:	Write 1: Initiates an L2 global write-back(L1P Effect: No effect. L1D Effect: All updated data written back to L2/external, but left valid in L1D. L2 Effect: All updated data written back externally, but left valid in L2 cache.)		

Table 14-222. Register Call Summary for Register L2WB

IVA2.2 Subsystem Basic Programming Model • Coherence Maintenance: [0]
IVA2.2 Subsystem Register Manual • XMC Register Mapping Summary: [1]

Table 14-223. L2WBINV

Address Offset	0x0000 5004		
Physical address	0x0184 5004	**Instance**	IVA2.2 GEMXMC
Description	L2 global writeback invalidate		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																															C

Bits	Field Name	Description	Type	Reset
31:1	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0x--------
0	C	L2 global write-back/invalidate command:	RW	0
		Write 0: No effect		
		Write 1: Initiates an L2 global write-back/invalidate(L1P Effect: All lines invalidated in L1P. L1D Effect: All updated data written back to L2/external. All lines invalidated within L1D. L2 Effect: All updated data written back externally. All lines invalidated in L2).		
		Read 0: Previous L2 global write-back/invalidate has completed		
		Read 1: Previous L2 global write-back/invalidate has not completed		

Table 14-224. Register Call Summary for Register L2WBINV

IVA2.2 Subsystem Basic Programming Model • Coherence Maintenance: [0]
IVA2.2 Subsystem Register Manual • XMC Register Mapping Summary: [1]

Table 14-225. L2INV

Address Offset	0x0000 500		
Physical address	0x0184 5008	**Instance**	IVA2.2 GEMXMC
Description	L2 global invalidate		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																															I

Bits	Field Name	Description	Type	Reset
31:1	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0x--------
0	I	L2 global invalidate command: Write 0: No effect Write 1: Initiates an L2 global invalidate(L1P Effect: All lines invalidated in L1P. L1D Effect: All lines invalidated in L1D. Updated data is dropped. L2 Effect: All lines invalidated in L2. Updated data is dropped). Read 0: Previous L2 global invalidate has completed Read 1: Previous L2 global invalidate has notcompleted	RW	0

Table 14-226. Register Call Summary for Register L2INV

IVA2.2 Subsystem Basic Programming Model
- Coherence Maintenance: [0] [1]

IVA2.2 Subsystem Register Manual
- XMC Register Mapping Summary: [2]

Table 14-227. L1PINV

Address Offset	0x0000 5028		
Physical address	0x0184 5028	**Instance**	IVA2.2 GEMXMC
Description	L1P Global Invalidate		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																															I

Bits	Field Name	Description	Type	Reset
31:1	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0x--------
0	I	L1P global invalidate command: Write 0: No effect Write 1: Initiates an L1P global invalidate Read 0: Previous L1P global invalidate has completed Read 1: Previous L1P global invalidate has notcompleted	RW	0

Table 14-228. Register Call Summary for Register L1PINV

IVA2.2 Subsystem Basic Programming Model
- Coherence Maintenance: [0]

IVA2.2 Subsystem Register Manual
- XMC Register Mapping Summary: [1]

Table 14-229. L1DWB

Address Offset	0x0000 5040		
Physical address	0x0184 5040	**Instance**	IVA2.2 GEMXMC
Description	L1D Global Writeback		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																															C

Bits	Field Name	Description	Type	Reset
31:1	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0x--------
0	C	L1D global write-back command: Write 0: No effect Write 1: Initiates an L1D global write-back Read 0: Previous L1D global write-back has completed Read 1: Previous L1D global write-back has notcompleted	RW	0

Table 14-230. Register Call Summary for Register L1DWB

IVA2.2 Subsystem Basic Programming Model
- Coherence Maintenance: [0]

IVA2.2 Subsystem Register Manual
- XMC Register Mapping Summary: [1]

Table 14-231. L1DWBINV

Address Offset	0x0000 5044		
Physical address	0x0184 5044	**Instance**	IVA2.2 GEMXMC
Description	L1D Global Writeback Invalidate		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																															C

Bits	Field Name	Description	Type	Reset
31:1	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0x--------
0	C	L1D global write-back/invalidate command: Write 0: No effect Write 1: Initiates an L1D global write-back/invalidate Read 0: Previous L1D global write-back/invalidate has completed Read 1: Previous L1D global write-back/invalidate has not completed	RW	0

Table 14-232. Register Call Summary for Register L1DWBINV

IVA2.2 Subsystem Basic Programming Model
- Coherence Maintenance: [0]

IVA2.2 Subsystem Register Manual
- XMC Register Mapping Summary: [1]

Table 14-233. L1DINV

Address Offset	0x0000 5048		
Physical address	0x0184 5048	**Instance**	IVA2.2 GEMXMC
Description	L1D Global Invalidate		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																															I

Bits	Field Name	Description	Type	Reset
31:1	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0x--------
0	I	L1D global invalidate command: Write 0: No effect Write 1: Initiates an L1D global invalidate Read 0: Previous L1D global invalidate has completed Read 1: Previous L1D global invalidate has notcompleted	RW	0

Table 14-234. Register Call Summary for Register L1DINV

IVA2.2 Subsystem Basic Programming Model
- Coherence Maintenance: [0]

IVA2.2 Subsystem Register Manual
- XMC Register Mapping Summary: [1]

Table 14-235. MARi

Address Offset	0x8000 + (0x4*i)		
Physical address	0x0184 8000 + (0x4*i)	Instance	IVA2.2 GEMXMC
Description	Memory Attribute Register i = 0 defines the cacheable memory attribute for Local L2 RAM (fixed) i = 1 to 255 define a cachable memory attribute for 0x0100 0000 memory range starting at 0x0100 0000		
Type	RW (R for i=0...15)		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																															PC

Bits	Field Name	Description	Type	Reset
31:1	Reserved	Write 0s for future compatibility. Read returns 0.	R	0x--------
0	PC		RW (R for i=0...15)	0 (1 for i=0)
		Read 0x0: Region is not cached		
		Read 0x1: Region is cached		

Table 14-236. Register Call Summary for Register MARi

IVA2.2 Subsystem Basic Programming Model
- IVA2.2 Boot Configuration: [0] [1]
- External Memory: [2]

IVA2.2 Subsystem Register Manual
- XMC Register Mapping Summary: [3]

Table 14-237. L2MPFAR

Address Offset	0x0000 A000		
Physical address	0x0184 A000	**Instance**	IVA2.2 GEMXMC
Description	L2 Memory Protection Fault Address Register		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDR																															

Bits	Field Name	Description	Type	Reset
31:0	ADDR	Block base address	W	0x00000000

Table 14-238. Register Call Summary for Register L2MPFAR

IVA2.2 Subsystem Basic Programming Model
- Internal Memory: [0]

IVA2.2 Subsystem Register Manual
- XMC Register Mapping Summary: [1]

Table 14-239. L2MPFSR

Address Offset	0x0000 A004		
Physical address	0x0184 A004	**Instance**	IVA2.2 GEMXMC
Description			
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																FLTID								Reserved		ATYP					

Bits	Field Name	Description	Type	Reset
31:16	Reserved	Write 0s for future compatibility. Read returns 0.	R	0x0000
15:8	FLTID	Faulted ID: VBUS PrivID of faulting requestor. This field is valid only if LE is zero.	R	0x00
7:6	Reserved	Write 0s for future compatibility. Read returns 0.	R	0x0
5:0	ATYP	Access type	R	0x00

Table 14-240. Register Call Summary for Register L2MPFSR

IVA2.2 Subsystem Basic Programming Model
- Internal Memory: [0]

IVA2.2 Subsystem Register Manual
- XMC Register Mapping Summary: [1]

Table 14-241. L2MPFCR

Address Offset	0x0000 A008		
Physical address	0x0184 A008	**Instance**	IVA2.2 GEMXMC
Description	L2 Memory Protection Fault Command Register		
Type	W		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																															MPFCLR

Bits	Field Name	Description	Type	Reset
31:1	Reserved	Write 0s for future compatibility. Read returns 0.	W	0x00000000
0	MPFCLR	Write 0: No effect Write 1: Clear fault logged information	W	0

Table 14-242. Register Call Summary for Register L2MPFCR

IVA2.2 Subsystem Basic Programming Model

- Internal Memory: [0]

IVA2.2 Subsystem Register Manual

- XMC Register Mapping Summary: [1]

Table 14-243. L2MPPAj

Address Offset	0x0000 A200 + (0x4*j) in 0x4 byte increments		
Physical address	0x0184 A200 + (0x4*j)	**Instance**	IVA2.2 GEMXMC
Description	L2 Memory Protection Attribute Register Addresses for the 16MB page number i		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																AID5	AID4	AID3	AID2	AID1	AID0	AIDX	LOCAL	Reserved		SR	SW	SX	UR	UW	UX

Bits	Field Name	Description	Type	Reset
31:16	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0x0000
15	AID5	0: ID 5 does not have access permission 1: ID 5 has access permission	RW	1
14	AID4	0: ID 4 does not have access permission 1: ID 4 has access permission	RW	1
13	AID3	0: ID 3 does not have access permission 1: ID 3 has access permission	RW	1
12	AID2	0: ID 2 does not have access permission 1: ID 2 has access permission	RW	1
11	AID1	0: ID 1 does not have access permission 1: ID 1 has access permission	RW	1
10	AID0	0: ID 0 does not have access permission 1: ID 0 has access permission	RW	1
9	AIDX	0: External access is not permitted 1: External access is permitted	RW	1
8	LOCAL	0: DSP megamodule access is not permitted 1: DSP megamodule access is permitted	RW	1
7:6	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0
5	SR	0: Supervisor Read access is not permitted 1: Supervisor Read access is permitted	RW	1

Bits	Field Name	Description	Type	Reset
4	SW	0: Supervisor Write access is not permitted 1: Supervisor Write access is permitted	RW	1
3	SX	0: Supervisor eXecute access is not permitted 1: Supervisor eXecute access is permitted	RW	1
2	UR	0: User Read access is not permitted 1: User Read access is permitted	RW	1
1	UW	0: User Write access is not permitted 1: User Write access is permitted	RW	1
0	UX	0: User eXecute access is not permitted 1: User eXecute access is permitted	RW	1

Table 14-244. Register Call Summary for Register L2MPPAj

IVA2.2 Subsystem Basic Programming Model

- Internal Memory: [0] [1] [2] [3] [4]
- Powering Down L2$ Memory While IVA2 is Active: [5] [6] [7]

IVA2.2 Subsystem Register Manual

- XMC Register Mapping Summary: [8]

Table 14-245. L1PMPFAR

Address Offset	0x0000 A400		
Physical address	0x0184 A400	**Instance**	IVA2.2 GEMXMC
Description	PMC		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDR																															

Bits	Field Name	Description	Type	Reset
31:0	ADDR	Fault Address	R	0x00000000

Table 14-246. Register Call Summary for Register L1PMPFAR

IVA2.2 Subsystem Basic Programming Model

- Internal Memory: [0]

IVA2.2 Subsystem Register Manual

- XMC Register Mapping Summary: [1]

Table 14-247. L1PMPFSR

Address Offset	0x0000 A404		
Physical address	0x0184 A404	**Instance**	IVA2.2 GEMXMC
Description	PMC		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																FLTID								Reserved				ATYP			

Bits	Field Name	Description	Type	Reset
31:16	Reserved	Write 0s for future compatibility. Read returns 0.	R	0x0000
15:8	FLTID	Faulted ID: VBUS PrivID of faulting requestor. This field is valid only if LE is zero.	R	0x00
7:6	Reserved	Write 0s for future compatibility. Read returns 0.	R	0x0
5:0	ATYP	Access type	R	0x00
		Read 0x1: Invalid user program fetch.		
		Read 0x2: Invalid user write		
		Read 0x4: Invalid user read		
		Read 0x8: Invalid supervisor program fetch		
		Read 0x10: Invalid supervisor read		
		Read 0x12: Invalid cache line writeback		
		Read 0x20: Invalid supervisor write		
		Read 0x3F: Invalid cache line fill		

Table 14-248. Register Call Summary for Register L1PMPFSR

IVA2.2 Subsystem Basic Programming Model • Internal Memory: [0]
IVA2.2 Subsystem Register Manual • XMC Register Mapping Summary: [1]

Table 14-249. L1PMPFCR

Address Offset	0x0000 A408		
Physical address	0x0184 A408	**Instance**	IVA2.2 GEMXMC
Description	PMC		
Type	W		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																															MPFCLR

Bits	Field Name	Description	Type	Reset
31:1	Reserved	Write 0s for future compatibility. Read returns 0.	W	0x00000000
0	MPFCLR	Write 0: No effect Write 1: Clear fault logged information	W	0

Table 14-250. Register Call Summary for Register L1PMPFCR

IVA2.2 Subsystem Basic Programming Model • Internal Memory: [0]
IVA2.2 Subsystem Register Manual • XMC Register Mapping Summary: [1]

Table 14-251. L1PMPPAk

Address Offset	0x0000 A600 + (0x4*k)		
Physical address	0x0184 A600 + (0x4*k)	**Instance**	IVA2.2 GEMXMC
Description	L1P Memory Protection Attribute Register Addresses for the 16MB page number i		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																AID5	AID4	AID3	AID2	AID1	AID0	AIDX	LOCAL	Reserved		SR	SW	SX	UR	UW	UX

Bits	Field Name	Description	Type	Reset
31:16	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0x0000
15	AID5	0: ID 5 does not have access permission 1: ID 5 has access permission	RW	1
14	AID4	0: ID 4 does not have access permission 1: ID 4 has access permission	RW	1
13	AID3	0: ID 3 does not have access permission 1: ID 3 has access permission	RW	1
12	AID2	0: ID 2 does not have access permission 1: ID 2 has access permission	RW	1
11	AID1	0: ID 1 does not have access permission 1: ID 1 has access permission	RW	1
10	AID0	0: ID 0 does not have access permission 1: ID 0 has access permission	RW	1
9	AIDX	0: External access is not permitted 1: External access is permitted	RW	1
8	LOCAL	0: DSP megamodule access is not permitted 1: DSP megamodule access is permitted	RW	1
7:6	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0
5	SR	0: Supervisor Read access is not permitted 1: Supervisor Read access is permitted	RW	1
4	SW	0: Supervisor Write access is not permitted 1: Supervisor Write access is permitted	RW	1
3	SX	0: Supervisor eXecute access is not permitted 1: Supervisor eXecute access is permitted	RW	1
2	UR	0: User Read access is not permitted 1: User Read access is permitted	RW	1
1	UW	0: User Write access is not permitted 1: User Write access is permitted	RW	1
0	UX	0: User eXecute access is not permitted 1: User eXecute access is permitted	RW	1

Table 14-252. Register Call Summary for Register L1PMPPAk

IVA2.2 Subsystem Basic Programming Model

- Internal Memory: [0] [1] [2] [3] [4]

IVA2.2 Subsystem Register Manual

- XMC Register Mapping Summary: [5]

Table 14-253. L1DMPFAR

Address Offset	0x0000 AC00		
Physical address	0x0184 AC00	**Instance**	IVA2.2 GEMXMC
Description	L1D Memory Protection Fault Address Register		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDR																															

Bits	Field Name	Description	Type	Reset
31:0	ADDR	Fault Address	R	0x00000000

Table 14-254. Register Call Summary for Register L1DMPFAR

IVA2.2 Subsystem Basic Programming Model
- Internal Memory: [0]

IVA2.2 Subsystem Register Manual
- XMC Register Mapping Summary: [1]

Table 14-255. L1DMPFSR

Address Offset	0x0000 AC04		
Physical address	0x0184 AC04	**Instance**	IVA2.2 GEMXMC
Description	L1D Memory Protection Fault Status Register		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																FLTID								Reserved		ATYP					

Bits	Field Name	Description	Type	Reset
31:16	Reserved	Write 0s for future compatibility. Read returns 0.	R	0x0000
15:8	FLTID	Faulted ID: VBUS PrivID of faulting requestor. This field is valid only if LE is zero.	R	0x00
7:6	Reserved	Write 0s for future compatibility. Read returns 0.	R	0x0
5:0	ATYP	Access type	R	0x00
		Read 0x1: Invalid user program fetch.		
		Read 0x2: Invalid user write		
		Read 0x4: Invalid user read		
		Read 0x8: Invalid supervisor program fetch		
		Read 0x10: Invalid supervisor read		
		Read 0x12: Invalid cache line writeback		
		Read 0x20: Invalid supervisor write		
		Read 0x3F: Invalid cache line fill		

Table 14-256. Register Call Summary for Register L1DMPFSR

IVA2.2 Subsystem Basic Programming Model
- Internal Memory: [0]

IVA2.2 Subsystem Register Manual
- XMC Register Mapping Summary: [1]

Table 14-257. L1DMPFCR

Address Offset	0x0000 AC08		
Physical address	0x0184 AC08	**Instance**	IVA2.2 GEMXMC
Description	L1D Memory Protection Fault Command Register		
Type	W		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																															MPFCLR

Bits	Field Name	Description	Type	Reset
31:1	Reserved	Write 0s for future compatibility. Read returns 0.	W	0x00000000
0	MPFCLR	Write 0: No effect Write 1: Clear fault logged information	W	0

Table 14-258. Register Call Summary for Register L1DMPFCR

IVA2.2 Subsystem Basic Programming Model

- Internal Memory: [0]

IVA2.2 Subsystem Register Manual

- XMC Register Mapping Summary: [1]

Table 14-259. L1DMPPAk

Address Offset	0x0000 AE00 + (0x4*k)		
Physical address	0x0184 AE00 + (0x4*k)	**Instance**	IVA2.2 GEMXMC
Description	L1D Memory Protection Attribute Register Addresses for the 16MB page number i		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																AID5	AID4	AID3	AID2	AID1	AID0	AIDX	LOCAL	Reserved		SR	SW	Reserved	UR	UW	Reserved

Bits	Field Name	Description	Type	Reset
31:16	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0x0000
15	AID5	0: ID 5 does not have access permission 1: ID 5 has access permission	RW	1
14	AID4	0: ID 4 does not have access permission 1: ID 4 has access permission	RW	1
13	AID3	0: ID 3 does not have access permission 1: ID 3 has access permission	RW	1
12	AID2	0: ID 2 does not have access permission 1: ID 2 has access permission	RW	1
11	AID1	0: ID 1 does not have access permission 1: ID 1 has access permission	RW	1
10	AID0	0: ID 0 does not have access permission 1: ID 0 has access permission	RW	1
9	AIDX	0: External access is not permitted 1: External access is permitted	RW	1
8	LOCAL	0: DSP megamodule access is not permitted 1: DSP megamodule access is permitted	RW	1

Bits	Field Name	Description	Type	Reset
7:6	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0
5	SR	0: Supervisor Read access is not permitted 1: Supervisor Read access is permitted	RW	1
4	SW	0: Supervisor Write access is not permitted 1: Supervisor Write access is permitted	RW	1
3	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0
2	UR	0: User Read access is not permitted 1: User Read access is permitted	RW	1
1	UW	0: User Write access is not permitted 1: User Write access is permitted	RW	1
0	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0

Table 14-260. Register Call Summary for Register L1DMPPAk

IVA2.2 Subsystem Basic Programming Model • Internal Memory: [0] [1] [2] [3] [4] [5]
IVA2.2 Subsystem Register Manual • XMC Register Mapping Summary: [6]

14.5.5 TPCC Registers

This section provides information about the TPCC module. Each register in the module is described separately below.

14.5.5.1 TPCC Register Mapping Summary

Table 14-261. TPCC Register Summary

Register Name	Type	Register Width (Bits)	Address Offset	Physical Address
TPCC_PID	R	32	0x0000	0x01C0 0000
TPCC_CCCFG	R	32	0x0004	0x01C0 0004
TPCC_DCHMAPi(1)	RW	32	0x0100 + (0x4*i)	0x01C0 0100 + (0x4*i)
TPCC_QCHMAPj(2)	RW	32	0x0200 + (0x4*j)	0x01C0 0200 + (0x4*j)
TPCC_DMAQNUM0	RW	32	0x0240	0x01C0 0240
TPCC_DMAQNUM1	RW	32	0x0244	0x01C0 0244
TPCC_DMAQNUM2	RW	32	0x0248	0x01C0 0248
TPCC_DMAQNUM3	RW	32	0x024C	0x01C0 024C
TPCC_DMAQNUM4	RW	32	0x0250	0x01C0 0250
TPCC_DMAQNUM5	RW	32	0x0254	0x01C0 0254
TPCC_DMAQNUM6	RW	32	0x0258	0x01C0 0258
TPCC_DMAQNUM7	RW	32	0x025C	0x01C0 025C
TPCC_QDMAQNUM	RW	32	0x0260	0x01C0 0260
TPCC_QUETCMAP	RW	32	0x0280	0x01C0 0280
TPCC_QUEPRI	RW	32	0x0284	0x01C0 0284
TPCC_EMR	R	32	0x0300	0x01C0 0300
TPCC_EMRH	R	32	0x0304	0x01C0 0304
TPCC_EMCR	W	32	0x0308	0x01C0 0308
TPCC_EMCRH	W	32	0x030C	0x01C0 030C
TPCC_QEMR	R	32	0x0310	0x01C0 0310

(1) i = 0 to 63
(2) j = 0 to 7

Table 14-261. TPCC Register Summary (continued)

Register Name	Type	Register Width (Bits)	Address Offset	Physical Address
TPCC_QEMCR	W	32	0x0314	0x01C0 0314
TPCC_CCERR	R	32	0x0318	0x01C0 0318
TPCC_CCERRCLR	W	32	0x031C	0x01C0 031C
TPCC_EEVAL	W	32	0x0320	0x01C0 0320
TPCC_DRAEj(2)	RW	32	0x0340 + (0x8*j)	0x01C0 0340 + (0x8*j)
TPCC_DRAEHj(2)	RW	32	0x0344 + (0x8*j)	0x01C0 0344 + (0x8*j)
TPCC_QRAEj(2)	RW	32	0x0380 + (0x4*j)	0x01C0 0380 + (0x4*j)
TPCC_Q0Ek(3)	R	32	0x0400 + (0x4*k)	0x01C0 0400 + (0x4*k)
TPCC_Q1Ek(3)	R	32	0x0440 + (0x4*k)	0x01C0 0440 + (0x4*k)
TPCC_QSTATl(4)	R	32	0x0600 + (0x4*l)	0x01C0 0600 + (0x4*l)
TPCC_QWMTHRA	RW	32	0x0620	0x01C0 0620
TPCC_QWMTHRB	RW	32	0x0624	0x01C0 0624
TPCC_CCSTAT	R	32	0x0640	0x01C0 0640
TPCC_MPFAR	R	32	0x0800	0x01C0 0800
TPCC_MPFSR	R	32	0x0804	0x01C0 0804
TPCC_MPFCR	W	32	0x0808	0x01C0 0808
TPCC_MPPAG	RW	32	0x080C	0x01C0 080C
TPCC_MPPAj(2)	RW	32	0x0810 + (0x4*j)	0x01C0 0810 + (0x4*j)
TPCC_ER	R	32	0x1000	0x01C0 1000
TPCC_ECR	W	32	0x1008	0x01C0 1008
TPCC_ECRH	W	32	0x100C	0x01C0 100C
TPCC_ESR	W	32	0x1010	0x01C0 1010
TPCC_ESRH	W	32	0x1014	0x01C0 1014
TPCC_CER	R	32	0x1018	0x01C0 1018
TPCC_CERH	R	32	0x101C	0x01C0 101C
TPCC_EER	R	32	0x1020	0x01C0 1020
TPCC_EECR	W	32	0x1028	0x01C0 1028
TPCC_EESR	W	32	0x1030	0x01C0 1030
TPCC_SER	R	32	0x1038	0x01C0 1038
TPCC_SERH	R	32	0x103C	0x01C0 103C
TPCC_SECR	W	32	0x1040	0x01C0 1040
TPCC_SECRH	W	32	0x1044	0x01C0 1044
TPCC_IER	R	32	0x1050	0x01C0 1050
TPCC_IERH	R	32	0x1054	0x01C0 1054
TPCC_IECR	W	32	0x1058	0x01C0 1058
TPCC_IECRH	W	32	0x105C	0x01C0 105C
TPCC_IESR	W	32	0x1060	0x01C0 1060
TPCC_IESRH	W	32	0x1064	0x01C0 1064
TPCC_IPR	R	32	0x1068	0x01C0 1068
TPCC_IPRH	R	32	0x106C	0x01C0 106C
TPCC_ICR	W	32	0x1070	0x01C0 1070
TPCC_ICRH	W	32	0x1074	0x01C0 1074
TPCC_IEVAL	W	32	0x1078	0x01C0 1078
TPCC_QER	R	32	0x1080	0x01C0 1080
TPCC_QEER	R	32	0x1084	0x01C0 1084

(3) k = 0 to 15
(4) l = 0 to 1

Table 14-261. TPCC Register Summary (continued)

Register Name	Type	Register Width (Bits)	Address Offset	Physical Address
TPCC_QEECR	W	32	0x1088	0x01C0 1088
TPCC_QEESR	W	32	0x108C	0x01C0 108C
TPCC_QSER	R	32	0x1090	0x01C0 1090
TPCC_QSECR	W	32	0x1094	0x01C0 1094
TPCC_ER_Rn(5)	R	32	0x2000 + (0x200*n)	0x01C0 2000 + (0x200*n)
TPCC_ECR_Rn(5)	W	32	0x2008 + (0x200*n)	0x01C0 2008 + (0x200*n)
TPCC_ECRH_Rn(5)	W	32	0x200C + (0x200*n)	0x01C0 200C + (0x200*n)
TPCC_ESR_Rn(5)	W	32	0x2010 + (0x200*n)	0x01C0 2010 + (0x200*n)
TPCC_ESRH_Rn(5)	W	32	0x2014 + (0x200*n)	0x01C0 2014 + (0x200*n)
TPCC_CER_Rn(5)	R	32	0x2018 + (0x200*n)	0x01C0 2018 + (0x200*n)
TPCC_CERH_Rn(5)	R	32	0x201C + (0x200*n)	0x01C0 201C + (0x200*n)
TPCC_EER_Rn(5)	R	32	0x2020 + (0x200*n)	0x01C0 2020 + (0x200*n)
TPCC_EECR_Rn(5)	W	32	0x2028 + (0x200*n)	0x01C0 2028 + (0x200*n)
TPCC_EESR_Rn(5)	W	32	0x2030 + (0x200*n)	0x01C0 2030 + (0x200*n)
TPCC_SER_Rn(5)	R	32	0x2038 + (0x200*n)	0x01C0 2038 + (0x200*n)
TPCC_SERH_Rn(5)	R	32	0x203C + (0x200*n)	0x01C0 203C + (0x200*n)
TPCC_SECR_Rn(5)	W	32	0x2040 + (0x200*n)	0x01C0 2040 + (0x200*n)
TPCC_SECRH_Rn(5)	W	32	0x2044 + (0x200*n)	0x01C0 2044 + (0x200*n)
TPCC_IER_Rn(5)	R	32	0x2050 + (0x200*n)	0x01C0 2050 + (0x200*n)
TPCC_IERH_Rn(5)	R	32	0x2054 + (0x200*n)	0x01C0 2054 + (0x200*n)
TPCC_IECR_Rn(5)	W	32	0x2058 + (0x200*n)	0x01C0 2058 + (0x200*n)
TPCC_IECRH_Rn(5)	W	32	0x205C + (0x200*n)	0x01C0 205C + (0x200*n)
TPCC_IESR_Rn(5)	W	32	0x2060 + (0x200*n)	0x01C0 2060 + (0x200*n)
TPCC_IESRH_Rn(5)	W	32	0x2064 + (0x200*n)	0x01C0 2064 + (0x200*n)
TPCC_IPR_Rn(5)	R	32	0x2068 + (0x200*n)	0x01C0 2068 + (0x200*n)
TPCC_IPRH_Rn(5)	R	32	0x206C + (0x200*n)	0x01C0 206C + (0x200*n)
TPCC_ICR_Rn(5)	W	32	0x2070 + (0x200*n)	0x01C0 2070 + (0x200*n)
TPCC_ICRH_Rn(5)	W	32	0x2074 + (0x200*n)	0x01C0 2074 + (0x200*n)
TPCC_IEVAL_Rn(5)	W	32	0x2078 + (0x200*n)	0x01C0 2078 + (0x200*n)
TPCC_QER_Rn(5)	R	32	0x2080 + (0x200*n)	0x01C0 2080 + (0x200*n)
TPCC_QEER_Rn(5)	R	32	0x2084 + (0x200*n)	0x01C0 2084 + (0x200*n)
TPCC_QEECR_Rn(5)	W	32	0x2088 + (0x200*n)	0x01C0 2088 + (0x200*n)
TPCC_QEESR_Rn(5)	W	32	0x208C + (0x200*n)	0x01C0 208C + (0x200*n)
TPCC_QSER_Rn(5)	R	32	0x2090 + (0x200*n)	0x01C0 2090 + (0x200*n)
TPCC_QSECR_Rn(5)	W	32	0x2094 + (0x200*n)	0x01C0 2094 + (0x200*n)
TPCC_OPTm(6)	RW	32	0x4000 + (0x20*m)	0x01C0 4000 + (0x20*m)
TPCC_SRCm(6)	RW	32	0x4004 + (0x20*m)	0x01C0 4004 + (0x20*m)
TPCC_ABCNTm(6)	RW	32	0x4008 + (0x20*m)	0x01C0 4008 + (0x20*m)
TPCC_DSTm(6)	RW	32	0x400C + (0x20*m)	0x01C0 400C + (0x20*m)
TPCC_BIDXm(6)	RW	32	0x4010 + (0x20*m)	0x01C0 4010 + (0x20*m)
TPCC_LNKm(6)	RW	32	0x4014 + (0x20*m)	0x01C0 4014 + (0x20*m)
TPCC_CIDXm(6)	RW	32	0x4018 + (0x20*m)	0x01C0 4018 + (0x20*m)
TPCC_CCNTm(6)	RW	32	0x401C + (0x20*m)	0x01C0 401C + (0x20*m)

(5) n = 0 to 7
(6) m = 0 to 127

14.5.5.2 TPCC Register Descriptions

Table 14-262. TPCC_PID

Address Offset	0x0000		
Physical address	0x01C0 0000	**Instance**	IVA2.2 TPCC
Description	Peripheral ID Register		
Type	R		

31 30	29 28	27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10 9 8	7 6	5 4 3 2 1 0
SCHEME	Reserved	FUNC		RTL	MAJOR	CUSTOM	MINOR

Bits	Field Name	Description	Type	Reset
31:30	SCHEME	PID Scheme: Used to distinguish between old ID scheme and current. Spare bit to encode future schemes EDMA uses new scheme, indicated with value of 0x1.	R	0x1
29:28	Reserved	Read returns 0.	R	0x0
27:16	FUNC	Function indicates a software compatible module family.	R	0x001
15:11	RTL	RTL Version	R	0x--
10:8	MAJOR	Major Revision	R	0x3
7:6	CUSTOM	Custom revision field: Not used on this version of EDMA.	R	0x0
5:0	MINOR	Minor Revision	R	0x--

Table 14-263. Register Call Summary for Register TPCC_PID

IVA2.2 Subsystem Register Manual

- TPCC Register Mapping Summary: [0]

Table 14-264. TPCC_CCCFG

Address Offset	0x0004		
Physical address	0x01C0 0004	**Instance**	IVA2.2 TPCC
Description	CC Configuration Register		
Type	R		

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10 9 8	7 6 5 4 3 2 1 0
Reserved	MPEXIST	CHMAPEXIST	Reserved	NUMREGN	Reserved	NUMTC	Reserved	NUMPAENTRY	Reserved	NUMINTCH	Reserved	NUMQDMACH	Reserved	NUMDMACH

Bits	Field Name	Description	Type	Reset
31:26	Reserved	Read returns 0.	R	0x00
25	MPEXIST	Memory Protection Existence MPEXIST = 0: No memory protection. MPEXIST = 1: Memory Protection logic included.	R	1
24	CHMAPEXIST	Channel Mapping Existence CHMAPEXIST = 0: No Channel mapping. CHMAPEXIST = 1: Channel mapping logic included.	R	1
23:22	Reserved	Read returns 0.	R	0x0
21:20	NUMREGN	Number of MP and Shadow regions	R	0x3

Bits	Field Name	Description	Type	Reset
		Read 0x0: 0 Regions		
		Read 0x1: 2 Regions		
		Read 0x2: 4 Regions		
		Read 0x3: 8 Regions		
19	Reserved	Read returns 0.	R	0
18:16	NUMTC	Number of Queues/Number of TCs	R	0x1
		Read 0x0: 1 TC/Event Queue		
		Read 0x1: 2 TC/Event Queue		
		Read 0x2: 3 TC/Event Queue		
		Read 0x3: 4 TC/Event Queue		
		Read 0x4: 5 TC/Event Queue		
		Read 0x5: 6 TC/Event Queue		
		Read 0x6: 7 TC/Event Queue		
		Read 0x7: 8 TC/Event Queue		
15	Reserved	Read returns 0.	R	0
14:12	NUMPAENTRY	Number of PaRAM entries	R	0x3
		Read 0x0: 16 entries		
		Read 0x1: 32 entries (unsupported setting)		
		Read 0x2: 64 entries		
		Read 0x3: 128 entries		
		Read 0x4: 256 entries		
		Read 0x5: 512 entries		
11	Reserved	Read returns 0.	R	0
10:8	NUMINTCH	Number of Interrupt Channels	R	0x4
		Read 0x1: 8 Interrupt channels		
		Read 0x2: 16 Interrupt channels		
		Read 0x3: 32 Interrupt channels		
		Read 0x4: 64 Interrupt channels		
7	Reserved	Read returns 0.	R	0
6:4	NUMQDMACH	Number of QDMA Channels	R	0x2
		Read 0x0: No QDMA Channels		
		Read 0x1: 2 QDMA Channels		
		Read 0x2: 4 QDMA Channels		
		Read 0x3: 6 QDMA Channels		
		Read 0x4: 8 QDMA Channels		
3	Reserved	Read returns 0.	R	0
2:0	NUMDMACH	Number of DMA Channels	R	0x5
		Read 0x0: No DMA Channels		
		Read 0x1: 4 DMA Channels		
		Read 0x2: 8 DMA Channels		
		Read 0x3: 16 DMA Channels		
		Read 0x4: 32 DMA Channels		
		Read 0x5: 64 DMA Channels		

Table 14-265. Register Call Summary for Register TPCC_CCCFG

IVA2.2 Subsystem Register Manual

- TPCC Register Mapping Summary: [0]

Table 14-266. TPCC_DCHMAPi

Address Offset	0x0100 + (0x4*i)		
Physical address	0x01C0 0100 + (0x4*i)	**Instance**	IVA2.2 TPCC
Description	DMA Channel i Mapping Register		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																PAENTRY														Reserved			

Bits	Field Name	Description	Type	Reset
31:14	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0x00000
13:5	PAENTRY	PaRAM Entry number for DMA Channel i.	RW	0x000
4:0	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0x00

Table 14-267. Register Call Summary for Register TPCC_DCHMAPi

IVA2.2 Subsystem Functional Description
- EDMA: [0] [1]

IVA2.2 Subsystem Basic Programming Model
- Starting the Transfer: [2] [3] [4]

IVA2.2 Subsystem Register Manual
- TPCC Register Mapping Summary: [5]

Table 14-268. TPCC_QCHMAPj

Address Offset	0x0200 + (0x4*j)		
Physical address	0x01C0 0200 + (0x4*j)	**Instance**	IVA2.2 TPCC
Description	QDMA Channel i Mapping Register		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																PAENTRY														TRWORD			Reserved

Bits	Field Name	Description	Type	Reset
31:14	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0x00000
13:5	PAENTRY	PaRAM Entry number for QDMA Channel i.	RW	0x000
4:2	TRWORD	TRWORD points to the specific trigger word of the PaRAM Entry defined by PAENTRY. A write to the trigger word results in a QDMA Event being recognized.	RW	0x0
1:0	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0x0

Table 14-269. Register Call Summary for Register TPCC_QCHMAPj

IVA2.2 Subsystem Functional Description
- EDMA: [0] [1] [2] [3] [4]

IVA2.2 Subsystem Basic Programming Model
- Starting the Transfer: [5] [6]

IVA2.2 Subsystem Register Manual
- TPCC Register Mapping Summary: [7]

Table 14-270. TPCC_DMAQNUM0

Address Offset	0x0240		
Physical address	0x01C0 0240	**Instance**	IVA2.2 TPCC
Description	DMA Queue Number Register 0 Contains the Event queue number to be used for the corresponding DMA Channel.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved	E7			Reserved	E6			Reserved	E5			Reserved	E4			Reserved	E3			Reserved	E2			Reserved	E1			Reserved	E0		

Bits	Field Name	Description	Type	Reset
31	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0
30:28	E7	DMA Queue Number for event #7 0x0: Event En is queued on Q0 0x1: Event En is queued on Q1 Others: Not applicable for IVA2.2	RW	0x0
27	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0
26:24	E6	DMA Queue Number for event #6 0x0: Event En is queued on Q0 0x1: Event En is queued on Q1 Others: Not applicable for IVA2.2	RW	0x0
23	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0
22:20	E5	DMA Queue Number for event #5 0x0: Event En is queued on Q0 0x1: Event En is queued on Q1 Others: Not applicable for IVA2.2	RW	0x0
19	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0
18:16	E4	DMA Queue Number for event #4 0x0: Event En is queued on Q0 0x1: Event En is queued on Q1 Others: Not applicable for IVA2.2	RW	0x0
15	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0
14:12	E3	DMA Queue Number for event #3 0x0: Event En is queued on Q0 0x1: Event En is queued on Q1 Others: Not applicable for IVA2.2	RW	0x0
11	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0
10:8	E2	DMA Queue Number for event #2 0x0: Event En is queued on Q0 0x1: Event En is queued on Q1 Others: Not applicable for IVA2.2	RW	0x0
7	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0
6:4	E1	DMA Queue Number for event #1	RW	0x0

Bits	Field Name	Description	Type	Reset
		0x0: Event En is queued on Q0 0x1: Event En is queued on Q1 Others: Not applicable for IVA2.2		
3	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0
2:0	E0	DMA Queue Number for event #0 0x0: Event En is queued on Q0 0x1: Event En is queued on Q1 Others: Not applicable for IVA2.2	RW	0x0

Table 14-271. Register Call Summary for Register TPCC_DMAQNUM0

IVA2.2 Subsystem Basic Programming Model
- Prioritizing Defined Transfers: [0]

IVA2.2 Subsystem Register Manual
- TPCC Register Mapping Summary: [1]

Table 14-272. TPCC_DMAQNUM1

Address Offset	0x0244		
Physical address	0x01C0 0244	**Instance**	IVA2.2 TPCC
Description	DMA Queue Number Register 1 Contains the Event queue number to be used for the corresponding DMA Channel.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved	E15			Reserved	E14			Reserved	E13			Reserved	E12			Reserved	E11			Reserved	E10			Reserved	E9			Reserved	E8		

Bits	Field Name	Description	Type	Reset
31	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0
30:28	E15	DMA Queue Number for event #15 0x0: Event En is queued on Q0 0x1: Event En is queued on Q1 Others: Not applicable for IVA2.2	RW	0x0
27	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0
26:24	E14	DMA Queue Number for event #14 0x0: Event En is queued on Q0 0x1: Event En is queued on Q1 Others: Not applicable for IVA2.2	RW	0x0
23	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0
22:20	E13	DMA Queue Number for event #13 0x0: Event En is queued on Q0 0x1: Event En is queued on Q1 Others: Not applicable for IVA2.2	RW	0x0
19	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0
18:16	E12	DMA Queue Number for event #12 0x0: Event En is queued on Q0	RW	0x0

Bits	Field Name	Description	Type	Reset
		0x1: Event En is queued on Q1		
		Others: Not applicable for IVA2.2		
15	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0
14:12	E11	DMA Queue Number for event #11	RW	0x0
		0x0: Event En is queued on Q0		
		0x1: Event En is queued on Q1		
		Others: Not applicable for IVA2.2		
11	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0
10:8	E10	DMA Queue Number for event #10	RW	0x0
		0x0: Event En is queued on Q0		
		0x1: Event En is queued on Q1		
		Others: Not applicable for IVA2.2		
7	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0
6:4	E9	DMA Queue Number for event #9	RW	0x0
		0x0: Event En is queued on Q0		
		0x1: Event En is queued on Q1		
		Others: Not applicable for IVA2.2		
3	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0
2:0	E8	DMA Queue Number for event #8	RW	0x0
		0x0: Event En is queued on Q0		
		0x1: Event En is queued on Q1		
		Others: Not applicable for IVA2.2		

Table 14-273. Register Call Summary for Register TPCC_DMAQNUM1

IVA2.2 Subsystem Basic Programming Model
- Prioritizing Defined Transfers: [0]

IVA2.2 Subsystem Register Manual
- TPCC Register Mapping Summary: [1]

Table 14-274. TPCC_DMAQNUM2

Address Offset 0x0248

Physical address 0x01C0 0248 **Instance** IVA2.2 TPCC

Description DMA Queue Number Register 2
Contains the Event queue number to be used for the corresponding DMA Channel.

Type RW

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved	E23			Reserved	E22			Reserved	E21			Reserved	E20			Reserved	E19			Reserved	E18			Reserved	E17			Reserved	E16		

Bits	Field Name	Description	Type	Reset
31	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0
30:28	E23	DMA Queue Number for event #23	RW	0x0
		0x0: Event En is queued on Q0		
		0x1: Event En is queued on Q1		

Bits	Field Name	Description	Type	Reset
		Others: Not applicable for IVA2.2		
27	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0
26:24	E22	DMA Queue Number for event #22	RW	0x0
		0x0: Event En is queued on Q0		
		0x1: Event En is queued on Q1		
		Others: Not applicable for IVA2.2		
23	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0
22:20	E21	DMA Queue Number for event #21	RW	0x0
		0x0: Event En is queued on Q0		
		0x1: Event En is queued on Q1		
		Others: Not applicable for IVA2.2		
19	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0
18:16	E20	DMA Queue Number for event #20	RW	0x0
		0x0: Event En is queued on Q0		
		0x1: Event En is queued on Q1		
		Others: Not applicable for IVA2.2		
15	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0
14:12	E19	DMA Queue Number for event #19	RW	0x0
		0x0: Event En is queued on Q0		
		0x1: Event En is queued on Q1		
		Others: Not applicable for IVA2.2		
11	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0
10:8	E18	DMA Queue Number for event #18	RW	0x0
		0x0: Event En is queued on Q0		
		0x1: Event En is queued on Q1		
		Others: Not applicable for IVA2.2		
7	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0
6:4	E17	DMA Queue Number for event #17	RW	0x0
		0x0: Event En is queued on Q0		
		0x1: Event En is queued on Q1		
		Others: Not applicable for IVA2.2		
3	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0
2:0	E16	DMA Queue Number for event #16	RW	0x0
		0x0: Event En is queued on Q0		
		0x1: Event En is queued on Q1		
		Others: Not applicable for IVA2.2		

Table 14-275. Register Call Summary for Register TPCC_DMAQNUM2

IVA2.2 Subsystem Register Manual

- TPCC Register Mapping Summary: [0]

Table 14-276. TPCC_DMAQNUM3

Address Offset	0x024C		
Physical address	0x01C0 024C	**Instance**	IVA2.2 TPCC
Description	DMA Queue Number Register 3 Contains the Event queue number to be used for the corresponding DMA Channel.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved	E31			Reserved	E30			Reserved	E29			Reserved	E28			Reserved	E27			Reserved	E26			Reserved	E25			Reserved	E24		

Bits	Field Name	Description	Type	Reset
31	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0
30:28	E31	DMA Queue Number for event #31 0x0: Event En is queued on Q0 0x1: Event En is queued on Q1 Others: Not applicable for IVA2.2	RW	0x0
27	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0
26:24	E30	DMA Queue Number for event #30 0x0: Event En is queued on Q0 0x1: Event En is queued on Q1 Others: Not applicable for IVA2.2	RW	0x0
23	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0
22:20	E29	DMA Queue Number for event #29 0x0: Event En is queued on Q0 0x1: Event En is queued on Q1 Others: Not applicable for IVA2.2	RW	0x0
19	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0
18:16	E28	DMA Queue Number for event #28 0x0: Event En is queued on Q0 0x1: Event En is queued on Q1 Others: Not applicable for IVA2.2	RW	0x0
15	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0
14:12	E27	DMA Queue Number for event #27 0x0: Event En is queued on Q0 0x1: Event En is queued on Q1 Others: Not applicable for IVA2.2	RW	0x0
11	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0
10:8	E26	DMA Queue Number for event #26 0x0: Event En is queued on Q0 0x1: Event En is queued on Q1 Others: Not applicable for IVA2.2	RW	0x0
7	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0
6:4	E25	DMA Queue Number for event #25	RW	0x0

Bits	Field Name	Description	Type	Reset
		0x0: Event En is queued on Q0		
		0x1: Event En is queued on Q1		
		Others: Not applicable for IVA2.2		
3	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0
2:0	E24	DMA Queue Number for event #24	RW	0x0
		0x0: Event En is queued on Q0		
		0x1: Event En is queued on Q1		
		Others: Not applicable for IVA2.2		

Table 14-277. Register Call Summary for Register TPCC_DMAQNUM3

IVA2.2 Subsystem Register Manual

- TPCC Register Mapping Summary: [0]

Table 14-278. TPCC_DMAQNUM4

Address Offset	**0x0250**		
Physical address	0x01C0 0250	Instance	IVA2.2 TPCC
Description	DMA Queue Number Register 4 Contains the Event queue number to be used for the corresponding DMA Channel.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved	E39			Reserved	E38			Reserved	E37			Reserved	E36			Reserved	E35			Reserved	E34			Reserved	E33			Reserved	E32		

Bits	Field Name	Description	Type	Reset
31	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0
30:28	E39	DMA Queue Number for event #39	RW	0x0
		0x0: Event En is queued on Q0		
		0x1: Event En is queued on Q1		
		Others: Not applicable for IVA2.2		
27	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0
26:24	E38	DMA Queue Number for event #38	RW	0x0
		0x0: Event En is queued on Q0		
		0x1: Event En is queued on Q1		
		Others: Not applicable for IVA2.2		
23	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0
22:20	E37	DMA Queue Number for event #37	RW	0x0
		0x0: Event En is queued on Q0		
		0x1: Event En is queued on Q1		
		Others: Not applicable for IVA2.2		
19	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0
18:16	E36	DMA Queue Number for event #36	RW	0x0
		0x0: Event En is queued on Q0		
		0x1: Event En is queued on Q1		
		Others: Not applicable for IVA2.2		

Bits	Field Name	Description	Type	Reset
15	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0
14:12	E35	DMA Queue Number for event #35 0x0: Event En is queued on Q0 0x1: Event En is queued on Q1 Others: Not applicable for IVA2.2	RW	0x0
11	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0
10:8	E34	DMA Queue Number for event #34 0x0: Event En is queued on Q0 0x1: Event En is queued on Q1 Others: Not applicable for IVA2.2	RW	0x0
7	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0
6:4	E33	DMA Queue Number for event #33 0x0: Event En is queued on Q0 0x1: Event En is queued on Q1 Others: Not applicable for IVA2.2	RW	0x0
3	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0
2:0	E32	DMA Queue Number for event #32 0x0: Event En is queued on Q0 0x1: Event En is queued on Q1 Others: Not applicable for IVA2.2	RW	0x0

Table 14-279. Register Call Summary for Register TPCC_DMAQNUM4

IVA2.2 Subsystem Register Manual

- TPCC Register Mapping Summary: [0]

Table 14-280. TPCC_DMAQNUM5

Address Offset	0x0254		
Physical address	0x01C0 0254	**Instance**	IVA2.2 TPCC
Description	DMA Queue Number Register 5 Contains the Event queue number to be used for the corresponding DMA Channel.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved	E47			Reserved	E46			Reserved	E45			Reserved	E44			Reserved	E43			Reserved	E42			Reserved	E41			Reserved	E40		

Bits	Field Name	Description	Type	Reset
31	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0
30:28	E47	DMA Queue Number for event #47 0x0: Event En is queued on Q0 0x1: Event En is queued on Q1 Others: Not applicable for IVA2.2	RW	0x0
27	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0
26:24	E46	DMA Queue Number for event #46	RW	0x0

Bits	Field Name	Description	Type	Reset
		0x0: Event En is queued on Q0 0x1: Event En is queued on Q1 Others: Not applicable for IVA2.2		
23	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0
22:20	E45	DMA Queue Number for event #45 0x0: Event En is queued on Q0 0x1: Event En is queued on Q1 Others: Not applicable for IVA2.2	RW	0x0
19	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0
18:16	E44	DMA Queue Number for event #44 0x0: Event En is queued on Q0 0x1: Event En is queued on Q1 Others: Not applicable for IVA2.2	RW	0x0
15	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0
14:12	E43	DMA Queue Number for event #43 0x0: Event En is queued on Q0 0x1: Event En is queued on Q1 Others: Not applicable for IVA2.2	RW	0x0
11	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0
10:8	E42	DMA Queue Number for event #42 0x0: Event En is queued on Q0 0x1: Event En is queued on Q1 Others: Not applicable for IVA2.2	RW	0x0
7	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0
6:4	E41	DMA Queue Number for event #41 0x0: Event En is queued on Q0 0x1: Event En is queued on Q1 Others: Not applicable for IVA2.2	RW	0x0
3	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0
2:0	E40	DMA Queue Number for event #40 0x0: Event En is queued on Q0 0x1: Event En is queued on Q1 Others: Not applicable for IVA2.2	RW	0x0

Table 14-281. Register Call Summary for Register TPCC_DMAQNUM5

IVA2.2 Subsystem Register Manual

- TPCC Register Mapping Summary: [0]

Table 14-282. TPCC_DMAQNUM6

Address Offset	0x0258		
Physical address	0x01C0 0258	**Instance**	IVA2.2 TPCC
Description	DMA Queue Number Register 6 Contains the Event queue number to be used for the corresponding DMA Channel.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved	E55			Reserved	E54			Reserved	E53			Reserved	E52			Reserved	E51			Reserved	E50			Reserved	E49			Reserved	E48		

Bits	Field Name	Description	Type	Reset
31	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0
30:28	E55	DMA Queue Number for event #55 0x0: Event En is queued on Q0 0x1: Event En is queued on Q1 Others: Not applicable for IVA2.2	RW	0x0
27	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0
26:24	E54	DMA Queue Number for event #54 0x0: Event En is queued on Q0 0x1: Event En is queued on Q1 Others: Not applicable for IVA2.2	RW	0x0
23	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0
22:20	E53	DMA Queue Number for event #53 0x0: Event En is queued on Q0 0x1: Event En is queued on Q1 Others: Not applicable for IVA2.2	RW	0x0
19	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0
18:16	E52	DMA Queue Number for event #52 0x0: Event En is queued on Q0 0x1: Event En is queued on Q1 Others: Not applicable for IVA2.2	RW	0x0
15	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0
14:12	E51	DMA Queue Number for event #51 0x0: Event En is queued on Q0 0x1: Event En is queued on Q1 Others: Not applicable for IVA2.2	RW	0x0
11	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0
10:8	E50	DMA Queue Number for event #50 0x0: Event En is queued on Q0 0x1: Event En is queued on Q1 Others: Not applicable for IVA2.2	RW	0x0
7	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0
6:4	E49	DMA Queue Number for event #49 0x0: Event En is queued on Q0	RW	0x0

Bits	Field Name	Description	Type	Reset
		0x1: Event En is queued on Q1		
		Others: Not applicable for IVA2.2		
3	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0
2:0	E48	DMA Queue Number for event #48	RW	0x0
		0x0: Event En is queued on Q0		
		0x1: Event En is queued on Q1		
		Others: Not applicable for IVA2.2		

Table 14-283. Register Call Summary for Register TPCC_DMAQNUM6

IVA2.2 Subsystem Register Manual

- TPCC Register Mapping Summary: [0]

Table 14-284. TPCC_DMAQNUM7

Address Offset	0x025C		
Physical address	0x01C0 025C	**Instance**	IVA2.2 TPCC
Description	DMA Queue Number Register 7 Contains the Event queue number to be used for the corresponding DMA Channel.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved	E63			Reserved	E62			Reserved	E61			Reserved	E60			Reserved	E59			Reserved	E58			Reserved	E57			Reserved	E56		

Bits	Field Name	Description	Type	Reset
31	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0
30:28	E63	DMA Queue Number for event #63	RW	0x0
		0x0: Event En is queued on Q0		
		0x1: Event En is queued on Q1		
		Others: Not applicable for IVA2.2		
27	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0
26:24	E62	DMA Queue Number for event #62	RW	0x0
		0x0: Event En is queued on Q0		
		0x1: Event En is queued on Q1		
		Others: Not applicable for IVA2.2		
23	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0
22:20	E61	DMA Queue Number for event #61	RW	0x0
		0x0: Event En is queued on Q0		
		0x1: Event En is queued on Q1		
		Others: Not applicable for IVA2.2		
19	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0
18:16	E60	DMA Queue Number for event #60	RW	0x0
		0x0: Event En is queued on Q0		
		0x1: Event En is queued on Q1		
		Others: Not applicable for IVA2.2		

Bits	Field Name	Description	Type	Reset
15	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0
14:12	E59	DMA Queue Number for event #59 0x0: Event En is queued on Q0 0x1: Event En is queued on Q1 Others: Not applicable for IVA2.2	RW	0x0
11	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0
10:8	E58	DMA Queue Number for event #58 0x0: Event En is queued on Q0 0x1: Event En is queued on Q1 Others: Not applicable for IVA2.2	RW	0x0
7	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0
6:4	E57	DMA Queue Number for event #57 0x0: Event En is queued on Q0 0x1: Event En is queued on Q1 Others: Not applicable for IVA2.2	RW	0x0
3	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0
2:0	E56	DMA Queue Number for event #56 0x0: Event En is queued on Q0 0x1: Event En is queued on Q1 Others: Not applicable for IVA2.2	RW	0x0

Table 14-285. Register Call Summary for Register TPCC_DMAQNUM7

IVA2.2 Subsystem Register Manual

- TPCC Register Mapping Summary: [0]

Table 14-286. TPCC_QDMAQNUM

Address Offset	0x0260		
Physical address	0x01C0 0260	**Instance**	IVA2.2 TPCC
Description	QDMA Queue Number Register Contains the Event queue number to be used for the corresponding QDMA Channel.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved	E7			Reserved	E6			Reserved	E5			Reserved	E4			Reserved	E3			Reserved	E2			Reserved	E1			Reserved	E0		

Bits	Field Name	Description	Type	Reset
31	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0
30:28	E7	QDMA Queue Number for event #7 0x0: Event En is queued on Q0 0x1: Event En is queued on Q1 Others: Not applicable for IVA2.2	RW	0x0
27	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0
26:24	E6	QDMA Queue Number for event #6	RW	0x0

Bits	Field Name	Description	Type	Reset
		0x0: Event En is queued on Q0 0x1: Event En is queued on Q1 Others: Not applicable for IVA2.2		
23	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0
22:20	E5	QDMA Queue Number for event #5 0x0: Event En is queued on Q0 0x1: Event En is queued on Q1 others: Not applicable for IVA2.2	RW	0x0
19	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0
18:16	E4	QDMA Queue Number for event #4 0x0: Event En is queued on Q0 0x1: Event En is queued on Q1 Others: Not applicable for IVA2.2	RW	0x0
15	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0
14:12	E3	QDMA Queue Number for event #3 0x0: Event En is queued on Q0 0x1: Event En is queued on Q1 Others: Not applicable for IVA2.2	RW	0x0
11	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0
10:8	E2	QDMA Queue Number for event #2 0x0: Event En is queued on Q0 0x1: Event En is queued on Q1 Others: Not applicable for IVA2.2	RW	0x0
7	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0
6:4	E1	QDMA Queue Number for event #1 0x0: Event En is queued on Q0 0x1: Event En is queued on Q1 Others: Not applicable for IVA2.2	RW	0x0
3	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0
2:0	E0	QDMA Queue Number for event #0 0x0: Event En is queued on Q0 0x1: Event En is queued on Q1 Others: Not applicable for IVA2.2	RW	0x0

Table 14-287. Register Call Summary for Register TPCC_QDMAQNUM

IVA2.2 Subsystem Register Manual

- TPCC Register Mapping Summary: [0]

Table 14-288. TPCC_QUETCMAP

Address Offset	0x0280		
Physical address	0x01C0 0280	**Instance**	IVA2.2 TPCC
Description	Queue to TC Mapping		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																								TCNUMQ1				Reserved	TCNUMQ0		

Bits	Field Name	Description	Type	Reset
31:7	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0x00
6:4	TCNUMQ1	TC Number for Queue 1: Defines the TC number that Event Queue 1 TRs are written to. 0x0: TRs from this queue are routed to TPTC0 0x1: TRs from this queue are routed to TPTC1 Others: Not applicable for IVA2.2	RW	0x1
3	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0
2:0	TCNUMQ0	TC Number for Queue 0: Defines the TC number that Event Queue 0 TRs are written to. 0x0: TRs from this queue are routed to TPTC0 0x1: TRs from this queue are routed to TPTC1 Others: Not applicable for IVA2.2	RW	0x0

Table 14-289. Register Call Summary for Register TPCC_QUETCMAP

IVA2.2 Subsystem Basic Programming Model
- Prioritizing Defined Transfers: [0]

IVA2.2 Subsystem Register Manual
- TPCC Register Mapping Summary: [1]

Table 14-290. TPCC_QUEPRI

Address Offset	0x0284		
Physical address	0x01C0 0284	**Instance**	IVA2.2 TPCC
Description	Queue Priority		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																								PRIQ1				Reserved	PRIQ0		

Bits	Field Name	Description	Type	Reset
31:7	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0x00
6:4	PRIQ1	Priority Level for Queue 1 Dictates the priority level used for the OPTIONS field programmation for Qn TRs. Sets the priority used for TC read and write commands. 0x0: Priority 0 - Highest priority 0x1: Priority 1 0x2: Priority 2	RW	0x0

Bits	Field Name	Description	Type	Reset
		0x3: Priority 3 0x4: Priority 4 0x5: Priority 5 0x6: Priority 6 0x7: Priority 7 - Lowest Priority		
3	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0
2:0	PRIQ0	Priority Level for Queue 0 Dictates the priority level used for the OPTIONS field programmation for Qn TRs. Sets the priority used for TC read and write commands. 0x0: Priority 0 - Highest priority 0x1: Priority 1 0x2: Priority 2 0x3: Priority 3 0x4: Priority 4 0x5: Priority 5 0x6: Priority 6 0x7: Priority 7 - Lowest Priority	RW	0x0

Table 14-291. Register Call Summary for Register TPCC_QUEPRI

IVA2.2 Subsystem Basic Programming Model
- Prioritizing Defined Transfers: [0] [1] [2]

IVA2.2 Subsystem Register Manual
- TPCC Register Mapping Summary: [3]

Table 14-292. TPCC_EMR

Address Offset	0x0300		
Physical address	0x01C0 0300	**Instance**	IVA2.2 TPCC
Description	Event Missed Register: The Event Missed register is set if 2 events are received without the first event being cleared or if a Null TR is serviced. Chained events (CER), Set Events (ESR), and normal events (ER) are treated individually. If any bit in the EMR register is set (and all errors (including QEMR/CCERR) were previously clear), then an error will be signaled with TPCC error interrupt.		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
E31	E30	E29	E28	E27	E26	E25	E24	E23	E22	E21	E20	E19	E18	E17	E16	E15	E14	E13	E12	E11	E10	E9	E8	E7	E6	E5	E4	E3	E2	E1	E0

Bits	Field Name	Description	Type	Reset
31	E31	Event Missed #31	R	0
30	E30	Event Missed #30	R	0
29	E29	Event Missed #29	R	0
28	E28	Event Missed #28	R	0
27	E27	Event Missed #27	R	0
26	E26	Event Missed #26	R	0
25	E25	Event Missed #25	R	0
24	E24	Event Missed #24	R	0
23	E23	Event Missed #23	R	0
22	E22	Event Missed #22	R	0
21	E21	Event Missed #21	R	0
20	E20	Event Missed #20	R	0

Bits	Field Name	Description	Type	Reset
19	E19	Event Missed #19	R	0
18	E18	Event Missed #18	R	0
17	E17	Event Missed #17	R	0
16	E16	Event Missed #16	R	0
15	E15	Event Missed #15	R	0
14	E14	Event Missed #14	R	0
13	E13	Event Missed #13	R	0
12	E12	Event Missed #12	R	0
11	E11	Event Missed #11	R	0
10	E10	Event Missed #10	R	0
9	E9	Event Missed #9	R	0
8	E8	Event Missed #8	R	0
7	E7	Event Missed #7	R	0
6	E6	Event Missed #6	R	0
5	E5	Event Missed #5	R	0
4	E4	Event Missed #4	R	0
3	E3	Event Missed #3	R	0
2	E2	Event Missed #2	R	0
1	E1	Event Missed #1	R	0
0	E0	Event Missed #0	R	0

Table 14-293. Register Call Summary for Register TPCC_EMR

IVA2.2 Subsystem Basic Programming Model

- Error Reporting for EDMA Module: [0]

IVA2.2 Subsystem Register Manual

- TPCC Register Mapping Summary: [1]

Table 14-294. TPCC_EMRH

Address Offset 0x0304

Physical address 0x01C0 0304 **Instance** IVA2.2 TPCC

Description Event Missed Register (High Part):

The Event Missed register is set if 2 events are received without the first event being cleared or if a Null TR is serviced. Chained events (CER), Set Events (ESR), and normal events (ER) are treated individually. If any bit in the EMR register is set (and all errors (including QEMR/CCERR)were previously clear), then an error will be signaled with TPCC error interrupt.

Type R

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
E63	E62	E61	E60	E59	E58	E57	E56	E55	E54	E53	E52	E51	E50	E49	E48	E47	E46	E45	E44	E43	E42	E41	E40	E39	E38	E37	E36	E35	E34	E33	E32

Bits	Field Name	Description	Type	Reset
31	E63	Event Missed #63	R	0
30	E62	Event Missed #62	R	0
29	E61	Event Missed #61	R	0
28	E60	Event Missed #60	R	0
27	E59	Event Missed #59	R	0
26	E58	Event Missed #58	R	0
25	E57	Event Missed #57	R	0
24	E56	Event Missed #56	R	0

Bits	Field Name	Description	Type	Reset
23	E55	Event Missed #55	R	0
22	E54	Event Missed #54	R	0
21	E53	Event Missed #53	R	0
20	E52	Event Missed #52	R	0
19	E51	Event Missed #51	R	0
18	E50	Event Missed #50	R	0
17	E49	Event Missed #49	R	0
16	E48	Event Missed #48	R	0
15	E47	Event Missed #47	R	0
14	E46	Event Missed #46	R	0
13	E45	Event Missed #45	R	0
12	E44	Event Missed #44	R	0
11	E43	Event Missed #43	R	0
10	E42	Event Missed #42	R	0
9	E41	Event Missed #41	R	0
8	E40	Event Missed #40	R	0
7	E39	Event Missed #39	R	0
6	E38	Event Missed #38	R	0
5	E37	Event Missed #37	R	0
4	E36	Event Missed #36	R	0
3	E35	Event Missed #35	R	0
2	E34	Event Missed #34	R	0
1	E33	Event Missed #33	R	0
0	E32	Event Missed #32	R	0

Table 14-295. Register Call Summary for Register TPCC_EMRH

IVA2.2 Subsystem Register Manual

- TPCC Register Mapping Summary: [0]

Table 14-296. TPCC_EMCR

Address Offset	0x0308		
Physical address	0x01C0 0308	**Instance**	IVA2.2 TPCC
Description	Event Missed Clear Register: CPU write of 1 to the EMCR.En bit causes the EMR.En bit to be cleared. CPU write of 0 has no effect. All error bits must be cleared before additional error interrupts will be asserted by CC.		
Type	W		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
E31	E30	E29	E28	E27	E26	E25	E24	E23	E22	E21	E20	E19	E18	E17	E16	E15	E14	E13	E12	E11	E10	E9	E8	E7	E6	E5	E4	E3	E2	E1	E0

Bits	Field Name	Description	Type	Reset
31	E31	Event Missed Clear #31	W	0
30	E30	Event Missed Clear #30	W	0
29	E29	Event Missed Clear #29	W	0
28	E28	Event Missed Clear #28	W	0
27	E27	Event Missed Clear #27	W	0
26	E26	Event Missed Clear #26	W	0
25	E25	Event Missed Clear #25	W	0

Bits	Field Name	Description	Type	Reset
24	E24	Event Missed Clear #24	W	0
23	E23	Event Missed Clear #23	W	0
22	E22	Event Missed Clear #22	W	0
21	E21	Event Missed Clear #21	W	0
20	E20	Event Missed Clear #20	W	0
19	E19	Event Missed Clear #19	W	0
18	E18	Event Missed Clear #18	W	0
17	E17	Event Missed Clear #17	W	0
16	E16	Event Missed Clear #16	W	0
15	E15	Event Missed Clear #15	W	0
14	E14	Event Missed Clear #14	W	0
13	E13	Event Missed Clear #13	W	0
12	E12	Event Missed Clear #12	W	0
11	E11	Event Missed Clear #11	W	0
10	E10	Event Missed Clear #10	W	0
9	E9	Event Missed Clear #9	W	0
8	E8	Event Missed Clear #8	W	0
7	E7	Event Missed Clear #7	W	0
6	E6	Event Missed Clear #6	W	0
5	E5	Event Missed Clear #5	W	0
4	E4	Event Missed Clear #4	W	0
3	E3	Event Missed Clear #3	W	0
2	E2	Event Missed Clear #2	W	0
1	E1	Event Missed Clear #1	W	0
0	E0	Event Missed Clear #0	W	0

Table 14-297. Register Call Summary for Register TPCC_EMCR

IVA2.2 Subsystem Register Manual

- TPCC Register Mapping Summary: [0]

Table 14-298. TPCC_EMCRH

Address Offset	0x030C
Physical address	0x01C0 030C **Instance** IVA2.2 TPCC
Description	Event Missed Clear Register (High Part): CPU write of 1 to the EMCR.En bit causes the EMR.En bit to be cleared. CPU write of 0 has no effect. All error bits must be cleared before additional error interrupts will be asserted by CC.
Type	W

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
E63	E62	E61	E60	E59	E58	E57	E56	E55	E54	E53	E52	E51	E50	E49	E48	E47	E46	E45	E44	E43	E42	E41	E40	E39	E38	E37	E36	E35	E34	E33	E32

Bits	Field Name	Description	Type	Reset
31	E63	Event Missed Clear #63	W	0
30	E62	Event Missed Clear #62	W	0
29	E61	Event Missed Clear #61	W	0
28	E60	Event Missed Clear #60	W	0
27	E59	Event Missed Clear #59	W	0
26	E58	Event Missed Clear #58	W	0

Bits	Field Name	Description	Type	Reset
25	E57	Event Missed Clear #57	W	0
24	E56	Event Missed Clear #56	W	0
23	E55	Event Missed Clear #55	W	0
22	E54	Event Missed Clear #54	W	0
21	E53	Event Missed Clear #53	W	0
20	E52	Event Missed Clear #52	W	0
19	E51	Event Missed Clear #51	W	0
18	E50	Event Missed Clear #50	W	0
17	E49	Event Missed Clear #49	W	0
16	E48	Event Missed Clear #48	W	0
15	E47	Event Missed Clear #47	W	0
14	E46	Event Missed Clear #46	W	0
13	E45	Event Missed Clear #45	W	0
12	E44	Event Missed Clear #44	W	0
11	E43	Event Missed Clear #43	W	0
10	E42	Event Missed Clear #42	W	0
9	E41	Event Missed Clear #41	W	0
8	E40	Event Missed Clear #40	W	0
7	E39	Event Missed Clear #39	W	0
6	E38	Event Missed Clear #38	W	0
5	E37	Event Missed Clear #37	W	0
4	E36	Event Missed Clear #36	W	0
3	E35	Event Missed Clear #35	W	0
2	E34	Event Missed Clear #34	W	0
1	E33	Event Missed Clear #33	W	0
0	E32	Event Missed Clear #32	W	0

Table 14-299. Register Call Summary for Register TPCC_EMCRH

IVA2.2 Subsystem Register Manual

- TPCC Register Mapping Summary: [0]

Table 14-300. TPCC_QEMR

Address Offset	0x0310		
Physical address	0x01C0 0310	**Instance**	IVA2.2 TPCC
Description	QDMA Event Missed Register: The QDMA Event Missed register is set if 2 QDMA events are detected without the first event being cleared or if a Null TR is serviced.. If any bit in the QEMR register is set (and all errors (including EMR/CCERR) were previously clear), then an error will be signaled with TPCC error interrupt.		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																								E7	E6	E5	E4	E3	E2	E1	E0

Bits	Field Name	Description	Type	Reset
31:8	Reserved	Read returns 0.	R	0x000000
7	E7	Event Missed #7	R	0
6	E6	Event Missed #6	R	0
5	E5	Event Missed #5	R	0
4	E4	Event Missed #4	R	0
3	E3	Event Missed #3	R	0

Bits	Field Name	Description	Type	Reset
2	E2	Event Missed #2	R	0
1	E1	Event Missed #1	R	0
0	E0	Event Missed #0	R	0

Table 14-301. Register Call Summary for Register TPCC_QEMR

IVA2.2 Subsystem Basic Programming Model
- Error Reporting for EDMA Module: [0]

IVA2.2 Subsystem Register Manual
- TPCC Register Mapping Summary: [1]

Table 14-302. TPCC_QEMCR

Address Offset	0x0314		
Physical address	0x01C0 0314	**Instance**	IVA2.2 TPCC
Description	QDMA Event Missed Clear Register: CPU write of 1 to the QEMCR.En bit causes the QEMR.En bit to be cleared. CPU write of 0 has no effect. All error bits must be cleared before additional error interrupts will be asserted by CC.		
Type	W		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																								E7	E6	E5	E4	E3	E2	E1	E0

Bits	Field Name	Description	Type	Reset
31:8	Reserved	Write 0s for future compatibility.	W	0x000000
7	E7	Event Missed Clear #7	W	0
6	E6	Event Missed Clear #6	W	0
5	E5	Event Missed Clear #5	W	0
4	E4	Event Missed Clear #4	W	0
3	E3	Event Missed Clear #3	W	0
2	E2	Event Missed Clear #2	W	0
1	E1	Event Missed Clear #1	W	0
0	E0	Event Missed Clear #0	W	0

Table 14-303. Register Call Summary for Register TPCC_QEMCR

IVA2.2 Subsystem Register Manual
- TPCC Register Mapping Summary: [0]

Table 14-304. TPCC_CCERR

Address Offset	0x0318		
Physical address	0x01C0 0318	**Instance**	IVA2.2 TPCC
Description	CC Error Register		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															TCERR	Reserved														QTHRXCD1	QTHRXCD0

Bits	Field Name	Description	Type	Reset
31:17	Reserved	Read returns 0.	R	0x0000
16	TCERR	Transfer Completion Code Error: TCCERR = 0: Total number of allowed TCCs outstanding has not been reached. TCCERR = 1: Total number of allowed TCCs has been reached. TCCERR can be cleared by writing 1 to corresponding bit in CCERRCLR register. If any bit in the CCERR register is set (and all errors were previously clear), then an error will be signaled with TPCC error interrupt.	R	0
15:2	Reserved	Read returns 0.	R	0x00
1	QTHRXCD1	Queue Threshold Error for Q1: QTHRXCD1 = 0: Watermark/threshold has not been exceeded. QTHRXCD1 = 1: Watermark/threshold has been exceeded. CCERR.QTHRXCD1 can be cleared by writing 1 to corresponding bit in CCERRCLR register. If any bit in the CCERR register is set (and all errors (including EMR/QEMR) were previously clear), then an error will be signaled with the TPCC error interrupt.	R	0
0	QTHRXCD0	Queue Threshold Error for Q0: QTHRXCD0 = 0: Watermark/threshold has not been exceeded. QTHRXCD0 = 1: Watermark/threshold has been exceeded. CCERR.QTHRXCD0 can be cleared by writing 1 to corresponding bit in CCERRCLR register. If any bit in the CCERR register is set (and all errors (including EMR/QEMR) were previously clear), then an error will be signaled with the TPCC error interrupt.	R	0

Table 14-305. Register Call Summary for Register TPCC_CCERR

IVA2.2 Subsystem Basic Programming Model

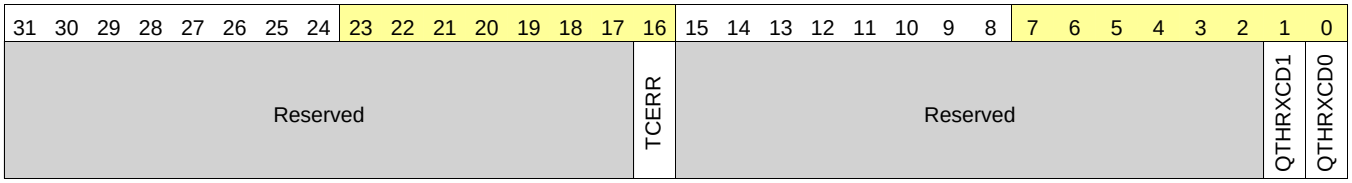
- Starting the Transfer: [0]
- Error Reporting for EDMA Module: [1] [2]

IVA2.2 Subsystem Register Manual

- TPCC Register Mapping Summary: [3]

Table 14-306. TPCC_CCERRCLR

Address Offset	0x031C		
Physical address	0x01C0 031C	**Instance**	IVA2.2 TPCC
Description	CC Error Clear Register		
Type	W		



Bits	Field Name	Description	Type	Reset
31:17	Reserved	Write 0s for future compatibility.	W	0x0000
16	TCERR	Clear Error for CCERR.TCERR: Write of 1 clears the value of CCERR bit N. Writes of 0 have no effect.	W	0
15:2	Reserved	Write 0s for future compatibility.	W	0x00
1	QTHRXCD1	Clear error for CCERR.QTHRXCD1: Write of 1 clears the values of QSTAT1.WM, QSTAT1.THRXCD, CCERR.QTHRXCD1 Writes of 0 have no effect.	W	0
0	QTHRXCD0	Clear error for CCERR.QTHRXCD0: Write of 1 clears the values of QSTAT0.WM, QSTAT0.THRXCD, CCERR.QTHRXCD0 Writes of 0 have no effect.	W	0

Table 14-307. Register Call Summary for Register TPCC_CCERRCLR

IVA2.2 Subsystem Register Manual

- TPCC Register Mapping Summary: [0]

Table 14-308. TPCC_EEVAL

Address Offset	0x0320		
Physical address	0x01C0 0320	**Instance**	IVA2.2 TPCC
Description	Error Eval Register		
Type	W		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																														SET	EVAL

Bits	Field Name	Description	Type	Reset
31:2	Reserved	Write 0s for future compatibility.	W	0x00000000
1	SET	Error Interrupt Set: CPU write of 1 to the SET bit causes the TPCC error interrupt to be pulsed regardless of state of EMR/EMRH, QEMR, or CCERR. CPU write of 0 has no effect.	W	0
0	EVAL	Error Interrupt Evaluate: CPU write of 1 to the EVAL bit causes the TPCC error interrupt to be pulsed if any errors have not been cleared in the EMR/EMRH, QEMR, or CCERR registers. CPU write of 0 has no effect.	W	0

Table 14-309. Register Call Summary for Register TPCC_EEVAL

IVA2.2 Subsystem Basic Programming Model

- Error Reporting for EDMA Module: [0]

IVA2.2 Subsystem Register Manual

- TPCC Register Mapping Summary: [1]

Table 14-310. TPCC_DRAEj

Address Offset	0x0340 + (0x8*j)		
Physical address	0x01C0 0340 + (0x8*j)	Instance	IVA2.2 TPCC
Description	DMA Region Access enable for bit N in Region i: En = 0: Accesses through Region i address space to Bit N in any DMA Channel Register are not allowed. Reads will return b0 on Bit N and writes will not modify the state of bit N. Enabled interrupt bits for bit N do not contribute to the generation of the TPCC region i interrupt. En = 1: Accesses through Region i address space to Bit N in any DMA Channel Register are allowed. Reads will return the value from Bit N and writes will modify the state of bit N. Enabled interrupt bits for bit N do contribute to the generation of the TPCC region i interrupt.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
E31	E30	E29	E28	E27	E26	E25	E24	E23	E22	E21	E20	E19	E18	E17	E16	E15	E14	E13	E12	E11	E10	E9	E8	E7	E6	E5	E4	E3	E2	E1	E0

Bits	Field Name	Description	Type	Reset
31	E31	DMA Region Access enable for Region i, bit #31	RW	0
30	E30	DMA Region Access enable for Region i, bit #30	RW	0
29	E29	DMA Region Access enable for Region i, bit #29	RW	0
28	E28	DMA Region Access enable for Region i, bit #28	RW	0
27	E27	DMA Region Access enable for Region i, bit #27	RW	0
26	E26	DMA Region Access enable for Region i, bit #26	RW	0

Bits	Field Name	Description	Type	Reset
25	E25	DMA Region Access enable for Region i, bit #25	RW	0
24	E24	DMA Region Access enable for Region i, bit #24	RW	0
23	E23	DMA Region Access enable for Region i, bit #23	RW	0
22	E22	DMA Region Access enable for Region i, bit #22	RW	0
21	E21	DMA Region Access enable for Region i, bit #21	RW	0
20	E20	DMA Region Access enable for Region i, bit #20	RW	0
19	E19	DMA Region Access enable for Region i, bit #19	RW	0
18	E18	DMA Region Access enable for Region i, bit #18	RW	0
17	E17	DMA Region Access enable for Region i, bit #17	RW	0
16	E16	DMA Region Access enable for Region i, bit #16	RW	0
15	E15	DMA Region Access enable for Region i, bit #15	RW	0
14	E14	DMA Region Access enable for Region i, bit #14	RW	0
13	E13	DMA Region Access enable for Region i, bit #13	RW	0
12	E12	DMA Region Access enable for Region i, bit #12	RW	0
11	E11	DMA Region Access enable for Region i, bit #11	RW	0
10	E10	DMA Region Access enable for Region i, bit #10	RW	0
9	E9	DMA Region Access enable for Region i, bit #9	RW	0
8	E8	DMA Region Access enable for Region i, bit #8	RW	0
7	E7	DMA Region Access enable for Region i, bit #7	RW	0
6	E6	DMA Region Access enable for Region i, bit #6	RW	0
5	E5	DMA Region Access enable for Region i, bit #5	RW	0
4	E4	DMA Region Access enable for Region i, bit #4	RW	0
3	E3	DMA Region Access enable for Region i, bit #3	RW	0
2	E2	DMA Region Access enable for Region i, bit #2	RW	0
1	E1	DMA Region Access enable for Region i, bit #1	RW	0
0	E0	DMA Region Access enable for Region i, bit #0	RW	0

Table 14-311. Register Call Summary for Register TPCC_DRAEj

IVA2.2 Subsystem Register Manual

- TPCC Register Mapping Summary: [0]

Table 14-312. TPCC_DRAEHj

Address Offset 0x0344 + (0x8*j)

Physical address 0x01C0 0344 + (0x8*j) **Instance** IVA2.2 TPCC

Description DMA Region Access enable for bit N in Region M:
En = 0: Accesses through Region i address space to Bit N in any DMA Channel Register are not allowed. Reads will return b0 on Bit N and writes will not modify the state of bit N. Enabled interrupt bits for bit N do not contribute to the generation of the TPCC region i interrupt.
En = 1: Accesses through Region i address space to Bit N in any DMA Channel Register are allowed. Reads will return the value from Bit N and writes will modify the state of bit N. Enabled interrupt bits for bit N do contribute to the generation of the TPCC region i interrupt.

Type RW

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
E63	E62	E61	E60	E59	E58	E57	E56	E55	E54	E53	E52	E51	E50	E49	E48	E47	E46	E45	E44	E43	E42	E41	E40	E39	E38	E37	E36	E35	E34	E33	E32

Bits	Field Name	Description	Type	Reset
31	E63	DMA Region Access enable for Region i, bit #63	RW	0
30	E62	DMA Region Access enable for Region i, bit #62	RW	0
29	E61	DMA Region Access enable for Region i, bit #61	RW	0

Bits	Field Name	Description	Type	Reset
28	E60	DMA Region Access enable for Region i, bit #60	RW	0
27	E59	DMA Region Access enable for Region i, bit #59	RW	0
26	E58	DMA Region Access enable for Region i, bit #58	RW	0
25	E57	DMA Region Access enable for Region i, bit #57	RW	0
24	E56	DMA Region Access enable for Region i, bit #56	RW	0
23	E55	DMA Region Access enable for Region i, bit #55	RW	0
22	E54	DMA Region Access enable for Region i, bit #54	RW	0
21	E53	DMA Region Access enable for Region i, bit #53	RW	0
20	E52	DMA Region Access enable for Region i, bit #52	RW	0
19	E51	DMA Region Access enable for Region i, bit #51	RW	0
18	E50	DMA Region Access enable for Region i, bit #50	RW	0
17	E49	DMA Region Access enable for Region i, bit #49	RW	0
16	E48	DMA Region Access enable for Region i, bit #48	RW	0
15	E47	DMA Region Access enable for Region i, bit #47	RW	0
14	E46	DMA Region Access enable for Region i, bit #46	RW	0
13	E45	DMA Region Access enable for Region i, bit #45	RW	0
12	E44	DMA Region Access enable for Region i, bit #44	RW	0
11	E43	DMA Region Access enable for Region i, bit #43	RW	0
10	E42	DMA Region Access enable for Region i, bit #42	RW	0
9	E41	DMA Region Access enable for Region i, bit #41	RW	0
8	E40	DMA Region Access enable for Region i, bit #40	RW	0
7	E39	DMA Region Access enable for Region i, bit #39	RW	0
6	E38	DMA Region Access enable for Region i, bit #38	RW	0
5	E37	DMA Region Access enable for Region i, bit #37	RW	0
4	E36	DMA Region Access enable for Region i, bit #36	RW	0
3	E35	DMA Region Access enable for Region i, bit #35	RW	0
2	E34	DMA Region Access enable for Region i, bit #34	RW	0
1	E33	DMA Region Access enable for Region i, bit #33	RW	0
0	E32	DMA Region Access enable for Region i, bit #32	RW	0

Table 14-313. Register Call Summary for Register TPCC_DRAEHj

IVA2.2 Subsystem Register Manual

- TPCC Register Mapping Summary: [0]

Table 14-314. TPCC_QRAEj

Address Offset	0x0380 + (0x4*j)		
Physical address	0x01C0 0380 + (0x4*j)	Instance	IVA2.2 TPCC
Description	QDMA Region Access enable for bit N in Region i: En = 0: Accesses through Region i address space to Bit N in any QDMA Channel Register are not allowed. Reads will return b0 on Bit N and writes will not modify the state of bit N. Enabled interrupt bits for bit N do not contribute to the generation of the TPCC region i interrupt. En = 1: Accesses through Region i address space to Bit N in any QDMA Channel Register are allowed. Reads will return the value from Bit N and writes will modify the state of bit N. Enabled interrupt bits for bit N do contribute to the generation of the TPCC region i interrupt.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																								E7	E6	E5	E4	E3	E2	E1	E0

Bits	Field Name	Description	Type	Reset
31:8	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0x000000
7	E7	QDMA Region Access enable for Region i, bit #7	RW	0
6	E6	QDMA Region Access enable for Region i, bit #6	RW	0
5	E5	QDMA Region Access enable for Region i, bit #5	RW	0
4	E4	QDMA Region Access enable for Region i, bit #4	RW	0
3	E3	QDMA Region Access enable for Region i, bit #3	RW	0
2	E2	QDMA Region Access enable for Region i, bit #2	RW	0
1	E1	QDMA Region Access enable for Region i, bit #1	RW	0
0	E0	QDMA Region Access enable for Region i, bit #0	RW	0

Table 14-315. Register Call Summary for Register TPCC_QRAEj

IVA2.2 Subsystem Register Manual

- TPCC Register Mapping Summary: [0]

Table 14-316. TPCC_Q0Ek

Address Offset	0x0400 + (0x4*k)		
Physical address	0x01C0 0400 + (0x4*k)	**Instance**	IVA2.2 TPCC
Description	Event Queue Entry Diagram for Queue j - Entry i (j =0 to1 and i=0 to 15)		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																								ETYPE		ENUM					

Bits	Field Name	Description	Type	Reset
31:8	Reserved	Read returns 0.	R	0x000000
7:6	ETYPE	Event Type: Specifies the specific Event Type for the given entry in the Event Queue. Read 0x0: Event Triggered through ER Read 0x1: Manual Triggered through ESR Read 0x2: Chain Triggered through CER Read 0x3: Auto-Triggered through QER	R	0x-
5:0	ENUM	Event Number: Specifies the specific Event Number for the given entry in the Event Queue. For DMA Channel events (ER/ESR/CER), ENUM will range between 0 and NUM_DMACH (up to 63). For QDMA Channel events (QER), ENUM will range between 0 and NUM_QDMACH (up to 7).	R	0x--

Table 14-317. Register Call Summary for Register TPCC_Q0Ek

IVA2.2 Subsystem Register Manual

- TPCC Register Mapping Summary: [0]

Table 14-318. TPCC_Q1Ek

Address Offset	0x0440 + (0x4*k)		
Physical address	0x01C0 0440 + (0x4*k)	**Instance**	IVA2.2 TPCC
Description	Event Queue Entry Diagram for Queue j - Entry i (j =0 to1 and i=0 to 15)		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																								ETYPE		ENUM					

Bits	Field Name	Description	Type	Reset
31:8	Reserved	Read returns 0.	R	0x000000
7:6	ETYPE	Event Type: Specifies the specific Event Type for the given entry in the Event Queue. Read 0x0: Event Triggered through ER Read 0x1: Manual Triggered through ESR Read 0x2: Chain Triggered through CER Read 0x3: Auto-Triggered through QER	R	0x-
5:0	ENUM	Event Number: Specifies the specific Event Number for the given entry in the Event Queue. For DMA Channel events (ER/ESR/CER), ENUM will range between 0 and NUM_DMACH (up to 63). For QDMA Channel events (QER), ENUM will range between 0 and NUM_QDMACH (up to 7).	R	0x--

Table 14-319. Register Call Summary for Register TPCC_Q1Ek

IVA2.2 Subsystem Register Manual

- TPCC Register Mapping Summary: [0]

Table 14-320. TPCC_QSTATl

Address Offset	0x0600 + (0x4*l)		
Physical address	0x01C0 0600 + (0x4*l)	**Instance**	IVA2.2 TPCC
Description	QSTATi Register Set		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							THRXCD	Reserved			WM					Reserved			NUMVAL					Reserved				STRTPTR			

Bits	Field Name	Description	Type	Reset
31:25	Reserved	Read returns 0.	R	0x00
24	THRXCD	Threshold Exceeded: THRXCD = 0: Threshold specified by QWMTHR(A|B).Qn has not been exceeded. THRXCD = 1: Threshold specified by QWMTHR(A|B).Qn has been exceeded. QSTATn.THRXCD is cleared by CCERRCLR.QTHRXCDn	R	0
23:21	Reserved	Read returns 0.	R	0x0
20:16	WM	Watermark for Maximum Queue Usage: Watermark tracks the most entries that have been in QueueN since reset or since the last time that the watermark (WM) was cleared. QSTATn.WM is cleared through CCERR.WMCLRn bit. Legal values = 0x0 (empty) to 0x10 (full)	R	0x00

Bits	Field Name	Description	Type	Reset
15:13	Reserved	Read returns 0.	R	0x0
12:8	NUMVAL	Number of Valid Entries in Queuei: Represents the total number of entries residing in the Queue Manager FIFO at a given instant. Always enabled. Legal values = 0x0 (empty) to 0x10 (full)	R	0x00
7:4	Reserved	Read returns 0.	R	0x0
3:0	STRTPTR	Start Pointer: Represents the offset to the head entry of Queuei, in units of *entries*. Always enabled. Legal values = 0x0 (0th entry) to 0xF (15th entry)	R	0x0

Table 14-321. Register Call Summary for Register TPCC_QSTATI

IVA2.2 Subsystem Basic Programming Model • Starting the Transfer: [0]
IVA2.2 Subsystem Register Manual • TPCC Register Mapping Summary: [1]

Table 14-322. TPCC_QWMTHRA

Address Offset	0x0620		
Physical address	0x01C0 0620	**Instance**	IVA2.2 TPCC
Description	Queue Threshold A, for Q[3:0]: CCERR.QTHRXCDn and QSTATn.THRXCD error bit is set when the number of Events in QueueN at an instant in time (visible through QSTATn.NUMVAL) equals or exceeds the value specified by QWMTHRA.Qn. Legal values = 0x0 (ever used?) to 0x10 (ever full?) A value of 0x11 disables threshold errors.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																Q1								Reserved				Q0			

Bits	Field Name	Description	Type	Reset
31:13	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0x0010001000
12:8	Q1	Queue Threshold for Q1 value	RW	0x10
7:5	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0x0
4:0	Q0	Queue Threshold for Q0 value	RW	0x10

Table 14-323. Register Call Summary for Register TPCC_QWMTHRA

IVA2.2 Subsystem Basic Programming Model • Starting the Transfer: [0]
IVA2.2 Subsystem Register Manual • TPCC Register Mapping Summary: [1]

Table 14-324. TPCC_QWMTHRB

Address Offset	0x0624		
Physical address	0x01C0 0624	Instance	IVA2.2 TPCC
Description	Queue Threshold B, for Q[7:4]: CCERR.QTHRXCDn and QSTATn.THRXCD error bit is set when the number of Events in QueueN at an instant in time (visible through QSTATn.NUMVAL) equals or exceeds the value specified by QWMTHRB.Qn. Legal values = 0x0 (ever used?) to 0x10 (ever full?) A value of 0x11 disables threshold errors.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																															

Bits	Field Name	Description	Type	Reset
31:0	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0x0010001000100010

Table 14-325. Register Call Summary for Register TPCC_QWMTHRB

IVA2.2 Subsystem Basic Programming Model

- Starting the Transfer: [0]

IVA2.2 Subsystem Register Manual

- TPCC Register Mapping Summary: [1]

Table 14-326. TPCC_CCSTAT

Address Offset	0x0640		
Physical address	0x01C0 0640	**Instance**	IVA2.2 TPCC
Description	CC Status Register		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								QUEACTV7	QUEACTV6	QUEACTV5	QUEACTV4	QUEACTV3	QUEACTV2	QUEACTV1	QUEACTV0	Reserved								Reserved				ACTV	Reserved	TRACTV	QEVTACTV	EVTACTV

Bits	Field Name	Description	Type	Reset
31:24	Reserved	Read returns 0.	R	0x00
23	QUEACTV7	Queue 7 Active QUEACTV7 = 0: No Evts are queued in Q7. QUEACTV7 = 1: At least one TR is queued in Q7.	R	0
22	QUEACTV6	Queue 6 Active QUEACTV6 = 0: No Evts are queued in Q6. QUEACTV6 = 1: At least one TR is queued in Q6.	R	0
21	QUEACTV5	Queue 5 Active QUEACTV5 = 0: No Evts are queued in Q5. QUEACTV5 = 1: At least one TR is queued in Q5.	R	0
20	QUEACTV4	Queue 4 Active QUEACTV4 = 0: No Evts are queued in Q4. QUEACTV4 = 1: At least one TR is queued in Q4.	R	0
19	QUEACTV3	Queue 3 Active QUEACTV3 = 0: No Evts are queued in Q3. QUEACTV3 = 1: At least one TR is queued in Q3.	R	0
18	QUEACTV2	Queue 2 Active QUEACTV2 = 0: No Evts are queued in Q2. QUEACTV2 = 1: At least one TR is queued in Q2.	R	0

Bits	Field Name	Description	Type	Reset
17	QUEACTV1	Queue 1 Active QUEACTV1 = 0: No Evts are queued in Q1. QUEACTV1 = 1: At least one TR is queued in Q1.	R	0
16	QUEACTV0	Queue 0 Active QUEACTV0 = 0: No Evts are queued in Q0. QUEACTV0 = 1: At least one TR is queued in Q0.	R	0
15:14	Reserved	Read returns 0.	R	0x0
13:8	COMPACTV	Completion Request Active: Counter that tracks the total number of completion requests submitted to the TC. The counter increments when a TR is submitted with TCINTEN or TCCHEN set to 1. The counter decrements for every valid completion code received from any of the external TCs. The CC will not service new TRs if COMPACTV count is already at the limit. COMPACTV = 0: No completion requests outstanding. COMPACTV = 1: Total of 1 completion request outstanding. ... COMPACTV = 63 : Total of 63 completion requests are outstanding. No additional TRs will be submitted until count is less than 63.	R	0x00
7:5	Reserved	Read returns 0.	R	0x0
4	ACTV	Channel Controller Active: Channel Controller Active is a logical-OR of each of the *ACTV signals. The ACTV bit must remain high through the life of a TR. ACTV = 0: Channel is idle. ACTV = 1: Channel is busy.	R	0
3	Reserved	Read returns 0.	R	0
2	TRACTV	Transfer Request Active: TRACTV = 0: Transfer Request processing/submission logic is inactive. TRACTV = 1: Transfer Request processing/submission logic is active.	R	0
1	QEVTACTV	QDMA Event Active: QEVTACTV = 0: No enabled QDMA Events are active within the CC. QEVTACTV = 1: At least one enabled DMA Event(ER & EER, ESR, CER) is active within the CC.	R	0
0	EVTACTV	DMA Event Active: EVTACTV = 0: No enabled DMA Events are active within the CC. EVTACTV = 1: At least one enabled DMA Event(ER & EER, ESR, CER) is active within the CC.	R	0

Table 14-327. Register Call Summary for Register TPCC_CCSTAT

IVA2.2 Subsystem Register Manual

- TPCC Register Mapping Summary: [0]

Table 14-328. TPCC_MPFAR

Address Offset	0x0800		
Physical address	0x01C0 0800	**Instance**	IVA2.2 TPCC
Description	Memory Protection Fault Address		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FADDR																															

Bits	Field Name	Description	Type	Reset
31:0	FADDR	Fault Address: 32-bit read-only status register containing the faulting address when a memory protection violation is detected. This register can only be cleared through the MPFCR.	R	0x00000000

Table 14-329. Register Call Summary for Register TPCC_MPFAR

IVA2.2 Subsystem Basic Programming Model

- Internal Memory: [0]

IVA2.2 Subsystem Register Manual

- TPCC Register Mapping Summary: [1]

Table 14-330. TPCC_MPFSR

Address Offset	0x0804		
Physical address	0x01C0 0804	**Instance**	IVA2.2 TPCC
Description	Memory Protection Fault Status Register		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																FID								Reserved		SRE	SWE	SXE	URE	UWE	UXE

Bits	Field Name	Description	Type	Reset
31:13	Reserved	Read returns 0.	R	0x00000
12:9	FID	Faulted ID: FID register contains valid info if any of the MP error bits (UXE, UWE, URE, SXE, SWE, SRE) are non-zero (i.e., if an error has been detected.) The FID field contains the VBus PrivID for the specific request/requestor that resulted in a MP Error.	R	0x0
8:6	Reserved	Read returns 0.	R	x0
5	SRE	Supervisor Read Error: SRE = 0: No error detected. SRE = 1: Supervisor level task attempted to Read from a MP Page without SR permissions.	R	0
4	SWE	Supervisor Write Error: SWE = 0: No error detected. SWE = 1: Supervisor level task attempted to Write to a MP Page without SW permissions.	R	0
3	SXE	Supervisor Execute Error: SXE = 0: No error detected. SXE = 1: Supervisor level task attempted to Execute from a MP Page without SX permissions.	R	0
2	URE	User Read Error: URE = 0: No error detected. URE = 1: User level task attempted to Read from a MP Page without UR permissions.	R	0
1	UWE	User Write Error: UWE = 0: No error detected. UWE = 1: User level task attempted to Write to a MP Page without UW permissions.	R	0
0	UXE	User Execute Error: UXE = 0: No error detected. UXE = 1: User level task attempted to Execute from a MP Page without UX permissions.	R	0

Table 14-331. Register Call Summary for Register TPCC_MPFSR

IVA2.2 Subsystem Basic Programming Model

- Internal Memory: [0]

IVA2.2 Subsystem Register Manual

- TPCC Register Mapping Summary: [1]

Table 14-332. TPCC_MPFCR

Address Offset	0x0808		
Physical address	0x01C0 0808	**Instance**	IVA2.2 TPCC
Description	Memory Protection Fault Command Register		
Type	W		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																															MPFCLR

Bits	Field Name	Description	Type	Reset
31:1	Reserved	Write 0s for future compatibility.	W	0x00000000
0	MPFCLR	Fault Clear register: CPU write of 1 to the MPFCLR bit causes any error conditions stored in MPFAR and MPFSR registers to be cleared. CPU write of 0 has no effect.	W	0

Table 14-333. Register Call Summary for Register TPCC_MPFCR

IVA2.2 Subsystem Basic Programming Model

- Internal Memory: [0]

IVA2.2 Subsystem Register Manual

- TPCC Register Mapping Summary: [1]

Table 14-334. TPCC_MPPAG

Address Offset	0x080C		
Physical address	0x01C0 080C	**Instance**	IVA2.2 TPCC
Description	Memory Protection Page Attribute for Global registers		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																AID5	AID4	AID3	AID2	AID1	AID0	EXT	Reserved			SR	SW	SX	UR	UW	UX

Bits	Field Name	Description	Type	Reset
31:16	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0x0000
15	AID5	Allowed ID 5: AID5 = 0: VBus requests with PrivID == 5 are notallowed regardless of permission settings (UW, UR, SW, SR). AID5 = 1: VBus requests with PrivID == 5 are permitted if access type is allowed as defined by permission settings (UW, UR, SW, SR).	RW	1
14	AID4	Allowed ID 4: AID4 = 0: VBus requests with PrivID == 4 are notallowed regardless of permission settings (UW, UR, SW, SR). AID4 = 1: VBus requests with PrivID == 4 are permitted if access type is allowed as defined by permission settings (UW, UR, SW, SR).	RW	1
13	AID3	Allowed ID 3: AID3 = 0: VBus requests with PrivID == 3 are notallowed regardless of permission settings (UW, UR, SW, SR). AID3 = 1: VBus requests with PrivID == 3 are permitted if access type is allowed as defined by permission settings (UW, UR, SW, SR).	RW	1

Bits	Field Name	Description	Type	Reset
12	AID2	Allowed ID 2: AID2 = 0: VBus requests with PrivID == 2 are notallowed regardless of permission settings (UW, UR, SW, SR). AID2 = 1: VBus requests with PrivID == 2 are permitted if access type is allowed as defined by permission settings (UW, UR, SW, SR).	RW	1
11	AID1	Allowed ID 1: AID1 = 0: VBus requests with PrivID == 1 are notallowed regardless of permission settings (UW, UR, SW, SR). AID1 = 1: VBus requests with PrivID == 1 are permitted if access type is allowed as defined by permission settings (UW, UR, SW, SR).	RW	1
10	AID0	Allowed ID 0: AID0 = 0: VBus requests with PrivID == 0 are notallowed regardless of permission settings (UW, UR, SW, SR). AID0 = 1: VBus requests with PrivID == 0 are permitted if access type is allowed as defined by permission settings (UW, UR, SW, SR).	RW	1
9	EXT	External Allowed ID: AIDm = 0: VBus requests with PrivID >= 6 are notallowed regardless of permission settings (UW, UR, SW, SR). AIDm = 1: VBus requests with PrivID >= 6 are permitted if access type is allowed as defined by permission settings (UW, UR, SW, SR).	RW	1
8:6	Reserved	Write reset value for future compatibility. Read returns reset value.	RW	0x7
5	SR	Supervisor Read permission: SR = 0: Supervisor read accesses are not allowed SR = 1: Supervisor write accesses are allowed	RW	1
4	SW	Supervisor Write permission: SW = 0: Supervisor write accesses are not allowed SW = 1: Supervisor write accesses are allowed	RW	1
3	SX	Supervisor Execute permission: SX = 0: Supervisor execute accesses are not allowed SX = 1: Supervisor execute accesses are allowed	RW	0
2	UR	User Read permission: UR = 0: User read accesses are not allowed UR = 1: User write accesses are allowed	RW	1
1	UW	User Write permission: UW = 0: User write accesses are not allowed UW = 1: User write accesses are allowed	RW	1
0	UX	User Execute permission: UX = 0: User execute accesses are not allowed UX = 1: User execute accesses are allowed	RW	0

Table 14-335. Register Call Summary for Register TPCC_MPPAG

IVA2.2 Subsystem Basic Programming Model

- Internal Memory: [0] [1] [2]

IVA2.2 Subsystem Register Manual

- TPCC Register Mapping Summary: [3]

Table 14-336. TPCC_MPPAj

Address Offset	0x0810 + (0x4*j)		
Physical address	0x01C0 0810 + (0x4*j)	**Instance**	IVA2.2 TPCC
Description	MP Permission Attribute for DMA Region n		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																AID5	AID4	AID3	AID2	AID1	AID0	EXT	Reserved			SR	SW	SX	UR	UW	UX

Bits	Field Name	Description	Type	Reset
31:16	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0x0000
15	AID5	Allowed ID 5: AID5 = 0: VBus requests with PrivID == 5 are notallowed regardless of permission settings (UW, UR, SW, SR). AID5 = 1: VBus requests with PrivID == 5 are permitted if access type is allowed as defined by permission settings (UW, UR, SW, SR).	RW	1
14	AID4	Allowed ID 4: AID4 = 0: VBus requests with PrivID == 4 are notallowed regardless of permission settings (UW, UR, SW, SR). AID4 = 1: VBus requests with PrivID == 4 are permitted if access type is allowed as defined by permission settings (UW, UR, SW, SR).	RW	1
13	AID3	Allowed ID 3: AID3 = 0: VBus requests with PrivID == 3 are notallowed regardless of permission settings (UW, UR, SW, SR). AID3 = 1: VBus requests with PrivID == 3 are permitted if access type is allowed as defined by permission settings (UW, UR, SW, SR).	RW	1
12	AID2	Allowed ID 2: AID2 = 0: VBus requests with PrivID == 2 are notallowed regardless of permission settings (UW, UR, SW, SR). AID2 = 1: VBus requests with PrivID == 2 are permitted if access type is allowed as defined by permission settings (UW, UR, SW, SR).	RW	1
11	AID1	Allowed ID 1: AID1 = 0: VBus requests with PrivID == 1 are notallowed regardless of permission settings (UW, UR, SW, SR). AID1 = 1: VBus requests with PrivID == 1 are permitted if access type is allowed as defined by permission settings (UW, UR, SW, SR).	RW	1
10	AID0	Allowed ID 0: AID0 = 0: VBus requests with PrivID == 0 are notallowed regardless of permission settings (UW, UR, SW, SR). AID0 = 1: VBus requests with PrivID == 0 are permitted if access type is allowed as defined by permission settings (UW, UR, SW, SR).	RW	1
9	EXT	External Allowed ID: AIDm = 0: VBus requests with PrivID >= 6 are notallowed regardless of permission settings (UW, UR, SW, SR). AIDm = 1: VBus requests with PrivID >= 6 are permitted if access type is allowed as defined by permission settings (UW, UR, SW, SR).	RW	1
8:6	Reserved	Write reset value for future compatibility. Read returns reset value.	RW	0x3
5	SR	Supervisor Read permission: SR = 0: Supervisor read accesses are not allowed SR = 1: Supervisor write accesses are allowed	RW	1
4	SW	Supervisor Write permission: SW = 0: Supervisor write accesses are not allowed SW = 1: Supervisor write accesses are allowed	RW	1

Bits	Field Name	Description	Type	Reset
3	SX	Supervisor Execute permission: SX = 0: Supervisor execute accesses are not allowed SX = 1: Supervisor execute accesses are allowed	RW	0
2	UR	User Read permission: UR = 0: User read accesses are not allowed UR = 1: User write accesses are allowed	RW	1
1	UW	User Write permission: UW = 0: User write accesses are not allowed UW = 1: User write accesses are allowed	RW	1
0	UX	User Execute permission: UX = 0: User execute accesses are not allowed UX = 1: User execute accesses are allowed	RW	0

Table 14-337. Register Call Summary for Register TPCC_MPPAj

IVA2.2 Subsystem Basic Programming Model

- Internal Memory: [0] [1] [2] [3] [4]

IVA2.2 Subsystem Register Manual

- TPCC Register Mapping Summary: [5]

Table 14-338. TPCC_ER

Address Offset	0x1000		
Physical address	0x01C0 1000	**Instance**	IVA2.2 TPCC
Description	Event Register: If ER.En bit is set and the EER.En bit is also set, then the corresponding DMA channel is prioritized vs. other pending DMA events for submission to the TC. ER.En bit is set when the input event #n transitions from inactive (low) to active (high), regardless of the state of EER.En bit. ER.En bit is cleared when the corresponding event is prioritized and serviced. If the ER.En bit is already set and a new inactive to active transition is detected on the input event #n input AND the corresponding bit in the EER register is set, then the corresponding bit in the Event Missed Register is set. Event N can be cleared through sw by writing 1 to the ECR pseudo-register.		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved												E19	E18	E17	E16	E15	E14	E13	E12	E11	E10	E9	E8	E7	E6	E5	E4	E3	E2	E1	E0

Bits	Field Name	Description	Type	Reset
31:20	Reserved	Reserved	R	0
19	E19	Event #19	R	0
18	E18	Event #18	R	0
17	E17	Event #17	R	0
16	E16	Event #16	R	0
15	E15	Event #15	R	0
14	E14	Event #14	R	0
13	E13	Event #13	R	0
12	E12	Event #12	R	0
11	E11	Event #11	R	0
10	E10	Event #10	R	0
9	E9	Event #9	R	0
8	E8	Event #8	R	0
7	E7	Event #7	R	0
6	E6	Event #6	R	0
5	E5	Event #5	R	0
4	E4	Event #4	R	0
3	E3	Event #3	R	0

Bits	Field Name	Description	Type	Reset
2	E2	Event #2	R	0
1	E1	Event #1	R	0
0	E0	Event #0	R	0

Table 14-339. Register Call Summary for Register TPCC_ER

IVA2.2 Subsystem Functional Description
- EDMA: [0] [1]

IVA2.2 Subsystem Register Manual
- TPCC Register Mapping Summary: [2]

Table 14-340. TPCC_ECR

Address Offset 0x1008

Physical address 0x01C0 1008 **Instance** IVA2.2 TPCC

Description Event Clear Register:
CPU write of 1 to the ECR.En bit causes the ER.En bit to be cleared.
CPU write of 0 has no effect.

Type W

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
E31	E30	E29	E28	E27	E26	E25	E24	E23	E22	E21	E20	E19	E18	E17	E16	E15	E14	E13	E12	E11	E10	E9	E8	E7	E6	E5	E4	E3	E2	E1	E0

Bits	Field Name	Description	Type	Reset
31	E31	Event #31	W	0
30	E30	Event #30	W	0
29	E29	Event #29	W	0
28	E28	Event #28	W	0
27	E27	Event #27	W	0
26	E26	Event #26	W	0
25	E25	Event #25	W	0
24	E24	Event #24	W	0
23	E23	Event #23	W	0
22	E22	Event #22	W	0
21	E21	Event #21	W	0
20	E20	Event #20	W	0
19	E19	Event #19	W	0
18	E18	Event #18	W	0
17	E17	Event #17	W	0
16	E16	Event #16	W	0
15	E15	Event #15	W	0
14	E14	Event #14	W	0
13	E13	Event #13	W	0
12	E12	Event #12	W	0
11	E11	Event #11	W	0
10	E10	Event #10	W	0
9	E9	Event #9	W	0
8	E8	Event #8	W	0
7	E7	Event #7	W	0
6	E6	Event #6	W	0
5	E5	Event #5	W	0

Bits	Field Name	Description	Type	Reset
4	E4	Event #4	W	0
3	E3	Event #3	W	0
2	E2	Event #2	W	0
1	E1	Event #1	W	0
0	E0	Event #0	W	0

Table 14-341. Register Call Summary for Register TPCC_ECR

IVA2.2 Subsystem Register Manual

- TPCC Register Mapping Summary: [0]

Table 14-342. TPCC_ECRH

Address Offset	0x100C		
Physical address	0x01C0 100C	**Instance**	IVA2.2 TPCC
Description	Event Clear Register (High Part): CPU write of 1 to the ECRH.En bit causes the ERH.En bit to be cleared. CPU write of 0 has no effect.		
Type	W		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
E63	E62	E61	E60	E59	E58	E57	E56	E55	E54	E53	E52	E51	E50	E49	E48	E47	E46	E45	E44	E43	E42	E41	E40	E39	E38	E37	E36	E35	E34	E33	E32

Bits	Field Name	Description	Type	Reset
31	E63	Event #63	W	0
30	E62	Event #62	W	0
29	E61	Event #61	W	0
28	E60	Event #60	W	0
27	E59	Event #59	W	0
26	E58	Event #58	W	0
25	E57	Event #57	W	0
24	E56	Event #56	W	0
23	E55	Event #55	W	0
22	E54	Event #54	W	0
21	E53	Event #53	W	0
20	E52	Event #52	W	0
19	E51	Event #51	W	0
18	E50	Event #50	W	0
17	E49	Event #49	W	0
16	E48	Event #48	W	0
15	E47	Event #47	W	0
14	E46	Event #46	W	0
13	E45	Event #45	W	0
12	E44	Event #44	W	0
11	E43	Event #43	W	0
10	E42	Event #42	W	0
9	E41	Event #41	W	0
8	E40	Event #40	W	0
7	E39	Event #39	W	0
6	E38	Event #38	W	0
5	E37	Event #37	W	0

Bits	Field Name	Description	Type	Reset
4	E36	Event #36	W	0
3	E35	Event #35	W	0
2	E34	Event #34	W	0
1	E33	Event #33	W	0
0	E32	Event #32	W	0

Table 14-343. Register Call Summary for Register TPCC_ECRH

IVA2.2 Subsystem Register Manual

- TPCC Register Mapping Summary: [0]

Table 14-344. TPCC_ESR

Address Offset	0x1010		
Physical address	0x01C0 1010	**Instance**	IVA2.2 TPCC
Description	Event Set Register: CPU write of 1 to the ESR.En bit causes the ER.En bit to be set. CPU write of 0 has no effect.		
Type	W		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
E31	E30	E29	E28	E27	E26	E25	E24	E23	E22	E21	E20	E19	E18	E17	E16	E15	E14	E13	E12	E11	E10	E9	E8	E7	E6	E5	E4	E3	E2	E1	E0

Bits	Field Name	Description	Type	Reset
31	E31	Event #31	W	0
30	E30	Event #30	W	0
29	E29	Event #29	W	0
28	E28	Event #28	W	0
27	E27	Event #27	W	0
26	E26	Event #26	W	0
25	E25	Event #25	W	0
24	E24	Event #24	W	0
23	E23	Event #23	W	0
22	E22	Event #22	W	0
21	E21	Event #21	W	0
20	E20	Event #20	W	0
19	E19	Event #19	W	0
18	E18	Event #18	W	0
17	E17	Event #17	W	0
16	E16	Event #16	W	0
15	E15	Event #15	W	0
14	E14	Event #14	W	0
13	E13	Event #13	W	0
12	E12	Event #12	W	0
11	E11	Event #11	W	0
10	E10	Event #10	W	0
9	E9	Event #9	W	0
8	E8	Event #8	W	0
7	E7	Event #7	W	0
6	E6	Event #6	W	0
5	E5	Event #5	W	0

Bits	Field Name	Description	Type	Reset
4	E4	Event #4	W	0
3	E3	Event #3	W	0
2	E2	Event #2	W	0
1	E1	Event #1	W	0
0	E0	Event #0	W	0

Table 14-345. Register Call Summary for Register TPCC_ESR

IVA2.2 Subsystem Functional Description • EDMA: [0] [1] [2]
IVA2.2 Subsystem Basic Programming Model • Starting the Transfer: [3]
IVA2.2 Subsystem Register Manual • TPCC Register Mapping Summary: [4]

Table 14-346. TPCC_ESRH

Address Offset	0x1014		
Physical address	0x01C0 1014	**Instance**	IVA2.2 TPCC
Description	Event Set Register (High Part) CPU write of 1 to the ESRH.En bit causes the ERH.En bit to be set. CPU write of 0 has no effect.		
Type	W		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
E63	E62	E61	E60	E59	E58	E57	E56	E55	E54	E53	E52	E51	E50	E49	E48	E47	E46	E45	E44	E43	E42	E41	E40	E39	E38	E37	E36	E35	E34	E33	E32

Bits	Field Name	Description	Type	Reset
31	E63	Event #63	W	0
30	E62	Event #62	W	0
29	E61	Event #61	W	0
28	E60	Event #60	W	0
27	E59	Event #59	W	0
26	E58	Event #58	W	0
25	E57	Event #57	W	0
24	E56	Event #56	W	0
23	E55	Event #55	W	0
22	E54	Event #54	W	0
21	E53	Event #53	W	0
20	E52	Event #52	W	0
19	E51	Event #51	W	0
18	E50	Event #50	W	0
17	E49	Event #49	W	0
16	E48	Event #48	W	0
15	E47	Event #47	W	0
14	E46	Event #46	W	0
13	E45	Event #45	W	0
12	E44	Event #44	W	0
11	E43	Event #43	W	0
10	E42	Event #42	W	0
9	E41	Event #41	W	0

Bits	Field Name	Description	Type	Reset
8	E40	Event #40	W	0
7	E39	Event #39	W	0
6	E38	Event #38	W	0
5	E37	Event #37	W	0
4	E36	Event #36	W	0
3	E35	Event #35	W	0
2	E34	Event #34	W	0
1	E33	Event #33	W	0
0	E32	Event #32	W	0

Table 14-347. Register Call Summary for Register TPCC_ESRH

IVA2.2 Subsystem Functional Description
- EDMA: [0] [1]

IVA2.2 Subsystem Basic Programming Model
- Starting the Transfer: [2]

IVA2.2 Subsystem Register Manual
- TPCC Register Mapping Summary: [3]

Table 14-348. TPCC_CER

Address Offset	0x1018		
Physical address	0x01C0 1018	**Instance**	IVA2.2 TPCC
Description	Chained Event Register: If CER.En bit is set (regardless of state of EER.En), then the corresponding DMA channel is prioritized vs. other pending DMA events for submission to the TC. CER.En bit is set when a chaining completion code is returned from one of the TPTCs through the completion interface, or is generated internally through Early Completion path. CER.En bit is cleared when the corresponding event is prioritized and serviced. If the CER.En bit is already set and the corresponding chaining completion code is returned from the TC, then the corresponding bit in the Event Missed Register is set. CER.En cannot be set or cleared through software.		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
E31	E30	E29	E28	E27	E26	E25	E24	E23	E22	E21	E20	E19	E18	E17	E16	E15	E14	E13	E12	E11	E10	E9	E8	E7	E6	E5	E4	E3	E2	E1	E0

Bits	Field Name	Description	Type	Reset
31	E31	Event #31	R	0
30	E30	Event #30	R	0
29	E29	Event #29	R	0
28	E28	Event #28	R	0
27	E27	Event #27	R	0
26	E26	Event #26	R	0
25	E25	Event #25	R	0
24	E24	Event #24	R	0
23	E23	Event #23	R	0
22	E22	Event #22	R	0
21	E21	Event #21	R	0
20	E20	Event #20	R	0
19	E19	Event #19	R	0
18	E18	Event #18	R	0
17	E17	Event #17	R	0
16	E16	Event #16	R	0

Bits	Field Name	Description	Type	Reset
15	E15	Event #15	R	0
14	E14	Event #14	R	0
13	E13	Event #13	R	0
12	E12	Event #12	R	0
11	E11	Event #11	R	0
10	E10	Event #10	R	0
9	E9	Event #9	R	0
8	E8	Event #8	R	0
7	E7	Event #7	R	0
6	E6	Event #6	R	0
5	E5	Event #5	R	0
4	E4	Event #4	R	0
3	E3	Event #3	R	0
2	E2	Event #2	R	0
1	E1	Event #1	R	0
0	E0	Event #0	R	0

Table 14-349. Register Call Summary for Register TPCC_CER

IVA2.2 Subsystem Functional Description

- EDMA: [0] [1]

IVA2.2 Subsystem Register Manual

- TPCC Register Mapping Summary: [2]

Table 14-350. TPCC_CERH

Address Offset	0x101C		
Physical address	0x01C0 101C	**Instance**	IVA2.2 TPCC
Description	Chained Event Register (High Part): If CERH.En bit is set (regardless of state of EERH.En), then the corresponding DMA channel is prioritized vs. other pending DMA events for submission to the TC. CERH.En bit is set when a chaining completion code is returned from one of the TPTCs through the completion interface, or is generated internally through Early Completion path. CERH.En bit is cleared when the corresponding event is prioritized and serviced. If the CERH.En bit is already set and the corresponding chaining completion code is returned from the TC, then the corresponding bit in the Event Missed Register is set. CERH.En cannot be set or cleared through software.		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
E63	E62	E61	E60	E59	E58	E57	E56	E55	E54	E53	E52	E51	E50	E49	E48	E47	E46	E45	E44	E43	E42	E41	E40	E39	E38	E37	E36	E35	E34	E33	E32

Bits	Field Name	Description	Type	Reset
31	E63	Event #63	R	0
30	E62	Event #62	R	0
29	E61	Event #61	R	0
28	E60	Event #60	R	0
27	E59	Event #59	R	0
26	E58	Event #58	R	0
25	E57	Event #57	R	0
24	E56	Event #56	R	0
23	E55	Event #55	R	0
22	E54	Event #54	R	0
21	E53	Event #53	R	0

Bits	Field Name	Description	Type	Reset
20	E52	Event #52	R	0
19	E51	Event #51	R	0
18	E50	Event #50	R	0
17	E49	Event #49	R	0
16	E48	Event #48	R	0
15	E47	Event #47	R	0
14	E46	Event #46	R	0
13	E45	Event #45	R	0
12	E44	Event #44	R	0
11	E43	Event #43	R	0
10	E42	Event #42	R	0
9	E41	Event #41	R	0
8	E40	Event #40	R	0
7	E39	Event #39	R	0
6	E38	Event #38	R	0
5	E37	Event #37	R	0
4	E36	Event #36	R	0
3	E35	Event #35	R	0
2	E34	Event #34	R	0
1	E33	Event #33	R	0
0	E32	Event #32	R	0

Table 14-351. Register Call Summary for Register TPCC_CERH

IVA2.2 Subsystem Functional Description

- EDMA: [0]

IVA2.2 Subsystem Register Manual

- TPCC Register Mapping Summary: [1]

Table 14-352. TPCC_EER

Address Offset	0x1020		
Physical address	0x01C0 1020	**Instance**	IVA2.2 TPCC
Description	Event Enable Register: Enables DMA transfers for ER.En pending events. ER.En is set based on externally asserted events (through tpcc_eventN_pi). This register has no effect on Chained Event Register (CER) or Event Set Register (ESR). Note that if a bit is set in ER.En while EER.En is disabled, no action is taken. If EER.En is enabled at a later point (and ER.En has not been cleared through SW) then the event will be recognized as a valid TR Sync EER.En is not directly writeable. Events can be enabled through writes to EESR and can be disabled through writes to EECR register. EER.En = 0: ER.En is not enabled to trigger DMA transfers. EER.En = 1: ER.En is enabled to trigger DMA transfers.		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved												E19	E18	E17	E16	E15	E14	E13	E12	E11	E10	E9	E8	E7	E6	E5	E4	E3	E2	E1	E0

Bits	Field Name	Description	Type	Reset
31:20	Reserved	Reserved	R	0
19	E19	Event #19	R	0
18	E18	Event #18	R	0
17	E17	Event #17	R	0
16	E16	Event #16	R	0

Bits	Field Name	Description	Type	Reset
15	E15	Event #15	R	0
14	E14	Event #14	R	0
13	E13	Event #13	R	0
12	E12	Event #12	R	0
11	E11	Event #11	R	0
10	E10	Event #10	R	0
9	E9	Event #9	R	0
8	E8	Event #8	R	0
7	E7	Event #7	R	0
6	E6	Event #6	R	0
5	E5	Event #5	R	0
4	E4	Event #4	R	0
3	E3	Event #3	R	0
2	E2	Event #2	R	0
1	E1	Event #1	R	0
0	E0	Event #0	R	0

Table 14-353. Register Call Summary for Register TPCC_EER

IVA2.2 Subsystem Functional Description
- EDMA: [0] [1]

IVA2.2 Subsystem Basic Programming Model
- Starting the Transfer: [2]

IVA2.2 Subsystem Register Manual
- TPCC Register Mapping Summary: [3]

Table 14-354. TPCC_EECR

Address Offset 0x1028

Physical address 0x01C0 1028 **Instance** IVA2.2 TPCC

Description Event Enable Clear Register:
CPU write of 1 to the EECR.En bit causes the EER.En bit to be cleared.
CPU write of 0 has no effect.

Type W

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved												E19	E18	E17	E16	E15	E14	E13	E12	E11	E10	E9	E8	E7	E6	E5	E4	E3	E2	E1	E0

Bits	Field Name	Description	Type	Reset
31:20	Reserved	Reserved	W	0
19	E19	Event #19	W	0
18	E18	Event #18	W	0
17	E17	Event #17	W	0
16	E16	Event #16	W	0
15	E15	Event #15	W	0
14	E14	Event #14	W	0
13	E13	Event #13	W	0
12	E12	Event #12	W	0
11	E11	Event #11	W	0
10	E10	Event #10	W	0
9	E9	Event #9	W	0

Bits	Field Name	Description	Type	Reset
8	E8	Event #8	W	0
7	E7	Event #7	W	0
6	E6	Event #6	W	0
5	E5	Event #5	W	0
4	E4	Event #4	W	0
3	E3	Event #3	W	0
2	E2	Event #2	W	0
1	E1	Event #1	W	0
0	E0	Event #0	W	0

Table 14-355. Register Call Summary for Register TPCC_EECR

IVA2.2 Subsystem Register Manual

- TPCC Register Mapping Summary: [0]

Table 14-356. TPCC_EESR

Address Offset	0x1030		
Physical address	0x01C0 1030	**Instance**	IVA2.2 TPCC
Description	Event Enable Set Register: CPU write of 1 to the EESR.En bit causes the EER.En bit to be set. CPU write of 0 has no effect.		
Type	W		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								E19	E18	E17	E16	E15	E14	E13	E12	E11	E10	E9	E8	E7	E6	E5	E4	E3	E2	E1	E0				

Bits	Field Name	Description	Type	Reset
31:20	Reserved	Reserved	W	0
19	E19	Event #19	W	0
18	E18	Event #18	W	0
17	E17	Event #17	W	0
16	E16	Event #16	W	0
15	E15	Event #15	W	0
14	E14	Event #14	W	0
13	E13	Event #13	W	0
12	E12	Event #12	W	0
11	E11	Event #11	W	0
10	E10	Event #10	W	0
9	E9	Event #9	W	0
8	E8	Event #8	W	0
7	E7	Event #7	W	0
6	E6	Event #6	W	0
5	E5	Event #5	W	0
4	E4	Event #4	W	0
3	E3	Event #3	W	0
2	E2	Event #2	W	0
1	E1	Event #1	W	0
0	E0	Event #0	W	0

Table 14-357. Register Call Summary for Register TPCC_EESR

IVA2.2 Subsystem Register Manual

- TPCC Register Mapping Summary: [0]

Table 14-358. TPCC_SER

Address Offset	0x1038		
Physical address	0x01C0 1038	**Instance**	IVA2.2 TPCC
Description	Secondary Event Register: The secondary event register is used along with the Event Register (ER) to provide information on the state of an Event. En = 0: Event is not currently in the Event Queue. En = 1: Event is currently stored in Event Queue. Event arbiter will not prioritize additional events.		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
E31	E30	E29	E28	E27	E26	E25	E24	E23	E22	E21	E20	E19	E18	E17	E16	E15	E14	E13	E12	E11	E10	E9	E8	E7	E6	E5	E4	E3	E2	E1	E0

Bits	Field Name	Description	Type	Reset
31	E31	Event #31	R	0
30	E30	Event #30	R	0
29	E29	Event #29	R	0
28	E28	Event #28	R	0
27	E27	Event #27	R	0
26	E26	Event #26	R	0
25	E25	Event #25	R	0
24	E24	Event #24	R	0
23	E23	Event #23	R	0
22	E22	Event #22	R	0
21	E21	Event #21	R	0
20	E20	Event #20	R	0
19	E19	Event #19	R	0
18	E18	Event #18	R	0
17	E17	Event #17	R	0
16	E16	Event #16	R	0
15	E15	Event #15	R	0
14	E14	Event #14	R	0
13	E13	Event #13	R	0
12	E12	Event #12	R	0
11	E11	Event #11	R	0
10	E10	Event #10	R	0
9	E9	Event #9	R	0
8	E8	Event #8	R	0
7	E7	Event #7	R	0
6	E6	Event #6	R	0
5	E5	Event #5	R	0
4	E4	Event #4	R	0
3	E3	Event #3	R	0
2	E2	Event #2	R	0
1	E1	Event #1	R	0
0	E0	Event #0	R	0

Table 14-359. Register Call Summary for Register TPCC_SER

IVA2.2 Subsystem Register Manual

- TPCC Register Mapping Summary: [0]

Table 14-360. TPCC_SERH

Address Offset	0x103C		
Physical address	0x01C0 103C	**Instance**	IVA2.2 TPCC
Description	Secondary Event Register (High Part): The secondary event register is used along with the Event Register (ERH) to provide information on the state of an Event. En = 0: Event is not currently in the Event Queue. En = 1: Event is currently stored in Event Queue. Event arbiter will not prioritize additional events.		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
E63	E62	E61	E60	E59	E58	E57	E56	E55	E54	E53	E52	E51	E50	E49	E48	E47	E46	E45	E44	E43	E42	E41	E40	E39	E38	E37	E36	E35	E34	E33	E32

Bits	Field Name	Description	Type	Reset
31	E63	Event #63	R	0
30	E62	Event #62	R	0
29	E61	Event #61	R	0
28	E60	Event #60	R	0
27	E59	Event #59	R	0
26	E58	Event #58	R	0
25	E57	Event #57	R	0
24	E56	Event #56	R	0
23	E55	Event #55	R	0
22	E54	Event #54	R	0
21	E53	Event #53	R	0
20	E52	Event #52	R	0
19	E51	Event #51	R	0
18	E50	Event #50	R	0
17	E49	Event #49	R	0
16	E48	Event #48	R	0
15	E47	Event #47	R	0
14	E46	Event #46	R	0
13	E45	Event #45	R	0
12	E44	Event #44	R	0
11	E43	Event #43	R	0
10	E42	Event #42	R	0
9	E41	Event #41	R	0
8	E40	Event #40	R	0
7	E39	Event #39	R	0
6	E38	Event #38	R	0
5	E37	Event #37	R	0
4	E36	Event #36	R	0
3	E35	Event #35	R	0
2	E34	Event #34	R	0
1	E33	Event #33	R	0
0	E32	Event #32	R	0

Table 14-361. Register Call Summary for Register TPCC_SERH

IVA2.2 Subsystem Register Manual

- TPCC Register Mapping Summary: [0]

Table 14-362. TPCC_SECR

Address Offset	0x1040		
Physical address	0x01C0 1040	**Instance**	IVA2.2 TPCC
Description	Secondary Event Clear Register: The secondary event clear register is used to clear the status of the SER registers. CPU write of 1 to the SECR.En bit clears the SER register. CPU write of 0 has no effect.		
Type	W		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
E31	E30	E29	E28	E27	E26	E25	E24	E23	E22	E21	E20	E19	E18	E17	E16	E15	E14	E13	E12	E11	E10	E9	E8	E7	E6	E5	E4	E3	E2	E1	E0

Bits	Field Name	Description	Type	Reset
31	E31	Event #31	W	0
30	E30	Event #30	W	0
29	E29	Event #29	W	0
28	E28	Event #28	W	0
27	E27	Event #27	W	0
26	E26	Event #26	W	0
25	E25	Event #25	W	0
24	E24	Event #24	W	0
23	E23	Event #23	W	0
22	E22	Event #22	W	0
21	E21	Event #21	W	0
20	E20	Event #20	W	0
19	E19	Event #19	W	0
18	E18	Event #18	W	0
17	E17	Event #17	W	0
16	E16	Event #16	W	0
15	E15	Event #15	W	0
14	E14	Event #14	W	0
13	E13	Event #13	W	0
12	E12	Event #12	W	0
11	E11	Event #11	W	0
10	E10	Event #10	W	0
9	E9	Event #9	W	0
8	E8	Event #8	W	0
7	E7	Event #7	W	0
6	E6	Event #6	W	0
5	E5	Event #5	W	0
4	E4	Event #4	W	0
3	E3	Event #3	W	0
2	E2	Event #2	W	0
1	E1	Event #1	W	0
0	E0	Event #0	W	0

Table 14-363. Register Call Summary for Register TPCC_SECR

IVA2.2 Subsystem Register Manual

- TPCC Register Mapping Summary: [0]

Table 14-364. TPCC_SECRH

Address Offset	0x1044		
Physical address	0x01C0 1044	**Instance**	IVA2.2 TPCC
Description	Secondary Event Clear Register (High Part): The secondary event clear register is used to clear the status of the SERH registers. CPU write of 1 to the SECRH.En bit clears the SERH register. CPU write of 0 has no effect.		
Type	W		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
E63	E62	E61	E60	E59	E58	E57	E56	E55	E54	E53	E52	E51	E50	E49	E48	E47	E46	E45	E44	E43	E42	E41	E40	E39	E38	E37	E36	E35	E34	E33	E32

Bits	Field Name	Description	Type	Reset
31	E63	Event #63	W	0
30	E62	Event #62	W	0
29	E61	Event #61	W	0
28	E60	Event #60	W	0
27	E59	Event #59	W	0
26	E58	Event #58	W	0
25	E57	Event #57	W	0
24	E56	Event #56	W	0
23	E55	Event #55	W	0
22	E54	Event #54	W	0
21	E53	Event #53	W	0
20	E52	Event #52	W	0
19	E51	Event #51	W	0
18	E50	Event #50	W	0
17	E49	Event #49	W	0
16	E48	Event #48	W	0
15	E47	Event #47	W	0
14	E46	Event #46	W	0
13	E45	Event #45	W	0
12	E44	Event #44	W	0
11	E43	Event #43	W	0
10	E42	Event #42	W	0
9	E41	Event #41	W	0
8	E40	Event #40	W	0
7	E39	Event #39	W	0
6	E38	Event #38	W	0
5	E37	Event #37	W	0
4	E36	Event #36	W	0
3	E35	Event #35	W	0
2	E34	Event #34	W	0
1	E33	Event #33	W	0
0	E32	Event #32	W	0

Table 14-365. Register Call Summary for Register TPCC_SECRH

IVA2.2 Subsystem Register Manual

- TPCC Register Mapping Summary: [0]

Table 14-366. TPCC_IER

Address Offset	0x1050		
Physical address	0x01C0 1050	**Instance**	IVA2.2 TPCC
Description	Int Enable Register: IER.In is not directly writeable. Interrupts can be enabled through writes to IESR and can be disabled through writes to IECR register. IER.In = 0: IPR.In is NOT enabled for interrupts. IER.In = 1: IPR.In IS enabled for interrupts.		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
I31	I30	I29	I28	I27	I26	I25	I24	I23	I22	I21	I20	I19	I18	I17	I16	I15	I14	I13	I12	I11	I10	I9	I8	I7	I6	I5	I4	I3	I2	I1	I0

Bits	Field Name	Description	Type	Reset
31	I31	Interrupt associated with TCC #31	R	0
30	I30	Interrupt associated with TCC #30	R	0
29	I29	Interrupt associated with TCC #29	R	0
28	I28	Interrupt associated with TCC #28	R	0
27	I27	Interrupt associated with TCC #27	R	0
26	I26	Interrupt associated with TCC #26	R	0
25	I25	Interrupt associated with TCC #25	R	0
24	I24	Interrupt associated with TCC #24	R	0
23	I23	Interrupt associated with TCC #23	R	0
22	I22	Interrupt associated with TCC #22	R	0
21	I21	Interrupt associated with TCC #21	R	0
20	I20	Interrupt associated with TCC #20	R	0
19	I19	Interrupt associated with TCC #19	R	0
18	I18	Interrupt associated with TCC #18	R	0
17	I17	Interrupt associated with TCC #17	R	0
16	I16	Interrupt associated with TCC #16	R	0
15	I15	Interrupt associated with TCC #15	R	0
14	I14	Interrupt associated with TCC #14	R	0
13	I13	Interrupt associated with TCC #13	R	0
12	I12	Interrupt associated with TCC #12	R	0
11	I11	Interrupt associated with TCC #11	R	0
10	I10	Interrupt associated with TCC #10	R	0
9	I9	Interrupt associated with TCC #9	R	0
8	I8	Interrupt associated with TCC #8	R	0
7	I7	Interrupt associated with TCC #7	R	0
6	I6	Interrupt associated with TCC #6	R	0
5	I5	Interrupt associated with TCC #5	R	0
4	I4	Interrupt associated with TCC #4	R	0
3	I3	Interrupt associated with TCC #3	R	0
2	I2	Interrupt associated with TCC #2	R	0
1	I1	Interrupt associated with TCC #1	R	0
0	I0	Interrupt associated with TCC #0	R	0

Table 14-367. Register Call Summary for Register TPCC_IER

IVA2.2 Subsystem Register Manual

- TPCC Register Mapping Summary: [0]

Table 14-368. TPCC_IERH

Address Offset	0x1054				
Physical address	0x01C0 1054	**Instance**	IVA2.2 TPCC		
Description	Int Enable Register (High Part): IERH.In is not directly writeable. Interrupts canbe enabled through writes to IESRH and can be disabled through writes to IECRH register. IERH.In = 0: IPRH.In is NOT enabled for interrupts. IERH.In = 1: IPRH.In IS enabled for interrupts.				
Type	R				

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
I63	I62	I61	I60	I59	I58	I57	I56	I55	I54	I53	I52	I51	I50	I49	I48	I47	I46	I45	I44	I43	I42	I41	I40	I39	I38	I37	I36	I35	I34	I33	I32

Bits	Field Name	Description	Type	Reset
31	I63	Interrupt associated with TCC #63	R	0
30	I62	Interrupt associated with TCC #62	R	0
29	I61	Interrupt associated with TCC #61	R	0
28	I60	Interrupt associated with TCC #60	R	0
27	I59	Interrupt associated with TCC #59	R	0
26	I58	Interrupt associated with TCC #58	R	0
25	I57	Interrupt associated with TCC #57	R	0
24	I56	Interrupt associated with TCC #56	R	0
23	I55	Interrupt associated with TCC #55	R	0
22	I54	Interrupt associated with TCC #54	R	0
21	I53	Interrupt associated with TCC #53	R	0
20	I52	Interrupt associated with TCC #52	R	0
19	I51	Interrupt associated with TCC #51	R	0
18	I50	Interrupt associated with TCC #50	R	0
17	I49	Interrupt associated with TCC #49	R	0
16	I48	Interrupt associated with TCC #48	R	0
15	I47	Interrupt associated with TCC #47	R	0
14	I46	Interrupt associated with TCC #46	R	0
13	I45	Interrupt associated with TCC #45	R	0
12	I44	Interrupt associated with TCC #44	R	0
11	I43	Interrupt associated with TCC #43	R	0
10	I42	Interrupt associated with TCC #42	R	0
9	I41	Interrupt associated with TCC #41	R	0
8	I40	Interrupt associated with TCC #40	R	0
7	I39	Interrupt associated with TCC #39	R	0
6	I38	Interrupt associated with TCC #38	R	0
5	I37	Interrupt associated with TCC #37	R	0
4	I36	Interrupt associated with TCC #36	R	0
3	I35	Interrupt associated with TCC #35	R	0
2	I34	Interrupt associated with TCC #34	R	0
1	I33	Interrupt associated with TCC #33	R	0
0	I32	Interrupt associated with TCC #32	R	0

Table 14-369. Register Call Summary for Register TPCC_IERH

IVA2.2 Subsystem Register Manual

- TPCC Register Mapping Summary: [0]

Table 14-370. TPCC_IECR

Address Offset	0x1058		
Physical address	0x01C0 1058	**Instance**	IVA2.2 TPCC
Description	Int Enable Clear Register: CPU write of 1 to the IECR.In bit causes the IER.In bit to be cleared. CPU write of 0 has no effect.		
Type	W		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
I31	I30	I29	I28	I27	I26	I25	I24	I23	I22	I21	I20	I19	I18	I17	I16	I15	I14	I13	I12	I11	I10	I9	I8	I7	I6	I5	I4	I3	I2	I1	I0

Bits	Field Name	Description	Type	Reset
31	I31	Interrupt associated with TCC #31	W	0
30	I30	Interrupt associated with TCC #30	W	0
29	I29	Interrupt associated with TCC #29	W	0
28	I28	Interrupt associated with TCC #28	W	0
27	I27	Interrupt associated with TCC #27	W	0
26	I26	Interrupt associated with TCC #26	W	0
25	I25	Interrupt associated with TCC #25	W	0
24	I24	Interrupt associated with TCC #24	W	0
23	I23	Interrupt associated with TCC #23	W	0
22	I22	Interrupt associated with TCC #22	W	0
21	I21	Interrupt associated with TCC #21	W	0
20	I20	Interrupt associated with TCC #20	W	0
19	I19	Interrupt associated with TCC #19	W	0
18	I18	Interrupt associated with TCC #18	W	0
17	I17	Interrupt associated with TCC #17	W	0
16	I16	Interrupt associated with TCC #16	W	0
15	I15	Interrupt associated with TCC #15	W	0
14	I14	Interrupt associated with TCC #14	W	0
13	I13	Interrupt associated with TCC #13	W	0
12	I12	Interrupt associated with TCC #12	W	0
11	I11	Interrupt associated with TCC #11	W	0
10	I10	Interrupt associated with TCC #10	W	0
9	I9	Interrupt associated with TCC #9	W	0
8	I8	Interrupt associated with TCC #8	W	0
7	I7	Interrupt associated with TCC #7	W	0
6	I6	Interrupt associated with TCC #6	W	0
5	I5	Interrupt associated with TCC #5	W	0
4	I4	Interrupt associated with TCC #4	W	0
3	I3	Interrupt associated with TCC #3	W	0
2	I2	Interrupt associated with TCC #2	W	0
1	I1	Interrupt associated with TCC #1	W	0
0	I0	Interrupt associated with TCC #0	W	0

Table 14-371. Register Call Summary for Register TPCC_IECR

IVA2.2 Subsystem Register Manual

- TPCC Register Mapping Summary: [0]

Table 14-372. TPCC_IECRH

Address Offset	0x105C		
Physical address	0x01C0 105C	**Instance**	IVA2.2 TPCC
Description	Int Enable Clear Register (High Part): CPU write of 1 to the IECRH.In bit causes the IERH.In bit to be cleared. CPU write of 0 has no effect.		
Type	W		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
I63	I62	I61	I60	I59	I58	I57	I56	I55	I54	I53	I52	I51	I50	I49	I48	I47	I46	I45	I44	I43	I42	I41	I40	I39	I38	I37	I36	I35	I34	I33	I32

Bits	Field Name	Description	Type	Reset
31	I63	Interrupt associated with TCC #63	W	0
30	I62	Interrupt associated with TCC #62	W	0
29	I61	Interrupt associated with TCC #61	W	0
28	I60	Interrupt associated with TCC #60	W	0
27	I59	Interrupt associated with TCC #59	W	0
26	I58	Interrupt associated with TCC #58	W	0
25	I57	Interrupt associated with TCC #57	W	0
24	I56	Interrupt associated with TCC #56	W	0
23	I55	Interrupt associated with TCC #55	W	0
22	I54	Interrupt associated with TCC #54	W	0
21	I53	Interrupt associated with TCC #53	W	0
20	I52	Interrupt associated with TCC #52	W	0
19	I51	Interrupt associated with TCC #51	W	0
18	I50	Interrupt associated with TCC #50	W	0
17	I49	Interrupt associated with TCC #49	W	0
16	I48	Interrupt associated with TCC #48	W	0
15	I47	Interrupt associated with TCC #47	W	0
14	I46	Interrupt associated with TCC #46	W	0
13	I45	Interrupt associated with TCC #45	W	0
12	I44	Interrupt associated with TCC #44	W	0
11	I43	Interrupt associated with TCC #43	W	0
10	I42	Interrupt associated with TCC #42	W	0
9	I41	Interrupt associated with TCC #41	W	0
8	I40	Interrupt associated with TCC #40	W	0
7	I39	Interrupt associated with TCC #39	W	0
6	I38	Interrupt associated with TCC #38	W	0
5	I37	Interrupt associated with TCC #37	W	0
4	I36	Interrupt associated with TCC #36	W	0
3	I35	Interrupt associated with TCC #35	W	0
2	I34	Interrupt associated with TCC #34	W	0
1	I33	Interrupt associated with TCC #33	W	0
0	I32	Interrupt associated with TCC #32	W	0

Table 14-373. Register Call Summary for Register TPCC_IECRH

IVA2.2 Subsystem Register Manual

- TPCC Register Mapping Summary: [0]

Table 14-374. TPCC_IESR

Address Offset	0x1060		
Physical address	0x01C0 1060	**Instance**	IVA2.2 TPCC
Description	Int Enable Set Register: CPU write of 1 to the IESR.In bit causes the IESR.In bit to be set. CPU write of 0 has no effect.		
Type	W		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
I31	I30	I29	I28	I27	I26	I25	I24	I23	I22	I21	I20	I19	I18	I17	I16	I15	I14	I13	I12	I11	I10	I9	I8	I7	I6	I5	I4	I3	I2	I1	I0

Bits	Field Name	Description	Type	Reset
31	I31	Interrupt associated with TCC #31	W	0
30	I30	Interrupt associated with TCC #30	W	0
29	I29	Interrupt associated with TCC #29	W	0
28	I28	Interrupt associated with TCC #28	W	0
27	I27	Interrupt associated with TCC #27	W	0
26	I26	Interrupt associated with TCC #26	W	0
25	I25	Interrupt associated with TCC #25	W	0
24	I24	Interrupt associated with TCC #24	W	0
23	I23	Interrupt associated with TCC #23	W	0
22	I22	Interrupt associated with TCC #22	W	0
21	I21	Interrupt associated with TCC #21	W	0
20	I20	Interrupt associated with TCC #20	W	0
19	I19	Interrupt associated with TCC #19	W	0
18	I18	Interrupt associated with TCC #18	W	0
17	I17	Interrupt associated with TCC #17	W	0
16	I16	Interrupt associated with TCC #16	W	0
15	I15	Interrupt associated with TCC #15	W	0
14	I14	Interrupt associated with TCC #14	W	0
13	I13	Interrupt associated with TCC #13	W	0
12	I12	Interrupt associated with TCC #12	W	0
11	I11	Interrupt associated with TCC #11	W	0
10	I10	Interrupt associated with TCC #10	W	0
9	I9	Interrupt associated with TCC #9	W	0
8	I8	Interrupt associated with TCC #8	W	0
7	I7	Interrupt associated with TCC #7	W	0
6	I6	Interrupt associated with TCC #6	W	0
5	I5	Interrupt associated with TCC #5	W	0
4	I4	Interrupt associated with TCC #4	W	0
3	I3	Interrupt associated with TCC #3	W	0
2	I2	Interrupt associated with TCC #2	W	0
1	I1	Interrupt associated with TCC #1	W	0
0	I0	Interrupt associated with TCC #0	W	0

Table 14-375. Register Call Summary for Register TPCC_IESR

IVA2.2 Subsystem Register Manual

- TPCC Register Mapping Summary: [0]

Table 14-376. TPCC_IESRH

Address Offset	0x1064				
Physical address	0x01C0 1064	Instance	IVA2.2 TPCC		
Description	Int Enable Set Register (High Part): CPU write of 1 to the IESRH.In bit causes the IESRH.In bit to be set. CPU write of 0 has no effect.				
Type	W				

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
I63	I62	I61	I60	I59	I58	I57	I56	I55	I54	I53	I52	I51	I50	I49	I48	I47	I46	I45	I44	I43	I42	I41	I40	I39	I38	I37	I36	I35	I34	I33	I32

Bits	Field Name	Description	Type	Reset
31	I63	Interrupt associated with TCC #63	W	0
30	I62	Interrupt associated with TCC #62	W	0
29	I61	Interrupt associated with TCC #61	W	0
28	I60	Interrupt associated with TCC #60	W	0
27	I59	Interrupt associated with TCC #59	W	0
26	I58	Interrupt associated with TCC #58	W	0
25	I57	Interrupt associated with TCC #57	W	0
24	I56	Interrupt associated with TCC #56	W	0
23	I55	Interrupt associated with TCC #55	W	0
22	I54	Interrupt associated with TCC #54	W	0
21	I53	Interrupt associated with TCC #53	W	0
20	I52	Interrupt associated with TCC #52	W	0
19	I51	Interrupt associated with TCC #51	W	0
18	I50	Interrupt associated with TCC #50	W	0
17	I49	Interrupt associated with TCC #49	W	0
16	I48	Interrupt associated with TCC #48	W	0
15	I47	Interrupt associated with TCC #47	W	0
14	I46	Interrupt associated with TCC #46	W	0
13	I45	Interrupt associated with TCC #45	W	0
12	I44	Interrupt associated with TCC #44	W	0
11	I43	Interrupt associated with TCC #43	W	0
10	I42	Interrupt associated with TCC #42	W	0
9	I41	Interrupt associated with TCC #41	W	0
8	I40	Interrupt associated with TCC #40	W	0
7	I39	Interrupt associated with TCC #39	W	0
6	I38	Interrupt associated with TCC #38	W	0
5	I37	Interrupt associated with TCC #37	W	0
4	I36	Interrupt associated with TCC #36	W	0
3	I35	Interrupt associated with TCC #35	W	0
2	I34	Interrupt associated with TCC #34	W	0
1	I33	Interrupt associated with TCC #33	W	0
0	I32	Interrupt associated with TCC #32	W	0

Table 14-377. Register Call Summary for Register TPCC_IESRH

IVA2.2 Subsystem Register Manual

- TPCC Register Mapping Summary: [0]

Table 14-378. TPCC_IPR

Address Offset	0x1068		
Physical address	0x01C0 1068	**Instance**	IVA2.2 TPCC
Description	Interrupt Pending Register: IPR.In bit is set when a interrupt completion code with TCC of N is detected. IPR.In bit is cleared through software by writing 1 to ICR.In bit.		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
I31	I30	I29	I28	I27	I26	I25	I24	I23	I22	I21	I20	I19	I18	I17	I16	I15	I14	I13	I12	I11	I10	I9	I8	I7	I6	I5	I4	I3	I2	I1	I0

Bits	Field Name	Description	Type	Reset
31	I31	Interrupt associated with TCC #31	R	0
30	I30	Interrupt associated with TCC #30	R	0
29	I29	Interrupt associated with TCC #29	R	0
28	I28	Interrupt associated with TCC #28	R	0
27	I27	Interrupt associated with TCC #27	R	0
26	I26	Interrupt associated with TCC #26	R	0
25	I25	Interrupt associated with TCC #25	R	0
24	I24	Interrupt associated with TCC #24	R	0
23	I23	Interrupt associated with TCC #23	R	0
22	I22	Interrupt associated with TCC #22	R	0
21	I21	Interrupt associated with TCC #21	R	0
20	I20	Interrupt associated with TCC #20	R	0
19	I19	Interrupt associated with TCC #19	R	0
18	I18	Interrupt associated with TCC #18	R	0
17	I17	Interrupt associated with TCC #17	R	0
16	I16	Interrupt associated with TCC #16	R	0
15	I15	Interrupt associated with TCC #15	R	0
14	I14	Interrupt associated with TCC #14	R	0
13	I13	Interrupt associated with TCC #13	R	0
12	I12	Interrupt associated with TCC #12	R	0
11	I11	Interrupt associated with TCC #11	R	0
10	I10	Interrupt associated with TCC #10	R	0
9	I9	Interrupt associated with TCC #9	R	0
8	I8	Interrupt associated with TCC #8	R	0
7	I7	Interrupt associated with TCC #7	R	0
6	I6	Interrupt associated with TCC #6	R	0
5	I5	Interrupt associated with TCC #5	R	0
4	I4	Interrupt associated with TCC #4	R	0
3	I3	Interrupt associated with TCC #3	R	0
2	I2	Interrupt associated with TCC #2	R	0
1	I1	Interrupt associated with TCC #1	R	0
0	I0	Interrupt associated with TCC #0	R	0

Table 14-379. Register Call Summary for Register TPCC_IPR

IVA2.2 Subsystem Basic Programming Model

- Starting the Transfer: [0] [1] [2] [3] [4] [5] [6] [7]

IVA2.2 Subsystem Register Manual

- TPCC Register Mapping Summary: [8]

Table 14-380. TPCC_IPRH

Address Offset	0x106C				
Physical address	0x01C0 106C	**Instance**	IVA2.2 TPCC		
Description	Interrupt Pending Register (High Part): IPRH.In bit is set when a interrupt completion code with TCC of N is detected. IPRH.In bit is cleared through software by writing 1 to ICRH.In bit.				
Type	R				

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
I63	I62	I61	I60	I59	I58	I57	I56	I55	I54	I53	I52	I51	I50	I49	I48	I47	I46	I45	I44	I43	I42	I41	I40	I39	I38	I37	I36	I35	I34	I33	I32

Bits	Field Name	Description	Type	Reset
31	I63	Interrupt associated with TCC #63	R	0
30	I62	Interrupt associated with TCC #62	R	0
29	I61	Interrupt associated with TCC #61	R	0
28	I60	Interrupt associated with TCC #60	R	0
27	I59	Interrupt associated with TCC #59	R	0
26	I58	Interrupt associated with TCC #58	R	0
25	I57	Interrupt associated with TCC #57	R	0
24	I56	Interrupt associated with TCC #56	R	0
23	I55	Interrupt associated with TCC #55	R	0
22	I54	Interrupt associated with TCC #54	R	0
21	I53	Interrupt associated with TCC #53	R	0
20	I52	Interrupt associated with TCC #52	R	0
19	I51	Interrupt associated with TCC #51	R	0
18	I50	Interrupt associated with TCC #50	R	0
17	I49	Interrupt associated with TCC #49	R	0
16	I48	Interrupt associated with TCC #48	R	0
15	I47	Interrupt associated with TCC #47	R	0
14	I46	Interrupt associated with TCC #46	R	0
13	I45	Interrupt associated with TCC #45	R	0
12	I44	Interrupt associated with TCC #44	R	0
11	I43	Interrupt associated with TCC #43	R	0
10	I42	Interrupt associated with TCC #42	R	0
9	I41	Interrupt associated with TCC #41	R	0
8	I40	Interrupt associated with TCC #40	R	0
7	I39	Interrupt associated with TCC #39	R	0
6	I38	Interrupt associated with TCC #38	R	0
5	I37	Interrupt associated with TCC #37	R	0
4	I36	Interrupt associated with TCC #36	R	0
3	I35	Interrupt associated with TCC #35	R	0
2	I34	Interrupt associated with TCC #34	R	0
1	I33	Interrupt associated with TCC #33	R	0
0	I32	Interrupt associated with TCC #32	R	0

Table 14-381. Register Call Summary for Register TPCC_IPRH

IVA2.2 Subsystem Register Manual

- TPCC Register Mapping Summary: [0]

Table 14-382. TPCC_ICR

Address Offset	0x1070		
Physical address	0x01C0 1070	Instance	IVA2.2 TPCC
Description	Interrupt Clear Register: CPU write of 1 to the ICR.In bit causes the IPR.In bit to be cleared. CPU write of 0 has no effect. All IPR.In bits must be cleared before additional interrupts will be asserted by CC.		
Type	W		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
I31	I30	I29	I28	I27	I26	I25	I24	I23	I22	I21	I20	I19	I18	I17	I16	I15	I14	I13	I12	I11	I10	I9	I8	I7	I6	I5	I4	I3	I2	I1	I0

Bits	Field Name	Description	Type	Reset
31	I31	Interrupt associated with TCC #31	W	0
30	I30	Interrupt associated with TCC #30	W	0
29	I29	Interrupt associated with TCC #29	W	0
28	I28	Interrupt associated with TCC #28	W	0
27	I27	Interrupt associated with TCC #27	W	0
26	I26	Interrupt associated with TCC #26	W	0
25	I25	Interrupt associated with TCC #25	W	0
24	I24	Interrupt associated with TCC #24	W	0
23	I23	Interrupt associated with TCC #23	W	0
22	I22	Interrupt associated with TCC #22	W	0
21	I21	Interrupt associated with TCC #21	W	0
20	I20	Interrupt associated with TCC #20	W	0
19	I19	Interrupt associated with TCC #19	W	0
18	I18	Interrupt associated with TCC #18	W	0
17	I17	Interrupt associated with TCC #17	W	0
16	I16	Interrupt associated with TCC #16	W	0
15	I15	Interrupt associated with TCC #15	W	0
14	I14	Interrupt associated with TCC #14	W	0
13	I13	Interrupt associated with TCC #13	W	0
12	I12	Interrupt associated with TCC #12	W	0
11	I11	Interrupt associated with TCC #11	W	0
10	I10	Interrupt associated with TCC #10	W	0
9	I9	Interrupt associated with TCC #9	W	0
8	I8	Interrupt associated with TCC #8	W	0
7	I7	Interrupt associated with TCC #7	W	0
6	I6	Interrupt associated with TCC #6	W	0
5	I5	Interrupt associated with TCC #5	W	0
4	I4	Interrupt associated with TCC #4	W	0
3	I3	Interrupt associated with TCC #3	W	0
2	I2	Interrupt associated with TCC #2	W	0
1	I1	Interrupt associated with TCC #1	W	0
0	I0	Interrupt associated with TCC #0	W	0

Table 14-383. Register Call Summary for Register TPCC_ICR

IVA2.2 Subsystem Register Manual

- TPCC Register Mapping Summary: [0]

Table 14-384. TPCC_ICRH

Address Offset	0x1074		
Physical address	0x01C0 1074	**Instance**	IVA2.2 TPCC
Description	Interrupt Clear Register (High Part): CPU write of 1 to the ICRH.In bit causes the IPRH.In bit to be cleared. CPU write of 0 has no effect. All IPRH.In bits must be cleared before additional interrupts will be asserted by CC.		
Type	W		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
I63	I62	I61	I60	I59	I58	I57	I56	I55	I54	I53	I52	I51	I50	I49	I48	I47	I46	I45	I44	I43	I42	I41	I40	I39	I38	I37	I36	I35	I34	I33	I32

Bits	Field Name	Description	Type	Reset
31	I63	Interrupt associated with TCC #63	W	0
30	I62	Interrupt associated with TCC #62	W	0
29	I61	Interrupt associated with TCC #61	W	0
28	I60	Interrupt associated with TCC #60	W	0
27	I59	Interrupt associated with TCC #59	W	0
26	I58	Interrupt associated with TCC #58	W	0
25	I57	Interrupt associated with TCC #57	W	0
24	I56	Interrupt associated with TCC #56	W	0
23	I55	Interrupt associated with TCC #55	W	0
22	I54	Interrupt associated with TCC #54	W	0
21	I53	Interrupt associated with TCC #53	W	0
20	I52	Interrupt associated with TCC #52	W	0
19	I51	Interrupt associated with TCC #51	W	0
18	I50	Interrupt associated with TCC #50	W	0
17	I49	Interrupt associated with TCC #49	W	0
16	I48	Interrupt associated with TCC #48	W	0
15	I47	Interrupt associated with TCC #47	W	0
14	I46	Interrupt associated with TCC #46	W	0
13	I45	Interrupt associated with TCC #45	W	0
12	I44	Interrupt associated with TCC #44	W	0
11	I43	Interrupt associated with TCC #43	W	0
10	I42	Interrupt associated with TCC #42	W	0
9	I41	Interrupt associated with TCC #41	W	0
8	I40	Interrupt associated with TCC #40	W	0
7	I39	Interrupt associated with TCC #39	W	0
6	I38	Interrupt associated with TCC #38	W	0
5	I37	Interrupt associated with TCC #37	W	0
4	I36	Interrupt associated with TCC #36	W	0
3	I35	Interrupt associated with TCC #35	W	0
2	I34	Interrupt associated with TCC #34	W	0
1	I33	Interrupt associated with TCC #33	W	0
0	I32	Interrupt associated with TCC #32	W	0

Table 14-385. Register Call Summary for Register TPCC_ICRH

IVA2.2 Subsystem Register Manual

- TPCC Register Mapping Summary: [0]

Table 14-386. TPCC_IEVAL

Address Offset	0x1078		
Physical address	0x01C0 1078	**Instance**	IVA2.2 TPCC
Description	Interrupt Eval Register		
Type	W		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																														SET	EVAL

Bits	Field Name	Description	Type	Reset
31:2	Reserved	Write 0s for future compatibility.	W	0x00000000
1	SET	Interrupt Set: CPU write of 1 to the SETn bit causes the tpcc_intN output signal to be pulsed egardless of state of interrupts enable (IERn) and status (IPRn). CPU write of 0 has no effect.	W	0
0	EVAL	Interrupt Evaluate: CPU write of 1 to the EVALn bit causes the tpcc_intN output signal to be pulsed if any enabled interrupts (IERn) are still pending (IPRn). CPU write of 0 has no effect.	W	0

Table 14-387. Register Call Summary for Register TPCC_IEVAL

IVA2.2 Subsystem Basic Programming Model

- Starting the Transfer: [0]

IVA2.2 Subsystem Register Manual

- TPCC Register Mapping Summary: [1]

Table 14-388. TPCC_QER

Address Offset	0x1080		
Physical address	0x01C0 1080	**Instance**	IVA2.2 TPCC
Description	QDMA Event Register: If QER.En bit is set, then the corresponding QDMA channel is prioritized vs. other qdma events for submission to the TC. QER.En bit is set when a vbus write byte matches the address defined in the QCHMAPn register. QER.En bit is cleared when the corresponding event is prioritized and serviced. QER.En is also cleared when user writes a 1 to the QSECR.En bit. If the QER.En bit is already set and a new QDMA event is detected due to user write to QDMA trigger location and QEER register is set, then the corresponding bit in the QDMA Event Missed Register is set.		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																								E7	E6	E5	E4	E3	E2	E1	E0

Bits	Field Name	Description	Type	Reset
31:8	Reserved	Read returns 0.	R	0x000000
7	E7	Event #7	R	0
6	E6	Event #6	R	0
5	E5	Event #5	R	0
4	E4	Event #4	R	0

Bits	Field Name	Description	Type	Reset
3	E3	Event #3	R	0
2	E2	Event #2	R	0
1	E1	Event #1	R	0
0	E0	Event #0	R	0

Table 14-389. Register Call Summary for Register TPCC_QER

IVA2.2 Subsystem Functional Description
- EDMA: [0] [1] [2] [3] [4]

IVA2.2 Subsystem Register Manual
- TPCC Register Mapping Summary: [5]

Table 14-390. TPCC_QEER

Address Offset	0x1084		
Physical address	0x01C0 1084	Instance	IVA2.2 TPCC
Description	QDMA Event Enable Register: Enabled/disabled QDMA address comparator for QDMA Channel N. QEER.En is not directly writeable. QDMA channels can be enabled through writes to QEESR and can be disabled through writes to QEECR register. QEER.En = 1, The corresponding QDMA channel comparator is enabled and Events will be recognized and latched in QER.En. QEER.En = 0, The corresponding QDMA channel comparator is disabled. Events will not be recognized/latched in QER.En.		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																								E7	E6	E5	E4	E3	E2	E1	E0

Bits	Field Name	Description	Type	Reset
31:8	Reserved	Read returns 0.	R	0x000000
7	E7	Event #7	R	0
6	E6	Event #6	R	0
5	E5	Event #5	R	0
4	E4	Event #4	R	0
3	E3	Event #3	R	0
2	E2	Event #2	R	0
1	E1	Event #1	R	0
0	E0	Event #0	R	0

Table 14-391. Register Call Summary for Register TPCC_QEER

IVA2.2 Subsystem Functional Description
- EDMA: [0] [1] [2]

IVA2.2 Subsystem Register Manual
- TPCC Register Mapping Summary: [3]

Table 14-392. TPCC_QEECR

Address Offset	0x1088		
Physical address	0x01C0 1088	Instance	IVA2.2 TPCC
Description	QDMA Event Enable Clear Register: CPU write of 1 to the QEECR.En bit causes the QEER.En bit to be cleared. CPU write of 0 has no effect.		
Type	W		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																								E7	E6	E5	E4	E3	E2	E1	E0

Bits	Field Name	Description	Type	Reset
31:8	Reserved	Write 0s for future compatibility.	W	0x000000
7	E7	Event #7	W	0
6	E6	Event #6	W	0
5	E5	Event #5	W	0
4	E4	Event #4	W	0
3	E3	Event #3	W	0
2	E2	Event #2	W	0
1	E1	Event #1	W	0
0	E0	Event #0	W	0

Table 14-393. Register Call Summary for Register TPCC_QEECR

IVA2.2 Subsystem Register Manual

- TPCC Register Mapping Summary: [0]

Table 14-394. TPCC_QEESR

Address Offset	0x108C		
Physical address	0x01C0 108C	**Instance**	IVA2.2 TPCC
Description	QDMA Event Enable Set Register: CPU write of 1 to the QEESR.En bit causes the QEESR.En bit to be set. CPU write of 0 has no effect.		
Type	W		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																								E7	E6	E5	E4	E3	E2	E1	E0

Bits	Field Name	Description	Type	Reset
31:8	Reserved	Write 0s for future compatibility.	W	0x000000
7	E7	Event #7	W	0
6	E6	Event #6	W	0
5	E5	Event #5	W	0
4	E4	Event #4	W	0
3	E3	Event #3	W	0
2	E2	Event #2	W	0
1	E1	Event #1	W	0
0	E0	Event #0	W	0

Table 14-395. Register Call Summary for Register TPCC_QEESR

IVA2.2 Subsystem Register Manual

- TPCC Register Mapping Summary: [0]

Table 14-396. TPCC_QSER

Address Offset	0x1090		
Physical address	0x01C0 1090	**Instance**	IVA2.2 TPCC
Description	QDMA Secondary Event Register: The QDMA secondary event register is used along with the QDMA Event Register (QER) to provide information on the state of a QDMA Event. En = 0: Event is not currently in the Event Queue. En = 1: Event is currently stored in Event Queue. Event arbiter will not prioritize additional events.		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																								E7	E6	E5	E4	E3	E2	E1	E0

Bits	Field Name	Description	Type	Reset
31:8	Reserved	Write 0s for future compatibility.	W	0x000000
7	E7	Event #7	W	0
6	E6	Event #6	W	0
5	E5	Event #5	W	0
4	E4	Event #4	W	0
3	E3	Event #3	W	0
2	E2	Event #2	W	0
1	E1	Event #1	W	0
0	E0	Event #0	W	0

Table 14-397. Register Call Summary for Register TPCC_QSER

IVA2.2 Subsystem Register Manual

- TPCC Register Mapping Summary: [0]

Table 14-398. TPCC_QSECR

Address Offset	0x1094		
Physical address	0x01C0 1094	**Instance**	IVA2.2 TPCC
Description	QDMA Secondary Event Clear Register: The secondary event clear register is used to clear the status of the QSER and QER register (note that this is slightly different than the SER operation, which does not clear the ER.En register). CPU write of 1 to the QSECR.En bit clears the QSER.En and QER.En register fields. CPU write of 0 has no effect.		
Type	W		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																								E7	E6	E5	E4	E3	E2	E1	E0

Bits	Field Name	Description	Type	Reset
31:8	Reserved	Write 0s for future compatibility.	W	0x000000
7	E7	Event #7	W	0
6	E6	Event #6	W	0
5	E5	Event #5	W	0
4	E4	Event #4	W	0
3	E3	Event #3	W	0

Bits	Field Name	Description	Type	Reset
2	E2	Event #2	W	0
1	E1	Event #1	W	0
0	E0	Event #0	W	0

Table 14-399. Register Call Summary for Register TPCC_QSECR

IVA2.2 Subsystem Register Manual

- TPCC Register Mapping Summary: [0]

Table 14-400. TPCC_ER_Rn

Address Offset	0x2000 + (0x200*n) n = 0 to 7		
Physical address	0x01C0 2000 + (0x200*n) n = 0 to 7	**Instance**	IVA2.2 TPCC
Description	Event Register: If ER.En bit is set and the EER.En bit is also set, then the corresponding DMA channel is prioritized vs. other pending DMA events for submission to the TC. ER.En bit is set when the input event #n transitions from inactive (low) to active (high), regardless of the state of EER.En bit. ER.En bit is cleared when the corresponding event is prioritized and serviced. If the ER.En bit is already set and a new inactive to active transition is detected on the input event #n input AND the corresponding bit in the EER register is set, then the corresponding bit in the Event Missed Register is set. Event N can be cleared through sw by writing 1 to the ECR pseudo-register.		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved												E19	E18	E17	E16	E15	E14	E13	E12	E11	E10	E9	E8	E7	E6	E5	E4	E3	E2	E1	E0

Bits	Field Name	Description	Type	Reset
31:20	Reserved	Reserved	R	0
19	E19	Event #19	R	0
18	E18	Event #18	R	0
17	E17	Event #17	R	0
16	E16	Event #16	R	0
15	E15	Event #15	R	0
14	E14	Event #14	R	0
13	E13	Event #13	R	0
12	E12	Event #12	R	0
11	E11	Event #11	R	0
10	E10	Event #10	R	0
9	E9	Event #9	R	0
8	E8	Event #8	R	0
7	E7	Event #7	R	0
6	E6	Event #6	R	0
5	E5	Event #5	R	0
4	E4	Event #4	R	0
3	E3	Event #3	R	0
2	E2	Event #2	R	0
1	E1	Event #1	R	0
0	E0	Event #0	R	0

Table 14-401. Register Call Summary for Register TPCC_ER_Rn

IVA2.2 Subsystem Register Manual

- TPCC Register Mapping Summary: [0]

Table 14-402. TPCC_ECR_Rn

Address Offset	0x2008 + (0x200*n) n = 0 to 7		
Physical address	0x01C0 2008 + (0x200*n) n = 0 to 7	**Instance**	IVA2.2 TPCC
Description	Event Clear Register: CPU write of 1 to the ECR.En bit causes the ER.En bit to be cleared. CPU write of 0 has no effect.		
Type	W		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
E31	E30	E29	E28	E27	E26	E25	E24	E23	E22	E21	E20	E19	E18	E17	E16	E15	E14	E13	E12	E11	E10	E9	E8	E7	E6	E5	E4	E3	E2	E1	E0

Bits	Field Name	Description	Type	Reset
31	E31	Event #31	W	0
30	E30	Event #30	W	0
29	E29	Event #29	W	0
28	E28	Event #28	W	0
27	E27	Event #27	W	0
26	E26	Event #26	W	0
25	E25	Event #25	W	0
24	E24	Event #24	W	0
23	E23	Event #23	W	0
22	E22	Event #22	W	0
21	E21	Event #21	W	0
20	E20	Event #20	W	0
19	E19	Event #19	W	0
18	E18	Event #18	W	0
17	E17	Event #17	W	0
16	E16	Event #16	W	0
15	E15	Event #15	W	0
14	E14	Event #14	W	0
13	E13	Event #13	W	0
12	E12	Event #12	W	0
11	E11	Event #11	W	0
10	E10	Event #10	W	0
9	E9	Event #9	W	0
8	E8	Event #8	W	0
7	E7	Event #7	W	0
6	E6	Event #6	W	0
5	E5	Event #5	W	0
4	E4	Event #4	W	0
3	E3	Event #3	W	0
2	E2	Event #2	W	0
1	E1	Event #1	W	0
0	E0	Event #0	W	0

Table 14-403. Register Call Summary for Register TPCC_ECR_Rn

IVA2.2 Subsystem Register Manual

- TPCC Register Mapping Summary: [0]

Table 14-404. TPCC_ECRH_Rn

Address Offset	0x200C + (0x200*n) n = 0 to 7		
Physical address	0x01C0 200C + (0x200*n) n = 0 to 7	**Instance**	IVA2.2 TPCC
Description	Event Clear Register (High Part): CPU write of 1 to the ECRH.En bit causes the ERH.En bit to be cleared. CPU write of 0 has no effect.		
Type	W		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
E63	E62	E61	E60	E59	E58	E57	E56	E55	E54	E53	E52	E51	E50	E49	E48	E47	E46	E45	E44	E43	E42	E41	E40	E39	E38	E37	E36	E35	E34	E33	E32

Bits	Field Name	Description	Type	Reset
31	E63	Event #63	W	0
30	E62	Event #62	W	0
29	E61	Event #61	W	0
28	E60	Event #60	W	0
27	E59	Event #59	W	0
26	E58	Event #58	W	0
25	E57	Event #57	W	0
24	E56	Event #56	W	0
23	E55	Event #55	W	0
22	E54	Event #54	W	0
21	E53	Event #53	W	0
20	E52	Event #52	W	0
19	E51	Event #51	W	0
18	E50	Event #50	W	0
17	E49	Event #49	W	0
16	E48	Event #48	W	0
15	E47	Event #47	W	0
14	E46	Event #46	W	0
13	E45	Event #45	W	0
12	E44	Event #44	W	0
11	E43	Event #43	W	0
10	E42	Event #42	W	0
9	E41	Event #41	W	0
8	E40	Event #40	W	0
7	E39	Event #39	W	0
6	E38	Event #38	W	0
5	E37	Event #37	W	0
4	E36	Event #36	W	0
3	E35	Event #35	W	0
2	E34	Event #34	W	0
1	E33	Event #33	W	0
0	E32	Event #32	W	0

Table 14-405. Register Call Summary for Register TPCC_ECRH_Rn

IVA2.2 Subsystem Register Manual

- TPCC Register Mapping Summary: [0]

Table 14-406. TPCC_ESR_Rn

Address Offset	0x2010 + (0x200*n) n = 0 to 7		
Physical address	0x01C0 2010 + (0x200*n) n = 0 to 7	**Instance**	IVA2.2 TPCC
Description	Event Set Register: CPU write of 1 to the ESR.En bit causes the ER.En bit to be set. CPU write of 0 has no effect.		
Type	W		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
E31	E30	E29	E28	E27	E26	E25	E24	E23	E22	E21	E20	E19	E18	E17	E16	E15	E14	E13	E12	E11	E10	E9	E8	E7	E6	E5	E4	E3	E2	E1	E0

Bits	Field Name	Description	Type	Reset
31	E31	Event #31	W	0
30	E30	Event #30	W	0
29	E29	Event #29	W	0
28	E28	Event #28	W	0
27	E27	Event #27	W	0
26	E26	Event #26	W	0
25	E25	Event #25	W	0
24	E24	Event #24	W	0
23	E23	Event #23	W	0
22	E22	Event #22	W	0
21	E21	Event #21	W	0
20	E20	Event #20	W	0
19	E19	Event #19	W	0
18	E18	Event #18	W	0
17	E17	Event #17	W	0
16	E16	Event #16	W	0
15	E15	Event #15	W	0
14	E14	Event #14	W	0
13	E13	Event #13	W	0
12	E12	Event #12	W	0
11	E11	Event #11	W	0
10	E10	Event #10	W	0
9	E9	Event #9	W	0
8	E8	Event #8	W	0
7	E7	Event #7	W	0
6	E6	Event #6	W	0
5	E5	Event #5	W	0
4	E4	Event #4	W	0
3	E3	Event #3	W	0
2	E2	Event #2	W	0
1	E1	Event #1	W	0
0	E0	Event #0	W	0

Table 14-407. Register Call Summary for Register TPCC_ESR_Rn

IVA2.2 Subsystem Register Manual

- TPCC Register Mapping Summary: [0]

Table 14-408. TPCC_ESRH_Rn

Address Offset	0x2014 + (0x200*n) n = 0 to 7		
Physical address	0x01C0 2014 + (0x200*n) n = 0 to 7	**Instance**	IVA2.2 TPCC
Description	Event Set Register (High Part) CPU write of 1 to the ESRH.En bit causes the ERH.En bit to be set. CPU write of 0 has no effect.		
Type	W		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
E63	E62	E61	E60	E59	E58	E57	E56	E55	E54	E53	E52	E51	E50	E49	E48	E47	E46	E45	E44	E43	E42	E41	E40	E39	E38	E37	E36	E35	E34	E33	E32

Bits	Field Name	Description	Type	Reset
31	E63	Event #63	W	0
30	E62	Event #62	W	0
29	E61	Event #61	W	0
28	E60	Event #60	W	0
27	E59	Event #59	W	0
26	E58	Event #58	W	0
25	E57	Event #57	W	0
24	E56	Event #56	W	0
23	E55	Event #55	W	0
22	E54	Event #54	W	0
21	E53	Event #53	W	0
20	E52	Event #52	W	0
19	E51	Event #51	W	0
18	E50	Event #50	W	0
17	E49	Event #49	W	0
16	E48	Event #48	W	0
15	E47	Event #47	W	0
14	E46	Event #46	W	0
13	E45	Event #45	W	0
12	E44	Event #44	W	0
11	E43	Event #43	W	0
10	E42	Event #42	W	0
9	E41	Event #41	W	0
8	E40	Event #40	W	0
7	E39	Event #39	W	0
6	E38	Event #38	W	0
5	E37	Event #37	W	0
4	E36	Event #36	W	0
3	E35	Event #35	W	0
2	E34	Event #34	W	0
1	E33	Event #33	W	0
0	E32	Event #32	W	0

Table 14-409. Register Call Summary for Register TPCC_ESRH_Rn

IVA2.2 Subsystem Register Manual

- TPCC Register Mapping Summary: [0]

Table 14-410. TPCC_CER_Rn

Address Offset	0x2018 + (0x200*n) n = 0 to 7		
Physical address	0x01C0 2018 + (0x200*n) n = 0 to 7	**Instance**	IVA2.2 TPCC
Description	Chained Event Register: If CER.En bit is set (regardless of state of EER.En), then the corresponding DMA channel is prioritized vs. other pending DMA events for submission to the TC. CER.En bit is set when a chaining completion code is returned from one of the TPTCs through the completion interface, or is generated internally through Early Completion path. CER.En bit is cleared when the corresponding event is prioritized and serviced. If the CER.En bit is already set and the corresponding chaining completion code is returned from the TC, then the corresponding bit in the Event Missed Register is set. CER.En cannot be set or cleared through software.		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
E31	E30	E29	E28	E27	E26	E25	E24	E23	E22	E21	E20	E19	E18	E17	E16	E15	E14	E13	E12	E11	E10	E9	E8	E7	E6	E5	E4	E3	E2	E1	E0

Bits	Field Name	Description	Type	Reset
31	E31	Event #31	R	0
30	E30	Event #30	R	0
29	E29	Event #29	R	0
28	E28	Event #28	R	0
27	E27	Event #27	R	0
26	E26	Event #26	R	0
25	E25	Event #25	R	0
24	E24	Event #24	R	0
23	E23	Event #23	R	0
22	E22	Event #22	R	0
21	E21	Event #21	R	0
20	E20	Event #20	R	0
19	E19	Event #19	R	0
18	E18	Event #18	R	0
17	E17	Event #17	R	0
16	E16	Event #16	R	0
15	E15	Event #15	R	0
14	E14	Event #14	R	0
13	E13	Event #13	R	0
12	E12	Event #12	R	0
11	E11	Event #11	R	0
10	E10	Event #10	R	0
9	E9	Event #9	R	0
8	E8	Event #8	R	0
7	E7	Event #7	R	0
6	E6	Event #6	R	0
5	E5	Event #5	R	0
4	E4	Event #4	R	0
3	E3	Event #3	R	0
2	E2	Event #2	R	0
1	E1	Event #1	R	0
0	E0	Event #0	R	0

Table 14-411. Register Call Summary for Register TPCC_CER_Rn

IVA2.2 Subsystem Register Manual

- TPCC Register Mapping Summary: [0]

Table 14-412. TPCC_CERH_Rn

Address Offset	0x201C + (0x200*n) n = 0 to 7		
Physical address	0x01C0 201C + (0x200*n) n = 0 to 7	**Instance**	IVA2.2 TPCC
Description	Chained Event Register (High Part): If CERH.En bit is set (regardless of state of EERH.En), then the corresponding DMA channel is prioritized vs. other pending DMA events for submission to the TC. CERH.En bit is set when a chaining completion code is returned from one of the TPTCs through the completion interface, or is generated internally through Early Completion path. CERH.En bit is cleared when the corresponding event is prioritized and serviced. If the CERH.En bit is already set and the corresponding chaining completion code is returned from the TC, then the corresponding bit in the Event Missed Register is set. CERH.En cannot be set or cleared through software.		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
E63	E62	E61	E60	E59	E58	E57	E56	E55	E54	E53	E52	E51	E50	E49	E48	E47	E46	E45	E44	E43	E42	E41	E40	E39	E38	E37	E36	E35	E34	E33	E32

Bits	Field Name	Description	Type	Reset
31	E63	Event #63	R	0
30	E62	Event #62	R	0
29	E61	Event #61	R	0
28	E60	Event #60	R	0
27	E59	Event #59	R	0
26	E58	Event #58	R	0
25	E57	Event #57	R	0
24	E56	Event #56	R	0
23	E55	Event #55	R	0
22	E54	Event #54	R	0
21	E53	Event #53	R	0
20	E52	Event #52	R	0
19	E51	Event #51	R	0
18	E50	Event #50	R	0
17	E49	Event #49	R	0
16	E48	Event #48	R	0
15	E47	Event #47	R	0
14	E46	Event #46	R	0
13	E45	Event #45	R	0
12	E44	Event #44	R	0
11	E43	Event #43	R	0
10	E42	Event #42	R	0
9	E41	Event #41	R	0
8	E40	Event #40	R	0
7	E39	Event #39	R	0
6	E38	Event #38	R	0
5	E37	Event #37	R	0
4	E36	Event #36	R	0
3	E35	Event #35	R	0
2	E34	Event #34	R	0

Bits	Field Name	Description	Type	Reset
1	E33	Event #33	R	0
0	E32	Event #32	R	0

Table 14-413. Register Call Summary for Register TPCC_CERH_Rn

IVA2.2 Subsystem Register Manual

- TPCC Register Mapping Summary: [0]

Table 14-414. TPCC_EER_Rn

Address Offset	0x2020 + (0x200*n) n = 0 to 7		
Physical address	0x01C0 2020 + (0x200*n) n = 0 to 7	**Instance**	IVA2.2 TPCC
Description	Event Enable Register: Enables DMA transfers for ER.En pending events. ER.En is set based on externally asserted events (through tpcc_eventN_pi). This register has no effect on Chained Event Register (CER) or Event Set Register (ESR). Note that if a bit is set in ER.En while EER.En is disabled, no action is taken. If EER.En is enabled at a later point (and ER.En has not been cleared through SW) then the event will be recognized as a valid TR Sync EER.En is not directly writeable. Events can be enabled through writes to EESR and can be disabled through writes to EECR register. EER.En = 0: ER.En is not enabled to trigger DMA transfers. EER.En = 1: ER.En is enabled to trigger DMA transfers.		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved												E19	E18	E17	E16	E15	E14	E13	E12	E11	E10	E9	E8	E7	E6	E5	E4	E3	E2	E1	E0

Bits	Field Name	Description	Type	Reset
31:20	Reserved	Reserved	R	0
19	E19	Event #19	R	0
18	E18	Event #18	R	0
17	E17	Event #17	R	0
16	E16	Event #16	R	0
15	E15	Event #15	R	0
14	E14	Event #14	R	0
13	E13	Event #13	R	0
12	E12	Event #12	R	0
11	E11	Event #11	R	0
10	E10	Event #10	R	0
9	E9	Event #9	R	0
8	E8	Event #8	R	0
7	E7	Event #7	R	0
6	E6	Event #6	R	0
5	E5	Event #5	R	0
4	E4	Event #4	R	0
3	E3	Event #3	R	0
2	E2	Event #2	R	0
1	E1	Event #1	R	0
0	E0	Event #0	R	0

Table 14-415. Register Call Summary for Register TPCC_EER_Rn

IVA2.2 Subsystem Register Manual

- TPCC Register Mapping Summary: [0]

Table 14-416. TPCC_EECR_Rn

Address Offset	0x2028 + (0x200*n) n = 0 to 7		
Physical address	0x01C0 2028 + (0x200*n) n = 0 to 7	**Instance**	IVA2.2 TPCC
Description	Event Enable Clear Register: CPU write of 1 to the EECR.En bit causes the EER.En bit to be cleared. CPU write of 0 has no effect.		
Type	W		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved												E19	E18	E17	E16	E15	E14	E13	E12	E11	E10	E9	E8	E7	E6	E5	E4	E3	E2	E1	E0

Bits	Field Name	Description	Type	Reset
31:20	Reserved	Reserved	W	0
19	E19	Event #19	W	0
18	E18	Event #18	W	0
17	E17	Event #17	W	0
16	E16	Event #16	W	0
15	E15	Event #15	W	0
14	E14	Event #14	W	0
13	E13	Event #13	W	0
12	E12	Event #12	W	0
11	E11	Event #11	W	0
10	E10	Event #10	W	0
9	E9	Event #9	W	0
8	E8	Event #8	W	0
7	E7	Event #7	W	0
6	E6	Event #6	W	0
5	E5	Event #5	W	0
4	E4	Event #4	W	0
3	E3	Event #3	W	0
2	E2	Event #2	W	0
1	E1	Event #1	W	0
0	E0	Event #0	W	0

Table 14-417. Register Call Summary for Register TPCC_EECR_Rn

IVA2.2 Subsystem Register Manual

- TPCC Register Mapping Summary: [0]

Table 14-418. TPCC_EESR_Rn

Address Offset	0x2030 + (0x200*n) n = 0 to 7
Physical address	0x01C0 2030 + (0x200*n) n = 0 to 7 **Instance** IVA2.2 TPCC
Description	Event Enable Set Register: CPU write of 1 to the EESR.En bit causes the EER.En bit to be set. CPU write of 0 has no effect.
Type	W

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved												E19	E18	E17	E16	E15	E14	E13	E12	E11	E10	E9	E8	E7	E6	E5	E4	E3	E2	E1	E0

Bits	Field Name	Description	Type	Reset
31:20	Reserved	Reserved	W	0
19	E19	Event #19	W	0
18	E18	Event #18	W	0
17	E17	Event #17	W	0
16	E16	Event #16	W	0
15	E15	Event #15	W	0
14	E14	Event #14	W	0
13	E13	Event #13	W	0
12	E12	Event #12	W	0
11	E11	Event #11	W	0
10	E10	Event #10	W	0
9	E9	Event #9	W	0
8	E8	Event #8	W	0
7	E7	Event #7	W	0
6	E6	Event #6	W	0
5	E5	Event #5	W	0
4	E4	Event #4	W	0
3	E3	Event #3	W	0
2	E2	Event #2	W	0
1	E1	Event #1	W	0
0	E0	Event #0	W	0

Table 14-419. Register Call Summary for Register TPCC_EESR_Rn

IVA2.2 Subsystem Register Manual

- TPCC Register Mapping Summary: [0]

Table 14-420. TPCC_SER_Rn

Address Offset	0x2038 + (0x200*n) n = 0 to 7		
Physical address	0x01C0 2038 + (0x200*n) n = 0 to 7	**Instance**	IVA2.2 TPCC
Description	Secondary Event Register: The secondary event register is used along with the Event Register (ER) to provide information on the state of an Event. En = 0: Event is not currently in the Event Queue. En = 1: Event is currently stored in Event Queue. Event arbiter will not prioritize additional events.		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved												E19	E18	E17	E16	E15	E14	E13	E12	E11	E10	E9	E8	E7	E6	E5	E4	E3	E2	E1	E0

Bits	Field Name	Description	Type	Reset
31:20	Reserved	Reserved	R	0
19	E19	Event #19	R	0
18	E18	Event #18	R	0
17	E17	Event #17	R	0
16	E16	Event #16	R	0
15	E15	Event #15	R	0
14	E14	Event #14	R	0
13	E13	Event #13	R	0
12	E12	Event #12	R	0
11	E11	Event #11	R	0
10	E10	Event #10	R	0
9	E9	Event #9	R	0
8	E8	Event #8	R	0
7	E7	Event #7	R	0
6	E6	Event #6	R	0
5	E5	Event #5	R	0
4	E4	Event #4	R	0
3	E3	Event #3	R	0
2	E2	Event #2	R	0
1	E1	Event #1	R	0
0	E0	Event #0	R	0

Table 14-421. Register Call Summary for Register TPCC_SER_Rn

IVA2.2 Subsystem Register Manual

- TPCC Register Mapping Summary: [0]

Table 14-422. TPCC_SERH_Rn

Address Offset	0x203C + (0x200*n) n = 0 to 7
Physical address	0x01C0 203C + (0x200*n) n = 0 to 7 **Instance** IVA2.2 TPCC
Description	Secondary Event Register (High Part): The secondary event register is used along with the Event Register (ERH) to provide information on the state of an Event. En = 0: Event is not currently in the Event Queue. En = 1: Event is currently stored in Event Queue. Event arbiter will not prioritize additional events.
Type	R

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
E63	E62	E61	E60	E59	E58	E57	E56	E55	E54	E53	E52	E51	E50	E49	E48	E47	E46	E45	E44	E43	E42	E41	E40	E39	E38	E37	E36	E35	E34	E33	E32

Bits	Field Name	Description	Type	Reset
31	E63	Event #63	R	0
30	E62	Event #62	R	0
29	E61	Event #61	R	0
28	E60	Event #60	R	0
27	E59	Event #59	R	0
26	E58	Event #58	R	0
25	E57	Event #57	R	0
24	E56	Event #56	R	0
23	E55	Event #55	R	0
22	E54	Event #54	R	0
21	E53	Event #53	R	0
20	E52	Event #52	R	0
19	E51	Event #51	R	0
18	E50	Event #50	R	0
17	E49	Event #49	R	0
16	E48	Event #48	R	0
15	E47	Event #47	R	0
14	E46	Event #46	R	0
13	E45	Event #45	R	0
12	E44	Event #44	R	0
11	E43	Event #43	R	0
10	E42	Event #42	R	0
9	E41	Event #41	R	0
8	E40	Event #40	R	0
7	E39	Event #39	R	0
6	E38	Event #38	R	0
5	E37	Event #37	R	0
4	E36	Event #36	R	0
3	E35	Event #35	R	0
2	E34	Event #34	R	0
1	E33	Event #33	R	0
0	E32	Event #32	R	0

Table 14-423. Register Call Summary for Register TPCC_SERH_Rn

IVA2.2 Subsystem Register Manual

- TPCC Register Mapping Summary: [0]

Table 14-424. TPCC_SECR_Rn

Address Offset	0x2040 + (0x200*n) n = 0 to 7		
Physical address	0x01C0 2040 + (0x200*n) n = 0 to 7	**Instance**	IVA2.2 TPCC
Description	Secondary Event Clear Register: The secondary event clear register is used to clear the status of the SER registers. CPU write of 1 to the SECR.En bit clears the SER register. CPU write of 0 has no effect.		
Type	W		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
E31	E30	E29	E28	E27	E26	E25	E24	E23	E22	E21	E20	E19	E18	E17	E16	E15	E14	E13	E12	E11	E10	E9	E8	E7	E6	E5	E4	E3	E2	E1	E0

Bits	Field Name	Description	Type	Reset
31	E31	Event #31	W	0
30	E30	Event #30	W	0
29	E29	Event #29	W	0
28	E28	Event #28	W	0
27	E27	Event #27	W	0
26	E26	Event #26	W	0
25	E25	Event #25	W	0
24	E24	Event #24	W	0
23	E23	Event #23	W	0
22	E22	Event #22	W	0
21	E21	Event #21	W	0
20	E20	Event #20	W	0
19	E19	Event #19	W	0
18	E18	Event #18	W	0
17	E17	Event #17	W	0
16	E16	Event #16	W	0
15	E15	Event #15	W	0
14	E14	Event #14	W	0
13	E13	Event #13	W	0
12	E12	Event #12	W	0
11	E11	Event #11	W	0
10	E10	Event #10	W	0
9	E9	Event #9	W	0
8	E8	Event #8	W	0
7	E7	Event #7	W	0
6	E6	Event #6	W	0
5	E5	Event #5	W	0
4	E4	Event #4	W	0
3	E3	Event #3	W	0
2	E2	Event #2	W	0
1	E1	Event #1	W	0
0	E0	Event #0	W	0

Table 14-425. Register Call Summary for Register TPCC_SECR_Rn

IVA2.2 Subsystem Register Manual

- TPCC Register Mapping Summary: [0]

Table 14-426. TPCC_SECRH_Rn

Address Offset	0x2044 + (0x200*n) n = 0 to 7
Physical address	0x01C0 2044 + (0x200*n) n = 0 to 7 **Instance** IVA2.2 TPCC
Description	Secondary Event Clear Register (High Part): The secondary event clear register is used to clear the status of the SERH registers. CPU write of 1 to the SECRH.En bit clears the SERH register. CPU write of 0 has no effect.
Type	W

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
E63	E62	E61	E60	E59	E58	E57	E56	E55	E54	E53	E52	E51	E50	E49	E48	E47	E46	E45	E44	E43	E42	E41	E40	E39	E38	E37	E36	E35	E34	E33	E32

Bits	Field Name	Description	Type	Reset
31	E63	Event #63	W	0
30	E62	Event #62	W	0
29	E61	Event #61	W	0
28	E60	Event #60	W	0
27	E59	Event #59	W	0
26	E58	Event #58	W	0
25	E57	Event #57	W	0
24	E56	Event #56	W	0
23	E55	Event #55	W	0
22	E54	Event #54	W	0
21	E53	Event #53	W	0
20	E52	Event #52	W	0
19	E51	Event #51	W	0
18	E50	Event #50	W	0
17	E49	Event #49	W	0
16	E48	Event #48	W	0
15	E47	Event #47	W	0
14	E46	Event #46	W	0
13	E45	Event #45	W	0
12	E44	Event #44	W	0
11	E43	Event #43	W	0
10	E42	Event #42	W	0
9	E41	Event #41	W	0
8	E40	Event #40	W	0
7	E39	Event #39	W	0
6	E38	Event #38	W	0
5	E37	Event #37	W	0
4	E36	Event #36	W	0
3	E35	Event #35	W	0
2	E34	Event #34	W	0
1	E33	Event #33	W	0
0	E32	Event #32	W	0

Table 14-427. Register Call Summary for Register TPCC_SECRH_Rn

IVA2.2 Subsystem Register Manual

- TPCC Register Mapping Summary: [0]

Table 14-428. TPCC_IER_Rn

Address Offset	0x2050 + (0x200*n) n = 0 to 7		
Physical address	0x01C0 2050 + (0x200*n) n = 0 to 7	**Instance**	IVA2.2 TPCC
Description	Int Enable Register: IER.In is not directly writeable. Interrupts can be enabled through writes to IESR and can be disabled through writes to IECR register. IER.In = 0: IPR.In is NOT enabled for interrupts. IER.In = 1: IPR.In IS enabled for interrupts.		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
I31	I30	I29	I28	I27	I26	I25	I24	I23	I22	I21	I20	I19	I18	I17	I16	I15	I14	I13	I12	I11	I10	I9	I8	I7	I6	I5	I4	I3	I2	I1	I0

Bits	Field Name	Description	Type	Reset
31	I31	Interrupt associated with TCC #31	R	0
30	I30	Interrupt associated with TCC #30	R	0
29	I29	Interrupt associated with TCC #29	R	0
28	I28	Interrupt associated with TCC #28	R	0
27	I27	Interrupt associated with TCC #27	R	0
26	I26	Interrupt associated with TCC #26	R	0
25	I25	Interrupt associated with TCC #25	R	0
24	I24	Interrupt associated with TCC #24	R	0
23	I23	Interrupt associated with TCC #23	R	0
22	I22	Interrupt associated with TCC #22	R	0
21	I21	Interrupt associated with TCC #21	R	0
20	I20	Interrupt associated with TCC #20	R	0
19	I19	Interrupt associated with TCC #19	R	0
18	I18	Interrupt associated with TCC #18	R	0
17	I17	Interrupt associated with TCC #17	R	0
16	I16	Interrupt associated with TCC #16	R	0
15	I15	Interrupt associated with TCC #15	R	0
14	I14	Interrupt associated with TCC #14	R	0
13	I13	Interrupt associated with TCC #13	R	0
12	I12	Interrupt associated with TCC #12	R	0
11	I11	Interrupt associated with TCC #11	R	0
10	I10	Interrupt associated with TCC #10	R	0
9	I9	Interrupt associated with TCC #9	R	0
8	I8	Interrupt associated with TCC #8	R	0
7	I7	Interrupt associated with TCC #7	R	0
6	I6	Interrupt associated with TCC #6	R	0
5	I5	Interrupt associated with TCC #5	R	0
4	I4	Interrupt associated with TCC #4	R	0
3	I3	Interrupt associated with TCC #3	R	0
2	I2	Interrupt associated with TCC #2	R	0
1	I1	Interrupt associated with TCC #1	R	0
0	I0	Interrupt associated with TCC #0	R	0

Table 14-429. Register Call Summary for Register TPCC_IER_Rn

IVA2.2 Subsystem Register Manual

- TPCC Register Mapping Summary: [0]

Table 14-430. TPCC_IERH_Rn

Address Offset	0x2054 + (0x200*n) n = 0 to 7
Physical address	0x01C0 2054 + (0x200*n) n = 0 to 7 **Instance** IVA2.2 TPCC
Description	Int Enable Register (High Part): IERH.In is not directly writeable. Interrupts can be enabled through writes to IESRH and can be disabled through writes to IECRH register. IERH.In = 0: IPRH.In is NOT enabled for interrupts. IERH.In = 1: IPRH.In IS enabled for interrupts.
Type	R

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
I63	I62	I61	I60	I59	I58	I57	I56	I55	I54	I53	I52	I51	I50	I49	I48	I47	I46	I45	I44	I43	I42	I41	I40	I39	I38	I37	I36	I35	I34	I33	I32

Bits	Field Name	Description	Type	Reset
31	I63	Interrupt associated with TCC #63	R	0
30	I62	Interrupt associated with TCC #62	R	0
29	I61	Interrupt associated with TCC #61	R	0
28	I60	Interrupt associated with TCC #60	R	0
27	I59	Interrupt associated with TCC #59	R	0
26	I58	Interrupt associated with TCC #58	R	0
25	I57	Interrupt associated with TCC #57	R	0
24	I56	Interrupt associated with TCC #56	R	0
23	I55	Interrupt associated with TCC #55	R	0
22	I54	Interrupt associated with TCC #54	R	0
21	I53	Interrupt associated with TCC #53	R	0
20	I52	Interrupt associated with TCC #52	R	0
19	I51	Interrupt associated with TCC #51	R	0
18	I50	Interrupt associated with TCC #50	R	0
17	I49	Interrupt associated with TCC #49	R	0
16	I48	Interrupt associated with TCC #48	R	0
15	I47	Interrupt associated with TCC #47	R	0
14	I46	Interrupt associated with TCC #46	R	0
13	I45	Interrupt associated with TCC #45	R	0
12	I44	Interrupt associated with TCC #44	R	0
11	I43	Interrupt associated with TCC #43	R	0
10	I42	Interrupt associated with TCC #42	R	0
9	I41	Interrupt associated with TCC #41	R	0
8	I40	Interrupt associated with TCC #40	R	0
7	I39	Interrupt associated with TCC #39	R	0
6	I38	Interrupt associated with TCC #38	R	0
5	I37	Interrupt associated with TCC #37	R	0
4	I36	Interrupt associated with TCC #36	R	0
3	I35	Interrupt associated with TCC #35	R	0
2	I34	Interrupt associated with TCC #34	R	0
1	I33	Interrupt associated with TCC #33	R	0
0	I32	Interrupt associated with TCC #32	R	0

Table 14-431. Register Call Summary for Register TPCC_IERH_Rn

IVA2.2 Subsystem Register Manual

- TPCC Register Mapping Summary: [0]

Table 14-432. TPCC_IECR_Rn

Address Offset	0x2058 + (0x200*n) n = 0 to 7		
Physical address	0x01C0 2058 + (0x200*n) n = 0 to 7	**Instance**	IVA2.2 TPCC
Description	Int Enable Clear Register: CPU write of 1 to the IECR.In bit causes the IER.In bit to be cleared. CPU write of 0 has no effect.		
Type	W		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
I31	I30	I29	I28	I27	I26	I25	I24	I23	I22	I21	I20	I19	I18	I17	I16	I15	I14	I13	I12	I11	I10	I9	I8	I7	I6	I5	I4	I3	I2	I1	I0

Bits	Field Name	Description	Type	Reset
31	I31	Interrupt associated with TCC #31	W	0
30	I30	Interrupt associated with TCC #30	W	0
29	I29	Interrupt associated with TCC #29	W	0
28	I28	Interrupt associated with TCC #28	W	0
27	I27	Interrupt associated with TCC #27	W	0
26	I26	Interrupt associated with TCC #26	W	0
25	I25	Interrupt associated with TCC #25	W	0
24	I24	Interrupt associated with TCC #24	W	0
23	I23	Interrupt associated with TCC #23	W	0
22	I22	Interrupt associated with TCC #22	W	0
21	I21	Interrupt associated with TCC #21	W	0
20	I20	Interrupt associated with TCC #20	W	0
19	I19	Interrupt associated with TCC #19	W	0
18	I18	Interrupt associated with TCC #18	W	0
17	I17	Interrupt associated with TCC #17	W	0
16	I16	Interrupt associated with TCC #16	W	0
15	I15	Interrupt associated with TCC #15	W	0
14	I14	Interrupt associated with TCC #14	W	0
13	I13	Interrupt associated with TCC #13	W	0
12	I12	Interrupt associated with TCC #12	W	0
11	I11	Interrupt associated with TCC #11	W	0
10	I10	Interrupt associated with TCC #10	W	0
9	I9	Interrupt associated with TCC #9	W	0
8	I8	Interrupt associated with TCC #8	W	0
7	I7	Interrupt associated with TCC #7	W	0
6	I6	Interrupt associated with TCC #6	W	0
5	I5	Interrupt associated with TCC #5	W	0
4	I4	Interrupt associated with TCC #4	W	0
3	I3	Interrupt associated with TCC #3	W	0
2	I2	Interrupt associated with TCC #2	W	0
1	I1	Interrupt associated with TCC #1	W	0
0	I0	Interrupt associated with TCC #0	W	0

Table 14-433. Register Call Summary for Register TPCC_IECR_Rn

IVA2.2 Subsystem Register Manual

- TPCC Register Mapping Summary: [0]

Table 14-434. TPCC_IECRH_Rn

Address Offset	0x205C + (0x200*n) n = 0 to 7		
Physical address	0x01C0 205C + (0x200*n) n = 0 to 7	**Instance**	IVA2.2 TPCC
Description	Int Enable Clear Register (High Part): CPU write of 1 to the IECRH.In bit causes the IERH.In bit to be cleared. CPU write of 0 has no effect.		
Type	W		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
I63	I62	I61	I60	I59	I58	I57	I56	I55	I54	I53	I52	I51	I50	I49	I48	I47	I46	I45	I44	I43	I42	I41	I40	I39	I38	I37	I36	I35	I34	I33	I32

Bits	Field Name	Description	Type	Reset
31	I63	Interrupt associated with TCC #63	W	0
30	I62	Interrupt associated with TCC #62	W	0
29	I61	Interrupt associated with TCC #61	W	0
28	I60	Interrupt associated with TCC #60	W	0
27	I59	Interrupt associated with TCC #59	W	0
26	I58	Interrupt associated with TCC #58	W	0
25	I57	Interrupt associated with TCC #57	W	0
24	I56	Interrupt associated with TCC #56	W	0
23	I55	Interrupt associated with TCC #55	W	0
22	I54	Interrupt associated with TCC #54	W	0
21	I53	Interrupt associated with TCC #53	W	0
20	I52	Interrupt associated with TCC #52	W	0
19	I51	Interrupt associated with TCC #51	W	0
18	I50	Interrupt associated with TCC #50	W	0
17	I49	Interrupt associated with TCC #49	W	0
16	I48	Interrupt associated with TCC #48	W	0
15	I47	Interrupt associated with TCC #47	W	0
14	I46	Interrupt associated with TCC #46	W	0
13	I45	Interrupt associated with TCC #45	W	0
12	I44	Interrupt associated with TCC #44	W	0
11	I43	Interrupt associated with TCC #43	W	0
10	I42	Interrupt associated with TCC #42	W	0
9	I41	Interrupt associated with TCC #41	W	0
8	I40	Interrupt associated with TCC #40	W	0
7	I39	Interrupt associated with TCC #39	W	0
6	I38	Interrupt associated with TCC #38	W	0
5	I37	Interrupt associated with TCC #37	W	0
4	I36	Interrupt associated with TCC #36	W	0
3	I35	Interrupt associated with TCC #35	W	0
2	I34	Interrupt associated with TCC #34	W	0
1	I33	Interrupt associated with TCC #33	W	0
0	I32	Interrupt associated with TCC #32	W	0

Table 14-435. Register Call Summary for Register TPCC_IECRH_Rn

IVA2.2 Subsystem Register Manual

- TPCC Register Mapping Summary: [0]

Table 14-436. TPCC_IESR_Rn

Address Offset	0x2060 + (0x200*n) n = 0 to 7		
Physical address	0x01C0 2060 + (0x200*n) n = 0 to 7	**Instance**	IVA2.2 TPCC
Description	Int Enable Set Register: CPU write of 1 to the IESR.In bit causes the IESR.In bit to be set. CPU write of 0 has no effect.		
Type	W		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
I31	I30	I29	I28	I27	I26	I25	I24	I23	I22	I21	I20	I19	I18	I17	I16	I15	I14	I13	I12	I11	I10	I9	I8	I7	I6	I5	I4	I3	I2	I1	I0

Bits	Field Name	Description	Type	Reset
31	I31	Interrupt associated with TCC #31	W	0
30	I30	Interrupt associated with TCC #30	W	0
29	I29	Interrupt associated with TCC #29	W	0
28	I28	Interrupt associated with TCC #28	W	0
27	I27	Interrupt associated with TCC #27	W	0
26	I26	Interrupt associated with TCC #26	W	0
25	I25	Interrupt associated with TCC #25	W	0
24	I24	Interrupt associated with TCC #24	W	0
23	I23	Interrupt associated with TCC #23	W	0
22	I22	Interrupt associated with TCC #22	W	0
21	I21	Interrupt associated with TCC #21	W	0
20	I20	Interrupt associated with TCC #20	W	0
19	I19	Interrupt associated with TCC #19	W	0
18	I18	Interrupt associated with TCC #18	W	0
17	I17	Interrupt associated with TCC #17	W	0
16	I16	Interrupt associated with TCC #16	W	0
15	I15	Interrupt associated with TCC #15	W	0
14	I14	Interrupt associated with TCC #14	W	0
13	I13	Interrupt associated with TCC #13	W	0
12	I12	Interrupt associated with TCC #12	W	0
11	I11	Interrupt associated with TCC #11	W	0
10	I10	Interrupt associated with TCC #10	W	0
9	I9	Interrupt associated with TCC #9	W	0
8	I8	Interrupt associated with TCC #8	W	0
7	I7	Interrupt associated with TCC #7	W	0
6	I6	Interrupt associated with TCC #6	W	0
5	I5	Interrupt associated with TCC #5	W	0
4	I4	Interrupt associated with TCC #4	W	0
3	I3	Interrupt associated with TCC #3	W	0
2	I2	Interrupt associated with TCC #2	W	0
1	I1	Interrupt associated with TCC #1	W	0
0	I0	Interrupt associated with TCC #0	W	0

Table 14-437. Register Call Summary for Register TPCC_IESR_Rn

IVA2.2 Subsystem Register Manual

- TPCC Register Mapping Summary: [0]

Table 14-438. TPCC_IESRH_Rn

Address Offset	0x2064 + (0x200*n) n = 0 to 7		
Physical address	0x01C0 2064 + (0x200*n) n = 0 to 7	**Instance**	IVA2.2 TPCC
Description	Int Enable Set Register (High Part): CPU write of 1 to the IESRH.In bit causes the IESRH.In bit to be set. CPU write of 0 has no effect.		
Type	W		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
I63	I62	I61	I60	I59	I58	I57	I56	I55	I54	I53	I52	I51	I50	I49	I48	I47	I46	I45	I44	I43	I42	I41	I40	I39	I38	I37	I36	I35	I34	I33	I32

Bits	Field Name	Description	Type	Reset
31	I63	Interrupt associated with TCC #63	W	0
30	I62	Interrupt associated with TCC #62	W	0
29	I61	Interrupt associated with TCC #61	W	0
28	I60	Interrupt associated with TCC #60	W	0
27	I59	Interrupt associated with TCC #59	W	0
26	I58	Interrupt associated with TCC #58	W	0
25	I57	Interrupt associated with TCC #57	W	0
24	I56	Interrupt associated with TCC #56	W	0
23	I55	Interrupt associated with TCC #55	W	0
22	I54	Interrupt associated with TCC #54	W	0
21	I53	Interrupt associated with TCC #53	W	0
20	I52	Interrupt associated with TCC #52	W	0
19	I51	Interrupt associated with TCC #51	W	0
18	I50	Interrupt associated with TCC #50	W	0
17	I49	Interrupt associated with TCC #49	W	0
16	I48	Interrupt associated with TCC #48	W	0
15	I47	Interrupt associated with TCC #47	W	0
14	I46	Interrupt associated with TCC #46	W	0
13	I45	Interrupt associated with TCC #45	W	0
12	I44	Interrupt associated with TCC #44	W	0
11	I43	Interrupt associated with TCC #43	W	0
10	I42	Interrupt associated with TCC #42	W	0
9	I41	Interrupt associated with TCC #41	W	0
8	I40	Interrupt associated with TCC #40	W	0
7	I39	Interrupt associated with TCC #39	W	0
6	I38	Interrupt associated with TCC #38	W	0
5	I37	Interrupt associated with TCC #37	W	0
4	I36	Interrupt associated with TCC #36	W	0
3	I35	Interrupt associated with TCC #35	W	0
2	I34	Interrupt associated with TCC #34	W	0
1	I33	Interrupt associated with TCC #33	W	0
0	I32	Interrupt associated with TCC #32	W	0

Table 14-439. Register Call Summary for Register TPCC_IESRH_Rn

IVA2.2 Subsystem Register Manual

- TPCC Register Mapping Summary: [0]

Table 14-440. TPCC_IPR_Rn

Address Offset	0x2068 + (0x200*n) n = 0 to 7		
Physical address	0x01C0 2068 + (0x200*n) n = 0 to 7	**Instance**	IVA2.2 TPCC
Description	Interrupt Pending Register: IPR.In bit is set when a interrupt completion code with TCC of N is detected. IPR.In bit is cleared through software by writing 1 to ICR.In bit.		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
I31	I30	I29	I28	I27	I26	I25	I24	I23	I22	I21	I20	I19	I18	I17	I16	I15	I14	I13	I12	I11	I10	I9	I8	I7	I6	I5	I4	I3	I2	I1	I0

Bits	Field Name	Description	Type	Reset
31	I31	Interrupt associated with TCC #31	R	0
30	I30	Interrupt associated with TCC #30	R	0
29	I29	Interrupt associated with TCC #29	R	0
28	I28	Interrupt associated with TCC #28	R	0
27	I27	Interrupt associated with TCC #27	R	0
26	I26	Interrupt associated with TCC #26	R	0
25	I25	Interrupt associated with TCC #25	R	0
24	I24	Interrupt associated with TCC #24	R	0
23	I23	Interrupt associated with TCC #23	R	0
22	I22	Interrupt associated with TCC #22	R	0
21	I21	Interrupt associated with TCC #21	R	0
20	I20	Interrupt associated with TCC #20	R	0
19	I19	Interrupt associated with TCC #19	R	0
18	I18	Interrupt associated with TCC #18	R	0
17	I17	Interrupt associated with TCC #17	R	0
16	I16	Interrupt associated with TCC #16	R	0
15	I15	Interrupt associated with TCC #15	R	0
14	I14	Interrupt associated with TCC #14	R	0
13	I13	Interrupt associated with TCC #13	R	0
12	I12	Interrupt associated with TCC #12	R	0
11	I11	Interrupt associated with TCC #11	R	0
10	I10	Interrupt associated with TCC #10	R	0
9	I9	Interrupt associated with TCC #9	R	0
8	I8	Interrupt associated with TCC #8	R	0
7	I7	Interrupt associated with TCC #7	R	0
6	I6	Interrupt associated with TCC #6	R	0
5	I5	Interrupt associated with TCC #5	R	0
4	I4	Interrupt associated with TCC #4	R	0
3	I3	Interrupt associated with TCC #3	R	0
2	I2	Interrupt associated with TCC #2	R	0
1	I1	Interrupt associated with TCC #1	R	0
0	I0	Interrupt associated with TCC #0	R	0

Table 14-441. Register Call Summary for Register TPCC_IPR_Rn

IVA2.2 Subsystem Register Manual

- TPCC Register Mapping Summary: [0]

Table 14-442. TPCC_IPRH_Rn

Address Offset	0x206C + (0x200*n) n = 0 to 7		
Physical address	0x01C0 206C + (0x200*n) n = 0 to 7	**Instance**	IVA2.2 TPCC
Description	Interrupt Pending Register (High Part): IPRH.In bit is set when a interrupt completion code with TCC of N is detected. IPRH.In bit is cleared through software by writing 1 to ICRH.In bit.		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
I63	I62	I61	I60	I59	I58	I57	I56	I55	I54	I53	I52	I51	I50	I49	I48	I47	I46	I45	I44	I43	I42	I41	I40	I39	I38	I37	I36	I35	I34	I33	I32

Bits	Field Name	Description	Type	Reset
31	I63	Interrupt associated with TCC #63	R	0
30	I62	Interrupt associated with TCC #62	R	0
29	I61	Interrupt associated with TCC #61	R	0
28	I60	Interrupt associated with TCC #60	R	0
27	I59	Interrupt associated with TCC #59	R	0
26	I58	Interrupt associated with TCC #58	R	0
25	I57	Interrupt associated with TCC #57	R	0
24	I56	Interrupt associated with TCC #56	R	0
23	I55	Interrupt associated with TCC #55	R	0
22	I54	Interrupt associated with TCC #54	R	0
21	I53	Interrupt associated with TCC #53	R	0
20	I52	Interrupt associated with TCC #52	R	0
19	I51	Interrupt associated with TCC #51	R	0
18	I50	Interrupt associated with TCC #50	R	0
17	I49	Interrupt associated with TCC #49	R	0
16	I48	Interrupt associated with TCC #48	R	0
15	I47	Interrupt associated with TCC #47	R	0
14	I46	Interrupt associated with TCC #46	R	0
13	I45	Interrupt associated with TCC #45	R	0
12	I44	Interrupt associated with TCC #44	R	0
11	I43	Interrupt associated with TCC #43	R	0
10	I42	Interrupt associated with TCC #42	R	0
9	I41	Interrupt associated with TCC #41	R	0
8	I40	Interrupt associated with TCC #40	R	0
7	I39	Interrupt associated with TCC #39	R	0
6	I38	Interrupt associated with TCC #38	R	0
5	I37	Interrupt associated with TCC #37	R	0
4	I36	Interrupt associated with TCC #36	R	0
3	I35	Interrupt associated with TCC #35	R	0
2	I34	Interrupt associated with TCC #34	R	0
1	I33	Interrupt associated with TCC #33	R	0
0	I32	Interrupt associated with TCC #32	R	0

Table 14-443. Register Call Summary for Register TPCC_IPRH_Rn

IVA2.2 Subsystem Register Manual

- TPCC Register Mapping Summary: [0]

Table 14-444. TPCC_ICR_Rn

Address Offset	0x2070 + (0x200*n) n = 0 to 7
Physical address	0x01C0 2070 + (0x200*n) n = 0 to 7 **Instance** IVA2.2 TPCC
Description	Interrupt Clear Register: CPU write of 1 to the ICR.In bit causes the IPR.In bit to be cleared. CPU write of 0 has no effect. All IPR.In bits must be cleared before additional interrupts will be asserted by CC.
Type	W

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
I31	I30	I29	I28	I27	I26	I25	I24	I23	I22	I21	I20	I19	I18	I17	I16	I15	I14	I13	I12	I11	I10	I9	I8	I7	I6	I5	I4	I3	I2	I1	I0

Bits	Field Name	Description	Type	Reset
31	I31	Interrupt associated with TCC #31	W	0
30	I30	Interrupt associated with TCC #30	W	0
29	I29	Interrupt associated with TCC #29	W	0
28	I28	Interrupt associated with TCC #28	W	0
27	I27	Interrupt associated with TCC #27	W	0
26	I26	Interrupt associated with TCC #26	W	0
25	I25	Interrupt associated with TCC #25	W	0
24	I24	Interrupt associated with TCC #24	W	0
23	I23	Interrupt associated with TCC #23	W	0
22	I22	Interrupt associated with TCC #22	W	0
21	I21	Interrupt associated with TCC #21	W	0
20	I20	Interrupt associated with TCC #20	W	0
19	I19	Interrupt associated with TCC #19	W	0
18	I18	Interrupt associated with TCC #18	W	0
17	I17	Interrupt associated with TCC #17	W	0
16	I16	Interrupt associated with TCC #16	W	0
15	I15	Interrupt associated with TCC #15	W	0
14	I14	Interrupt associated with TCC #14	W	0
13	I13	Interrupt associated with TCC #13	W	0
12	I12	Interrupt associated with TCC #12	W	0
11	I11	Interrupt associated with TCC #11	W	0
10	I10	Interrupt associated with TCC #10	W	0
9	I9	Interrupt associated with TCC #9	W	0
8	I8	Interrupt associated with TCC #8	W	0
7	I7	Interrupt associated with TCC #7	W	0
6	I6	Interrupt associated with TCC #6	W	0
5	I5	Interrupt associated with TCC #5	W	0
4	I4	Interrupt associated with TCC #4	W	0
3	I3	Interrupt associated with TCC #3	W	0
2	I2	Interrupt associated with TCC #2	W	0
1	I1	Interrupt associated with TCC #1	W	0
0	I0	Interrupt associated with TCC #0	W	0

Table 14-445. Register Call Summary for Register TPCC_ICR_Rn

IVA2.2 Subsystem Register Manual

- TPCC Register Mapping Summary: [0]

Table 14-446. TPCC_ICRH_Rn

Address Offset	0x2074 + (0x200*n) n = 0 to 7
Physical address	0x01C0 2074 + (0x200*n) n = 0 to 7 **Instance** IVA2.2 TPCC
Description	Interrupt Clear Register (High Part): CPU write of 1 to the ICRH.In bit causes the IPRH.In bit to be cleared. CPU write of 0 has no effect. All IPRH.In bits must be cleared before additional interrupts will be asserted by CC.
Type	W

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
I63	I62	I61	I60	I59	I58	I57	I56	I55	I54	I53	I52	I51	I50	I49	I48	I47	I46	I45	I44	I43	I42	I41	I40	I39	I38	I37	I36	I35	I34	I33	I32

Bits	Field Name	Description	Type	Reset
31	I63	Interrupt associated with TCC #63	W	0
30	I62	Interrupt associated with TCC #62	W	0
29	I61	Interrupt associated with TCC #61	W	0
28	I60	Interrupt associated with TCC #60	W	0
27	I59	Interrupt associated with TCC #59	W	0
26	I58	Interrupt associated with TCC #58	W	0
25	I57	Interrupt associated with TCC #57	W	0
24	I56	Interrupt associated with TCC #56	W	0
23	I55	Interrupt associated with TCC #55	W	0
22	I54	Interrupt associated with TCC #54	W	0
21	I53	Interrupt associated with TCC #53	W	0
20	I52	Interrupt associated with TCC #52	W	0
19	I51	Interrupt associated with TCC #51	W	0
18	I50	Interrupt associated with TCC #50	W	0
17	I49	Interrupt associated with TCC #49	W	0
16	I48	Interrupt associated with TCC #48	W	0
15	I47	Interrupt associated with TCC #47	W	0
14	I46	Interrupt associated with TCC #46	W	0
13	I45	Interrupt associated with TCC #45	W	0
12	I44	Interrupt associated with TCC #44	W	0
11	I43	Interrupt associated with TCC #43	W	0
10	I42	Interrupt associated with TCC #42	W	0
9	I41	Interrupt associated with TCC #41	W	0
8	I40	Interrupt associated with TCC #40	W	0
7	I39	Interrupt associated with TCC #39	W	0
6	I38	Interrupt associated with TCC #38	W	0
5	I37	Interrupt associated with TCC #37	W	0
4	I36	Interrupt associated with TCC #36	W	0
3	I35	Interrupt associated with TCC #35	W	0
2	I34	Interrupt associated with TCC #34	W	0
1	I33	Interrupt associated with TCC #33	W	0
0	I32	Interrupt associated with TCC #32	W	0

Table 14-447. Register Call Summary for Register TPCC_ICRH_Rn

IVA2.2 Subsystem Register Manual

- TPCC Register Mapping Summary: [0]

Table 14-448. TPCC_IEVAL_Rn

Address Offset	0x2078 + (0x200*n) n = 0 to 7		
Physical address	0x01C0 2078 + (0x200*n) n = 0 to 7	**Instance**	IVA2.2 TPCC
Description	Interrupt Eval Register		
Type	W		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																														SET	EVAL

Bits	Field Name	Description	Type	Reset
31:2	Reserved	Write 0s for future compatibility.	W	0x00000000
1	SET	Interrupt Set: CPU write of 1 to the SETn bit causes the tpcc_intN output signal to be pulsed egardless of state of interrupts enable (IERn) and status (IPRn). CPU write of 0 has no effect.	W	0
0	EVAL	Interrupt Evaluate: CPU write of 1 to the EVALn bit causes the tpcc_intN output signal to be pulsed if any enabled interrupts (IERn) are still pending (IPRn). CPU write of 0 has no effect.	W	0

Table 14-449. Register Call Summary for Register TPCC_IEVAL_Rn

IVA2.2 Subsystem Register Manual

- TPCC Register Mapping Summary: [0]

Table 14-450. TPCC_QER_Rn

Address Offset	0x2080 + (0x200*n) n = 0 to 7		
Physical address	0x01C0 2080 + (0x200*n) n = 0 to 7	**Instance**	IVA2.2 TPCC
Description	QDMA Event Register: If QER.En bit is set, then the corresponding QDMA channel is prioritized vs. other qdma events for submission to the TC. QER.En bit is set when a vbus write byte matches the address defined in the QCHMAPn register. QER.En bit is cleared when the corresponding event is prioritized and serviced. QER.En is also cleared when user writes a 1 to the QSECR.En bit. If the QER.En bit is already set and a new QDMA event is detected due to user write to QDMA trigger location and QEER register is set, then the corresponding bit in the QDMA Event Missed Register is set.		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																								E7	E6	E5	E4	E3	E2	E1	E0

Bits	Field Name	Description	Type	Reset
31:8	Reserved	Read returns 0.	R	0x000000
7	E7	Event #7	R	0
6	E6	Event #6	R	0
5	E5	Event #5	R	0
4	E4	Event #4	R	0
3	E3	Event #3	R	0
2	E2	Event #2	R	0
1	E1	Event #1	R	0
0	E0	Event #0	R	0

Table 14-451. Register Call Summary for Register TPCC_QER_Rn

IVA2.2 Subsystem Register Manual

- TPCC Register Mapping Summary: [0]

Table 14-452. TPCC_QEER_Rn

Address Offset	0x2084 + (0x200*n) n = 0 to 7		
Physical address	0x01C0 2084 + (0x200*n) n = 0 to 7	**Instance**	IVA2.2 TPCC
Description	QDMA Event Enable Register: Enabled/disabled QDMA address comparator for QDMA Channel N. QEER.En is not directly writeable. QDMA channels can be enabled through writes to QEESR and can be disabled through writes to QEECR register. QEER.En = 1, The corresponding QDMA channel comparator is enabled and Events will be recognized and latched in QER.En. QEER.En = 0, The corresponding QDMA channel comparator is disabled. Events will not be recognized/latched in QER.En.		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																								E7	E6	E5	E4	E3	E2	E1	E0

Bits	Field Name	Description	Type	Reset
31:8	Reserved	Read returns 0.	R	0x000000
7	E7	Event #7	R	0
6	E6	Event #6	R	0
5	E5	Event #5	R	0
4	E4	Event #4	R	0
3	E3	Event #3	R	0
2	E2	Event #2	R	0
1	E1	Event #1	R	0
0	E0	Event #0	R	0

Table 14-453. Register Call Summary for Register TPCC_QEER_Rn

IVA2.2 Subsystem Register Manual

- TPCC Register Mapping Summary: [0]

Table 14-454. TPCC_QEECR_Rn

Address Offset	0x2088 + (0x200*n) n = 0 to 7		
Physical address	0x01C0 2088 + (0x200*n) n = 0 to 7	**Instance**	IVA2.2 TPCC
Description	QDMA Event Enable Clear Register: CPU write of 1 to the QEECR.En bit causes the QEER.En bit to be cleared. CPU write of 0 has no effect.		
Type	W		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																								E7	E6	E5	E4	E3	E2	E1	E0

Bits	Field Name	Description	Type	Reset
31:8	Reserved	Write 0s for future compatibility.	W	0x000000
7	E7	Event #7	W	0
6	E6	Event #6	W	0
5	E5	Event #5	W	0
4	E4	Event #4	W	0
3	E3	Event #3	W	0

Bits	Field Name	Description	Type	Reset
2	E2	Event #2	W	0
1	E1	Event #1	W	0
0	E0	Event #0	W	0

Table 14-455. Register Call Summary for Register TPCC_QEECR_Rn

IVA2.2 Subsystem Register Manual

- TPCC Register Mapping Summary: [0]

Table 14-456. TPCC_QEESR_Rn

Address Offset	0x208C + (0x200*n) n = 0 to 7		
Physical address	0x01C0 208C + (0x200*n) n = 0 to 7	**Instance**	IVA2.2 TPCC
Description	QDMA Event Enable Set Register: CPU write of 1 to the QEESR.En bit causes the QEESR.En bit to be set. CPU write of 0 has no effect.		
Type	W		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																								E7	E6	E5	E4	E3	E2	E1	E0

Bits	Field Name	Description	Type	Reset
31:8	Reserved	Write 0s for future compatibility.	W	0x000000
7	E7	Event #7	W	0
6	E6	Event #6	W	0
5	E5	Event #5	W	0
4	E4	Event #4	W	0
3	E3	Event #3	W	0
2	E2	Event #2	W	0
1	E1	Event #1	W	0
0	E0	Event #0	W	0

Table 14-457. Register Call Summary for Register TPCC_QEESR_Rn

IVA2.2 Subsystem Register Manual

- TPCC Register Mapping Summary: [0]

Table 14-458. TPCC_QSER_Rn

Address Offset	0x2090 + (0x200*n) n = 0 to 7		
Physical address	0x01C0 2090 + (0x200*n) n = 0 to 7	**Instance**	IVA2.2 TPCC
Description	QDMA Secondary Event Register: The QDMA secondary event register is used along with the QDMA Event Register (QER) to provide information on the state of a QDMA Event. En = 0: Event is not currently in the Event Queue. En = 1: Event is currently stored in Event Queue. Event arbiter will not prioritize additional events.		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																								E7	E6	E5	E4	E3	E2	E1	E0

Bits	Field Name	Description	Type	Reset
31:8	Reserved	Read returns 0.	R	0x000000
7	E7	Event #7	R	0
6	E6	Event #6	R	0
5	E5	Event #5	R	0
4	E4	Event #4	R	0
3	E3	Event #3	R	0
2	E2	Event #2	R	0
1	E1	Event #1	R	0
0	E0	Event #0	R	0

Table 14-459. Register Call Summary for Register TPCC_QSER_Rn

IVA2.2 Subsystem Register Manual

- TPCC Register Mapping Summary: [0]

Table 14-460. TPCC_QSECR_Rn

Address Offset	0x2094 + (0x200*n) n = 0 to 7		
Physical address	0x01C0 2094 + (0x200*n) n = 0 to 7	**Instance**	IVA2.2 TPCC
Description	QDMA Secondary Event Clear Register: The secondary event clear register is used to clear the status of the QSER and QER register (note that this is slightly different than the SER operation, which does not clear the ER.En register). CPU write of 1 to the QSECR.En bit clears the QSER.En and QER.En register fields. CPU write of 0 has no effect.		
Type	W		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																								E7	E6	E5	E4	E3	E2	E1	E0

Bits	Field Name	Description	Type	Reset
31:8	Reserved	Write 0s for future compatibility.	W	0x000000
7	E7	Event #7	W	0
6	E6	Event #6	W	0
5	E5	Event #5	W	0
4	E4	Event #4	W	0
3	E3	Event #3	W	0
2	E2	Event #2	W	0
1	E1	Event #1	W	0
0	E0	Event #0	W	0

Table 14-461. Register Call Summary for Register TPCC_QSECR_Rn

IVA2.2 Subsystem Register Manual

- TPCC Register Mapping Summary: [0]

Table 14-462. TPCC_OPTm

Address Offset	0x4000 + (0x20*m)		
Physical address	0x01C0 4000 + (0x20*m)	**Instance**	IVA2.2 TPCC
Description	Options Parameter		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PRIV	Reserved							PRIVID				ITCCHEN	TCCHEN	ITCINTEN	TCINTEN	Reserved				TCC				TCCMODE	FWID			Reserved								STATIC	SYNCDIM	DAM	SAM

Bits	Field Name	Description	Type	Reset
31	PRIV	Privilege level: Privilege level (supervisor vs. user) for the host/cpu/dma that programmed this PaRAM Entry. Value is set with the vbus priv value when any part of the PaRAM Entry is written. Not writeable through vbus wdata bus. Is readable through VBus rdata bus. PRIV = 0: User level privilege PRIV = 1: Supervisor level privilege	R	-
30:27	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0x-
26:24	PRIVID	Privilege ID: Privilege ID for the external host/cpu/dma that programmed this PaRAM Entry. This value is set with the vbus privid value when any part of the PaRAM Entry is written. Not writeable through vbus wdata bus. Is readable through VBus rdata bus.	R	0x-
23	ITCCHEN	Intermediate transfer completion chaining enable: 0: Intermediate transfer complete chaining is disabled. 1: Intermediate transfer complete chaining is enabled.	RW	-
22	TCCHEN	Transfer complete chaining enable: 0: Transfer complete chaining is disabled. 1: Transfer complete chaining is enabled.	RW	-
21	ITCINTEN	Intermediate transfer completion interrupt enable: 0: Intermediate transfer complete interrupt is disabled. 1: Intermediate transfer complete interrupt is enabled (corresponding IER[TCC] bit must be set to 1 to generate interrupt)	RW	-
20	TCINTEN	Transfer complete interrupt enable: 0: Transfer complete interrupt is disabled. 1: Transfer complete interrupt is enabled (corresponding IER[TCC] bit must be set to 1 to generate interrupt)	RW	-
19:18	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0x-
17:12	TCC	Transfer Complete Code: The 6-bit code is used to set the relevant bit in CER (bit CER[TCC]) for chaining or in IER (bit IER[TCC]) for interrupts.	RW	0x--
11	TCCMODE	Transfer complete code mode: Indicates the point at which a transfer is considered completed. Applies to both chaining and interrupt. 0: Normal Completion, A transfer is considered completed after the transfer parameters are returned to the CC from the TC (which was returned from the peripheral). 1: Early Completion, A transfer is considered completed after the CC submits a TR to the TC. CC generates completion code internally .	RW	-
10:8	FWID	FIFO width: Applies if either SAM or DAM is set to FIFO mode. Pass-thru to TC. 0x0: FIFO width is 8-bit 0x1: FIFO width is 16-bit 0x2: FIFO width is 32-bit 0x3: FIFO width is 64-bit 0x4: FIFO width is 128-bit	RW	0x-

Bits	Field Name	Description	Type	Reset
		0x5: FIFO width is 256-bit		
7:4	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0x-
3	STATIC	Static Entry: 0: Entry is updated as normal 1: Entry is static, Count and Address updates are not updated after TRP is submitted. Linking is not performed.	RW	-
2	SYNCDIM	Transfer Synchronization Dimension: 0: A-Sync, Each event triggers the transfer of ACNT elements. 1: AB-Sync, Each event triggers the transfer of BCNT arrays of ACNT elements	RW	-
1	DAM	Destination Address Mode: Destination Address Mode within an array. Pass-thru to TC. 0: INCR, Dst addressing within an array increments. Dst is not a FIFO. 1: FIFO, Dst addressing within an array wraps around upon reaching FIFO width.	RW	-
0	SAM	Source Address Mode: Source Address Mode within an array. Pass-thru to TC. 0: INCR, Src addressing within an array increments. Source is not a FIFO. 1: FIFO, Src addressing within an array wraps around upon reaching FIFO width.	RW	-

Table 14-463. Register Call Summary for Register TPCC_OPTm

IVA2.2 Subsystem Functional Description
- EDMA: [0]

IVA2.2 Subsystem Register Manual
- TPCC Register Mapping Summary: [1]

Table 14-464. TPCC_SRCm

Address Offset	0x4004 + (0x20*m)		
Physical address	0x01C0 4004 + (0x20*m)4	**Instance**	IVA2.2 TPCC
Description	Source Address		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SRC																															

Bits	Field Name	Description	Type	Reset
31:0	SRC	Source Address: The 32-bit source address parameters specify the starting byte address of the source. If SAM is set to FIFO mode then the user should program the Source address to be aligned to the value specified by the OPT.FWID field. No errors are recognized here but TC will assert error if this is not true.	RW	0x--------

Table 14-465. Register Call Summary for Register TPCC_SRCm

IVA2.2 Subsystem Register Manual
- TPCC Register Mapping Summary: [0]

Table 14-466. TPCC_ABCNTm

Address Offset	0x4008 + (0x20*m)		
Physical address	0x01C0 4008 + (0x20*m)	**Instance**	IVA2.2 TPCC
Description	A and B byte count		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BCNT																ACNT															

Bits	Field Name	Description	Type	Reset
31:16	BCNT	BCNT : Count for 2nd Dimension: BCNT is a 16-bit unsigned value that specifies the number of arrays of length ACNT. For normal operation, valid values for BCNT can be anywhere between 1 and 65535. Therefore, the maximum Number of arrays in a frame is 65535 (64K-1 arrays). BCNT=1 means 1 array in the frame, and BCNT=0 means 0 arrays in the frame. In normal mode, a BCNT of 0 is considered as either a Null or Dummy transfer. A Dummy or Null transfer will generate a Completion code depending on the settings of the completion bit fields of the OPT field.	RW	0x----
15:0	ACNT	ACNT : number of bytes in 1st dimension: ACNT represents the number of bytes within the first dimension of a transfer. ACNT is a 16-bit unsigned value with valid values between 0 and 65535. Therefore, the maximum number of bytes in an array is 65535 bytes (64K-1 bytes). ACNT must be greater than or equal to 1 for a TR to be submitted to TC. An ACNT of 0 is considered as either a null or dummy transfer. A Dummy or Null transfer will generate a Completion code depending on the settings of the completion bit fields of the OPT field.	RW	0x----

Table 14-467. Register Call Summary for Register TPCC_ABCNTm

IVA2.2 Subsystem Register Manual

- TPCC Register Mapping Summary: [0]

Table 14-468. TPCC_DSTm

Address Offset	0x400C + (0x20*m)		
Physical address	0x01C0 400C + (0x20*m)	**Instance**	IVA2.2 TPCC
Description	Destination Address		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DST																															

Bits	Field Name	Description	Type	Reset
31:0	DST	Destination Address: The 32-bit destination address parameters specify the starting byte address of the destination. If DAM is set to FIFO mode then the user should program the Destination address to be aligned to the value specified by the OPTi.FWID field. No errors are recognized here but TC will assert error if this is not true.	RW	0x--------

Table 14-469. Register Call Summary for Register TPCC_DSTm

IVA2.2 Subsystem Register Manual

- TPCC Register Mapping Summary: [0]

Table 14-470. TPCC_BIDXm

Address Offset	0x4010 + (0x20*m)		
Physical address	0x01C0 4010 + (0x20*m)	**Instance**	IVA2.2 TPCC
Description			
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DBIDX																SBIDX															

Bits	Field Name	Description	Type	Reset
31:16	DBIDX	Destination 2nd Dimension Index: DBIDX is a 16-bit signed value (2s complement) used for destination address modification in between each array in the 2nd dimension. It is a signed value between -32768 and 32767. It provides a byte address offset from the beginning of the destination array to the beginning of the next destination array within the current frame. It applies to both A-Sync and AB-Sync transfers.	RW	0x----
15:0	SBIDX	Source 2nd Dimension Index: SBIDX is a 16-bit signed value (2s complement) used for source address modification in between each array in the 2nd dimension. It is a signed value between - 32768 and 32767. It provides a byte address offset from the beginning of the source array to the beginning of the next source array. It applies to both A-sync and AB-sync transfers.	RW	0x----

Table 14-471. Register Call Summary for Register TPCC_BIDXm

IVA2.2 Subsystem Register Manual

- TPCC Register Mapping Summary: [0]

Table 14-472. TPCC_LNKm

Address Offset	0x4014 + (0x20*m)		
Physical address	0x01C0 4014 + (0x20*m)	**Instance**	IVA2.2 TPCC
Description	Link and Reload parameters		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BCNTRLD																LINK															

Bits	Field Name	Description	Type	Reset
31:16	BCNTRLD	BCNT Reload: BCNTRLD is a 16-bit unsigned value used to reload the BCNT field once the last array in the 2nd dimension is transferred. This field is only used for A-Synced transfers. In this case, the CC decrements the BCNT value by one on each TR submission. When BCNT (conceptually) reaches zero, then the CC decrements CCNT and uses the BCNTRLD value to reinitialize the BCNT value. For AB-synchronized transfers, the CC submits the BCNT in the TR and therefore the TC is responsible to keep track of BCNT, not CC-thus BCNTRLD is a dont care field.	RW	0x----

Bits	Field Name	Description	Type	Reset
15:0	LINK	Link Address: The CC provides a mechanism to reload the current PaRAM Entry upon its natural termination (i.e., after count fields are decremented to 0) with a new PaRAM Entry. This is called linking. The 16-bit parameter LINK specifies the byte address offset in the PaRAM from which the CC loads/reloads the next PaRAM entry in the link. The CC should disregard the value in the upper 2 bits of the LINK field as well as the lower 5-bits of the LINK field. The upper two bits are ignored such that the user can program either the literal byte address of the LINK parameter or the PaRAM base-relative address of the link field. Therefore, if the user uses the literal address with a range from 0x4000 to 0x7FFF, it will be treated as a PaRAM-base-relative value of 0x0000 to 0x3FFF. The lower-5 bits are ignored and treated as b00000, thereby guaranteeing that all Link pointers point to a 32-byte aligned PaRAM entry. In the latter case (5-lsbs), behavior is undefined for the user (i.e., dont have to test it). In the former case (2 msbs), user should be able to take advantage of this feature (i.e., do have to test it). If a Link Update is requested to a PaRAM address that is beyond the actual range of implemented PaRAM, then the Link will be treated as a Null Linkand all 0s plus 0xFFFF will be written to the current entry location. A LINK value of 0xFFFF is referred to as a NULL link which should cause the CC to write 0x0 to all entries of the current PaRAM Entry except for the LINK field which is set to 0xFFFF. The Priv/Privid state is overwritten to 0x0 when linking. MSBs and LSBS should not be masked when comparing against the 0xFFFF value. I.e., a value of 0x3FFE is a non-NULL PaRAM link field.	RW	0x----

Table 14-473. Register Call Summary for Register TPCC_LNKm

IVA2.2 Subsystem Register Manual

- TPCC Register Mapping Summary: [0]

Table 14-474. TPCC_CIDXm

Address Offset	0x4018 + (0x20*m)		
Physical address	0x01C0 4018 + (0x20*m)	**Instance**	IVA2.2 TPCC
Description			
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DCIDX																SCIDX															

Bits	Field Name	Description	Type	Reset
31:16	DCIDX	Destination Frame Index: DCIDX is a 16-bit signed value (2s complement) used for destination address modification for the 3rd dimension. It is a signed value between -32768 and 32767. It provides a byte address offset from the beginning of the current array (pointed to by DST address) to the beginning of the first destination array in the next frame. It applies to both A-sync and AB-sync transfers. Note that when DCIDX is applied, the current array in anA-sync transfer is the last array in the frame, while the current array in a ABsync transfer is the first array in the frame.	RW	0x----

Bits	Field Name	Description	Type	Reset
15:0	SCIDX	Source Frame Index: SCIDX is a 16-bit signed value (2s complement) used for source address modification for the 3rd dimension. It is a signed value between -32768 and 32767. It provides a byte address offset from the beginning of the current array (pointed to by SRC address) to the beginning of the first source array in the next frame. It applies to both A-sync and AB-sync transfers. Note that when SCIDX is applied, the current arraya in an A-sync transfer is the last array in the frame, while the current array in a AB-sync transfer is the first array in the frame.	RW	0x----

Table 14-475. Register Call Summary for Register TPCC_CIDXm

IVA2.2 Subsystem Register Manual

- TPCC Register Mapping Summary: [0]

Table 14-476. TPCC_CCNTm

Address Offset	0x401C + (0x20*m)		
Physical address	0x01C0 401C + (0x20*m)	**Instance**	IVA2.2 TPCC
Description	C byte count		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																CCNT															

Bits	Field Name	Description	Type	Reset
31:16	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0x0000
15:0	CCNT	CCNT Count for 3rd Dimension: CCNT is a 16-bit unsigned value that specifies the number of frames in a block. Valid values for CCNT can be anywhere between 1 and 65535. Therefore, the maximum number of frames in a block is 65535 (64K-1 frames). CCNT of 1 means 1 frame in the block, and CCNT of 0 means 0 frames in the block. A CCNT value of 0 is considered as either a null or dummy transfer. A Dummy or Null transfer will generate a Completion code depending on the settings of the completion bit fields of the OPTi field.	RW	0x----

Table 14-477. Register Call Summary for Register TPCC_CCNTm

IVA2.2 Subsystem Register Manual

- TPCC Register Mapping Summary: [0]

14.5.6 TPTC0 and TPTC1 Registers

This section provides information about the TPTC0 and TPTC1 Modules. Each register in the Modules is described separately below.

14.5.6.1 TPTC0 and TPTC1 Register Mapping Summary

Table 14-478. TPTC0 and TPTC1 Register Summary

Register Name (j = 0 or 1)	Type	RegisterWidth (Bits)	Address Offset	TPTC0 Physical Address	TPTC1 Physical Address
TPTCj_PID	R	32	0x000	0x01C1 0000	0x01C1 0400
TPTCj_TCCFG	R	32	0x004	0x01C1 0004	0x01C1 0404
TPTCj_TCSTAT	R	32	0x100	0x01C1 0100	0x01C1 0500

Table 14-478. TPTC0 and TPTC1 Register Summary (continued)

Register Name (j = 0 or 1)	Type	RegisterWidth (Bits)	Address Offset	TPTC0 Physical Address	TPTC1 Physical Address
TPTCj_INTSTAT	R	32	0x104	0x01C1 0104	0x01C1 0504
TPTCj_INTEN	RW	32	0x108	0x01C1 0108	0x01C1 0508
TPTCj_INTCLR	W	32	0x10C	0x01C1 010C	0x01C1 050C
TPTCj_INTCMD	W	32	0x110	0x01C1 0110	0x01C1 0510
TPTCj_ERRSTAT	R	32	0x120	0x01C1 0120	0x01C1 0520
TPTCj_ERREN	RW	32	0x124	0x01C1 0124	0x01C1 0524
TPTCj_ERRCLR	W	32	0x128	0x01C1 0128	0x01C1 0528
TPTCj_ERRDET	R	32	0x12C	0x01C1 012C	0x01C1 052C
TPTCj_ERRCMD	W	32	0x130	0x01C1 0130	0x01C1 0530
TPTCj_RDRATE	RW	32	0x140	0x01C1 0140	0x01C1 0540
TPTCj_POPT	RW	32	0x200	0x01C1 0200	0x01C1 0600
TPTCj_PSRC	RW	32	0x204	0x01C1 0204	0x01C1 0604
TPTCj_PCNT	RW	32	0x208	0x01C1 0208	0x01C1 0608
TPTCj_PDST	RW	32	0x20C	0x01C1 020C	0x01C1 060C
TPTCj_PBIDX	RW	32	0x210	0x01C1 0210	0x01C1 0610
TPTCj_PMPPRXY	R	32	0x214	0x01C1 0214	0x01C1 0614
TPTCj_SAOPT	R	32	0x240	0x01C1 0240	0x01C1 0640
TPTCj_SASRC	R	32	0x244	0x01C1 0244	0x01C1 0644
TPTCj_SACNT	R	32	0x248	0x01C1 0248	0x01C1 0648
TPTCj_SADST	R	32	0x24C	0x01C1 024C	0x01C1 064C
TPTCj_SABIDX	R	32	0x250	0x01C1 0250	0x01C1 0650
TPTCj_SAMPPRXY	R	32	0x254	0x01C1 0254	0x01C1 0654
TPTCj_SACNTRLD	R	32	0x258	0x01C1 0258	0x01C1 0658
TPTCj_SASRCBREF	R	32	0x25C	0x01C1 025C	0x01C1 065C
TPTCj_SADSTBREF	R	32	0x260	0x01C1 0260	0x01C1 0660
TPTCj_DFCNTRLD	R	32	0x280	0x01C1 0280	0x01C1 0680
TPTCj_DFSRCBREF	R	32	0x284	0x01C1 0284	0x01C1 0684
TPTCj_DFDSTBREF	R	32	0x288	0x01C1 0288	0x01C1 0688
TPTCj_DFOPTi(1)	R	32	0x300 + (0x40*i)	0x01C1 0300 + (0x40*i)	0x01C1 0700 + (0x40*i)
TPTCj_DFSRCi(1)	R	32	0x304 + (0x40*i)	0x01C1 0304 + (0x40*i)	0x01C1 0704 + (0x40*i)
TPTCj_DFCNTi(1)	R	32	0x308 + (0x40*i)	0x01C1 0308 + (0x40*i)	0x01C1 0708 + (0x40*i)
TPTCj_DFDSTi(1)	R	32	0x30C + (0x40*i)	0x01C1 030C + (0x40*i)	0x01C1 070C + (0x40*i)
TPTCj_DFBIDXi(1)	R	32	0x310 + (0x40*i)	0x01C1 0310 + (0x40*i)	0x01C1 0710 + (0x40*i)
TPTCj_DFMPPRXYi(1)	R	32	0x314 + (0x40*i)	0x01C1 0314 + (0x40*i)	0x01C1 0714 + (0x40*i)

(1) i = 0 to 3 for TPTC0
i = 0 to 1 for TPTC1

14.5.6.2 TPTC0 and TPTC1 Register Descriptions

Table 14-479. TPTCj_PID

Address Offset	0x000		
Physical address	0x01C1 0000	**Instance**	IVA2.2 TPTC0
Physical address	0x01C1 0400	**Instance**	IVA2.2 TPTC1
Description	Peripheral ID Register		
Type	R		

31 30	29 28	27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10 9 8	7 6	5 4 3 2 1 0
SCHEME	Reserved	FUNC		RTL	MAJOR	CUSTOM	MINOR

Bits	Field Name	Description	Type	Reset
31:30	SCHEME	PID Scheme: Used to distinguish between old ID scheme and current. Spare bit to encode future schemes EDMA uses new scheme, indicated with value of 0x1.	RW	0x1
29:28	Reserved	Read returns 0.	R	0x0
27:16	FUNC	Function indicates a software compatible modulefamily.	RW	0x000
15:11	RTL	RTL Version	RW	0x--
10:8	MAJOR	Major Revision	RW	0x3
7:6	CUSTOM	Custom revision field: Not used on this version of EDMA.	RW	0x0
5:0	MINOR	Minor Revision	RW	0x--

Table 14-480. Register Call Summary for Register TPTCj_PID

IVA2.2 Subsystem Register Manual

- TPTC0 and TPTC1 Register Mapping Summary: [0]

Table 14-481. TPTCj_TCCFG

Address Offset	0x004		
Physical address	0x01C1 0004	**Instance**	IVA2.2 TPTC0
Physical address	0x01C1 0404	**Instance**	IVA2.2 TPTC1
Description	TC Configuration Register		
Type	R		

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10	9 8	7 6	5 4	3	2 1 0
Reserved			DREGDEPTH	Reserved	BUSWIDTH	Reserved	FIFOSIZE

Bits	Field Name	Description	Type	Reset
31:10	Reserved	Read returns 0.	R	0x000000
9:8	DREGDEPTH	Dst Register FIFO Depth Parameterization Read 0x0: 1 entry Read 0x1: 2 entries Read 0x2: 4 entries	R	0x-[(1)]
7:6	Reserved	Read returns 0.	R	0x0

(1) Depends on the hardware parameters of TPTC0 and TPTC1.

Bits	Field Name	Description	Type	Reset
5:4	BUSWIDTH	Bus Width Parameterization Read 0x0: 32-bit Read 0x1: 64-bit Read 0x2: 128-bit	R	0x1
3	Reserved	Read returns 0.	R	0
2:0	FIFOSIZE	Fifo Size Parameterization Read 0x0: 32 byte FIFO Read 0x1: 64 byte FIFO Read 0x2: 128 byte FIFO Read 0x3: 256 byte FIFO Read 0x4: 512 byte FIFO	R	0x-(1)

Table 14-482. Register Call Summary for Register TPTCj_TCCFG

IVA2.2 Subsystem Register Manual

- TPTC0 and TPTC1 Register Mapping Summary: [0]

Table 14-483. TPTCj_TCSTAT

Address Offset	0x100		
Physical address	0x01C1 0100	**Instance**	IVA2.2 TPTC0
Physical address	0x01C1 0500	**Instance**	IVA2.2 TPTC1
Description	TC Status Register		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																		DFSTRTPTR		Reserved				ACTV	Reserved			DSTACTV				Reserved		WSACTV	SRCACTV	PROGBUSY

Bits	Field Name	Description	Type	Reset
31:14	Reserved	Read returns 0.	R	0x00000
13:12	DFSTRTPTR	Dst FIFO Start Pointer Represents the offset to the head entry of Dst Register FIFO, in units of *entries*. Legal values = 0x0 to 0x3	R	0x0
11:9	Reserved	Read returns 0.	R	0x0
8	ACTV	Channel Active Channel Active is a logical-OR of each of the *BUSY/ACTV signals. The ACTV bit must remain high through the life of a TR. ACTV = 0: Channel is idle. ACTV = 1: Channel is busy.	R	1
7	Reserved	Read returns 0.	R	0
6:4	DSTACTV	Destination Active State Specifies the number of TRs that are resident in the Dst Register FIFO at a given instant. Legal values are constrained by the DSTREGDEPTH parameter. Read 0x0: FIFO set is empty. Read 0x1: Dst FIFO contains 1 TR Read 0x2: Dst FIFO contains 2 TR Read 0x3: Dst FIFO contains 3 TR Read 0x4: Dst FIFO contains 4 TR	R	0x0
3	Reserved	Read returns 0.	R	0

Bits	Field Name	Description	Type	Reset
2	WSACTV	Write Status Active WSACTV = 0: Write status is not pending. Write status has been received for all previously issued write commands. WSACTV = 1: Write Status is pending. Write status has not been received for all previously issued write commands.	R	0
1	SRCACTV	Source Active State SRCACTV = 0: Source Active set is idle. Any TR written to Prog Set will immediately transition to Source Active set as long as the Dst FIFO Set is not full (DSTFULL == 1). SRCACTV = 1: Source Active set is busy either performing read transfers or waiting to perform read transfers for current Transfer Request.	R	0
0	PROGBUSY	Program Register Set Busy PROGBUSY = 0: Prog set idle and is available for programming. PROGBUSY = 1: Prog set busy. User should poll for PROGBUSY equal to 0 prior to re-programming the Program Register set.	R	0

Table 14-484. Register Call Summary for Register TPTCj_TCSTAT

IVA2.2 Subsystem Functional Description
- EDMA: [0]

IVA2.2 Subsystem Register Manual
- TPTC0 and TPTC1 Register Mapping Summary: [1]

Table 14-485. TPTCj_INTSTAT

Address Offset	0x104		
Physical address	0x01C1 0104	**Instance**	IVA2.2 TPTC0
Physical address	0x01C1 0504	**Instance**	IVA2.2 TPTC1
Description	Interrupt Status Register		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																														TRDONE	PROGEMPTY

Bits	Field Name	Description	Type	Reset
31:2	Reserved	Read returns 0.	R	0x00000000
1	TRDONE	TR Done Event Status: TRDONE = 0: Condition not detected. TRDONE = 1: Set when TC has completed a Transfer Request. TRDONE should be set when the write status is returned for the final write of a TR. Cleared when user writes 1 to INTCLR.TRDONE register bit.	R	0
0	PROGEMPTY	Program Set Empty Event Status: PROGEMPTY = 0: Condition not detected. PROGEMPTY = 1: Set when Program Register set transitions to empty state. Cleared when user writes 1 to INTCLR.PROGEMPTY register bit.	R	0

Table 14-486. Register Call Summary for Register TPTCj_INTSTAT

IVA2.2 Subsystem Functional Description
- EDMA: [0] [1] [2]

IVA2.2 Subsystem Basic Programming Model
- Error Reporting for EDMA Module: [3]

Table 14-486. Register Call Summary for Register TPTCj_INTSTAT (continued)

IVA2.2 Subsystem Register Manual • TPTC0 and TPTC1 Register Mapping Summary: [4]

Table 14-487. TPTCj_INTEN

Address Offset	0x108		
Physical address	0x01C1 0108	**Instance**	IVA2.2 TPTC0
Physical address	0x01C1 0508	**Instance**	IVA2.2 TPTC1
Description	Interrupt Enable Register		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																														TRDONE	PROGEMPTY

Bits	Field Name	Description	Type	Reset
31:2	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0x00000000
1	TRDONE	TR Done Event Enable: INTEN.TRDONE = 0: TRDONE Event is disabled. INTEN.TRDONE = 1: TRDONE Event is enabled,and contributes to interrupt generation	RW	0
0	PROGEMPTY	Program Set Empty Event Enable: INTEN.PROGEMPTY = 0: PROGEMPTY Event isdisabled. INTEN.PROGEMPTY = 1: PROGEMPTY Event isenabled, and contributes to interrupt generation	RW	0

Table 14-488. Register Call Summary for Register TPTCj_INTEN

IVA2.2 Subsystem Functional Description • EDMA: [0]
IVA2.2 Subsystem Register Manual • TPTC0 and TPTC1 Register Mapping Summary: [1]

Table 14-489. TPTCj_INTCLR

Address Offset	0x10C		
Physical address	0x01C1 010C	**Instance**	IVA2.2 TPTC0
Physical address	0x01C1 050C	**Instance**	IVA2.2 TPTC1
Description	Interrupt Clear Register		
Type	W		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																														TRDONE	PROGEMPTY

Bits	Field Name	Description	Type	Reset
31:2	Reserved	Write 0s for future compatibility.	W	0x00000000
1	TRDONE	TR Done Event Clear: INTCLR.TRDONE = 0: Writes of 0 have no effect. INTCLR.TRDONE = 1: Write of 1 clears INTSTAT.TRDONE bit	W	0
0	PROGEMPTY	Program Set Empty Event Clear: INTCLR.PROGEMPTY = 0: Writes of 0 have no effect. INTCLR.PROGEMPTY = 1: Write of 1 clears INTSTAT.PROGEMPTY bit	W	0

Table 14-490. Register Call Summary for Register TPTCj_INTCLR

IVA2.2 Subsystem Register Manual

- TPTC0 and TPTC1 Register Mapping Summary: [0]

Table 14-491. TPTCj_INTCMD

Address Offset	0x110		
Physical address	0x01C1 0110	**Instance**	IVA2.2 TPTC0
Physical address	0x01C1 0510	**Instance**	IVA2.2 TPTC1
Description	Interrupt Command Register		
Type	W		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																														SET	EVAL

Bits	Field Name	Description	Type	Reset
31:2	Reserved	Write 0s for future compatibility.	W	0x00000000
1	SET	Set TPTC interrupt: Write of 1 to SET causes TPTC interrupt to be pulsed unconditionally. Writes of 0 have no effect.	W	0
0	EVAL	Evaluate state of TPTC interrupt Write of 1 to EVAL causes TPTC interrupt to be pulsed if any of the INTSTAT bits are set to 1. Writes of 0 have no effect.	W	0

Table 14-492. Register Call Summary for Register TPTCj_INTCMD

IVA2.2 Subsystem Register Manual

- TPTC0 and TPTC1 Register Mapping Summary: [0]

Table 14-493. TPTCj_ERRSTAT

Address Offset	0x120		
Physical address	0x01C1 0120	**Instance**	IVA2.2 TPTC0
Physical address	0x01C1 0520	**Instance**	IVA2.2 TPTC1
Description	Error Status Register		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																												MMRAERR	TRERR	Reserved	BUSERR

Bits	Field Name	Description	Type	Reset
31:4	Reserved	Read returns 0.	R	0x0000000
3	MMRAERR	MMR Address Error: MMRAERR = 0: Condition not detected. MMRAERR = 1: User attempted to read or write to invalid address configuration memory map. (Is only be set for non-emulation accesses). No additional error information is recorded.	R	0
2	TRERR	TR Error: TR detected that violates FIFO Mode transfer (SAM or DAM is 1) alignment rules or has ACNT or BCNT == 0. No additional error information is recorded.	R	0
1	Reserved	Read returns 0.	R	0
0	BUSERR	Bus Error Event: BUSERR = 0: Condition not detected. BUSERR = 1: TC has detected an error code on the write response bus or read response bus. Error information is stored in Error Details Register (ERRDET).	R	0

Table 14-494. Register Call Summary for Register TPTCj_ERRSTAT

IVA2.2 Subsystem Basic Programming Model

- Error Reporting for EDMA Module: [0] [1] [2] [3] [4] [5]

IVA2.2 Subsystem Register Manual

- TPTC0 and TPTC1 Register Mapping Summary: [6]

Table 14-495. TPTCj_ERREN

Address Offset	0x124		
Physical address	0x01C1 0124	**Instance**	IVA2.2 TPTC0
Physical address	0x01C1 0524	**Instance**	IVA2.2 TPTC1
Description	Error Enable Register		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																												MMRAERR	TRERR	Reserved	BUSERR

Bits	Field Name	Description	Type	Reset
31:4	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0x0000000
3	MMRAERR	Interrupt enable for ERRSTAT.MMRAERR: ERREN.MMRAERR = 0: BUSERR is disabled. ERREN.MMRAERR = 1: MMRAERR is enabled, and contributes to the TPTC error interrupt generation.	RW	0
2	TRERR	Interrupt enable for ERRSTAT.TRERR: ERREN.TRERR = 0: BUSERR is disabled. ERREN.TRERR = 1: TRERR is enabled, andcontributes to the TPTC error interrupt generation.	RW	0
1	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0
0	BUSERR	Interrupt enable for ERRSTAT.BUSERR: ERREN.BUSERR = 0: BUSERR is disabled. ERREN.BUSERR = 1: BUSERR is enabled, andcontributes to the TPTC error interrupt generation.	RW	0

Table 14-496. Register Call Summary for Register TPTCj_ERREN

IVA2.2 Subsystem Basic Programming Model

- Error Reporting for EDMA Module: [0]

IVA2.2 Subsystem Register Manual

- TPTC0 and TPTC1 Register Mapping Summary: [1]

Table 14-497. TPTCj_ERRCLR

Address Offset	0x128		
Physical address	0x01C1 0128	**Instance**	IVA2.2 TPTC0
Physical address	0x01C1 0528	**Instance**	IVA2.2 TPTC1
Description	Error Clear Register		
Type	W		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																												MMRAERR	TRERR	Reserved	BUSERR

Bits	Field Name	Description	Type	Reset
31:4	Reserved	Write 0s for future compatibility.	W	0x0000000
3	MMRAERR	Interrupt clear for ERRSTAT.MMRAERR: ERRCLR.MMRAERR = 0: Writes of 0 have no effect. ERRCLR.MMRAERR = 1: Write of 1 clearsERRSTAT.MMRAERR bit. Write of 1 to ERRCLR.MMRAERR does not clear the ERRDET register.	W	0
2	TRERR	Interrupt clear for ERRSTAT.TRERR: ERRCLR.TRERR = 0: Writes of 0 have no effect. ERRCLR.TRERR = 1: Write of 1 clears ERRSTAT.TRERR bit. Write of 1 to ERRCLR.TRERR does not clear the ERRDET register.	W	0
1	Reserved	Write 0s for future compatibility.	W	0
0	BUSERR	Interrupt clear for ERRSTAT.BUSERR: ERRCLR.BUSERR = 0: Writes of 0 have no effect. ERRCLR.BUSERR = 1: Write of 1 clearsERRSTAT.BUSERR bit. Write of 1 to ERRCLR.BUSERR clears the ERRDET register.	W	0

Table 14-498. Register Call Summary for Register TPTCj_ERRCLR

IVA2.2 Subsystem Basic Programming Model

- Error Reporting for EDMA Module: [0]

IVA2.2 Subsystem Register Manual

- TPTC0 and TPTC1 Register Mapping Summary: [1]

Table 14-499. TPTCj_ERRDET

Address Offset	0x12C		
Physical address	0x01C1 012C	**Instance**	IVA2.2 TPTC0
Physical address	0x01C1 052C	**Instance**	IVA2.2 TPTC1
Description	Error Details Register		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														TCCHEN	TCINTEN	Reserved		TCC						Reserved				STAT			

Bits	Field Name	Description	Type	Reset
31:18	Reserved	Read returns 0.	R	0x0000
17	TCCHEN	Contains the OPT.TCCHEN value programmed by the user for the Read or Write transaction that resulted in an error.	R	0
16	TCINTEN	Contains the OPT.TCINTEN value programmed by the user for the Read or Write transaction that resulted in an error.	R	0
15:14	Reserved	Read returns 0.	R	0x0
13:8	TCC	Transfer Complete Code: Contains the OPT.TCC value programmed by the user for the Read or Write transaction that resulted in an error.	R	0x00
7:4	Reserved	Read returns 0.	R	0x0
3:0	STAT	Transaction Status: Stores the non-zero status/error code that was detected on the read status or write status bus. MS-bit effectively serves as the read vs. write error code. If read status and write status are returned on the same cycle, then the TC chooses non-zero version. If both are non-zero then write status is treated as higherpriority. Encoding of errors matches the CBA spec and issummarized here: 0xF = Read 0x0: No Error (should not cause error to be latched) Read 0x1: Read Addressing error Read 0x2: Read Privilege error Read 0x3: Read Timeout error Read 0x4: Read Data error Read 0x7: Read Exclusive-operation failure Read 0x8: No Error (should not cause error to be latched) Read 0x9: Write Addressing error Read 0xA: Write Privilege error Read 0xB: Write Timeout error Read 0xC: Write Data error Read 0xF: Write Exclusive-operation failure	R	0x0

Table 14-500. Register Call Summary for Register TPTCj_ERRDET

IVA2.2 Subsystem Basic Programming Model

- Error Reporting for EDMA Module: [0] [1] [2] [3] [4]

IVA2.2 Subsystem Register Manual

- TPTC0 and TPTC1 Register Mapping Summary: [5]

Table 14-501. TPTCj_ERRCMD

Address Offset	0x130		
Physical address	0x01C1 0130	**Instance**	IVA2.2 TPTC0
Physical address	0x01C1 0530	**Instance**	IVA2.2 TPTC1
Description	Error Command Register		
Type	W		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																														SET	EVAL

Bits	Field Name	Description	Type	Reset
31:2	Reserved	Write 0s for future compatibility.	W	0x00000000
1	SET	Set TPTC error interrupt: Write of 1 to SET causes TPTC error interrupt to be pulsed unconditionally. Writes of 0 have no effect.	W	0
0	EVAL	Evaluate state of TPTC error interrupt Write of 1 to EVAL causes TPTC error interrupt to be pulsed if any of the ERRSTAT bits are set to 1. Writes of 0 have no effect.	W	0

Table 14-502. Register Call Summary for Register TPTCj_ERRCMD

IVA2.2 Subsystem Register Manual

- TPTC0 and TPTC1 Register Mapping Summary: [0]

Table 14-503. TPTCj_RDRATE

Address Offset	0x140		
Physical address	0x01C1 0140	**Instance**	IVA2.2 TPTC0
Physical address	0x01C1 0540	**Instance**	IVA2.2 TPTC1
Description	Read Rate Register		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																														RDRATE	

Bits	Field Name	Description	Type	Reset
31:3	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0x00000000
2:0	RDRATE	Read Rate Control: Controls the number of cycles between read commands. This is a global setting that applies to all TRs for this TC. 0x0: Reads issued as fast as possible. 0x1: 4 cycles between reads 0x2: 8 cycles between reads 0x3: 16 cycles between reads 0x4: 32 cycles between reads	RW	0x0

Table 14-504. Register Call Summary for Register TPTCj_RDRATE

IVA2.2 Subsystem Register Manual

- TPTC0 and TPTC1 Register Mapping Summary: [0]

Table 14-505. TPTCj_POPT

Address Offset	0x200		
Physical address	0x01C1 0200	**Instance**	IVA2.2 TPTC0
Physical address	0x01C1 0600	**Instance**	IVA2.2 TPTC1
Description	Prog Set Options		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								TCCHEN	Reserved	TCINTEN	Reserved					TCC				Reserved	FWID			Reserved	PRI			Reserved		DAM	SAM

Bits	Field Name	Description	Type	Reset
31:23	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0x000
22	TCCHEN	Transfer complete chaining enable: 0: Transfer complete chaining is disabled. 1: Transfer complete chaining is enabled.	RW	0
21	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0
20	TCINTEN	Transfer complete interrupt enable: 0: Transfer complete interrupt is disabled. 1: Transfer complete interrupt is enabled.	RW	0
19:18	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0x0
17:12	TCC	Transfer Complete Code: The 6-bit code is used to set the relevant bit in CER or IPR of the TPCC module.	RW	0x00
11	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0
10:8	FWID	FIFO width control: Applies if either SAM or DAM is set to FIFO mode. 0x0: FIFO width is 8-bit 0x1: FIFO width is 16-bit 0x2: FIFO width is 32-bit 0x3: FIFO width is 64-bit 0x4: FIFO width is 128-bit 0x5: FIFO width is 256-bit	RW	0x0
7	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0
6:4	PRI	Transfer Priority: 0: Priority 0 - Highest priority 1: Priority 1 ... 7: Priority 7 - Lowest priority 0x0: Priority 0 - Highest priority 0x1: Priority 1 0x2: Priority 2 0x3: Priority 3 0x4: Priority 4 0x5: Priority 5 0x6: Priority 6 0x7: Priority 7 - Lowest Priority	RW	0x0
3:2	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0x0

Bits	Field Name	Description	Type	Reset
1	DAM	Destination Address Mode within an array: 0: INCR, Dst addressing within an array increments. 1: FIFO, Dst addressing within an array wraps around upon reaching FIFO width.	RW	0
0	SAM	Source Address Mode within an array: 0: INCR, Src addressing within an array increments. 1: FIFO, Src addressing within an array wraps around upon reaching FIFO width.	RW	0

Table 14-506. Register Call Summary for Register TPTCj_POPT

IVA2.2 Subsystem Functional Description
- EDMA: [0]

IVA2.2 Subsystem Register Manual
- TPTC0 and TPTC1 Register Mapping Summary: [1]

Table 14-507. TPTCj_PSRC

Address Offset	0x204		
Physical address	0x01C1 0204	**Instance**	IVA2.2 TPTC0
Physical address	0x01C1 0604	**Instance**	IVA2.2 TPTC1
Description	Prog Set Src Address		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SADDR																															

Bits	Field Name	Description	Type	Reset
31:0	SADDR	Source address for Program Register Set	RW	0x00000000

Table 14-508. Register Call Summary for Register TPTCj_PSRC

IVA2.2 Subsystem Functional Description
- EDMA: [0]

IVA2.2 Subsystem Register Manual
- TPTC0 and TPTC1 Register Mapping Summary: [1]

Table 14-509. TPTCj_PCNT

Address Offset	0x208		
Physical address	0x01C1 0208	**Instance**	IVA2.2 TPTC0
Physical address	0x01C1 0608	**Instance**	IVA2.2 TPTC1
Description	Prog Set Count		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BCNT																ACNT															

Bits	Field Name	Description	Type	Reset
31:16	BCNT	B-Dimension count. Number of arrays to be transferred, where each array is ACNT in length.	RW	0x0000
15:0	ACNT	A-Dimension count. Number of bytes to be transferred in first dimension.	RW	0x0000

Table 14-510. Register Call Summary for Register TPTCj_PCNT

IVA2.2 Subsystem Functional Description

- EDMA: [0]

IVA2.2 Subsystem Register Manual

- TPTC0 and TPTC1 Register Mapping Summary: [1]

Table 14-511. TPTCj_PDST

Address Offset	0x20C		
Physical address	0x01C1 020C	**Instance**	IVA2.2 TPTC0
Physical address	0x01C1 060C	**Instance**	IVA2.2 TPTC1
Description	Prog Set Dst Address		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DADDR																															

Bits	Field Name	Description	Type	Reset
31:0	DADDR	Destination address for Program Register Set	RW	0x00000000

Table 14-512. Register Call Summary for Register TPTCj_PDST

IVA2.2 Subsystem Functional Description

- EDMA: [0]

IVA2.2 Subsystem Register Manual

- TPTC0 and TPTC1 Register Mapping Summary: [1]

Table 14-513. TPTCj_PBIDX

Address Offset	0x210		
Physical address	0x01C1 0210	**Instance**	IVA2.2 TPTC0
Physical address	0x01C1 0610	**Instance**	IVA2.2 TPTC1
Description	Prog Set B-Dim Idx		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DBIDX																SBIDX															

Bits	Field Name	Description	Type	Reset
31:16	DBIDX	Dest B-Idx for Program Register Set: B-Idx offset between Destination arrays: Represents the offset in bytes between the starting address of each destination array (recall that there are BCNT arrays of ACNT elements). DBIDX is always used, regardless of whether DAM is Increment or FIFO mode.	RW	0x0000
15:0	SBIDX	Source B-Idx for Program Register Set: B-Idx offset between Source arrays: Represents the offset in bytes between the starting address of each source array (recall that there are BCNT arrays of ACNT elements). SBIDX is always used, regardless of whether SAM is Increment or FIFO mode.	RW	0x0000

Table 14-514. Register Call Summary for Register TPTCj_PBIDX

IVA2.2 Subsystem Functional Description

- EDMA: [0]

IVA2.2 Subsystem Register Manual

- TPTC0 and TPTC1 Register Mapping Summary: [1]

Table 14-515. TPTCj_PMPPRXY

Address Offset	0x214		
Physical address	0x01C1 0214	**Instance**	IVA2.2 TPTC0
Physical address	0x01C1 0614	**Instance**	IVA2.2 TPTC1
Description	Prog Set Mem Protect Proxy		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																							PRIV	Reserved				PRIVID			

Bits	Field Name	Description	Type	Reset
31:9	Reserved	Read returns 0.	R	0x000000
8	PRIV	Privilege Level: PRIV = 0: User level privilege PRIV = 1: Supervisor level privilege PMPPRXY.PRIV is always updated with the value from the configuration bus Privilege field on any/every write to Program Set BIDX Register (trigger register). The PRIV value for the SA Set and DF Set are copied from the value in the Program set along with the remainder of the parameter values. The privilege ID is issued on the VBusM read and write command bus such that the target endpoints can perform memory protection checks based on the PRIV of the external host that sets up the DMA transaction.	R	0
7:4	Reserved	Read returns 0.	R	0x0
3:0	PRIVID	Privilege ID: PMPPRXY.PRIVID is always updated with the value from configuration bus Privilege ID field on any/every write to Program Set BIDX Register (trigger register). The PRIVID value for the SA Set and DF Set are copied from the value in the Program set along with the remainder of the parameter values. The privilege ID is issued on the VBusM read and write command bus such that the target endpoints can perform memory protection checks based on the privid of the external host that sets up the DMA transaction.	R	0x0

Table 14-516. Register Call Summary for Register TPTCj_PMPPRXY

IVA2.2 Subsystem Functional Description

- EDMA: [0]

IVA2.2 Subsystem Register Manual

- TPTC0 and TPTC1 Register Mapping Summary: [1]

Table 14-517. TPTCj_SAOPT

Address Offset	0x240		
Physical address	0x01C1 0240	**Instance**	IVA2.2 TPTC0
Physical address	0x01C1 0640	**Instance**	IVA2.2 TPTC1
Description	Src Actv Set Options		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								TCCHEN	Reserved		TCINTEN	Reserved				TCC				Reserved		FWID		Reserved		PRI		Reserved			DAM	SAM

Bits	Field Name	Description	Type	Reset
31:23	Reserved	Read returns 0.	R	0x000
22	TCCHEN	Transfer complete chaining enable: 0: Transfer complete chaining is disabled. 1: Transfer complete chaining is enabled.	R	0
21	Reserved	Read returns 0.	R	0
20	TCINTEN	Transfer complete interrupt enable: 0: Transfer complete interrupt is disabled. 1: Transfer complete interrupt is enabled.	R	0
19:18	Reserved	Read returns 0.	R	0x0
17:12	TCC	Transfer Complete Code: The 6-bit code is used to set the relevant bit in CER or IPR of the TPCC module.	R	0x00
11	Reserved	Read returns 0.	R	0
10:8	FWID	FIFO width control: Applies if either SAM or DAM is set to FIFO mode. Read 0x0: FIFO width is 8-bit Read 0x1: FIFO width is 16-bit Read 0x2: FIFO width is 32-bit Read 0x3: FIFO width is 64-bit Read 0x4: FIFO width is 128-bit Read 0x5: FIFO width is 256-bit	R	0x0
7	Reserved	Read returns 0.	R	0
6:4	PRI	Transfer Priority: 0: Priority 0 - Highest priority 1: Priority 1 ... 7: Priority 7 - Lowest priority Read 0x0: Priority 0 - Highest priority Read 0x1: Priority 1 Read 0x2: Priority 2 Read 0x3: Priority 3 Read 0x4: Priority 4 Read 0x5: Priority 5 Read 0x6: Priority 6 Read 0x7: Priority 7 - Lowest Priority	R	0x0
3:2	Reserved	Read returns 0.	R	0x0
1	DAM	Destination Address Mode within an array: 0: INCR, Dst addressing within an array increments. 1: FIFO, Dst addressing within an array wraps around upon reaching FIFO width.	R	0
0	SAM	Source Address Mode within an array: 0: INCR, Src addressing within an array increments. 1: FIFO, Src addressing within an array wraps around upon reaching FIFO width.	R	0

Table 14-518. Register Call Summary for Register TPTCj_SAOPT

IVA2.2 Subsystem Functional Description • EDMA: [0]
IVA2.2 Subsystem Register Manual • TPTC0 and TPTC1 Register Mapping Summary: [1]

Table 14-519. TPTCj_SASRC

Address Offset	0x244		
Physical address	0x01C1 0244	**Instance**	IVA2.2 TPTC0
Physical address	0x01C1 0644	**Instance**	IVA2.2 TPTC1
Description	Src Actv Set Src Address		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SADDR																															

Bits	Field Name	Description	Type	Reset
31:0	SADDR	Source address for Source Active Register Set: Initial value is copied from PSRC.SADDR. TC updates value according to source addressing mode (OPT.SAM) and/or source index value (BIDX.SBIDX) after each read command is issued. When a TR is complete, the final value should be the address of the last read command issued.	R	0x00000000

Table 14-520. Register Call Summary for Register TPTCj_SASRC

IVA2.2 Subsystem Functional Description

- EDMA: [0]

IVA2.2 Subsystem Register Manual

- TPTC0 and TPTC1 Register Mapping Summary: [1]

Table 14-521. TPTCj_SACNT

Address Offset	0x248		
Physical address	0x01C1 0248	**Instance**	IVA2.2 TPTC0
Description	Src Actv Set Count		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BCNT																ACNT															

Bits	Field Name	Description	Type	Reset
31:16	BCNT	B-Dimension count: Number of arrays to be transferred, where each array is ACNT in length. Count Remaining for Src Active Register Set. Represents the amount of data remaining to be read. Initial value is copied from PCNT. TC decrements ACNT and BCNT as necessary after each read command isissued. Final value should be 0 when TR is complete.	R	0x0000
15:0	ACNT	A-Dimension count: Number of bytes to be transferred in first dimension. Count Remaining for Src Active Register Set. Represents the amount of data remaining to be read. Initial value is copied from PCNT. TC decrements ACNT and BCNT as necessary after each read command isissued. Final value should be 0 when TR is complete.	R	0x0000

Table 14-522. Register Call Summary for Register TPTCj_SACNT

IVA2.2 Subsystem Functional Description

- EDMA: [0]

Table 14-522. Register Call Summary for Register TPTCj_SACNT (continued)

IVA2.2 Subsystem Register Manual

- TPTC0 and TPTC1 Register Mapping Summary: [1]

Table 14-523. TPTCj_SADST

Address Offset	0x24C		
Physical address	0x01C1 024C	**Instance**	IVA2.2 TPTC0
Physical address	0x01C1 064C	**Instance**	IVA2.2 TPTC1
Description	rsvd, return 0x0 w/o AERROR		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DADDR																															

Bits	Field Name	Description	Type	Reset
31:0	DADDR	Destination address is not applicable for Src Active Register Set. Reads return 0x0	R	0x00000000

Table 14-524. Register Call Summary for Register TPTCj_SADST

IVA2.2 Subsystem Functional Description

- EDMA: [0]

IVA2.2 Subsystem Register Manual

- TPTC0 and TPTC1 Register Mapping Summary: [1]

Table 14-525. TPTCj_SABIDX

Address Offset	0x250		
Physical address	0x01C1 0250	**Instance**	IVA2.2 TPTC0
Physical address	0x01C1 0650	**Instance**	IVA2.2 TPTC1
Description	Src Actv Set B-Dim Idx		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DBIDX																SBIDX															

Bits	Field Name	Description	Type	Reset
31:16	DBIDX	Dest B-Idx for Source Active Register Set. Value copied from PBIDX: B-Idx offset between Destination arrays: Represents the offset in bytes between the startingaddress of each destination array (recall that there are BCNT arrays of ACNT elements). DBIDX is always used, regardless of whether DAM is Increment or FIFO mode.	R	0x0000
15:0	SBIDX	Source B-Idx for Source Active Register Set. Value copied from PBIDX: B-Idx offset between Source arrays: Represents the offset in bytes between the startingaddress of each source array (recall that there are BCNT arrays of ACNT elements). SBIDX is always used, regardless of whether SAM is Increment or FIFO mode.	R	0x0000

Table 14-526. Register Call Summary for Register TPTCj_SABIDX

IVA2.2 Subsystem Functional Description

- EDMA: [0]

IVA2.2 Subsystem Register Manual

- TPTC0 and TPTC1 Register Mapping Summary: [1]

Table 14-527. TPTCj_SAMPPRXY

Address Offset	0x254		
Physical address	0x01C1 0254	**Instance**	IVA2.2 TPTC0
Physical address	0x01C1 0654	**Instance**	IVA2.2 TPTC1
Description	Src Actv Set Mem Protect Proxy		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																							PRIV	Reserved						PRIVID	

Bits	Field Name	Description	Type	Reset
31:9	Reserved	Read returns 0.	R	0x000000
8	PRIV	Privilege Level: PRIV = 0: User level privilege PRIV = 1: Supervisor level privilege SAMPPRXY.PRIV is always updated with the value from the configuration bus Privilege field on any/every write to Program Set BIDX Register (trigger register). The PRIV value for the SA Set and DF Set are copied from the value in the Program set along with the remainder of the parameter values. The privilege ID is issued on the VBusM read and write command bus such that the target endpoints can perform memory protection checks based on the PRIV of the external host that sets up the DMA transaction.	R	0
7:4	Reserved	Read returns 0.	R	0x0
3:0	PRIVID	Privilege ID: SAMPPRXY.PRIVID is always updated with the value from configuration bus Privilege ID field on any/every write to Program Set BIDX Register (trigger register). The PRIVID value for the SA Set and DF Set are copied from the value in the Program set along with the remainder of the parameter values. The privilege ID is issued on the VBusM read and write command bus such that the target endpoints can perform memory protection checks based on the privid of the external host that sets up the DMA transaction.	R	0x0

Table 14-528. Register Call Summary for Register TPTCj_SAMPPRXY

IVA2.2 Subsystem Functional Description

- EDMA: [0]

IVA2.2 Subsystem Register Manual

- TPTC0 and TPTC1 Register Mapping Summary: [1]

Table 14-529. TPTCj_SACNTRLD

Address Offset	0x258		
Physical address	0x01C1 0258	**Instance**	IVA2.2 TPTC0
Physical address	0x01C1 0658	**Instance**	IVA2.2 TPTC1
Description	Src Actv Set Cnt Reload		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																ACNTRLD															

Bits	Field Name	Description	Type	Reset
31:16	Reserved	Read returns 0.	R	0x0000
15:0	ACNTRLD	A-Cnt Reload value for Source Active Register set. Value copied from PCNT.ACNT: Represents the originally programmed value of ACNT. The Reload value is used to reinitialize ACNT after each array is serviced (i.e., ACNT decrements to 0). by the Src offset in bytes between the starting address of each source array (recall that there are BCNT arrays of ACNT bytes)	R	0x0000

Table 14-530. Register Call Summary for Register TPTCj_SACNTRLD

IVA2.2 Subsystem Functional Description

- EDMA: [0]

IVA2.2 Subsystem Register Manual

- TPTC0 and TPTC1 Register Mapping Summary: [1]

Table 14-531. TPTCj_SASRCBREF

Address Offset	0x25C		
Physical address	0x01C1 025C	**Instance**	IVA2.2 TPTC0
Physical address	0x01C1 065C	**Instance**	IVA2.2 TPTC1
Description	Src Actv Set Src Addr A-Reference		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SADDRBREF																															

Bits	Field Name	Description	Type	Reset
31:0	SADDRBREF	Source address reference for Source Active Register Set: Represents the starting address for the array currently being read. The next arrays starting address is calculated as the reference address plus the source b-idx value.	R	0x00000000

Table 14-532. Register Call Summary for Register TPTCj_SASRCBREF

IVA2.2 Subsystem Functional Description

- EDMA: [0]

IVA2.2 Subsystem Register Manual

- TPTC0 and TPTC1 Register Mapping Summary: [1]

Table 14-533. TPTCj_SADSTBREF

Address Offset	0x260		
Physical address	0x01C1 0260	**Instance**	IVA2.2 TPTC0
Physical address	0x01C1 0660	**Instance**	IVA2.2 TPTC1
Description	rsvd, return 0x0 w/o AERROR		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DADDRBREF																															

Bits	Field Name	Description	Type	Reset
31:0	DADDRBREF	Dst address reference is not applicable for Src Active Register Set. Reads return 0x0.	R	0x00000000

Table 14-534. Register Call Summary for Register TPTCj_SADSTBREF

IVA2.2 Subsystem Functional Description

- EDMA: [0]

IVA2.2 Subsystem Register Manual

- TPTC0 and TPTC1 Register Mapping Summary: [1]

Table 14-535. TPTCj_DFCNTRLD

Address Offset	0x280		
Physical address	0x01C1 0280	**Instance**	IVA2.2 TPTC0
Physical address	0x01C1 0680	**Instance**	IVA2.2 TPTC1
Description	Dst FIFO Set Cnt Reload		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																ACNTRLD															

Bits	Field Name	Description	Type	Reset
31:16	Reserved	Read returns 0.	R	0x0000
15:0	ACNTRLD	A-Cnt Reload value for Destination FIFO Register set. Value copied from PCNT.ACNT: Represents the originally programmed value of ACNT. The Reload value is used to reinitialize ACNT after each array is serviced (i.e., ACNT decrements to 0). by the Src offset in bytes between the starting address of each source array (recall that there are BCNT arrays of ACNT bytes)	R	0x0000

Table 14-536. Register Call Summary for Register TPTCj_DFCNTRLD

IVA2.2 Subsystem Register Manual

- TPTC0 and TPTC1 Register Mapping Summary: [0]

Table 14-537. TPTCj_DFSRCBREF

Address Offset	0x284		
Physical address	0x01C1 0284	Instance	IVA2.2 TPTC0
Physical address	0x01C1 0684	Instance	IVA2.2 TPTC1
Description	rsvd, return 0x0 w/o AERROR		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SADDRBREF																															

Bits	Field Name	Description	Type	Reset
31:0	SADDRBREF	Source address reference is not applicable for Dst FIFO Register Set. Reads return 0x0.	R	0x00000000

Table 14-538. Register Call Summary for Register TPTCj_DFSRCBREF

IVA2.2 Subsystem Register Manual

- TPTC0 and TPTC1 Register Mapping Summary: [0]

Table 14-539. TPTCj_DFDSTBREF

Address Offset	0x288		
Physical address	0x01C1 0288	**Instance**	IVA2.2 TPTC0
Physical address	0x01C1 0688	**Instance**	IVA2.2 TPTC1
Description	Dst FIFO Set Dst Addr A-Reference		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DADDRBREF																															

Bits	Field Name	Description	Type	Reset
31:0	DADDRBREF	Destination address reference for Dst FIFO Register Set: Represents the starting address for the array currently being written. The next arrays starting address is calculated as the reference address plus the dest bidxvalue.	R	0x00000000

Table 14-540. Register Call Summary for Register TPTCj_DFDSTBREF

IVA2.2 Subsystem Register Manual

- TPTC0 and TPTC1 Register Mapping Summary: [0]

Table 14-541. TPTCj_DFOPTi

Address Offset	0x300 + (0x40*i)		
Physical address	0x01C1 0300 + (0x40*i)	**Instance**	IVA2.2 TPTC0
Physical address	0x01C1 0700 + (0x40*i)	**Instance**	IVA2.2 TPTC1
Description	Dst FIFO i Set Options		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								TCCHEN	Reserved	TCINTEN	Reserved			TCC			Reserved	FWID		Reserved	PRI			Reserved			DAM	SAM		

Bits	Field Name	Description	Type	Reset
31:23	Reserved	Read returns 0.	R	0x000
22	TCCHEN	Transfer complete chaining enable: 0: Transfer complete chaining is disabled. 1: Transfer complete chaining is enabled.	R	0
21	Reserved	Read returns 0.	R	0
20	TCINTEN	Transfer complete interrupt enable: 0: Transfer complete interrupt is disabled. 1: Transfer complete interrupt is enabled.	R	0
19:18	Reserved	Read returns 0.	R	0x0
17:12	TCC	Transfer Complete Code: The 6-bit code is used to set the relevant bit in CER or IPR of the TPCC module.	R	0x00
11	Reserved	Read returns 0.	R	0
10:8	FWID	FIFO width control: Applies if either SAM or DAM is set to FIFO mode. Read 0x0: FIFO width is 8-bit Read 0x1: FIFO width is 16-bit Read 0x2: FIFO width is 32-bit Read 0x3: FIFO width is 64-bit Read 0x4: FIFO width is 128-bit Read 0x5: FIFO width is 256-bit	R	0x0
7	Reserved	Read returns 0.	R	0

Bits	Field Name	Description	Type	Reset
6:4	PRI	Transfer Priority: 0: Priority 0 - Highest priority 1: Priority 1 ... 7: Priority 7 - Lowest priority Read 0x0: Priority 0 - Highest priority Read 0x1: Priority 1 Read 0x2: Priority 2 Read 0x3: Priority 3 Read 0x4: Priority 4 Read 0x5: Priority 5 Read 0x6: Priority 6 Read 0x7: Priority 7 - Lowest Priority	R	0x0
3:2	Reserved	Read returns 0.	R	0x0
1	DAM	Destination Address Mode within an array: 0: INCR, Dst addressing within an array increments. 1: FIFO, Dst addressing within an array wraps around upon reaching FIFO width.	R	0
0	SAM	Source Address Mode within an array: 0: INCR, Src addressing within an array increments. 1: FIFO, Src addressing within an array wraps around upon reaching FIFO width.	R	0

Table 14-542. Register Call Summary for Register TPTCj_DFOPTi

IVA2.2 Subsystem Functional Description • EDMA: [0] [1] [2] [3]
IVA2.2 Subsystem Register Manual • TPTC0 and TPTC1 Register Mapping Summary: [4]

Table 14-543. TPTCj_DFSRCi

Address Offset	0x304 + (0x40*i)		
Physical address	0x01C1 0304 + (0x40*i)	**Instance**	IVA2.2 TPTC0
Physical address	0x01C1 0704 + (0x40*i)	**Instance**	IVA2.2 TPTC1
Description	rsvd, return 0x0 w/o AERROR		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SADDR																															

Bits	Field Name	Description	Type	Reset
31:0	SADDR	Source address is not applicable for Dst FIFORegister Set: Reads return 0x0.	R	0x00000000

Table 14-544. Register Call Summary for Register TPTCj_DFSRCi

IVA2.2 Subsystem Functional Description • EDMA: [0]
IVA2.2 Subsystem Register Manual • TPTC0 and TPTC1 Register Mapping Summary: [1]

Table 14-545. TPTCj_DFCNTi

Address Offset	0x308 + (0x40*i)		
Physical address	0x01C1 0308 + (0x40*i)	**Instance**	IVA2.2 TPTC0
Physical address	0x01C1 0708 + (0x40*i)	**Instance**	IVA2.2 TPTC1
Description	Dst FIFO i Set Count		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BCNT																ACNT															

Bits	Field Name	Description	Type	Reset
31:16	BCNT	B-Count Remaining for Dst Register Set: Number of arrays to be transferred, where each array is ACNT in length. Represents the amount of data remaining to be written. Initial value is copied from PCNT. TC decrements ACNT and BCNT as necessary after each write dataphase is issued. Final value should be 0 when TR is complete.	R	0x0000
15:0	ACNT	A-Count Remaining for Dst Register Set: Number of bytes to be transferred in first dimension. Represents the amount of data remaining to be written. Initial value is copied from PCNT. TC decrements ACNT and BCNT as necessary after each write dataphase is issued. Final value should be 0 when TR is complete.	R	0x0000

Table 14-546. Register Call Summary for Register TPTCj_DFCNTi

IVA2.2 Subsystem Functional Description

- EDMA: [0]

IVA2.2 Subsystem Register Manual

- TPTC0 and TPTC1 Register Mapping Summary: [1]

Table 14-547. TPTCj_DFDSTi

Address Offset	0x30C + (0x40*i)		
Physical address	0x01C1 030C + (0x40*i)	**Instance**	IVA2.2 TPTC0
Physical address	0x01C1 070C + (0x40*i)	**Instance**	IVA2.2 TPTC1
Description	Dst FIFO i Set Dst Address		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DADDR																															

Bits	Field Name	Description	Type	Reset
31:0	DADDR	Destination address for Dst FIFO Register Set: Initial value is copied from PDST.DADDR. TC updates value according to destination addressing mode (OPT.SAM) and/or dest index value (BIDX.DBIDX) after each write command is issued. When a TR is complete, the final value should be the address of the last write command issued.	R	0x00000000

Table 14-548. Register Call Summary for Register TPTCj_DFDSTi

IVA2.2 Subsystem Functional Description

- EDMA: [0]

IVA2.2 Subsystem Register Manual

- TPTC0 and TPTC1 Register Mapping Summary: [1]

Table 14-549. TPTCj_DFBIDXi

Address Offset	0x310 + (0x40*i)		
Physical address	0x01C1 0310 + (0x40*i)	**Instance**	IVA2.2 TPTC0
Physical address	0x01C1 0710 + (0x40*i)	**Instance**	IVA2.2 TPTC1
Description	Dst FIFO i Set B-Dim Idx		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DBIDX																SBIDX															

Bits	Field Name	Description	Type	Reset
31:16	DBIDX	Dest B-Idx for Dest FIFO Register Set. Value copied from PBIDX: B-Idx offset between Destination arrays: Represents the offset in bytes between the startingaddress of each destination array (recall that there are BCNT arrays of ACNT elements). DBIDXi is always used, regardless of whether DAM is Increment or FIFO mode.	R	0x0000
15:0	SBIDX	Dest B-Idx for Dest FIFO Register Set. Value copied from PBIDX: B-Idx offset between Source arrays: Represents the offset in bytes between the starting address of each source array (recall that there are BCNT arrays of ACNT elements). SBIDX is always used, regardless of whether SAM is Increment or FIFO mode.	R	0x0000

Table 14-550. Register Call Summary for Register TPTCj_DFBIDXi

IVA2.2 Subsystem Functional Description

- EDMA: [0]

IVA2.2 Subsystem Register Manual

- TPTC0 and TPTC1 Register Mapping Summary: [1]

Table 14-551. TPTCj_DFMPPRXYi

Address Offset	0x314 + (0x40*i)		
Physical address	0x01C1 0314 + (0x40*i)	**Instance**	IVA2.2 TPTC0
Physical address	0x01C1 0714 + (0x40*i)	**Instance**	IVA2.2 TPTC1
Description	Dst FIFO i Mem Protect Proxy		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																								PRIV	Reserved				PRIVID		

Bits	Field Name	Description	Type	Reset
31:9	Reserved	Read returns 0.	R	0x000000
8	PRIV	Privilege Level: PRIV = 0: User level privilege PRIV = 1: Supervisor level privilege DFMPPRXYi.PRIV is always updated with the value from the configuration bus Privilege field on any/every write to Program Set BIDX Register (trigger register). The PRIV value for the SA Set and DF Set are copied from the value in the Program set along with the remainder of the parameter values. The privilege ID is issued on the VBusM read and write command bus such that the target endpoints can perform memory protection checks based on the PRIV of the external host that sets up the DMA transaction.	R	0
7:4	Reserved	Read returns 0.	R	0x0

Bits	Field Name	Description	Type	Reset
3:0	PRIVID	Privilege ID: DFMPPRXYi.PRIVID is always updated with the value from configuration bus Privilege ID field on any/every write to Program Set BIDX Register (trigger register). The PRIVID value for the SA Set and DF Set are copied from the value in the Program set along with the remainder of the parameter values. The privilege ID is issued on the VBusM read and write command bus such that the target endpoints can perform memory protection checks based on the privid of the external host that sets up the DMA transaction.	R	0x0

Table 14-552. Register Call Summary for Register TPTCj_DFMPPRXYi

IVA2.2 Subsystem Functional Description
- EDMA: [0]

IVA2.2 Subsystem Register Manual
- TPTC0 and TPTC1 Register Mapping Summary: [1]

14.5.7 SYSC Registers

This section provides information about the IVA2_SYSC Module. Each register in the Module is described separately below.

14.5.7.1 SYSC Register Mapping Summary

Table 14-553. SYSC Register Summary

Register Name	Type	Register Width (Bits)	Address Offset	Physical Address
SYSC_REVISION	R	32	0x000	0x01C2 0000
SYSC_SYSCONFIG	RW	32	0x008	0x01C2 0008
SYSC_LICFG0	RW	32	0x040	0x01C2 0040
SYSC_LICFG1	RW	32	0x048	0x01C2 0048
SYSC_BOOTADDR	R	32	0x100	0x01C2 0100
SYSC_BOOTMOD	R	32	0x104	0x01C2 0104

14.5.7.2 SYSC Register Descriptions

Table 14-554. SYSC_REVISION

Address Offset	0x000		
Physical address	0x01C2 0000	**Instance**	IVA2.2 SYSC
Description	This register contains the IP revision code		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																								REV							

Bits	Field Name	Description	Type	Reset
31:8	Reserved	Read returns 0.	R	0x000000
7:0	REV	IP revision 3:0 Minor revision 7:4 Major revision	R	See (1)

(1) TI internal data

Table 14-555. Register Call Summary for Register SYSC_REVISION

IVA2.2 Subsystem Register Manual
- SYSC Register Mapping Summary: [0]

Table 14-556. SYSC_SYSCONFIG

Address Offset	0x008		
Physical address	0x01C2 0008	**Instance**	IVA2.2 SYSC
Description	This register allows controlling various parameters of the SYSC module		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																															AUTOIDLE

Bits	Field Name	Description	Type	Reset
31:1	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0x00000000
0	AUTOIDLE	Internal auto-clock gating strategy 0: Clock is free running 1: Automatic clock gating strategy is applied	RW	1

Table 14-557. Register Call Summary for Register SYSC_SYSCONFIG

IVA2.2 Subsystem Basic Programming Model
- Clock Management: [0] [1]

IVA2.2 Subsystem Register Manual
- SYSC Register Mapping Summary: [2]

Table 14-558. SYSC_LICFG0

Address Offset	0x040		
Physical address	0x01C2 0040	**Instance**	IVA2.2 SYSC
Description	This register controls various parameters of the IVA2.2 local interconnect network		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															GEMBURSTOPTEN	GEMTRUECOMPEN	Reserved					DMA2DOPTEN	DMATRUECOMPEN	Reserved				Reserved		PAGEXINGEN	Reserved	

Bits	Field Name	Description	Type	Reset
31:17	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0x0000
16	GEMBURSTOPTEN	DSP megamodule cache-line operation transfers optimizationcontrol: 0: DSP megamodule cache operation transfers are not optimized 1: DSP megamodule cache operation transfers are optimized	RW	0
15	GEMTRUECOMPEN	DSP megamodule program-initiated write-back transfer true completion control: 0: DSP megamodule program-initiated write-back transfer completion is not accurate 1: DSP megamodule program initiated write-back transfer completion is accurate	RW	0
14:10	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0x00
9	DMA2DOPTEN	2D transfers optimization control: 0: 2D DMA transfers optimization is disabled 1: 2D DMA transfers optimization is enabled	RW	0
8	DMATRUECOMPEN	DMA write transfer true completion control: 0: DMA write transfer completion is not accurate 1: DMA write transfer completion is accurate	RW	0
7:4	Reserved	Write 0xF for compatibilityRead returns 0xF	RW	0xF
3:2	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0x0
1	PAGEXINGEN	MMU page crossing enable: 0: Bursts are not allowed to cross 4KB page boundaries 1: Bursts are allowed to cross 4KB page boundaries	RW	0
0	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0

Table 14-559. Register Call Summary for Register SYSC_LICFG0

IVA2.2 Subsystem Functional Description

- Interconnect Optimization: [0]

IVA2.2 Subsystem Basic Programming Model

- Coherence Maintenance: [1] [2] [3] [4] [5] [6] [7] [8] [9] [10]
- Prioritizing Defined Transfers: [11] [12] [13]
- Starting the Transfer: [14]
- Recommendations for Static Settings: [15] [16] [17] [18] [19] [20]

IVA2.2 Subsystem Register Manual

- SYSC Register Mapping Summary: [21]

Table 14-560. SYSC_LICFG1

Address Offset	0x048		
Physical address	0x01C2 0048	**Instance**	IVA2.2 SYSC
Description	This register allows controlling various parameters of the IVA2.2 local interconnect network		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																								APINTERVAL							

Bits	Field Name	Description	Type	Reset
31:5	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0x0000000
4:0	APINTERVAL	Control of the Aged Priority (priority inversion) for DMA accesses: When set to 0x0, the aged priority is disabled When set to another value, this controls how often the priority is adjusted: every APInterval cycles, the priority of the port is decreased until the associated request is accepted or the priority of the port equals 0. The priority of the port is reinitialized to the transaction priority each time a new VBUS command is generated, i.e. the last one has been accepted. 0x0: Aged Priority Disabled: There is no aged priority in SCR. DMA transaction keeps the fixed priority defined internally, and has to wait for the bus to be freed by higher priority initiators	RW	0x00

Table 14-561. Register Call Summary for Register SYSC_LICFG1

IVA2.2 Subsystem Functional Description
- Interconnect Optimization: [0]

IVA2.2 Subsystem Basic Programming Model
- Prioritizing Defined Transfers: [1] [2]

IVA2.2 Subsystem Register Manual
- SYSC Register Mapping Summary: [3]

Table 14-562. SYSC_BOOTADDR

Address Offset	0x100		
Physical address	0x01C2 0100	**Instance**	IVA2.2 SYSC
Description	This register defines the physical address of the IVA2 boot loader. This is a copy of the CONTROL_IVA2_BOOTADDR when the IVA2 is released from reset.		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BOOTLOADADDR																				Reserved											

Bits	Field Name	Description	Type	Reset
31:12	BOOTLOADADDR	Physical address of the IVA2 boot loader: This is the read-only copy of the CONTROL_IVA2_BOOTADDR when the IVA2 is released fromreset This is an index to a 4K-byte page	R	0x-----
11:0	Reserved	Read returns 0.	R	0x000

Table 14-563. Register Call Summary for Register SYSC_BOOTADDR

IVA2.2 Subsystem Functional Description
- Boot Configuration: [0] [1]

Table 14-563. Register Call Summary for Register SYSC_BOOTADDR (continued)

IVA2.2 Subsystem Basic Programming Model • IVA2.2 Boot: [2] • Example of IVA2.2 Boot: [3]
IVA2.2 Subsystem Register Manual • SYSC Register Mapping Summary: [4]

Table 14-564. SYSC_BOOTMOD

Address Offset	0x104		
Physical address	0x01C2 0104	**Instance**	IVA2.2 SYSC
Description	This register defines the IVA2 boot mode. This is a copy of the CONTROL_IVA2_BOOTMOD when the IVA2 is released from reset.		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																												BOOTMODE			

Bits	Field Name	Description	Type	Reset
31:4	Reserved	Read returns 0.	R	0x0000000
3:0	BOOTMODE	Boot mode of the IVA2: This is the read-only copy of the IVA2_BOOTMOD when the IVA2 is released from reset The value meaning is defined by the IVA2 ROM boot code	R	0x-

Table 14-565. Register Call Summary for Register SYSC_BOOTMOD

IVA2.2 Subsystem Functional Description • Boot Configuration: [0] [1]
IVA2.2 Subsystem Basic Programming Model • IVA2.2 Boot: [2]
IVA2.2 Subsystem Register Manual • SYSC Register Mapping Summary: [3]

14.5.8 WUGEN Registers

This section provides information about the WUGEN Module. Each register in the Module is described separately below.

14.5.8.1 WUGEN Register Mapping Summary

Table 14-566. WUGEN Register Summary

Register Name	Type	Register Width (Bits)	Address Offset	Physical Address
WUGEN_REVISION	R	32	0x000	0x01C2 1000
WUGEN_SYSCONFIG	RW	32	0x008	0x01C2 1008
WUGEN_MEVT0	R	32	0x060	0x01C2 1060
WUGEN_MEVT1	R	32	0x064	0x01C2 1064
WUGEN_MEVT2	R	32	0x068	0x01C2 1068
WUGEN_MEVTCLR0	W	32	0x070	0x01C2 1070
WUGEN_MEVTCLR1	W	32	0x074	0x01C2 1074
WUGEN_MEVTCLR2	W	32	0x078	0x01C2 1078
WUGEN_MEVTSET0	W	32	0x080	0x01C2 1080
WUGEN_MEVTSET1	W	32	0x084	0x01C2 1084
WUGEN_MEVTSET2	W	32	0x088	0x01C2 1088
WUGEN_PENDEVT0	R	32	0x090	0x01C2 1090
WUGEN_PENDEVT1	R	32	0x094	0x01C2 1094
WUGEN_PENDEVT2	R	32	0x098	0x01C2 1098
WUGEN_PENDEVTCLR0	W	32	0x100	0x01C2 1100
WUGEN_PENDEVTCLR1	W	32	0x104	0x01C2 1104
WUGEN_PENDEVTCLR2	W	32	0x108	0x01C2 1108

14.5.8.2 WUGEN Register Descriptions

Table 14-567. WUGEN_REVISION

Address Offset	0x000		
Physical address	0x01C2 1000	**Instance**	IVA2.2 WUGEN
Description	This register contains the IP revision code		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																								REV							

Bits	Field Name	Description	Type	Reset
31:8	Reserved	Read returns 0.	R	0x000000
7:0	REV	IP revision 3:0 Minor revision 7:4 Major revision	R	See (1)

(1) TI internal data

Table 14-568. Register Call Summary for Register WUGEN_REVISION

IVA2.2 Subsystem Register Manual
- WUGEN Register Mapping Summary: [0]

Table 14-569. WUGEN_SYSCONFIG

Address Offset	0x008		
Physical address	0x01C2 1008	**Instance**	IVA2.2 WUGEN
Description	This register allows controlling various parameters of the WUGEN module		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																															AUTOIDLE

Bits	Field Name	Description	Type	Reset
31:1	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0x00000000
0	AUTOIDLE	Internal auto-clock gating strategy 0: Clock is free running 1: Automatic clock gating strategy is applied	RW	1

Table 14-570. Register Call Summary for Register WUGEN_SYSCONFIG

IVA2.2 Subsystem Basic Programming Model
- Clock Management: [0] [1]

IVA2.2 Subsystem Register Manual
- WUGEN Register Mapping Summary: [2]

Table 14-571. WUGEN_MEVT0

Address Offset	0x060		
Physical address	0x01C2 1060	**Instance**	IVA2.2 WUGEN
Description	This register contains the interrupt mask (LSB)		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MIRQ31	MIRQ30	MIRQ29	MIRQ28	MIRQ27	MIRQ26	MIRQ25	MIRQ24	MIRQ23	MIRQ22	MIRQ21	MIRQ20	MIRQ19	MIRQ18	MIRQ17	MIRQ16	MIRQ15	MIRQ14	MIRQ13	MIRQ12	MIRQ11	MIRQ10	MIRQ9	MIRQ8	MIRQ7	MIRQ6	MIRQ5	MIRQ4	MIRQ3	MIRQ2	MIRQ1	MIRQ0

Bits	Field Name	Description	Type	Reset
31	MIRQ31	Interrupt Mask bit #31	R	1
30	MIRQ30	Interrupt Mask bit #30	R	1
29	MIRQ29	Interrupt Mask bit #29	R	1
28	MIRQ28	Interrupt Mask bit #28	R	1
27	MIRQ27	Interrupt Mask bit #27	R	1
26	MIRQ26	Interrupt Mask bit #26	R	1
25	MIRQ25	Interrupt Mask bit #25	R	1
24	MIRQ24	Interrupt Mask bit #24	R	1
23	MIRQ23	Interrupt Mask bit #23	R	1
22	MIRQ22	Interrupt Mask bit #22	R	1
21	MIRQ21	Interrupt Mask bit #21	R	1
20	MIRQ20	Interrupt Mask bit #20	R	1
19	MIRQ19	Interrupt Mask bit #19	R	1
18	MIRQ18	Interrupt Mask bit #18	R	1
17	MIRQ17	Interrupt Mask bit #17	R	1
16	MIRQ16	Interrupt Mask bit #16	R	1
15	MIRQ15	Interrupt Mask bit #15	R	1
14	MIRQ14	Interrupt Mask bit #14	R	1
13	MIRQ13	Interrupt Mask bit #13	R	1
12	MIRQ12	Interrupt Mask bit #12	R	1
11	MIRQ11	Interrupt Mask bit #11	R	1
10	MIRQ10	Interrupt Mask bit #10	R	1
9	MIRQ9	Interrupt Mask bit #9	R	1
8	MIRQ8	Interrupt Mask bit #8	R	1
7	MIRQ7	Interrupt Mask bit #7	R	1
6	MIRQ6	Interrupt Mask bit #6	R	1
5	MIRQ5	Interrupt Mask bit #5	R	1
4	MIRQ4	Interrupt Mask bit #4	R	1
3	MIRQ3	Interrupt Mask bit #3	R	1
2	MIRQ2	Interrupt Mask bit #2	R	1
1	MIRQ1	Interrupt Mask bit #1	R	1
0	MIRQ0	Interrupt Mask bit #0	R	1

Table 14-572. Register Call Summary for Register WUGEN_MEVT0

IVA2.2 Subsystem Functional Description

- Interrupts, DMA Requests, and Event Management: [0] [1]

Table 14-572. Register Call Summary for Register WUGEN_MEVT0 (continued)

IVA2.2 Subsystem Basic Programming Model • Interrupt Controller Basic Programming Model for Power Down of IVA2.2 Subsystem: [2] [3] • Interrupt Controller Basic Programming Model for Power On of IVA2.2 Subsystem: [4] [5] [6]
IVA2.2 Subsystem Register Manual • WUGEN Register Mapping Summary: [7] • WUGEN Register Descriptions: [8] [9]

Table 14-573. WUGEN_MEVT1

Address Offset	0x064		
Physical address	0x01C2 1064	**Instance**	IVA2.2 WUGEN
Description	This register contains the interrupt mask (MSB)		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																MIRQ47	MIRQ46	MIRQ45	MIRQ44	MIRQ43	MIRQ42	MIRQ41	MIRQ40	MIRQ39	MIRQ38	MIRQ37	MIRQ36	MIRQ35	MIRQ34	MIRQ33	MIRQ32

Bits	Field Name	Description	Type	Reset
31:16	Reserved	Read returns 0.	R	0x0000
15	MIRQ47	Interrupt Mask bit #47	R	1
14	MIRQ46	Interrupt Mask bit #46	R	1
13	MIRQ45	Interrupt Mask bit #45	R	1
12	MIRQ44	Interrupt Mask bit #44	R	1
11	MIRQ43	Interrupt Mask bit #43	R	1
10	MIRQ42	Interrupt Mask bit #42	R	1
9	MIRQ41	Interrupt Mask bit #41	R	1
8	MIRQ40	Interrupt Mask bit #40	R	1
7	MIRQ39	Interrupt Mask bit #39	R	1
6	MIRQ38	Interrupt Mask bit #38	R	1
5	MIRQ37	Interrupt Mask bit #37	R	1
4	MIRQ36	Interrupt Mask bit #36	R	1
3	MIRQ35	Interrupt Mask bit #35	R	1
2	MIRQ34	Interrupt Mask bit #34	R	1
1	MIRQ33	Interrupt Mask bit #33	R	1
0	MIRQ32	Interrupt Mask bit #32	R	1

Table 14-574. Register Call Summary for Register WUGEN_MEVT1

IVA2.2 Subsystem Functional Description • Interrupts, DMA Requests, and Event Management: [0]
IVA2.2 Subsystem Basic Programming Model • Interrupt Controller Basic Programming Model for Power Down of IVA2.2 Subsystem: [1] [2] • Interrupt Controller Basic Programming Model for Power On of IVA2.2 Subsystem: [3] [4] [5]
IVA2.2 Subsystem Register Manual • WUGEN Register Mapping Summary: [6] • WUGEN Register Descriptions: [7] [8]

Table 14-575. WUGEN_MEVT2

Address Offset	0x068		
Physical address	0x01C2 1068	**Instance**	IVA2.2 WUGEN
Description	This register contains the dma request mask		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved												MDMARQ19	MDMARQ18	MDMARQ17	MDMARQ16	MDMARQ15	MDMARQ14	MDMARQ13	MDMARQ12	MDMARQ11	MDMARQ10	MDMARQ9	MDMARQ8	MDMARQ7	MDMARQ6	MDMARQ5	MDMARQ4	MDMARQ3	MDMARQ2	MDMARQ1	MDMARQ0

Bits	Field Name	Description	Type	Reset
31:20	Reserved	Read returns 0.	R	0x000
19	MDMARQ19	DMA request Mask bit #19	R	1
18	MDMARQ18	DMA request Mask bit #18	R	1
17	MDMARQ17	DMA request Mask bit #17	R	1
16	MDMARQ16	DMA request Mask bit #16	R	1
15	MDMARQ15	DMA request Mask bit #15	R	1
14	MDMARQ14	DMA request Mask bit #14	R	1
13	MDMARQ13	DMA request Mask bit #13	R	1
12	MDMARQ12	DMA request Mask bit #12	R	1
11	MDMARQ11	DMA request Mask bit #11	R	1
10	MDMARQ10	DMA request Mask bit #10	R	1
9	MDMARQ9	DMA request Mask bit #9	R	1
8	MDMARQ8	DMA request Mask bit #8	R	1
7	MDMARQ7	DMA request Mask bit #7	R	1
6	MDMARQ6	DMA request Mask bit #6	R	1
5	MDMARQ5	DMA request Mask bit #5	R	1
4	MDMARQ4	DMA request Mask bit #4	R	1
3	MDMARQ3	DMA request Mask bit #3	R	1
2	MDMARQ2	DMA request Mask bit #2	R	1
1	MDMARQ1	DMA request Mask bit #1	R	1
0	MDMARQ0	DMA request Mask bit #0	R	1

Table 14-576. Register Call Summary for Register WUGEN_MEVT2

IVA2.2 Subsystem Functional Description

- Interrupts, DMA Requests, and Event Management: [0] [1]

IVA2.2 Subsystem Register Manual

- WUGEN Register Mapping Summary: [2]
- WUGEN Register Descriptions: [3] [4]

Table 14-577. WUGEN_MEVTCLR0

Address Offset	0x070		
Physical address	0x01C2 1070	**Instance**	IVA2.2 WUGEN
Description	This register is used to clear the interrupt mask bits (LSB) Write 0: No effect Write 1: Clears the corresponding mask bit in the WUGEN_MEVT0 register Reads always return 0		
Type	W 1toSet		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MIRQCLR31	MIRQCLR30	MIRQCLR29	MIRQCLR28	MIRQCLR27	MIRQCLR26	MIRQCLR25	MIRQCLR24	MIRQCLR23	MIRQCLR22	MIRQCLR21	MIRQCLR20	MIRQCLR19	MIRQCLR18	MIRQCLR17	MIRQCLR16	MIRQCLR15	MIRQCLR14	MIRQCLR13	MIRQCLR12	MIRQCLR11	MIRQCLR10	MIRQCLR9	MIRQCLR8	MIRQCLR7	MIRQCLR6	MIRQCLR5	MIRQCLR4	MIRQCLR3	MIRQCLR2	MIRQCLR1	MIRQCLR0

Bits	Field Name	Description	Type	Reset
31	MIRQCLR31	MIRQ clear #31	W 1toSet	0
30	MIRQCLR30	MIRQ clear #30	W 1toSet	0
29	MIRQCLR29	MIRQ clear #29	W 1toSet	0
28	MIRQCLR28	MIRQ clear #28	W 1toSet	0
27	MIRQCLR27	MIRQ clear #27	W 1toSet	0
26	MIRQCLR26	MIRQ clear #26	W 1toSet	0
25	MIRQCLR25	MIRQ clear #25	W 1toSet	0
24	MIRQCLR24	MIRQ clear #24	W 1toSet	0
23	MIRQCLR23	MIRQ clear #23	W 1toSet	0
22	MIRQCLR22	MIRQ clear #22	W 1toSet	0
21	MIRQCLR21	MIRQ clear #21	W 1toSet	0
20	MIRQCLR20	MIRQ clear #20	W 1toSet	0
19	MIRQCLR19	MIRQ clear #19	W 1toSet	0
18	MIRQCLR18	MIRQ clear #18	W 1toSet	0
17	MIRQCLR17	MIRQ clear #17	W 1toSet	0
16	MIRQCLR16	MIRQ clear #16	W 1toSet	0
15	MIRQCLR15	MIRQ clear #15	W 1toSet	0
14	MIRQCLR14	MIRQ clear #14	W 1toSet	0
13	MIRQCLR13	MIRQ clear #13	W 1toSet	0
12	MIRQCLR12	MIRQ clear #12	W 1toSet	0
11	MIRQCLR11	MIRQ clear #11	W 1toSet	0

Bits	Field Name	Description	Type	Reset
10	MIRQCLR10	MIRQ clear #10	W 1toSet	0
9	MIRQCLR9	MIRQ clear #9	W 1toSet	0
8	MIRQCLR8	MIRQ clear #8	W 1toSet	0
7	MIRQCLR7	MIRQ clear #7	W 1toSet	0
6	MIRQCLR6	MIRQ clear #6	W 1toSet	0
5	MIRQCLR5	MIRQ clear #5	W 1toSet	0
4	MIRQCLR4	MIRQ clear #4	W 1toSet	0
3	MIRQCLR3	MIRQ clear #3	W 1toSet	0
2	MIRQCLR2	MIRQ clear #2	W 1toSet	0
1	MIRQCLR1	MIRQ clear #1	W 1toSet	0
0	MIRQCLR0	MIRQ clear #0	W 1toSet	0

Table 14-578. Register Call Summary for Register WUGEN_MEVTCLR0

IVA2.2 Subsystem Integration
- Interrupt Requests: [0]

IVA2.2 Subsystem Functional Description
- Interrupts, DMA Requests, and Event Management: [1] [2]

IVA2.2 Subsystem Register Manual
- WUGEN Register Mapping Summary: [3]

Table 14-579. WUGEN_MEVTCLR1

Address Offset	0x074		
Physical address	0x01C2 1074	**Instance**	IVA2.2 WUGEN
Description	This register is used to clear the interrupt mask bits (MSB) Write 0: No effect Write 1: Clears the corresponding mask bit in the WUGEN_MEVT1 register Reads always return 0		
Type	W 1toSet		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																MIRQCLR47	MIRQCLR46	MIRQCLR45	MIRQCLR44	MIRQCLR43	MIRQCLR42	MIRQCLR41	MIRQCLR40	MIRQCLR39	MIRQCLR38	MIRQCLR37	MIRQCLR36	MIRQCLR35	MIRQCLR34	MIRQCLR33	MIRQCLR32

Bits	Field Name	Description	Type	Reset
31:16	Reserved	Write 0s for future compatibility.	W	0x0000
15	MIRQCLR47	MIRQ clear #47	W 1toSet	0
14	MIRQCLR46	MIRQ clear #46	W 1toSet	0
13	MIRQCLR45	MIRQ clear #45	W 1toSet	0

Bits	Field Name	Description	Type	Reset
12	MIRQCLR44	MIRQ clear #44	W 1toSet	0
11	MIRQCLR43	MIRQ clear #43	W 1toSet	0
10	MIRQCLR42	MIRQ clear #42	W 1toSet	0
9	MIRQCLR41	MIRQ clear #41	W 1toSet	0
8	MIRQCLR40	MIRQ clear #40	W 1toSet	0
7	MIRQCLR39	MIRQ clear #39	W 1toSet	0
6	MIRQCLR38	MIRQ clear #38	W 1toSet	0
5	MIRQCLR37	MIRQ clear #37	W 1toSet	0
4	MIRQCLR36	MIRQ clear #36	W 1toSet	0
3	MIRQCLR35	MIRQ clear #35	W 1toSet	0
2	MIRQCLR34	MIRQ clear #34	W 1toSet	0
1	MIRQCLR33	MIRQ clear #33	W 1toSet	0
0	MIRQCLR32	MIRQ clear #32	W 1toSet	0

Table 14-580. Register Call Summary for Register WUGEN_MEVTCLR1

IVA2.2 Subsystem Integration
- Interrupt Requests: [0]

IVA2.2 Subsystem Functional Description
- Interrupts, DMA Requests, and Event Management: [1] [2]

IVA2.2 Subsystem Register Manual
- WUGEN Register Mapping Summary: [3]

Table 14-581. WUGEN_MEVTCLR2

Address Offset	0x078		
Physical address	0x01C2 1078	**Instance**	IVA2.2 WUGEN
Description	This register is used to clear the dma request mask bits Write 0: No effect Write 1: Clears the corresponding mask bit in the WUGEN_MEVT2 register Reads always return 0		
Type	W 1toSet		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved												MDMARQCLR19	MDMARQCLR18	MDMARQCLR17	MDMARQCLR16	MDMARQCLR15	MDMARQCLR14	MDMARQCLR13	MDMARQCLR12	MDMARQCLR11	MDMARQCLR10	MDMARQCLR9	MDMARQCLR8	MDMARQCLR7	MDMARQCLR6	MDMARQCLR5	MDMARQCLR4	MDMARQCLR3	MDMARQCLR2	MDMARQCLR1	MDMARQCLR0

Bits	Field Name	Description	Type	Reset
31:20	Reserved	Write 0s for future compatibility.	W	0x000
19	MDMARQCLR19	MDMARQ clear #19	W 1toSet	0
18	MDMARQCLR18	MDMARQ clear #18	W 1toSet	0
17	MDMARQCLR17	MDMARQ clear #17	W 1toSet	0
16	MDMARQCLR16	MDMARQ clear #16	W 1toSet	0
15	MDMARQCLR15	MDMARQ clear #15	W 1toSet	0
14	MDMARQCLR14	MDMARQ clear #14	W 1toSet	0
13	MDMARQCLR13	MDMARQ clear #13	W 1toSet	0
12	MDMARQCLR12	MDMARQ clear #12	W 1toSet	0
11	MDMARQCLR11	MDMARQ clear #11	W 1toSet	0
10	MDMARQCLR10	MDMARQ clear #10	W 1toSet	0
9	MDMARQCLR9	MDMARQ clear #9	W 1toSet	0
8	MDMARQCLR8	MDMARQ clear #8	W 1toSet	0
7	MDMARQCLR7	MDMARQ clear #7	W 1toSet	0
6	MDMARQCLR6	MDMARQ clear #6	W 1toSet	0
5	MDMARQCLR5	MDMARQ clear #5	W 1toSet	0
4	MDMARQCLR4	MDMARQ clear #4	W 1toSet	0
3	MDMARQCLR3	MDMARQ clear #3	W 1toSet	0
2	MDMARQCLR2	MDMARQ clear #2	W 1toSet	0
1	MDMARQCLR1	MDMARQ clear #1	W 1toSet	0
0	MDMARQCLR0	MDMARQ clear #0	W 1toSet	0

Table 14-582. Register Call Summary for Register WUGEN_MEVTCLR2

IVA2.2 Subsystem Functional Description • Interrupts, DMA Requests, and Event Management: [0] [1]
IVA2.2 Subsystem Register Manual • WUGEN Register Mapping Summary: [2]

Table 14-583. WUGEN_MEVTSET0

Address Offset	0x080		
Physical address	0x01C2 1080	**Instance**	IVA2.2 WUGEN
Description	This register is used to set the interrupt mask bits (LSB) Write 0: No effect Write 1: Sets the corresponding mask bit in the WUGEN_MEVT0 register Reads always return 0		
Type	W 1toSet		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MIRQSET31	MIRQSET30	MIRQSET29	MIRQSET28	MIRQSET27	MIRQSET26	MIRQSET25	MIRQSET24	MIRQSET23	MIRQSET22	MIRQSET21	MIRQSET20	MIRQSET19	MIRQSET18	MIRQSET17	MIRQSET16	MIRQSET15	MIRQSET14	MIRQSET13	MIRQSET12	MIRQSET11	MIRQSET10	MIRQSET9	MIRQSET8	MIRQSET7	MIRQSET6	MIRQSET5	MIRQSET4	MIRQSET3	MIRQSET2	MIRQSET1	MIRQSET0

Bits	Field Name	Description	Type	Reset
31	MIRQSET31	MIRQ set #31	W 1toSet	0
30	MIRQSET30	MIRQ set #30	W 1toSet	0
29	MIRQSET29	MIRQ set #29	W 1toSet	0
28	MIRQSET28	MIRQ set #28	W 1toSet	0
27	MIRQSET27	MIRQ set #27	W 1toSet	0
26	MIRQSET26	MIRQ set #26	W 1toSet	0
25	MIRQSET25	MIRQ set #25	W 1toSet	0
24	MIRQSET24	MIRQ set #24	W 1toSet	0
23	MIRQSET23	MIRQ set #23	W 1toSet	0
22	MIRQSET22	MIRQ set #22	W 1toSet	0
21	MIRQSET21	MIRQ set #21	W 1toSet	0
20	MIRQSET20	MIRQ set #20	W 1toSet	0
19	MIRQSET19	MIRQ set #19	W 1toSet	0
18	MIRQSET18	MIRQ set #18	W 1toSet	0
17	MIRQSET17	MIRQ set #17	W 1toSet	0
16	MIRQSET16	MIRQ set #16	W 1toSet	0
15	MIRQSET15	MIRQ set #15	W 1toSet	0
14	MIRQSET14	MIRQ set #14	W 1toSet	0
13	MIRQSET13	MIRQ set #13	W 1toSet	0
12	MIRQSET12	MIRQ set #12	W 1toSet	0
11	MIRQSET11	MIRQ set #11	W 1toSet	0

Bits	Field Name	Description	Type	Reset
10	MIRQSET10	MIRQ set #10	W 1toSet	0
9	MIRQSET9	MIRQ set #9	W 1toSet	0
8	MIRQSET8	MIRQ set #8	W 1toSet	0
7	MIRQSET7	MIRQ set #7	W 1toSet	0
6	MIRQSET6	MIRQ set #6	W 1toSet	0
5	MIRQSET5	MIRQ set #5	W 1toSet	0
4	MIRQSET4	MIRQ set #4	W 1toSet	0
3	MIRQSET3	MIRQ set #3	W 1toSet	0
2	MIRQSET2	MIRQ set #2	W 1toSet	0
1	MIRQSET1	MIRQ set #1	W 1toSet	0
0	MIRQSET0	MIRQ set #0	W 1toSet	0

Table 14-584. Register Call Summary for Register WUGEN_MEVTSET0

IVA2.2 Subsystem Integration • Interrupt Requests: [0]
IVA2.2 Subsystem Functional Description • Interrupts, DMA Requests, and Event Management: [1] [2]
IVA2.2 Subsystem Register Manual • WUGEN Register Mapping Summary: [3]

Table 14-585. WUGEN_MEVTSET1

Address Offset	0x084		
Physical address	0x01C2 1084	**Instance**	IVA2.2 WUGEN
Description	This register is used to set the interrupt mask bits (MSB) Write 0: No effect Write 1: Sets the corresponding mask bit in the WUGEN_MEVT1 register Reads always return 0		
Type	W 1toSet		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																MIRQSET47	MIRQSET46	MIRQSET45	MIRQSET44	MIRQSET43	MIRQSET42	MIRQSET41	MIRQSET40	MIRQSET39	MIRQSET38	MIRQSET37	MIRQSET36	MIRQSET35	MIRQSET34	MIRQSET33	MIRQSET32

Bits	Field Name	Description	Type	Reset
31:16	Reserved	Write 0s for future compatibility.	W	0x0000
15	MIRQSET47	MIRQ set #47	W 1toSet	0
14	MIRQSET46	MIRQ set #46	W 1toSet	0
13	MIRQSET45	MIRQ set #45	W 1toSet	0

Bits	Field Name	Description	Type	Reset
12	MIRQSET44	MIRQ set #44	W 1toSet	0
11	MIRQSET43	MIRQ set #43	W 1toSet	0
10	MIRQSET42	MIRQ set #42	W 1toSet	0
9	MIRQSET41	MIRQ set #41	W 1toSet	0
8	MIRQSET40	MIRQ set #40	W 1toSet	0
7	MIRQSET39	MIRQ set #39	W 1toSet	0
6	MIRQSET38	MIRQ set #38	W 1toSet	0
5	MIRQSET37	MIRQ set #37	W 1toSet	0
4	MIRQSET36	MIRQ set #36	W 1toSet	0
3	MIRQSET35	MIRQ set #35	W 1toSet	0
2	MIRQSET34	MIRQ set #34	W 1toSet	0
1	MIRQSET33	MIRQ set #33	W 1toSet	0
0	MIRQSET32	MIRQ set #32	W 1toSet	0

Table 14-586. Register Call Summary for Register WUGEN_MEVTSET1

IVA2.2 Subsystem Integration • Interrupt Requests: [0]
IVA2.2 Subsystem Functional Description • Interrupts, DMA Requests, and Event Management: [1] [2]
IVA2.2 Subsystem Register Manual • WUGEN Register Mapping Summary: [3]

Table 14-587. WUGEN_MEVTSET2

Address Offset	0x088		
Physical address	0x01C2 1088	**Instance**	IVA2.2 WUGEN
Description	This register is used to set the dma requests mask bits Write 0: No effect Write 1: Sets the corresponding mask bit in the WUGEN_MEVT2 register Reads always return 0		
Type	W 1toSet		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved												MDMARQSET19	MDMARQSET18	MDMARQSET17	MDMARQSET16	MDMARQSET15	MDMARQSET14	MDMARQSET13	MDMARQSET12	MDMARQSET11	MDMARQSET10	MDMARQSET9	MDMARQSET8	MDMARQSET7	MDMARQSET6	MDMARQSET5	MDMARQSET4	MDMARQSET3	MDMARQSET2	MDMARQSET1	MDMARQSET0

Bits	Field Name	Description	Type	Reset
31:20	Reserved	Write 0s for future compatibility.	W	0x000
19	MDMARQSET19	MDMARQ set #19	W 1toSet	0
18	MDMARQSET18	MDMARQ set #18	W 1toSet	0
17	MDMARQSET17	MDMARQ set #17	W 1toSet	0
16	MDMARQSET16	MDMARQ set #16	W 1toSet	0
15	MDMARQSET15	MDMARQ set #15	W 1toSet	0
14	MDMARQSET14	MDMARQ set #14	W 1toSet	0
13	MDMARQSET13	MDMARQ set #13	W 1toSet	0
12	MDMARQSET12	MDMARQ set #12	W 1toSet	0
11	MDMARQSET11	MDMARQ set #11	W 1toSet	0
10	MDMARQSET10	MDMARQ set #10	W 1toSet	0
9	MDMARQSET9	MDMARQ set #9	W 1toSet	0
8	MDMARQSET8	MDMARQ set #8	W 1toSet	0
7	MDMARQSET7	MDMARQ set #7	W 1toSet	0
6	MDMARQSET6	MDMARQ set #6	W 1toSet	0
5	MDMARQSET5	MDMARQ set #5	W 1toSet	0
4	MDMARQSET4	MDMARQ set #4	W 1toSet	0
3	MDMARQSET3	MDMARQ set #3	W 1toSet	0
2	MDMARQSET2	MDMARQ set #2	W 1toSet	0
1	MDMARQSET1	MDMARQ set #1	W 1toSet	0
0	MDMARQSET0	MDMARQ set #0	W 1toSet	0

Table 14-588. Register Call Summary for Register WUGEN_MEVTSET2

IVA2.2 Subsystem Functional Description • Interrupts, DMA Requests, and Event Management: [0] [1]
IVA2.2 Subsystem Register Manual • WUGEN Register Mapping Summary: [2]

Table 14-589. WUGEN_PENDEVT0

Address Offset	0x090		
Physical address	0x01C2 1090	**Instance**	IVA2.2 WUGEN
Description	This register holds the masked pending interrupts (LSB)		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PENDIRQ31	PENDIRQ30	PENDIRQ29	PENDIRQ28	PENDIRQ27	PENDIRQ26	PENDIRQ25	PENDIRQ24	PENDIRQ23	PENDIRQ22	PENDIRQ21	PENDIRQ20	PENDIRQ19	PENDIRQ18	PENDIRQ17	PENDIRQ16	PENDIRQ15	PENDIRQ14	PENDIRQ13	PENDIRQ12	PENDIRQ11	PENDIRQ10	PENDIRQ9	PENDIRQ8	PENDIRQ7	PENDIRQ6	PENDIRQ5	PENDIRQ4	PENDIRQ3	PENDIRQ2	PENDIRQ1	PENDIRQ0

Bits	Field Name	Description	Type	Reset
31	PENDIRQ31	Masked pending interrupt number 31	R	0
30	PENDIRQ30	Masked pending interrupt number 30	R	0
29	PENDIRQ29	Masked pending interrupt number 29	R	0
28	PENDIRQ28	Masked pending interrupt number 28	R	0
27	PENDIRQ27	Masked pending interrupt number 27	R	0
26	PENDIRQ26	Masked pending interrupt number 26	R	0
25	PENDIRQ25	Masked pending interrupt number 25	R	0
24	PENDIRQ24	Masked pending interrupt number 24	R	0
23	PENDIRQ23	Masked pending interrupt number 23	R	0
22	PENDIRQ22	Masked pending interrupt number 22	R	0
21	PENDIRQ21	Masked pending interrupt number 21	R	0
20	PENDIRQ20	Masked pending interrupt number 20	R	0
19	PENDIRQ19	Masked pending interrupt number 19	R	0
18	PENDIRQ18	Masked pending interrupt number 18	R	0
17	PENDIRQ17	Masked pending interrupt number 17	R	0
16	PENDIRQ16	Masked pending interrupt number 16	R	0
15	PENDIRQ15	Masked pending interrupt number 15	R	0
14	PENDIRQ14	Masked pending interrupt number 14	R	0
13	PENDIRQ13	Masked pending interrupt number 13	R	0
12	PENDIRQ12	Masked pending interrupt number 12	R	0
11	PENDIRQ11	Masked pending interrupt number 11	R	0
10	PENDIRQ10	Masked pending interrupt number 10	R	0
9	PENDIRQ9	Masked pending interrupt number 9	R	0
8	PENDIRQ8	Masked pending interrupt number 8	R	0
7	PENDIRQ7	Masked pending interrupt number 7	R	0
6	PENDIRQ6	Masked pending interrupt number 6	R	0
5	PENDIRQ5	Masked pending interrupt number 5	R	0
4	PENDIRQ4	Masked pending interrupt number 4	R	0
3	PENDIRQ3	Masked pending interrupt number 3	R	0
2	PENDIRQ2	Masked pending interrupt number 2	R	0
1	PENDIRQ1	Masked pending interrupt number 1	R	0
0	PENDIRQ0	Masked pending interrupt number 0	R	0

Table 14-590. Register Call Summary for Register WUGEN_PENDEVT0

IVA2.2 Subsystem Functional Description

- Interrupts, DMA Requests, and Event Management: [0]

Table 14-590. Register Call Summary for Register WUGEN_PENDEVT0 (continued)

IVA2.2 Subsystem Register Manual

- WUGEN Register Mapping Summary: [1]
- WUGEN Register Descriptions: [2]

Table 14-591. WUGEN_PENDEVT1

Address Offset	0x094		
Physical address	0x01C2 1094	**Instance**	IVA2.2 WUGEN
Description	This register holds the masked pending interrupts (MSB)		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																PENDIRQ47	PENDIRQ46	PENDIRQ45	PENDIRQ44	PENDIRQ43	PENDIRQ42	PENDIRQ41	PENDIRQ40	PENDIRQ39	PENDIRQ38	PENDIRQ37	PENDIRQ36	PENDIRQ35	PENDIRQ34	PENDIRQ33	PENDIRQ32

Bits	Field Name	Description	Type	Reset
31:16	Reserved	Read returns 0.	R	0x0000
15	PENDIRQ47	Masked pending interrupt number 47	R	0
14	PENDIRQ46	Masked pending interrupt number 46	R	0
13	PENDIRQ45	Masked pending interrupt number 45	R	0
12	PENDIRQ44	Masked pending interrupt number 44	R	0
11	PENDIRQ43	Masked pending interrupt number 43	R	0
10	PENDIRQ42	Masked pending interrupt number 42	R	0
9	PENDIRQ41	Masked pending interrupt number 41	R	0
8	PENDIRQ40	Masked pending interrupt number 40	R	0
7	PENDIRQ39	Masked pending interrupt number 39	R	0
6	PENDIRQ38	Masked pending interrupt number 38	R	0
5	PENDIRQ37	Masked pending interrupt number 37	R	0
4	PENDIRQ36	Masked pending interrupt number 36	R	0
3	PENDIRQ35	Masked pending interrupt number 35	R	0
2	PENDIRQ34	Masked pending interrupt number 34	R	0
1	PENDIRQ33	Masked pending interrupt number 33	R	0
0	PENDIRQ32	Masked pending interrupt number 32	R	0

Table 14-592. Register Call Summary for Register WUGEN_PENDEVT1

IVA2.2 Subsystem Functional Description

- Interrupts, DMA Requests, and Event Management: [0]

IVA2.2 Subsystem Register Manual

- WUGEN Register Mapping Summary: [1]
- WUGEN Register Descriptions: [2]

Table 14-593. WUGEN_PENDEVT2

Address Offset	0x098		
Physical address	0x01C2 1098	**Instance**	IVA2.2 WUGEN
Description	This register holds the masked pending dma requests		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved												PENDDMARQ19	PENDDMARQ18	PENDDMARQ17	PENDDMARQ16	PENDDMARQ15	PENDDMARQ14	PENDDMARQ13	PENDDMARQ12	PENDDMARQ11	PENDDMARQ10	PENDDMARQ9	PENDDMARQ8	PENDDMARQ7	PENDDMARQ6	PENDDMARQ5	PENDDMARQ4	PENDDMARQ3	PENDDMARQ2	PENDDMARQ1	PENDDMARQ0

Bits	Field Name	Description	Type	Reset
31:20	Reserved	Read returns 0.	R	0x000
19	PENDDMARQ19	Masked pending dma request number 0	R	0
18	PENDDMARQ18	Masked pending dma request number 0	R	0
17	PENDDMARQ17	Masked pending dma request number 0	R	0
16	PENDDMARQ16	Masked pending dma request number 0	R	0
15	PENDDMARQ15	Masked pending dma request number 0	R	0
14	PENDDMARQ14	Masked pending dma request number 0	R	0
13	PENDDMARQ13	Masked pending dma request number 0	R	0
12	PENDDMARQ12	Masked pending dma request number 0	R	0
11	PENDDMARQ11	Masked pending dma request number 0	R	0
10	PENDDMARQ10	Masked pending dma request number 0	R	0
9	PENDDMARQ9	Masked pending dma request number 0	R	0
8	PENDDMARQ8	Masked pending dma request number 0	R	0
7	PENDDMARQ7	Masked pending dma request number 0	R	0
6	PENDDMARQ6	Masked pending dma request number 0	R	0
5	PENDDMARQ5	Masked pending dma request number 0	R	0
4	PENDDMARQ4	Masked pending dma request number 0	R	0
3	PENDDMARQ3	Masked pending dma request number 0	R	0
2	PENDDMARQ2	Masked pending dma request number 0	R	0
1	PENDDMARQ1	Masked pending dma request number 0	R	0
0	PENDDMARQ0	Masked pending dma request number 0	R	0

Table 14-594. Register Call Summary for Register WUGEN_PENDEVT2

IVA2.2 Subsystem Functional Description

- Interrupts, DMA Requests, and Event Management: [0]

IVA2.2 Subsystem Register Manual

- WUGEN Register Mapping Summary: [1]
- WUGEN Register Descriptions: [2]

Table 14-595. WUGEN_PENDEVTCLR0

Address Offset	0x100
Physical address	0x01C2 1100 **Instance** IVA2.2 WUGEN
Description	This register clears the masked pending interrupts (LSB): Write 0: No effect Write 1: Clears the corresponding mask bit in the WUGEN_PENDEVT0 register Reads always return 0
Type	W

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PENDIRQ31	PENDIRQ30	PENDIRQ29	PENDIRQ28	PENDIRQ27	PENDIRQ26	PENDIRQ25	PENDIRQ24	PENDIRQ23	PENDIRQ22	PENDIRQ21	PENDIRQ20	PENDIRQ19	PENDIRQ18	PENDIRQ17	PENDIRQ16	PENDIRQ15	PENDIRQ14	PENDIRQ13	PENDIRQ12	PENDIRQ11	PENDIRQ10	PENDIRQ9	PENDIRQ8	PENDIRQ7	PENDIRQ6	PENDIRQ5	PENDIRQ4	PENDIRQ3	PENDIRQ2	PENDIRQ1	PENDIRQ0

Bits	Field Name	Description	Type	Reset
31	PENDIRQ31	Masked pending interrupt number 31	W	0
30	PENDIRQ30	Masked pending interrupt number 30	W	0
29	PENDIRQ29	Masked pending interrupt number 29	W	0
28	PENDIRQ28	Masked pending interrupt number 28	W	0
27	PENDIRQ27	Masked pending interrupt number 27	W	0
26	PENDIRQ26	Masked pending interrupt number 26	W	0
25	PENDIRQ25	Masked pending interrupt number 25	W	0
24	PENDIRQ24	Masked pending interrupt number 24	W	0
23	PENDIRQ23	Masked pending interrupt number 23	W	0
22	PENDIRQ22	Masked pending interrupt number 22	W	0
21	PENDIRQ21	Masked pending interrupt number 21	W	0
20	PENDIRQ20	Masked pending interrupt number 20	W	0
19	PENDIRQ19	Masked pending interrupt number 19	W	0
18	PENDIRQ18	Masked pending interrupt number 18	W	0
17	PENDIRQ17	Masked pending interrupt number 17	W	0
16	PENDIRQ16	Masked pending interrupt number 16	W	0
15	PENDIRQ15	Masked pending interrupt number 15	W	0
14	PENDIRQ14	Masked pending interrupt number 14	W	0
13	PENDIRQ13	Masked pending interrupt number 13	W	0
12	PENDIRQ12	Masked pending interrupt number 12	W	0
11	PENDIRQ11	Masked pending interrupt number 11	W	0
10	PENDIRQ10	Masked pending interrupt number 10	W	0
9	PENDIRQ9	Masked pending interrupt number 9	W	0
8	PENDIRQ8	Masked pending interrupt number 8	W	0
7	PENDIRQ7	Masked pending interrupt number 7	W	0
6	PENDIRQ6	Masked pending interrupt number 6	W	0
5	PENDIRQ5	Masked pending interrupt number 5	W	0
4	PENDIRQ4	Masked pending interrupt number 4	W	0
3	PENDIRQ3	Masked pending interrupt number 3	W	0
2	PENDIRQ2	Masked pending interrupt number 2	W	0
1	PENDIRQ1	Masked pending interrupt number 1	W	0
0	PENDIRQ0	Masked pending interrupt number 0	W	0

Table 14-596. Register Call Summary for Register WUGEN_PENDEVTCLR0

IVA2.2 Subsystem Register Manual

- WUGEN Register Mapping Summary: [0]

Table 14-597. WUGEN_PENDEVTCLR1

Address Offset	0x104		
Physical address	0x01C2 1104	**Instance**	IVA2.2 WUGEN
Description	This register clears the masked pending interrupts (MSB) : Write 0: No effect Write 1: Clears the corresponding mask bit in the WUGEN_PENDEVT1 register Reads always return 0		
Type	W		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																PENDIRQ47	PENDIRQ46	PENDIRQ45	PENDIRQ44	PENDIRQ43	PENDIRQ42	PENDIRQ41	PENDIRQ40	PENDIRQ39	PENDIRQ38	PENDIRQ37	PENDIRQ36	PENDIRQ35	PENDIRQ34	PENDIRQ33	PENDIRQ32

Bits	Field Name	Description	Type	Reset
31:16	Reserved	Write 0s for future compatibility.	W	0x0000
15	PENDIRQ47	Masked pending interrupt number 47	W	0
14	PENDIRQ46	Masked pending interrupt number 46	W	0
13	PENDIRQ45	Masked pending interrupt number 45	W	0
12	PENDIRQ44	Masked pending interrupt number 44	W	0
11	PENDIRQ43	Masked pending interrupt number 43	W	0
10	PENDIRQ42	Masked pending interrupt number 42	W	0
9	PENDIRQ41	Masked pending interrupt number 41	W	0
8	PENDIRQ40	Masked pending interrupt number 40	W	0
7	PENDIRQ39	Masked pending interrupt number 39	W	0
6	PENDIRQ38	Masked pending interrupt number 38	W	0
5	PENDIRQ37	Masked pending interrupt number 37	W	0
4	PENDIRQ36	Masked pending interrupt number 36	W	0
3	PENDIRQ35	Masked pending interrupt number 35	W	0
2	PENDIRQ34	Masked pending interrupt number 34	W	0
1	PENDIRQ33	Masked pending interrupt number 33	W	0
0	PENDIRQ32	Masked pending interrupt number 32	W	0

Table 14-598. Register Call Summary for Register WUGEN_PENDEVTCLR1

IVA2.2 Subsystem Register Manual

- WUGEN Register Mapping Summary: [0]

Table 14-599. WUGEN_PENDEVTCLR2

Address Offset	0x108		
Physical address	0x01C2 1108	**Instance**	IVA2.2 WUGEN
Description	This register clears the masked pending dma_requests: Write 0: No effect Write 1: Clears the corresponding mask bit in the WUGEN_PENDEVT2 register Reads always return 0		
Type	W		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved												PENDDMARQ19	PENDDMARQ18	PENDDMARQ17	PENDDMARQ16	PENDDMARQ15	PENDDMARQ14	PENDDMARQ13	PENDDMARQ12	PENDDMARQ11	PENDDMARQ10	PENDDMARQ9	PENDDMARQ8	PENDDMARQ7	PENDDMARQ6	PENDDMARQ5	PENDDMARQ4	PENDDMARQ3	PENDDMARQ2	PENDDMARQ1	PENDDMARQ0

Bits	Field Name	Description	Type	Reset
31:20	Reserved	Write 0s for future compatibility.	W	0x000
19	PENDDMARQ19	Masked pending dma request number 0	W	0
18	PENDDMARQ18	Masked pending dma request number 0	W	0
17	PENDDMARQ17	Masked pending dma request number 0	W	0
16	PENDDMARQ16	Masked pending dma request number 0	W	0
15	PENDDMARQ15	Masked pending dma request number 0	W	0
14	PENDDMARQ14	Masked pending dma request number 0	W	0
13	PENDDMARQ13	Masked pending dma request number 0	W	0
12	PENDDMARQ12	Masked pending dma request number 0	W	0
11	PENDDMARQ11	Masked pending dma request number 0	W	0
10	PENDDMARQ10	Masked pending dma request number 0	W	0
9	PENDDMARQ9	Masked pending dma request number 0	W	0
8	PENDDMARQ8	Masked pending dma request number 0	W	0
7	PENDDMARQ7	Masked pending dma request number 0	W	0
6	PENDDMARQ6	Masked pending dma request number 0	W	0
5	PENDDMARQ5	Masked pending dma request number 0	W	0
4	PENDDMARQ4	Masked pending dma request number 0	W	0
3	PENDDMARQ3	Masked pending dma request number 0	W	0
2	PENDDMARQ2	Masked pending dma request number 0	W	0
1	PENDDMARQ1	Masked pending dma request number 0	W	0
0	PENDDMARQ0	Masked pending dma request number 0	W	0

Table 14-600. Register Call Summary for Register WUGEN_PENDEVTCLR2

IVA2.2 Subsystem Register Manual

- WUGEN Register Mapping Summary: [0]

14.5.9 SEQ Registers

This section provides information about the SEQ Module. Each register in the module is described separately below.

14.5.9.1 SEQ Register Mapping Summary

Table 14-601. SEQ Register Mapping Summary

Register Name	Type	Register Width (Bits)	Address Offset	Physical Address
SEQ_REVISION	R	32	0x0000 0000	0x0009 0000
SEQ_SYSCONFIG	RW	32	0x0000 0008	0x0009 0008
SEQ_IRQMASK	RW	32	0x0000 0040	0x0009 0040
SEQ_IRQCLR	W	32	0x0000 0044	0x0009 0044
SEQ_IRQSET	W	32	0x0000 0048	0x0009 0048
SEQ_IRQSTATE	R	32	0x0000 004C	0x0009 004C
SEQ_SWICLR	W	32	0x0000 0060	0x0009 0060
SEQ_SWISET	W	32	0x0000 0064	0x0009 0064
SEQ_SWISTATE	R	32	0x0000 0068	0x0009 0068

14.5.9.2 SEQ Register Descriptions

Table 14-602. SEQ_REVISION

Address Offset	0x0000 0000		
Physical Address	0x0009 0000	**Instance**	SEQ
Description	This register contains the IP revision code (reset value to be defined by design team for each version of the module)		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																								REV							

Bits	Field Name	Description	Type	Reset
31:8	RESERVED	Reserved Read returns 0	R	0x000000
7:0	REV	IP revision 3:0 Minor revision 7:4 Major revision	R	See (1)

(1) TI internal data

Table 14-603. Register Call Summary for Register SEQ_REVISION

IVA2.2 Subsystem Register Manual

- SEQ Register Mapping Summary: [0]

Table 14-604. SEQ_SYSCONFIG

Address Offset	0x0000 0008		
Physical Address	0x0009 0008	**Instance**	SEQ
Description	This register allows controlling various parameters of the sequencer module		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																															AUTOIDLE

Bits	Field Name	Description	Type	Reset
31:1	RESERVED	Reserved Write 0s for future compatibility Read returns 0	RW	0x00000000
0	AUTOIDLE	Internal auto-clock gating strategy 0: Clock is free running 1: Automatic clock gating strategy is applied	RW	0x1

Table 14-605. Register Call Summary for Register SEQ_SYSCONFIG

IVA2.2 Subsystem Basic Programming Model
- Video and Sequencer Module Management: [0]

IVA2.2 Subsystem Register Manual
- SEQ Register Mapping Summary: [1]

Table 14-606. SEQ_IRQMASK

Address Offset	0x0000 0040		
Physical Address	0x0009 0040	**Instance**	SEQ
Description	This register contains the interrupt mask bits: when SEQ_IRQMASK.MirqN is set, input event N does not trigger the interrupt line to the sequencer (default) when SEQ_IRQMASK.MirqN is clear, input event N triggers the interrupt line to the sequencer		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								TCERRINT1	TCERRINT0	CCERRINT	CCINT2	CCINT1	CCINT8	CCINT7	CCINT6	CCINT5	CCINT4	CCINT3	CCINTG	CCMPINT	RESERVED	HOST_MBX	SPARE_2	SPARE_1	SEQ_ERROR	DMA_ERROR	SPARE_0	RSERVED	iLF	iME	

Bits	Field Name	Description	Type	Reset
31:23	RESERVED	Reserved Reads returns 0 Write 0 for SW forward compatibility	RW	0x07
22	TCERRINT1	TCERRINT1 IRQ mask	w/1toSet	0x1
21	TCERRINT0	TCERRINT0 IRQ mask	w/1toSet	0x1
20	CCERRINT	CCERRINT IRQ mask	w/1toSet	0x1
19	CCINT2	CCINT2 IRQ mask	w/1toSet	0x1
18	CCINT1	CCINT1 IRQ mask	w/1toSet	0x1
17	CCINT8	CCINT8 IRQ mask	w/1toSet	0x1
16	CCINT7	CCINT7 IRQ mask	w/1toSet	0x1
15	CCINT6	CCINT6 IRQ mask	w/1toSet	0x1
14	CCINT5	CCINT5 IRQ mask	w/1toSet	0x1
13	CCINT4	CCINT4 IRQ mask	w/1toSet	0x1
12	CCINT3	CCINT3 IRQ mask	w/1toSet	0x1
11	CCINTG	CCINTG IRQ mask	w/1toSet	0x1
10	CCMPINT	CCMPINT IRQ mask	w/1toSet	0x1
9	RESERVED	Reserved	R	0x1
8	HOST_MBX	HOST_MBX IRQ mask	w/1toSet	0x1
7	SPARE_2	Spare #2 interrupt mask (reserved for future use) Reads returns 0 Write 0 for SW forward compatibility	w/1toSet	0x1

Bits	Field Name	Description	Type	Reset
6	SPARE_1	Spare #1 interrupt mask (reserved for future use) Reads returns 0 Write 0 for SW forward compatibility	w/1toSet	0x1
5	SEQ_ERROR	SEQ_ERROR IRQ mask	w/1toSet	0x1
4	DMA_ERROR	DMA_ERROR IRQ mask	w/1toSet	0x1
3	SPARE_0	Spare #0 interrupt mask (reserved for future use) Reads returns 0 Write 0 for SW forward compatibility	w/1toSet	0x1
2	RSERVED	Reserved	w/1toSet	0x1
1	iLF	iLF IRQ mask	w/1toSet	0x1
0	iME	iME IRQ mask	w/1toSet	0x1

Table 14-607. Register Call Summary for Register SEQ_IRQMASK

IVA2.2 Subsystem Functional Description

- Sequencer Interrupts: [0]
- IRQ Generic: [1] [2]

IVA2.2 Subsystem Register Manual

- SEQ Register Mapping Summary: [3]
- SEQ Register Descriptions: [4] [5] [6]
- Video System Controller Register Descriptions: [7]

Table 14-608. SEQ_IRQCLR

Address Offset	0x0000 0044		
Physical Address	0x0009 0044	**Instance**	SEQ
Description			
Type	W		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								TCERRINT1	TCERRINT0	CCERRINT	CCINT2	CCINT1	CCINT8	CCINT7	CCINT6	CCINT5	CCINT4	CCINT3	CCINTG	CCMPINT	RESERVED	HOST_MBX	SPARE_2	SPARE_1	SEQ_ERROR	DMA_ERROR	SPARE_0	RESERVED	iLF	iME	

Bits	Field Name	Description	Type	Reset
31:23	RESERVED	Reserved Reads returns 0 Write 0 for SW forward compatibility	W	0x0
22	TCERRINT1	TCERRINT1 IRQ clear	w/1toSet	0x0
21	TCERRINT0	TCERRINT0 IRQ clear	w/1toSet	0x0
20	CCERRINT	CCERRINT IRQ clear	w/1toSet	0x0
19	CCINT2	CCINT2 IRQ clear	w/1toSet	0x0
18	CCINT1	CCINT1 IRQ clear	w/1toSet	0x0
17	CCINT8	CCINT8 IRQ clear	w/1toSet	0x0
16	CCINT7	CCINT7 IRQ clear	w/1toSet	0x0
15	CCINT6	CCINT6 IRQ clear	w/1toSet	0x0
14	CCINT5	CCINT5 IRQ clear	w/1toSet	0x0
13	CCINT4	CCINT4 IRQ clear	w/1toSet	0x0
12	CCINT3	CCINT3 IRQ clear	w/1toSet	0x0
11	CCINTG	CCINTG IRQ clear	w/1toSet	0x0
10	CCMPINT	CCMPINT IRQ clear	w/1toSet	0x0

Bits	Field Name	Description	Type	Reset
9	RESERVED	Reserved	R	0x0
8	HOST_MBX	HOST_MBX IRQ clear	w/1toSet	0x0
7	SPARE_2	Spare #2 interrupt clear (reserved for future use) Reads returns 0 Write 0 for SW forward compatibility	w/1toSet	0x0
6	SPARE_1	Spare #1 interrupt clear (reserved for future use) Reads returns 0 Write 0 for SW forward compatibility	w/1toSet	0x0
5	SEQ_ERROR	SEQ_ERROR IRQ clear	w/1toSet	0x0
4	DMA_ERROR	DMA_ERROR IRQ clear	w/1toSet	0x0
3	SPARE_0	Spare #0 interrupt clear (reserved for future use) Reads returns 0 Write 0 for SW forward compatibility	w/1toSet	0x0
2	RESERVED	Reserved	w/1toSet	0x0
1	iLF	iLF IRQ clear	w/1toSet	0x0
0	iME	iME IRQ clear	w/1toSet	0x0

Table 14-609. Register Call Summary for Register SEQ_IRQCLR

IVA2.2 Subsystem Basic Programming Model

- Video and Sequencer Module interrupt Handling: [0]

IVA2.2 Subsystem Register Manual

- SEQ Register Mapping Summary: [1]

Table 14-610. SEQ_IRQSET

Address Offset	0x0000 0048		
Physical Address	0x0009 0048	**Instance**	SEQ
Description	This register is used to set the interrupt bits (used to test interrupt): write 0: no effect write 1: sets the corresponding bit in the SEQ_IRQSTATE register, and triggers the interrupt line if not already active and the associated event is enabled in SEQ_IRQMASK reads always return 0		
Type	W		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								TCERRINT1	TCERRINT0	CCERRINT	CCINT2	CCINT1	CCINT8	CCINT7	CCINT6	CCINT5	CCINT4	CCINT3	CCINTG	CCMPINT	RESERVED	HOST_MBX	SPARE_2	SPARE_1	SEQ_ERROR	DMA_ERROR	SPARE_0	RESERVED	iLF	iME	

Bits	Field Name	Description	Type	Reset
31:23	RESERVED	Reserved Reads returns 0 Write 0 for SW forward compatibility	W	0x0
22	TCERRINT1	TCERRINT1 IRQ set	w/1toSet	0x0
21	TCERRINT0	TCERRINT0 IRQ set	w/1toSet	0x0
20	CCERRINT	CCERRINT IRQ set	w/1toSet	0x0
19	CCINT2	CCINT2 IRQ set	w/1toSet	0x0
18	CCINT1	CCINT1 IRQ set	w/1toSet	0x0
17	CCINT8	CCINT8 IRQ set	w/1toSet	0x0
16	CCINT7	CCINT7 IRQ set	w/1toSet	0x0
15	CCINT6	CCINT6 IRQ set	w/1toSet	0x0
14	CCINT5	CCINT5 IRQ set	w/1toSet	0x0
13	CCINT4	CCINT4 IRQ set	w/1toSet	0x0
12	CCINT3	CCINT3 IRQ set	w/1toSet	0x0

Bits	Field Name	Description	Type	Reset
11	CCINTG	CCINTG IRQ set	w/1toSet	0x0
10	CCMPINT	CCMPINT IRQ set	w/1toSet	0x0
9	RESERVED	Reserved	R	0x0
8	HOST_MBX	HOST_MBX IRQ set	w/1toSet	0x0
7	SPARE_2	Spare #2 interrupt set (reserved for future use) Reads returns 0 Write 0 for SW forward compatibility	w/1toSet	0x0
6	SPARE_1	Spare #1 interrupt set (reserved for future use) Reads returns 0 Write 0 for SW forward compatibility	w/1toSet	0x0
5	SEQ_ERROR	SEQ_ERROR IRQ set	w/1toSet	0x0
4	DMA_ERROR	DMA_ERROR IRQ set	w/1toSet	0x0
3	SPARE_0	Spare #0 interrupt set (reserved for future use) Reads returns 0 Write 0 for SW forward compatibility	w/1toSet	0x0
2	RESERVED	Reserved	w/1toSet	0x0
1	iLF	iLF IRQ set	w/1toSet	0x0
0	iME	iME IRQ set	w/1toSet	0x0

Table 14-611. Register Call Summary for Register SEQ_IRQSET

IVA2.2 Subsystem Register Manual

- SEQ Register Mapping Summary: [0]

Table 14-612. SEQ_IRQSTATE

Address Offset	0x0000 004C		
Physical Address	0x0009 004C	**Instance**	SEQ
Description	This register holds the interrupt status bits		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								TCERRINT1	TCERRINT0	CCERRINT	CCINT2	CCINT1	CCINT8	CCINT7	CCINT6	CCINT5	CCINT4	CCINT3	CCINTG	CCMPINT	RESERVED	HOST_MBX	SPARE_2	SPARE_1	SEQ_ERROR	DMA_ERROR	SPARE_0	RESERVED	iLF	iME	

Bits	Field Name	Description	Type	Reset
31:23	RESERVED	Reserved Reads returns 0	R	0x0
22	TCERRINT1	TCERRINT1 IRQ status	R	0x0
21	TCERRINT0	TCERRINT0 IRQ status	R	0x0
20	CCERRINT	CCERRINT IRQ status	R	0x0
19	CCINT2	CCINT2 IRQ status	R	0x0
18	CCINT1	CCINT1 IRQ status	R	0x0
17	CCINT8	CCINT8 IRQ status	R	0x0
16	CCINT7	CCINT7 IRQ status	R	0x0
15	CCINT6	CCINT6 IRQ status	R	0x0
14	CCINT5	CCINT5 IRQ status	R	0x0
13	CCINT4	CCINT4 IRQ status	R	0x0
12	CCINT3	CCINT3 IRQ status	R	0x0
11	CCINTG	CCINTG IRQ status	R	0x0
10	CCMPINT	CCMPINT IRQ status	R	0x0

Bits	Field Name	Description	Type	Reset
9	RESERVED	Reserved	R	0x0
8	HOST_MBX	HOST_MBX IRQ status	R	0x0
7	SPARE_2	Spare #2 interrupt set (reserved for future use) Reads returns 0	R	0x0
6	SPARE_1	Spare #1 interrupt set (reserved for future use) Reads returns 0	R	0x0
5	SEQ_ERROR	SEQ_ERROR IRQ status	R	0x0
4	DMA_ERROR	DMA_ERROR IRQ status	R	0x0
3	SPARE_0	Spare #0 interrupt set (reserved for future use) Reads returns 0	R	0x0
2	RESERVED	Reserved	R	0x0
1	iLF	iLF IRQ status	R	0x0
0	iME	iME IRQ status	R	0x0

Table 14-613. Register Call Summary for Register SEQ_IRQSTATE

IVA2.2 Subsystem Functional Description
- Sequencer Interrupts: [0] [1]
- IRQ Generic: [2] [3]

IVA2.2 Subsystem Register Manual
- SEQ Register Mapping Summary: [4]
- SEQ Register Descriptions: [5]

Table 14-614. SEQ_SWICLR

Address Offset	0x0000 0060		
Physical Address	0x0009 0060	**Instance**	SEQ
Description	This register is used to clear the software interrupt bit		
Type	W		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																															SWICLR

Bits	Field Name	Description	Type	Reset
31:1	RESERVED	Reserved Write 0s for future compatibility	W	0x00000000
0	SWICLR	SW interrupt clear	w/1toSet	0x0

Table 14-615. Register Call Summary for Register SEQ_SWICLR

IVA2.2 Subsystem Functional Description
- Sequencer to DSP Megamodule Interrupts: [0]

IVA2.2 Subsystem Register Manual
- SEQ Register Mapping Summary: [1]

Table 14-616. SEQ_SWISET

Address Offset	0x0000 0064		
Physical Address	0x0009 0064	**Instance**	SEQ
Description	This register is used to set the software interrupt bit		
Type	W		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																															SWISET

Bits	Field Name	Description	Type	Reset
31:1	RESERVED	Reserved Write 0s for future compatibility	W	0x00000000
0	SWISET	SW interrupt set	w/1toSet	0x0

Table 14-617. Register Call Summary for Register SEQ_SWISET

IVA2.2 Subsystem Functional Description

- Sequencer to DSP Megamodule Interrupts: [0]

IVA2.2 Subsystem Register Manual

- SEQ Register Mapping Summary: [1]

Table 14-618. SEQ_SWISTATE

Address Offset	0x0000 0068		
Physical Address	0x0009 0068	**Instance**	SEQ
Description	This register holds the SW interrupt status		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																															SWISTATE

Bits	Field Name	Description	Type	Reset
31:1	RESERVED	Reserved Read returns 0	R	0x00000000
0	SWISTATE	SW interrupt status	w/1toSet	0x0

Table 14-619. Register Call Summary for Register SEQ_SWISTATE

IVA2.2 Subsystem Register Manual

- SEQ Register Mapping Summary: [0]

14.5.10 Video System Controller Registers

This section provides information about the Video System Controller module. Each register in the module is described separately below.

14.5.10.1 Video System Controller Register Mapping Summary

Table 14-620. VIDEOSYSC Register Mapping Summary

Register Name	Type	Register Width (Bits)	Address Offset	Physical Address
VIDEOSYSC_REVISION	R	32	0x0000 0000	0x0009 C000
VIDEOSYSC_SYSCONFIG	RW	32	0x0000 0008	0x0009 C008
VIDEOSYSC_IRQMASK	w/1toSet	32	0x0000 0040	0x0009 C040
VIDEOSYSC_IRQCLR	w/1toSet	32	0x0000 0044	0x0009 C044
VIDEOSYSC_IRQSET	w/1toSet	32	0x0000 0048	0x0009 C048
VIDEOSYSC_IRQSTATE	w/1toSet	32	0x0000 004C	0x0009 C04C
VIDEOSYSC_CLKCTL	W	32	0x0000 0060	0x0009 C060
VIDEOSYSC_CLKDIV	RW	32	0x0000 0064	0x0009 C064
VIDEOSYSC_CLKST	R	32	0x0000 0068	0x0009 C068

14.5.10.2 Video System Controller Register Descriptions

Table 14-621. VIDEOSYSC_REVISION

Address Offset	0x0000 0000		
Physical Address	0x0009 C000	**Instance**	VIDEOSYSC
Description	This register contains the IP revision code (reset value to be defined by design team for each version of the module)		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																								REV							

Bits	Field Name	Description	Type	Reset
31:8	RESERVED	Reserved Read returns 0	R	0x000000
7:0	REV	IP revision 3:0 Minor revision 7:4 Major revision	R	See (1)

(1) TI internal data

Table 14-622. Register Call Summary for Register VIDEOSYSC_REVISION

IVA2.2 Subsystem Register Manual

- Video System Controller Register Mapping Summary: [0]

Table 14-623. VIDEOSYSC_SYSCONFIG

Address Offset	0x0000 0008		
Physical Address	0x0009 C008	**Instance**	VIDEOSYSC
Description	This register allows controlling various parameters of the video system controller module		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																															AUTOIDLE

Bits	Field Name	Description	Type	Reset
31:1	RESERVED	Reserved Write 0s for future compatibility Read returns 0	RW	0x00000000
0	AUTOIDLE	Internal auto-clock gating strategy 0: Clock is free running 1: Automatic clock gating strategy is applied	RW	0x1

Table 14-624. Register Call Summary for Register VIDEOSYSC_SYSCONFIG

IVA2.2 Subsystem Basic Programming Model

- Video and Sequencer Module Management: [0]

IVA2.2 Subsystem Register Manual

- Video System Controller Register Mapping Summary: [1]
- Video System Controller Register Descriptions: [2]

Table 14-625. VIDEOSYSC_IRQMASK

Address Offset	0x0000 0040		
Physical Address	0x0009 C040	**Instance**	VIDEOSYSC
Description	This register contains the interrupt mask bits: when VIDEOSYSC_IRQMASK.MirqN is set, input event N does not trigger the interrupt line to the sequencer (default) when SEQ_IRQMASK.MirqN is clear, input event N triggers the interrupt line to the sequencer		
Type	w/1toSet		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																								SPARE_1	SEQ_MBX	DMA_ERROR	HOST_ERROR	SPARE_0	Reserved	iLF	iME

Bits	Field Name	Description	Type	Reset
31:8	RESERVED	Reserved (not implemented)	w/1toSet	0x000000
7	SPARE_1	Spare interrupt (reserved for future use)	w/1toSet	0x1
6	SEQ_MBX	SEQ Mailbox IRQ mask	w/1toSet	0x1
5	DMA_ERROR	DMA error IRQ mask	w/1toSet	0x1
4	HOST_ERROR	HOST error IRQ mask	w/1toSet	0x1
3	SPARE_0	Spare interrupt (reserved for future use)	w/1toSet	0x1
2	Reserved	Reserved	w/1toSet	0x1
1	iLF	iLF IRQ mask	w/1toSet	0x1
0	iME	iME IRQ mask	w/1toSet	0x1

Table 14-626. Register Call Summary for Register VIDEOSYSC_IRQMASK

IVA2.2 Subsystem Functional Description
- Video Accelerator/Sequencer SYSC: [0]

IVA2.2 Subsystem Register Manual
- Video System Controller Register Mapping Summary: [1]
- Video System Controller Register Descriptions: [2] [3] [4]

Table 14-627. VIDEOSYSC_IRQCLR

Address Offset	0x0000 0044		
Physical Address	0x0009 C044	**Instance**	VIDEOSYSC
Description	This register is used to clear the interrupt bits in VIDEOSYSC_IRQSTATE write 0: no effect write 1: clears the corresponding bit in the VIDEOSYSC_IRQSTATE register and clears the interrupt line if this action clears last active and enabled (in VIDEOSYSC_IRQMASK) input event(s). reads always return 0		
Type	w/1toSet		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																								SPARE_1	SEQ_MBX	DMA_ERROR	HOST_ERROR	SPARE_0	Reserved	iLF	iME

Bits	Field Name	Description	Type	Reset
31:8	RESERVED	Reserved (not implemented)	w/1toSet	0x000000
7	SPARE_1	Spare interrupt (reserved for future use)	w/1toSet	0x0
6	SEQ_MBX	SEQ Mailbox IRQ clear	w/1toSet	0x0
5	DMA_ERROR	DMA error IRQ clear	w/1toSet	0x0
4	HOST_ERROR	HOST error IRQ clear	w/1toSet	0x0
3	SPARE_0	Spare interrupt (reserved for future use)	w/1toSet	0x0
2	Reserved	Reserved	w/1toSet	0x0
1	iLF	iLF IRQ clear	w/1toSet	0x0
0	iME	iME IRQ clear	w/1toSet	0x0

Table 14-628. Register Call Summary for Register VIDEOSYSC_IRQCLR

IVA2.2 Subsystem Functional Description
- Video Accelerator/Sequencer SYSC: [0]

IVA2.2 Subsystem Basic Programming Model
- Video and Sequencer Module interrupt Handling: [1]

IVA2.2 Subsystem Register Manual
- Video System Controller Register Mapping Summary: [2]

Table 14-629. VIDEOSYSC_IRQSET

Address Offset	0x0000 0048		
Physical Address	0x0009 C048	**Instance**	VIDEOSYSC
Description	This register is used to set the interrupt bits (used to test interrupt): write 0: no effect write 1: sets the corresponding bit in the VIDEOSYSC_IRQSTATE register, and triggers the interrupt line if not already active and the associated event is enabled in VIDEOSYSC_IRQMASK reads always return 0		
Type	w/1toSet		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																								SPARE_1	SEQ_MBX	DMA_ERROR	HOST_ERROR	SPARE_0	Reserved	iLF	iME

Bits	Field Name	Description	Type	Reset
31:8	RESERVED	Reserved (not implemented)	w/1toSet	0x000000
7	SPARE_1	Spare interrupt (reserved for future use)	w/1toSet	0x0
6	SEQ_MBX	SEQ Mailbox IRQ set	w/1toSet	0x0
5	DMA_ERROR	DMA error IRQ set	w/1toSet	0x0
4	HOST_ERROR	HOST error IRQ set	w/1toSet	0x0
3	SPARE_0	Spare interrupt (reserved for future use)	w/1toSet	0x0
2	Reserved	Reserved	w/1toSet	0x0
1	iLF	iLF IRQ set	w/1toSet	0x0
0	iME	iME IRQ set	w/1toSet	0x0

Table 14-630. Register Call Summary for Register VIDEOSYSC_IRQSET

IVA2.2 Subsystem Functional Description

- Video Accelerator/Sequencer SYSC: [0]

IVA2.2 Subsystem Register Manual

- Video System Controller Register Mapping Summary: [1]

Table 14-631. VIDEOSYSC_IRQSTATE

Address Offset	0x0000 004C		
Physical Address	0x0009 C04C	**Instance**	VIDEOSYSC
Description	This register holds the interrupt status bits		
Type	w/1toSet		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																								SPARE_1	SEQ_MBX	DMA_ERROR	HOST_ERROR	SPARE_0	Reserved	iLF	iME

Bits	Field Name	Description	Type	Reset
31:8	RESERVED	Reserved (not implemented)	w/1toSet	0x000000
7	SPARE_1	Spare interrupt (reserved for future use)	w/1toSet	0x0
6	SEQ_MBX	SEQ Mailbox IRQ status	w/1toSet	0x0
5	DMA_ERROR	DMA error IRQ status	w/1toSet	0x0
4	HOST_ERROR	HOST error IRQ status	w/1toSet	0x0
3	SPARE_0	Spare interrupt (reserved for future use)	w/1toSet	0x0

Bits	Field Name	Description	Type	Reset
2	Reserved	Reserved	w/1toSet	0x0
1	iLF	iLF IRQ status	w/1toSet	0x0
0	iME	iME IRQ status	w/1toSet	0x0

Table 14-632. Register Call Summary for Register VIDEOSYSC_IRQSTATE

IVA2.2 Subsystem Functional Description
- Video Accelerator/Sequencer SYSC: [0] [1] [2] [3]

IVA2.2 Subsystem Basic Programming Model
- Video and Sequencer Module interrupt Handling: [4]

IVA2.2 Subsystem Register Manual
- Video System Controller Register Mapping Summary: [5]
- Video System Controller Register Descriptions: [6] [7] [8]

Table 14-633. VIDEOSYSC_CLKCTL

Address Offset	0x0000 0060		
Physical Address	0x0009 C060	**Instance**	VIDEOSYSC
Description	Video accelerator clock control: Writing a 0 forces the module to leave the idle state (modules input clock goes active) Writing a 1 requests the module enter the idle state when no request/commands pending for the module. (Allows module input clock to be stopped). Module automatically exits the idle state and clock starts each time new requests/commands are present. Clock status can be checked in VIDEOSYSC_CLKST register		
Type	W		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																										SL2IFCLKEN	SEQMEMCLKEN	RESERVED		iMECLKEN	iLFCLKEN

Bits	Field Name	Description	Type	Reset
31:6	RESERVED	Reserved Read returns 0	W	0x0
5	SL2IFCLKEN	Clock control of the SL2IF module (iLF and iME modules must be idled as well): 0: Exit idle state and start SL2IF clock 1: Request SL2IF logic to go to idle and stop SL2IF input clock	W	0x0
4	SEQMEMCLKEN	Clock control of the sequencer memory and slave port module: 0: Exit idle state and start SEQ memory and slave port clock 1: Request SEQ slave port logic to go to idle and stop SEQ clock (if SEQ in standby)	W	0x0
3:2	RESERVED	Reserved Read returns 0	W	0x0
1	iMECLKEN	Clock control of the iME module: 0: Exit idle state and start iME clock 1: Request iME logic to go to idle and stop iME input clock	W	0x0
0	iLFCLKEN	Clock control of the iLF module: 0: Exit idle state and start iLF clock 1: Request iLF logic to go to idle and stop iLF input clock	W	0x0

Table 14-634. Register Call Summary for Register VIDEOSYSC_CLKCTL

IVA2.2 Subsystem Functional Description
- Video Accelerator/Sequencer SYSC: [0] [1] [2] [3] [4]

Table 14-634. Register Call Summary for Register VIDEOSYSC_CLKCTL (continued)

IVA2.2 Subsystem Basic Programming Model • Video and Sequencer Module Management: [5] [6]
IVA2.2 Subsystem Register Manual • Video System Controller Register Mapping Summary: [7]

Table 14-635. VIDEOSYSC_CLKDIV

Address Offset	0x0000 0064		
Physical Address	0x0009 C064	**Instance**	VIDEOSYSC
Description	Video accelerator sequencer clock division		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																														SEQCLKDIV	

Bits	Field Name	Description	Type	Reset
31:2	RESERVED	Reserved Read returns 0	RW	0x00000000
1:0	SEQCLKDIV	Sequencer clock division control 0x0: No clock division - SEQ operates at the same clock as video accelerators 0x1: Clock divide by two - SEQ operates at half the video accelerators clock 0x2: Clock divide by three - SEQ operates at one third the video accelerators clock 0x3: Clock divide by four - SEQ operates at one fourth the video accelerators clock	RW	0x0

Table 14-636. Register Call Summary for Register VIDEOSYSC_CLKDIV

IVA2.2 Subsystem Functional Description • Video Accelerator/Sequencer SYSC: [0]
IVA2.2 Subsystem Basic Programming Model • Sequencer Boot/Reset: [1] • Video and Sequencer Module Management: [2] [3]
IVA2.2 Subsystem Register Manual • Video System Controller Register Mapping Summary: [4]

Table 14-637. VIDEOSYSC_CLKST

Address Offset	0x0000 0068		
Physical Address	0x0009 C068	**Instance**	VIDEOSYSC
Description	Video accelerator clock status: 0: Logic is active and the input modules input clock is running 1: Logic is idled (Clock stopped if autoidle bit also 1) Clock is only automatically stopped when logic is idled if the autoidle bit in VIDEOSYSC_SYSCONFIG register is set		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																							SEQSTANDBY	RESERVED		SL2IFCLKST	SEQMEMCLKST	RESERVED		iMECLKST	iLFCLKST

Bits	Field Name	Description	Type	Reset
31:9	RESERVED	Reserved Read returns 0	R	0x000000
8	SEQSTANDBY	SEQ is in Standby: 0: SEQ is not in standby 1: SEQ is in Standby	R	0x0
7:6	RESERVED	Reserved Read returns 0	R	0x0
5	SL2IFCLKST	Clock control of the SL2IF module 0: SL2IF logic is active 1: SL2IF logic is idled	R	0x0
4	SEQMEMCLKST	Clock control of the SEQ memory and slave port: 0: SEQ Slave port is active 1: SEQ Slave port is Idled	R	0x0
3:2	RESERVED	Reserved Read returns 0	R	0x0
1	iMECLKST	Clock control of the iME module: 0: iME logic is active 1: iME logic is idled	R	0x0
0	iLFCLKST	Clock control of the iLF module: 0: iLF logic is active 1: iLF logic is idled	R	0x0

Table 14-638. Register Call Summary for Register VIDEOSYSC_CLKST

IVA2.2 Subsystem Basic Programming Model

- Video and Sequencer Module Management: [0]

IVA2.2 Subsystem Register Manual

- Video System Controller Register Mapping Summary: [1]
- Video System Controller Register Descriptions: [2]

14.5.11 iME Registers

This section provides information about the improved Motion Estimation module. Each register in the module is described separately below.

14.5.11.1 iME Register Mapping Summary

Table 14-639. iME Register Mapping Summary

Register Name	Type	Register Width (Bits)	Address Offset	Physical Address
iME_REVISION	R	32	0x0000 0000	0x000A 0000
iME_SYSCONFIG	RW	32	0x0000 0010	0x000A 0010
iME_SYSSTATUS	R	32	0x0000 0014	0x000A 0014
iME_PROGRAMBUFFERLINENLSBi(1)	RW	32	0x0000 0040 + (0x8*i)	0x000A 0040 + (0x8*i)
iME_PROGRAMBUFFERLINENMSBi(1)	RW	32	0x0000 0044 + (0x8*i)	0x000A 0044 + (0x8*i)
iME_ERRORTABLEj(2)	RW	32	0x0000 0840 + (0x4*j)	0x000A 0840 + (0x4*j)
iME_REFERENCEBLOCKk(3)	RW	32	0x0000 0880 + (0x4*k)	0x000A 0880 + (0x4*k)
iME_COEFFREGBANKl(4)	RW	32	0x0000 0980 + (0x4*l)	0x000A 0980 + (0x4*l)
iME_PARAMETERSTACKLj(2)	RW	32	0x0000 0990 + (0x4*j)	0x000A 0990 + (0x4*j)
iME_PARAMETERSTACKHj(2)	RW	32	0x0000 09D0 + (0x4*j)	0x000A 09D0 + (0x4*j)
iME_XMVCTm(5)	RW	32	0x0000 0AA0 + (0x4*m)	0x000A 0AA0 + (0x4*m)
iME_YMVCTm(5)	RW	32	0x0000 0AC0 + (0x4*m)	0x000A 0AC0 + (0x4*m)
iME_MINERRORTHRESHOLD	RW	32	0x0000 0AE0	0x000A 0AE0
iME_ABSMINREACHED	RW	32	0x0000 0AE4	0x000A 0AE4
iME_CPUSTATUSREG	R	32	0x0000 0AE8	0x000A 0AE8
iME_IRQLOG	R	32	0x0000 0AEC	0x000A 0AEC
iME_LATESTERRORS	RW	32	0x0000 0AF0	0x000A 0AF0
iME_CONFIGREG	RW	32	0x0000 0AF4	0x000A 0AF4
iME_SL2INSTADDRESS	RW	32	0x0000 0AF8	0x000A 0AF8
iME_COMMANDREG	W	32	0x0000 0FFC	0x000A 0FFC

(1) i = 0 to 255
(2) j = 0 to 15
(3) k = 0 to 63
(4) l = 0 to 3
(5) m = 0 to 7

14.5.11.2 iME Register Descriptions

Table 14-640. iME_REVISION

Address Offset	0x0000 0000		
Physical Address	0x000A 0000	**Instance**	iME
Description	This register contains the iME revision code		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																								REV							

Bits	Field Name	Description	Type	Reset
31:8	RESERVED	Read returns 0	R	0x000000
7:0	REV	iME Revision [3:0] Minor Revision [7:4] Major Revision	R	See (1)

(1) TI internal data

Table 14-641. Register Call Summary for Register iME_REVISION

IVA2.2 Subsystem Register Manual

- iME Register Mapping Summary: [0]

Table 14-642. iME_SYSCONFIG

Address Offset	0x0000 0010		
Physical Address	0x000A 0010	**Instance**	iME
Description	This register allows controlling various parameters of the OCP interface		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																							CLOCKACTIVITY	RESERVED			SIDLEMODE		RESERVED	SOFTRESET	AUTOIDLE

Bits	Field Name	Description	Type	Reset
31:9	RESERVED	Read returns 0.	R	0x0
8	CLOCKACTIVITY	Clock activity during wake up mode period. 0 - OCP clock can be switched-off.	R	0x0
7:5	RESERVED	Read returns 0.	R	0x0
4:3	SIDLEMODE	Slave interface power management, req/ack control "10" = Smart-idle. Acknowledgement to an idle request is given based on the internal activity of iME	R	0x2
2	RESERVED	Read returns 0.	R	0x0
1	SOFTRESET	Software reset. Set this bit to 1 to trigger the iME reset. The bit is automatically reset by the hardware. During reads, it always returns 0.	RW	0x0
0	AUTOIDLE	Internal OCP clock gating strategy: 0: OCP clock is free running 1: Automatic OCP clock-gating strategy is applied based on the OCP interface activity	RW	0x1

Table 14-643. Register Call Summary for Register iME_SYSCONFIG

IVA2.2 Subsystem Basic Programming Model
- Video and Sequencer Module Management: [0]

IVA2.2 Subsystem Register Manual
- iME Register Mapping Summary: [1]

Table 14-644. iME_SYSSTATUS

Address Offset	0x0000 0014		
Physical Address	0x000A 0014	**Instance**	iME
Description	The register provides status information about the module, excluding the interrupt information.		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																															RESETDONE

Bits	Field Name	Description	Type	Reset
31:1	RESERVED	Reserved for OCP socket status information. Read returns 0.	R	0x00
0	RESETDONE	Internal reset monitoring 0: Internal module reset is on-going 1: Reset completed	R	0x-

Table 14-645. Register Call Summary for Register iME_SYSSTATUS

IVA2.2 Subsystem Register Manual
- iME Register Mapping Summary: [0]

Table 14-646. iME_PROGRAMBUFFERLINENLSBi

Address Offset	0x0000 0040 + (0x8*i)		
Physical Address	0x000A 0040 + (0x8*i)	**Instance**	iME
Description	Lower part of the macro-instruction : bits [31:0]		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MACROINST_LSB																															

Bits	Field Name	Description	Type	Reset
31:0	MACROINST_LSB	Line i of 256, Lower part of the macro-instruction : bits [31:0]	RW	0x--------

Table 14-647. Register Call Summary for Register iME_PROGRAMBUFFERLINENLSBi

IVA2.2 Subsystem Functional Description
- Instruction Set: [0]
- Parameters in L2: [1]
- Instructions: [2]

IVA2.2 Subsystem Basic Programming Model
- Typical Use: [3]

IVA2.2 Subsystem Register Manual
- iME Register Mapping Summary: [4]

Table 14-648. iME_PROGRAMBUFFERLINENMSBi

Address Offset	0x0000 0044 + (0x8*i)		
Physical Address	0x000A 0044 + (0x8*i)	**Instance**	iME
Description	High part of the macro-instruction : bits [54:32]		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								MACROINST_MSB																							

Bits	Field Name	Description	Type	Reset
31:23	Reserved	Read returns 0	R	0x--
22:0	MACROINST_MSB	Line i of 256, high part of the macro-instruction : bits [54:32]	RW	0x------

Table 14-649. Register Call Summary for Register iME_PROGRAMBUFFERLINENMSBi

IVA2.2 Subsystem Functional Description
- Instruction Set: [0]
- Parameters in L2: [1]
- Instructions: [2]

IVA2.2 Subsystem Basic Programming Model
- Typical Use: [3]

IVA2.2 Subsystem Register Manual
- iME Register Mapping Summary: [4]

Table 14-650. iME_ERRORTABLEj

Address Offset	0x0000 0840 + (0x4*j)		
Physical Address	0x000A 0840 + (0x4*j)	**Instance**	iME
Description	Register file (Error Table) for SAD computation containing final errors. Each entry in the register comprises a 16-bit Error field and a 16-bit Address field.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ET_ADDRESSN																ET_ERRORN															

Bits	Field Name	Description	Type	Reset
31:16	ET_ADDRESSN	Error Table line i, Address value	RW	0x----
15:0	ET_ERRORN	Error Table line i, Error value	RW	0x----

Table 14-651. Register Call Summary for Register iME_ERRORTABLEj

IVA2.2 Subsystem Functional Description
- Instructions: [0] [1]

IVA2.2 Subsystem Register Manual
- iME Register Mapping Summary: [2]

Table 14-652. iME_REFERENCEBLOCKk

Address Offset	0x0000 0880 + (0x4*k)		
Physical Address	0x000A 0880 + (0x4*k)	**Instance**	iME
Description	16 lines of 16 bytes register file containing the reference block data for SAD computation.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
REFBLOCK_WORD																															

Bits	Field Name	Description	Type	Reset
31:0	REFBLOCK_WORD	This word contains 4 pixels.	RW	0x--------

Table 14-653. Register Call Summary for Register iME_REFERENCEBLOCKk

IVA2.2 Subsystem Register Manual
- iME Register Mapping Summary: [0]

Table 14-654. iME_COEFFREGBANKl

Address Offset	0x0000 0980 + (0x4*l)		
Physical Address	0x000A 0980 + (0x4*l)	**Instance**	iME
Description	Coefficients Register Bank: The coefficient is 7 bit wide and 2 coefficients are stored in one word in the following format: [31:23] = 0x00 [22:16] = Coeff_odd [15:7] = 0x00 [6:0] = Coeff_even		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								COEFF_ODD								RESERVED								COEFF_EVEN							

Bits	Field Name	Description	Type	Reset
31:23	RESERVED	Read returns 0.	RW	0x000
22:16	COEFF_ODD	Coefficient (odd index)	RW	0x--
15:7	RESERVED	read returns 0.	RW	0x000
6:0	COEFF_EVEN	Coefficient (even index)	RW	0x--

Table 14-655. Register Call Summary for Register iME_COEFFREGBANKl

IVA2.2 Subsystem Functional Description
- Parameters: [0]

IVA2.2 Subsystem Register Manual
- iME Register Mapping Summary: [1]

Table 14-656. iME_PARAMETERSTACKLj

Address Offset	0x0000 0990 + (0x4*j)		
Physical Address	0x000A 0990 + (0x4*j)	**Instance**	iME
Description	Parameter stack register 0 to 15 (16-bit wide). Contains parameters used by program to control the iME units.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																PARAMSTACK															

Bits	Field Name	Description	Type	Reset
31:16	RESERVED	Read returns zero.	R	0x0000
15:0	PARAMSTACK	Parameter of 0 to 15	RW	0x----

Table 14-657. Register Call Summary for Register iME_PARAMETERSTACKLj

IVA2.2 Subsystem Register Manual

- iME Register Mapping Summary: [0]

Table 14-658. iME_PARAMETERSTACKHj

Address Offset	0x0000 09D0 + (0x4*j)		
Physical Address	0x000A 09D0 + (0x4*j)	**Instance**	iME
Description	Parameter stack register 16 to 31 (32-bit wide). Contains parameters used by program to control the iME units.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PARAMSTACK																															

Bits	Field Name	Description	Type	Reset
31:0	PARAMSTACK	Parameter of 16 to 31	RW	0x--------

Table 14-659. Register Call Summary for Register iME_PARAMETERSTACKHj

IVA2.2 Subsystem Register Manual

- iME Register Mapping Summary: [0]

Table 14-660. iME_XMVCTm

Address Offset	0x0000 0AA0 + (0x4*m)		
Physical Address	0x000A 0AA0 + (0x4*m)	**Instance**	iME
Description	Two Motion Vector Cost value (16-bits wide) are packed in one 32 bit word.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
XMVCT_ODD																XMVCT_EVEN															

Bits	Field Name	Description	Type	Reset
31:16	XMVCT_ODD	MV Cost value of odd index	RW	0x----
15:0	XMVCT_EVEN	MV Cost value of even index	RW	0x----

Table 14-661. Register Call Summary for Register iME_XMVCTm

IVA2.2 Subsystem Functional Description

- Instructions: [0]
- Parameters: [1]

IVA2.2 Subsystem Register Manual

- iME Register Mapping Summary: [2]

Table 14-662. iME_YMVCTm

Address Offset	0x0000 0AC0 + (0x4*m)		
Physical Address	0x000A 0AC0 + (0x4*m)	**Instance**	iME
Description	Two Motion Vector Cost value (16-bits wide) are packed in one 32 bit word.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
YMVCT_ODD																YMVCT_EVEN															

Bits	Field Name	Description	Type	Reset
31:16	YMVCT_ODD	MV Cost value of odd index	RW	0x----
15:0	YMVCT_EVEN	MV Cost value of even index	RW	0x----

Table 14-663. Register Call Summary for Register iME_YMVCTm

IVA2.2 Subsystem Functional Description
- Instructions: [0]
- Parameters: [1]

IVA2.2 Subsystem Register Manual
- iME Register Mapping Summary: [2]

Table 14-664. iME_MINERRORTHRESHOLD

Address Offset	0x0000 0AE0		
Physical Address	0x000A 0AE0	**Instance**	iME
Description	Minimum Error Threshold register, used in Mcomp() operator.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																MINTHRESHOLD															

Bits	Field Name	Description	Type	Reset
31:16	RESERVED	Read returns 0.	R	0x0000
15:0	MINTHRESHOLD	Min Threshold value in Mcomp() block.	RW	0x----

Table 14-665. Register Call Summary for Register iME_MINERRORTHRESHOLD

IVA2.2 Subsystem Register Manual
- iME Register Mapping Summary: [0]

Table 14-666. iME_ABSMINREACHED

Address Offset	0x0000 0AE4		
Physical Address	0x000A 0AE4	**Instance**	iME
Description	Absolute Minimum Reached bit register, used in Mcomp() operator.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																															ABSMINREACHED

Bits	Field Name	Description	Type	Reset
31:1	RESERVED	Read returns 0.	R	0x00000000
0	ABSMINREACHED	Abs Min Reached bit, in Mcomp block.	RW	0x-

Table 14-667. Register Call Summary for Register iME_ABSMINREACHED

IVA2.2 Subsystem Functional Description
- Instructions: [0]

Table 14-667. Register Call Summary for Register iME_ABSMINREACHED (continued)

IVA2.2 Subsystem Register Manual

- iME Register Mapping Summary: [1]

Table 14-668. iME_CPUSTATUSREG

Address Offset	0x0000 0AE8		
Physical Address	0x000A 0AE8	**Instance**	iME
Description	CPU Status Register provides information about the progress of the CPU execution		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED	DETECTEDENDOFPGM	DETECTEDSTOPSEQ	ENDPGMERROR	OPCODEERROR	WRITEREGERROR	EXECSTATE		PC								CYCLECOUNT															

Bits	Field Name	Description	Type	Reset
31	RESERVED	read returns 0.	R	0x0
30	DETECTEDENDOFPGM	This bit is set to '1' when in Debug mode, an EndOfPgm instruction or the last instruction of ProgramBuffer has been reached.	R	0x0
29	DETECTEDSTOPSEQ	This bit is set to '1' when a StopSeq() command has been issued and the iME is in Halted state.	R	0x0
28	ENDPGMERROR	This bit is set to '1' when the last instruction of the Program Buffer is reached, and no EndPgm() or LoadInstBuf() instruction have been detected in the program buffer. This bit is cleared by a StartSeq() command.	R	0x0
27	OPCODEERROR	This bit is set to '1' when a unknown opcode is decoded from the main program. This bit is cleared by StartSeq() command when in INITIALIZED state, or by a Stop() command.	R	0x0
26	WRITEREGERROR	This bit is set to '1' when attempting to write to an internal register through an OCP Write, while in EXECUTING state. This bit is cleared by StartSeq() command when in INITIALIZED state, or by a Stop() command.	R	0x0
25:24	EXECSTATE	Execution States: 00 = Initialized, 10 = Executing, 01 = Halted, 11 = Completed.	R	0x0
23:16	PC	Number of instruction currently executing.	R	0x00
15:0	CYCLECOUNT	Total number of cycles executed.	R	0x0000

Table 14-669. Register Call Summary for Register iME_CPUSTATUSREG

IVA2.2 Subsystem Functional Description

- iLF/iME Control: [0] [1]

IVA2.2 Subsystem Register Manual

- iME Register Mapping Summary: [2]

Table 14-670. iME_IRQLOG

Address Offset	0x0000 0AEC		
Physical Address	0x000A 0AEC	**Instance**	iME
Description	IRQ Log register captures a one on bit 0 if the endpgm() instruction has been executed and a one on bits 1 to 15 for the first 15 GenerateIT() instructions executed (beyond 15, GenerateIT() events are logged into bit 15)		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																GENEITEVENTLOG															ENDPGMEVENTLOG

Bits	Field Name	Description	Type	Reset
31:16	RESERVED	Read returns 0.	R	0x0000
15:1	GENEITEVENTLOG	GenerateIT() instructions event log	R	0x0000
0	ENDPGMEVENTLOG	endpgm() instruction event log.	R	0x0

Table 14-671. Register Call Summary for Register iME_IRQLOG

IVA2.2 Subsystem Functional Description
- Instructions: [0]

IVA2.2 Subsystem Register Manual
- iME Register Mapping Summary: [1]

Table 14-672. iME_LATESTERRORS

Address Offset	0x0000 0AF0		
Physical Address	0x000A 0AF0	**Instance**	iME
Description	Best Match Location (minimum error) data and its address generated by Multi compare unit		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BESTMATCHADDRESS																BESTMATCHERRORVALUE															

Bits	Field Name	Description	Type	Reset
31:16	BESTMATCHADDRESS	Best Match Error Address	RW	0x----
15:0	BESTMATCHERRORVALUE	Best Match Error Value	RW	0x----

Table 14-673. Register Call Summary for Register iME_LATESTERRORS

IVA2.2 Subsystem Functional Description
- Instructions: [0] [1] [2] [3] [4] [5] [6] [7] [8]
- Parameters: [9] [10]

IVA2.2 Subsystem Register Manual
- iME Register Mapping Summary: [11]

Table 14-674. iME_CONFIGREG

Address Offset	0x0000 0AF4		
Physical Address	0x000A 0AF4	**Instance**	iME
Description	Configuration Register		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																								WAITING_FOR_SYNC_SIGNAL	SYNC_SIGNAL_SET	RESERVED			DEBUGMODESTATUS	MINTHRESHOLDEN	ITENABLE

Bits	Field Name	Description	Type	Reset
31:7	RESERVED	read returns 0.	R	0x000000
6	WAITING_FOR_SYNC_SIGNAL	Bit is set when the WaitForSignal instruction has been decoded.	R	0
5	SYNC_SIGNAL_SET	Bit is set when iME has received the synchronisation signal.	R	0
4:3	RESERVED	read returns 0.	R	0x0
2	DEBUGMODESTATUS	"Debug Mode" status bit	R	0x0
1	MINTHRESHOLDEN	Enable Min Threshold Comparison bit	RW	0x0
0	ITENABLE	Interrupt Enable bit	RW	0x0

Table 14-675. Register Call Summary for Register iME_CONFIGREG

IVA2.2 Subsystem Functional Description

- iME Control: [0] [1]
- Instructions: [2] [3] [4] [5] [6]

IVA2.2 Subsystem Register Manual

- iME Register Mapping Summary: [7]

Table 14-676. iME_SL2INSTADDRESS

Address Offset	0x0000 0AF8		
Physical Address	0x000A 0AF8	**Instance**	iME
Description	This register contains the SL2 address passed in the instruction.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																SL2INSTADDRESS															

Bits	Field Name	Description	Type	Reset
31:16	RESERVED	Read returns 0.	R	0x0000
15:0	SL2INSTADDRESS	This register contains the SL2 address passed in the instruction.	RW	0x0000

Table 14-677. Register Call Summary for Register iME_SL2INSTADDRESS

IVA2.2 Subsystem Register Manual

- iME Register Mapping Summary: [0]

Table 14-678. iME_COMMANDREG

Address Offset	0x0000 0FFC		
Physical Address	0x000A 0FFC	**Instance**	iME
Description	iME command register: a write to this register decodes a command, a read returns an error. 0x1 -> StartSeq() 0x2 -> StopSeq() 0x3 -> DbgEnable() 0x4 -> DbgDisable() 0x5 -> DbgStep() 0x6 -> Halt()		
Type	W		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																														CMD	

Bits	Field Name	Description	Type	Reset
31:3	RESERVED	Read returns an error	W	0x--------
2:0	CMD	DATA/COMMAND 0x1 -> StartSeq() 0x2 -> StopSeq() 0x3 -> DbgEnable() 0x4 -> DbgDisable() 0x5 -> DbgStep() 0x6 -> Halt() 0x7 -> Sync()	W	0x-

Table 14-679. Register Call Summary for Register iME_COMMANDREG

IVA2.2 Subsystem Functional Description

- iLF/iME Control: [0] [1]
- Instructions: [2]

IVA2.2 Subsystem Basic Programming Model

- Typical Use: [3]

IVA2.2 Subsystem Register Manual

- iME Register Mapping Summary: [4]

14.5.12 iLF Registers

This section provides information about the improved Loop Filter module. Each register in the module is described separately below.

14.5.12.1 iLF Register Mapping Summary

Table 14-680. iLF Register Mapping Summary

Register Name	Type	Register Width (Bits)	Address Offset	Physical Address
iLF_REVISION	R	32	0x0000 0000	0x000A 1000
iLF_SYSCONFIG	RW	32	0x0000 0010	0x000A 1010
iLF_SYSSTATUS	R	32	0x0000 0014	0x000A 1014
iLF_PROGRAMBUFFERLINENLSBi(1)	RW	32	0x0000 0040 + (0x8*i)	0x000A 1040 + (0x8*i)
iLF_PROGRAMBUFFERLINENMSBi(1)	RW	32	0x0000 0044 + (0x8*i)	0x000A 1044 + (0x8*i)
iLF_PARAMETERSTACKUPj(2)	RW	32	0x0000 0440 + (0x4*j)	0x000A 1440 + (0x4*j)
iLF_PARAMETERSTACKLWk(3)	RW	32	0x0000 0460 + (0x4*k)	0x000A 1460 + (0x4*k)
iLF_EFPTABLEENTRYl(4)	RW	32	0x0000 04C0 + (0x4*l)	0x000A 14C0 + (0x4*l)
iLF_INOUTBUFFERm(5)	RW	32	0x0000 0550 + (0x4*m)	0x000A 1550 + (0x4*m)
iLF_CPUSTATUSREG	R	32	0x0000 05F0	0x000A 15F0
iLF_IRQLOG	R	32	0x0000 05F4	0x000A 15F4
iLF_EFPTD	R	32	0x0000 05F8	0x000A 15F8
iLF_CONFIGREG	RW	32	0x0000 05FC	0x000A 15FC
iLF_PARSEDDATAREG0	RW	32	0x0000 0600	0x000A 1600
iLF_PARSEDDATAREG1	RW	32	0x0000 0604	0x000A 1604
iLF_PARSEDDATAREG2	RW	32	0x0000 0608	0x000A 1608
iLF_INSTBUFFER_ADDRESS	RW	32	0x0000 060C	0x000A 160C
iLF_LINESFILTERPROTOTYPES	R	32	0x0000 0610	0x000A 1610
iLF_CLIPLIMITSENTRYn(6)	R	32	0x0000 0614 + (0x4*n)	0x000A 1614 + (0x4*n)
iLF_COMMANDREG	W	32	0x0000 0FFC	0x000A 1FFC

(1) i = 0 to 127
(2) j = 0 to 7
(3) k = 0 to 23
(4) l = 0 to 35
(5) m = 0 to 39
(6) n = 0 to 3

14.5.12.2 iLF Register Descriptions

Table 14-681. iLF_REVISION

Address Offset	0x0000 0000		
Physical Address	0x000A 1000	**Instance**	iLF
Description	This register contains the iLF revision code		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																								REV							

Bits	Field Name	Description	Type	Reset
31:8	RESERVED	Read returns 0	R	0x000000
7:0	REV	iLF Revision [3:0] Minor Revision [7:4] Major Revision	R	See (1)

(1) TI internal data

Table 14-682. Register Call Summary for Register iLF_REVISION

IVA2.2 Subsystem Register Manual

- iLF Register Mapping Summary: [0]

Table 14-683. iLF_SYSCONFIG

Address Offset	0x0000 0010		
Physical Address	0x000A 1010	**Instance**	iLF
Description	This register allows controlling various parameters of the OCP interface		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																							CLOCKACTIVITY	RESERVED			SIDLEMODE		RESERVED	SOFTRESET	AUTOIDLE

Bits	Field Name	Description	Type	Reset
31:9	RESERVED	Read returns 0.	R	0x0
8	CLOCKACTIVITY	Clock activity during wake up mode period. 0 - OCP clock can be switched-off.	R	0x0
7:5	RESERVED	read returns 0.	R	0x0
4:3	SIDLEMODE	Slave interface power management, req/ack control "10" = Smart-idle. Acknowledgement to an idle request is given based on the internal activity of iME	R	0x2
2	RESERVED	read returns 0.	R	0x0
1	SOFTRESET	Software reset. Set this bit to 1 to trigger the iME reset. The bit is automatically reset by the hardware. During reads, it always returns 0.	RW	0x0
0	AUTOIDLE	Internal OCP clock gating strategy: 0: OCP clock is free running 1: Automatic OCP clock-gating strategy is applied based on the OCP interface activity	RW	0x1

Table 14-684. Register Call Summary for Register iLF_SYSCONFIG

IVA2.2 Subsystem Basic Programming Model
- Video and Sequencer Module Management: [0]

IVA2.2 Subsystem Register Manual
- iLF Register Mapping Summary: [1]

Table 14-685. iLF_SYSSTATUS

Address Offset	0x0000 0014		
Physical Address	0x000A 1014	**Instance**	iLF
Description	The register provides status information about the module, excluding the interrupt information.		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																															RESETDONE

Bits	Field Name	Description	Type	Reset
31:1	RESERVED	Reserved for OCP socket status information. Read returns 0.	R	0x00
0	RESETDONE	Internal reset monitoring 0: Internal module reset is on-going 1: Reset completed	R	0x-

Table 14-686. Register Call Summary for Register iLF_SYSSTATUS

IVA2.2 Subsystem Register Manual
- iLF Register Mapping Summary: [0]

Table 14-687. iLF_PROGRAMBUFFERLINENLSBi

Address Offset	0x0000 0040 + (0x8*i)		
Physical Address	0x000A 1040 + (0x8*i)	**Instance**	iLF
Description	Lower part of the macro-instruction : bits [31:0]		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MACROINST_LSB																															

Bits	Field Name	Description	Type	Reset
31:0	MACROINST_LSB	Line i of 128, Lower part of the macro-instruction : bits [31:0]	RW	0x--------

Table 14-688. Register Call Summary for Register iLF_PROGRAMBUFFERLINENLSBi

IVA2.2 Subsystem Functional Description
- Instruction Set: [0]
- Parameters in L2: [1]
- Instructions: [2]

IVA2.2 Subsystem Basic Programming Model
- Typical Use: [3]

IVA2.2 Subsystem Register Manual
- iLF Register Mapping Summary: [4]

Table 14-689. iLF_PROGRAMBUFFERLINENMSBi

Address Offset	0x0000 0044 + (0x8*i)		
Physical Address	0x000A 1040 + (0x8*i)	**Instance**	iLF
Description	High part of the macro-instruction : bits [54:32]		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								MACROINST_MSB																							

Bits	Field Name	Description	Type	Reset
31:23	Reserved	Read returns 0	R	0x--
22:0	MACROINST_MSB	Line i of 128, high part of the macro-instruction : bits [54:32]	RW	0x------

Table 14-690. Register Call Summary for Register iLF_PROGRAMBUFFERLINENMSBi

IVA2.2 Subsystem Functional Description
- Instruction Set: [0]
- Parameters in L2: [1]
- Instructions: [2]

IVA2.2 Subsystem Basic Programming Model
- Typical Use: [3]

IVA2.2 Subsystem Register Manual
- iLF Register Mapping Summary: [4]

Table 14-691. iLF_PARAMETERSTACKUPj

Address Offset	0x0000 0440 + (0x4*j)		
Physical Address	0x000A 1440 + (0x4*j)	**Instance**	iLF
Description	parameter stack register file contains parameters used by program to control the iLF units.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																PARAMSTACK															

Bits	Field Name	Description	Type	Reset
31:16	RESERVED	Read returns 0.	R	0x----
15:0	PARAMSTACK	Parameter of 0 to 7	RW	0x----

Table 14-692. Register Call Summary for Register iLF_PARAMETERSTACKUPj

IVA2.2 Subsystem Register Manual
- iLF Register Mapping Summary: [0]

Table 14-693. iLF_PARAMETERSTACKLWk

Address Offset	0x0000 0460 + (0x4*k)		
Physical Address	0x000A 1460 + (0x4*k)	**Instance**	iLF
Description	parameter stack register file contains parameters used by program to control the iLF units.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PARAMSTACK																															

Bits	Field Name	Description	Type	Reset
31:0	PARAMSTACK	Parameter of 8 to 31	RW	0x--------

Table 14-694. Register Call Summary for Register iLF_PARAMETERSTACKLWk

IVA2.2 Subsystem Register Manual

- iLF Register Mapping Summary: [0]

Table 14-695. iLF_EFPTABLEENTRYl

Address Offset	0x0000 04C0 + (0x4*l)		
Physical Address	0x000A 14C0 + (0x4*l)	**Instance**	iLF
Description	EFP Table entry: contains various data organizations, depending on the standard.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED	TABLE_ENTRY																														

Bits	Field Name	Description	Type	Reset
31	RESERVED	Read returns 0.	R	0x-
30:0	TABLE_ENTRY	Entry i of the table	RW	0x--------

Table 14-696. Register Call Summary for Register iLF_EFPTABLEENTRYl

IVA2.2 Subsystem Register Manual

- iLF Register Mapping Summary: [0]

Table 14-697. iLF_INOUTBUFFERm

Address Offset	0x0000 0550 + (0x4*m)		
Physical Address	0x000A 1550 + (0x4*m)	**Instance**	iLF
Description	32-bit entry in the buffer. Contains 4 8-bit fields.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IOFB_BYTE3								IOFB_BYTE2								IOFB_BYTE1								IOFB_BYTE0							

Bits	Field Name	Description	Type	Reset
31:24	IOFB_BYTE3	High order byte in each entry of the IO Filter buffer	RW	0x--
23:16	IOFB_BYTE2	3rd low order byte in each entry of the IO Filter buffer	RW	0x--
15:8	IOFB_BYTE1	2nd low order byte in each entry of the IO Filter buffer	RW	0x--
7:0	IOFB_BYTE0	low order byte in each entry of the IO filter buffer	RW	0x--

Table 14-698. Register Call Summary for Register iLF_INOUTBUFFERm

IVA2.2 Subsystem Register Manual

- iLF Register Mapping Summary: [0]

Table 14-699. iLF_CPUSTATUSREG

Address Offset	0x0000 05F0		
Physical Address	0x000A 15F0	**Instance**	iLF
Description	CPU Status Register provides information about the progress of the CPU execution		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED	DETECTEDENDOFPGM	DETECTEDSTOPSEQ	ENDPGMERROR	OPCODEERROR	WRITEREGERROR	EXECSTATE		PC								CYCLECOUNT															

Bits	Field Name	Description	Type	Reset
31	RESERVED	Read returns 0.	R	0x0
30	DETECTEDENDOFPGM	This bit is set to '1' when, in Debug mode, an EndPgm() instruction or the last instruction of the program buffer has been reached and it is not a LoadInstBuf() instruction. This bit is cleared by a StartSeq() command when in INITIALIZED or COMPLETED state.	R	0x0
29	DETECTEDSTOPSEQ	This bit is set to '1' when a StopSeq() has been issued and the iME is in Halted state. Execution of StopSeq() is postponed to the time when the CPU gets out of the Halted state. This bit is cleared when the StopSeq() command is effectively executed.	R	0x0
28	ENDPGMERROR	This bit is set to '1' when the last instruction of the program buffer is reached and no EndPgm() or LoadInstBuf() instruction have been detected in the program buffer. This bit is cleared by a StartSeq() command when in INITIALIZED or COMPLETED state.	R	0x0
27	OPCODEERROR	This bit is set to '1' when an unknown opcode is decoded from the main program. The following instruction of the sequence is then executed. This bit is cleared by a StartSeq() command when in INITIALIZED or COMPLETED state.	R	0x0
26	WRITEREGERROR	This bit is set to '1' when attempting to write an internal register through an OCP write, while in EXECUTING state. This bit is cleared by a StartSeq() command when in INITIALIZED or COMPLETED state.	R	0x0
25:24	EXECSTATE	Execution States: 00 = Initialized, 10 = Executing, 01 = Halted, 11 = Completed.	R	0x0
23:16	PC	Number of instruction currently executing. Bit #7 is always 0 (PC values range from 0x00 to 0x7f)	R	0x00
15:0	CYCLECOUNT	Total number of cycles executed.	R	0x0000

Table 14-700. Register Call Summary for Register iLF_CPUSTATUSREG

IVA2.2 Subsystem Functional Description

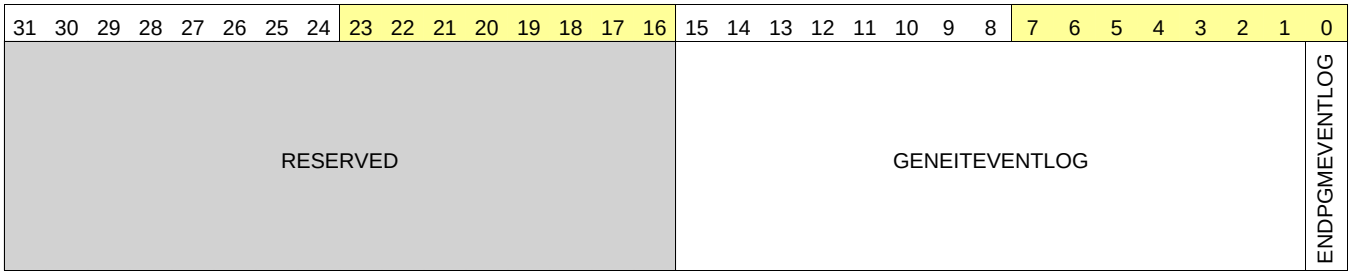
- iLF/iME Control: [0] [1]

IVA2.2 Subsystem Register Manual

- iLF Register Mapping Summary: [2]

Table 14-701. iLF_IRQLOG

Address Offset	0x0000 05F4		
Physical Address	0x000A 15F4	**Instance**	iLF
Description	IRQ Log register captures a one on bit 0 if the endpgm() instruction has been executed and a one on bits 1 to 15 for the first 15 GenerateIT() instructions executed (beyond 15, GenerateIT() events are logged into bit 15)		
Type	R		



Bits	Field Name	Description	Type	Reset
31:16	RESERVED	Read returns 0.	R	0x0000
15:1	GENEITEVENTLOG	GenerateIT() instructions event log	R	0x0000
0	ENDPGMEVENTLOG	endpgm() instruction event log.	R	0x0

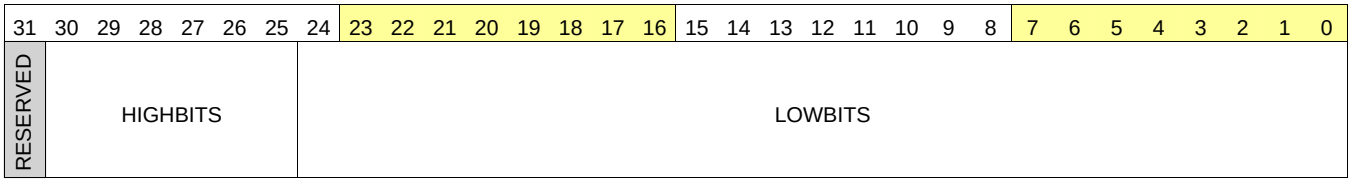
Table 14-702. Register Call Summary for Register iLF_IRQLOG

IVA2.2 Subsystem Register Manual

- iLF Register Mapping Summary: [0]

Table 14-703. iLF_EFPTD

Address Offset	0x0000 05F8		
Physical Address	0x000A 15F8	**Instance**	iLF
Description	32-bit generic data extracted from EFPT		
Type	R		



Bits	Field Name	Description	Type	Reset
31	RESERVED	read returns 0.	R	0x-
30:25	HIGHBITS	5 bit bitfield, extracted with a second address (EFP_A2), out of EFPT	R	0x--
24:0	LOWBITS	25 bit bitfield, extracted with a first address (EFP_A1), out of EFPT	R	0x-------

Table 14-704. Register Call Summary for Register iLF_EFPTD

IVA2.2 Subsystem Register Manual

- iLF Register Mapping Summary: [0]

Table 14-705. iLF_CONFIGREG

Address Offset	0x0000 05FC		
Physical Address	0x000A 15FC	**Instance**	iLF
Description	Configuration Register		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																													DEBUGHALTEN	RESERVED	ITENABLE

Bits	Field Name	Description	Type	Reset
31:3	RESERVED	read returns 0.	R	0x0000000
2	DEBUGHALTEN	"Debug Halt" control bit	R	0x0
1	RESERVED	read returns 0.	R	0x0
0	ITENABLE	Interrupt Enable bit	RW	0x0

Table 14-706. Register Call Summary for Register iLF_CONFIGREG

IVA2.2 Subsystem Register Manual

- iLF Register Mapping Summary: [0]

Table 14-707. iLF_PARSEDDATAREG0

Address Offset	0x0000 0600 in 0xc byte increments		
Physical Address	0x000A 1600	**Instance**	iLF
Description	Lower part of the Loop Filter parameters set		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED	TC0_FIELD					RESERVED	TC0B_FIELD					BETA2_FIELD							BETA_FIELD					ALPHA_FIELD							

Bits	Field Name	Description	Type	Reset
31	RESERVED	read returns 0.	R	0x-
30:26	TC0_FIELD	Tc0 parameter, 5 bits, unsigned	RW	0x--
25	RESERVED	read returns 0.	RW	0x--
24:20	TC0B_FIELD	Tc0B parameter, 5 bits, unsigned	RW	0x--
19:13	BETA2_FIELD	Beta2 parameter, 7 bits, unsigned	RW	0x--
12:8	BETA_FIELD	Beta parameter, 5 bits, unsigned	RW	0x--
7:0	ALPHA_FIELD	Alpha parameter, 8 bits, unsigned	RW	0x--

Table 14-708. Register Call Summary for Register iLF_PARSEDDATAREG0

IVA2.2 Subsystem Register Manual

- iLF Register Mapping Summary: [0]

Table 14-709. iLF_PARSEDDATAREG1

Address Offset	0x604-0x604 in 0xC byte increments		
Physical Address	0x5E0A 1604-0x5E0A 1604	**Instance**	iLF
Description	Mid part of the Loop Filter parameters set		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED			H264_BSB			REAL9_FLT	REAL9_STR	H264_BS			CFG					EFFEDGE_FIELD								CR_FIELD				CL_FIELD			

Bits	Field Name	Description	Type	Reset
31:29	RESERVED	Read returns 0.	R	0x--
28:26	H264_BSB	H264 Secondary Strength for Chroma	R	0x--
25	REAL9_FLT	Filter Flag for REAL9	RW	-
24	REAL9_STR	Strong bit for REAL9	RW	-
23:21	H264_BS	H264 Strength	RW	0x-
20:13	CFG	Configuration data, 8 bits: Bits 1:0: CMR (Condition Mode), unsigned Bit 2: Luma, boolean (0,1) Bits 7:3: Pquant, unsigned (WMV9/H263)	RW	0x--
12:8	EFFEDGE_FIELD	Effective Edge Number parameter, 5 bits: Bit 4: Vertical/Horizontal orientation, boolean (0,1) Bits 3 down to 0: Effective edge number, unsigned	RW	0x--
7:4	CR_FIELD	Clipping limit, right parameter, 4 bits, unsigned	RW	0x-
3:0	CL_FIELD	Clipping limit, left parameter, 4 bits, unsigned	RW	0x-

Table 14-710. Register Call Summary for Register iLF_PARSEDDATAREG1

IVA2.2 Subsystem Functional Description

- Parameters: [0]

IVA2.2 Subsystem Register Manual

- iLF Register Mapping Summary: [1]

Table 14-711. iLF_PARSEDDATAREG2

Address Offset	0x608-0x608 in 0xC byte increments		
Physical Address	0x5E0A 1608-0x5E0A 1608	**Instance**	iLF
Description	Higher part of the Loop Filter parameters set, used for REAL9 only.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								DTER3			DTER2			DTER1			DTER0			DTEL3			DTEL2			DTEL1			DTEL0		

Bits	Field Name	Description	Type	Reset
31:24	RESERVED	Read returns 0.	R	0x--
23:21	DTER3	Dither right, set 3	RW	0x-
20:18	DTER2	Dither right, set 2	RW	0x-
17:15	DTER1	Dither right, set 1	RW	0x-

Bits	Field Name	Description	Type	Reset
14:12	DTER0	Dither right, set 0	RW	0x-
11:9	DTEL3	Dither left, set 3	RW	0x-
8:6	DTEL2	Dither left, set 2	RW	0x-
5:3	DTEL1	Dither left, set 1	RW	0x-
2:0	DTEL0	Dither left, set 0	RW	0x-

Table 14-712. Register Call Summary for Register iLF_PARSEDDATAREG2

IVA2.2 Subsystem Register Manual

- iLF Register Mapping Summary: [0]

Table 14-713. iLF_INSTBUFFER_ADDRESS

Address Offset	0x0000 060C		
Physical Address	0x000A 160C	**Instance**	iLF
Description	The register provides status information about the module, excluding the interrupt information.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																SL2_BUFFPAGE_ADDRESS															

Bits	Field Name	Description	Type	Reset
31:16	RESERVED	Read returns 0.	R	0x0000
15:0	SL2_BUFFPAGE_ADDRESS	Page address of the Instruction buffer content in SL2 memory	RW	0x0000

Table 14-714. Register Call Summary for Register iLF_INSTBUFFER_ADDRESS

IVA2.2 Subsystem Register Manual

- iLF Register Mapping Summary: [0]

Table 14-715. iLF_LINESFILTERPROTOTYPES

Address Offset	0x0000 0610		
Physical Address	0x000A 1610	**Instance**	iLF
Description	Lines Filter Prototypes : contains, for each line of pixel orthogonal to the edge, the filter prototype reference on Right and Left side of the edge: LFPC_Qi and LFPC_Pi. These references are indexes into the Filter tables of the parameter stack, to extract all filter parameters. Status register that can be used to reconstruct the filter structure used at any place, along the edge.		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
LFPC_Q3				LFPC_P3				LFPC_Q2				LFPC_P2				LFPC_Q1				LFPC_P1				LFPC_Q0				LFPC_P0			

Bits	Field Name	Description	Type	Reset
31:28	LFPC_Q3	Filter on the Q side, line 3	R	0x0
27:24	LFPC_P3	Filter on the P side, line 3	R	0x0
23:20	LFPC_Q2	Filter on the Q side, line 2	R	0x0
19:16	LFPC_P2	Filter on the P side, line 2	R	0x0
15:12	LFPC_Q1	Filter on the Q side, line 1	R	0x0
11:8	LFPC_P1	Filter on the P side, line 1	R	0x0
7:4	LFPC_Q0	Filter on the Q side, line 0	R	0x0
3:0	LFPC_P0	Filter on the P side, line 0	R	0x0

Table 14-716. Register Call Summary for Register iLF_LINESFILTERPROTOTYPES

IVA2.2 Subsystem Register Manual

- iLF Register Mapping Summary: [0]

Table 14-717. iLF_CLIPLIMITSENTRYn

Address Offset	0x0000 0614 + (0x4*n)		
Physical Address	0x000A 1614 + (0x4*n)	**Instance**	iLF
Description	Clip limits entry: contains clip limits, organized in 3 fields, depending on the standard Clip_A: 9 bits, signed, Clip_B: 5 bits, unsigned, Clip_C: 4 bits, unsigned.		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED												SRUNQ	SRUNP	CLIP_C				CLIP_B						CLIP_A							

Bits	Field Name	Description	Type	Reset
31:20	RESERVED	read returns 0.	R	0x000
19	SRUNQ	1 bit, unsigned Value when not used (according to the standard): 0	R	0x0
18	SRUNP	1 bit, unsigned Value when not used (according to the standard): 0	R	0x0
17:14	CLIP_C	4 bits, unsigned Value when not used (according to the standard): 15	R	0x0
13:9	CLIP_B	5 bits, unsigned Value when not used (according to the standard): 31	R	0x00
8:0	CLIP_A	9 bits, signed Value when not used (according to the standard): 255	R	0x000

Table 14-718. Register Call Summary for Register iLF_CLIPLIMITSENTRYn

IVA2.2 Subsystem Register Manual

- iLF Register Mapping Summary: [0]

Table 14-719. iLF_COMMANDREG

Address Offset	0x0000 0FFC		
Physical Address	0x000A 1FFC	**Instance**	iLF
Description	iLF command register:: a write to this register decodes a command, a read returns an error. DATA/COMMAND 0x1 -> StartSeq() 0x2 -> StopSeq() 0x3 -> DbgEnable() 0x4 -> DbgDisable() 0x5 -> DbgStep() 0x6 -> Halt()		
Type	W		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																														CMD	

Bits	Field Name	Description	Type	Reset
31:3	RESERVED	Read returns an error	W	0x--------
2:0	CMD	DATA/COMMAND 0x1 -> StartSeq() 0x2 -> StopSeq() 0x3 -> DbgEnable() 0x4 -> DbgDisable() 0x5 -> DbgStep() 0x6 -> Halt()	W	0x-

Table 14-720. Register Call Summary for Register iLF_COMMANDREG

IVA2.2 Subsystem Functional Description • iLF/iME Control: [0] [1]
IVA2.2 Subsystem Basic Programming Model • Typical Use: [2]
IVA2.2 Subsystem Register Manual • iLF Register Mapping Summary: [3]

14.5.13 IA_GEM Registers

This section provides information about the Initiator Agent for the DSP megamodule Module. Each register in the module is described separately below.

14.5.13.1 IA_GEM Register Mapping Summary

Table 14-721. IA_GEM Register Mapping Summary

Register Name	Type	Register Width (Bits)	Address Offset	Physical Address
GEM_AGENT_STATUS	R	32	0x0000 0028	0x000F 8828

14.5.13.2 IA_GEM Register Descriptions

Table 14-722. GEM_AGENT_STATUS

Address Offset	0x0000 0028		
Physical Address	0x000F 8828	**Instance**	IA_GEM
Description	Agent Status Register		
Type	R		

31 30	29	28	27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10 9 8	7	6	5	4	3 2 1 0
RESERVED	INBAND_ERROR_SECONDARY	INBAND_ERROR_PRIMARY	RESERVED	MERROR	RESERVED	BURST_TIMEOUT	TIMEBASE	RESERVED	RESP_TIMEOUT	READEX	BURST	RESP_WAITING	REQ_ACTIVE	RESERVED	CORE_RESET

Bits	Field Name	Description	Type	Reset
31:30	RESERVED	Reserved	R	0x0
29	INBAND_ERROR_ SECONDARY	Error Status for in-band errors with MErrSteer indicating a secondary error. Read 0x0: No in-band error received Write 0x0: Ignored Read 0x1: In-band error received Write 0x1: Clear in-band error	RW	0x0

Bits	Field Name	Description	Type	Reset
28	INBAND_ERROR_PRIMARY	Error Status for in-band errors with MErrSteer indicating Primary Error Read 0x0: No in-band error received Write 0x0: Ignored Read 0x1: In-band error received Write 0x1: Clear in-band error	RW	0x0
27:25	RESERVED	Reserved	R	0x0
24	MERROR	MError assertion detected	R	0x0
23:17	RESERVED	Reserved	R	0x00
16	BURST_TIMEOUT	Status of open burst and	R	0x0
15:12	TIMEBASE	Observation of timebase signals for internal verification	R	0x0
11:9	RESERVED	Reserved	R	0x0
8	RESP_TIMEOUT	Response timeout status	R	0x0
7	READEX	Status of ReadEx/Write	R	0x0
6	BURST	Status of open burst	R	0x0
5	RESP_WAITING	Responses waiting	R	0x0
4	REQ_ACTIVE	Requests outstanding	R	0x0
3:1	RESERVED	Reserved	R	0x0
0	CORE_RESET	Reset input from core interface	R	0x0

Table 14-723. Register Call Summary for Register GEM_AGENT_STATUS

IVA2.2 Subsystem Register Manual
- IA_GEM Register Mapping Summary: [0]

14.5.14 IA_EDMA Registers

This section provides information about the Initiator Agent for the EDMA module. Each register in the module is described separately below.

14.5.14.1 IA_EDMA Register Mapping Summary

Table 14-724. IA_EDMA Register Mapping Summary

Register Name	Type	Register Width (Bits)	Address Offset	Physical Address
EDMA_AGENT_STATUS	R	32	0x0000 0028	0x000F 8C28

14.5.14.2 IA_EDMA Register Descriptions

Table 14-725. EDMA_AGENT_STATUS

Address Offset	0x0000 0028		
Physical Address	0x000F 8C28	**Instance**	IA_EDMA
Description	Agent Status Register		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED		INBAND_ERROR_SECONDARY	INBAND_ERROR_PRIMARY	RESERVED			MERROR	RESERVED							BURST_TIMEOUT	TIMEBASE				RESERVED			RESP_TIMEOUT	READEX	BURST	RESP_WAITING	REQ_ACTIVE	RESERVED			CORE_RESET

Bits	Field Name	Description	Type	Reset
31:30	RESERVED	Reserved	R	0x0
29	INBAND_ERROR_ SECONDARY	Error Status for in-band errors with MErrSteer indicating a secondary error. Read 0x0: No in-band error received Write 0x0: Ignored Read 0x1: In-band error received Write 0x1: Clear in-band error	RW	0x0
28	INBAND_ERROR_PRIMARY	Error Status for in-band errors with MErrSteer indicating Primary Error Read 0x0: No in-band error received Write 0x0: Ignored Read 0x1: In-band error received Write 0x1: Clear in-band error	RW	0x0
27:25	RESERVED	Reserved	R	0x0
24	MERROR	MError assertion detected	R	0x0
23:17	RESERVED	Reserved	R	0x00
16	BURST_TIMEOUT	Status of open burst and	R	0x0
15:12	TIMEBASE	Observation of timebase signals for internal verification	R	0x0
11:9	RESERVED	Reserved	R	0x0
8	RESP_TIMEOUT	Response timeout status	R	0x0
7	READEX	Status of ReadEx/Write	R	0x0
6	BURST	Status of open burst	R	0x0
5	RESP_WAITING	Responses waiting	R	0x0
4	REQ_ACTIVE	Requests outstanding	R	0x0
3:1	RESERVED	Reserved	R	0x0
0	CORE_RESET	Reset input from core interface	R	0x0

Table 14-726. Register Call Summary for Register EDMA_AGENT_STATUS

IVA2.2 Subsystem Register Manual

- IA_EDMA Register Mapping Summary: [0]

14.5.15 IA_SEQ Registers

This section provides information about the Initiator Agent for the Sequencer module. Each register in the module is described separately below.

14.5.15.1 IA_SEQ Register Mapping Summary

Table 14-727. IA_SEQ Register Mapping Summary

Register Name	Type	Register Width (Bits)	Address Offset	Physical Address
SEQ_AGENT_STATUS	R	32	0x0000 0028	0x000F 9028

14.5.15.2 IA_SEQ Register Descriptions

Table 14-728. SEQ_AGENT_STATUS

Address Offset	0x0000 0028		
Physical Address	0x000F 9028	**Instance**	IA_SEQ
Description	Agent Status Register		
Type	R		

31 30	29	28	27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12	11 10 9	8	7	6	5	4	3 2 1	0
RESERVED	INBAND_ERROR_SECONDARY	INBAND_ERROR_PRIMARY	RESERVED	RESERVED	TIMEBASE	RESERVED	RESP_TIMEOUT	READEX	BURST	RESP_WAITING	REQ_ACTIVE	RESERVED	CORE_RESET

Bits	Field Name	Description	Type	Reset
31:30	RESERVED	Reserved	R	0x0
29	INBAND_ERROR_ SECONDARY	Error Status for in-band errors with MErrSteer indicating a secondary error. Read 0x0: No in-band error received Write 0x0: Ignored Read 0x1: In-band error received Write 0x1: Clear in-band error	RW	0x0
28	INBAND_ERROR_PRIMARY	Error Status for in-band errors with MErrSteer indicating Primary Error Read 0x0: No in-band error received Write 0x0: Ignored Read 0x1: In-band error received Write 0x1: Clear in-band error	RW	0x0
27:25	RESERVED	Reserved	R	0x0
24	MERROR	MError assertion detected	R	0x0
23:17	RESERVED	Reserved	R	0x00
16	BURST_TIMEOUT	Status of open burst and	R	0x0
15:12	TIMEBASE	Observation of timebase signals for internal verification	R	0x0
11:9	RESERVED	Reserved	R	0x0
8	RESP_TIMEOUT	Response timeout status	R	0x0
7	READEX	Status of ReadEx/Write	R	0x0
6	BURST	Status of open burst	R	0x0
5	RESP_WAITING	Responses waiting	R	0x0
4	REQ_ACTIVE	Requests outstanding	R	0x0

Bits	Field Name	Description	Type	Reset
3:1	RESERVED	Reserved	R	0x0
0	CORE_RESET	Reset input from core interface	R	0x0

Table 14-729. Register Call Summary for Register SEQ_AGENT_STATUS

IVA2.2 Subsystem Register Manual

- IA_SEQ Register Mapping Summary: [0]

Display Subsystem

This chapter describes the display subsystem for the OMAP34xx multimedia device.

Topic **Page**

15.1 Display Subsystem Overview 2124
15.2 Display Subsystem Environment 2129
15.3 Display Subsystem Integration 2179
15.4 Display Subsystem Functional Description 2196
15.5 Display Subsystem Basic Programming Model 2268
15.6 Display Subsystem Use Cases and Tips 2346
15.7 Display Subsystem Register Manual 2409

15.1 Display Subsystem Overview

The display subsystem provides the logic to display a video frame from the memory frame buffer (either SDRAM or SRAM) on a liquid-crystal display (LCD) panel or a TV set. The display subsystem integrates the following elements:

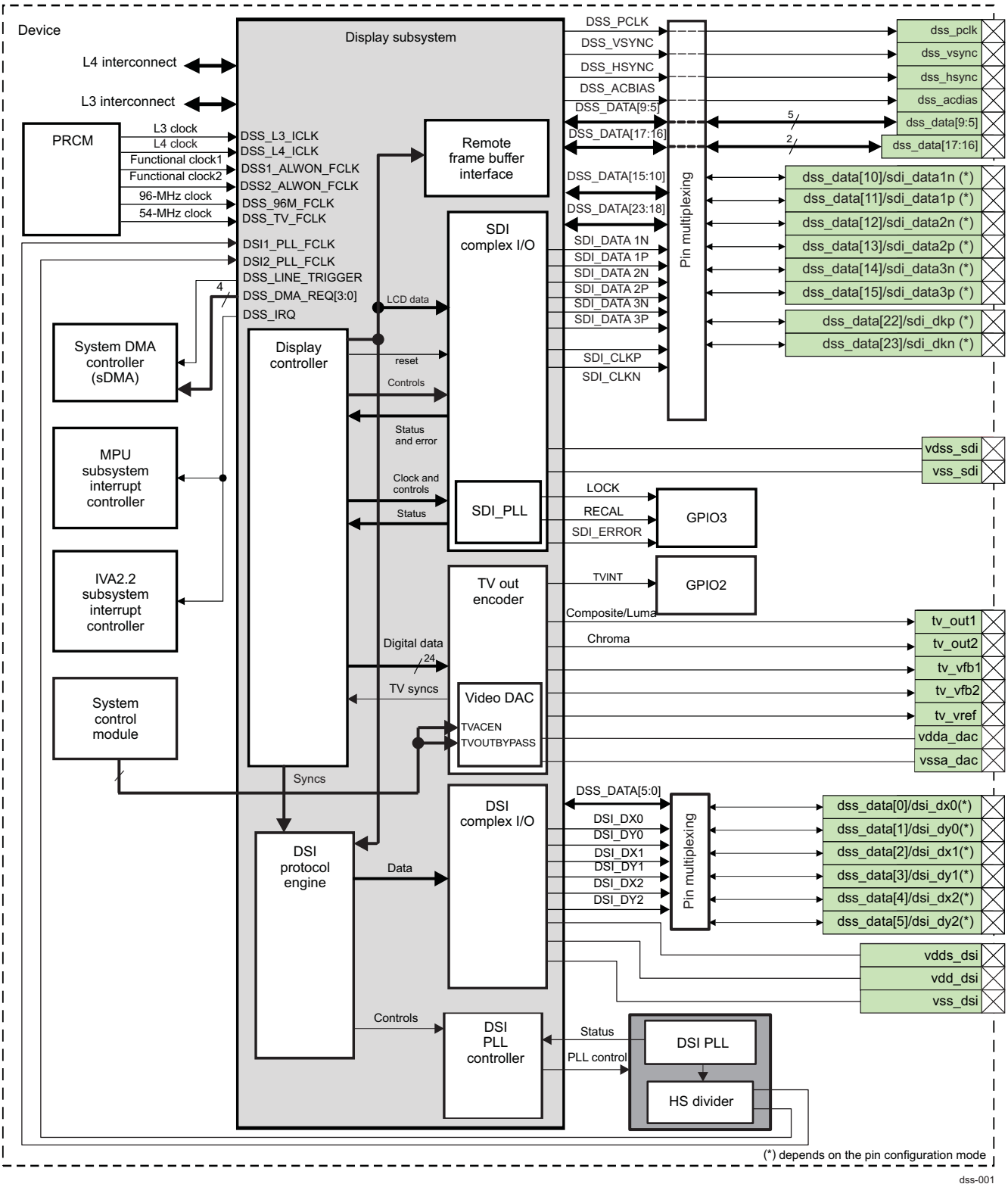
- Display controller (DISPC) module
- Remote frame buffer interface (RFBI) module
- Serial display interface (SDI) complex input/output (I/O) module with the associated phased-locked loop (PLL)
- Display serial interface (DSI) complex I/O module and a DSI protocol engine
- DSI PLL controller that drives a DSI PLL and high-speed (HS) divider.
- NTSC/PAL video encoder

The display controller and the DSI protocol engine are connected to the L3 and L4 interconnect; the RFBI and the TV out encoder modules are connected to the L4 interconnect.

Note: The SDI module, the DSI complex I/O module, and the DSI PLL controller are not connected to an L3 or L4 interconnect. Specific display subsystem registers manage their programmable features.

Figure 15-1 shows a block diagram of the display subsystem.

Figure 15-1. Display Subsystem Highlight



Note: For more information about connecting the LOCK, RECAL, SDI_ERROR, and TVINT signals through the GPIO2 and GPIO3 modules, see Chapter 25, *GPIO*.

The display subsystem includes the following main features:

- Display controller
  - Display modes
    - Programmable pixel display modes (1, 2, 4, 8, 12, 16, and 24 bits-per-pixel [BPP] modes)
    - Programmable display size supported:
      - XGA - 1024 × 768 VESA timings at 60 fps (pixel clock = 63.5 MHz)
      - WXGA - 1280 × 800 VESA timings at 59.91 fps (pixel clock = 71 MHz)
      - SXGA+ - 1400 × 1050 direct drive of LCD with minimal blanking at 50 fps (pixel clock = 75 MHz)
      - HD 720p - 1280 × 720 CEA 861-D timings at 60 fps (pixel clock = 74.25 MHz)
    - 256 × 24-bit entries palette in red, green, and blue (RGB)
    - Programmable pixel rate up to 75 MHz

    Note: The panel size is programmable and can be any width that is a multiple of 8 pixels (line length) in the range [1:2048] pixels (in the case of the RFBI mode, the minimum transfer size is a byte). The maximum resolution is 2048 (lines) x 2048 (pixels).
  - Display support
    - Four types of displays are supported: Passive (super-twist nematic [STN]) and active (thin-film transistor [TFT]) colors, passive (STN), and active (TFT) monochromes.
    - 4-/8-bit monochrome passive matrix panel interface support (15 grayscale levels supported using dithering block)
    - 8-bit color passive matrix panel interface support (3375 colors supported for a color panel using dithering block)
    - 12-/16-/18-/24-bit active matrix panel interface support (replicated or dithered encoded pixel values)
    - Remote frame buffer support through the RFBI module
    - Partial display through the RFBI module
    - Second 24-bit digital output
    - Multiple-cycle output format on 8-/9-/12-/16-bit interface time division multiplexing (TDM)
    - HDMI through external bridge
  - Signal processing
    - Overlay support for graphics (ARGB, RGBA, RGB, or Color Look-Up Table (CLUT)) and video1 (YCbCr 4:2:2, or ARGB, RGBA, RGB), video2 (YCbCr 4:2:2, or ARGB, RGBA, RGB)
    - Programmable video resizer independent horizontal and vertical resampling: Upsampling (up to x8) and downsampling (down to 1/4), maximum input width of 1024 pixels in 5-tap mode, and 2048 pixels in 3-tap configurations; no limitation on input height
    - Rotation 90-, 180-, and 270-degrees
    - Transparency color key (source and destination)
    - Synchronized buffer update
    - Programmable video color space conversion YCbCr 4:2:2 into RGB
    - Hardware cursor
    - Gamma curve support on LCD output
    - Multiple-buffer support
    - Mirroring support
    - Programmable color phase rotation (CPR)
    - Alpha blending support (no rescaling in ARGB or RGBA formats)
  - Advanced
    - Self-refresh using the DMA FIFO
    - Arbitration between high/low priority (graphics video1 and video2)
    - FIFO handcheck in STALL mode
  - Power modes: Low-power saving modes

- RFBI (MIPI® DBI protocol)
  - Access to remote frame buffer (RFB) direct microprocessor unit (MPU) interface
    - Sends commands to the RFB panel through the L4 interconnect
    - Sends data, received from the display controller or from the MPU through the DISPC pixel data bus, to the RFB panel
    - Reads data/status from the RFB to the L4 interconnect
  - RFB interface
    - 8-/9-/12-/16-bit 8086-series parallel interface
    - Two programmable configurations for two devices connected to the RFBI module
  - Data formats
    - Programmable pixel modes (12-/16-/18-/24-BPP modes in RGB format)
    - Programmable output formats on one/multiple cycles per pixel (data from the display controller and from the L4 interconnect)
  - Interconnect/FIFO
    - One slave port with DMA request and interconnect FIFO of 24x32-bit depth (for write access to DSS.RFBI_DATA register only)
    - One video port FIFO of 8 × 24-bit depth receiving data from the display controller
- SDI
  - TI Flatlink™ 3G display interface support
  - Pin multiplexing allows simultaneous operation with a single 9-bit RFBI-driven display (no support for simultaneous dual RFBI panel and SDI)

Note: The SDI pins are multiplexed with LCD parallel outputs.

- MIPI DSI
  - Transfer pixels and data received on the video port or L4 interconnect to the display through the DSI DSI_PHY
  - The maximum resolution supported on the video port is XGA at 60 fps with 24-bit pixels (maximum pixel clock of 67 MHz) for low voltage.
  - Supports video mode and command mode
  - Bidirectional data link support (only one data lane is used in reverse direction in command mode)
  - Supports up to two data-configurable lanes, in addition to the clock signaling (minimum of one data link and maximum of two, depending on speed, signal integrity requirements, and number of displays)
  - Maximum data rate of 800 Mbps per data pair
  - Data splitter for 2-data lane configuration
  - Error-correction code (ECC) and check-sum generation
  - Burst support for the video mode
  - RGB16, RGB18 packed and nonpacked, and RGB24 formats supported for video mode
  - Serial configuration port (SCP) for the DSI_PHY complex I/O and DSI PLL
  - Connection to the DSI_PHY complex I/O through PPI
  - Data interleaving support for one synchronous stream (video mode) from the display controller and up to three interleaved asynchronous streams (command mode) from the interconnect concurrently
  - Data interleaving supports up to four interleaved asynchronous streams (command mode) from the interconnect or video port when there is no video mode
  - MIPI DCS support (transparent to the protocol engine, no decoding and interpretation of the information from and to the peripheral)
  - Supports selection between low-power state and HS mode between HS packet transfers
  - Generic data type (DT) support

Note: The DSI pins are multiplexed with LCD parallel outputs.

- Video encoder

- NTSC/PAL encoder outputs with the following standards:
  - NTSC-J, M
  - PAL-B, D, G, H, I
  - PAL-M
- CGMS-A as described in the CEA-608-x Standard.
- Input data interface compatible with the following protocols:
  - 24-bit input bus compatible with external sync
  - RGB 4:4:4
- Dual output data 10-bit interface for internal digital-to-analog converter (DAC) that supports:
  - Composite video (CVBS)
  - Separate video (S-video)
- TV output data supports ITU-R BT 470-7 recommendation standard for consumer market
- Master clock input 13.5 MHz, 27 MHz (supports ITU-R 601 sampling for NTSC/PAL), and 54 MHz
- Programmable horizontal sync, vertical timing, and waveforms
- Programmable subcarrier frequency and SCH
- Internal test pattern generation (color bar, flat field, color burst)
- 2x/4x oversampling
- Supports square pixel sampling (NTSC: 12.27 MHz, 24.54 MHz, 49.09 MHz PAL: 14.75 MHz, 29.5 MHz, 59 MHz)

CAUTION

In square pixel mode, an external clock generator is required to provide sampling frequencies.

- TV detection gating pulse generation

15.2 Display Subsystem Environment

This section describes the two main functions handled by the display subsystem:

- LCD support
- TV display support

15.2.1 LCD Support

LCD panels can be connected to the display subsystem of the device using parallel and/or serial interfaces.

15.2.1.1 Parallel Interface

In parallel interface, the paths of the display subsystem modules are the display controller and the RFBI.

The display controller provides the required control signals to interface the memory frame buffer (SDRAM or SRAM) directly to the external displays. The display controller is connected to the memory through the L3 interconnect and has its own DMA (with embedded FIFOs) to read data from the system memory. The L3 interconnect is the master port, while the L4 interconnect is the slave port of the display subsystem.

The display controller has two I/O pad modes at the module level:

- RFBI mode (RFBI enabled), which implements the MIPI DBI protocol
- Bypass mode (RFBI disabled), which implements the MIPI DPI protocol

The DSS.DISPC_CONTROL[16:15] GPOUT[1:0] bits control selection of the display subsystem modules (see Table 15-1).

Table 15-1. I/O Pad Mode Selection

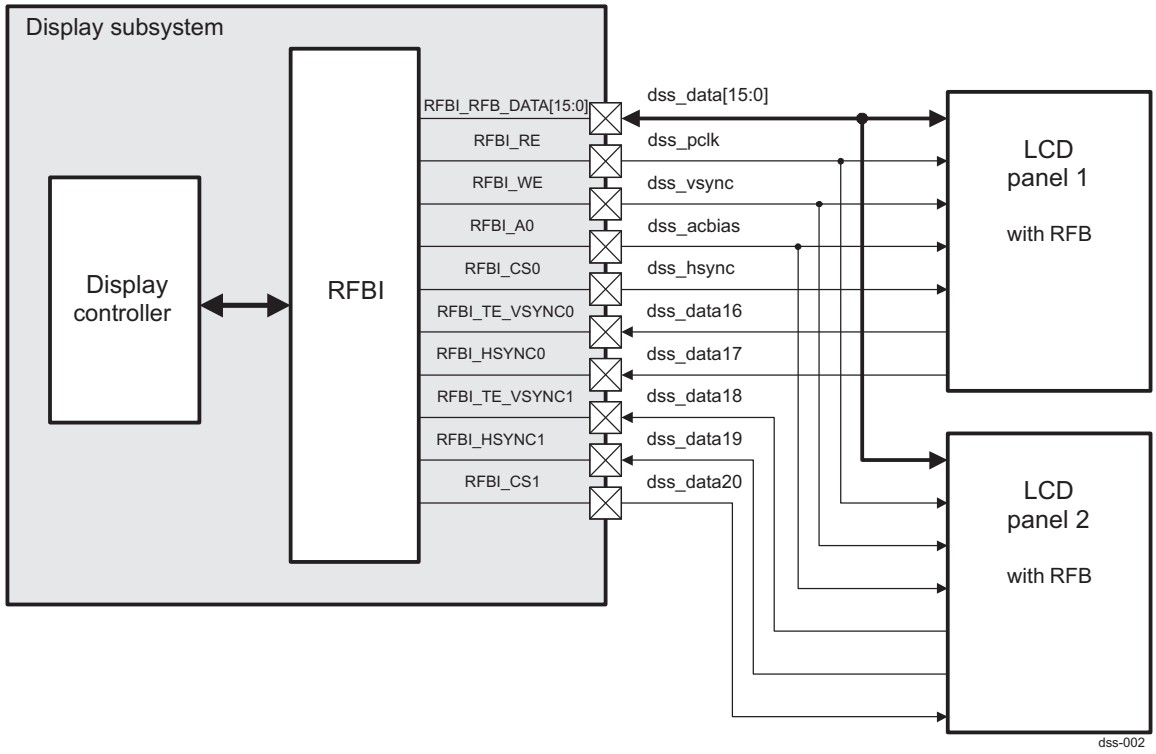
	DSS.DISPC_CONTROL[15] GPOUT0 Bit	
DSS.DISPC_CONTROL[16] GPOUT1 Bit	**0**	**1**
0	00 (reset)	01 (RFBI mode)
1	10 (invalid)	11 (bypass mode)

The RFB of the LCD panel is directly connected to the RFBI module of the device. The RFBI controls the reads/writes from/to the RFB. The RFBI receives the output from the DISPC (which takes data from the memory) and generates the signals to control the LCD panel. Through the RFBI, the MPU can send commands or parameter/display data to the LCD panel and directly set the DISPC registers to read/write the data from/to the memory in the LCD panel. The RFBI can manage up to two LCD panels when the serial interface is not used.

15.2.1.1.1 Parallel Interface in RFBI Mode (MIPI DBI Protocol)

Figure 15-2 shows the LCD support parallel interface in RFBI mode (example for 16-bit data interface).

Figure 15-2. LCD Support Parallel Interface (RFBI Mode)



Note: If the SDI is used, the second display device connected to RFB#1 is limited to 9-bit data.

Configure the DSS.RFBI_CONTROL[3:2] CONFIGSELECT bit field to drive signals for LCD 1 only, LCD 2 only, or both LCD 1 and LCD 2.

Table 15-2 describes the interface signals to/from the LCD panel in RFBI mode.

Table 15-2. LCD Interface Signals (RFBI Mode)

Signal Name	Type[(1)]	Description
RFBI_RFB_DATA[15:0]	I/O	RFBI I/O data
RFBI_RE	O	Read access signal
RFBI_WE	O	Write access signal
RFBI_A0	O	Command/data selection signal
RFBI_CS0	O	Chip-select (CS) signal for LCD 1
RFBI_CS1	O	CS signal for LCD 2
RFBI_TE_VSYNC0	I	Tearing effect (TE) synchronization signal (TE or VSYNC for LCD panel 1)
RFBI_HSYNC0	I	HSYNC from LCD panel 1
RFBI_TE_VSYNC1	I	TE synchronization signal (TE or VSYNC for LCD panel 2)
RFBI_HSYNC1	I	HSYNC from LCD panel 2

(1) I = Input, O = Output

- RFBI_RFB_DATA[15:0]: The pixel data comprises the RFBI pixel data (bits 15:0). A write/read command must be sent to the LCD panel to send/read the data.

  Before any data access, the application must send commands and parameters when it is necessary to configure an LCD panel. The data is used as input in read operations during production test and also to read the status of the registers in the LCD panel and pixels from the embedded frame buffer in the LCD panel module. RFBI_RFB_DATA is multiplexed at the chip-level boundary with dss_data [15:0].
- RFBI_RE: This is the read-enable signal used to indicate when a read from the embedded memory in the LCD panel is ongoing. The RFBI registers describe the behavior of the read signal (off/on/cycle time). The polarity of the read-enable signal is programmable. This signal is multiplexed at the chip-level boundary with dss_pclk. The read is used to get status/data information from the LCD panel.
- RFBI_WE: The write-enable signal is used to indicate when a write is ongoing. The RFBI registers describe the behavior of the write signal (off/on/cycle time). The polarity of the write-enable signal is programmable. This signal is multiplexed at the chip-level boundary with dss_vsync.
- RFBI_A0: The signal is asserted to indicate its status: Command or data. The polarity is programmable and the status of the signal depends on the RFBI registers written by the application (CMD/READ/STATUS/PARAM/PIXEL). The register in use by the hardware defines the status of RFBI_A0. The order of the writes/reads to the RFBI registers CMD/READ/STATUS/PARAM/PIXEL defines the transitions of A0. This signal is multiplexed at the chip-level boundary with dss_acbias.
- RFBI_CS*x*: The signal is the chip-select (CS*x*) asserted to indicate which LCD panel is selected and must be ready to receive/transmit commands and data. When RE or WE is on, CS*x* must not be changed (*x* = 0 for LCD panel 1; *x* = 1 for LCD panel 2). CS0 is multiplexed at the chip-level boundary with dss_hsync, and CS1 is multiplexed at the chip-level boundary with dss_data[20].
- RFBI_TE_VSYNC*x*: Based on the trigger mode selected, the signal is the TE pulse signal or the LCD panel VSYNC (vertical synchronization) pulse signal. RFBI_TE_VSYNC*x* is used by the TE logic as the synchronization signal to send the pixel to the LCD panel.

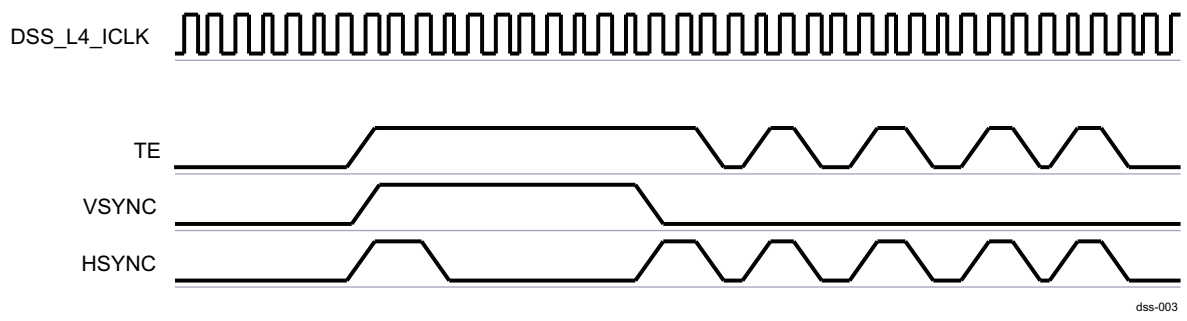
  To select the trigger mode, configure the DSS.RFBI_CONFIGi[3:2] TRIGGERMODE bit field (0x0: Internal trigger mode with the DSS.RFBI_CONTROL[4] ITE bit, 0x1: External trigger mode with the TE signal RFBI_TE_VSYNC*x*, 0x2: External trigger mode with the RFBI_TE_VSYNC*x*, and RFBI_HSYNC*x* signals with the programmable line counter).

  These signals are multiplexed at the chip-level boundary with dss_data[16] (RFBI_TE_VSYNC0) and dss_data[18] (RFBI_TE_VSYNC1) (LCD panel 1: *x* = 0; LCD panel 2: *x* = 1).
- RFBI_HSYNC*x*: The HSYNC pulse signals indicate to the RFBI module when horizontal synchronization occurs. The polarity of the HSYNC signals is programmable. The minimum pulse width of the signal is two L4 cycles. RFBI_HSYNC is used by the TE logic as a synchronization signal to send the pixel to the LCD panel. These signals are multiplexed at the chip-level boundary with dss_data[17] (RFBI_HSYNC0) and dss_data[19] (RFBI_HSYNC1) (LCD panel 1: *x* = 0; LCD panel 2: *x* = 1).

15.2.1.1.1.1 Description of the TE Pulse Signal

The externally-generated TE synchronization signal is a logical OR or AND operation between the HSYNC and VSYNC signals (see Figure 15-3). The logical operation (OR or AND) depends on the HSYNC and VSYNC signals polarity. The VSYNC signal indicates to the RFBI module when vertical synchronization occurs; the HSYNC signal indicates to the RFBI module when horizontal synchronization occurs.

Figure 15-3. External Generation of TE Signal Based on Logical OR Operation Between HSYNC and VSYNC (Active-High)



The RFBI module detects the VSYNC and HSYNC pulses embedded in the received signal. VSYNC is detected based on the minimum pulse width defined by the DSS.RFBI_VSYNC_WIDTH register.

HSYNC is detected based on the minimum pulse width defined by the DSS.RFBI_HSYNC_WIDTH register.

The signal is generated from external logic based on the VSYNC/HSYNC of the LCD panel. The automatic trigger can be programmed based on the RFBI_TE signal or use a bit field in the RFBI registers to start data capture.

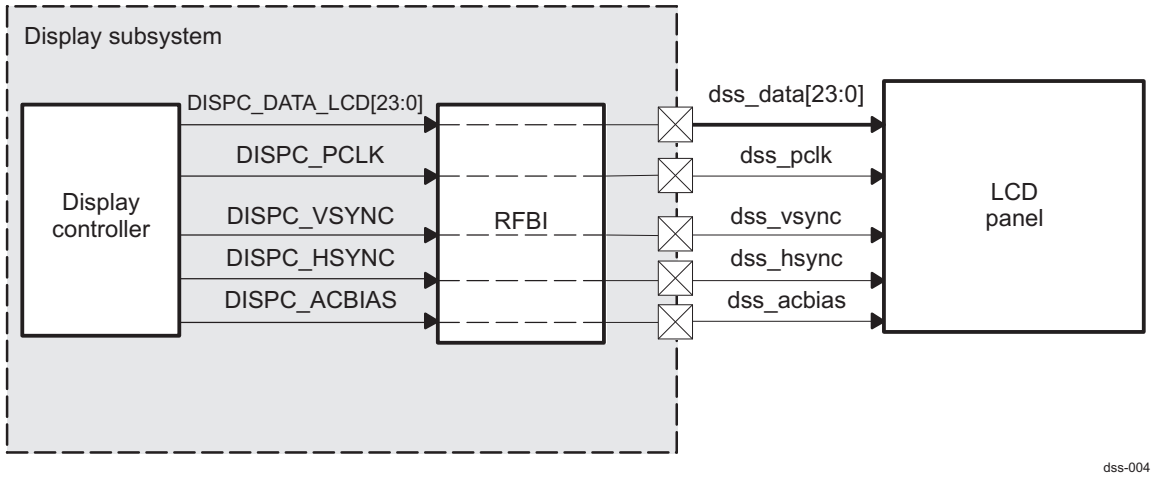
The polarity of the TE signal is programmable. The HSYNC and VSYNC pulses embedded in the TE signal have the same polarity, which is active high for an ORed signal and active low for an ANDed signal. The minimum pulse width of the signal is two L4 cycles. Hardware resets the line counter when the VSYNC occurs and increments it at every HSYNC. Transfer to the LCD panel begins when the line counter reaches the programmable line number.

15.2.1.1.2 Parallel Interface in Bypass Mode (MIPI DPI Protocol)

When bypass mode is enabled, the display controller must be set to use it.

Figure 15-4 shows the LCD support parallel interface in bypass mode.

Figure 15-4. LCD Support Parallel Interface (Bypass Mode)



Note: In bypass mode, the SDI and the parallel interface cannot be used at the same time.

Table 15-3 describes the interface signals to/from the LCD panel in bypass mode.

Table 15-3. LCD Interface Signals (Bypass Mode)

Signal Name	Type[(1)]	Description
DISPC_DATA_LCD[23:0]	O	LCD data from the display controller module
DISPC_PCLK	O	Pixel CLK from the display controller module
DISPC_VSYNC	O	VSYNC from the display controller module
DISPC_HSYNC	O	HSYNC from the display controller module
DISPC_ACBIAS	O	ACBIAS from the display controller module

(1) I = Input, O = Output, I/O = Input/Output

- DISPC_DATA_LCD[23:0]: The panel pixel data comes directly from the display controller module. DISPC_DATA_LCD is connected at the chip-level boundary with dss_data[23:0].
- DISPC_PCLK: This signal is the pixel clock that comes directly from the display controller. This signal is multiplexed at the chip-level boundary with dss_pclk.
- DISPC_VSYNC: Uses the vertical synchronization signal from the display controller. The LCD frame

clock (VSYNC) toggles after all the lines in a frame are transmitted to the LCD panel and a programmable number of line clock cycles has elapsed both at the beginning and at the end of each frame. This signal is multiplexed with dss_vsync at the chip-level boundary.

- DISPC_HSYNC: Uses the horizontal synchronization signal from the display controller. The LCD line clock (HSYNC) toggles after all pixels in a line are transmitted to the LCD panel and a programmable number of pixel clock wait-states elapse, both at the beginning and at the end of each line. This signal is multiplexed on the chip-level boundary with dss_hsync.
- DISPC_ACBIAS: Uses the ac-bias signal from the display controller.
  - In passive matrix technology, the ac-bias signal is configured to transition each time a programmable number of line clocks occurs. To prevent a dc charge within the screen pixels, the power and ground supplies of the panel are periodically switched. The DISPC signals the panel to switch the polarity by toggling the ac-bias pin.
  - In active matrix technology, the ac-bias signal acts as an output-enable signal to indicate when data must be latched using the pixel clock. This signal is multiplexed on the chip-level boundary with dss_acbias.

15.2.1.1.3 LCD Output and Data Format for the Parallel Interface

This section describes the pixel data bus and shows timing diagrams of transactions and synchronizations in both RFBI and bypass modes.

Figure 15-5 through Figure 15-11 show the pixel data bus for bypass mode, depending on the use of 4-, 8-, 12-, 16-, 18-, or 24-pixel data output pins. In RFBI mode, the pixel data bus is reformatted in accordance with the input and output data bus width.

Table 15-4 lists the number of displayed pixels per pixel clock cycle based on the type of display panel.

Table 15-4. Number of Displayed Pixels per Pixel Clock Cycle Based on Display Type

Display Panel	Number of Displayed Pixels per Pixel Clock Cycle
Monochrome 4-bit	4
Monochrome 8-bit	8
Passive matrix color	8/3
Active matrix	1

- Passive matrix technology, Monochrome mode

  Monochrome displays use either a 4-bit or 8-bit interface. Each bit represents one pixel (on or off), which means that either 4 or 8 pixels are sent to the LCD at each pixel clock.

  Figure 15-5 and Figure 15-6 show 4- and 8-bit monochrome displays, respectively.

Figure 15-5. LCD Pixel Data Monochrome4 Passive Matrix

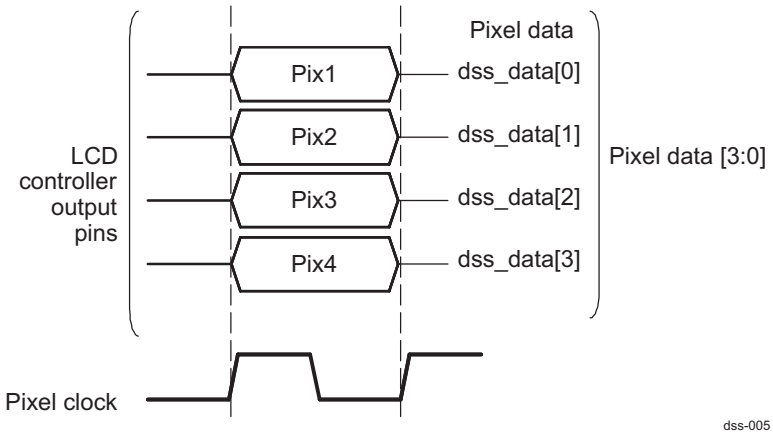
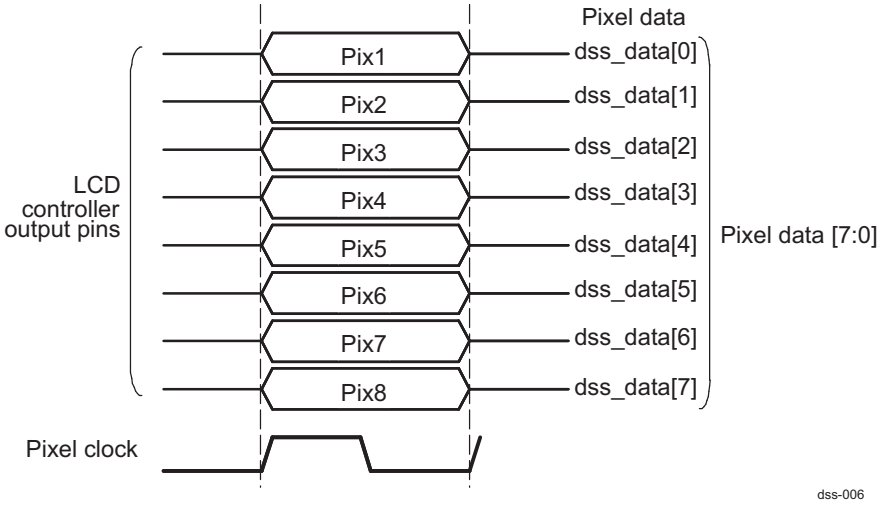


Figure 15-6. LCD Pixel Data Monochrome8 Passive Matrix

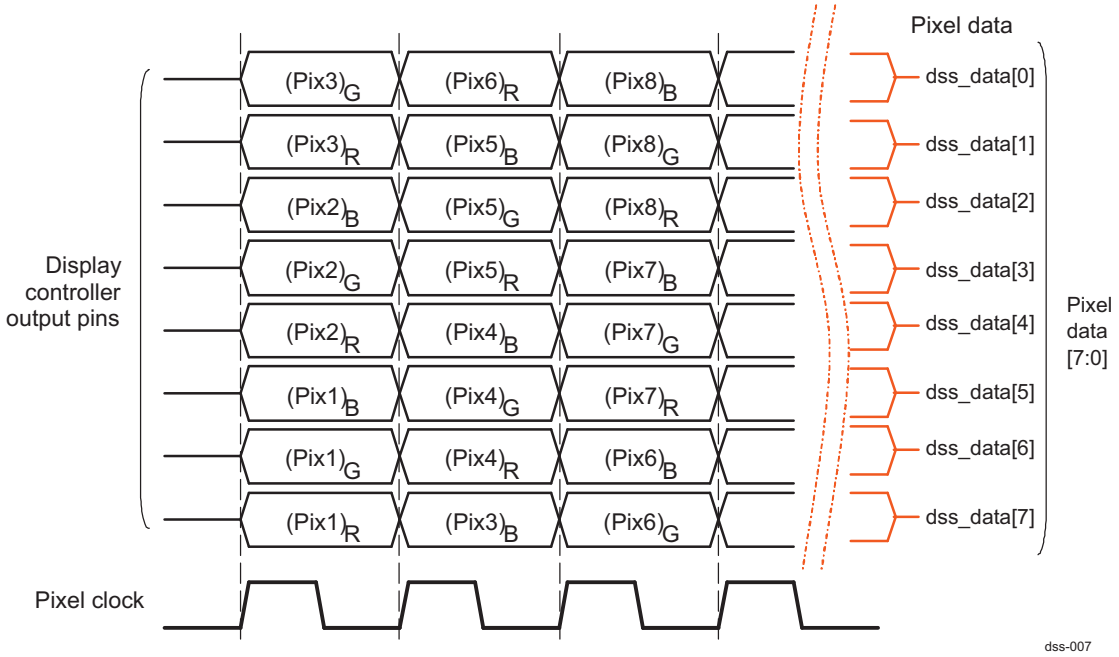


- Passive matrix technology, color mode

  Color passive displays use 8-bit data input lines. In a given pixel clock cycle, each line represents one color component (red, green, or blue).

  Figure 15-7 shows an 8-bit color passive matrix display.

Figure 15-7. LCD Pixel Data Color Passive Matrix



- Active matrix technology

  Active matrix displays bypass the STN dithering logic block and the output FIFO. Each line represents one pixel.

  Figure 15-8 through Figure 15-11 show 12-, 16-, 18-, and 24-active matrix displays, respectively.

Figure 15-8. LCD Pixel Data Color12 Active Matrix

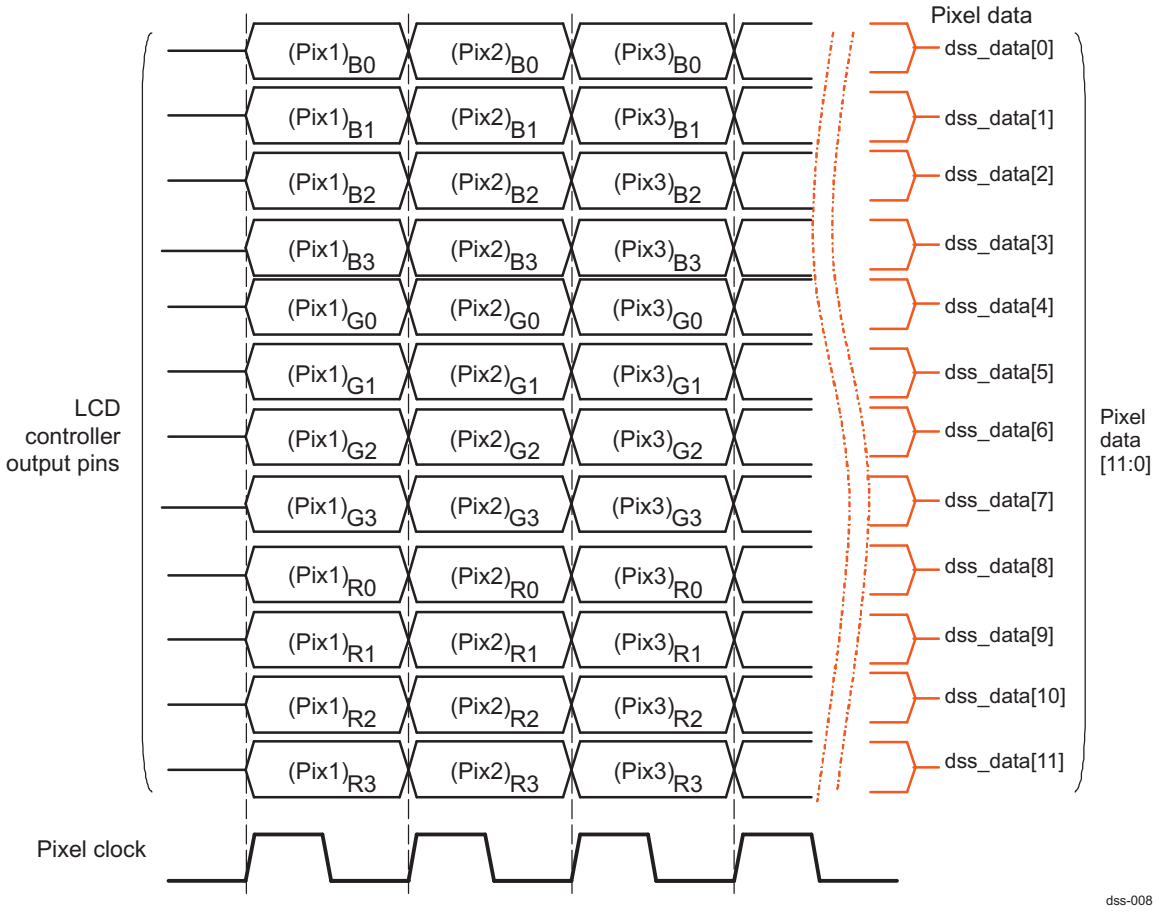


Figure 15-9. LCD Pixel Data Color16 Active Matrix

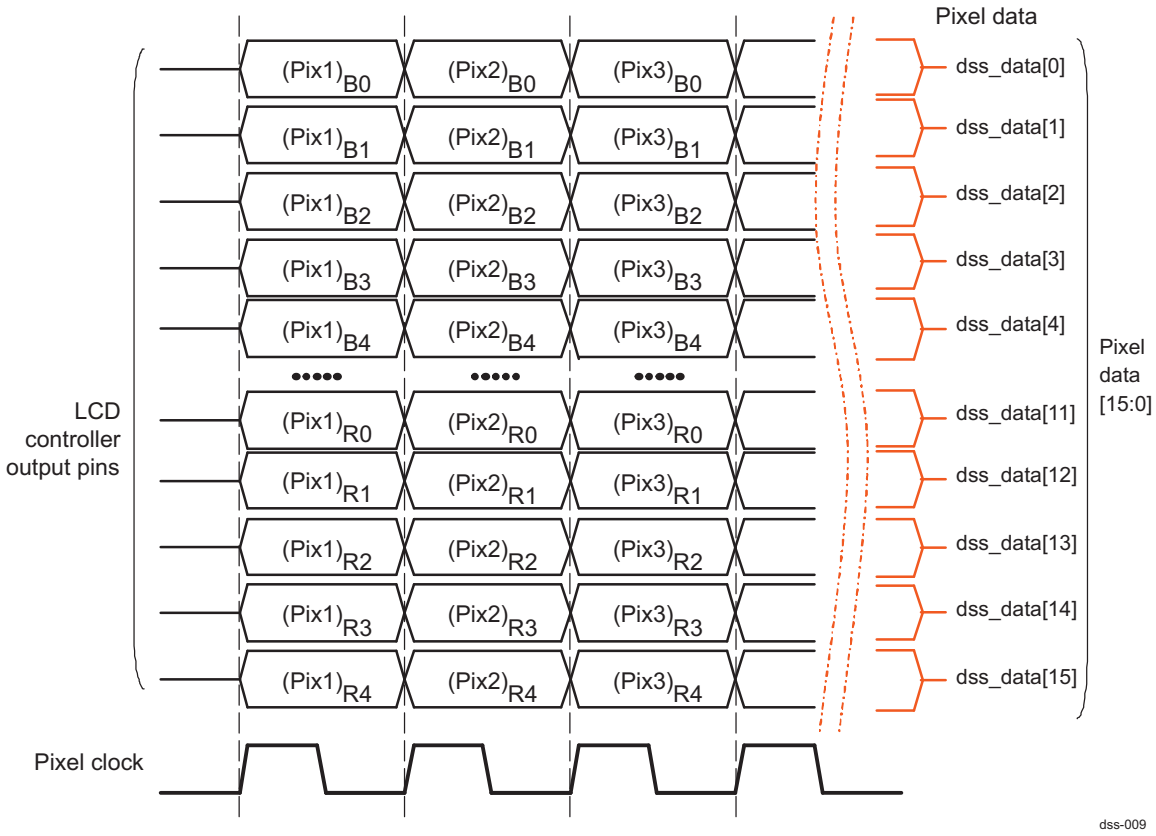


Figure 15-10. LCD Pixel Data Color18 Active Matrix

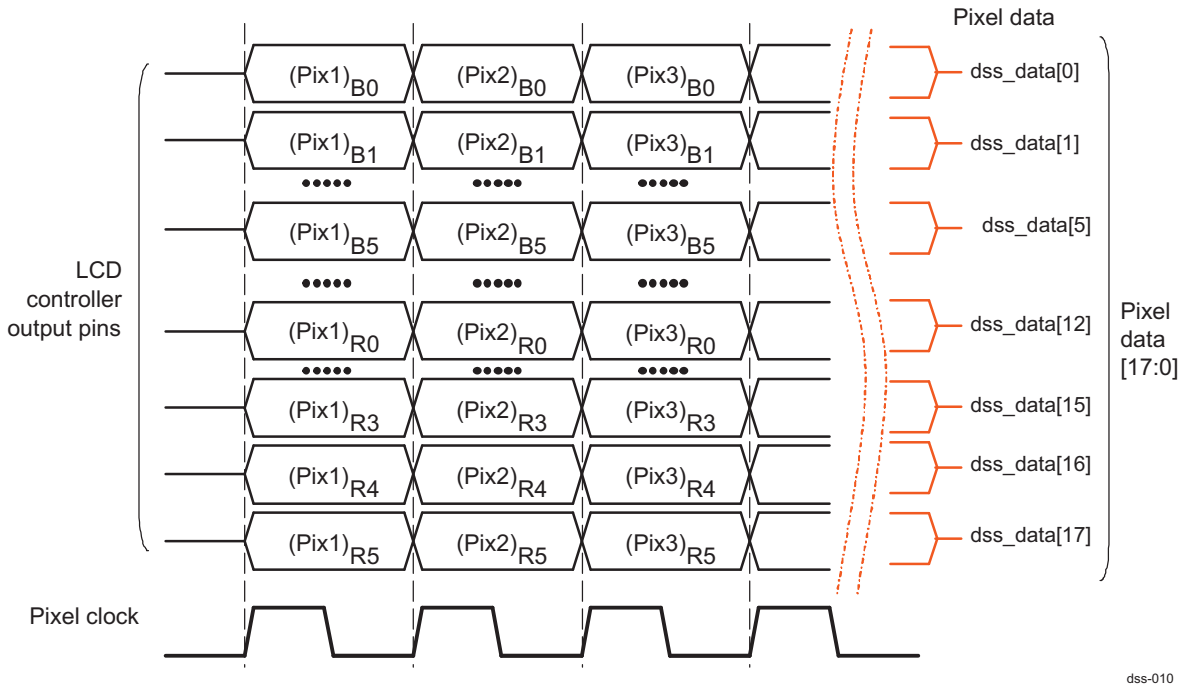
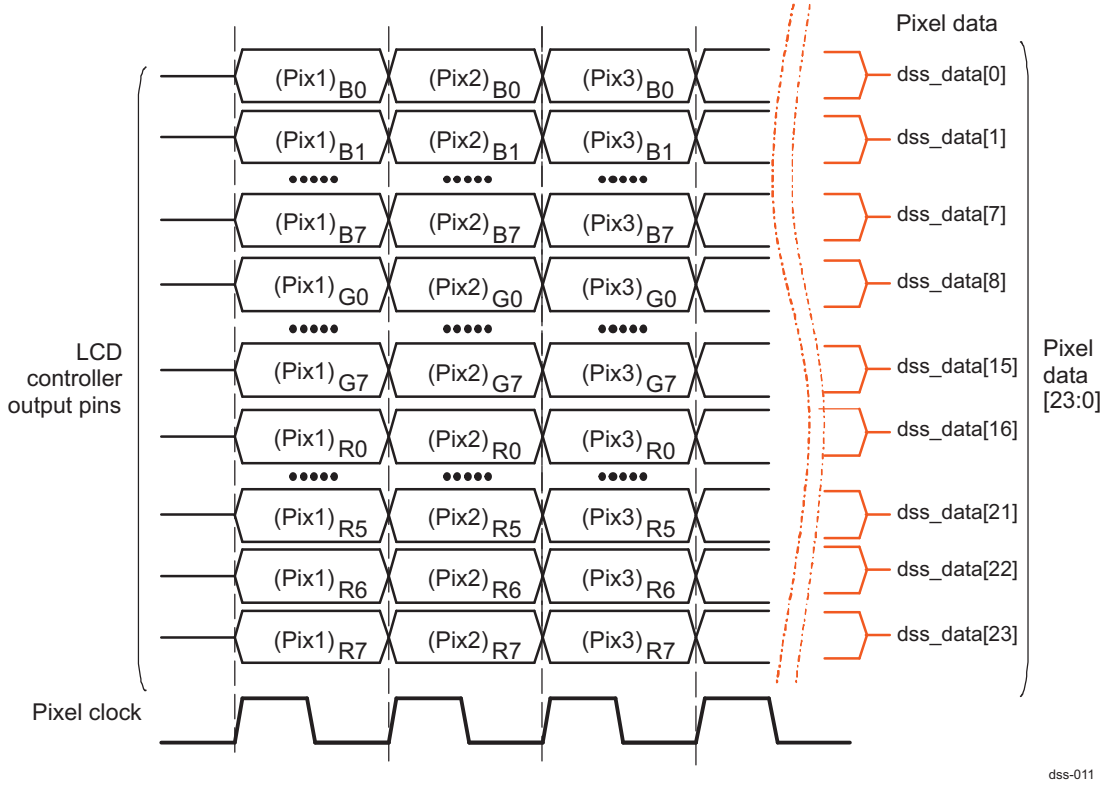


Figure 15-11. LCD Pixel Data Color24 Active Matrix

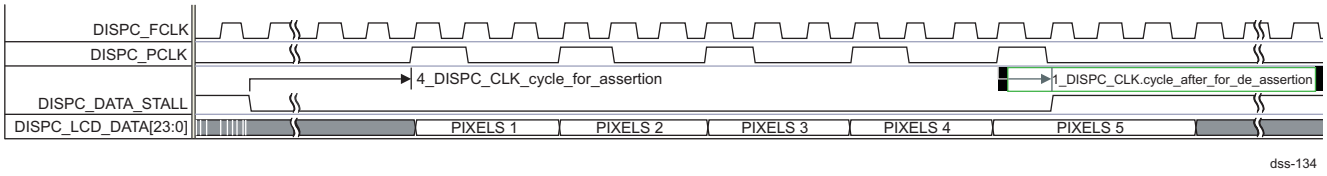


15.2.1.1.4 Transaction Timing Diagrams

- Timing diagrams in flow control mode
  - Stall signal

    The stall signal is used in RFBI and DSI modes. In the case of RFBI mode, it is used to indicate when the display controller must stop sending data over the LCD output interface. The RFBI module asserts the stall signal to stop data output by the display controller. It is deasserted to indicate when new data must be outputted by the display controller.

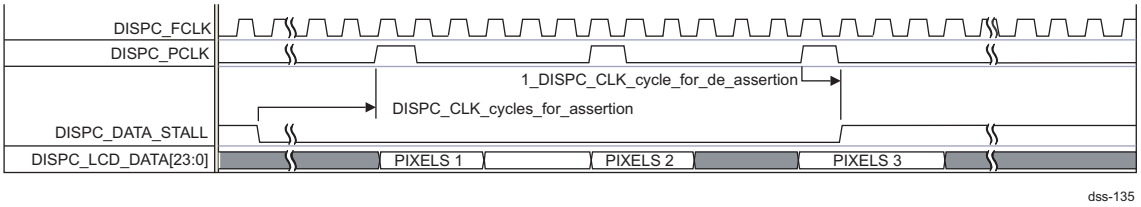
Figure 15-12. RFBI Data Stall Signal Diagram



To avoid underflow of the DMA FIFO, the FIFO handcheck feature can be enabled by setting the DSS.DISPC_CONFIG[16] FIFOHANDCHECK bit to 1. The fullness of the FIFOs associated with the pipelines used for the LCD output is checked when the STALL signal is inactive before providing data to the pipeline. This prevents emptying the FIFO when the RFBI module requests data and there is not enough data in the display controller DMA FIFO. This feature must be enabled only when the STALL mode is used (DSS.DISPC_CONTROL[11] STALLMODE bit set to 1).

When the FIFO handcheck feature is activated, the pixel transfer to the RFBI module during STALL inactivity period can be stopped (no DISPC_PCLK pulse) and restarted when there is enough data in the FIFO. The FIFO handcheck ensures that underflow can not occur for the pipelines associated with the LCD output in RFBI mode. Figure 15-13 details the RFBI data stall with FIFO handcheck mode activated.

Figure 15-13. RFBI Data Stall Signal Diagram With Handcheck



– RFBI timing diagrams

Table 15-5 lists the programmable timing fields. Figure 15-14 through Figure 15-16 show timing diagrams of read/write transactions to the LCD panel for the RFBI mode.

Table 15-5. Programmable Timing Fields in RFBI Mode

Timing Name	Register Field	Description
CSOnTime	DSS.RFBI_ONOFF_TIMEi[3:0] CSONTIME (with I = 0 or 1)	CS assertion time from start access time
CSOffTime	DSS.RFBI_ONOFF_TIMEi[9:4] CSOFFTIME (with I = 0 or 1)	CS deassertion time from start access time
WeCycleTime	DSS.RFBI_CYCLE_TIMEi[5:0] WECYCLETIME (with I = 0 or 1)	The time when A0 becomes valid until write cycle completion
WEOnTime	DSS_RFBI_ONOFF_TIMEi[13:10] WEONTIME (with I = 0 or 1)	WE assertion delay time from start access time
WEOffTime	DSS_RFBI_ONOFF_TIMEi[19:14] WEOFFTIME (with I = 0 or 1)	WE deassertion delay time from start access time
RECycleTime	DSS.RFBI_CYCLE_TIMEi[11:6] RECYCLETIME (with I = 0 or 1)	The time when A0 becomes valid until read cycle completion
REOnTime	DSS_RFBI_ONOFF_TIMEi[23:20] REONTIME (with I = 0 or 1)	RE assertion delay time from start access time
REOffTime	DSS.RFBI_ONOFF_TIMEi[29:24] REOFFTIME (with I = 0 or 1)	RE assertion delay time from start access time
CSPulseWidth	DSS.RFBI_CYCLE_TIMEi[17:12] CSPULSEWIDTH (with I = 0 or 1)	The time when write cycle time or read cycle time completes

Figure 15-14. Command Data Write

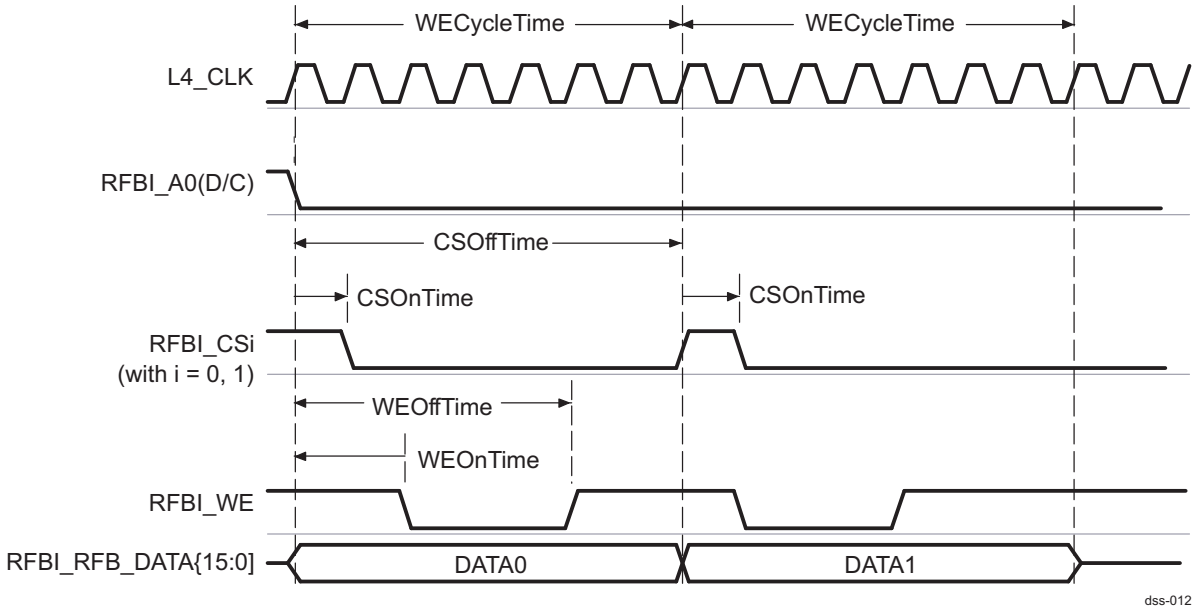


Figure 15-15. Display Data Read

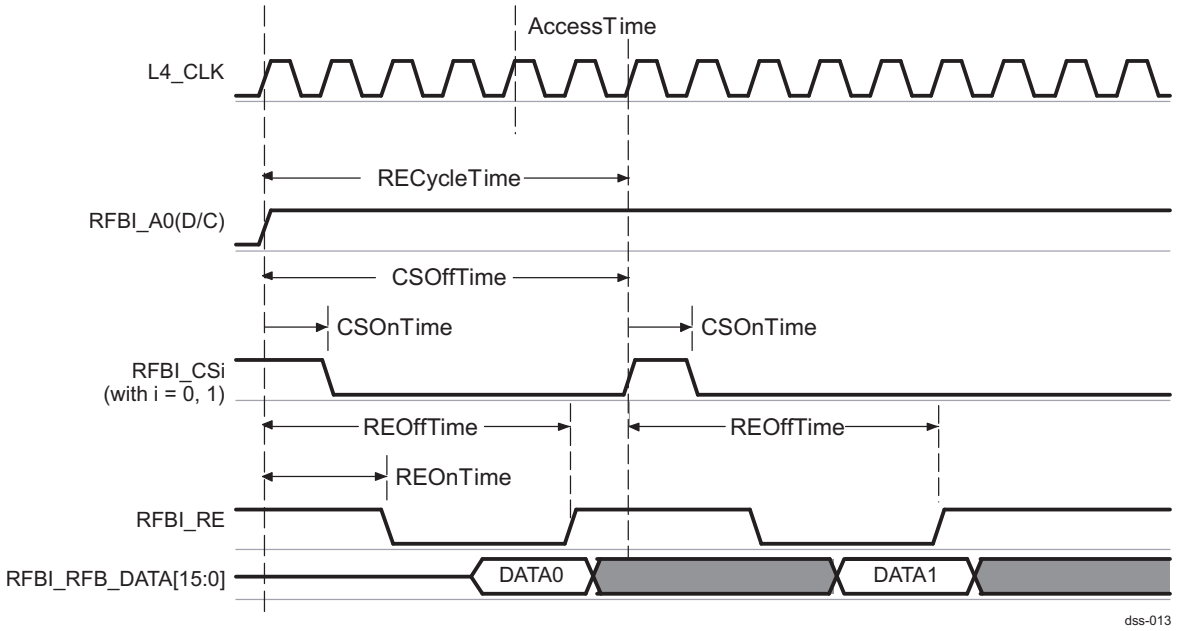
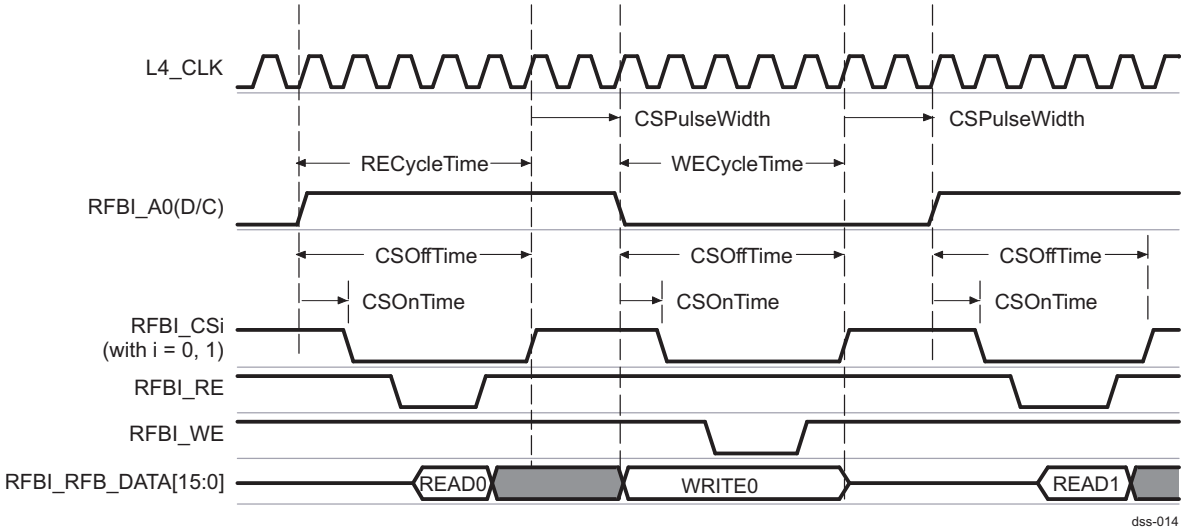


Figure 15-16. Read to Write and Write to Read



- Timing diagrams in bypass mode

  Figure 15-17 through Figure 15-32 show timing diagrams of synchronization signals and pixel clock in bypass mode for both passive matrix and active matrix panels. The display controller directly drives these signals, which are related to the programmable fields described in Table 15-6.

Table 15-6. Programmable Fields in Bypass Mode

Name	Register	Description
PPL	DSS.DISPC_SIZE_LCD[10:0] PPL bit field value + 1	Pixels per line (PPL)
LPP	DSS.DISPC_SIZE_LCD[26:16] LPP bit field value + 1	Lines per panel
HBP	DSS.DISPC_TIMING_H[31:20] HBP bit field value + 1	Horizontal back porch
HFP	DSS.DISPC_TIMING_H[19:8] HFP bit field value + 1	Horizontal front porch
HSW	DSS.DISPC_TIMING_H[7:0] HSW bit field value + 1	Horizontal synchronization pulse width

Table 15-6. Programmable Fields in Bypass Mode (continued)

Name	Register	Description
VBP	DSS.DISPC_TIMING_V[31:20] VBP bit field value	Vertical back porch
VFP	DSS.DISPC_TIMING_V[19:8] VFP bit field value	Vertical front porch
VSW	DSS.DISPC_TIMING_V[7:0] VSW bit field value + 1	Vertical synchronization pulse width
ONOFF	DSS.DISPC_POL_FREQ[17] ONOFF bit	DISPC_HSYNC and DISPC_VSYNC pixel clock control
RF	DSS.DISPC_POL_FREQ[16] RF bit	DISPC_HSYNC and DISPC_VSYNC pixel clock edge control
IEO	DSS.DISPC_POL_FREQ[15] IEO bit	Invert DISPC_ACBIAS
IPC	DSS.DISPC_POL_FREQ[14] IPC bit	Invert DISPC_PCLK
IHS	DSS.DISPC_POL_FREQ[13] IHS bit	Invert DISPC_HSYNC
IVS	DSS.DISPC_POL_FREQ[12] IVS bit	Invert DISPC_VSYNC

- Active matrix timing configuration 1
  - DSS.DISPC_POL_FREQ[17] ONOFF bit = 0
  - DSS.DISPC_POL_FREQ[16] RF bit = 0
    The DISPC_HSYNC and DISPC_VSYNC signals are driven on the opposite edge of DISPC_PCLK from the pixel data.
  - DSS.DISPC_POL_FREQ[15] IEO = 0
    The DISPC_ACBIAS signal is active high.
  - DSS.DISPC_POL_FREQ[14] IPC = 0
    The pixel data are driven on the rising edge of DISPC_PCLK.
  - DSS.DISPC_POL_FREQ[13] IHS = 0
    The DISPC_HSYNC signal is active high.
  - DSS.DISPC_POL_FREQ[12] IVS = 0
    The DISPC_VSYNC signal is active high.

Figure 15-17. Active Matrix Timing Diagram of Configuration 1 (Start of Frame)

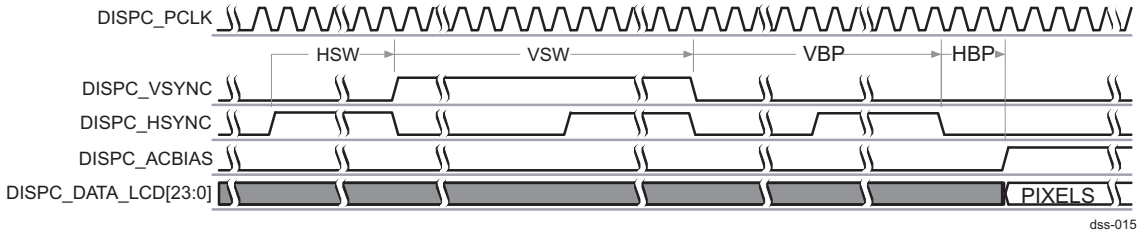


Figure 15-18. Active Matrix Timing Diagram of Configuration 1 (Between Lines)

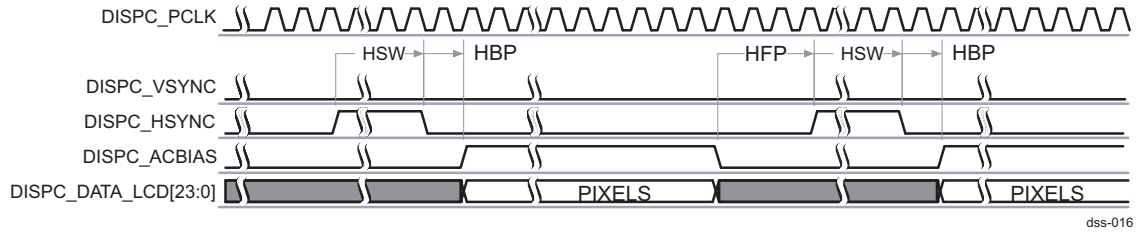


Figure 15-19. Active Matrix Timing Diagram of Configuration 1 (Between Frames)

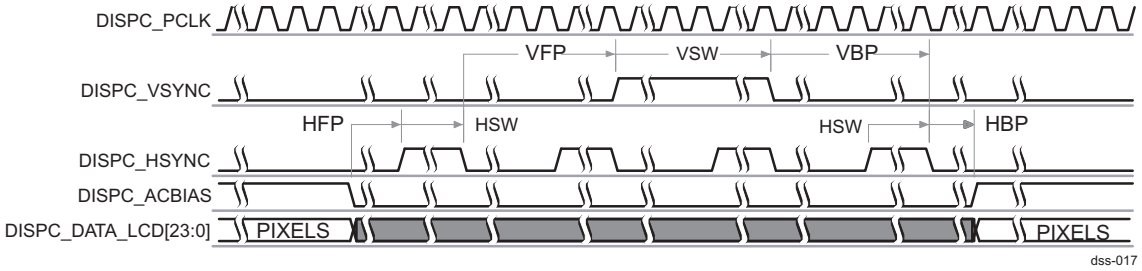
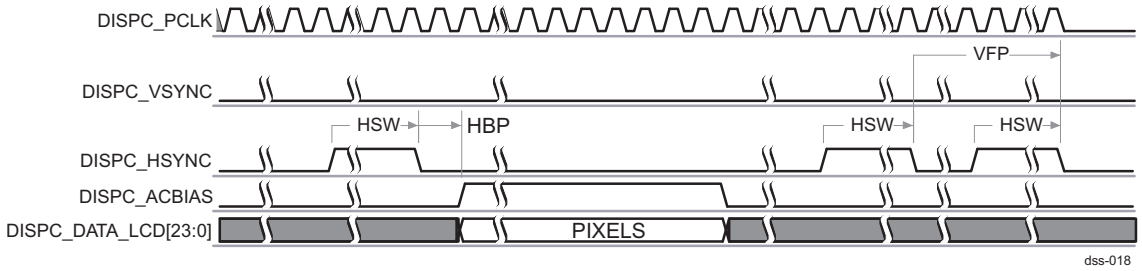


Figure 15-20. Active Matrix Timing Diagram of Configuration 1 (End of Frame)



- Active matrix timing configuration 2
  - DSS.DISPC_POL_FREQ[17] ONOFF bit = 1
  - DSS.DISPC_POL_FREQ[16] RF bit = 1
    The DISPC_HSYNC and DISPC_VSYNC signals are driven on the rising edge of DISPC_PCLK.
  - DSS.DISPC_POL_FREQ[15] IEO = 1
    The DISPC_ACBIAS signal is active low.
  - DSS.DISPC_POL_FREQ[14] IPC = 1
    The pixel data is driven on the falling edge of DISPC_PCLK.
  - DSS.DISPC_POL_FREQ[13] IHS = 1
    The DISPC_HSYNC signal is active low.
  - DSS.DISPC_POL_FREQ[12] IVS = 1
    The DISPC_VSYNC signal is active low.

Figure 15-21. Active Matrix Timing Diagram of Configuration 2 (Start of Frame)

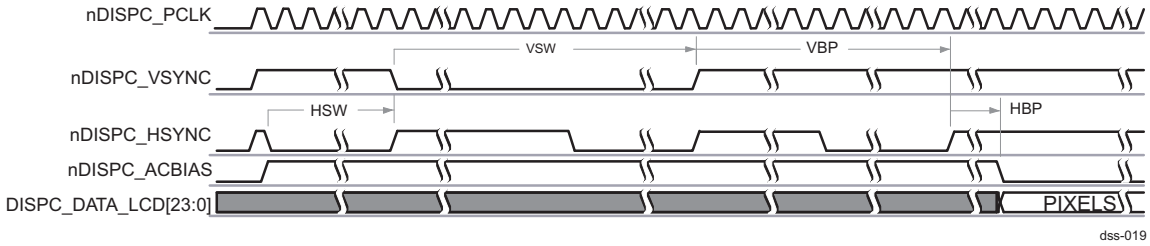


Figure 15-22. Active Matrix Timing Diagram of Configuration 2 (Between Lines)

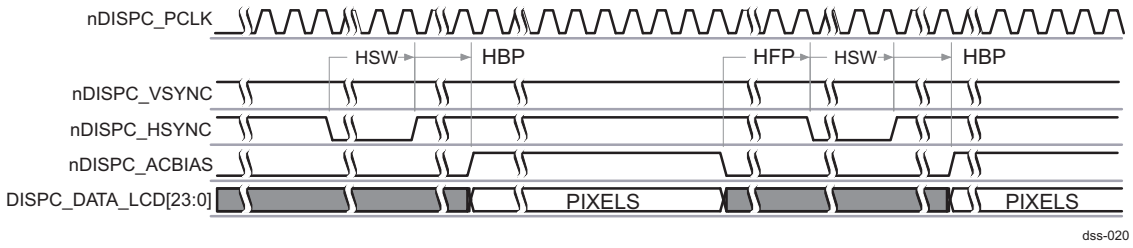


Figure 15-23. Active Matrix Timing Diagram of Configuration 2 (Between Frames)

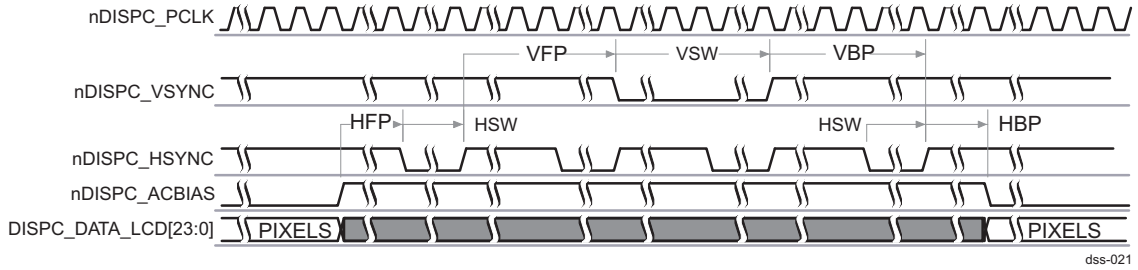
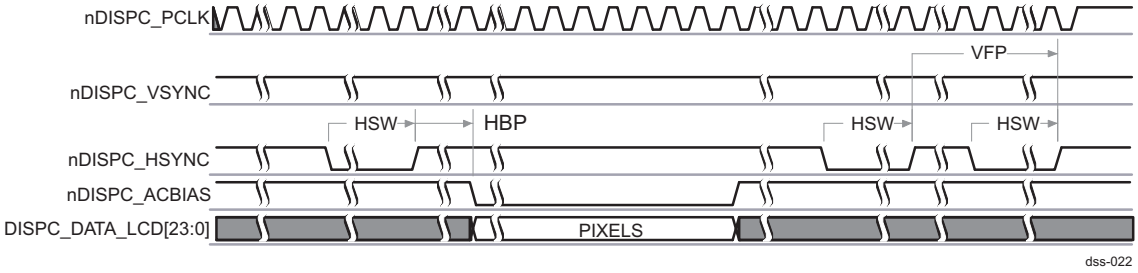


Figure 15-24. Active Matrix Timing Diagram of Configuration 2 (End of Frame)



- Active matrix timing configuration 3
  - DSS.DISPC_POL_FREQ[17] ONOFF bit = 1
  - DSS.DISPC_POL_FREQ[16] RF bit = 1
    The DISPC_HSYNC and DISPC_VSYNC signals are driven on the rising edge of DISPC_PCLK.
  - DSS.DISPC_POL_FREQ[15] IEO = 0
    The DISPC_ACBIAS signal is active high.
  - DSS.DISPC_POL_FREQ[14] IPC = 0
    The pixel data are driven on the rising edge of DISPC_PCLK.
  - DSS.DISPC_POL_FREQ[13] IHS = 0
    The DISPC_HSYNC signal is active high.
  - DSS.DISPC_POL_FREQ[12] IVS = 0
    The DISPC_VSYNC signal is active high.

Figure 15-25. Active Matrix Timing Diagram of Configuration 3 (Start of Frame)

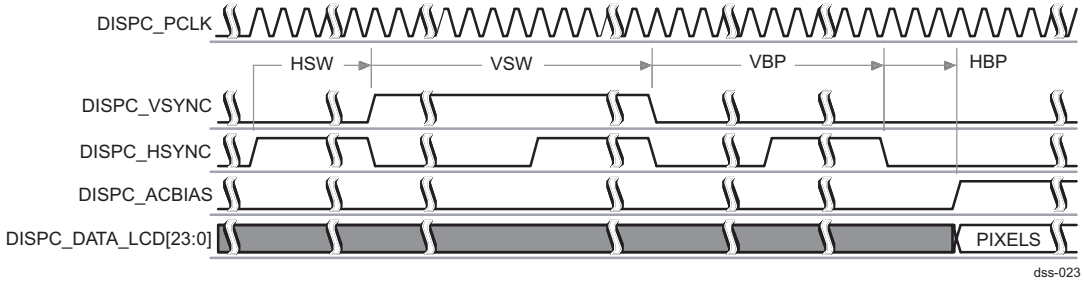


Figure 15-26. Active Matrix Timing Diagram of Configuration 3 (Between Lines)

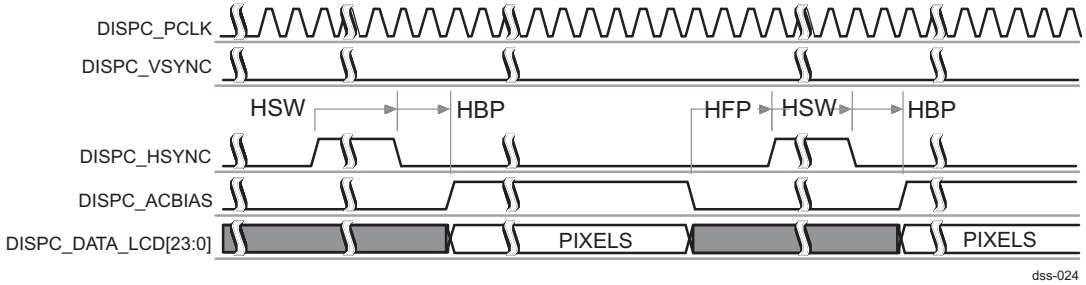


Figure 15-27. Active Matrix Timing Diagram of Configuration 3 (Between Frames)

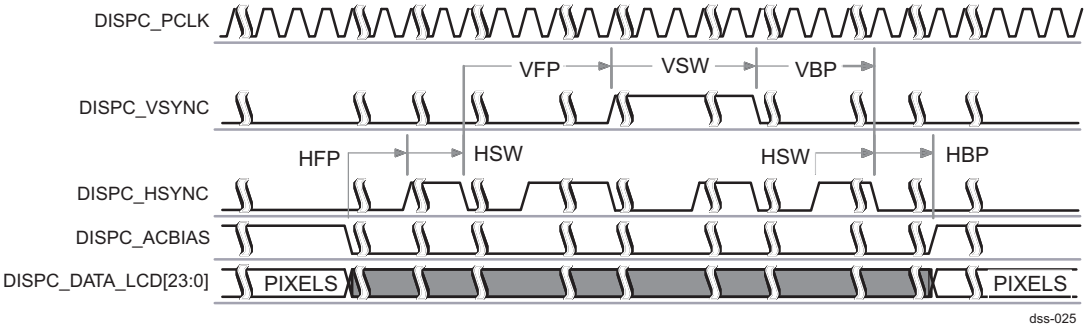
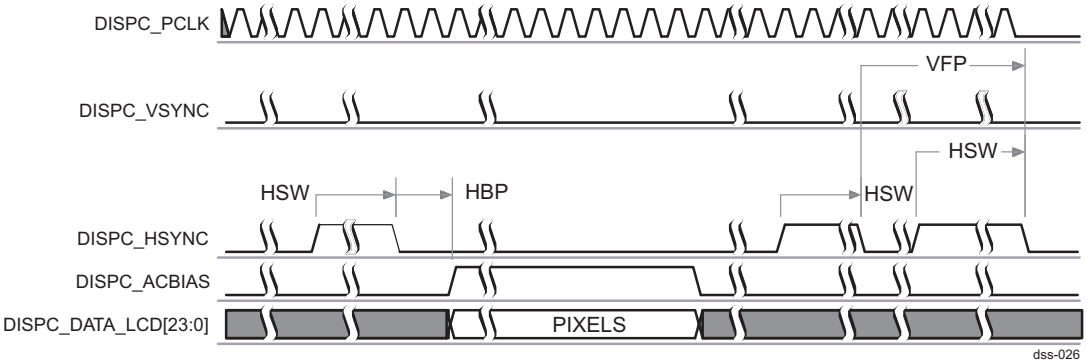


Figure 15-28. Active Matrix Timing Diagram of Configuration 3 (End of Frame)



- Passive matrix timing configuration
  - DSS.DISPC_POL_FREQ[17] ONOFF bit = 0
  - DSS.DISPC_POL_FREQ[16] RF bit = 0
    The DISPC_HSYNC and DISPC_VSYNC signals are driven on the opposite edge of DISPC_PCLK from the pixel data.
  - DSS.DISPC_POL_FREQ[15] IEO = 0
    The DISPC_ACBIAS signal is active high.
  - DSS.DISPC_POL_FREQ[14] IPC = 0
    The pixel data are driven on the rising edge of DISPC_PCLK.
  - DSS.DISPC_POL_FREQ[13] IHS = 0
    The DISPC_HSYNC signal is active high.
  - DSS.DISPC_POL_FREQ[12] IVS = 0
    The DISPC_VSYNC signal is active high.

Figure 15-29. Passive Matrix Timing Diagram (Start of Frame)

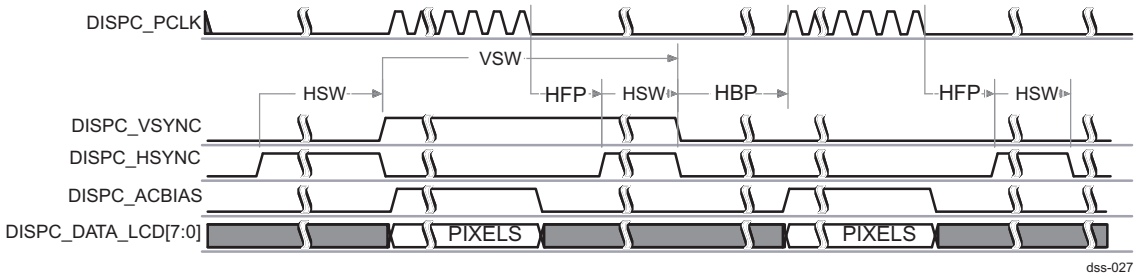


Figure 15-30. Passive Matrix Timing Diagram (Between Lines)

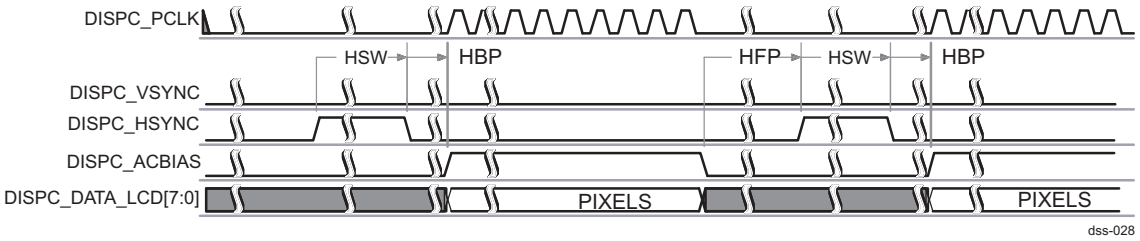


Figure 15-31. Passive Matrix Timing Diagram (Between Frames)

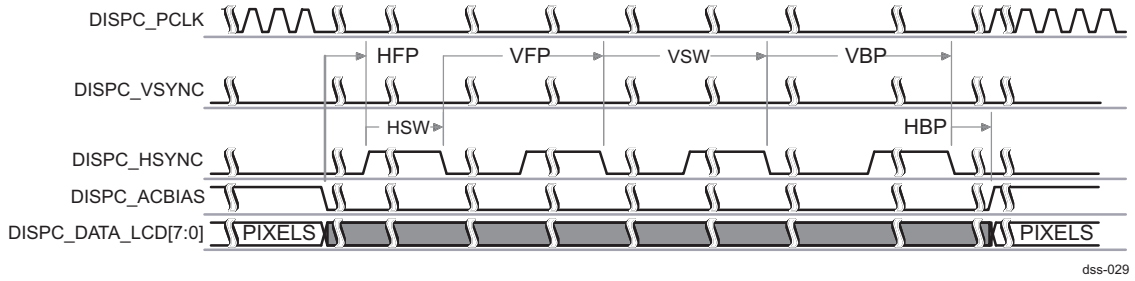
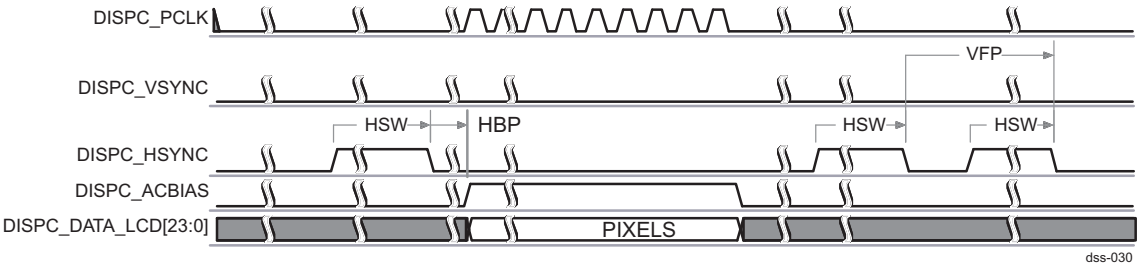


Figure 15-32. Passive Matrix Timing Diagram (End of Frame)



15.2.1.2 SDI Serial Interface

In the serial interface, the modules in the display subsystem path are the display controller and the SDI with its associated PLL. The display controller provides the required control signals to interface the memory frame buffer (either SDRAM or SRAM) directly to the external displays. The display controller is connected to the memory through the L3 interconnect and has its own DMA to read the data from the system memory.

The SDI module is an additional display output mode that facilitates the connection of displays over relatively few conductors.

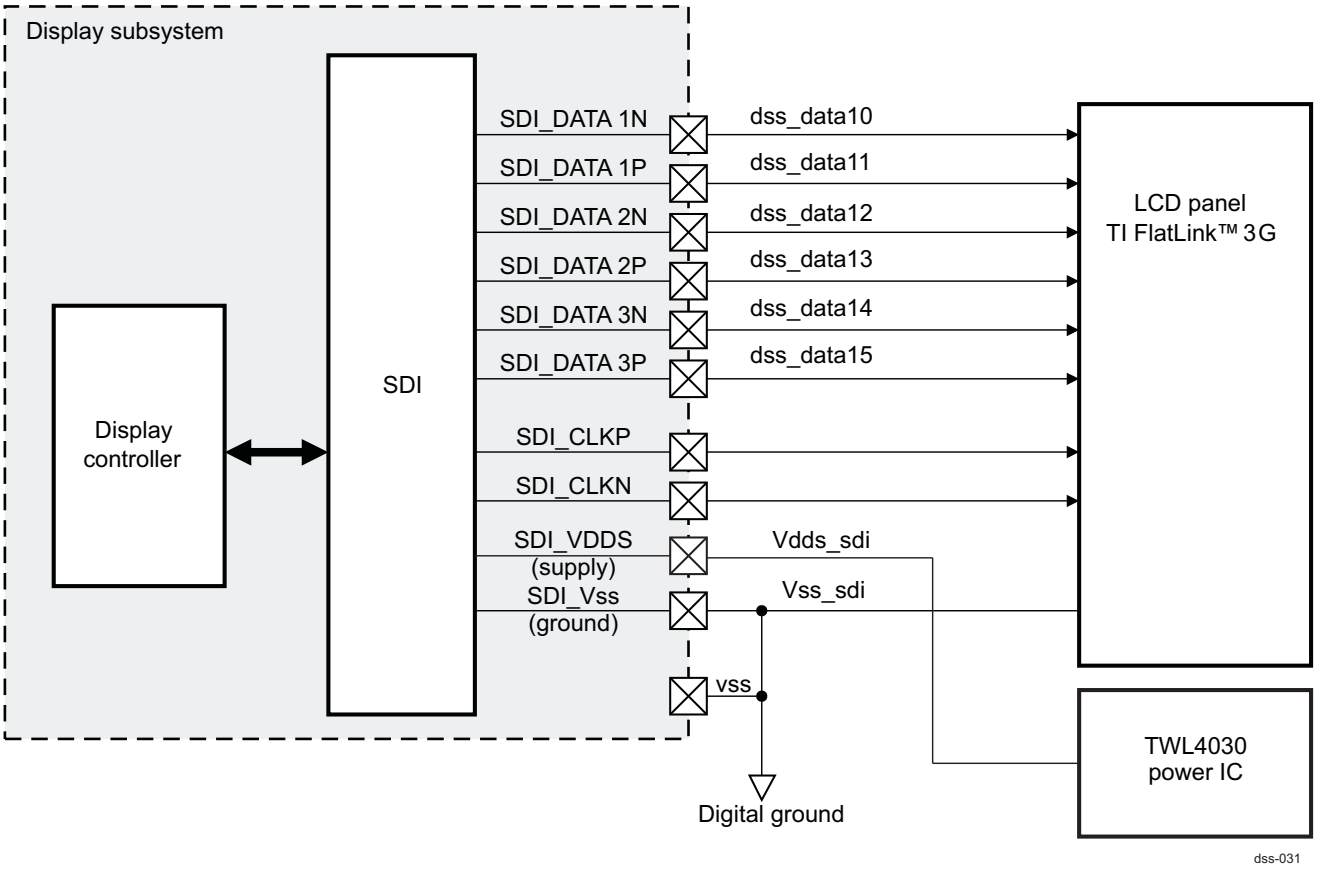
The SDI module implements the TI FlatLink3G protocol and converts the conventional parallel display signals (including pixel data) from the display controller to serial form. Texas Instruments provides a global solution with OMAP device associated to a programmable 27-bit serial display interface receiver - SN65LVDS302 receiver for 3-data pair, SN65LVDS304 receiver for 2-data pair, and SN65LVDS306 receiver for 1-data pair.

The SDI module contains a PLL to multiply the pixel clock by an appropriate factor. The resulting serialized data is then transmitted on 1 to 3 differential data pairs and an additional clock pair that has a reference transition at the boundary between pixels.

Note: Only one 9-bit RFBI driven display can be run at the same time as the SDI.

Figure 15-33 shows a typical connection between the SDI module and a compliant panel display.

Figure 15-33. Typical SDI Connection



CAUTION

When routing the PCB, ensure that the connections between the OMAP device and the LCD panel meet FlatLink3G specifications, which are available in the 27-bit serial display interface receiver data sheet.

Notes:

- The SDI pins are multiplexed in mode 1 with some LCD parallel pins. See Chapter 7, *System Control Module*, for more details on pad multiplexing.
- The connection from the vss_sdi pin of the device to the PCB digital ground must be as short as possible.
- The vdds_sdi dedicated power supply is provided by a power IC. Texas Instruments provides a global solution with the OMAP device associated with a TWL4030 power IC.

Table 15-7 describes the interface signals between the SDI module of the OMAP device and the LCD panel.

Table 15-7. Interface Signals Between the SDI Module and LCD Panel

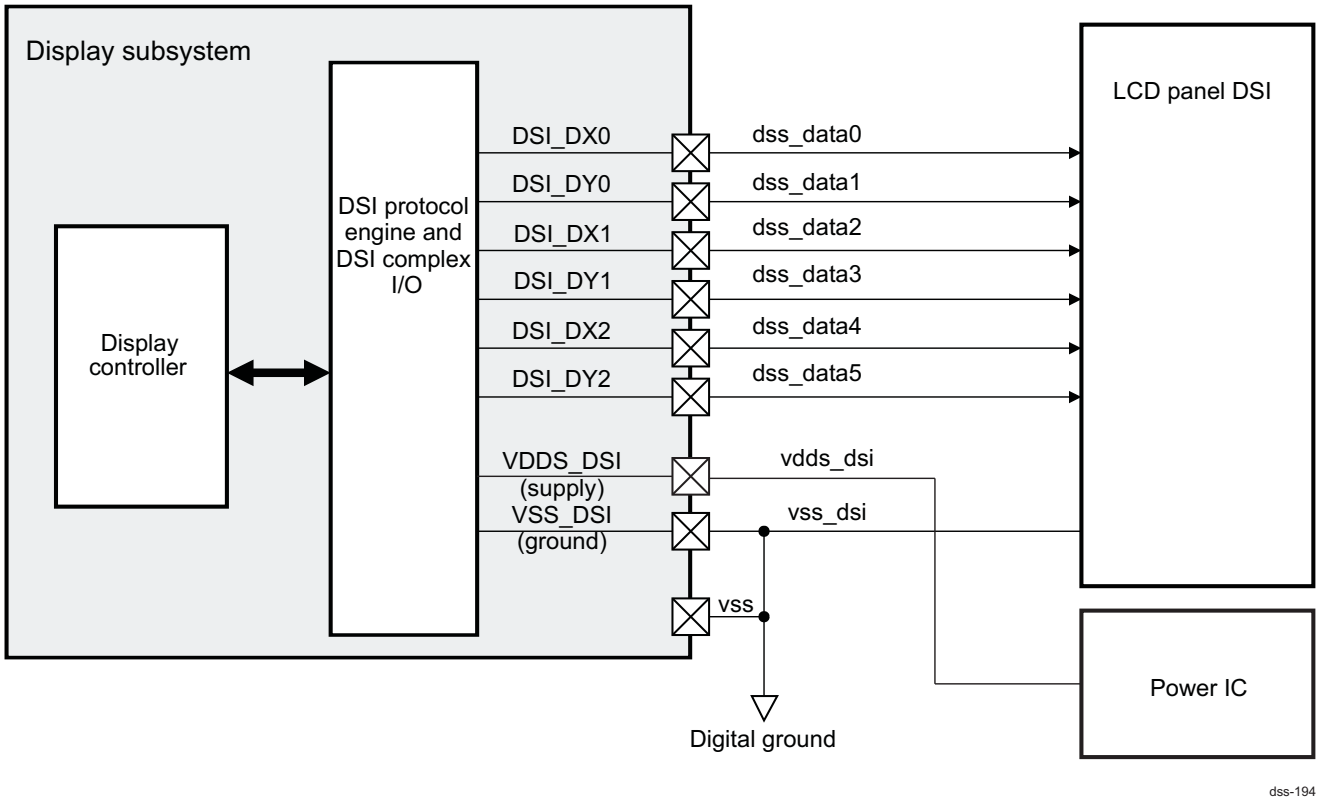
Signal Name	Type[(1)]	Description
SDI_DATA1N	O	First pair of differential data signals
SDI_DATA1P		
SDI_DATA2N	O	Second pair of differential data signals
SDI_DATA2P		
SDI_DATA3N	O	Third pair of differential data signals
SDI_DATA3P		
SDI_CLKN	O	Pair of differential clock signals
SDI_CLKP		
VDDS_SDI	PWR	Dedicated vdds power supply for SDI module
VSS_SDI	PWR	Dedicated vss ground for SDI module

(1) O=Output PWR=Power

15.2.1.3 DSI Serial Interface

Figure 15-34 shows a typical connection between the DSI modules and a compliant panel display.

Figure 15-34. Typical DSI Connection



Note: The DSI pins are multiplexed in mode 1 with some LCD parallel pins. See Chapter 7, *System Control Module*, for more details on pad multiplexing.

15.2.2 LCD Support With MIPI DSI 1.0 Protocol and Data Format

This section summarizes the MIPI DSI1.0 protocol and data format.

15.2.2.1 Physical Layer

Table 15-8 lists the DSI interface I/O.

Table 15-8. I/O Description for DSI Serial Interface

Signal Name	I/O[1]	Description	Value at Reset	
dsi_dx0	line 1	I/O	Serial data/clock input/output	N/A
dsi_dy0				
dsi_dx1	line 2	I/O	Serial data/clock input/output	N/A
dsi_dy1				
dsi_dx2	line 3	I/O	Serial data/clock input/output	N/A
dsi_dy2				

(1) I/O = Input/Output

Note: Each serial line (line 1, line 2, and line 3) can be used as clock lane or data lane. All polarities are supported. The MIPI DSI 1.0 protocol requires at least one clock line and one data lane.

Lanes support four operating modes:

- HS mode: High-speed transmit mode
- LP mode: Low-power transmit mode (also called low-power state [LPS])
- ULPS: Ultra-low power state used between two transmissions
- Off mode: Lane is off.

15.2.2.1.1 Data/Clock Configuration

From the device point of view, the DSI interface consists of six input/output signals representing three differential signals: The serial clock and one or two serial data. The minimum configuration is one data pair and one clock pair.

- The data signal carries the bit-serial data. The DSI transmitter in the host sends the data in-quadrature with the DDR clock in high speed mode; otherwise, the clock is extracted from the received data in low-speed mode. The data is transmitted byte-wise least significant bit (LSB) first.
- The clock signal carries the DDR clock signal in high speed transmission.
- Software users must configure the order of the data lanes to indicate the byte order while splitting the byte stream for each DSI_PHY into bytes.

Table 15-9 details all the DSI lanes configuration.

Table 15-9. DSI Lane Configuration

DSI DSI_PHY Lane Configuration	Data/Clock Lane Position			Description
	1	2	3	
Mode CLK + DATA1	CLK	DATA1	Not used	Single data lane
	CLK	Not used	DATA1	
	DATA1	CLK	Not used	
	Not used	CLK	DATA1	
	DATA1	Not used	CLK	
	Not used	DATA1	CLK	
Mode CLK + DATA1 + DATA2	CLK	DATA1	DATA2	Two data lanes
	CLK	DATA2	DATA1	
	DATA1	CLK	DATA2	
	DATA2	CLK	DATA1	
	DATA1	DATA2	CLK	
	DATA2	DATA1	CLK	

Notes:

- The byte on Dn is sent before the byte on Dn+1, all the combinations of data and clock are supported through programming of the DSS.DSI_COMPLEXIO_CFG1 register. The CLOCK_POSITION and CLOCK_POL bit fields configure which lane transmits the clock and define its polarity. Four bit fields (DATA1_POSITION, DATA1_POL, DATA2_POSITION, and DATA2_POL) configure the data lanes and their polarity. The DATA2_POSITION bit field can be set to 0; in this case, only the data lane defined in the DATA1_POSITION bit field is used, and data is transmitted on only one clock lane and one data lane.
- The configuration of the DSI complex I/O (number of data lanes, position, differential order) should not be changed while DSS.DSI_CLK_CTRL[20] LP_CLK_ENABLE bit is set to 1. For the hardware to recognize a new configuration of the complex I/O (done in DSS.DSI_COMPLEXIO_CFG1 register), it is recommended to follow this sequence: First set the DSS.DSI_CTRL[0] IF_EN bit to 1, next reset the DSS.DSI_CTRL[0] IF_EN to 0, then set DSS.DSI_CLK_CTRL[20] LP_CLK_ENABLE to 1, and finally, set again the DSS.DSI_CTRL[0] IF_EN bit to 1. If the sequence is not followed, the DSI complex I/O configuration is undetermined.
- Only DATA1 is bidirectional in command mode. The low-power received information is always sent by the display panel using DATA1. Since any lane of the DSI complex I/O can be configured as data lane DATA1, all lanes of the complex I/O are bidirectional

15.2.2.1.2 ULPS

Each lane can be put in ultra-low power state (ULPS) by software configuration. The ULPS mode requires all the following conditions:

- The lane must be in stop state.
- For data lanes, no data must be pending in the DSI module.
- For data lane 1, no BTA should have been sent. The DSI module should have control of the bus.

The control of each lane is independently controlled by the DSS.DSI_COMPLEXIO_CFG2 register.

15.2.2.2 Video Port (VP) Interface

Note: The signals described in this section are internal and not bounded outside the device. This section aims at helping software users understand the internal connections between the display controller (DISPC) and the DSI protocol engine.

Table 15-10 summarizes the video interface signals. This interface is used to connect the display controller to the DSI protocol engine to send real time data streams. Note that only the active matrix timings are supported by DSI protocol engine. The HSYNC/VSYNC/DE/DATA signals are driven on the rising or falling edge of the pixel clock (VP_PCLK).

Table 15-10. Video Interface for DSI Protocol Engine

Signal Name	Type[(1)]	Description
VP_HSYNC	I	Horizontal sync signal
VP_VSYNC	I	Vertical sync signal
VP_DATA[23:0]	I	Parallel output data: Bits 0 to 23
VP_PCLK	I	Pixel clock. In case of STALL configuration, it is used to indicate when new data is on the data bus during the clock period of VP_CLK.
VP_DE	I	Data enable
VP_STALL	O	The stall signal must be deasserted to receive pixel and asserted to stop receiving pixel. (It can be used only while the display controller is configured in STALL mode; in that mode, HSYNC and VSYNC are not generated).

(1) I = Input, O = Output

Table 15-10. Video Interface for DSI Protocol Engine (continued)

Signal Name	Type[1]	Description
VP_CLK	I	Display controller internal clock. It is a free-running clock. The VP_CLK is a divided clock from VP_PCLK.

Notes:

- The polarities of VP_HSYNC and VP_VSYNC are programmable by setting the DSS.DSI_CTRL register.
- The maximum frequency for VP_CLK is 173 MHz at nominal voltage, and 96 MHz at low voltage.
- Clocks VP_CLK and VP_PCLK can have the same frequency.
- The number of bits to be captured on the video port is defined in the DSS.DSI_CTRL[7:6] VP_DATA_BUS_WIDTH bit field.
- VP_DE is connected to the dss_acbias signal in the display controller, and its polarity can be controlled by setting the DISPC_POL_FREQ[15] IEO bit.

The data received on the video port can be stored into the line buffer memories or sent directly on the DSI interface in two cases:

- The line buffer size is too small compared to the line from the display controller.
- There is no line buffer instantiated. If there is no line buffer, the burst mode, defined as frequency burst mode, can not be used. Only the transparency burst mode is supported.

Note: The DSS.DSI_CTRL[13:12] LINE_BUFFER bit field defines the number of lines to be used for transferring data from the video port to the DSI link.

15.2.2.2.1 Video Port Used for Video Mode

If the video port is used for video mode, the VP_STALL is not used. Table 15-11 lists the active signals on the video port.

Table 15-11. Video Interface in the Context of Video Mode

Signal Name	Type[1]	Description
VP_HSYNC	I	Horizontal sync signal
VP_VSYNC	I	Vertical sync signal
VP_DATA[23:0]	I	Parallel output data: bits 0 to 23
VP_PCLK	I	Pixel clock.
VP_DE	I	Data enable
VP_CLK	I	It is a free running clock used as the display controller functional clock. The maximum frequency is 173 MHz at nominal voltage, and 96 MHz at low voltage.

(1) I = Input, O = Output

Three video modes are available:

- No line buffer: The data received on the video port are directly output of the DSI port without buffering. The ration of VP_CLK and the DSI high-speed (HS) clock period must ensure the same throughput on the two ports (the two clocks must be generated using the same PLL; the subsystem must provide such configuration).
- One line buffer: The data are first stored in the line buffer; once all the data for one line are received, the DSI protocol engine sends the whole line. Software must adjust timings to allow for storing all line data into the line buffer before sending it to DSI outputs. The synchronization packets are never stored into the line buffer.
- Two line buffers: One line is stored when the second line is output on the DSI port. It allows burst capability. While receiving the first line of the frame, there is no RGB packets sent because the line

buffers are empty. To send the last line of the frame, a dummy line must be provided by the display controller to flush the line buffers. The synchronization packets are never stored into the line buffer.

Note: If more active lines are received on the video port than the number defined in the DSS.DSI_VM_TIMING3[15:0] VACT bit field, the extra lines are discarded by the DSI protocol engine. These lines are treated as blanking lines.

Figure 15-35, Figure 15-36, and Figure 15-37 show these three video modes.

Figure 15-35. DSI Video Mode Without Burst (No-Line Buffer)

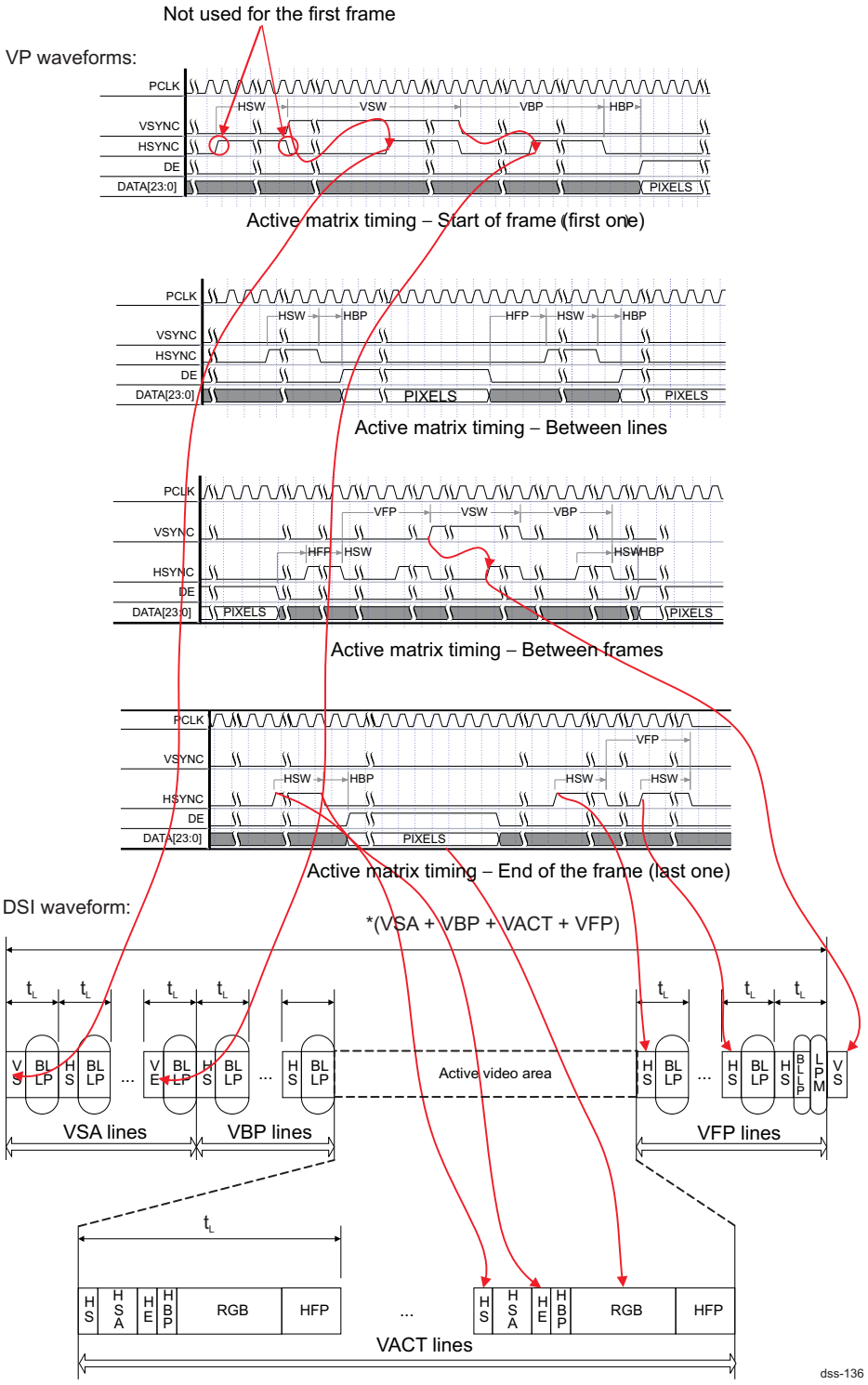
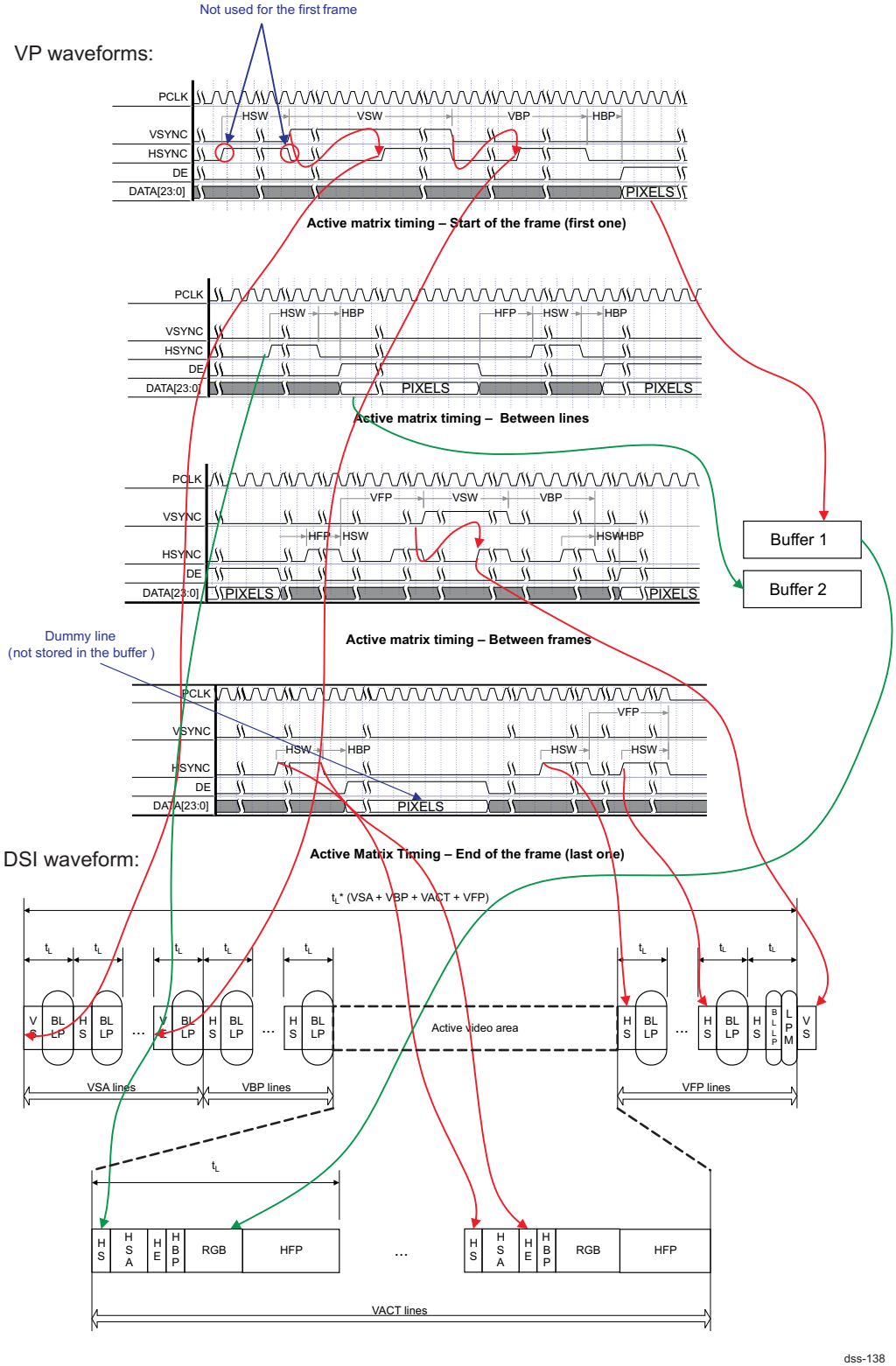


Figure 15-36. DSI Video Mode Without Burst (One-Line Buffer)



Figure 15-37. DSI Video Mode With Burst (Two-Line Buffers)



Note: In Figure 15-35, Figure 15-36, and Figure 15-37:

- When HSync start and Hsync end short packets are not generated (HSA does not exist), HBP signal must be different from 0.
- The software must ensure that VBP is always defined so that there is at least one HSYNC during VBP.
- In blanking low-power mode (BL-LP), two options are possible
  - The lane remains in ULPS, and the DSI_CTRL[20] BLANKING_MODE bit is set to 0x0.
  - Dummy bytes are sent during LP with the DSI_CTRL[20] BLANKING_MODE bit set to 0x1; the number of sent bytes is determined by the DSI_VM_TIMING6[15:0] BL_LP_INTERLEAVING bit field.

If the signal VP_DE is not asserted during enough VP_PCLK cycles to be able to capture the number of bytes defined in the word count of the header, the module must send the valid data received on the video port followed by bytes of 0s to match the required number of bytes to transmit. The VP_PCLK must be present during all extra cycles where the DSI protocol engine is expecting pixels.

If the VP_DE signal is asserted for too many VP_PCLK cycles, the module should stop capturing the data on the video port while the number of bytes to capture, as defined in the word count field of the header, is reached.

The HS must check that the received synchronization events on the video port (VSYNC and HSYNC) are within the synchronization window based on expected timings. If the timings (internal and received) are out of sync, the interrupt for out-of-sync should be generated and the interface must be disabled (DSS.DSI_CTRL[0] IF_EN bit is automatically reset by hardware). The unsynchronization window is defined by the DSS.DSI_VM_TIMING2[27:24] WINDOW_SYNC bit field.

15.2.2.2.2 Video Port Used on Command Mode

If the video port is used for command mode, the VP_HSYNC, VP_VSYNC, and VP_DE signals are not used. Table 15-12 describes the active signals on the video port.

Table 15-12. Video Interface in the Context of Command Mode

Signal Name	Type[(1)]	Description
VP_DATA[23:0]	I	Parallel output data: bits 0 to 23
VP_PCLK	I	One pulse is generated every time new data is output on the data bus
VP_STALL	O	The stall signal must be deasserted to receive pixel and asserted to stop receiving pixel. (It can be used only while the display controller is configured in STALL mode; in that mode, HSYNC and VSYNC are not generated).
VP_CLK	I	Display controller internal clock: It is a free running clock used as the display controller functional clock. The maximum frequency is 173 MHz at nominal voltage and 96 MHz at low voltage.

(1) I = Input, O = Output

Note: The stall signal must be deasserted to receive pixels and asserted to stop receiving pixels.

Figure 15-38 and Figure 15-39 show the VP_STALL signal assertion and deassertion on rising and falling edges, respectively.

Note: In DSI command mode, the display controller must be configured in stall mode by setting the DSS.DISPC_CONTROL[11] STALLMODE bit to 1.

Figure 15-38. Stall Timing With Pixel on Rising Edge

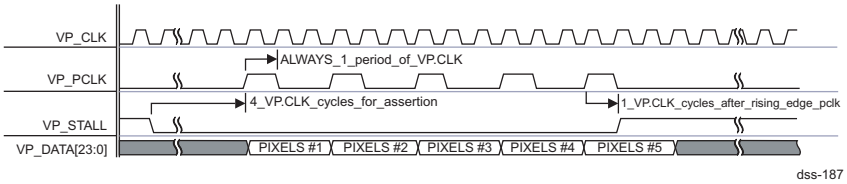
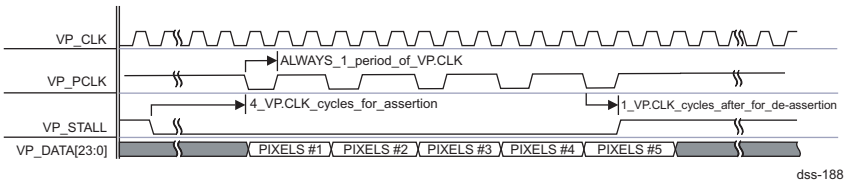


Figure 15-39. Stall Timing With Pixel on Falling Edge



To stop the transfer, the VP_STALL signal must be asserted when the last data is output. The data can be output on the rising or falling edge of the VP_PCLK through registers in the display controller module. The case with data output on falling edge of VP_PCLK is supported by the DSI protocol engine.

The VP_PCLK clock is generated from VP_CLK; these two clocks are balanced. Assertion and deassertion of VP_PCLK is done on the rising edge of VP_CLK. The width of the VP_PCLK pulse depends on the configuration of the clock divisor in the display controller (DSS.DISPC_DIVISOR[7:0] PCD bit field). In the DSI protocol engine, the information is defined in the DSS.DSI_CTRL[4] VP_CLK_RATIO bit and should be aligned with the display controller configuration.

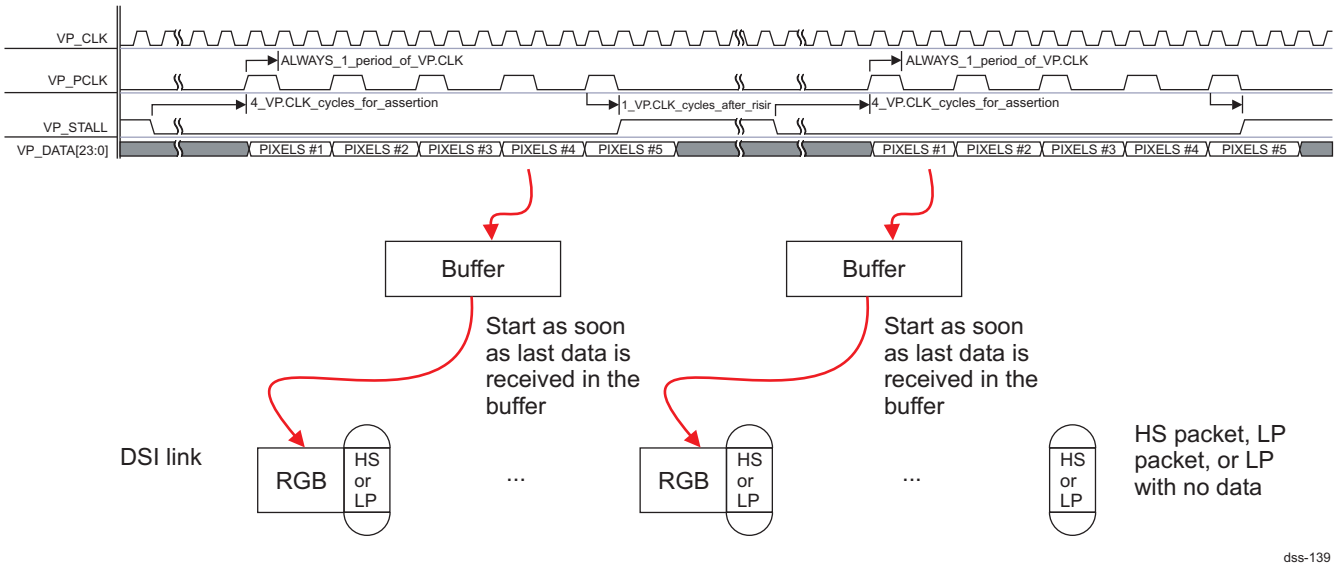
Deassertion of the VP_STALL signal should occur at least 4 VP_CLK cycles before assertion of VP_PCLK. Assertion of VP_STALL should occur one cycle VP_CLK after deassertion of VP_PCLK for the last pixel to be transferred. The VP_CLK clock is generated by the display controller under software control. It can be kept running between assertion and deassertion of VP_STALL.

The word count (WC) defined in the DSS.DSI_VCn_LONG_PACKET_HEADER register for the virtual channel (VC)associated with the video port, indicates the number of bytes to receive (one line or two lines can be used, depending on the WC and size of the line buffer). The total size defined in the WC of the header register can not exceed the size of the line buffer multiplied by the number of buffer lines.

The stall assertion/deassertion depends on the number of bytes to be received considering the size of the video port bus defined in the DSS.DSI_CTRL[7:6] VP_DATA_BUS_WIDTH bit field.

Figure 15-40 shows the data flow in command mode using the video port.

Figure 15-40. Data Flow in Command Mode Using the Video Port



Two command modes are available:

- One line buffer: The data are stored in the line buffer before being sent.
- Two line buffers: The two lines are used if the word count defined in the DSS.DSI_VCn_LONG_PACKET_HEADER register is bigger than the line size; otherwise, one line buffer is used.

Note: In command mode, the video port can only be used in one or two line buffer configuration. The no-line buffer configuration is not allowed.

The packets can be sent using high-speed or low-speed.

Note: The DCS command in the payload can be inserted automatically using the DSI_CTRL[24] DCS_CMD_ENABLE bit. If TE is used, hardware automatically inserts the DCS Write Start command for the first packet of the frame transfer and the DCS Write Continue command for all subsequent packets.

15.2.2.2.3 Burst Mode

When the burst mode is enabled, the video port receives data from the display controller at the pixel clock. The DSI protocol engine buffers the data in its own line FIFO (double-line buffer of 1024 x 24-bit pixels maximum). The read speed of the line can be twice the pixel clock to increase the blanking time of the video mode and to allow command mode traffic to be interleaved during the blanking period. The burst mode uses a dual-line buffer.

The DSI port can output data from one line buffer while the second one is accessed by the video port. The two processes are concurrent but they do not access the same line at the same time. The DSI transfer can start only when the whole video port line is transferred into a line buffer. The switch is controlled by the VP_HS signal on the video port side and by internal signal on the DSI port indicating that the last data for the current line has been written into the line buffer.

Note: The line buffers are used to store the pixels only. The synchronization codes are not stored in the line buffers. They should be sent according to the video port timings.

15.2.2.3 Multilane Layer

Peripherals do not typically have high bandwidth requirements for returning data to the host processor. To keep designs simple and improve interoperability, all DSI-compliant systems should only use Lane 1 in LP mode for returning data from a peripheral to the host processor.

15.2.2.3.1 SoT and EoT in Multilane Configurations

Since a HS transmission is composed of an arbitrary number of bytes that may not be an integer multiple of the number of lanes, some lanes may run out of data before others. Therefore, the lane management layer, as it buffers up the final set of less-than-N bytes, deasserts its *valid data* signal into all lanes for which there is no further data. Although all lanes start simultaneously with parallel SoTs, each lane operates independently and may complete the HS transmission before the other lanes, sending an EoT one cycle (byte) earlier.

15.2.2.3.2 Lane Splitter

The lane splitter can split the byte stream into 2 lanes (for 1 lane, the splitter is bypassed). Figure 15-41 and Figure 15-42 show the byte position into each serial link for 1 and 2 data lane configurations. The byte stream always starts from lane 1. It finishes on one of the lanes, depending on the number of bytes to send and the number of lanes. The splitter module is only used when packets are sent using high-speed mode (HS). In low-power mode (LP), only one data lane is used (Lane 1).

Figure 15-41. Two Data Lane Configuration

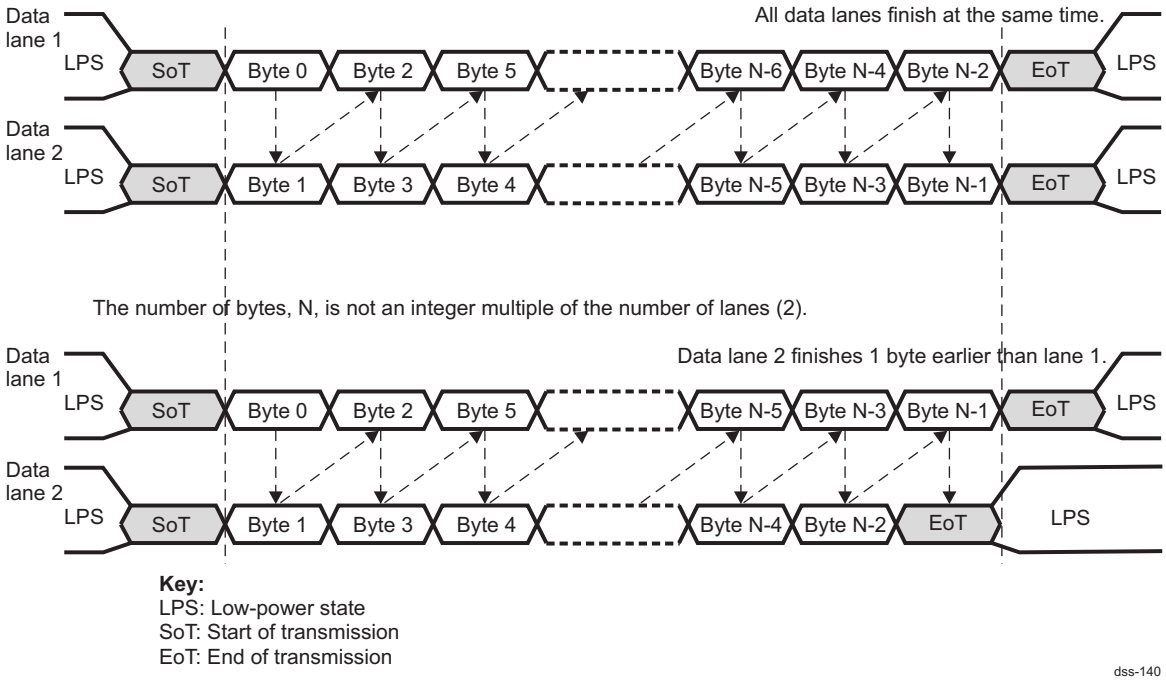
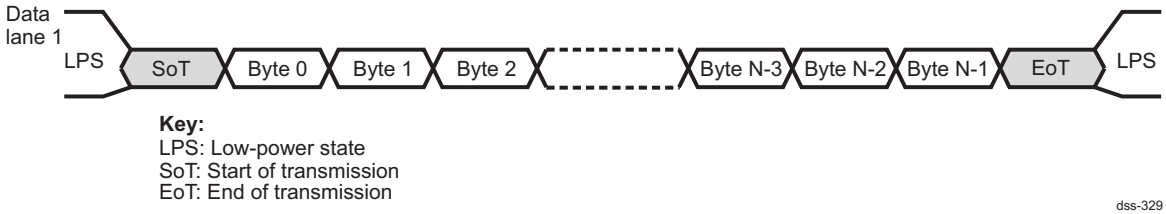
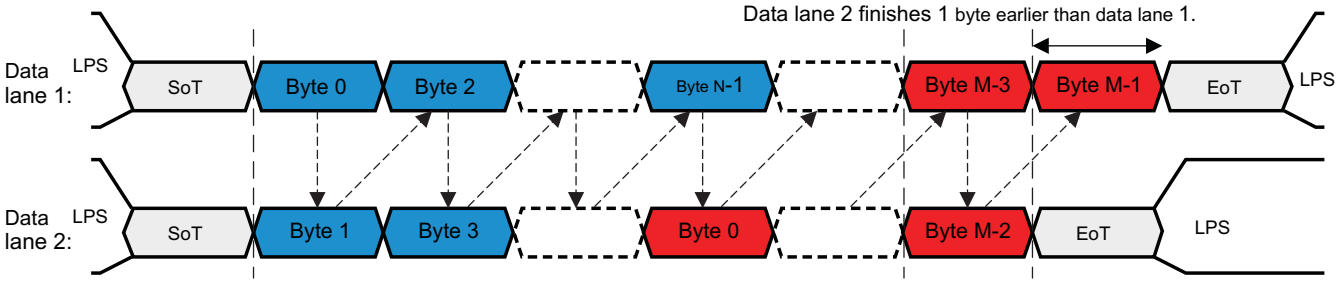


Figure 15-42. One Data Lane Configuration



In case of back-to-back packets, the byte stream is considered as a single packet by the splitter module. Figure 15-43 shows the example of two packets sent back to back. N bytes for the first packet and M bytes for the second one.

Figure 15-43. Two Packets Using Two-Data Lane Configuration (Example)



Key:
LPS: Low-power state
SoT: Start of transmission
EoT: End of transmission

dss-142

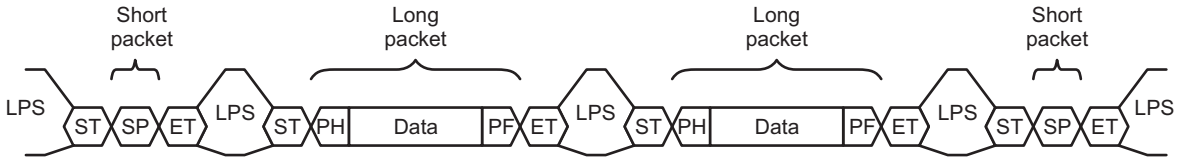
15.2.2.4 Protocol Layer

The low level protocol (LLP) is a byte-oriented protocol. It supports short and long packet formats. The DSI protocol layer defines how the display data is transported onto the physical layer. Packets can be sent using high-speed or low-speed mode. LLP is selected through DSI registers. The features of the DSI protocol layer are:

- Transport of arbitrary data (payload independent)
- 8-bit word size
- Support for up to four interleaved VCs on the same link
- Special packets for frame start, frame end, line start, and line end information
- Descriptor for the type, pixel depth, and format of application-specific payload data
- ECC for 1-bit or 2-bit error detection in the header
- 16-bit checksum code for error detection

Figure 15-44 shows the protocol layer with short and long packets.

Figure 15-44. Protocol Layer With Short and Long Packets



Key:

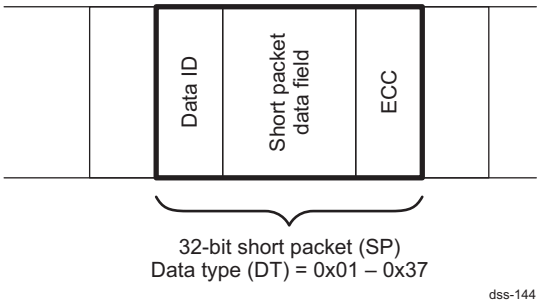
ST: Start of transmission	ET: End of transmission
PH: Packet header	PF: Packet footer
LPS: Low-power state	SP: Short packet

dss-143

15.2.2.4.1 Short Packet

Figure 15-45 shows the structure of the short packet. A short packet should contain an 8-bit data ID followed by two command or data bytes and an 8-bit ECC; a packet footer should not be present. Short packets should be four bytes in length. The ECC byte allows correction of single-bit errors and detection of 2-bit errors in the short packet.

Figure 15-45. Short Packet Structure

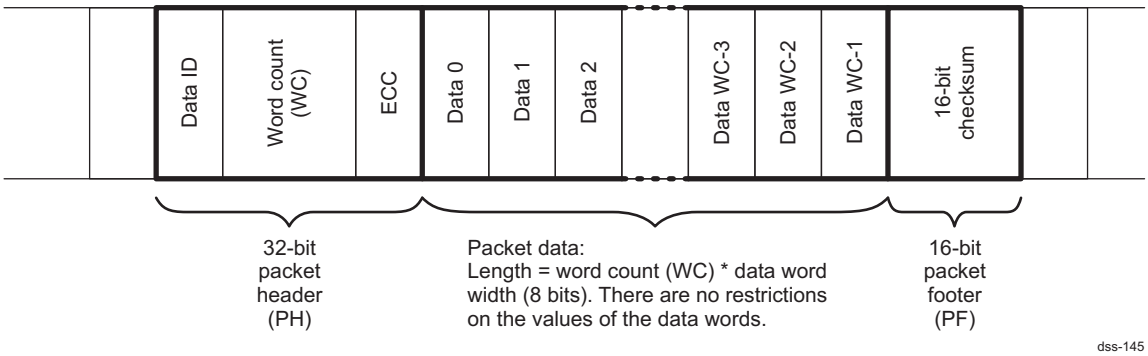


Note: The short packets can be sent in low-power mode or in high-speed mode.

15.2.2.4.2 Long Packet

Figure 15-46 shows the structure of the low-level protocol long packet. A long packet must consist of three elements: A 32-bit packet header (PH), an application-specific data payload with a variable number of bytes, and a 16-bit packet footer (PF). The packet header is further composed of three elements: An 8-bit data identifier, a 16-bit word count, and 8-bit ECC. The packet footer has one element, a 16-bit checksum. Long packets can be from 6 to 65,541 bytes in length.

Figure 15-46. Long Packet Structure



- The data identifier defines the VC for the data and the DT for the application-specific payload data.
- The word count defines the number of bytes in the data payload between the end of the packet header and the start of the packet footer. Neither the packet header nor the packet footer should be included in the word count.
- The ECC byte allows single-bit errors to be corrected and 2-bit errors to be detected in the packet header. This includes the data identifier and the word count fields.

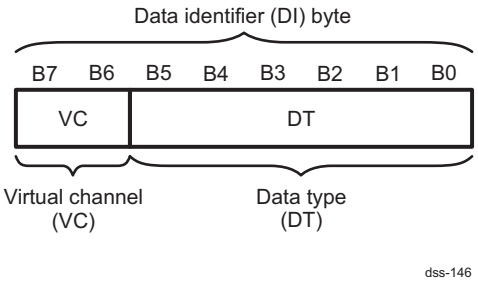
After the end of the packet header, the receiver reads the next word count * bytes of the data payload. There are no limitations on the value of a data word within the data payload block, that is, no embedded codes are used. Once the receiver has read the data payload, it reads the checksum in the packet footer. The host processor should always calculate and transmit a checksum in the packet footer. Peripherals are not required to calculate a checksum. Also note the special case of zero-byte data payload: If the payload has length 0, then the checksum calculation results in (0xFFFF). If the checksum is not calculated, the packet footer should consist of two bytes of all zeros (0x0000). In the generic case, the length of the data payload should be a multiple of bytes. In addition, each data format may impose additional restrictions on the length of the payload data, that is, multiple of four bytes. Each byte is transmitted LSB first. Payload data may be transmitted in any byte order, restricted only by data format requirements. Multibyte elements such as word count and checksum should be transmitted least-significant byte first.

Note: The long packets can be sent in low-power mode or in high-speed mode.

15.2.2.4.3 Data Identifier

The data identifier byte contains the virtual VC ID value and the DT value as illustrated in Figure 15-47. The VC ID is contained in the two MS bits of the data identifier byte. The DT value is contained in the six LSBs of the data identifier byte. DI[7:6]: These two bits identify the data as directed to one of four VCs. DI[5:0]: These six bits specify the DT.

Figure 15-47. Data Identifier Structure

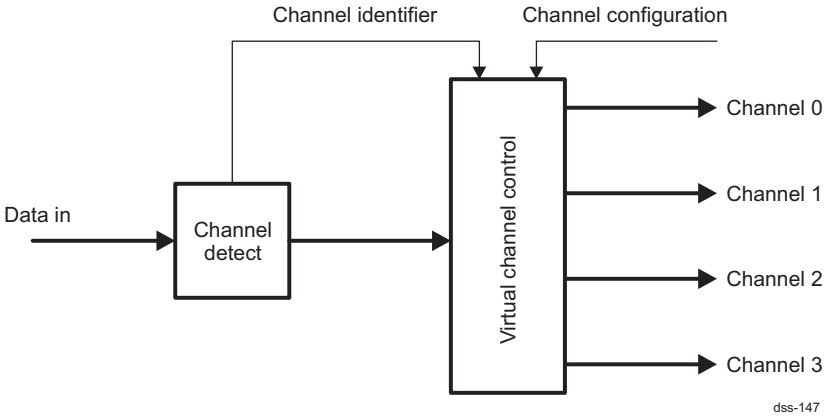


15.2.2.4.4 Virtual Channel ID - VC Field, DI[7:6]

The host can service up to four peripherals with tagged commands or blocks of data using the VC ID field of the header for packets targeted at different peripherals. The VC ID enables one serial stream to service two or more virtual peripherals by multiplexing packets onto a common transmission channel. Note that each packet sent in a single transmission have its own VC assignment and can be directed to different peripherals. The VC ID is defined in the DSS.DSI_VCn_SHORT_PACKET_HEADER and DSS.DSI_VCn_LONG_PACKET_HEADER registers for short and long packets, respectively. It should not be modified by hardware. There is one set of registers for each VC. Each set of registers defines the characteristics of the traffic between the host and the display associated with the VC.

Figure 15-48 shows the VC controller.

Figure 15-48. Virtual Channel Controller



15.2.2.4.5 Data Type Field DT[5:0]

The DT field specifies whether the packet is a long or short packet type and the packet format. The DT field, along with the word count field for long packets, informs the receiver on how many bytes to expect in the remainder of the packet. This is necessary because there are no special packet start/end sync codes to indicate the beginning and end of a packet. This permits packets to convey arbitrary data, but it also requires the packet header to explicitly specify the size of the packet.

15.2.2.4.6 Pixel Data Formats in Video Mode

The host can send different pixel formats in video mode. Table 15-13 summarizes the pixel formats supported by the DSI interface in video mode.

Table 15-13. Pixel Data Format in Video Mode

Mode	Description
RGB888 (using 24-bit container)	RGB888
RGB666 (using 24-bit container)	RGB666
RGB666 (18-bit packet using 18-bit container)	RGB666_PACKET
RGB565 (using 16-bit container)	RGB565

15.2.2.4.7 Synchronization Codes

Each frame can be identified by two synchronization codes: One for the start of vertical synchronization pulse (VSSC) and one for the end of the vertical synchronization pulse (VSEC). Each line can be identified by two synchronization codes: One for the start of horizontal synchronization pulse (HSSC) and one for the end of the horizontal synchronization pulse (HSEC). The synchronization events may not be required by the display (peripheral): They are optional. Users can program which sync events are generated to the display from the timings received from the display controller in video mode. When data are received on the L4 interconnect slave port, the synchronization codes are not automatically generated by the protocol engine. They can be provided on the L4 interconnect port by writing to the registers with limited timing control. It is highly recommended to use the video port from the display controller to receive the synchronization events to automatically generate the short synchronization packets to the peripheral. When the DSI protocol engine detects that the VSYNC signal from the display controller transitions from inactive to active state, the VSSC short packet replaces the following HSSC corresponding to the following HSYNC synchronization short packet (if the generation is enabled). When the transition from active to inactive state is detected, the VSEC short packet is generated (if the generation is enabled) replacing the HSSC synchronization packet corresponding to the following HSYNC. When the DSI protocol engine detects that the HSYNC signal from the display controller transition from inactive to active state, the HSSC short packet is generated (if the generation is enabled). When the transition from active to inactive state is detected, the HSEC short packet is generated (if the generation is enabled). For the first frame, any HSYNC and data received on the video port prior to the first VSYNC should be ignored. Since the first VSYNC sent to the display is also recognized as a HSYNC for the first line, there should be no HSYNC sent for the first line. To send the synchronization codes, the DSI protocol engine uses the short packets. Table 15-14 summarizes the 6-bit DT synchronization code values.

Table 15-14. Synchronization Codes

Synchronization Code	Value	Comments
V sync start code (VSSC)	0x1	Optional
V sync end code (VSEC)	0x11	Optional
H sync start code (HSSC)	0x21	Optional
H sync end code (HSEC)	0x31	Optional

15.2.2.4.8 Blanking

To keep the DSI link in HS state while using the video mode, during blanking periods, the long blanking packets are sent to the display. The DSS.DSI_VM_TIMINGi (I between 1 and 7) registers define the size of the long blanking packets after:

- Horizontal sync start code (short packet)
- Horizontal sync end code (short packet)
- Vertical sync start code (short packet)
- Vertical sync end code (short packet)
- Pixels (long packet)

Table 15-15 defines the short packet values for the synchronization packets:

Table 15-15. Sync Short Packet Values

Virtual Channel ID	Sync Code	Header (1st Byte)	Header (2nd Byte): Data Field LSB	Header (3rd Byte): Data Field MSB	Header (ECC)
0x0	0x1	0x1	0x0	0x0	See note following this table
	0x11	0x11			
	0x21	0x21			
	0x31	0x31			
0x1	0x1	0x41			
	0x11	0x51			
	0x21	0x61			
	0x31	0x91			
0x2	0x1	0x81			
	0x11	0x81			
	0x21	0xA1			
	0x31	0xB1			
0x3	0x1	0xC1			
	0x11	0xD1			
	0x21	0xC1			
	0x31	0xF1			

Notes:

- If the ECC is enabled by setting the DSS.DSI_VCn_CTRL[8] ECC_TX_EN bit to 1 for the VC in video mode, the ECC value is calculated; otherwise, 0x00 is used for the blanking long packets and sync short packets. If the CRC is enabled by setting the DSS.DSI_VCn_CTRL[7] CS_TX_EN bit to 1 for the VC in video mode, the check-sum value is calculated; otherwise, 0x00 is used for the blanking long packets.
- In other cases, when the DSS.DSI_VCn_CTRL[7] CS_TX_EN bit is set to 0, the value 0x00 is always used for the CRC (long packets). When the DSS.DSI_VCn_CTRL[8] ECC_TX_EN bit is set to 0, the value 0x00 is used for the ECC for short and long packets, except when the header is provided by the register, since the ECC field is available in the register. It can be used to generate invalid ECC values when the header is provided by the register.

The link [lane(s) and clock separately] can be put in ULPS mode. While using the blanking values formerly defined, the packets (short and long) are considered in HS mode.

Timing parameters VSA, VBP, VFP, HSA, HBP, HFP, VACT, and t_L are defined in the DSS.DSI_VM_TIMINGx (x between 1 and 7) register. HSA, HBP, HFP, and t_L are defined using the byte clock unit (TxByteClkHS) and also in low-power clock cycles (TxClkEsc). VSA, VBP, VFP, and VACT are defined in term of number of lines. When the HS blanking packets are sent during the blanking periods, the parameters are used to determine the blanking packet payload size (taking into account the 4-byte header and the 2-byte check sum).

The configuration of the display controller timing generator must be used when the display controller timings are used to generate the DSI HS video mode transfer.

Special care must be taken in the case of the last line of the frame. The LPS transition is required when the link is in HS mode for the whole frame.

When BTA is sent for the data packets, the following blanking period can not be used for sending any data from the TX FIFO. When the blanking period starts with one HS packet from one VC, it can only be followed by another HS packet from the same VC, or by trigger (BTA for example). When there is no more HS data to send for this VC, the lane is in LPS. When the blanking period starts with one LP packet from one VC, it can only be followed by another LP packet from the same VC, by another VC, by trigger (BTA

for example), or by extra LP NULL packets. If the trigger has been sent, it is not possible to send any more data. When there is no more data from the TX FIFO to send in LP mode or the trigger has been sent, the lane is put into LPS. If the lanes must be kept in HS mode during blanking periods (except for the last blanking period of the frame), the HS blanking packets must be used. In case one trigger is sent at the beginning of the blanking period, the rest of the blanking period is in ULPS.

Figure 15-49 and Figure 15-50 show a nonburst transfer in DSI video mode with, and without VE and HE, respectively.

Figure 15-49. DSI Video Mode: Nonburst Transfer With VE and HE

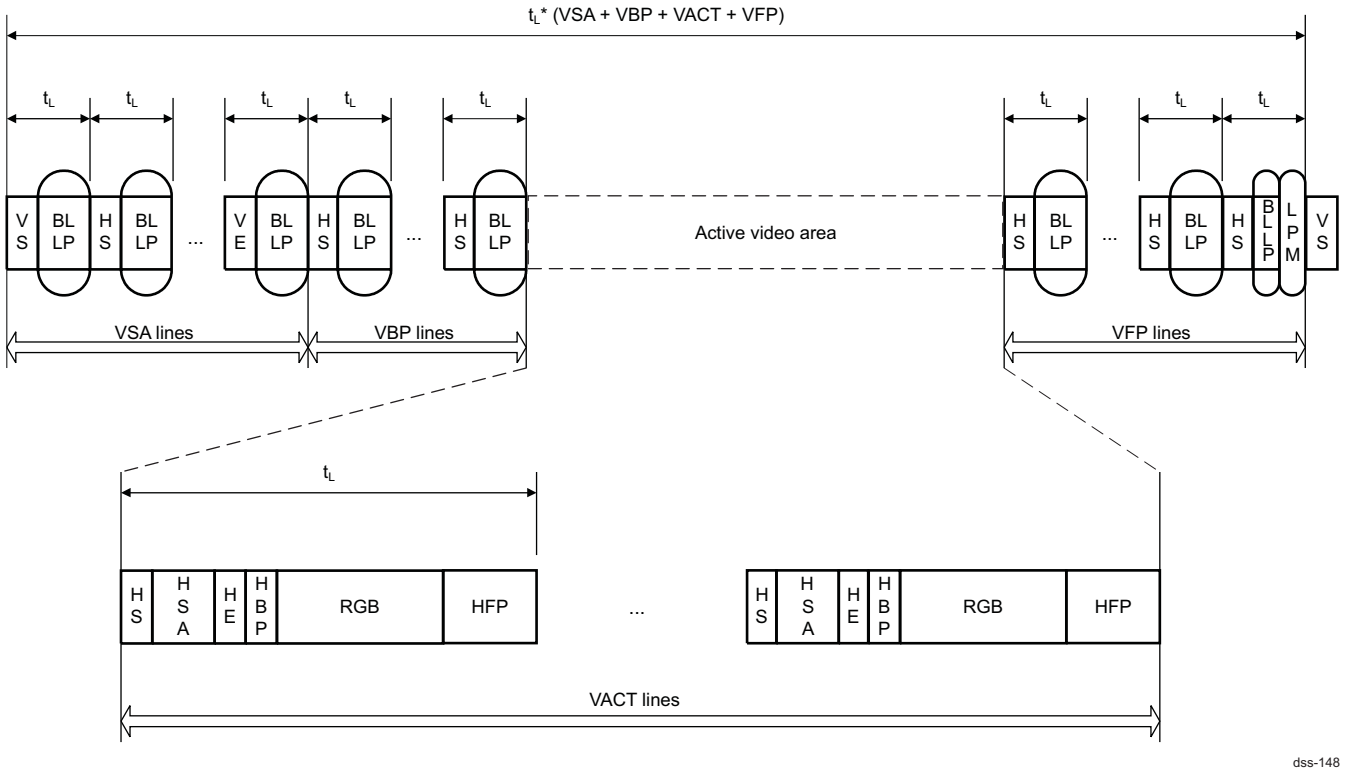
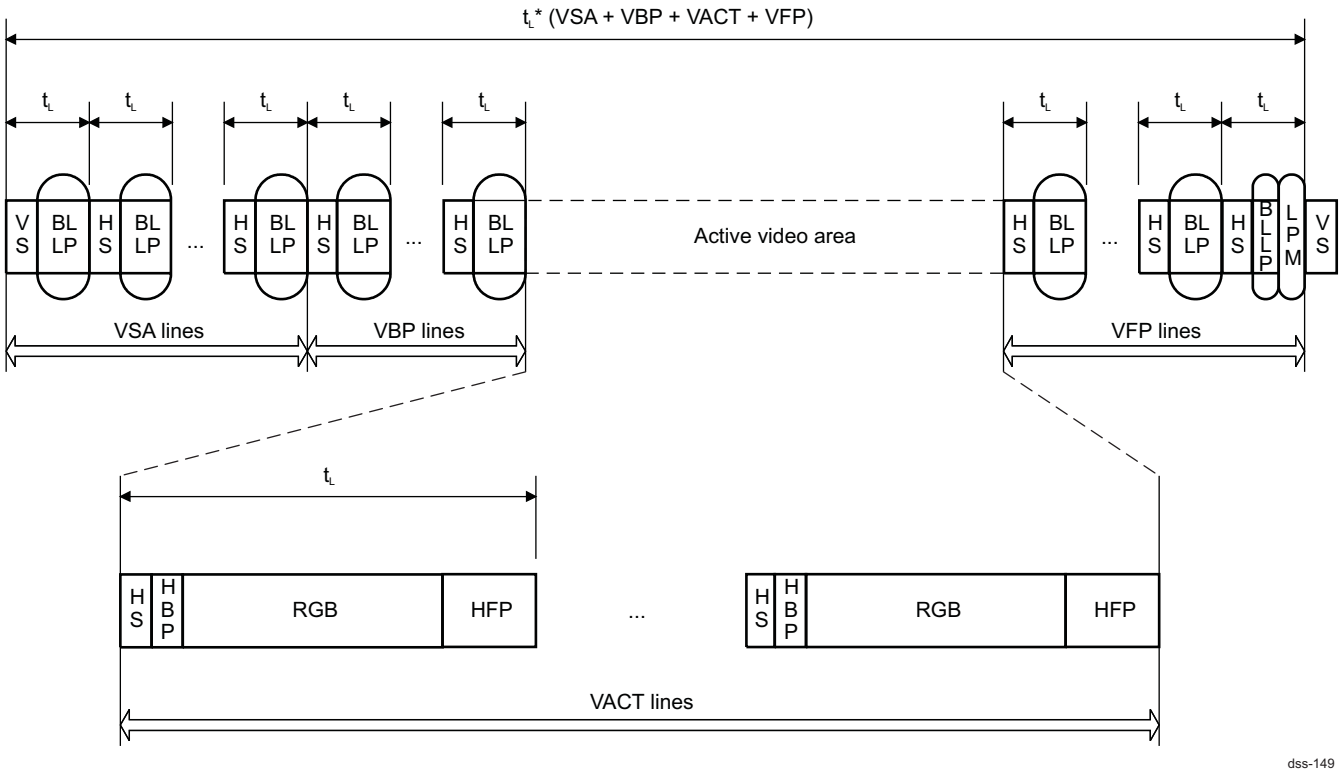
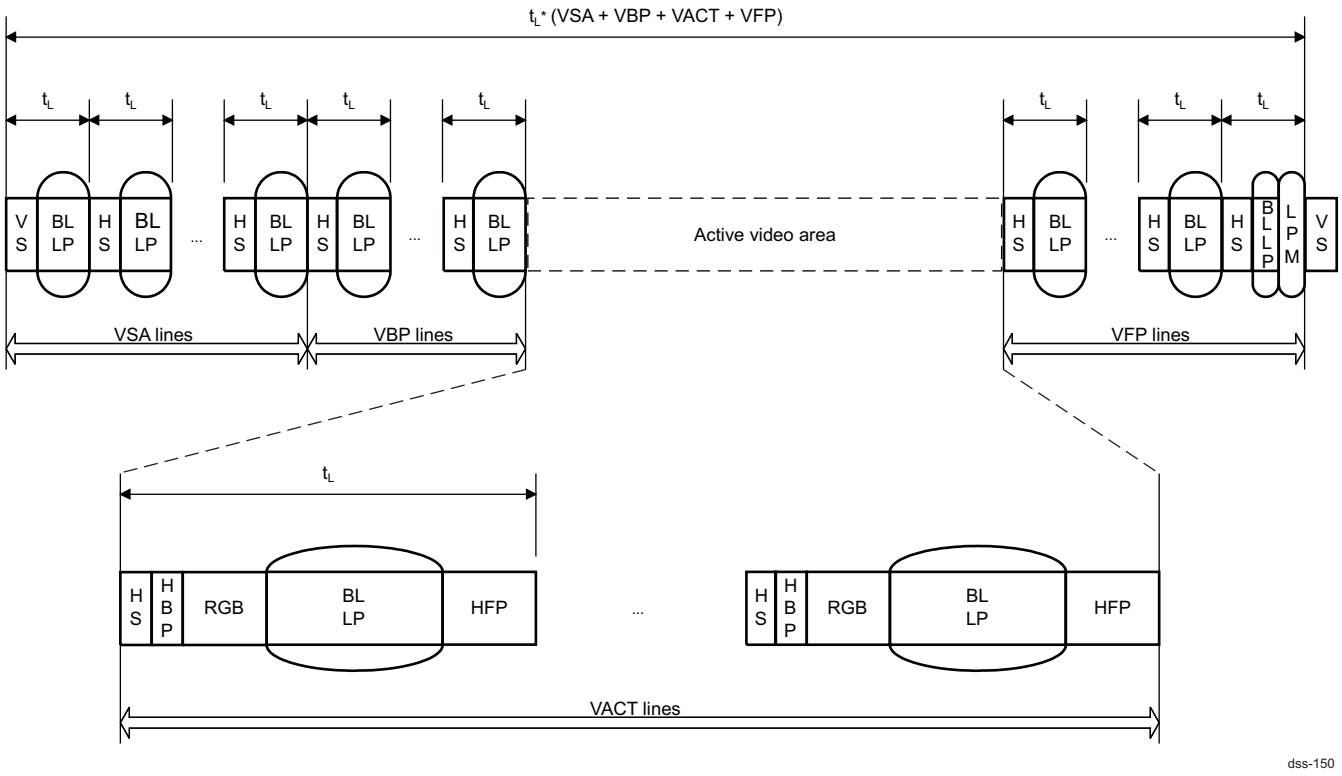


Figure 15-50. DSI Video Mode: Nonburst Transfer Without VE and HE



Note: HSA timing is not used and does not have to be programmed when HE short packet is not generated.

Figure 15-51. DSI Video Mode: Burst Transfer Without VE and HE



Note: HSA timing is not used and does not have to be programmed when HE short packet is not generated.

In Figure 15-50 and Figure 15-51, if HSync end short packet is not generated (HSA does not exist), HBP should be other than 0.

15.2.2.4.9 Frame Structures

Figure 15-52 shows the general DSI frame structure.

Figure 15-52. DSI General Frame Structure

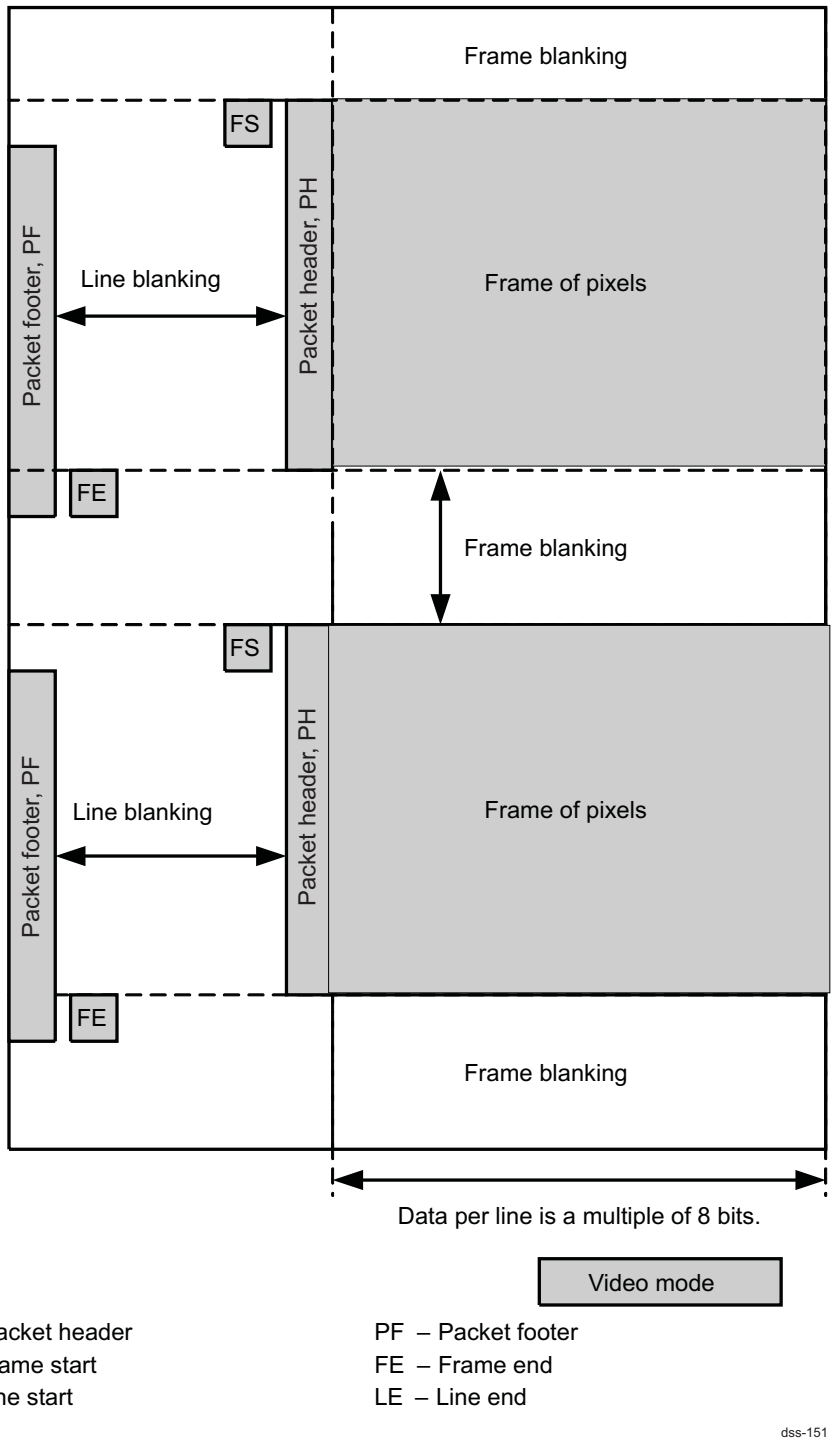


Figure 15-53 shows the general frame structure using burst mode.

Figure 15-53. DSI General Frame Structure Using Burst Mode

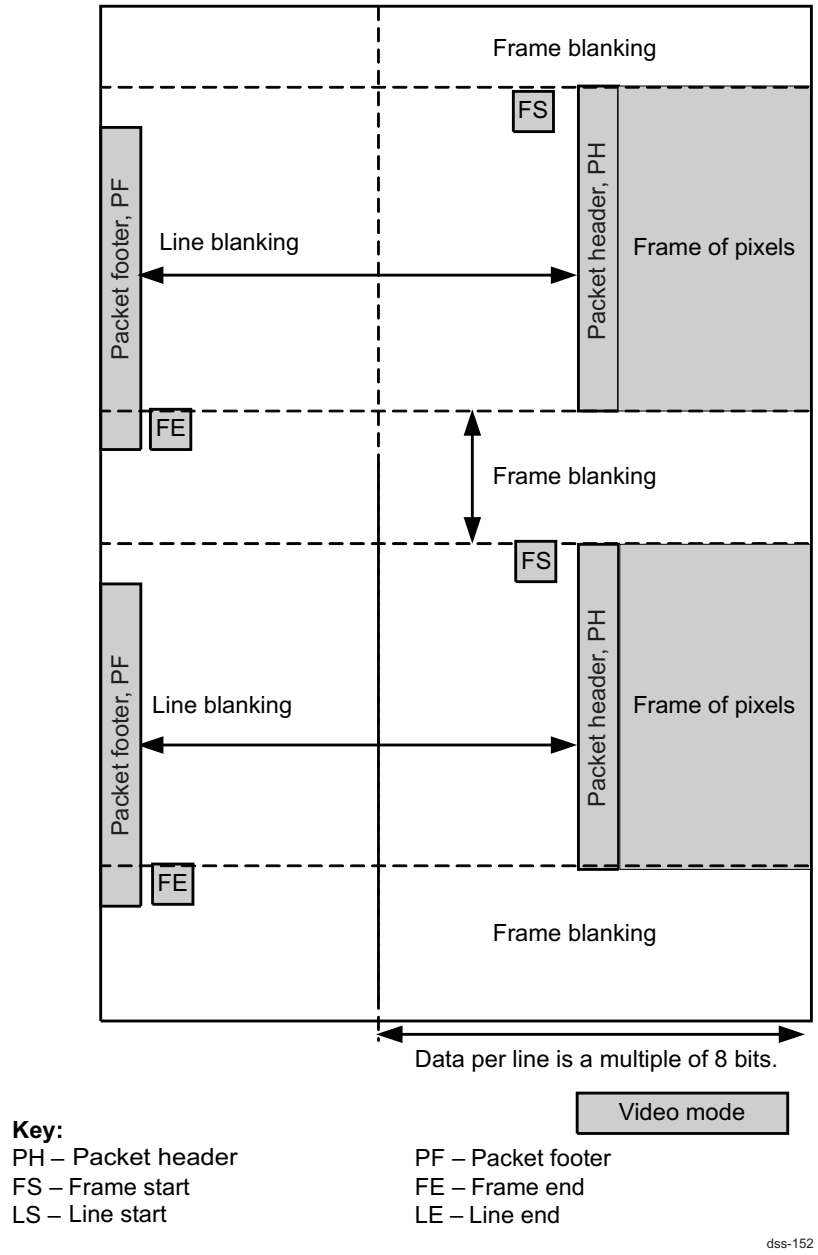
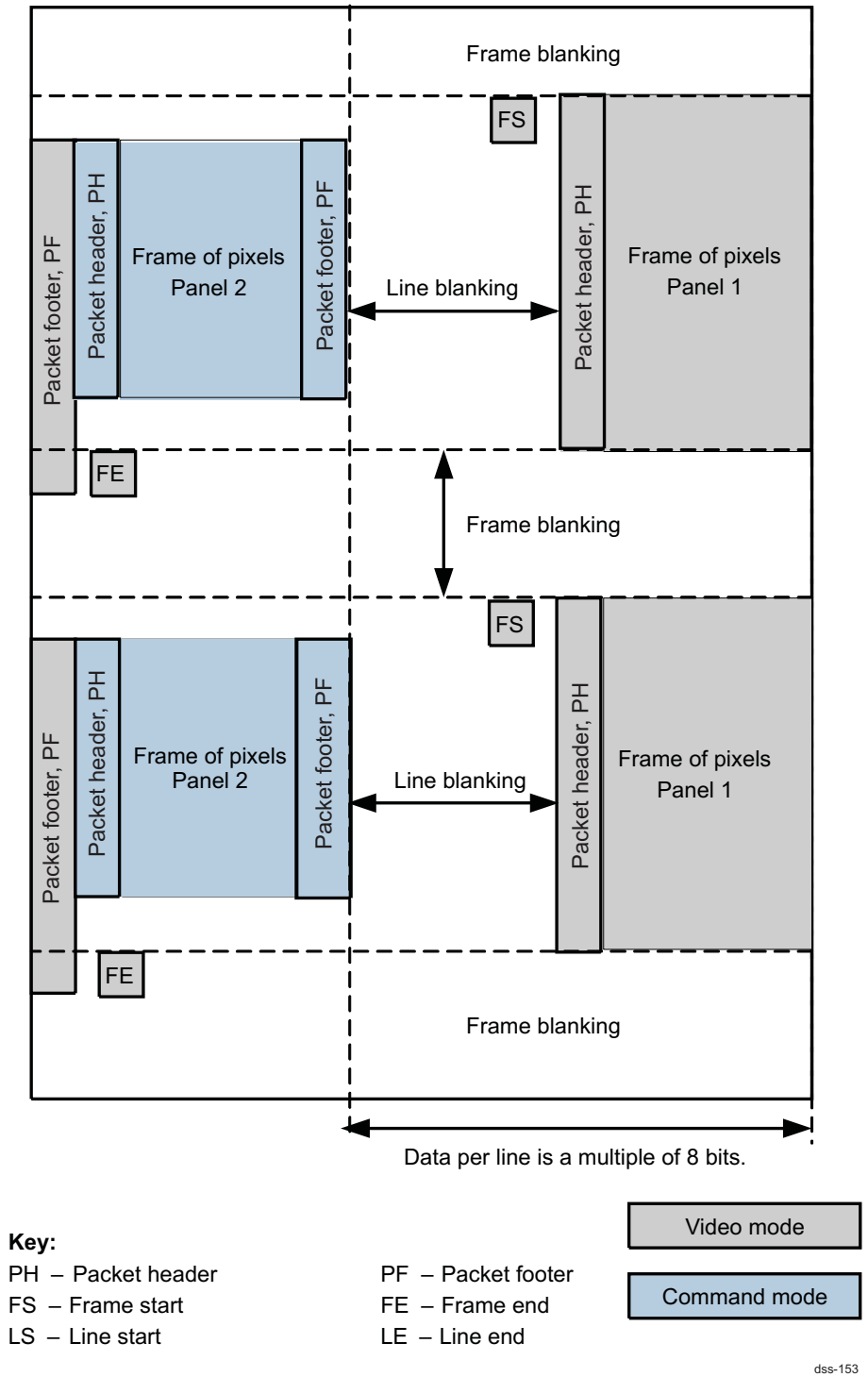


Figure 15-54 shows the general frame structure using burst mode and interleaving.

Figure 15-54. DSi General Frame Structure Using Burst Mode and Interleaving



15.2.2.4.10 Virtual Channels

The DSI protocol layer transports VCs. The purpose of VCs is to separate different data flows, which are interleaved in a same data stream. Each VC is identified by a unique channel identification number in the packet header. The channel identification number is encoded in 2-bits. The DSI protocol engine determines the channel identifier number to be used for generating the packet header and multiplexes the interleaved data streams. The DSI protocol engine supports multiple concurrent VCs: Up to 4. Table 15-16 summarizes the VC values used for each channel.

Table 15-16. Virtual Channel Values

Virtual Channel Number	Value
Virtual channel 0	0x0
Virtual channel 1	0x1
Virtual channel 2	0x2
Virtual channel 3	0x3

In the case of multiple displays connected to the single DSI port on the host, a hub may be used to root the data stream to the appropriate display based on the VC ID. Typically, VC ID 0x0 is used for the primary display and 0x1 for the secondary. The hub may have its own VC ID to provide communication capability between the host and the hub.

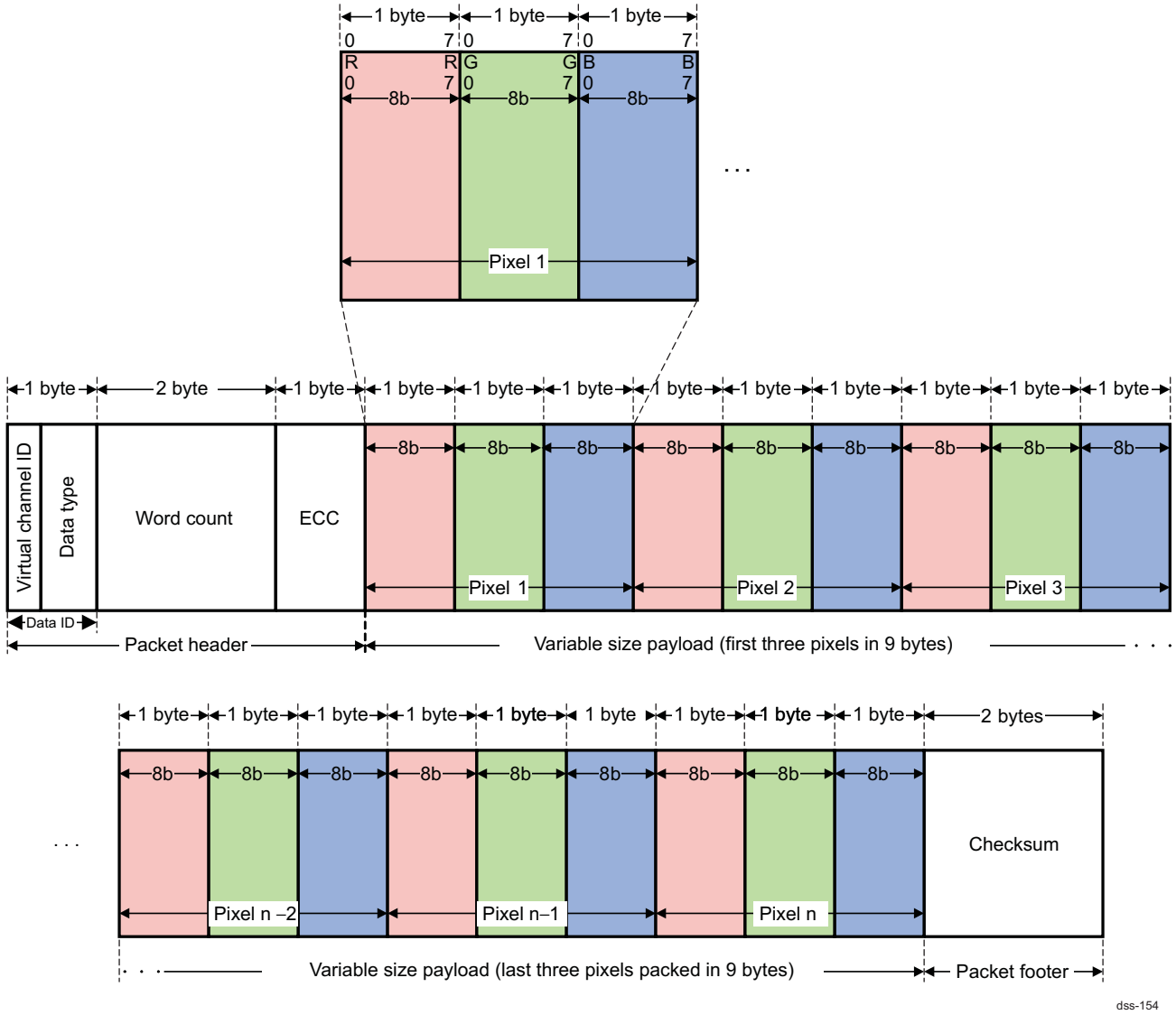
15.2.2.5 Pixel Data Formats

This section summarizes how the DSI supported pixel data formats in video mode are transmitted over the serial interface. For pixel formats in command mode. The DSI protocol engine can cope with all data formats given that the data line length sent through the DSI physical protocol is a multiple of a pixel. This condition is required for the DSI protocol engine to work properly.

15.2.2.5.1 24 Bits per Pixel - RGB Color Format, Long Packet

Figure 15-55 shows the RGB888 format.

Figure 15-55. 24 Bits per Pixel RGB Color Format, Long Packet

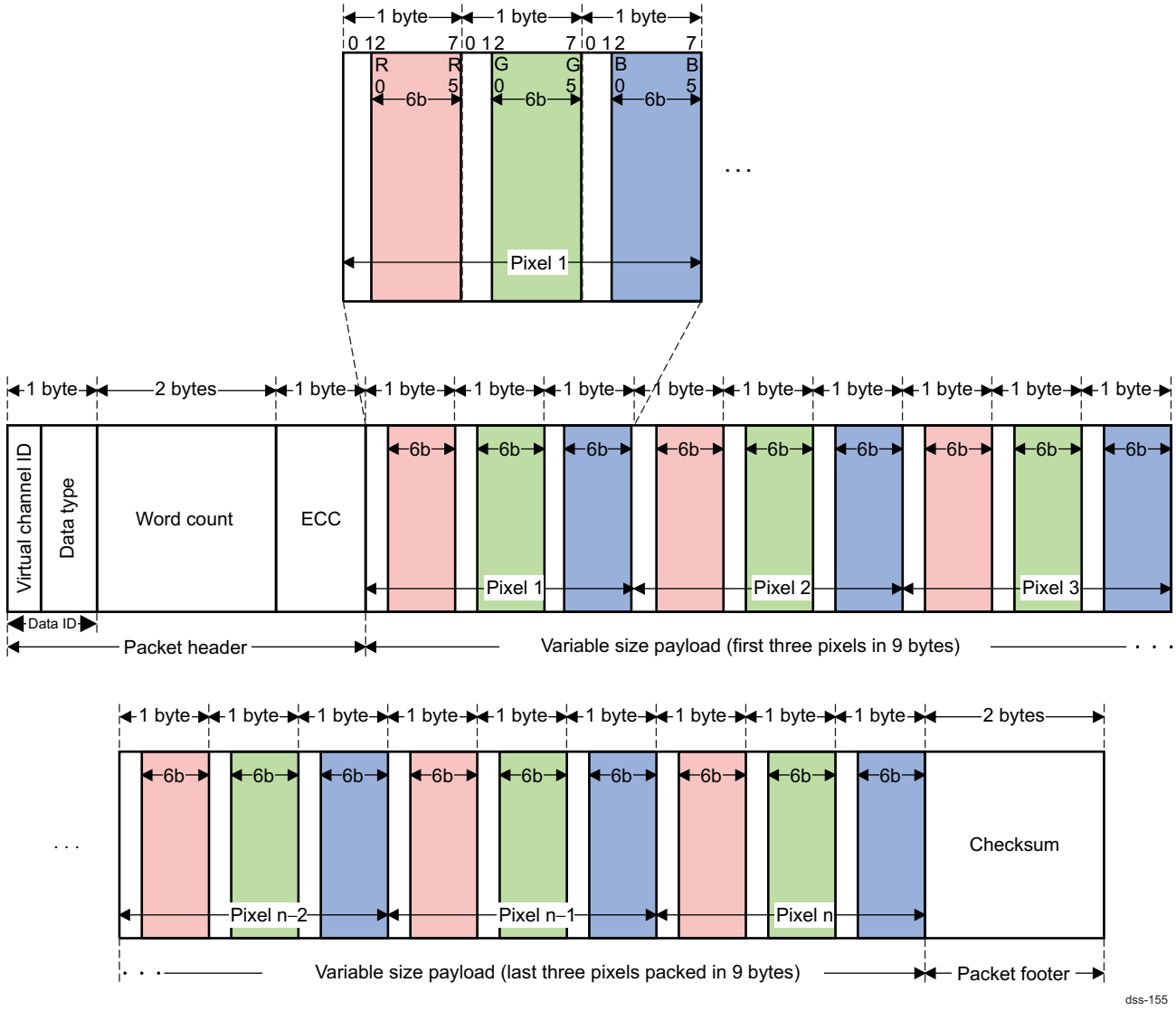


Packed pixel stream 24-bit format is a long packet used to transmit image data formatted as 24-bit pixels to a video mode display module. The packet consists of the DI byte, a two-byte WC, an ECC byte, a payload of length WC bytes, and a two-byte checksum. The pixel format is red (8 bits), green (8 bits), and blue (8 bits), in that order. Each color component occupies one byte in the pixel stream; no components are split across byte boundaries. Within a color component, the LSB is sent first, the MSB last.

15.2.2.5.2 18 Bits per Pixel (Loosely Packed) - RGB Color Format, Long Packet

Figure 15-56 details the RGB666 format.

Figure 15-56. 18 Bits per Pixel (Loosely Packed) RGB Color Format, Long Packet

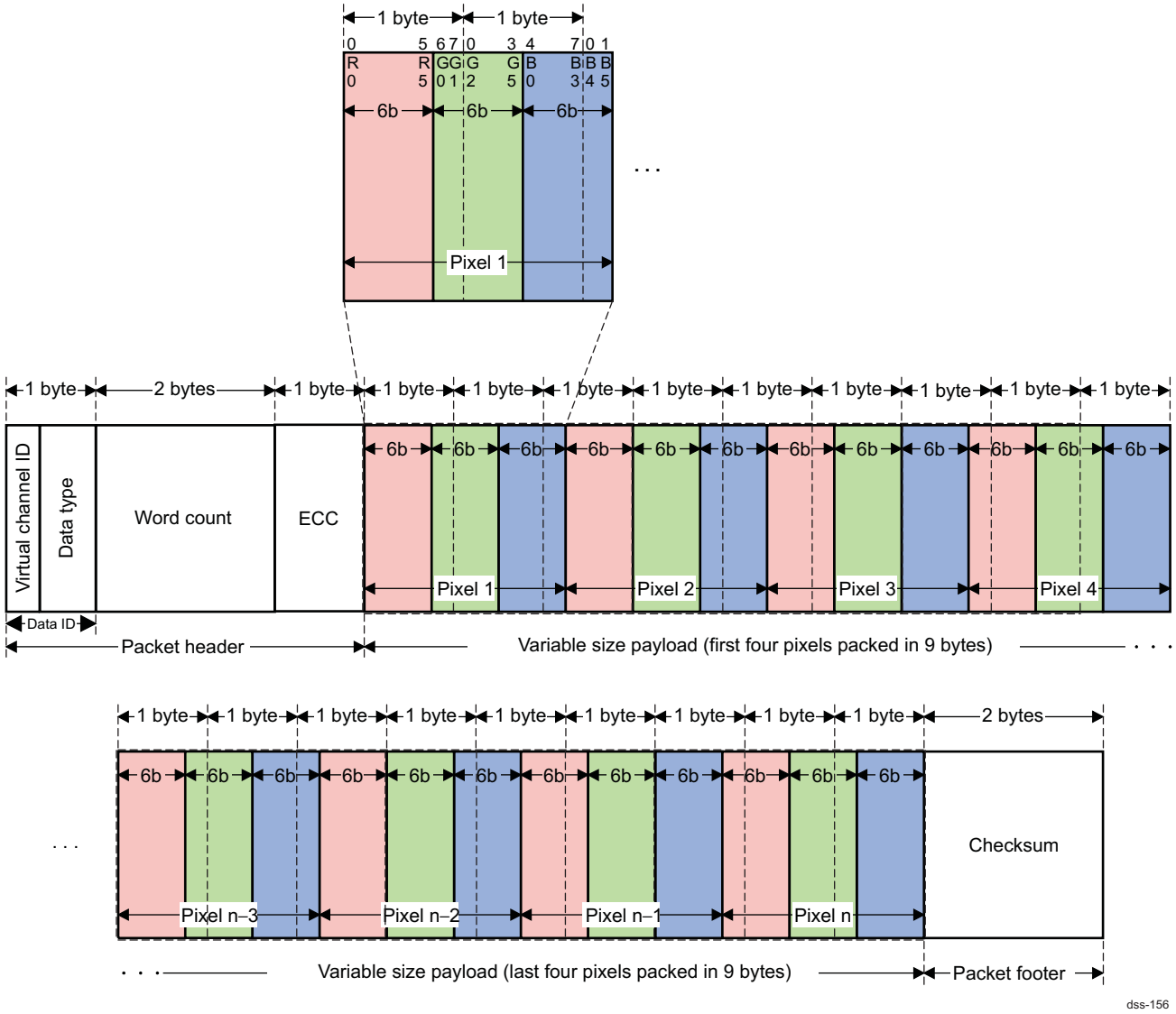


In the 18-bit pixel loosely-packed format, each R, G, or B color component is six bits but is shifted to the upper bits of the byte, such that the valid pixel bits occupy bits [7:2] of each byte. Bits [1:0] of each payload byte representing active pixels are ignored. As a result, each pixel requires three bytes as it is transmitted across the link. This requires more bandwidth than the packed format, but requires less shifting and multiplexing logic in the packing and unpacking functions on each end of the link. This format is used to transmit RGB image data formatted as pixels to a video mode display module that displays 18-bit pixels. The packet consists of the DI byte, a two-byte WC, an ECC byte, a payload of length WC bytes, and a two-byte checksum. The pixel format is red (6 bits), green (6 bits), and blue (6 bits) in that order. Within a color component, the LSB is sent first, the MSB last.

15.2.2.5.3 18 Bits per Pixel (Packed) - RGB Color Format, Long Packet

Figure 15-57 details the RGB666_PACKED format.

Figure 15-57. 18 Bits per Pixel (Packed) RGB Color Format, Long Packet

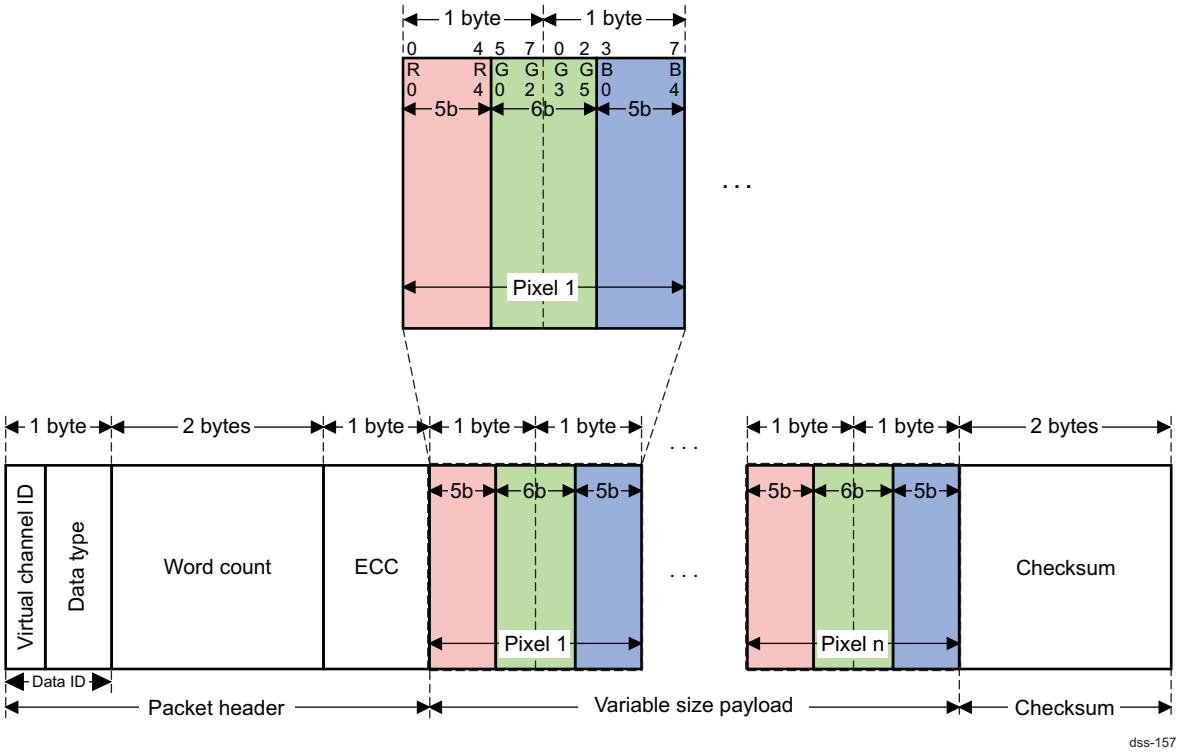


Packed pixel stream 18-bit format (packed) is a long packet. It is used to transmit RGB image data formatted as pixels to a video mode display module that displays 18-bit pixels The packet consists of the DI byte, a two-byte WC, an ECC byte, a payload of length WC bytes, and a two-byte checksum. Pixel format is red (6 bits), green (6 bits), and blue (6 bits), in that order. Within a color component, the LSB is sent first, the MSB last. With this format, it is strongly recommended that the total line width be a multiple of four pixels (nine bytes).

15.2.2.5.4 16 Bits per Pixel - RGB Color Format, Long Packet

Figure 15-58 details the RGB565 format.

Figure 15-58. 16 Bits per Pixel RGB Color Format, Long Packet



Packed pixel stream 16-bit format is a long packet used to transmit image data formatted as 16-bit pixels to a video mode display module. The packet consists of the DI byte, a two-byte WC, an ECC byte, a payload of length WC bytes, and a two-byte checksum. Pixel format is five bits red, six bits green, five bits blue, in that order. Note that the *green* component is split across two bytes. Within a color component, the LSB is sent first, the MSB last.

15.2.3 LCD Output With TI FlatLink3G Data Format for the SDI Module

The parallel-to-serial conversion of the 24-BPP pixel data depends on the number of data channels in use.

Figure 15-59 through Figure 15-61 summarize the pixel order for each data format of the TI FlatLink3G LCD panel.

The following apply to Figure 15-59 through Figure 15-61:

- VS: Vertical synchronization
- HS: Horizontal synchronizations
- DE: Data enable
- CP: Pixel parity. Parity of the whole 30-bit data is odd
- F1, F0: Reserved bits for the TI FlatLink3G protocol
- Clock channel: Differential clock signal between SDI_CLKP and SDI_CLKN

Figure 15-59. 24 Bits Per Pixel With One Data Channel

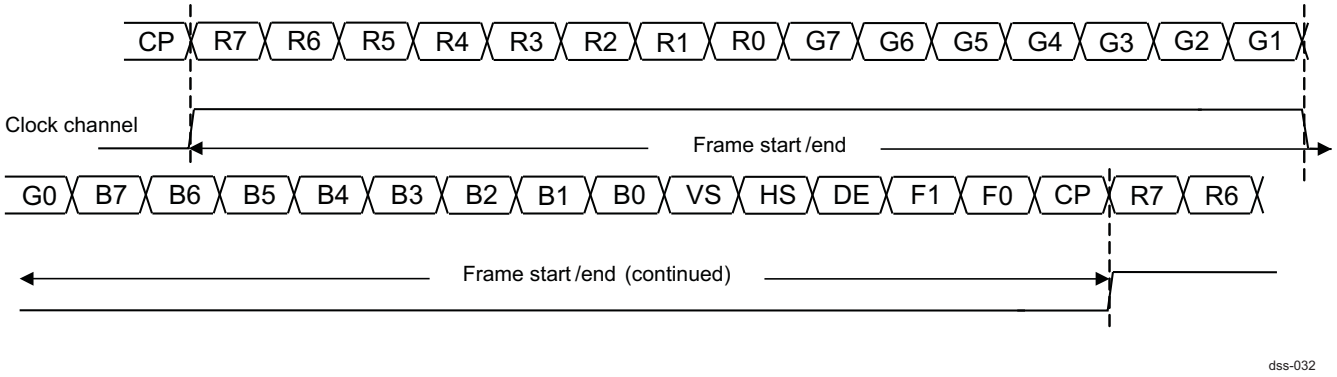


Figure 15-60. 24 Bits Per Pixel With Two Data Channels

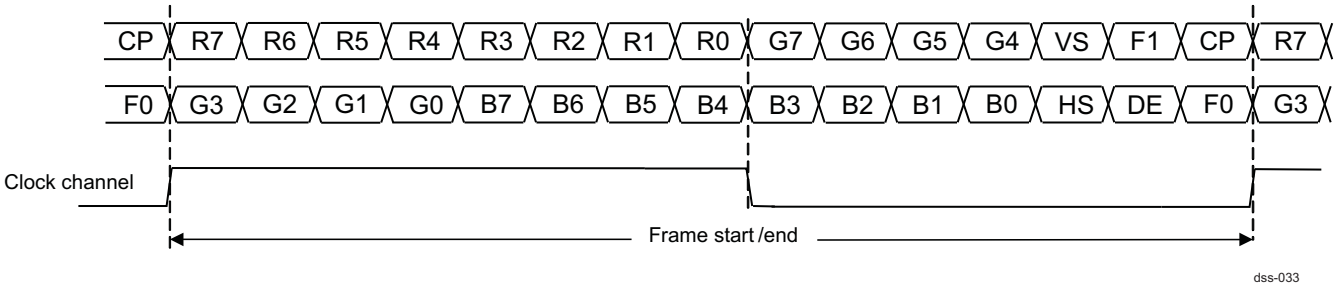
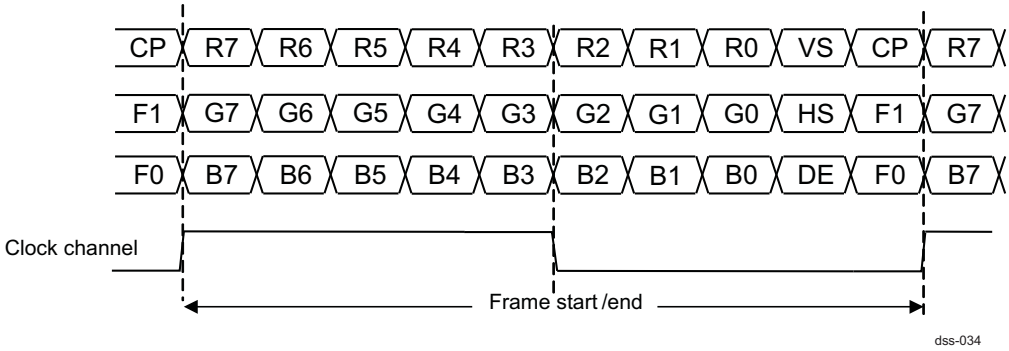


Figure 15-61. 24 Bits Per Pixel With Three Data Channels



Note: The range of the data rates permitted is 120 Mbps per pair minimum (4 MHz pixel clock, one data pair) to 650 Mbps per pair maximum (65 MHz pixel clock, three data pairs).

15.2.4 TV Display Support

The TV display path includes the following modules:

- Display controller
- Video encoder
- Dual 10-bit DAC with video amplifiers

The display controller module receives synchronization signals from the video encoder and synchronously sends pixel data to the video encoder with these signals. The digital output of the display controller is always a 24-bit RGB value based on a pixel request from the video encoder.

The video encoder converts RGB video signals to conform to the NTSC/PAL standard analog video. The video encoder includes an integrated synchronization signal generator and a 2-channel video digital-to-analog converter (DAC) with video amplifiers, data manager, luma stage, chroma stage, modulator, and a control interface.

The video encoder also provides the synchronization signals to the display controller: VSYNC, active VIDeo (AVID), and field ID (FID).

Figure 15-62 is a block diagram of the TV display interface (s-video mode).

Figure 15-62. TV Display Interface (s-video mode)

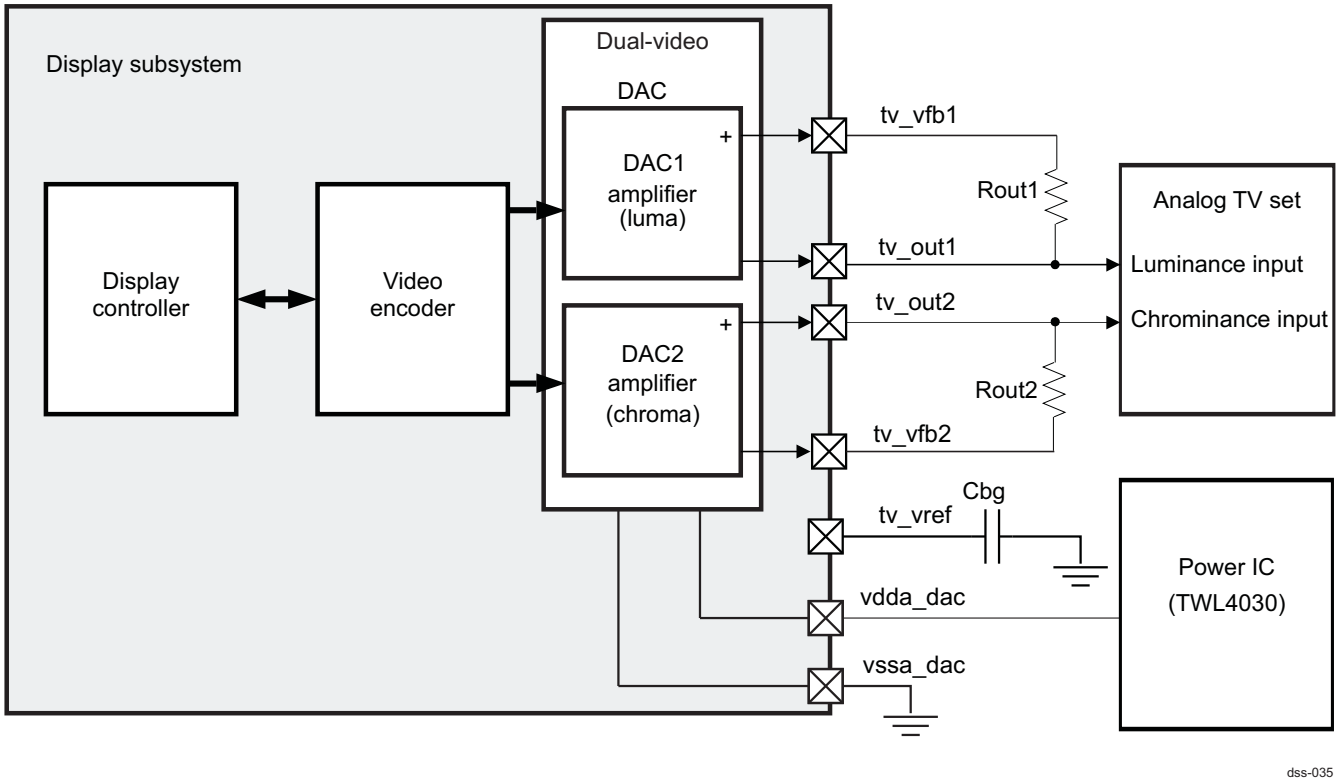
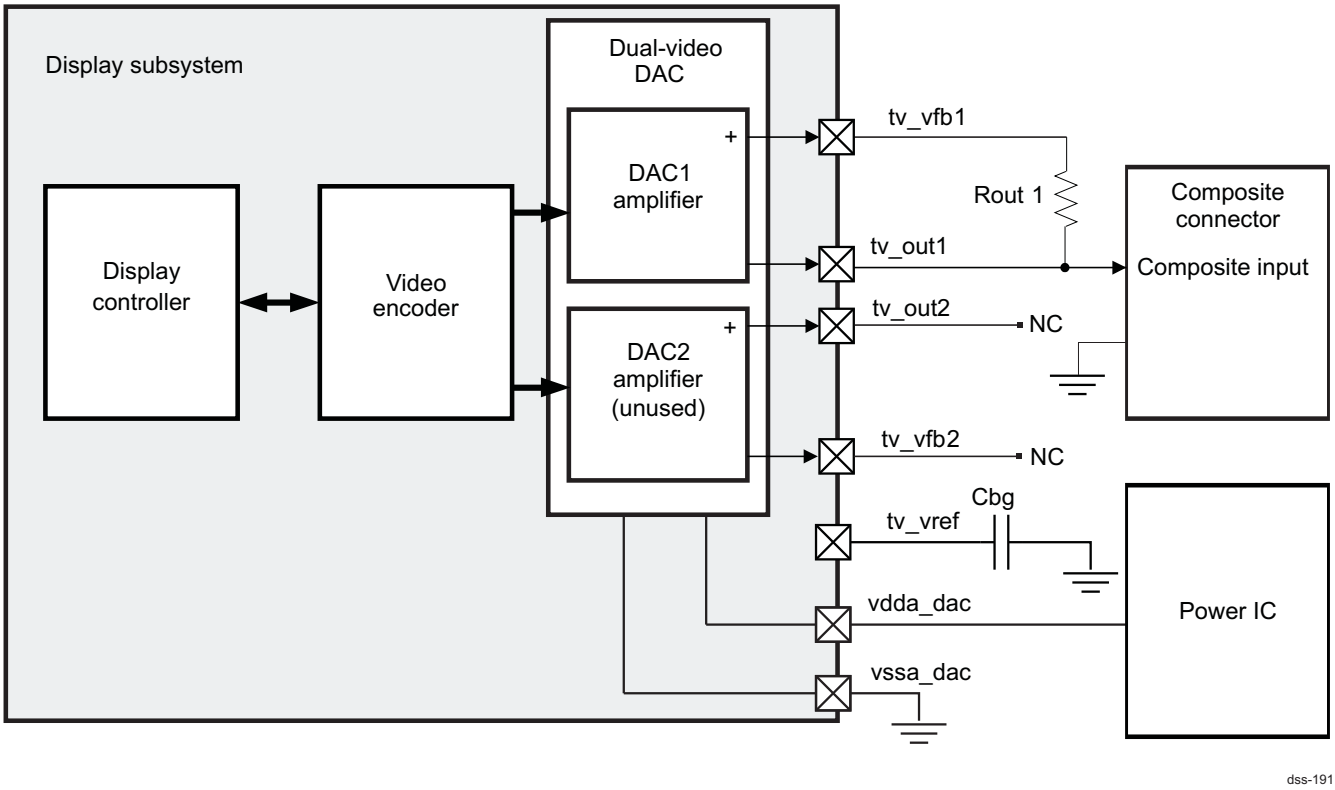


Figure 15-63 is a block diagram of the TV display interface (composite mode).

Figure 15-63. TV Display Interface (Composite Mode)



Note: In composite video mode, the video DAC2 chroma output must be disabled by setting the DSS.VENC_OUTPUT_CONTROL[2] CHROMA_ENABLE bit to 0.

Table 15-17 describes the interface signals to/from the TV set for TV display support.

Table 15-17. TV Display Interface Pins

Pin Name	Type[1]	Description
tv_out1	O	Analog luma or composite video output. An external resistor is connected between this node and the tv_vfb1 pin. The nominal value of Rout1 is 1650 Ω. Note that this is the output node that drives the load (75 Ω).
tv_out2	O	Analog chroma video output. An external resistor is connected between this node and the tv_vfb2 pin. The nominal value of Rout2 is 1650 Ω. This is the output node that drives the load (75 Ω).
tv_vfb1	O	Amplifier feedback node. An external resistor is connected between this node and tv_out1. The nominal value of Rout1 is 1650 Ω.
tv_vfb2	O	Amplifier feedback node. An external resistor is connected between this node and tv_out2. The nominal value of Rout2 is 1650 Ω.
tv_vref	O	Reference output voltage from internal bandgap. A decoupling capacitor must be connected for optimum performance. tv_vref is generated internally to the OMAP device.
vdda_dac	Power	Analog supply voltage for the dual video DAC
vssa_dac	Power	Analog ground for the dual video DAC

(1) O = Output, Power = Power pin

CAUTION

- tv_out1 and tv_out2 are very high-frequency analog signals and must be routed with extreme care. As a result, the path of these signals must be as short as possible, and as isolated as possible from other interfering signals.
- During board design, the onboard traces and parasites must be matched for the two channels. tv_vfb1 and tv_vfb2 pins are the most sensitive pins in the TV out system. The onboard parasitic capacitance associated with these two pins should be less than 0.5 pF. Low onboard resistance is required for the traces that connect the Rout1/Rout2 to the tv_vfb1/tv_vfb2 and TV OUT pins (tv_out1 and tv_out2). The resistance on those trace affects output impedance matching. Therefore, Rout1 and Rout2 resistors are suggested to be placed as close as possible to the OMAP device pins. The onboard traces lead to the TV OUT pins must have a characteristic impedance of 75 Ω starting from the closest possible place to the OMAP device pin output. For typical values of Rout1, Rout2, Cout, and Cbg, see the device data manual.
- If the TV output is not used, the analog pins tv_ref and vdda_dac must be grounded. The tv_out1, tv_out2, tv_vfb1, and tv_vfb2 pins can be grounded or left unconnected. To avoid current leakage, the following bits must be set to 0:
  - DSS.DSS_CONTROL[5] DAC_POWERDN_BGZ
  - DSS.VENC_OUTPUT_CONTROL[2:0]
  - PRCM.CM_FCLKEN_DSS[2] EN_TV
  - CONTROL.CONTROL_DEVCONF[18] TVOUTBYPASS

Texas Instruments provides a global solution with an OMAP device associated with a TWL4030 power IC. The power pin vdda_dac is software controlled by the power IC.

15.2.4.1 TV Output and Data Format

The output data to the TV set are the analog composite data from the DAC. The following video standards are supported:

- NTSC-J, M
- PAL-B, D, G, H, I
- PAL-M

15.2.4.2 Digital-to-Analog Converter

The DAC includes the following main features:

- 1.0-V-to-1.3-V digital power supply, 1.8-V analog power supply
- 10-bit resolution
- DNL within 1 LSB and INL within 1 LSB
- Sample rate of up to 60 megasamples per second (MSPS)
- Support composite/S-video dc or ac coupled output
- Full-scale voltage output: 0.88 Vpp with a 75-Ω load
- Internal TV detect feature
- Signal-to-noise ratio (SNR) is 54 dB (taking into account the ac coupling)
- Suitable for low-power wireless applications; total power: 10 mA (no load in dc-coupled mode)
- Power-down mode with less than 2.5-µA standby current
- Differential gain error and differential phase error: 1.5% and 1%, respectively

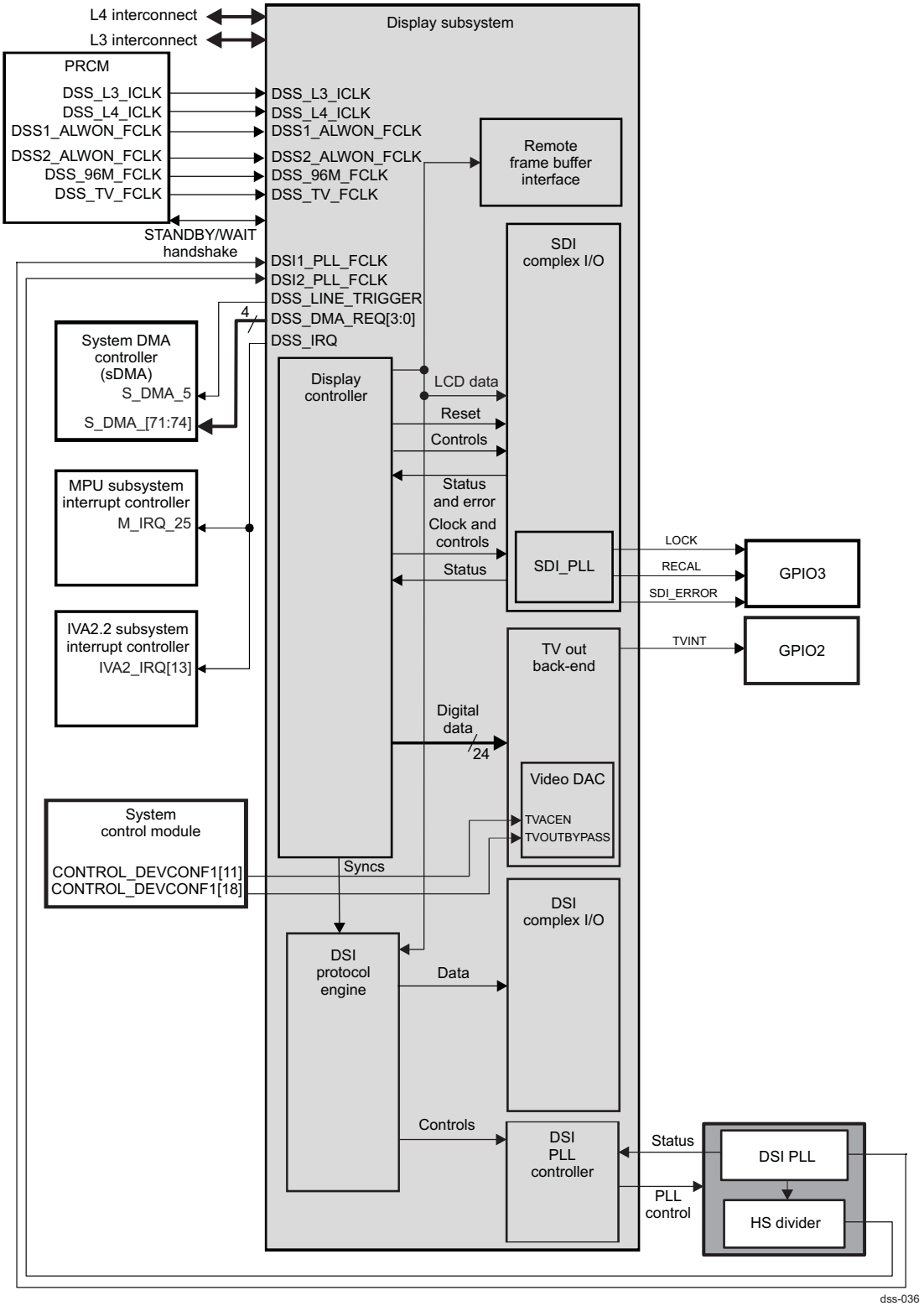
Note: To enhance the TV color display, it is highly recommended to set the DSS.DSS_CONTROL[4] DAC_DEMEN bit.

15.3 Display Subsystem Integration

This section describes the integration of the display subsystem and details clocks, resets, hardware requests, and power modes.

Figure 15-64 shows the integration of the display subsystem in the device.

Figure 15-64. Display Subsystem Integration



15.3.1 Clocking, Reset, and Power-Management Scheme

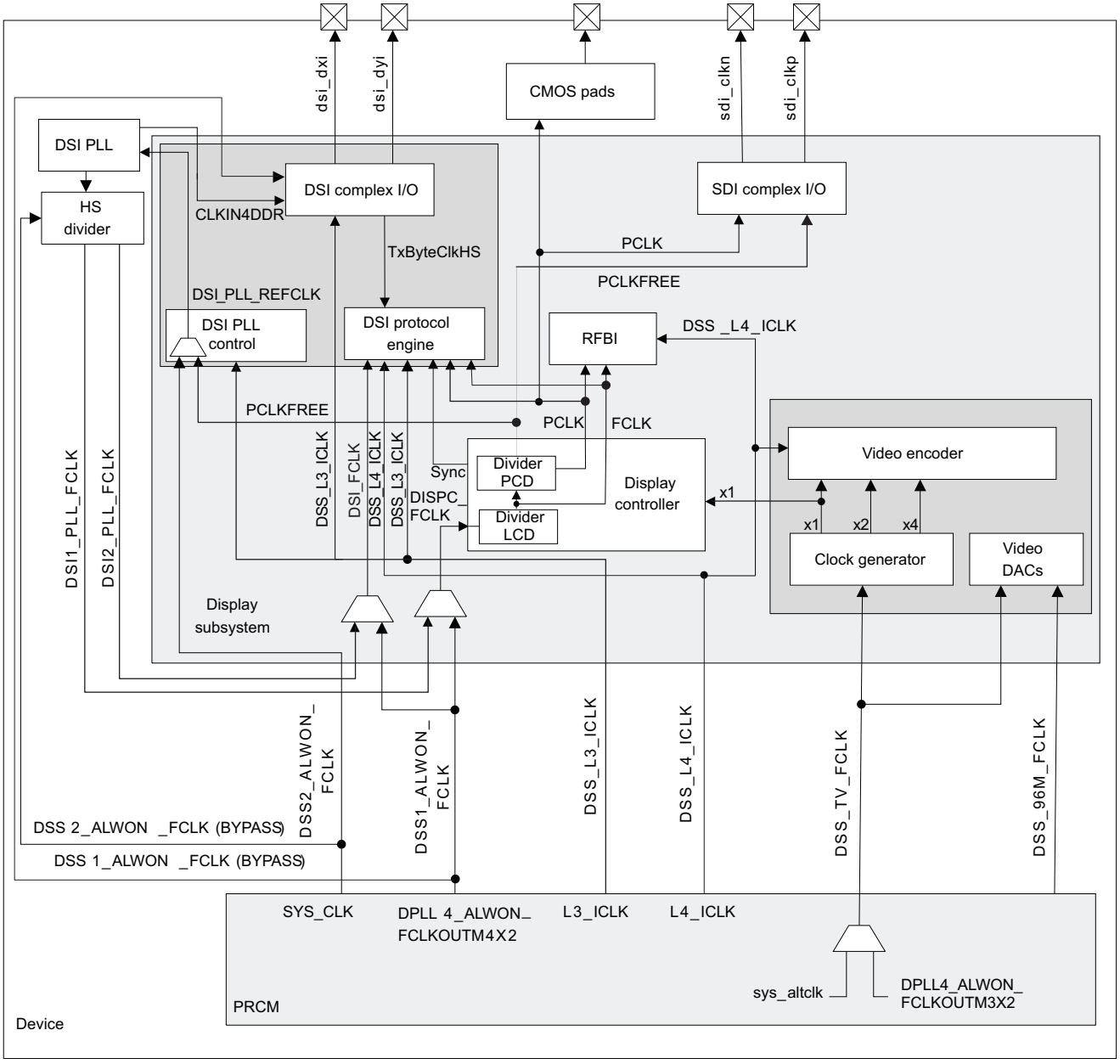
15.3.1.1 Clocks

The power, reset, and clock management (PRCM) module provides six clock signals to the display subsystem: The L3 interface clock (DSS_L3_ICLK) and the L4 interface clock (DSS_L4_ICLK), with frequencies equal to the L3 interconnect clock and the L4 interconnect clock, respectively; two configurable functional clocks (DSS1_ALWON_FCLK and DSS2_ALWON_FCLK); and two other functional clocks (DSS_TV_FCLK and DSS_96M_FCLK).

The DSI PLL provides two functional clock signals to the display subsystem: DSI1_PLL_FCLK and DSI2_PLL_FCLK.

Figure 15-65 details the clock tree for the display subsystem.

Figure 15-65. Display Subsystem Clock Tree



Note: A synchronization signal is sent by the display controller (DISPC) to the DSI protocol engine. This signal, named DISPC_UPDATE_SYNC, is used to inform the DSI protocol engine that it must be unsynchronized with the display controller.

Table 15-18 describes the different clocks with their possible frequency values.

Table 15-18. Display Subsystem Clocks

Clock Signal	Attribute	Module	Frequency	Comment
DSS_L3_ICLK	L3 interface clock	DSS	(See Chapter 4, *PRCM*)	L3 interface clock from PRCM
DSS_L4_ICLK	L4 interface clock	DSS	(See Chapter 4, *PRCM*)	L4 interface clock from PRCM
DSS1_ALWON_FCLK	Functional clock	DISPC, DSI protocol engine	Up to 173 MHz at nominal voltage	From PRCM: DPLL4 (source: DPLL4_ALWON_FCLK)
DSS2_ALWON_FCLK	Functional clock	DSI PLL	12/13/16.8/19.2/26/38.4 MHz	From PRCM: SYS_CLK
DSI1_PLL_FCLK	Functional clock	DISPC	Up to 173 MHz at nominal voltage	From DSI PLL and HS divider
DSI2_PLL_FCLK	Functional clock	DSI protocol engine	Up to 173 MHz at nominal voltage	From DSI PLL and HS divider
DSS_TV_FCLK	Functional clock	DSS, video mode DAC	54 MHz or	From PRCM: DPLL4 (source: DPLL4_ALWON_FCLK) or
			sys_alt_clk (up to 59 MHz)	External input clock (See Chapter 4, *PRCM*)
DSS_96M_FCLK	Functional clock	Video DAC	96 MHz	From PRCM: 96M_FCLK

To enable or disable each functional clock, set the following bit (1: Enable, 0: Disable):

- PRCM.CM_FCLKEN_DSS[0] EN_DSS1 bit to enable DSS1_ALWON_FCLK
- PRCM.CM_FCLKEN_DSS[1] EN_DSS2 bit to enable DSS2_ALWON_FCLK
- PRCM.CM_FCLKEN_DSS[2] EN_TV bit to enable DSS_TV_FCLK and DSS_96M_FCLK

To enable or disable the DSS_L3_ICLK and DSS_L4_ICLK interface clocks, write (1: Enable, 0: Disable) to the PRCM.CM_ICLKEN_DSS[0] EN_DSS bit.

Note: Note that it is not possible to gate/stop L3 clock and keep L4 clock running.

- L3 and L4 interface clock

  The DSS_L3_ICLK clock is only used by the display controller interface to fetch the pixel data. The DSS_L4_ICLK is used to access the L4 interconnect for configuring all the display subsystem registers.

  Note: A clock generated internally from the L3 interface clock allows the submodules to be configured and is the functional clock for the RFBI. The SDI module and all the display subsystem registers are configured through the display subsystem register.

- Display controller functional clocks

  The display controller can use either the DSS1_ALWON_FCLK or the DSI1_PLL_FCLK functional clock.

  To select the DSS1_ALWON_FCLK functional clock (the default clock selected after reset), write 0 in the DSS.DSS_CONTROL[0] DISPC_CLK_SWITCH bit; to select the DSI1_PLL_FCLK functional clock, write 1 in the DSS.DSS_CONTROL[0] DISPC_CLK_SWITCH bit.

Note: The DSS1_ALWON_FCLK and DSI1_PLL_FCLK functional clocks must be active (the PRCM.CM_FCLKEN_DSS[0] EN_DSS1 and DSI PLL programmed correctly) to switch from one functional clock to another. The new functional clock is effective when the next vertical blanking interval occurs. This is true only if the DSS.DISPC_CONTROL[5] GOLCD bit is set to 1.

Depending on the DPLL4 input clock frequency, the DSS1_ALWON_FCLK can be adjusted by setting the PRCM.CM_CLKSEL_DSS[4:0] CLKSEL_DSS1 bit field.

- DSI PLL functional clock (DSI_PLL_REFCLK)

  The DSI PLL controller module can use either the DSS2_ALWON_FCLK (from PRCM) or the PCLKFREE (from DISPC) functional clock. To select the DSS2_ALWON_FCLK functional clock (default clock selected after reset), write 0 in the DSS.DSI_PLL_CONFIGURATION2[11] DSI_PLL_CLKSEL bit; to select the PCLKFREE functional clock, write 1 in the DSS.DSI_PLL_CONFIGURATION2[11] DSI_PLL_CLKSEL bit.

- DSI protocol engine functional clocks (DSI_FCLK)

  The DSI protocol engine can use either the DSS1_ALWON_FCLK (from PRCM) or the DSI2_PLL_FCLK (from DSI PLL) functional clock. To select the DSS1_ALWON_FCLK functional clock (default clock selected after reset), write 0 in the DSS.DSS_CONTROL[1] DSI_CLK_SWITCH bit; to select the DSI2_PLL_FCLK functional clock, write 1 in the DSS.DSS_CONTROL[1] DSI_CLK_SWITCH bit.

Note: It is possible to switch between these two clocks, even when both of them are not active.

- There are five clock domains in the DSI module:
  - Byte clock domain:

    TxByteClkHS is generated from the bit clock and converted into a byte clock. The maximum frequency is 100 MHz (all OPPs). It is generated by the DSI complex I/O.
  - Functional clock domain

    The DSI_FCLK is the functional clock for the DSI protocol engine module. The maximum frequency is 173 MHz (nominal voltage) and 124 MHz (low voltage). It should always be equal to or higher than the byte (TxByteClkHS), L4 interconnect (DSS_L4_ICLK), and video port (VP_CLK) clocks. The software must configure the clocks correctly.
  - L4 interface clock domain

    The DSS_L4_ICLK is used in the L4 interconnect port domain. The maximum frequency is 166 MHz at nominal voltage and 83 MHz at low voltage.
  - The video port domain

    The pixel clock (PCLK) on the video port is used by the video port domain to capture the pixels from the display controller. The maximum frequency of VP_CLK used as the functional clock for the video port domain is 173 MHz at nominal voltage and 96 MHz at low voltage.
  - Serial Configuration Port (SCP) and Power Control (PWR) interfaces

    The DSS_L4_ICLK is the functional clock.

Notes:

- There is no clock domain for RxClkEsc because it is used as an enable and not as a clock by the DSI protocol engine module
- The clock domains are asynchronous (except for L4 interconnect port and SCP/PWR because both of them use DSS_L4_ICLK). The clocks used for the L4 interconnect port and SCP/PWR interface should be balanced.
- If video mode is used, the display controller functional clock must be generated using a clock from the DSI PLL.

- Video encoder functional clock

  The DSS_TV_FCLK is divided into three balanced clocks, depending on the clock mode selected (see Table 15-19).

Table 15-19. Possible Digital Clock Division for the Video Encoder

Clock Output	Clock Mode	
	Clock mode 0	Clock mode 1
Video encoder clock 4x	DSS_TV_FCLK	DSS_TV_FCLK or 0 (gated)
Video encoder clock 2x	DSS_TV_FCLK/2	DSS_TV_FCLK
Video encoder clock 1x	DSS_TV_FCLK/4	DSS_TV_FCLK/2

Note: Clock mode 1 can be used for power-saving purposes, or if a 27-MHz external clock is provided to the video encoder.

DSS_TV_FCLK can be adjusted depending on the DPLL4 input clock frequency by setting the PRCM.CM_CLKSEL_DSS[12:8] CLKSEL_TV bit field. If the DPLL4 is selected, the DSS_TV_FCLK is provided by the DPLL4_ALWON_FCLKOUTM3X2 clock.

Note: If the DSS_TV_FCLK is not provided by DPLL4 but rather by the sys_alt_clk pin, an external clock generator must be connected to this pin. In this case, a 54-MHz clock is needed for PAL or NTSC 601, a 49.09-MHz clock is needed for NTSC square pixel, and a 59-MHz clock is needed for PAL square pixel.

- Dual video DAC clocks

  The dual video DAC uses two distinct clocks: The DSS_TV_FCLK and the DSS_96M_FCLK. The video data are latched on the positive edge of the DSS_TV_FCLK clock. On the other hand, the video DAC uses the DSS_96M_FCLK fixed-frequency clock internally as the clock input for the switch capacitor resistor.
- SDI functional clock

  The display controller provides the functional clocks PCLK (pixel clock) and PCLKFREE (free-running pixel clock) of the SDI module, which is derived from the display controller functional clock. The SDI module functional clock drives the phase-locked loop SDI_PLL that multiplies and divides the PCLK frequency by two programmable factors for the serial connection.

Note: When using the SDI, configure the PCLK in the free-running mode by setting the DSS.DISPC_CONTROL[27] PCLKFREEENABLE bit to 1.

15.3.1.2 Resets

15.3.1.2.1 Hardware Reset

The display subsystem receives its reset signal DSS_RST (the reset signal of the display subsystem [DSS] power domain) from the PRCM module.

15.3.1.2.2 Software Reset

The display subsystem can receive a software reset propagated through all of the submodules and used to initialize the display subsystem. To apply the reset, write to the DSS.DSS_SYSCONFIG[1] SOFTRESET bit (1: Reset; 0: Normal). The DSS.DSS_SYSSTATUS[0] RESETDONE bit indicates that the software reset is complete when its value is 1.

Note: The display controller, the DSI protocol engine, and the RFBI modules also have their own software reset functionality. To access this reset, access the DSS.DISPC_SYSCONFIG[1] SOFTRESET bit for the display controller, the DSS.DSI_SYSCONFIG[1] SOFTRESET bit for the DSI protocol engine, and the DSS.RFBI_SYSCONFIG[1] SOFTRESET bit for the RFBI module.

To properly reset these modules, 0x2 is the only valid value to write to these registers.

CAUTION

All the interface and functional clocks, even for the TV output, must be provided to the display subsystem to update the RESETDONE status bit correctly.

15.3.1.3 Power Domain

The display subsystem modules are on the display subsystem (DSS) power domain and on the VDD2 voltage domain, except for the dual video DACs, which are on the analog vdda_dac voltage domain.

15.3.1.4 Power Management

15.3.1.4.1 Clock Activity Mode

The display controller clocks can be configured in one of the following clock activity modes:

- DSS.DISPC_SYSCONFIG[9:8] CLOCKACTIVITY bit field set to 0x0 (reset value): The interface and functional clocks can be switched off.
- DSS.DISPC_SYSCONFIG[9:8] CLOCKACTIVITY bit field set to 0x1: The functional clocks can be can be switched off and the interface clocks are maintained during the wake-up period.
- DSS.DISPC_SYSCONFIG[9:8] CLOCKACTIVITY bit field set to 0x2: The interface clocks can be can be switched off and the functional clocks are maintained during the wake-up period.
- DSS.DISPC_SYSCONFIG[9:8] CLOCKACTIVITY bit field set to 0x3: The interface and functional clocks are maintained during the wake-up period.

The DSI protocol engine clocks can be configured in one of the following clock activity modes:

- DSS.DSI_SYSCONFIG[9:8] CLOCKACTIVITY bit field set to 0x0 (reset value): The interface and functional clocks can be switched off.
- DSS.DSI_SYSCONFIG[9:8] CLOCKACTIVITY bit field set to 0x1: The functional clocks can be switched off and the interface clocks are maintained during the wake-up period.
- DSS.DSI_SYSCONFIG[9:8] CLOCKACTIVITY bit field set to 0x2: The interface clocks can be switched off and the functional clocks are maintained during the wake-up period.
- DSS.DISPC_SYSCONFIG[9:8] CLOCKACTIVITY bit field set to 0x3: The interface and functional clocks are maintained during the wake-up period.

The DSS power domain clock activity status is logged in the PRCM.CM_CLKSTST_DSS[0] CLKACTIVITY_DSS status bit. When set to 0, there is no domain clock activity. When set to 1, the DSS power domain clock is active.

Note: The display subsystem interface clock can be dependent on the DSS power domain state. This is configured with PRCM.CM_AUTOIDLE_DSS[0] AUTO_DSS bit:

- When the AUTO_DSS bit is set to 0 (reset value): The display subsystem interface clock is not related to the DSS power domain state transition.
- When the AUTO_DSS bit is set to 1: The display subsystem interface clock is automatically enabled or disabled along with the DSS power domain state transition.

15.3.1.4.2 Autoidle Mode

The RFBI, display controller, DSI protocol engine, and L4 interfaces can internally gate their clocks to decrease power consumption if no transaction is present on the related bus. The following bits must be set to enable this functionality:

- DSS.DSS_SYSCONFIG[0] AUTOIDLE bit (1: Autoidle; 0: Clock free-running) for the display subsystem
- DSS.RFBI_SYSCONFIG[0] AUTOIDLE bit (1: Autoidle; 0: Clock free-running) for the RFBI
- DSS.DISPC_SYSCONFIG[0] AUTOIDLE bit (1: Autoidle; 0: Clock free-running) for the display controller
- DSS.DSI_SYSCONFIG[0] AUTOIDLE bit (1: Autoidle; 0: Clock free-running) for the DSI protocol engine
- DSS.DISPC_CONFIG[9] FUNCGATED bit (1: Functional clocks gated enabled; 0: Functional clocks gated disabled) for the display controller

Note: All the bits listed above (except for the FUNCGATED bit) are set to 1 by default. It is highly recommended to set all the bits to 1 to save power.

15.3.1.4.3 Idle Mode

The display controller, DSI protocol engine, and RFBI can be configured into one of the following acknowledgment modes:

- Force-idle mode: The module immediately enters the idle state on receiving a low-power mode request from the PRCM module. In this mode, the software must ensure that there are no asserted output interrupts before requesting this mode to go into the idle state. Set the DSS.DISPC_SYSCONFIG[4:3] SIDLEMODE bit field to 0x0 (reset value) for display controller, set the DSS.DSI_SYSCONFIG[4:3] SIDLEMODE bit field to 0x0 (reset value) for DSI protocol engine, and, finally, the DSS.RFBI_SYSCONFIG[4:3] SIDLEMODE bit field to 0x0 (reset value) for RFBI.
- No-idle mode: The module never enters the idle state. Set the DSS.DISPC_SYSCONFIG[4:3] SIDLEMODE bit field to 0x1 for display controller, set the DSS.DSI_SYSCONFIG[4:3] SIDLEMODE bit field to 0x1 for DSI protocol engine, and, finally, the DSS.RFBI_SYSCONFIG[4:3] SIDLEMODE bit field to 0x1 for RFBI.
- Smart-idle mode:
  - Display controller: After receiving a low-power-mode request from the PRCM module, the display controller module enters the idle state when all the following conditions are satisfied:
    - All asserted output interrupts are acknowledged (no interrupt pending).
    - The display controller does not use anymore the L4 interface clock (DSS_L4_ICLK).
  - DSI protocol engine: After receiving a low-power-mode request from the PRCM module, the DSI protocol engine enters the idle state when all the following conditions are satisfied:
    - All asserted output interrupts are acknowledged (no interrupt pending).
    - The DSI protocol engine does not use the L4 interface clock (DSS_L4_ICLK) anymore.
    - The SCP and PWR transactions are complete.
    - No data remains in the TX FIFO (data waiting in the FIFO to be sent to the peripheral).

  To configure the display subsystem in smart-idle mode, set the DSS.DISPC_SYSCONFIG[4:3] SIDLEMODE bit field to 0x2 for display controller, set the DSS.DSI_SYSCONFIG[4:3] SIDLEMODE bit field to 0x2 for DSI protocol engine, and, finally, the DSS.RFBI_SYSCONFIG[4:3] SIDLEMODE bit field to 0x2 for RFBI.

Once the idle handshake protocol is over:

- The DSS L4 interface clock (DSS_L4_ICLK) can be shutdown at any time.
- Any transaction on the L4 configuration port is ignored.

15.3.1.4.4 SDI Idle Mode

To save power consumption, the PLL (SDI_PLL) of the SDI module can be configured in one of the operation modes shown in Table 15-20 when not locked.

Table 15-20. SDI PLL Operation Modes

SDI_PLL Operation Mode	Stop mode Low power[1]	Stop mode Low power[1]	Idle bypass Low power	Idle bypass Fast relock	Limp
	Output clock stopped Lowest power standby	Output clock stopped Fastest start-up time	Output clock at PCLK rate Lowest power standby	Output clock at PCLK rate Fastest start-up time	Output at full speed
DSS.DSS_PLL_CONTROL[0] SDI_PLL_IDLE	0	0	1	1	1 or PCLK stops
DSS.DSS_PLL_CONTROL[20] SDI_PLL_LOWCURRSTBY	1	0	1	0	X
DSS.DSS_PLL_CONTROL[17] SDI_PLL_STOPMODE	1	1	1	1	0

(1) Recommended

15.3.1.4.5 Wake-Up Mode

The Display Controller (DISPC) supports the wake-up protocol. The mode is selected by programming the appropriate value in the DSS.DISPC_SYSCONFIG[2] ENWAKEUP bit. The wake-up signal is asserted when the DISPC is in idle mode and when anyone of the following events occur:

- Graphics pipe is enabled and data fetch is not completed for graphics window, and the number of data bytes in FIFO is less than the low threshold programmed value.
- Video1 pipe is enabled and data fetch is not completed for video1 window, and the number of data bytes in FIFO is less than the low threshold programmed value.
- Video2 pipe is enabled and data fetch is not completed for video2 window, and the number of data bytes in FIFO is less than the low threshold programmed value.
- The current pixel is the last pixel displayed on the LCD panel if it is not the last frame.
- The current pixel is the last pixel displayed on the digital panel if it is not the last frame.

  If software users set the DSS.DISPC_CONFIG[17] FIFOFILLING bit, when one of the active pipe reaches the low threshold and should refill the FIFO for the current frame, the other pipes also refill their own FIFOs, even if the low threshold has not been reached. This is used to improve the probability of increasing the time when there is no access to the L3 interconnect (MStandby asserted, affects power savings).

  Once the wake-up signal is asserted, the WAKEUP interrupt request is generated. The wake-up signal is deasserted when the idle request is no longer activated.

15.3.1.4.6 Standby Mode

As part of the system-wide power-management scheme, the display controller can enter standby mode. To configure the display controller, write the DSS.DISPC_SYSCONFIG[13:12] MIDLEMODE bit field (00: Forced standby; 01: No standby; 10: Smart standby) in one of the following standby modes:

- Forced standby mode (default mode): The module enters standby mode when the module is disabled.
- No standby mode: The module never enters standby mode.
- Smart standby mode: The module enters standby state when the DISPC module is disabled or when all the three following events occur:
  - Graphics pipe is disabled or graphics pipe is enabled but data fetch completed for graphics window, or graphics pipe is enabled and data fetch is not completed and number of data bytes in FIFO is greater than the high threshold programmed value.
  - Video1 pipe is disabled or video1 pipe is enabled but data fetch completed for video1 window, or video1 pipe is enabled and data fetch is not completed and number of data bytes in FIFO is greater than the high threshold programmed value.
  - Video2 pipe is disabled or video2 pipe is enabled but data fetch completed for video2 window, or video2 pipe is enabled and data fetch is not completed and number of data bytes in FIFO is greater

than the high threshold programmed value.

When in standby mode, the display controller does not generate transactions on the L3 master port. Standby is active when the PRCM module confirms this mode.

The display subsystem standby mode activity can be monitored with the PRCM.CM_IDLEST_DSS[0] ST_DSS status register. When this register is read to 0, the display subsystem is accessible and the interface clock running; when it is read to 1, the display subsystem is in standby mode.

15.3.1.4.6.1 Conditions to Exit Standby Mode

The following conditions allow the subsystem to exit standby mode:

- Forced standby mode: Standby mode is exited when the display controller is enabled.
- Smart standby mode: Standby mode is exited when any one of the following events occurs:
  - Graphics pipe is enabled and data fetch is not completed for graphics window, and number of data bytes in FIFO is less than the low threshold programmed value.
  - Video1 pipe is enabled and data fetch is not completed for video1 window, and number of data bytes in FIFO is less than the low threshold programmed value.
  - Video2 pipe is enabled and data fetch is not completed for video2 window, and number of data bytes in FIFO is less than the low threshold programmed value.

15.3.1.4.6.2 Standby Transition Dependency

The sleep transition of the DSS power domain can be dependent or not with respect to MPU domain. This is configured by PRCM.CM_SLEEPDEP_DSS[1] EN_MPU bit:

- When the EN_MPU bit is set to 0 (reset value): The DSS power domain sleep dependency with the MPU power domain is disabled. The DSS power domain will not enter idle unless the MPU power has previously entered idle.
- When the EN_MPU bit is set to 1: The DSS power domain sleep dependency with the MPU power domain is enabled. The DSS power domain will enter idle regardless of the power domain state of the MPU.

The sleep transition of the DSS power domain may, or may not depend on the IVA2.2 domain, depending on the configuration of the PRCM.CM_SLEEPDEP_DSS[2] EN_IVA2 bit:

- When the EN_IVA2 bit is set to 0 (reset value): The DSS power domain sleep dependency on the IVA2.2 power domain is disabled.
- When the EN_IVA2 bit is set to 1: The DSS power domain sleep dependency on the IVA2.2 power domain is enabled.

15.3.1.4.6.3 Standby Procedure Description

When the display subsystem initiates a standby procedure, it also initiates an standby/wait handshake protocol with the PRCM module that lets the PRCM cut the display subsystem clocks. Depending on the PRCM setting, two modes are available:

- Manual mode
  - DSS1_ALWON_FCLK is shut down when the PRCM.CM_FCLKEN_DSS[0] EN_DSS1 bit is set to 0 and the display subsystem is in standby mode.
  - DSS2_ALWON_FCLK is shut down when the PRCM.CM_FCLKEN_DSS[1] EN_DSS2 bit is set to 0 and the display subsystem is in standby mode.
  - DSS_L3_ICLK and DSS_L4_ICLK are controlled together. They are shut down when the PRCM.CM_ICLKEN_DSS[0] EN_DSS bit is set to 0 and the display subsystem is in standby mode.

CAUTION

Do not stop DSS1_ALWON_FCLK, or DSS2_ALWON_FCLK clock (if used) if the display subsystem is not disabled.

CAUTION

DSS_TV_FCLK does not depend on the display subsystem standby state. Ensure correct clock management for DSS_TV_FCLK.

The clocks are reactivated when the related bits are set to 1 and the display subsystem exits from standby. For more information, see Chapter 4, *Power, Reset, and Clock Management*.

- Hardware- or software-supervised mode

  The DSS-power state-transition between active and inactive states can be either hardware or software supervised. This is programmed with the PRCM.CM_CLKSTCTRL_DSS[1:0] CLKTRCTRL_DSS bit field:

  – When the CLKTRCTRL_DSS bit field is set to 0x0 (reset value), the automatic transition is disabled
  – When the CLKTRCTRL_DSS bit field is set to 0x1, the software-supervised sleep transition is started on the DSS power domain.
  – When the CLKTRCTRL_DSS bit field is set to 0x2, the software-supervised wake-up transition is started on the DSS power domain.
  – When the CLKTRCTRL_DSS bit field is set to 0x3, the automatic transition is enabled. Any transition on the DSS power domain is supervised by the hardware.

15.3.1.4.6.4 Display Subsystem Standby Mode, Power-Saving Use Cases

- Setup
  – Set the display subsystem in smart standby mode.
  – Manually enable DSS_L3_ICLK and DSS_L4_ICLK.
  – Manually enable DSS1_ALWON_FCLK or DSS2_ALWON_FCLK.
  – Manually enable DSS_TV_FCLK if the video encoder is used.
  – Set the PRCM.CM_CLKSTCTRL_DSS[1:0] CLKTRCTRL_DSS bit field to 0x3 (autocontrol mode supervised by hardware).
- Shut down the display subsystem.
  – Disable the display subsystem.
  – Manually disable DSS1_ALWON_FCLK, DSS2_ALWON_FCLK, DSS_TV_FCLK, DSS_L3_ICLK, and DSS_L4_ICLK.

For more details on low-power programming settings, see Section 15.6.2.

15.3.2 Hardware Requests

15.3.2.1 DMA Requests

The display controller, the DSI protocol engine, and the RFBI generate some DMA requests to the sDMA. Figure 15-66 details the DMA tree.

Figure 15-66. Display Subsystem DMA Tree

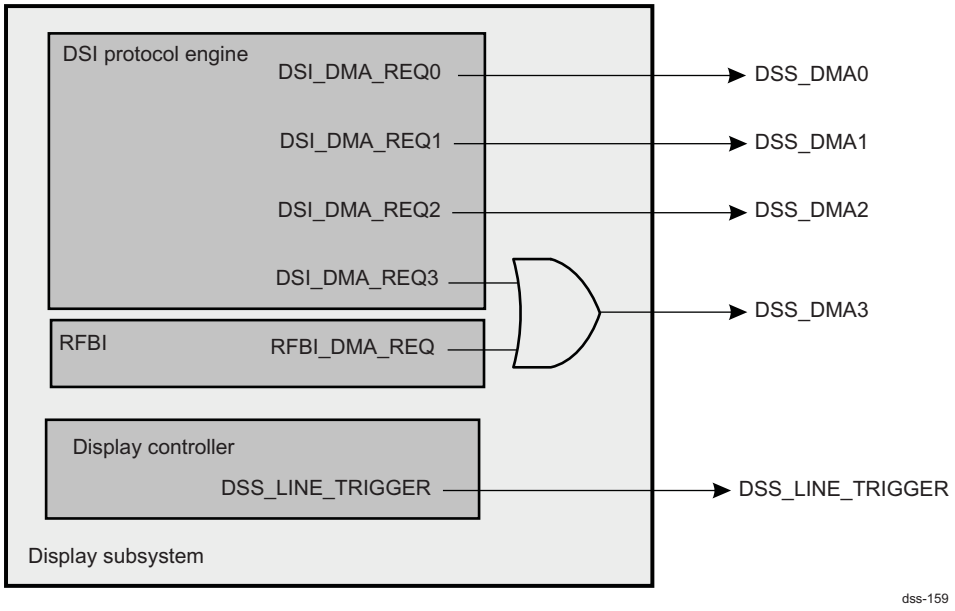


Table 15-21. DSS DMA Requests Description

DMA Request Name	DSS Module	Mapping	Description
DSS_LINE_TRIGGER	DISPC	S_DMA_5	See Section 15.3.2.1.1
DSI_DMA_REQ0	DSI protocol engine	S_DMA_71	See Section 15.3.2.1.2
DSI_DMA_REQ1	DSI protocol engine	S_DMA_72	See Section 15.3.2.1.2
DSI_DMA_REQ2	DSI protocol engine	S_DMA_73	See Section 15.3.2.1.2
DSI_DMA_REQ3	DSI protocol engine	S_DMA_74	See Section 15.3.2.1.2
RFBI_DMA_REQ	RFBI	S_DMA_74	See Section 15.3.2.1.3

Note: The DMA requests from the RFBI module (RFBI_DMA_REQ) and the DSI protocol engine (DSI_DMA_REQ3) are merged on line DSS_DMA3. The software must only use DSS_DMA3 on one module at a time (RFBI or DSI protocol engine).

15.3.2.1.1 Display Controller DMA Request (Line Trigger)

One DMA synchronization line (DSS_LINE_TRIGGER) is connected to the sDMA by the sDMA controller (S_DMA_5) input line. This DMA request is not a classical one but a synchronization signal from the display subsystem to the sDMA informing the sDMA that a programmable number of lines are output to the LCD, and that the system memory can be updated. This request is related to an interrupt event described in Section 15.3.2.2, *Interrupt Requests*. This allows the sDMA channel to be synchronized with the display subsystem internal DMA controller. In other words, it allows to synchronize a memory to memory frame buffer update based on the scan line of the frame buffer in system memory (SDRAM or SRAM) by the display controller. The DSS_LINE_TRIGGER DMA request is generated at a programmable line number defined in DSS.DISPC_LINE_NUMBER[10:0] LINENUMBER bit field.

15.3.2.1.2 DSI Protocol Engine DMA Request

The DSI DMA requests are used to allow automatic transfer by the sDMA or MPU (with less efficiency and through-put capability) from the DSI RX FIFO to the system memory and from the system memory to the DSI TX FIFO. Two independent DMA requests for RX FIFO and TX FIFO for the same VC are supported.

15.3.2.1.3 RFBI DMA Request

The RFBI_DMA_REQ is used to receive data into the RFBI FIFO. The DMA request is always generated when there is enough room in the FIFO to accept the full burst.

15.3.2.2 Interrupt Requests

The DSI protocol engine, the DSI complex I/O (DSI_IRQ), and the display controller (DISPC_IRQ) generate one interrupt request each. The DSI_IRQ and DISPC_IRQ lines are merged together in a single interrupt line.

One interrupt line (DSS_IRQ) is connected to two interrupt controllers:

- MPU interrupt controller (M_IRQ_25 input line)
- IVA interrupt handler (IVA2_IRQ[13] input line)

Figure 15-67 shows the interrupt tree for the DSI protocol engine and DSI complex I/O in detail.

Figure 15-67. DSI Interrupt Tree

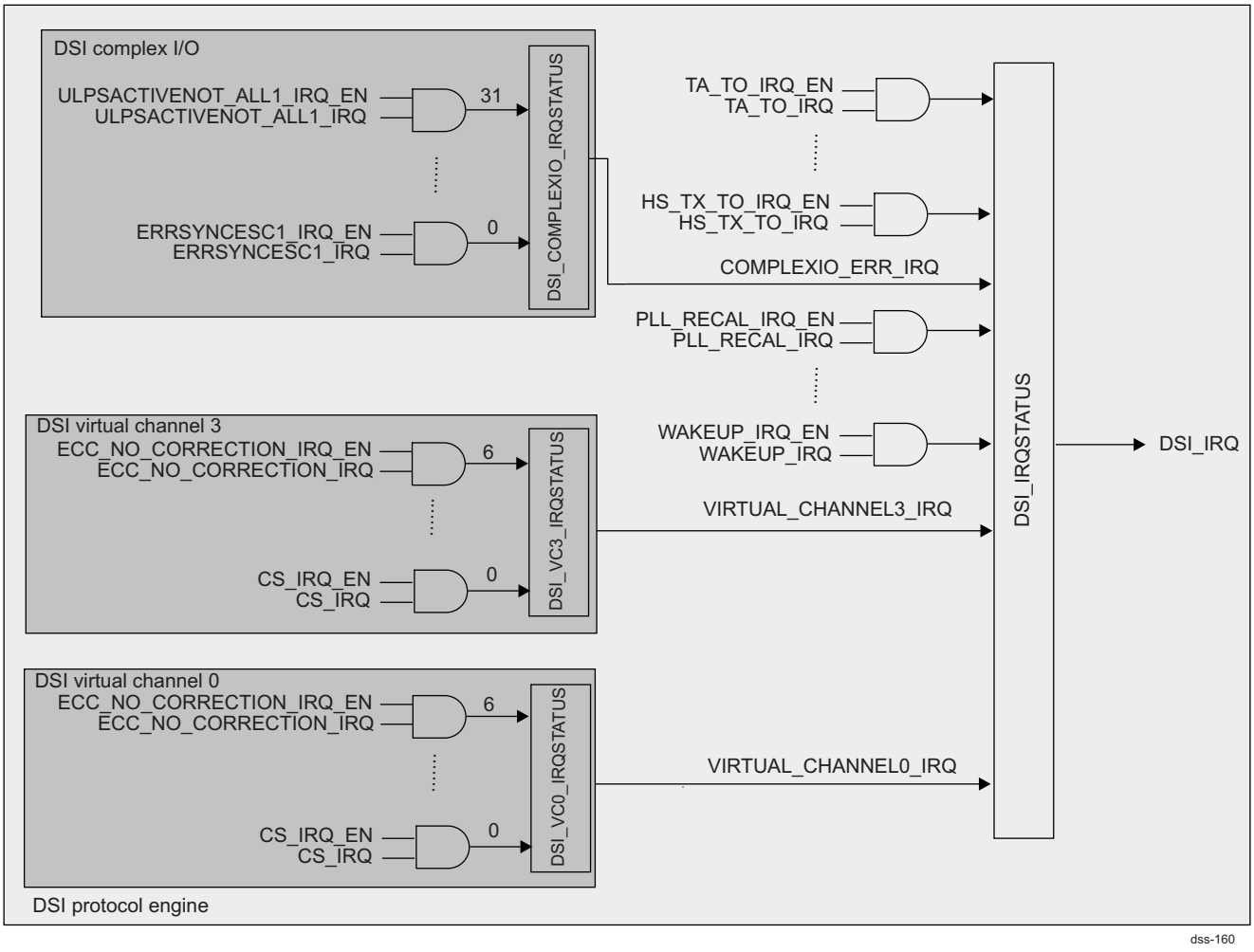
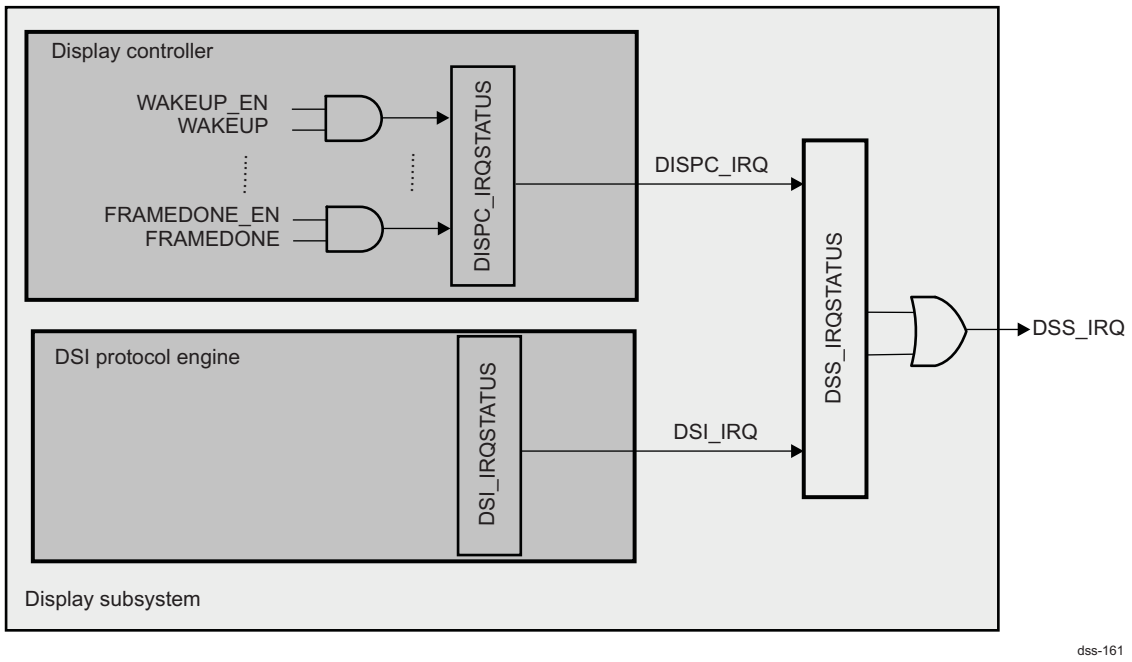


Figure 15-68 details the interrupt tree for the DISPC and the display subsystem.

Figure 15-68. DISPC and DSS Interrupts Tree



15.3.2.2.1 DISPC Interrupt Request

The interrupt line indicates when one or more events are detected by the hardware. Each event is independently maskable by setting the DSS.DISPC_IRQENABLE register.

To check when a particular interrupt event occurs and to reset a particular event, the DSS.DISPC_IRQSTATUS register must be accessed. This register regroups all the status of the module internal events that generate an interrupt (read 0: No interrupt occurred; read 1: Interrupt occurred; write 1: Status bit reset). See Section 15.7, *Display Subsystem Register Manual*, for more information on checking and clearing interrupt events.

Table 15-22 lists the display subsystem interrupt events.

Table 15-22. Display Subsystem Interrupts

Interrupt Name	Description
FRAMEDONE	Active frame is complete and LCD output is disabled.
VSYNC	VSYNC interrupt occurred at the end of the frame.
EVSYNC_EVEN(1)	EVSYNC_EVEN interrupt occurred at the end of the frame. (EVSYNC is received and the field polarity is even.)
EVSYNC_ODD(1)	EVSYNC_ODD interrupt occurred at the end of the frame. (EVSYNC is received and the field polarity is odd.)
ACBIASCOUNTSTATUS	The ac-bias transition counter decremented to 0.
PROGRAMMEDLINENUMBER	The LCD reached the user-programmed line number.
GFXFIFOUNDERFLOW	The input graphics FIFO goes underflow.
GFXENDWINDOW	The screen reached the end of the graphics window. All data for the graphics window are fetched from memory and displayed on the screen.
PALETTEGAMMALOADING	The palette/gamma table is loaded.
OCPERROR	L3 interconnect sent SResp = ERR.
VID1FIFOUNDERFLOW	The input video1 FIFO goes underflow.

(1) EVYNC interrupts (EVSYNC_EVEN and EVSYNC_ODD) are external interrupts received by the display controller and generated by the video encoder (VENC) module.

Table 15-22. Display Subsystem Interrupts (continued)

Interrupt Name	Description
VID1ENDWINDOW	The screen reached the end of video1 window. All data for the video window are fetched from the memory and displayed on the screen.
VID2FIFOUNDERFLOW	The input video2 FIFO goes underflow.
VID2ENDWINDOW	The screen reached the end of video2 window. All data for the video window are fetched from the memory and displayed on the screen.
SYNCLOST	Interrupt occurs when VSYNC width/front or back porches are not wide enough to load the pipelines with data (LCD output).
SYNCLOSTDIGITAL	Interrupt occurs when the display controller is not ready to output data when a digital request occurs. This interrupt informs that the timings of the NTSC/PAL video encoder are not set correctly.
WAKEUP	Occurs when the wakeup signal is asserted

Note: To clear a synchronization lost interrupt, follow this sequence:

1. Clear the DSS.DISPC_CONTROL[0] LCDENABLE (LCD: SYNCLOST interrupt) or DSS.DISPC_CONTROL[1] DIGITALENABLE (TV: SYNCLOSTDIGITAL interrupt) bits.
   Check the interrupts.
   LCD: Verify that a FRAMEDONE interrupt occurs.
   TV : Verify that EVSYNC_EVEN or EVSYNC_ODD interrupts occur.
2. Set the DSS.DSS_SYSCONFIG[1] SOFTRESET bit to reset the display subsystem.
3. Set the display subsystem registers again.

Note: The SYNCLOSTDIGITAL interrupts, which occur before the first VSYNC pulse signal (from the video encoder), should not be considered.

After the first VSYNC pulse signal, the SYNCLOSTDIGITAL interrupt status bit must be cleared by writing 1 in the DSS.DISPC_IRQSTATUS[15] SYNCLOSTDIGITAL bit; then the SYNCLOSTDIGITAL interrupt can be enabled by setting the DSS.DISPC_IRQENABLE[15] SYNCLOSTDIGITAL bit.

15.3.2.2.2 DSI Interrupt Request

The DSI protocol engine requires a single interrupt line, DSI_IRQ. The DSS.DSI_IRQSTATUS register indicates the general interrupt events. Refer to Table 15-23. Each VC and complex I/O has a dedicated interrupt register: DSS.DSI_VCn_IRQSTATUS and DSS.DSI_COMPLEXIO_IRQSTATUS respectively. Refer to Table 15-24 and Table 15-25.

Table 15-23 indicates the DSI global interrupt events.

Table 15-23. DSI Global Interrupts

Interrupt Name	Description
RESYNCHRONIZATION_IRQ	Resynchronization in video mode
TA_TO_IRQ	Turn-around timer expired
LDO_POWER_GOOD_IRQ	Signal LDOPWRGOOD from the DSI_PHY changes its state for the supply VDDALDODSIPLL from up to down or down to up.
SYNC_LOST_IRQ	Synchronization with video mode port is lost (video mode only)
ACK_TRIGGER_IRQ	Acknowledge trigger is received
TE_TRIGGER_IRQ	Tearing effect trigger is received
WAKEUP_IRQ	Occurs when the SWakeUp signal is asserted
HS_TX_TO_IRQ	High speed TX Time-out Interrupt
LP_RX_TO_IRQ	Low speed RX Time-out Interrupt

Table 15-23. DSI Global Interrupts (continued)

Interrupt Name	Description
COMPLEXIO_ERR_IRQ	Error signaling from complex I/O: The interrupt is triggered when any error is received from the complex I/O (events are defined in DSI_COMPLEXIO_IRQSTATUS).
PLL_RECAL_IRQ	PLL recal event (assertion of DSIRecal signal from the DSI PLL Control module)
PLL_UNLOCK_IRQ	PLL unlock event (deassertion of DSILock signal from the DSI PLL Control module)
PLL_LOCK_IRQ	PLL lock event (assertion of DSILock signal from the DSI PLL Control module)
VIRTUAL_CHANNEL3_IRQ	Virtual channel #3 Error signaling from DSI Virtual Channel3: The interrupt is triggered when an error is received from DSI Virtual Channel3 (events are defined in DSI_VC3_IRQENABLE).
VIRTUAL_CHANNEL2_IRQ	Virtual channel #2 Error signaling from DSI Virtual Channel2: The interrupt is triggered when an error is received from DSI Virtual Channel2 (events are defined in DSI_VC2_IRQENABLE).
VIRTUAL_CHANNEL1_IRQ	Virtual channel #1 Error signaling from DSI Virtual Channel1: The interrupt is triggered when an error is received from DSI Virtual Channel1 (events are defined in DSI_VC1_IRQENABLE).
VIRTUAL_CHANNEL0_IRQ	Virtual channel #0 Error signaling from DSI Virtual Channel0: The interrupt is triggered when an error is received from DSI Virtual Channel0 (events are defined in DSI_VC0_IRQENABLE).

Table 15-24 indicates the DSI complex I/O interrupt events.

Table 15-24. DSI Complex I/O Interrupts

Interrupt Name	Description
ULPSActiveNot_ALL0_IRQ	All signals ULPSActiveNOT are 0
ULPSActiveNot_ALL1_IRQ	All the ULPSActiveNOT signals corresponding to the lanes with TXULPSExit being high are high
STATEULPS3_IRQ	Lane #3 in ultralow-power state
STATEULPS2_IRQ	Lane #2 in ultralow-power state
STATEULPS1_IRQ	Lane #1 in ultralow-power state
ERRCONTROL3_IRQ	Control error for lane #3
ERRCONTROL2_IRQ	Control error for lane #2
ERRCONTROL1_IRQ	Control error for lane #1
ERRESC3_IRQ	Escape entry error for lane #3(edge trigger interrupt)
ERRESC2_IRQ	Escape entry error for lane #2 (edge trigger interrupt)
ERRESC1_IRQ	Escape entry error for lane #1 (edge trigger interrupt)
ERRCONTENTIONLP1_1_IRQ	Contention LP1 error for lane #1
ERRCONTENTIONLP0_1_IRQ	Contention LP0 error for lane #1
ERRCONTENTIONLP1_2_IRQ	Contention LP1 error for lane #2
ERRCONTENTIONLP0_2_IRQ	Contention LP0 error for lane #2
ERRCONTENTIONLP1_3_IRQ	Contention LP1 error for lane #3
ERRCONTENTIONLP0_3_IRQ	Contention LP0 error for lane #3
ERRSYNCESC3_IRQ	Low power Data transmission synchronization error for lane #3
ERRSYNCESC2_IRQ	Low power Data transmission synchronization error for lane #2
ERRSYNCESC1_IRQ	Low power Data transmission synchronization error for lane #1

Note: The error contention signals for DX and DY signals of each lane are ORed together.

Table 15-25 indicates the DSI VCs interrupt events

Table 15-25. DSI Virtual Channel Interrupts

Interrupt Name	Description
ECC_CORRECTION_IRQ	Virtual channel - ECC has been used to correct a 1-bit error (short and long packet only). Indicates whether a 1-bit error correction occurred using the ECC.
PACKET_SENT_IRQ	Indicates that a packet has been sent. It is used when BTA manual mode is used
CS_IRQ	Virtual channel - Check-Sum of the payload mismatch detection
FIFO_RX_OVF_IRQ	RX FIFO overflow. The FIFO used on the L4 interconnect slave port for buffering the data received on the DSI link has overflowed
FIFO_TX_OVF_IRQ	TX FIFO overflow. The FIFO used on the L4 interconnect slave port for buffering the data received on the L4 interconnect slave port has overflowed
BTA_IRQ	Bus turnaround is received from the peripheral (the VC ID used for the last BTA request transfer to the peripheral is used to determine which VC is used to flag the interrupt)
ECC_NO_CORRECTION_IRQ	ECC error (short and long packets). No correction of the header because of more than 1-bit error
FIFO_TX_UDF_IRQ	TX FIFO underflow. The FIFO used on the slave port for buffering the data received on the L4 interconnect port has under-flowed in the middle of a packet transfer

15.4 Display Subsystem Functional Description

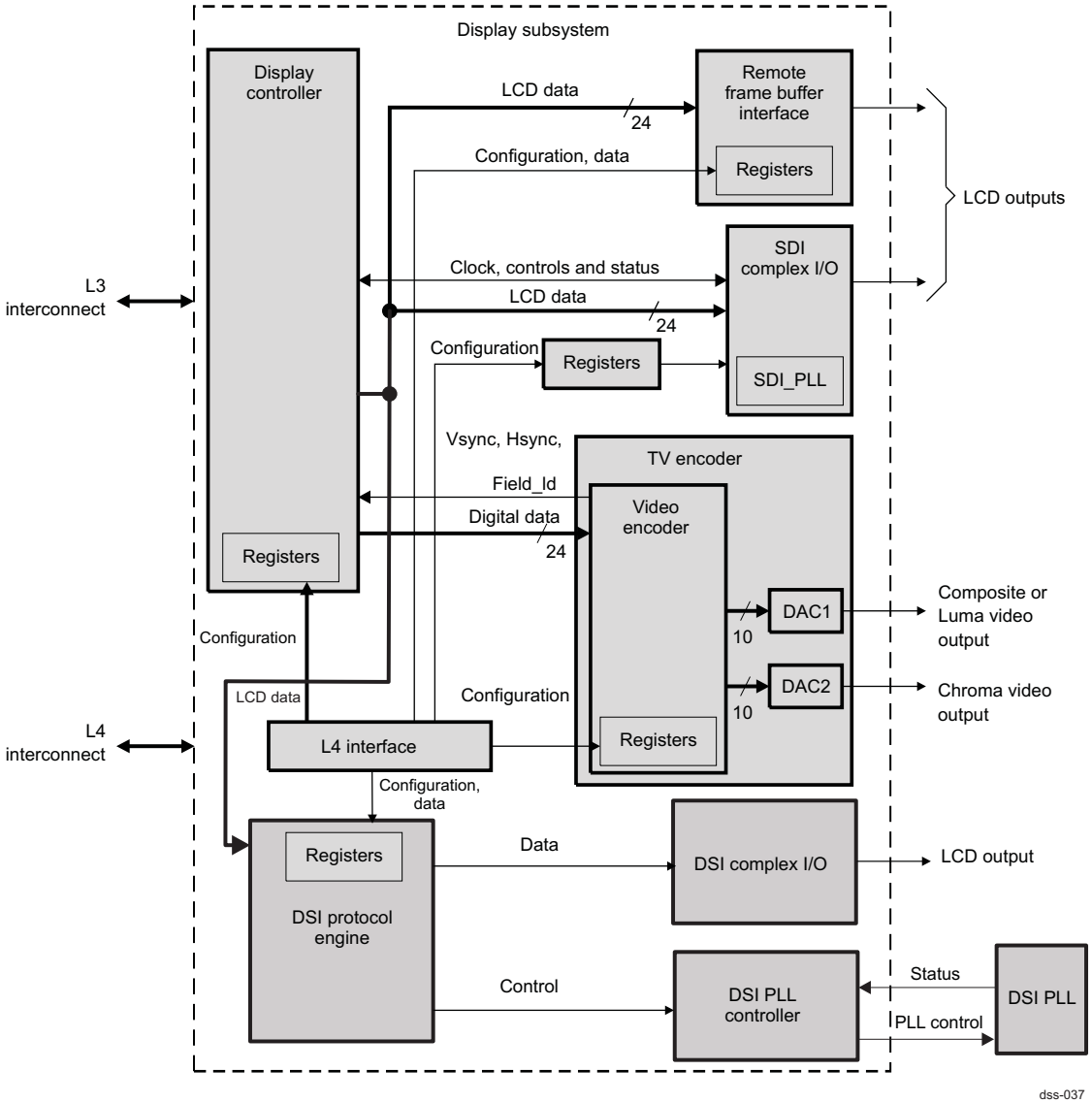
This section describes the functionalities of the LCD and TV display supports by describing the following modules: Display controller, DSI protocol engine, DSI PLL controller, DSI complex I/O, RFBI, SDI, and video encoder.

The functionalities of the display controller are common to both LCD and TV data paths; the RFBI and SDI functionalities are LCD-specific; and the video encoder functionalities are specific to the TV set.

15.4.1 Block Diagram

Figure 15-69 is a schematic of the display subsystem.

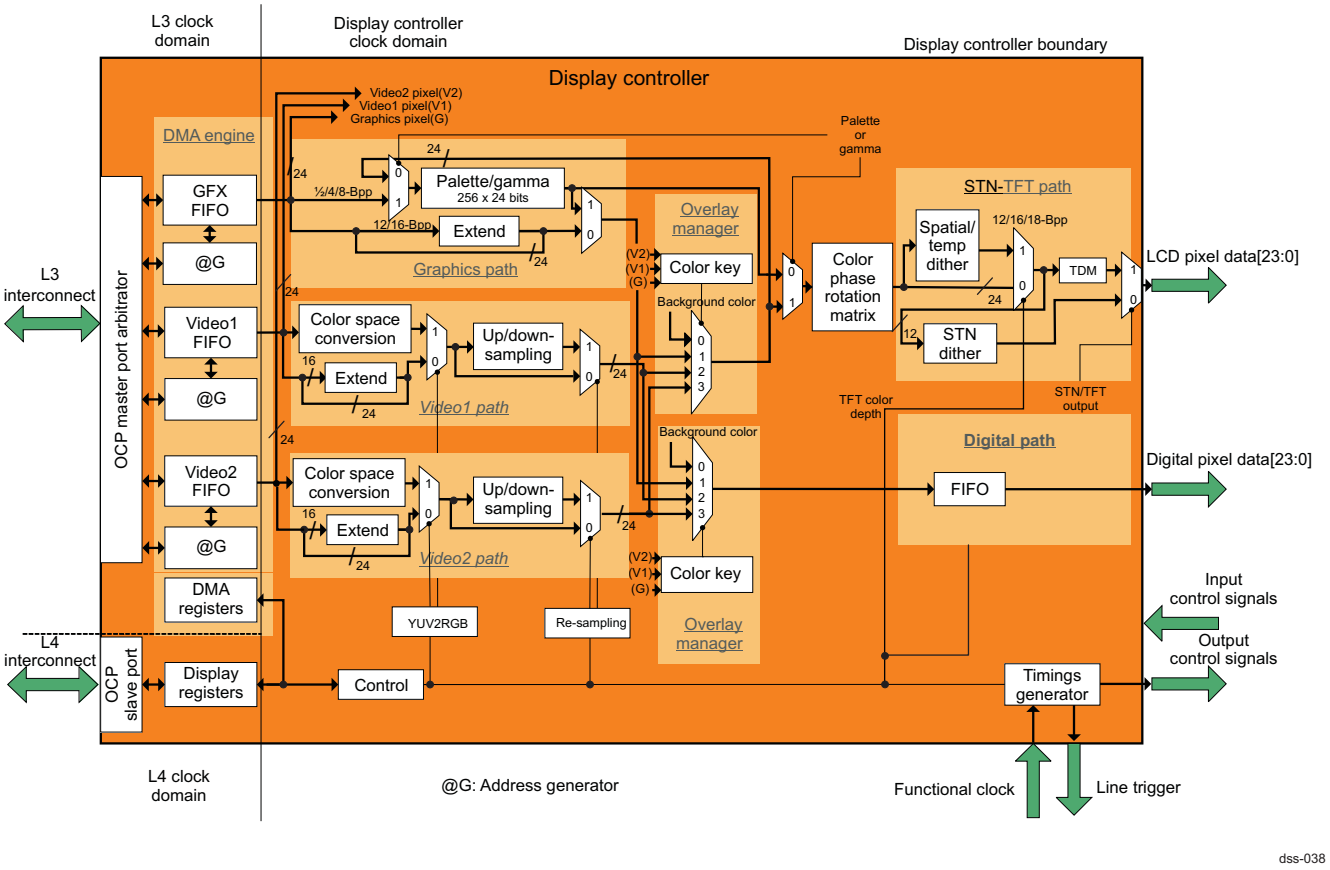
Figure 15-69. Display Subsystem Full Schematic



15.4.2 Display Controller Functionalities

The display controller can read and display the encoded pixel data stored in memory (see Figure 15-70).

Figure 15-70. Display Controller Architecture Overview



Several processes can be configured to manage the graphics pipeline (palette, gamma table correction) and video pipeline (color space conversion, upsampling, downsampling, overlay, and transparency features).

The internal timing generator logic generates the LCD input signals. The external timing generator generates the appropriated signals to drive the digital output. The data from the two overlay managers are sent on the two concurrent 24-bit buses outside the display controller module. The memory accessed by the display controller is either the SDRAM memory or the SRAM memory.

15.4.2.1 Display Modes

15.4.2.1.1 LCD Output

The display subsystem supports two types of display technologies (both monochrome and color modes):

- Passive matrix displays
- Active matrix displays

The passive matrix display mode supports 3375 possible colors, allowing 16, 256, or 3375 colors to be displayed in each frame, depending on the color depth. The monochrome LCD has 15 grayscale levels available.

In active matrix display mode, the configuration of colors depends on the color depth:

- 24 BPP supports 16,777,216 colors.
- 18 BPP supports 262,144 colors.
- 16 BPP supports 65,536 colors.
- 12 BPP supports 4096 colors.

15.4.2.1.2 Digital Output

The digital output is always a 24-bit RGB value based on an external pixel request.

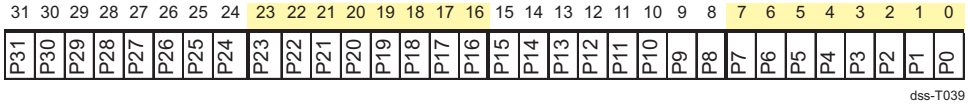
15.4.2.2 Graphics Pipeline

The graphics pipeline is connected to the graphics FIFO controller for the input port and to the two overlay managers (LCD and digital). It consists of one 256-entry palette and some programmable replication logic. The replication logic is used to convert the RGB pixels, excluding the RGB24 format, into RGB24 format based on user programming (replication of the most-significant bits [MSBs] for the RGB24 LSBs or use of 0s). The first unit connected to the input port of the graphics pipeline is the replication logic used for RGB pixels, then the second unit is the palette for concerned pixels.

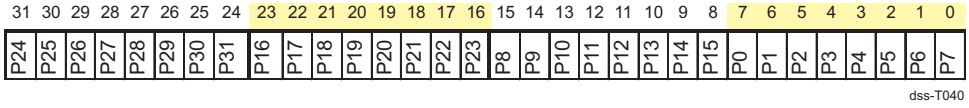
15.4.2.2.1 Graphics Memory Format

The supported formats for the graphics layer are CLUT bitmaps (1-, 2-, 4-, and 8-BPP) and true color bitmaps in RGB formats (12-, 16-, and 24-BPP [packet and nonpacket RGB24]) and in ARGB or RGBA formats (ARGB 16-, and 32-BPP, and RGBA 32-BPP) as follows:

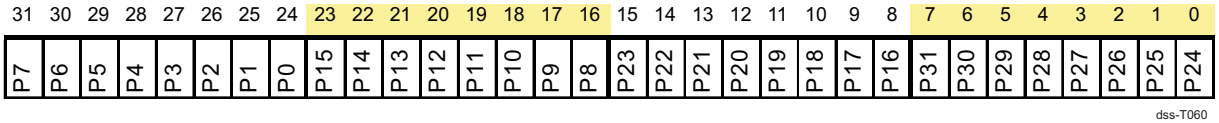
- BITMAP 1-BPP data memory organization (CLUT) (little endian)



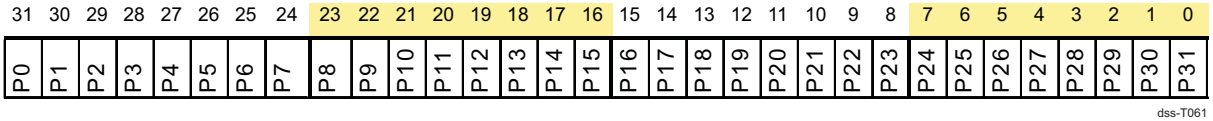
- BITMAP 1-BPP data memory organization (CLUT) (little endian + nibble mode)



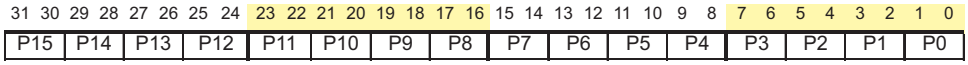
- BITMAP 1-BPP data memory organization (CLUT) (big endian)



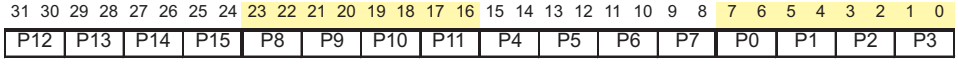
- BITMAP 1-BPP data memory organization (CLUT) (big endian + nibble mode)



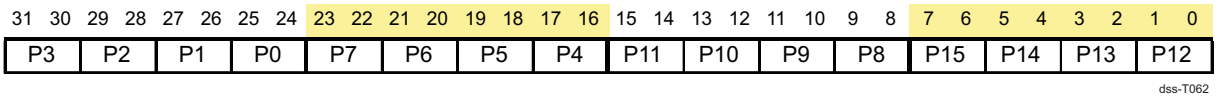
- BITMAP 2-BPP data memory organization (CLUT) (little endian)



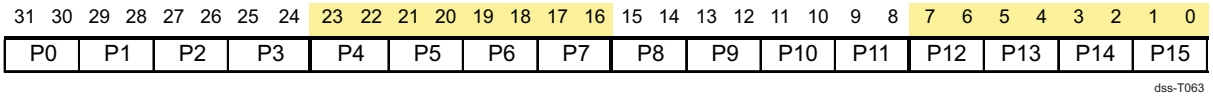
- BITMAP 2-BPP data memory organization (CLUT) (little endian + nibble mode)



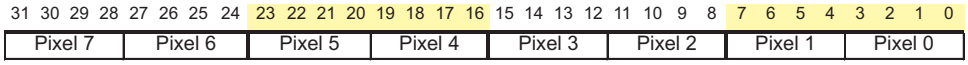
- BITMAP 2-BPP data memory organization (CLUT) (big endian)



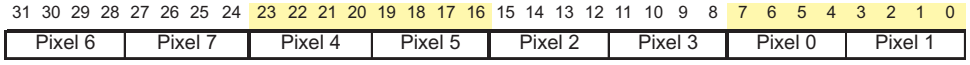
- BITMAP 2-BPP data memory organization (CLUT) (big endian + nibble mode)



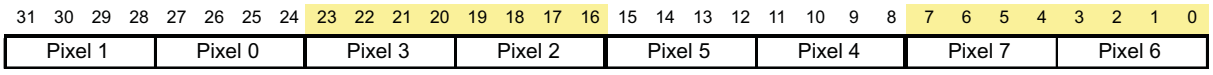
- BITMAP 4-BPP data memory organization (CLUT) (little endian)



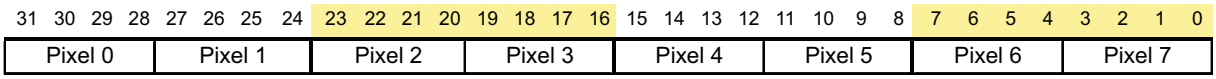
- BITMAP 4-BPP data memory organization (CLUT) (little endian + nibble mode)



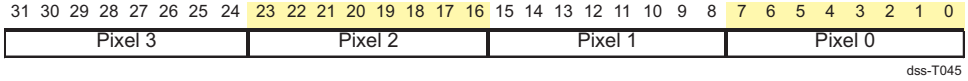
- BITMAP 4-BPP data memory organization (CLUT) (big endian)



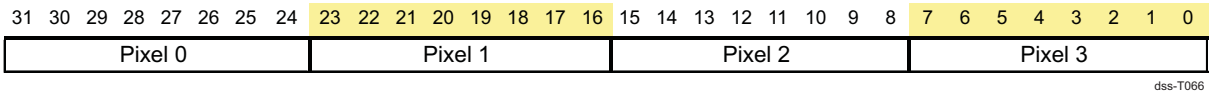
- BITMAP 4-BPP data memory organization (CLUT) (big endian + nibble mode)



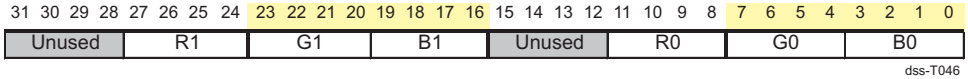
- BITMAP 8-BPP data memory organization (CLUT) (little endian)



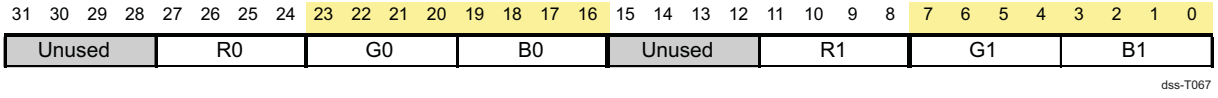
- BITMAP 8-BPP data memory organization (CLUT) (big endian)



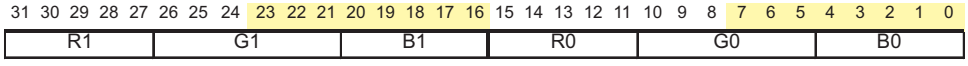
- RGB 12-BPP data memory organization (little endian)



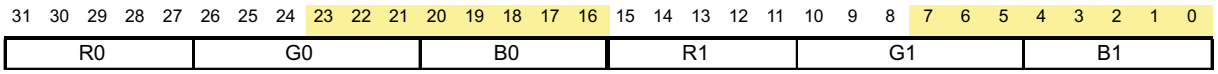
- RGB 12-BPP data memory organization (big endian)



- RGB 16-BPP data memory organization (little endian)

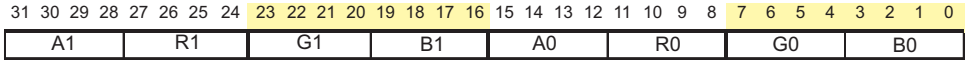


- RGB 16-BPP data memory organization (big endian)



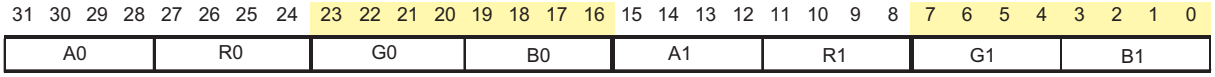
dss-T070

- ARGB 16-BPP data memory organization (little endian)



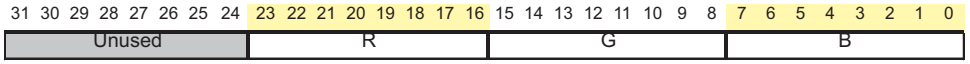
dss-T049

- ARGB 16-BPP data memory organization (big endian)



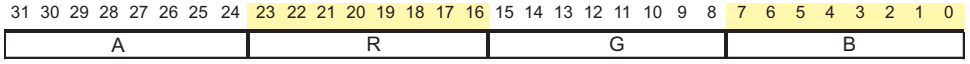
dss-T069

- RGB 24-BPP data memory organization (little or big endian)



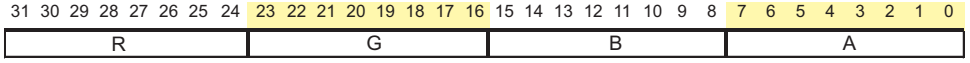
dss-T050

- ARGB 32-BPP data memory organization (little or big endian)



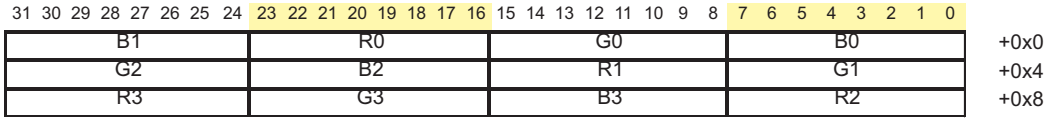
dss-T051

- RGBA 32-BPP data memory organization (little or big endian)



dss-T052

- RGB 24-BPP packet data memory organization (little or big endian)



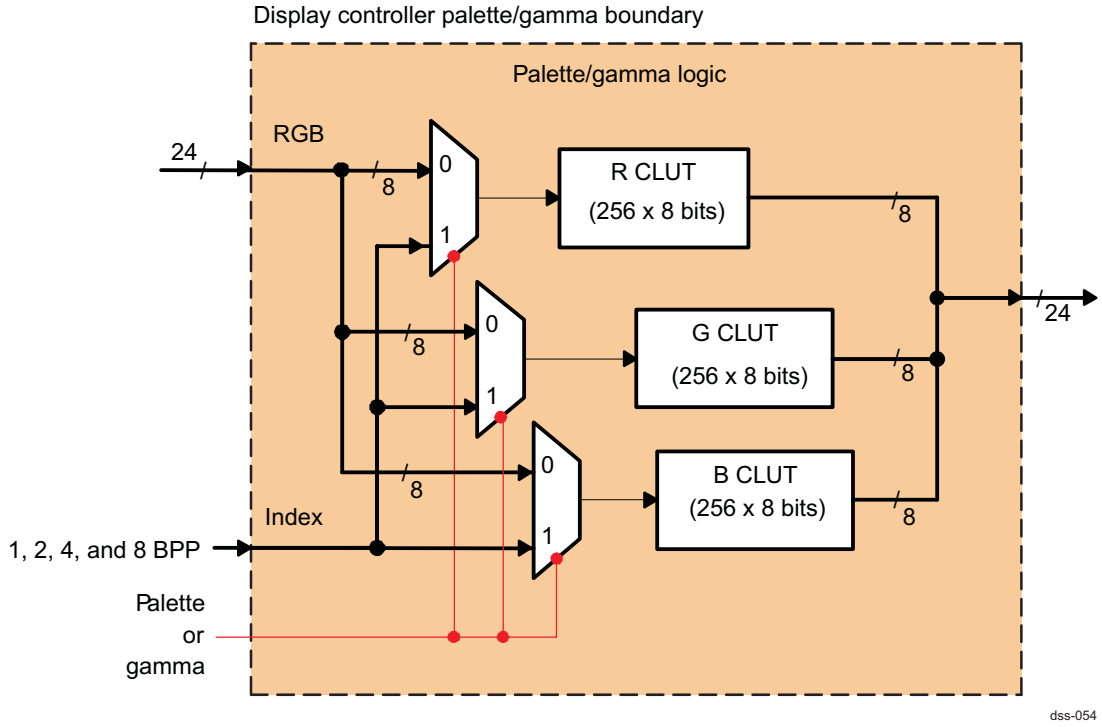
dss-T053

15.4.2.2.2 Color Look-Up Table/Gamma Table

The graphics path supports the palette/gamma table. Figure 15-71 shows the internal architecture of the color look-up/gamma table.

The palette is split into three memories of 256-bit x 8-bit entries. For bitmap (CLUT) indexes, the same value (1-, 2-, 4-, or 8-BPP) indexes the three memories. For gamma curve correction, each R, G, and B component indexes the corresponding memory to combine the three gamma curve values into a 24-bit value. The table can be reloaded every frame, once or never (at the beginning of the frame before fetching the pixels for the graphics and/or video windows).

Figure 15-71. Palette/Gamma Correction Architecture



15.4.2.2.2.1 Color Look-Up Table

The palette mode uses the encoded pixel values from the input graphics FIFO as pointers to index the 24-bit-wide palette: 1-BPP pixels address 2 palette entries, 2-BPP pixels address 4 palette entries, 4-BPP pixels address 16 palette entries, and 8-BPP pixels address 256 palette entries.

When a palette entry is selected by the encoded pixel value, the content of the entry is sent to the color/grayscale space/time base passive matrix dithering circuit, or to the color time base active matrix dithering circuit.

In the color mode, the value within the palette is made up of three 8-bit fields, one for each color component (red, green, and blue). For color operation, an individual frame is limited to a selection of 256 colors (the number of palette entries). In the monochrome mode, only one 8-bit value is present.

After passing through the palette, 256 gray scales and 16,777,216 colors are numbers obtained. A redundancy introduced in the dithering logic step reduces these numbers when displaying. For passive matrix panels, the colors are limited to 15 gray scales and 3375 colors.

- Passive matrix technology
  The palette is bypassed in 12, 16, and 24 BPP. The palette is not used.
- Active matrix technology
  The palette is bypassed in 12, 16, and 24 BPP, allowing up to 2^{24} = 16,777,216 colors to be displayed.

15.4.2.2.2.2 Gamma Table

In the gamma curve mode, the selected encoded pixel values based on the color keys from the video or graphics paths are sent to the gamma curve table. The mode is available only if the color look-up palette is not used for graphics. The output of the gamma curve processing is always sent to the LCD output. It is not available on digital output.

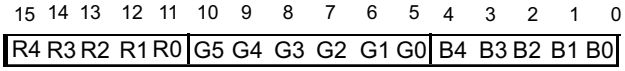
Each component of encoded pixel value is used as a pointer to index 1 out of 256 24-bit gamma curve entries in the table. Each 8-bit component is replaced with the 8-bit table value corresponding to an R, G, or B component.

15.4.2.2.2.1 Replication Logic

The replication logic increases the color depth of the graphics and video encoded pixels (from true color RGB 12-, and 16-BPP to 24-BPP). The encoded value is shifted to the 24-bit alignment. The MSB bits are copied to the LSB missing ones. Then the graphics are merged with the video data based on the transparency color keys. When the replication logic is not selected, the encoded pixel values are shifted to the MSB boundary of the 24-bit format. The missing bit values are filled up with 0s.

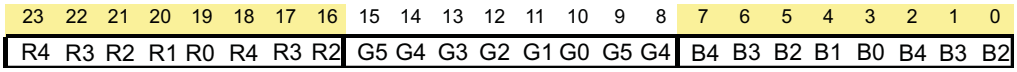
This is an example for RGB16 extension:

- Original 16-BPP data:



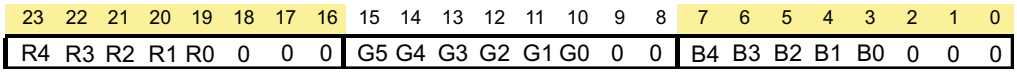
dss-T055

- If replication logic is ON:



dss-T056

- If replication logic is OFF:



dss-T057

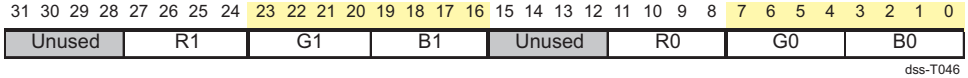
15.4.2.3 Video Pipeline

The video pipeline is connected to the video FIFO controller for the input port and to the two overlay managers (LCD and digital). It consists of the Re-Sampling unit, the Color Space Conversion Unit, and some programmable replication logic. The replication logic is used to convert the RGB pixels, excluding the RGB24 format, into RGB24 format based on user programming (replication of the MSBs for the RGB24 LSBs or use of 0s). The first unit connected to the input port of the video pipeline is the Re-Sampling Unit, then the replication logic used for RGB pixels, then the Color Space Conversion Unit for YUV422 pixels.

15.4.2.3.1 Video Memory Formats

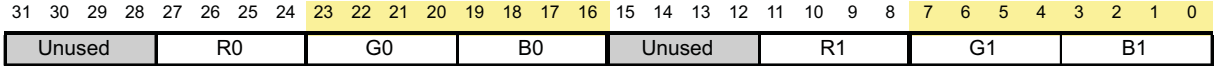
The display subsystem supports the following formats for the video layer: YUV2, UYVY, RGB12, RGB16, RGB24 (non-packed and packed formats), ARGB16 (video channel 2 only), ARGB32 (video channel 2 only), and RGBA32 (video channel 2 only).

- RGB 12-BPP data memory organization (little endian)



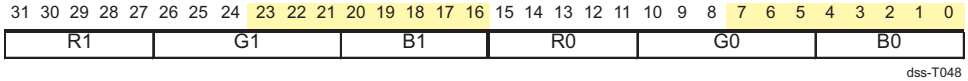
dss-T046

- RGB 12-BPP data memory organization (big endian)



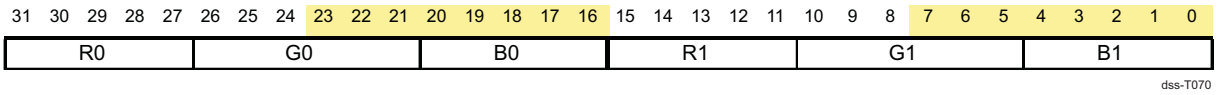
dss-T067

- RGB 16-BPP data memory organization (little endian)

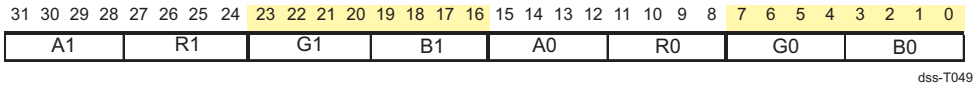


dss-T048

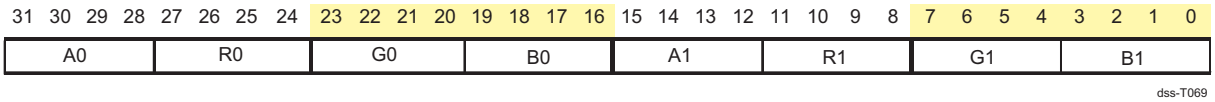
- RGB 16-BPP data memory organization (big endian)



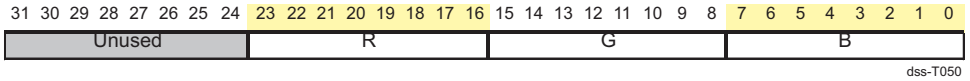
- ARGB 16-BPP data memory organization (little endian + video 2 channel only)



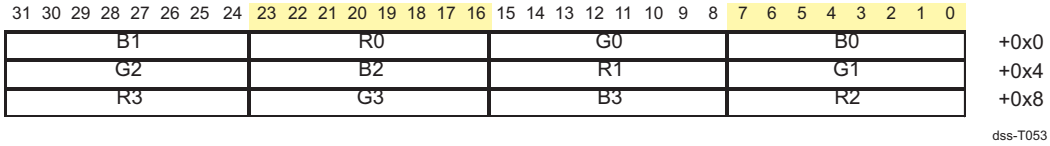
- ARGB 16-BPP data memory organization (big endian + video 2 channel only)



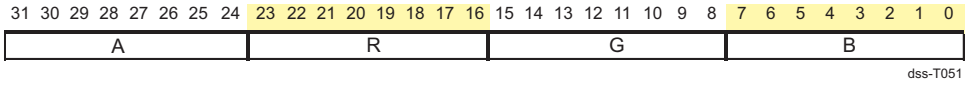
- RGB 24-BPP data memory organization (little or big endian)



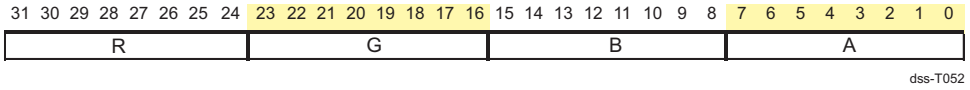
- RGB 24-BPP packet data memory organization (little or big endian)



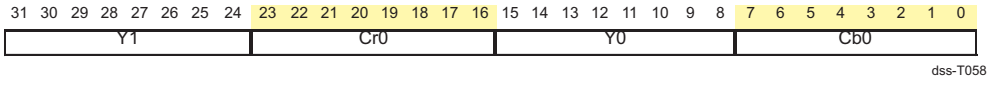
- ARGB 32-BPP data memory organization (little or big endian + video 2 channel only)



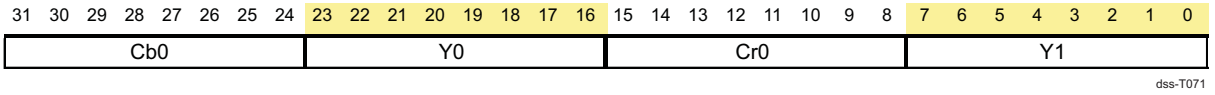
- RGBA 32-BPP data memory organization (little or big endian + video 2 channel only)



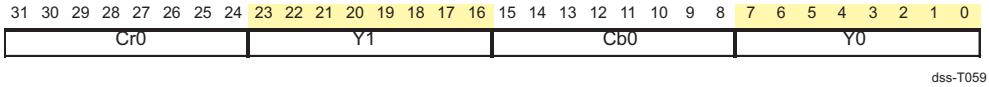
- UYVY 4:2:2 data memory organization (little endian)



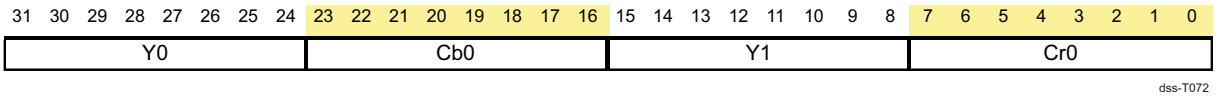
- UYVY 4:2:2 data memory organization (big endian)



- YUV2 4:2:2 data memory organization (little endian)



- YUV2 4:2:2 data memory organization (big endian)



15.4.2.3.2 Color Space Conversion

The color space conversion module converts the video-encoded pixel values from YCbCr 4:2:2 format into RGB24. Figure 15-72 and Figure 15-73 detail the YCbCr 4:2:2 conversion to YCbCr 4:4:4 depending on the rotation parameters.

Figure 15-72. YCbCr 4:2:2 to YCbCr 4:4:4 (0- or 180-Degree Rotation)

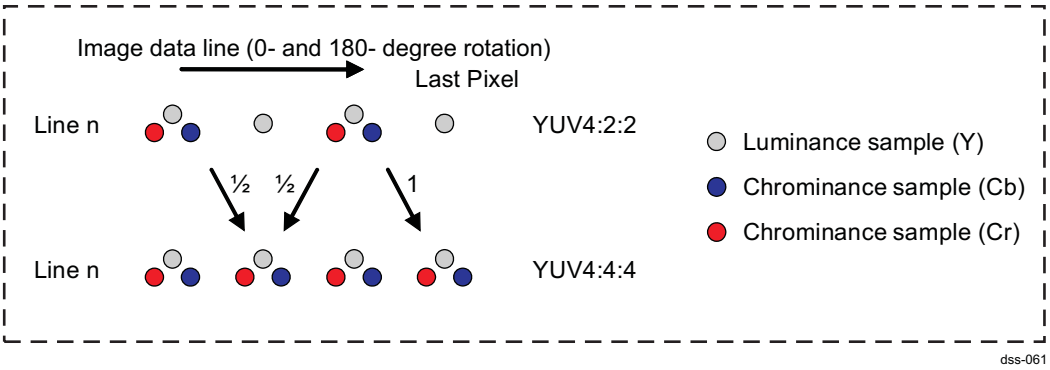
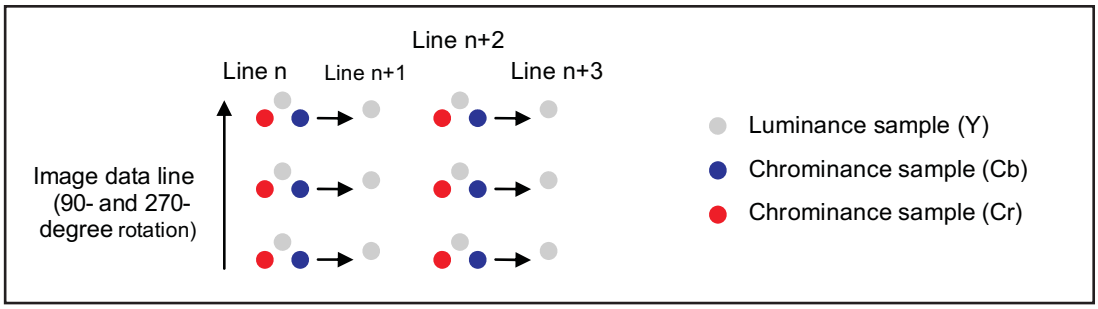


Figure 15-73. YCbCr 4:2:2 to YCbCr 4:4:4 (90- or 270-Degree Rotation)



The interpolation of the missing chrominance component is given by the equation in Figure 15-74.

Figure 15-74. Interpolation of the Missing Chrominance Component

$$Cb_n(YCbCr\ 444) = \frac{Cb_{n-1}(YCbCr\ 422) + Cb_{n+1}(YCbCr\ 422)}{2}\ (\text{n odd})$$

$$Cr_n(YCbCr\ 444) = \frac{Cr_{n-1}(YCbCr\ 422) + Cr_{n+1}(YCbCr\ 422)}{2}\ (\text{n odd})$$

dss-E062

First, to convert the YCbCr 4:2:2 encoded pixel values into YCbCr 4:4:4 format, the missing chrominance samples (Cb and Cr) are interpolated using the average values of the two closest values on the same line (1/2, 1/2) or are repeated from the second pixel in the same 32-bit container.

- In case of rotation 0-degree, for the last pixel, the chrominance samples are duplicated using the values from the previous pixel; otherwise, the chrominance samples are averaged using the two adjacent values.
- In case of 180-degree rotation, for the first pixel the chrominance samples missing are duplicated from the adjacent pixel; otherwise, the chrominance samples are averaged using the two adjacent values.
- In case of rotation 90- and 270-degree, the missing chrominance components are duplicated from the adjacent pixel in the same 32-bit container.

  In case of 5-tap configuration for the vertical filtering, the missing chrominance samples are always duplicated using the second chrominance samples in the same 32-bit value.

  Then the pixels are converted from YCbCr color space into the RGB color space, because the output format of the color space conversion is RGB24 (8-bit value per component: Red, green, and blue). The following matrices show the 11-bit coefficients registers used to convert from YCbCr 4:4:4 into RGB24. Users set the coefficients according to the standard used to encode the pixel data in YCbCr color space.

  In case of resampling, the YUV422 format is converted into YUV444. The YUV422-to-YUV444 processing is bypassed in the color space conversion unit.

If the active range for the luminance samples (Y) is [16235] and [16240] for the chrominance samples (Cb and Cr), the values of R, G, and B output components are clipped to the range [0:255]. The equation shown in Figure 15-75 gives the 11-bit coefficients of the YCbCr to RGB color space conversion.

Figure 15-75. YCbCr to RGB Registers (VIDFULLRANGE=0)

$$\begin{bmatrix} R \\ G \\ B \end{bmatrix} = \frac{1}{256} * \begin{bmatrix} RY & RCr & RCb \\ GY & GCr & GCb \\ BY & BCr & BCb \end{bmatrix} * \begin{bmatrix} Y - 16 \\ Cr - 128 \\ Cb - 128 \end{bmatrix}$$

dss-E063

If the active range for the luminance samples (Y) and chrominance samples (Cb and Cr) is [0:255], the values of R, G, and B output components are clipped to the range [0:255]. The equation shown in Figure 15-76 gives the 11-bit coefficients of the YCbCr-to-RGB color space conversion.

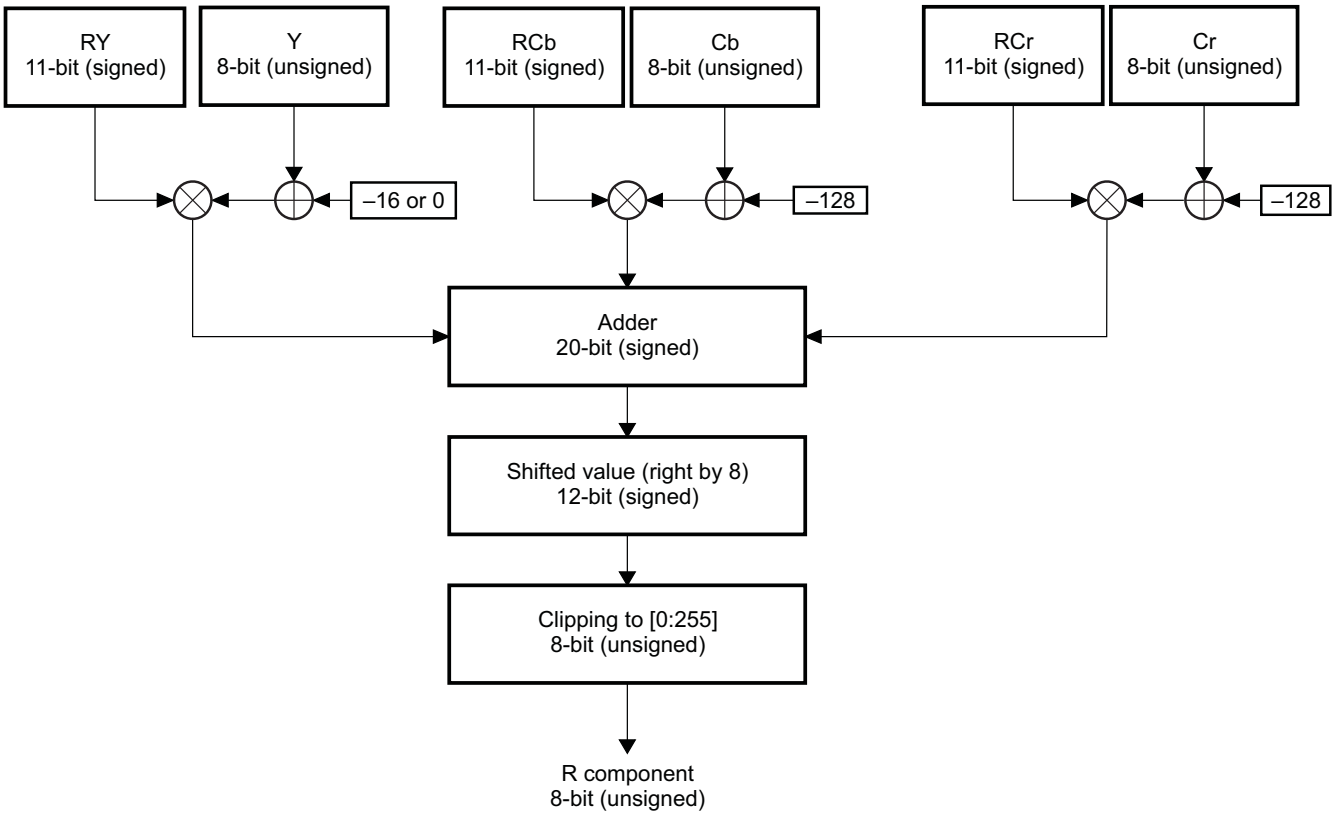
Figure 15-76. YCbCr to RGB Registers (VIDFULLRANGE=1)

$$\begin{bmatrix} R \\ G \\ B \end{bmatrix} = \frac{1}{256} * \begin{bmatrix} RY & RCr & RCb \\ GY & GCr & GCb \\ BY & BCr & BCb \end{bmatrix} * \begin{bmatrix} Y \\ Cr - 128 \\ Cb - 128 \end{bmatrix}$$

dss-E064

Figure 15-77 describes the computation for the calculation of the R component. The same computation applies for the G and B components:

Figure 15-77. Color Space Conversion Macro-Architecture



15.4.2.3.3 Hardware Cursor

The video layer can be used to display the hardware cursor. The encoded pixel data for the cursor image are in RGB12, RGB16 or RGB24 formats and the color space conversion block is bypassed. The transparency color key can be used when a non rectangle shape is used.

The alpha blending can be used to show a partial transparent cursor. When the alpha blender is enabled, the graphics layer is on top of the video layers. The cursor uses the graphics layer. The pixel alpha blending or the transparency color key can be used.

15.4.2.3.4 Up-/Down-Sampling

The video layer has a dedicated resizing block to upsample and downsample the video-encoded pixels. The supported input formats from memory are RGB24, RGB16, and YUV422

(RGB12 and all the alpha formats like ARGB and RGBA are not supported)

Users must set the right size and position of the original video before resizing for the upsampled/downsampled video to be inside the display screen boundaries.

The filtering applies on each component independently R, G, and B).

For the horizontal up-/downsampling, the equation is R component with five taps):

$$Rout(n) = \left(\sum_{i=-2}^{i=2} Ci(\Phi) \times Rin(n+i) \right) >> 7 \quad \text{(15-1)}$$

dss-E066

For the vertical up-/downsampling, the equation is R component with three taps):

$$Rout(n) = \left(\sum_{i=-1}^{i=1} Ci(\Phi) \times Rin(n+i) \right) >> 7$$ dss-E067 (15-2)

For the vertical up-/downsampling, the equation is R component with five taps):

$$Rout(n) = \left(\sum_{i=-2}^{i=2} Ci(\Phi) \times Rin(n+i) \right) >> 7$$ dss-E068 (15-3)

Rout: R component output

$Ci(\Phi)$ dss-E069: FIR filter coefficients

Rin: R component input

The pixel (n + 1) is older than pixel (n). The line (n + 1) is older than line (n).

Note: The coefficients Ci() depend on the phase between input and output pixels.

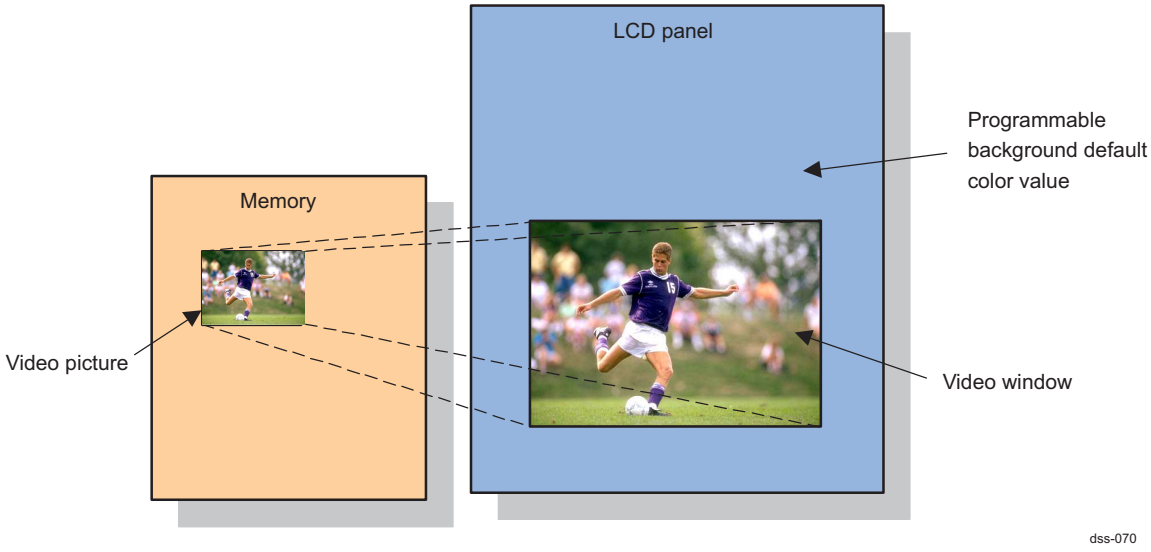
Note: If the 5-tap resizer is used for RGB16 and YUV422 picture formats, the width of the input picture must be a multiple of 2 pixels and more than 5 pixels.

DISPC_VIDn_ATTRIBUTES[21] VIDVERTICALTAPS == 1

DISPC_VIDn_PICTURE_SIZE[10:0] VIDORGSIZEX > 4 and even

Figure 15-78 shows an example of video upsampling.

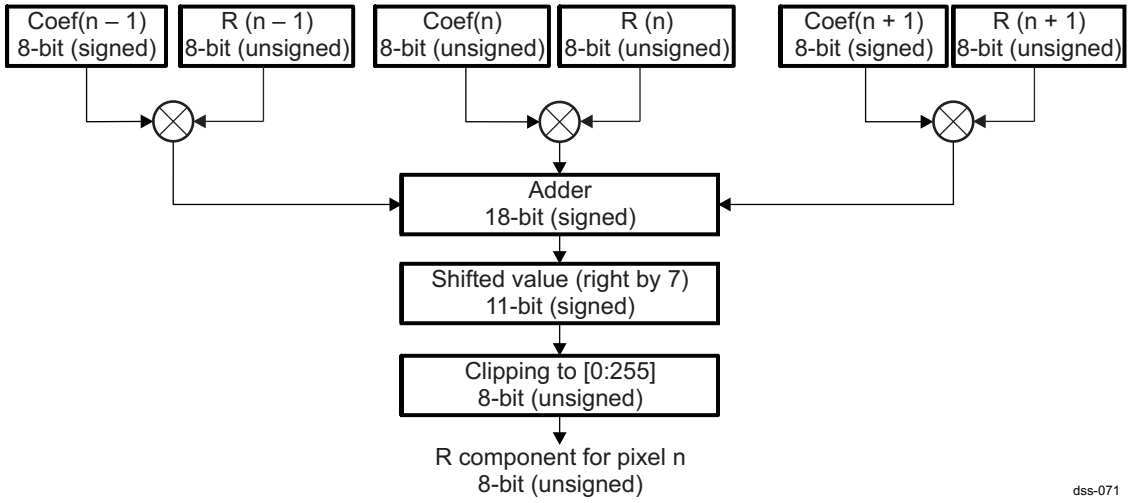
Figure 15-78. Video Upsampling



Filter Description

The up/downsampling filter is a poly-phase filter with five taps and eight phases for the horizontal filter and a programmable number of taps (three or five) and eight phases for vertical filter. The upsampling ratio is up to x8. The downsampling ratio using 3-tap configuration is/2. The downsampling ratio using 5-tap configuration is/4. The vertical filter is first applied to the encoded input pixel data; and then the horizontal filter is applied on the resulting pixel values to generate the output pixel values. Figure 15-79 shows the computation for the R component in the case of three coefficients (vertical filtering). The same computation applies to the G and B components.

Figure 15-79. Resampling Macro-Architecture (3-Coefficient Processing)



To determine if the minimum functional clock matches the down sampling ratio and the desired Pixel clock, the following formula must be used in conjunction with Table 15-26 and Table 15-27.

Ratio V when performing a vertical down-sampling only

$$h_ratio = \frac{DISPC_SIZE_LCD.PPL}{DISPC_VID_SIZE.VidSizeX}$$

$$v_ratio = \frac{DISPC_VID_PICTURE_SIZE.VidOrgSizeY}{DISPC_VID_SIZE.VidSizeY}$$

$$Ratio = \frac{v_ration}{2 \times h_ratio} \qquad If \quad 1 < v_ratio \le 2$$

$$Ratio = \max\left(\frac{v_ration}{2 \times h_ratio}, \frac{v_ration - 2}{2 \times (h_ratio - 1)}\right) \qquad If \quad 2 < v_ratio \le 4$$

dss_swpu108-E135

Note: For frequency ratio calculation on the TV output, it is correct to replace DISPC_SIZE_LCD with DISPC_SIZE_DIG.

When the down-sampling ratio is below 0.5, it is not possible to use a video in full screen.

Ratio H when performing a horizontal down-sampling only

$$Ratio = \frac{DISPC_VID_PICTURE_SIZE.VidOrgSizeX}{DISPC_VID_SIZE.VidSizeX}$$

dss_swpu108-E136

Ratio H+V when performing a horizontal and vertical down-sampling

Ratio = max (horizontal Ratio, vertical Ratio) as previously defined.

Table 15-26. Functional Clock Frequency Requirement in RGB16 & YUV422—Active Matrix Display

Minimum Functional Clock (MHz)		Horizontal Resampling				
		Off	Up	1:1 – 1:2	1:2 – 1:3	1:3 – 1:4
Vertical Resampling	Off	AxPCLK	AxPCLK	2xPCLK	3xPCLK	4xPCLK
	Up	AxPCLK	AxPCLK	2xPCLK	3xPCLK	4xPCLK
	3-tap 1:1 to 1:2	2xPCLK	2xPCLK	4xPCLK	6xPCLK	8xPCLK
	5-tap 1:1 to 1:4	RatioxPCLK	RatioxPCLK	RatioxPCLK	RatioxPCLK	RatioxPCLK

With A = 1 in case all the data and synchronization signals are asserted and deasserted on the rising edge of the PCLK; otherwise, A = 2.

Table 15-27. Functional Clock Frequency Requirement in RGB24—Active Matrix Display

Minimum Functional Clock (MHz)		Horizontal Resampling				
		Off	Up	1:1 – 1:2	1:2 – 1:3	1:3 – 1:4
Vertical Resampling	Off	AxPCLK	AxPCLK	2xPCLK	3xPCLK	4xPCLK
	Up	AxPCLK	AxPCLK	2xPCLK	3xPCLK	4xPCLK
	3-tap 1:1 to 1:2	2xPCLK	2xPCLK	4xPCLK	6xPCLK	8xPCLK
	5-tap 1:1 to 1:4	RatioxPCLK	RatioxPCLK	2xRatioxPCLK	2xRatioxPCLK	2xRatioxPCLK

With A = 1 in case all the data and synchronization signals are asserted and deasserted on the rising edge of the PCLK; otherwise, A = 2.

Use case example:

An input picture of 1024*768 is scaled to an output picture of size of 800*600 and displayed onto a LCD of resolution1280*768 at a PCLK of 74.25 MHz with a DSS functional clock of 133 MHz.

In this example, a H+V down-sampling is done on the input picture. Firstly the Ratio V and H are determined and the resulting maximum value is taken to calculate the functional clock frequency required.

Ratio V: h_ratio = 1.6 and v_ratio = 1.28 then Ratio = 0.4

Ratio H: Ratio = 1.28

Ratio H+V: Ratio = max (1.28, 0.4) = 1.28

In this use case, the horizontal and vertical down sampling range are 1:1–1:2. The 3-tap or 5-tap configuration can be taken into consideration. Therefore, from Table 15-26 and Table 15-27, If in RGB16-YUV422:

- 3-taps → DSS functional clock = 4 * PCLK = 297 MHz
- 5-taps → DSS functional clock = Ratio * PCLK = 95.36 MHz

If in RGB24,

- 3-taps → DSS functional clock = 4 * PCLK = 297 MHz
- 5-taps → DSS functional clock = 2 * Ratio * PCLK = 190.72 MHz

In this use case, the pixel format supported is RGB16-YUV422 in a 5-tap configuration.

15.4.2.4 Overlay Support

CAUTION

Enabling overlay optimization (setting the DSS.DISPC_CONTROL [12] OVERLAYOPTIMIZATION bit) if no overlay region effectively exists (the DSS.DISPC_VIDn_ATTRIBUTES [0] VIDENABLE bit is cleared, with n = 1, 2) leads to unpredictable behavior. The overlay optimization feature must be enabled only when an overlay area exists. Before enabling the overlay optimization, the DSS.DISPC_GFX_WINDOW_SKIP[31:0] GFXWINDOWSKIP bit field must be first set according to the video1 and graphics windows overlap.

The overlay mechanism consists of displaying more than one layer (graphics and video layers) using rules based on priority and transparency color keys.

When the pixel format is ARGB or RGBA, the color key match logic uses only the RGB value defined by ARGB or RGBA. The alpha blending factor is ignored.

The overlay managers are based on the same rules for priority and transparency color keys (see Figure 15-82).

Each data pipeline is assigned a single overlay related to a single display controller output.

The overlay manager is connected to all three outputs of the pipelines (graphics, video1 and video2). The output of the LCD overlay manger is connected to the Spatial/Temporal Dithering, and Passive Matrix units and back to the palette unit in the case of Gamma correction.

15.4.2.4.1 Priority Rule

The overlay manager can be configured in two distinct modes:

- Alpha mode (only source color key with the graphics layer)
- Normal mode (no alpha support)

The following rules apply in normal mode:

The video1 layer is always on top of the graphics layer. The video2 layer is always on top of the video1 and graphics. The display controller reads the data for each buffer from the system memory and, depending on the transparency color key values, displays either the pixels in the video layer, the pixels in the graphics layer, or the solid background color.

Each layer can have any size up to full-display screen. If there are no graphics or video-encoded pixels at a specific position, the programmable, solid background color appears (see Figure 15-84).

Figure 15-80. Overlay Manager in Normal Mode

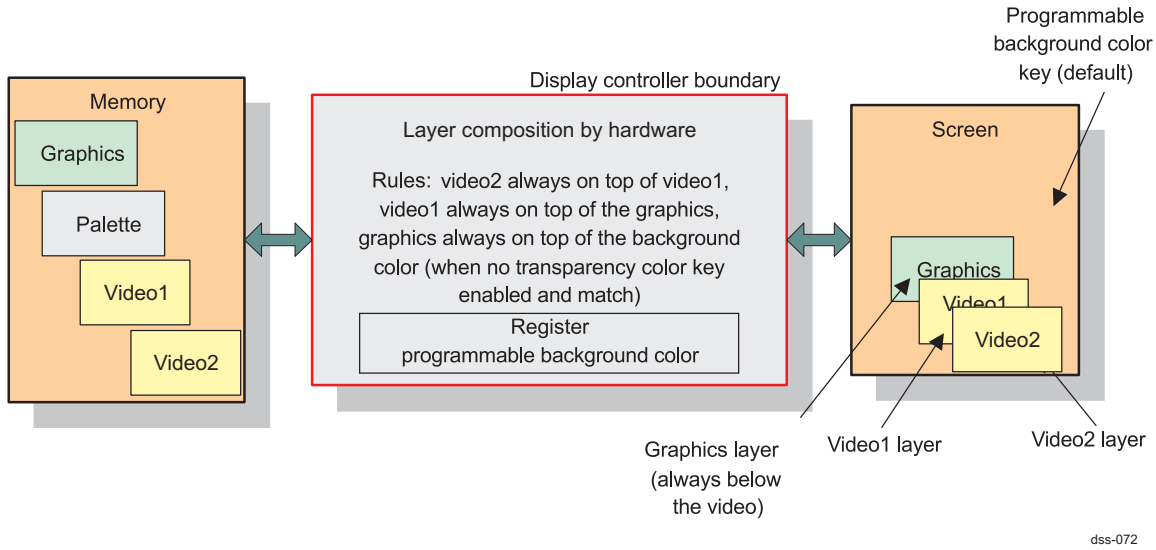
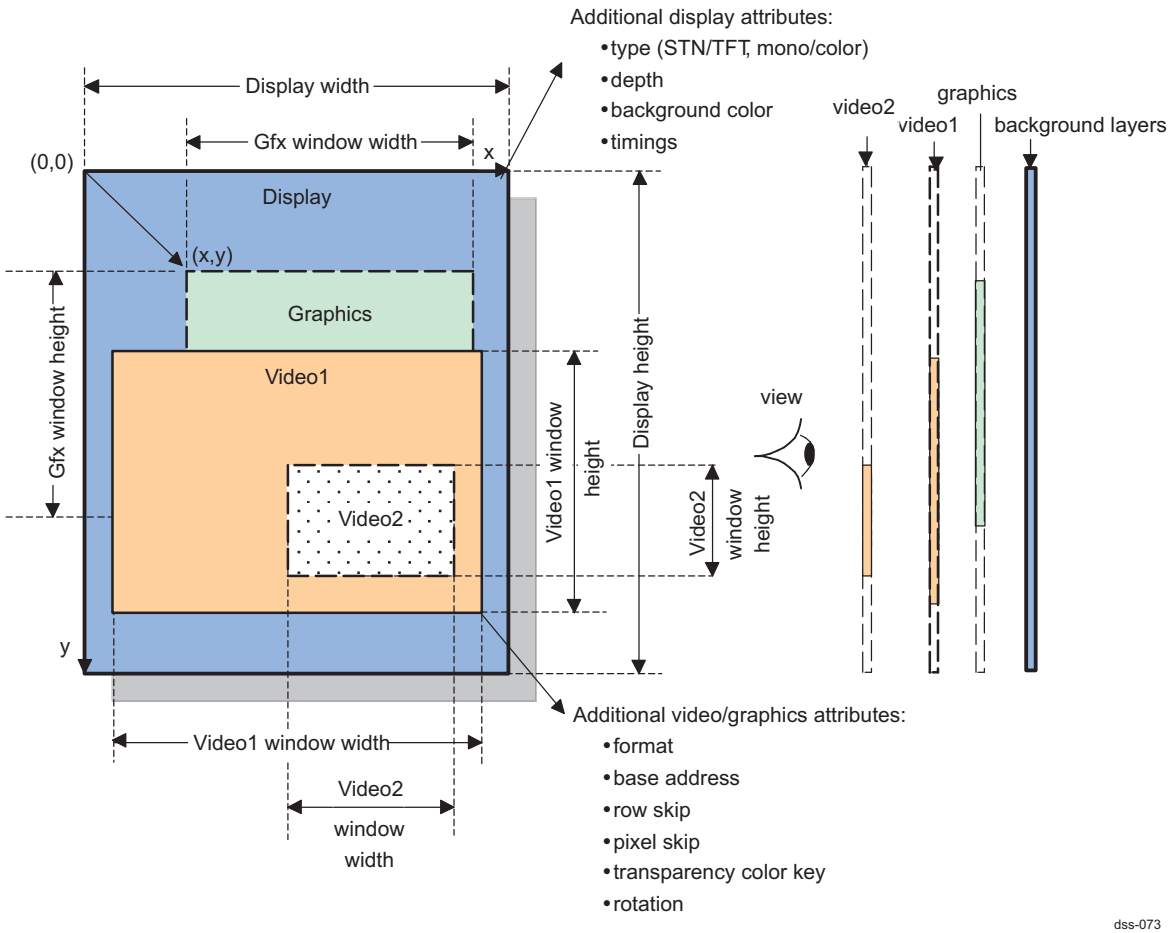


Figure 15-81. Display Attributes in Normal Mode



The following rules apply in alpha mode:

The video2 layer is always on top of the video1 layer. The graphics layer is always on top of the video1 and video2. The display controller reads the data for each buffer from the system memory and, depending on the transparency color key values, displays either the pixels in the video layer, the pixels in the graphics layer, or the solid background color.

Each layer can have any size up to full-display screen. If there are no graphics or video-encoded pixels at a specific position, the programmable, solid background color appears (see Figure 15-84).

Figure 15-82. Overlay Manager in Alpha Mode

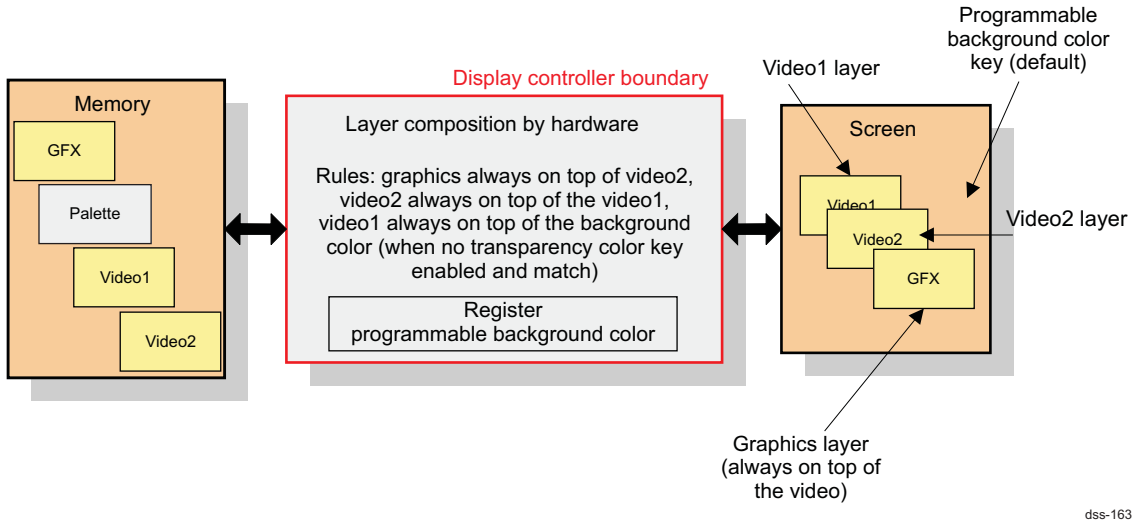


Figure 15-83. Display Attributes in Alpha Mode

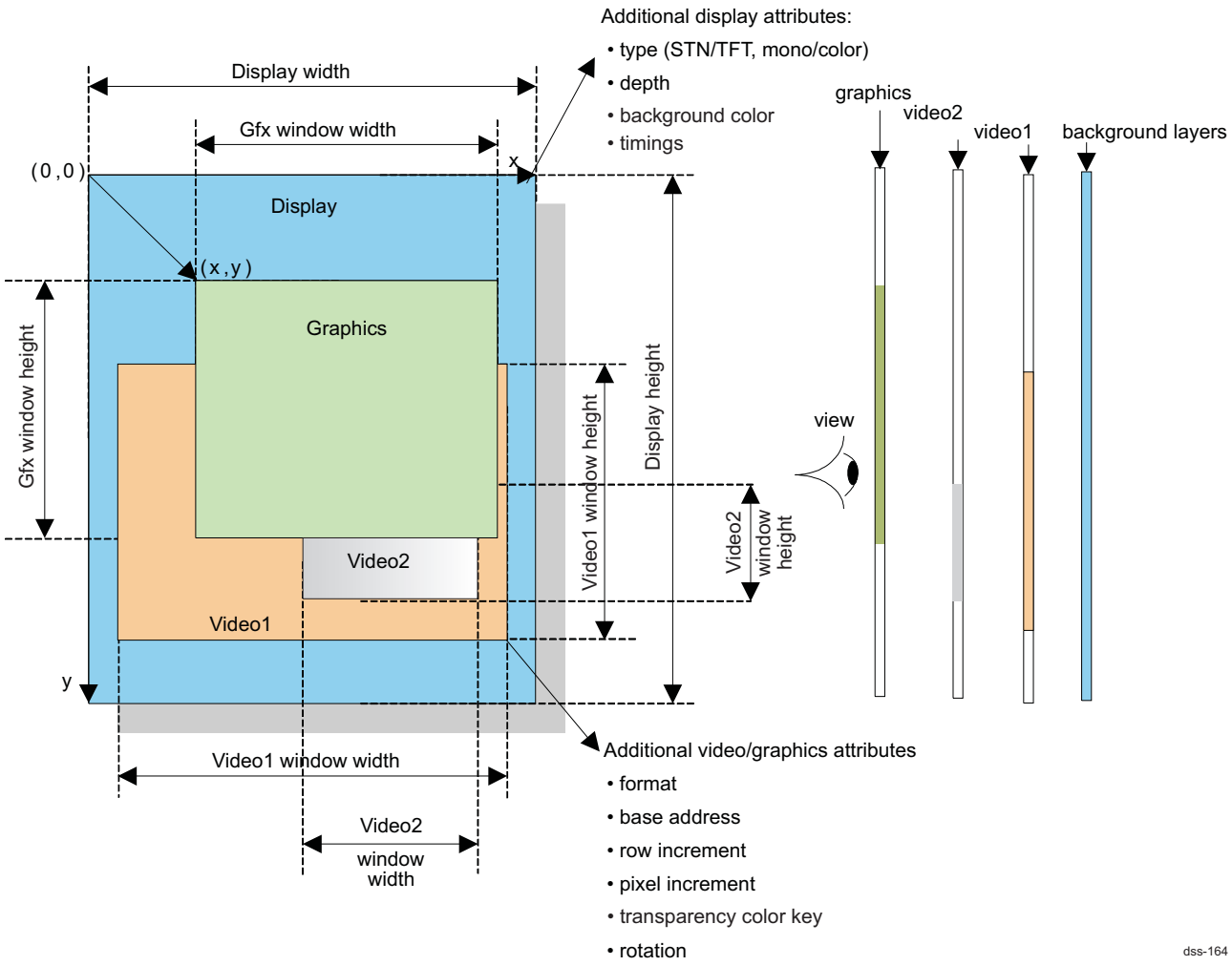
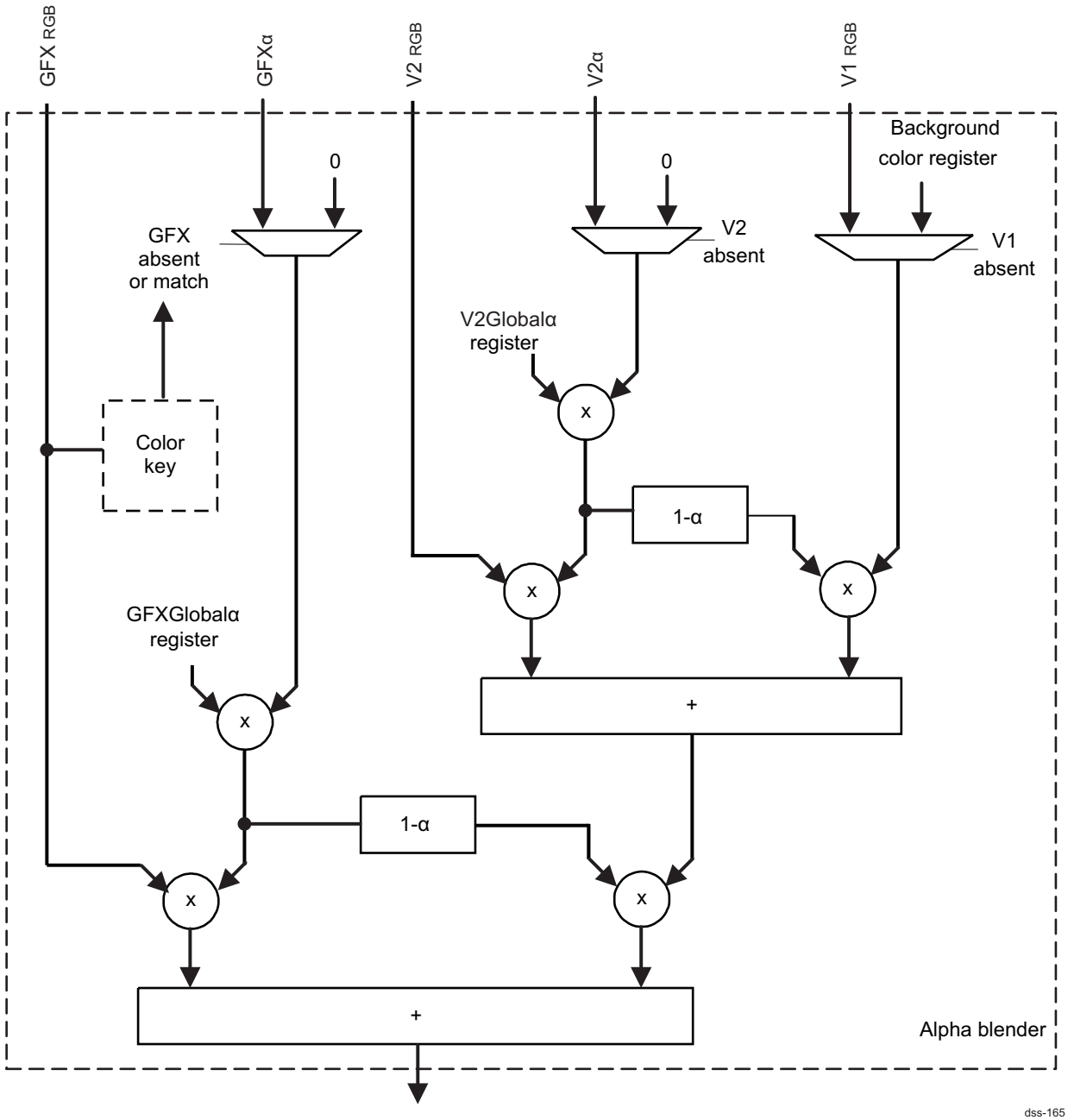


Figure 15-84 shows the alpha blending processing in detail.

Figure 15-84. Alpha Blending Macro Architecture



Note: "1-alpha" operator corresponds to the basic 1's complement operation.

The alpha blending value is defined by the pixel value (ARGB or RGBA formats). A global alpha blending value can be defined and used in combination with the pixel alpha blending value. If the pixel format contains no alpha blending value, the pixel alpha value is considered to be 0xFF.

In case of ARGB-444, the alpha blending is defined using a 4-bit value. It is converted into an 8-bit value by duplicating the 4-bit value. Table 15-28 details the alpha blending 4-bit values and the corresponding blending percentage.

Table 15-28. Alpha Blending 4-Bit Values

Alpha Blending 4-Bit Value (ARGB-444)	Alpha Blending 8-Bit Value (Converted Value)	% Blending
0x0	0x00	100% (transparent)
0x1	0x11	93.33%
0x2	0x22	86.6%
...	...	...
0xE	0xEE	6.6%
0xF	0xFF	0% (opaque)

15.4.2.4.2 Transparency Color Keys

15.4.2.4.2.1 Normal Mode

This section describes the features available in normal mode.

The two transparency color keys are the video source transparency color key and the graphics destination transparency color key. The encoded pixel color value is compared to the transparency color key. For CLUT bitmaps, the palette index is compared to the transparency color key and not to the palette value pointed out by the palette index.

Note: The video source transparency color key and graphics destination transparency color key cannot be active at the same time.

- Video source transparency color key value:

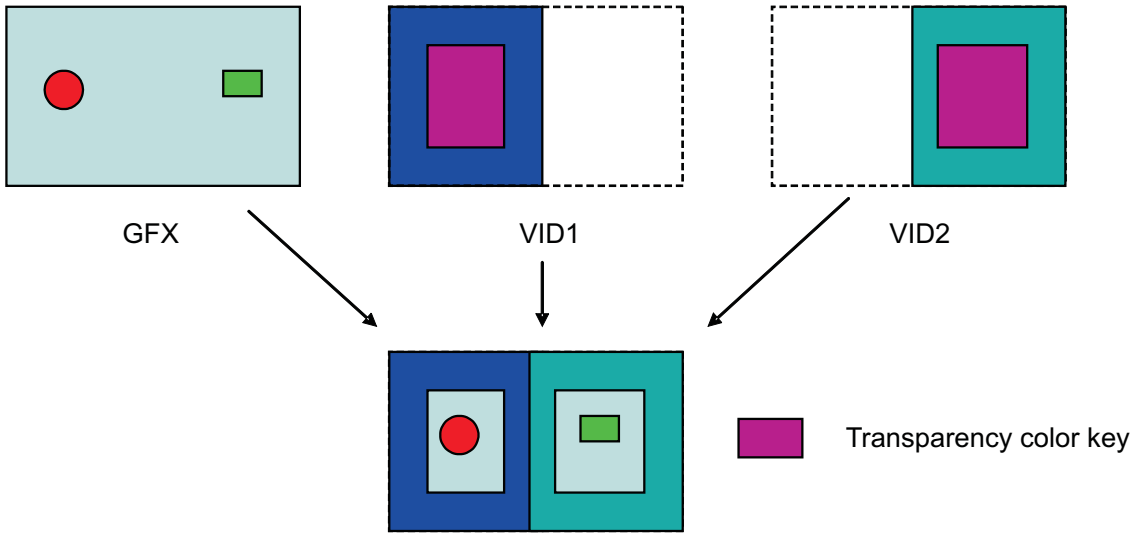
  The video source transparency color key value defines the encoded pixel data considered as the transparent pixel. The encoded pixel values with the source color key value are pixels not visible on the screen, and the underlayer encoded pixel values or solid background color are visible.

  The video source transparency color key can be used only if the color space conversion and the up-/down-scaling modules are disabled. The format of the data is RGB 16. (This feature handles the hardware cursor displayed by one of the video layers.)

  To enable the video source transparency color key, set to 0x1 the DSS.DISPC_CONFIG[11] TCKLCDSELECTION bit for LCD output or the DSS.DISPC_CONFIG[13] TCKDIGSELECTION bit for digital output. Program the DSS.DISPC_CONFIG[10] TCKLCDENABLE bit (LCD output) or the DSS.DISPC_CONFIG[12] TCKDIGENABLE bit (digital output) to enable or disable the transparency color key.

  An example is shown in Figure 15-85: The video source transparency is applied on video1 (VID1) and video2 (VID2) layers. The pixels with the transparency color key are not displayed; instead, underlying layers are shown.

Figure 15-85. Video Source Transparency Example



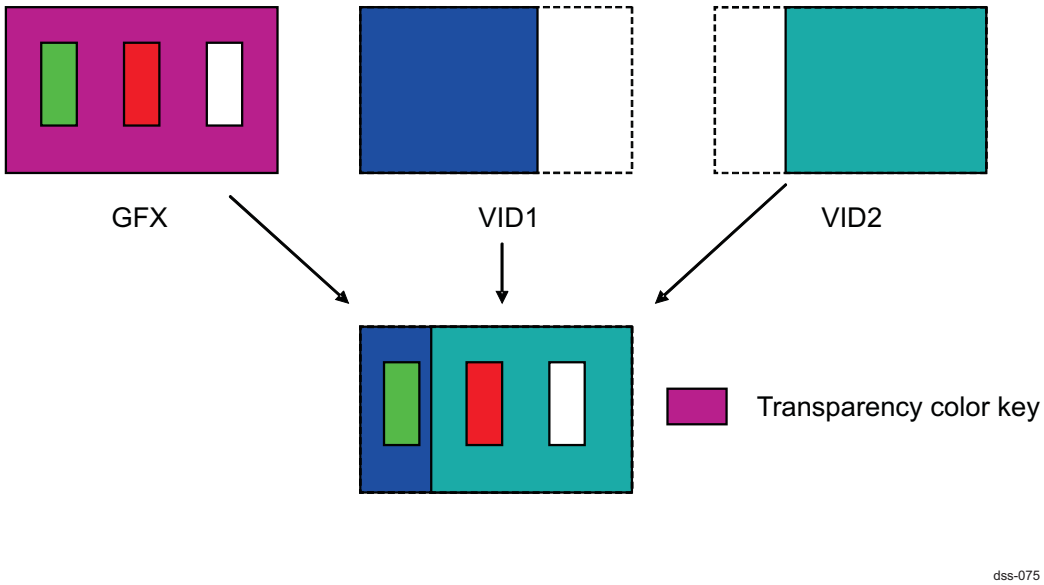
- Graphics destination transparency color key value:

  The graphics destination transparency color key value defines the encoded pixels in the video layers to be displayed. The encoded pixel values with the destination color key value are pixels not visible on the screen and the pixels different from the transparency color key are displayed over the video layers. The destination transparency color key is applicable only in the graphics region when graphics and video overlap; otherwise, the destination transparency color key is ignored.

  To enable the graphics destination transparency color key, set to 0x0 the DSS.DISPC_CONFIG[11] TCKLCDSELECTION bit for LCD output or the DSS.DISPC_CONFIG[13] TCKDIGSELECTION bit for digital output. Program the DSS.DISPC_CONFIG[10] TCKLCDENABLE bit (LCD output) or the DSS.DISPC_CONFIG[12] TCKDIGENABLE bit (digital output) to enable or disable the transparency color key.

  An example is shown in Figure 15-86: The destination transparency is applied on graphics (GFX) layer and the pixels without the transparency color key are displayed over the overlying layers.

Figure 15-86. Graphics Destination Transparency Example



15.4.2.4.2.2 Alpha Mode

This section describes the features available in alpha mode.

Only the graphics source transparency color key is available The encoded graphics pixel color value is compared to the transparency color key. The encoded pixel values with the source transparency key are not visible and the under-layer encoded pixel values or solid background color are visible. To enable the graphics source transparency color key, set to 0x0 the DSS.DISPC_CONFIG[11] TCKLCDSELECTION bit for LCD output or the DSS.DISPC_CONFIG[13] TCKDIGSELECTION bit for digital output. Program the DSS.DISPC_CONFIG[10] TCKLCDENABLE bit (LCD output) or the DSS.DISPC_CONFIG[12] TCKDIGENABLE bit (digital output) to enable or disable the transparency color key. In the case of CLUT bit maps, the palette index is compared to the transparency color key and not the palette value pointed out by the palette index.

15.4.2.4.3 Overlay Optimization (Only Available in Normal Mode)

The display controller can be configured to take advantage of the fact that the graphics pixels under video window 1 are not visible when the transparency color key is not used. The optimization can be selected to reduce the bandwidth used to fetch the pixels for graphics. The color key must be disabled. The graphics pixels under the video window 1 are not fetched from system memory. At least the video window 1 and the graphics window must be enabled. The following graphic formats are supported: RGB (RGB16 and RGB24 packed and unpacked), YUV422, and BITMAP 8. The formats BITMAP 1, 2, and 4 are not supported. The video format can be RGB (RGB16, RGB24 packed and unpacked, and YUV422 formats). The DMA engine does not fetch the unnecessary graphics pixels to avoid extra bandwidth use. Only visible pixels from graphics and video buffers in system memory are fetched and displayed by the display controller.

15.4.2.5 Active/Passive Matrix Display Data Path

For active matrix display data path, the following blocks are serial and each of them can be bypassed:

- Color phase rotation
- Spatial/temporal dithering
- Multiple cycle data format

For passive matrix display data path, the following blocks are serial and each of them can be bypassed:

- Color phase rotation
- Spatial/temporal dithering
- Passive matrix technology

15.4.2.5.1 Color Phase Rotation

The Color Phase Rotation (CPR) can be used to correct the LCD output colorimetry in case of non pure white backlight.

The color phase rotation can be selected for passive matrix and active matrix panel. The logic is integrated after the LCD overlay manager or the palette while using the gamma correction and before the spatial/temporal dithering. The color phase rotation can be selected to correct the nonpure white backlight of the LCD module by using a programmable matrix to convert the 24-bit RGB pixel value into a new 24-bit RGB pixel value. The matrix is programmed through a set of nine 10-bit signed coefficients. The output of the calculation is clipped to [0:255]. The color phase rotation is processed by the equation shown in Figure 15-87.

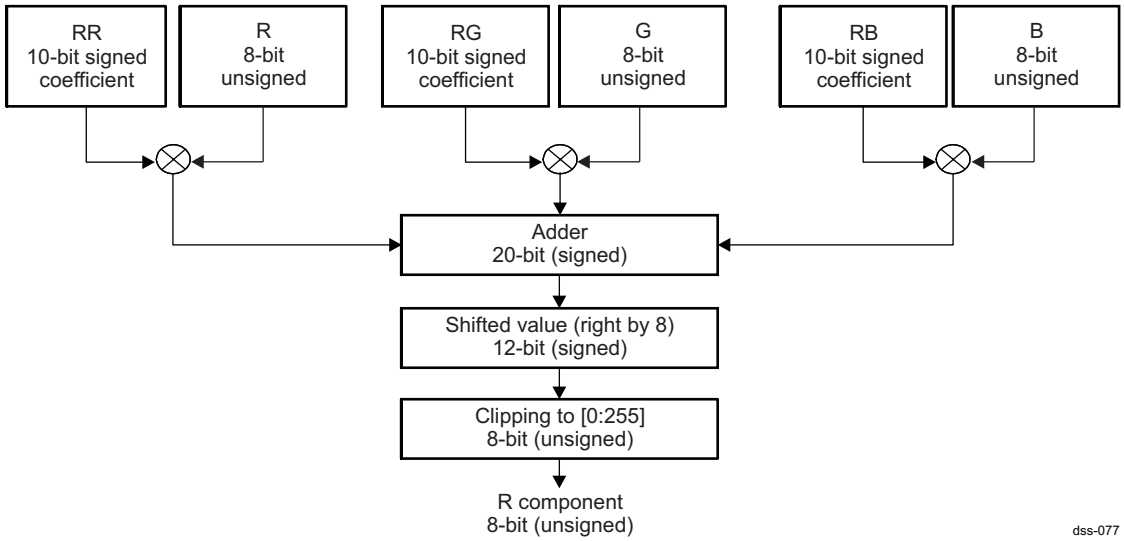
Figure 15-87. Color Phase Rotation Matrix

$$\begin{bmatrix} Rout \\ Gout \\ Bout \end{bmatrix} = \frac{1}{256} * \left(\begin{bmatrix} RR & RG & RB \\ GR & GG & GB \\ BR & BG & BB \end{bmatrix} * \begin{bmatrix} Rin \\ Gin \\ Bin \end{bmatrix} \right)$$

dss-E076

Figure 15-88 shows the color phase rotation macro-architecture.

Figure 15-88. Color Phase Rotation Macro Architecture



15.4.2.5.1.1 Spatial/Temporal Dithering

The spatial/temporal dithering logic can be selected for passive matrix and active matrix panel. The dithering logic is integrated after the color phase rotation and before the TDM and passive matrix units. The spatial/temporal dithering logic can be selected to enhance the quality of the passive matrix and active matrix outputs. The dithering logic can process the pixels over a single frame, two frames, or four frames. In the case of a single frame, only spatial processing is applied. In the case of multiple frames, spatial and temporal processing is applied to the pixels.

- Passive Matrix Technology: The passive matrix display dithering logic path is used. The spatial/temporal dithering logic can be selected. When selected, the pixels are preprocessed by the spatial/temporal dithering logic before the passive matrix display dithering logic. The output format of the spatial/temporal dithering logic is RGB 12-bit (not configurable).
- Active Matrix Technology: The encoded pixel values are used by spatial/temporal dithering logic to display the data in a lower color depth on the LCD panel. The spatial/temporal dithering algorithm is based on the (x,y) pixel position, the value of removed bits and the frame number. The picture quality is improved when enabling the spatial/temporal dithering logic. When spatial/temporal dithering is not enabled, the three MSBs of the pixel color components are output on the interface data bus if the interface data bus is smaller than the pixel format size. If the interface data bus is wider than the pixel format size, by programming the pixel components replication active/inactive, the MSB is replicated to the LSB of the interface data bus or the LSB is filled with 0s.

15.4.2.5.2 Passive Matrix Display Dithering Logic

- Passive matrix technology

  After the graphics data are merged with the video data from the video layers depending on the transparency status, the result is sent to the color/grayscale space-/time-based dither generator. The monochrome data and each RGB color component are encoded on 4 bits, which are the 4 MSBs of

the pixel-encoded component 8-bit value defined by the merge of the graphics data and the video data.

These 4-bit values are used to select on the 16 intensity levels. The gray/color intensity is controlled by turning individual pixels on and off at varying period rates, making the average time the pixel is off longer than the average time the pixel is on, thus producing more intense grays/colors. The dithering generator also uses the intensity of adjacent pixels in the calculation to give the screen image a smooth appearance. The proprietary dither algorithm is optimized to provide a range of intensity values that matches the visual perception of color/gray graduations.

- Active matrix technology

  The passive matrix dithering logic is always bypassed in active displays.

Note: If the interface data bus is smaller than the pixel format size, dithering logic can be enabled. If the interface data bus is wider than the pixel format size, the dithering logic cannot be enabled and replication feature can be used.

15.4.2.5.3 Passive Matrix Display Output FIFO

- Passive matrix technology

  The display controller contains a 2-entry by 8-bit-wide output FIFO used to store pixel data before it is driven out to the LCD pins. Each time a modulated pixel value is output from the dither generator, it is placed into a serial shifter. The shifter can be configured to be 4 or 8 bits wide. Single-panel monochrome screens use either four or eight data lines; single-panel color screens use eight data pins.
- Active matrix technology

  The output FIFO is bypassed in active matrix mode.

15.4.2.5.4 Multiple Cycle Output Format

The pixels after the active matrix display processing are formatted on one or multiple cycles (from one to three cycles). The interface width can be 8-, 9-, 12-, or 16-bit. On three cycles, two pixels can concatenate and send to the panel. When the TDM is disabled, the display controller outputs the pixels using the conventional formats: Passive matrix display/active matrix display monochrome/color.

The following example shows an output configuration based on the interface width (8-bit) and the pixel format output (24-bit) (also see Table 15-29):

- The DSS.DISPC_CONTROL[24:23] TDMCYCLEFORMAT bit field is set to 0x2 (three cycles for one pixel).
- The DSS.DISPC_DATA_CYCLEk (k=0) register is set to 0x00000008 (8 bits from pixel 1 for the first cycle).
- The DSS.DISPC_DATA_CYCLEk (k=1) register is set to 0x00000008 (8 bits from pixel 1 for the second cycle).
- The DSS.DISPC_DATA_CYCLEk (k=2) register is set to 0x00000008 (8 bits from pixel 1 for the third cycle).

Table 15-29. 8-Bit Interface Configuration/24-Bit Mode

	24-Bit Mode		
	1st Cycle	2nd Cycle	3rd Cycle
Data[7]	R0[7]	G0[7]	B0[7]
Data[6]	R0[6]	G0[6]	B0[6]
Data[5]	R0[5]	G0[5]	B0[5]
Data[4]	R0[4]	G0[4]	B0[4]
Data[3]	R0[3]	G0[3]	B0[3]
Data[2]	R0[2]	G0[2]	B0[2]
Data[1]	R0[1]	G0[1]	B0[1]

Table 15-29. 8-Bit Interface Configuration/24-Bit Mode (continued)

Data[0]	R0[0]	G0[0]	B0[0]

15.4.2.6 Video Line Buffer

The line buffer size is 1024 x 24-bit. There are six line buffers (1024 x 24-bit) that can be merged into three lines (2048 x 24-bit). Table 15-30 lists the maximum width depending on the TAP configuration and the pixel format.

Table 15-30. Maximum Width Allowed

Vertical Tap	Pixel Format	Maximum Width (Pixels)
3	RGB16	2048
	RGB24	
	YUV422	
5	RGB16	1024
	RGB24	
	YUV422	

15.4.2.7 Synchronized Buffer Update

A synchronization mismatch between the frame buffer and the display refreshes, named tearing effect, can lead to images that appear to be stretched on the screen. To avoid this, a synchronization mechanism is needed between the display controller and the process that updates the buffer. An interrupt is generated when the display reaches a predefined line number. This PROGRAMMEDLINENUMBER interrupt is a level signal and stays active during the programmed line of the display.

15.4.2.8 Rotation

In case of SDRAM buffer, the display controller accesses the encoded pixels in burst, always considering the consecutive data in memory. The rotation engine (VRFB) in the SDRAM scheduler (SDRC) is in charge of translating the addresses from virtual to physical SDRAM addresses (see Chapter 11, *Memory Subsystem*).

Note: It is highly recommended to use the VRFB Rotation engine when possible and not the display subsystem DMA engine for rotating frame buffer to have a good performance in the L3 interconnect and SDRAM memory efficiency.

The rotation using the SMS-VRFB rotation engine is supported for BITMAP8, RGB12 (16-bit container) and ARGB16, RGB16, ARGB16, RGB24 (using 32-bit container) and ARGB32 and RGBA32 and YUV422 (YUV2 and YUYV). The formats not supported are BITMAP1, BITMAP2, BITMAP4 and RGB24 (using 24-bit container).

15.4.2.9 Multiple Buffer Support

Users update the base address of the buffer when the update of the working buffer has finished and is ready to be displayed. The register that contains the base address of the buffer is a shadow register that is read by the hardware at the next Vertical Front Porch (VFP).

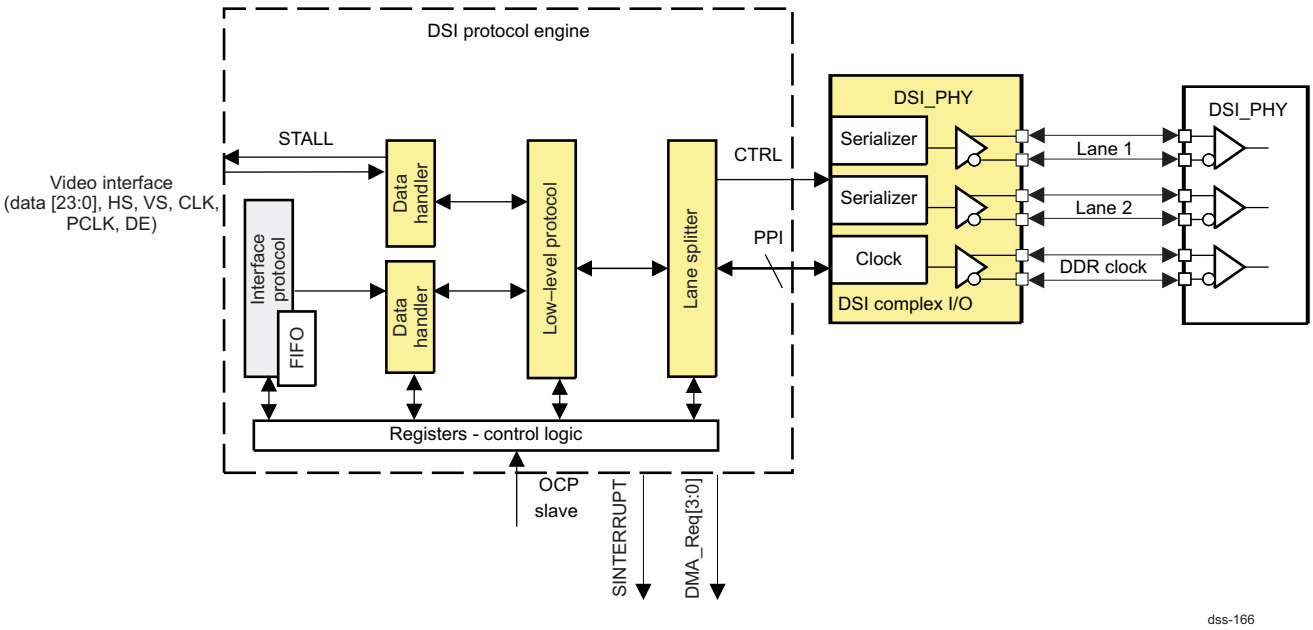
15.4.3 DSI Protocol Engine Functionalities

The DSI protocol engine integrates DSI interface to the display through the DSI DSI_PHY module, L4 interconnect interface and video interface from the display controller. The DSI DSI_PHY or complex I/O module is detailed in Section 15.4.5. The DSI Transmitter (Protocol Engine + PHY) port can be connected to multiple displays using a single DSI host port. The DSI protocol engine controls the DSI PLL Control module detailed in Section 15.4.4. The DSI Transmitter port can be used in video mode or/and command mode.

15.4.3.1 DSI Protocol Architecture

The DSI protocol engine receives data from the video port and/or the L4 interconnect slave port, encapsulates them with the VC ID, generates the ECC and check-sum, and splits the data into byte stream to the DSI_PHY to be sent using the low-speed (LS) or high-speed (HS) protocol. The DSI protocol engine receives data and acknowledge from the display using the same DSI link in case of bidirectional display. Multiple data streams can be interleaved to support multiple panels connected to the same host DSI port. Figure 15-89 details the DSI protocol engine architecture.

Figure 15-89. DSI Protocol Engine

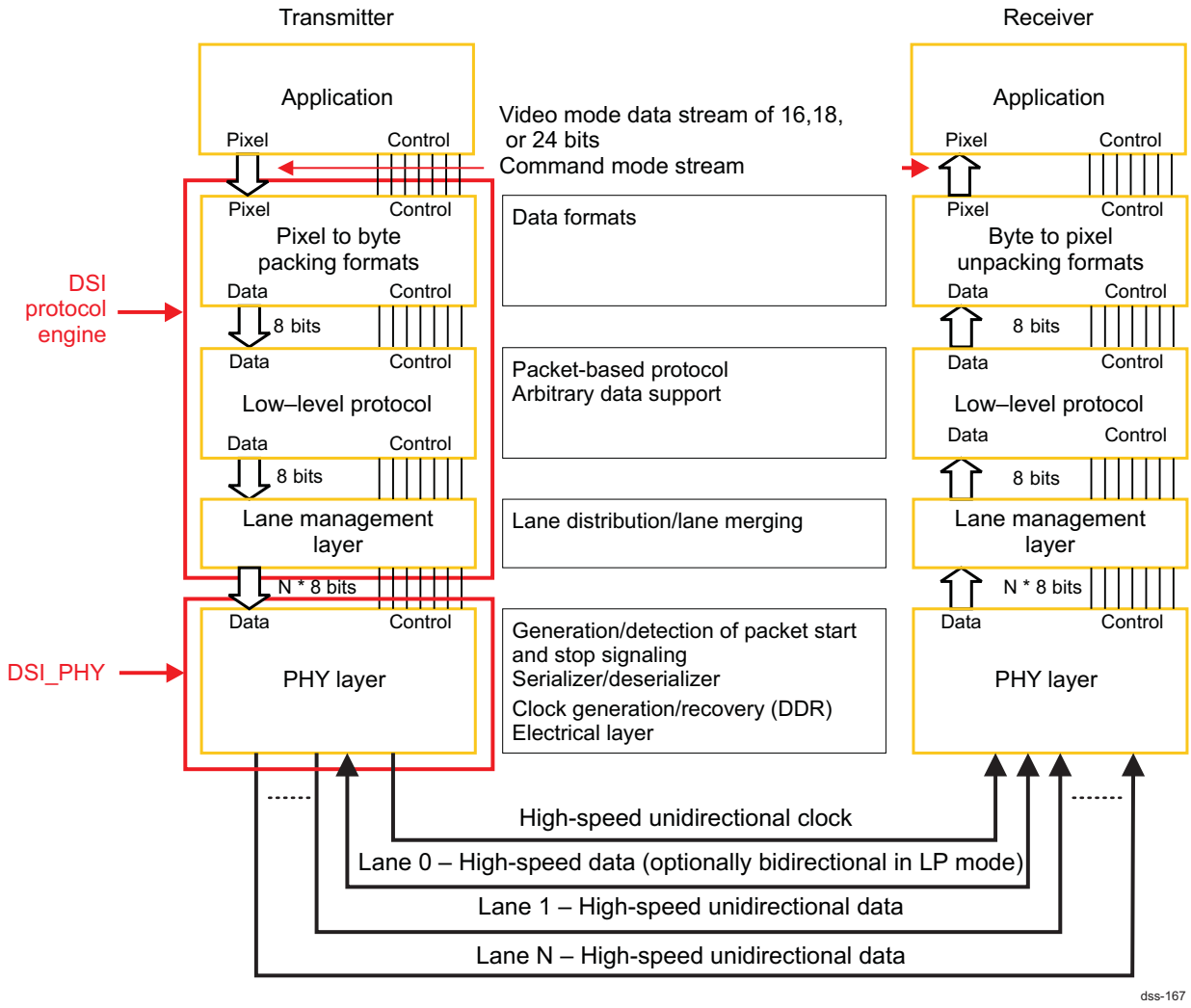


Note: The order of the PHY pairs (clock and data lanes) is informative. Each PHY pair can be Clock or Data. The DSI complex I/O receives the configuration for pin order and the differential +/- in a pair from the settings in DSS.DSI_COMPLEXIO_CFG1 register.

The DSI serial interface is a bidirectional differential serial interface with data/clock for the physical layer (configured in unidirectional link in case the display module is only unidirectional). The maximum DSI data transfer capacity is 800 Mbps per channel. The speed of the link can be software configured only when the DSI_PHY is in stop state or in ULPS.

Figure 15-90 shows the DSI transmitter/receiver high-level data flow.

Figure 15-90. DSI Transmitter/Receiver Data Flow



15.4.3.2 Clock Requirements

The serial clock generated by the DSI host and sent to the display can be a continuous clock. The clock lane supports clock transmission even there is no data to send for displays that require continuous clock. It is software programmed through the DSS.DSI_CLK_CTRL[13] DDR_CLK_ALWAYS_ON bit: This bit can be programmed only when the interface is disabled (that is, DSS.DSI_CTRL[0] IF_EN bit set to 0).

The peripheral can use two different kinds of clocks. The first one is the DDR clock provided on the clock lane. The second clock is some transitions on the data lane 1 even if there is no valid data to send using low power mode.

The LP clock (TxClkEsc) frequency provided to the DSI complex I/O is in the range of 67% to 150% of the peripheral Low-Power (LP) clock frequency. It is generated internally by the DSI protocol engine module using the DSI functional clock. The DSI functional clock is divided by 1, 2, 3, up to 8191 using the value programmed in the DSS.DSI_CLK_CTRL[12:0] LP_CLK_DIVISOR bit field. The LP clock generated from DSI functional clock should be in the range of 20 MHz down to 32 KHz. The duty cycle should be 50/50 (tolerance of 45/55 for maximum value). LP clock frequency visible on the pads (DP xor DN) is half the frequency of TxClkEsc.

The DSS.DSI_CLK_CTRL[20] LP_CLK_ENABLE bit is used to enable or disable the clock. When disabled, the value of DSS.DSI_CLK_CTRL[12:0] LP_CLK_DIVISOR bit field is ignored and does not have to be programmed by software users.

15.4.3.2.1 Timing Parameters for an LP to HS Transaction

Figure 15-91 shows the timing requirement when switching the data and clock lane state from LP to HS. Table 15-31 lists the LP to HS timing parameters.

Figure 15-91. LP to HS Timing

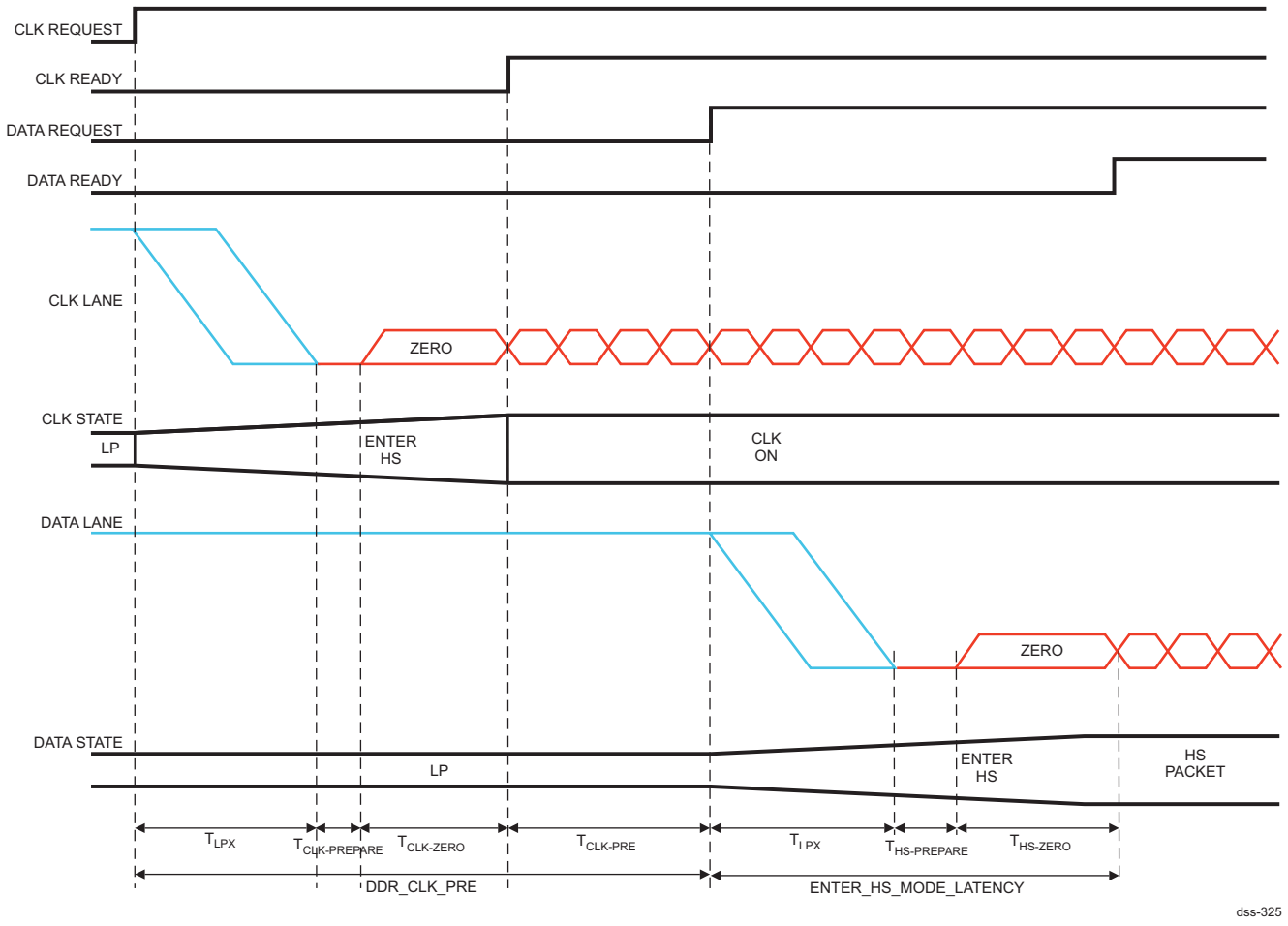


Table 15-31. LP to HS Timing Parameters

Timing	Description	Register
T_{LPX}	Length of any low-power state period. The value set in DSI_PHY_CFG1[22:16] TLPX_HALF bit field is half of the T_{LPX}.	DSI_PHY_CFG1[22:16] TLPX_HALF
$T_{CLK-PREPARE}$	Time to drive the CLK lane to LP-00 state, to prepare for HS clock transmission	DSI_PHY_CFG2[7:0] TCLK_PREPARE
$T_{CLK-ZERO}$	Time to drive the CLK lane to HS-0 state before starting the clock	DSI_PHY_CFG1[7:0]TCLK_ZERO
$T_{CLK-PRE}$	Time that the HS clock must be driven before any associated data lane begins the transition from LP to HS mode $T_{CLK-PRE}$ = DDR_CLK_PRE - T_{LPX} - $T_{CLK-PREPARE}$ - $T_{CLK-ZERO}$	See the MIPI DSI_PHY specification.
$T_{HS-PREPARE}$	Time to drive the data lane to LP-00 state, to prepare for HS packet transmission	DSI_PHY_CFG0[31:24] THS_PREPARE
$T_{HS-ZERO}$	Time to drive the data lane to HS-0 state before the synchronous sequence. $T_{HS-ZERO}$= DSI_PHY_CFG0[23:16] THS_PREPARE_THS_ZERO - $T_{HS-PREPARE}$	DSI_PHY_CFG0[23:16] THS_PREPARE_THS_ZERO

Table 15-31. LP to HS Timing Parameters (continued)

Timing	Description	Register
DDR_CLK_PRE	Time between the CLK lane request assertion and the data request assertion to switch the data lanes to HS	DSI_CLK_TIMING[15:8] DDR_CLK_PRE
ENTER_HS_MODE_LATENCY	Time to enter into HS mode. It is critical that ENTER_HS_MODE_LATENCY = 1 + DIVROUNDUP (2 * TLPX_HALF, 4) + DIVROUNDUP (THS_PREPARE ,4) + DIVROUNDUP (THS_PREPARE_THS_ZERO - THS_PREPARE + 3, 4) (1)	DSI_VM_TIMING7[31:16] ENTER_HS_MODE_LATENCY

(1) The formula DIVROUNDUP(value, div) is equivalent to ROUNDUP(value/div).

15.4.3.2.2 Timing Parameters for an HS to LP Transaction

Figure 15-92 shows the timing requirement when switching the state of the data and clock lanes from HS to LP. Figure 15-92 lists the HS to LP timing parameters.

Figure 15-92. HS to LP Timing

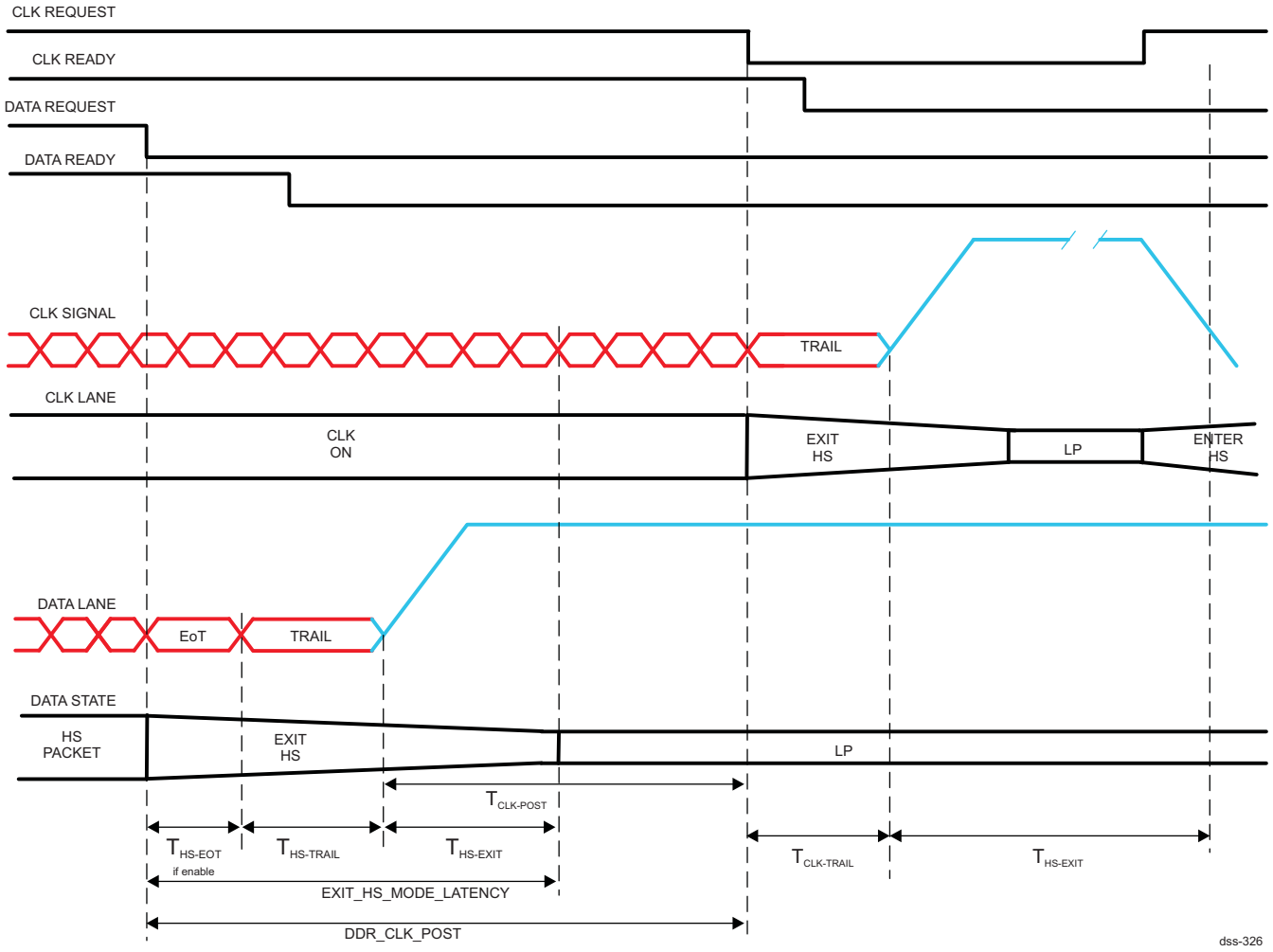


Table 15-32. HS to LP Timing Parameters

Timing	Description	Register
T_{HS-EOT}	If EoT is enabled, a delay is added to EXIT_HS_MODE_LATENCY to send the EoT packet. The EoT period depends on the number of data lanes: 1 data lane = 4 DDR clocks 2 data lanes = 2 DDR clocks	
$T_{HS-TRAIL}$	Time to drive flipped differential state after last payload data bit of a HS transmission burst	DSI_PHY_CFG0[15:8] THS_TRAIL
$T_{HS-EXIT}$	Time to drive data lane to LP-11 state, after HS burst	DSI_PHY_CFG0[7:0] THS_EXIT
$T_{CLK-POST}$	Time that the transmitter must continue sending HS clock after the last associated data lane has transitioned to LP mode	See the MIPI DSI-PHY specification.
$T_{CLK-TRAIL}$	Time to drive HS differential state after last payload clock bit of a HS transmission burst	DSI_PHY_CFG1[15:8]TCLK_TRAIL
DDR_CLK_POST	Time between the data lane request deassertion and the CLK request deassertion to switch the data lanes into LP mode. The DDR_CLK_POST value must follow the rule: DDR_CLK_POST $\geq T_{HS-TRAIL} + T_{HS-EOT} + T_{CLK-POST}$	DSI_CLK_TIMING[7:0] DDR_CLK_POST
EXIT_HS_MODE_LATENCY	Time to exit in HS mode. It is critical that EXIT_HS_MODE_LATENCY = DIVROUNDUP(($T_{HS-TRAIL} + T_{HS-EXIT}$),4) + 1 + T_{HS-EOT} [(1)]	DSI_VM_TIMING7[15:0] EXIT_HS_MODE_LATENCY

(1) The formula DIVROUNDUP(value, div) is equivalent to ROUNDUP(value/div).

15.4.3.2.3 Extra LP Transitions

Some DSI receivers require extra clock cycles in LP mode to process the data. The DSI protocol engine can be programmed to send automatically one NULL long packet. It applies only when no more data are ready to be sent from the internal FIFO to the peripheral on the last low speed transfer. The same value is used for all the VCs sending packets in low speed mode.

The size of the payload is defined by the DSS.DSI_CLK_CTRL[17:16] LP_CLK_NULL_PACKET_SIZE bit field. The header value depends on the VC ID and the size of the payload as detailed in Table 15-33 and Table 15-34.

Table 15-33. Extra NULL Packet Header

Virtual Channel ID	Payload size (DSI_CLK_CTRL[17:16] LP_CLK_NULL_PACKET_SIZE)	Header (1st Byte)	Header (2nd Byte): WC LSB	Header (3rd Byte): WC MSB	Header (ECC)
0x0	0	0x9	0x0	0x0	0x9
	1		0x1		0x13
	2		0x2		0x2F
	3		0x3		0x35
0x1	0	0x49	0x0		0x1F
	1		0x1		0x05
	2		0x2		0x39
	3		0x3		0x23
0x2	0	0x89	0x0		0x10
	1		0x1		0x0A
	2		0x2		0x36
	3		0x3		0x2C
0x3	0	0xC9	0x0		0x06
	1		0x1		0x1C
	2		0x2		0x20

Table 15-33. Extra NULL Packet Header (continued)

Virtual Channel ID	Payload size (DSI_CLK_CTRL[17:16] LP_CLK_NULL_PACKET_SIZE)	Header (1st Byte)	Header (2nd Byte): WC LSB	Header (3rd Byte): WC MSB	Header (ECC)
	3		0x3		0x3A

Table 15-34. Extra NULL Packet Payload

Payload size (DSI_CLK_CTRL[17:16] LP_CLK_NULL_PACKET_SIZE)	Payload (1st byte)	Payload (2nd byte)	Payload (3rd byte)	Payload (CRC) LSB	Payload (CRC) MSB
0	NA	NA	NA	0xFF	0xFF
1	0	NA	NA	0x87	0x0F
2	0	0	NA	0xB8	0xF0
3	0	0	0	0x33	0x39

Notes:

- In Table 15-33 and Table 15-34, both ECC and checksum are enabled.
- NA means not available.

15.4.3.3 DSI Transfer Modes

There are two transfer modes supported by the DSI module:

- Video mode (VM): Pixels are received from the video port, there are some real time constraints (pixels must be sent at the pixel frequency required by the display module) for sending the data to the display;
- Command mode (CM): Pixels can be received from the video port or from the L4 interconnect, there are no real time constraints except that TE should be avoided by starting the transfer at the right time during scan of the display and should be fast enough.

15.4.3.3.1 Video Mode

The video mode refers to the MIPI DPI standard. The sync events and pixels should be sent according to the display mode timings. Data are received from the video port. The display controller is in charge of fetching the data from the system memory and providing the data to the DSI protocol engine using the video port. The short packets used for the sync event are using precalculated 32-bit values. The long packets are constructed using the header defined in DSS.DSI_VCn_LONG_PACKET_HEADER registers.

15.4.3.3.2 Command Mode

The command mode refers to the MIPI DCS standard. The commands, parameters and pixels are sent to the display module with limited real time constraints (as defined in Section 15.4.3.3.1). The pixels can be provided on the video port by the display controller or on the L4 interconnect port.

Note: In DSI command mode, the display controller must be configured in stall mode by setting the DSS.DISPC_CONTROL[11] STALLMODE bit to 1.

The DSS.DSI_VCn_LONG_PACKET_HEADER registers are used for the header of long packets, the DSS.DSI_VCn_SHORT_PACKET_HEADER registers are used for the short packets.

The error correction code (ECC) can be provided while writing the ECC value directly into the DSS.DSI_VCn_LONG_PACKET_HEADER and DSS.DSI_VCn_SHORT_PACKET_HEADER registers. The DSS.DSI_VCn_CTRL[8] ECC_TX_EN bit indicates if the ECC value should be calculated or if the value written in the register should be used instead for command and video modes. In case of

synchronization short packets for video mode, since the hardware generates the short packets without using DSS.DSI_VCn_SHORT_PACKET_HEADER registers, if the DSS.DSI_VCn_CTRL[8] ECC_TX_EN bit is set to 1, the ECC is calculated otherwise the value zero is used. The feature is used to generate incorrect ECC for debug purpose and to ease the check for the link and peripheral error detection and correction.

For the payload, the DSS.DSI_VCn_LONG_PACKET_PAYLOAD registers are used. Each 32-bit PAYLOAD data is written into the DSS.DSI_VCn_LONG_PACKET_PAYLOAD register from the MPU subsystem or system DMA. It is buffered to be able to send packets with higher rate than the L4 interconnect frequency can provide. The word count defined in the DSS.DSI_VCn_LONG_PACKET_HEADER registers is used to determine the number of bytes to be sent using the DSS.DSI_VCn_LONG_PACKET_PAYLOAD registers. The write into the DSS.DSI_VCn_LONG_PACKET_HEADER registers is required before accessing the DSS.DSI_VCn_LONG_PACKET_PAYLOAD register. The hardware should be able to extract the length of the payload and be able to discard extra data sent using the DSS.DSI_VCn_LONG_PACKET_PAYLOAD register. The hardware takes into account the write into the DSS.DSI_VCn_LONG_PACKET_HEADER register only if the VC is enabled otherwise the write is ignored by hardware.

In the case of pixels received on the video port, only the DSS.DSI_VCn_LONG_PACKET_HEADER register is used. The video port pixels are used for the payload. When the pixel data is coming from the display controller video port, the DSI protocol can add a DCS command byte between the packet header and pixel data by setting the DSS.DSI_CTRL[24] DCS_CMD_ENABLE bit to 0x1. The value will be either 0x2c (write_memory_start) by setting the DSS.DSI_CTRL[25] DCS_CMD_CODE bit to 0x1, or 0x3c (write_memory_continue) by setting the DSS.DSI_CTRL[25] DCS_CMD_CODE bit to 0x0.

When transmitting RGB 16-BPP data, the DSS.DSI_CTRL[26] RGB565_ORDER bit must be set to 0x1 to maintain the pixel byte order as in video mode .

A 2-line ping-pong buffer is implemented to allow the DSI protocol engine to store incoming pixels from the display controller through the video port while sending the DSI formatted frame to the DSI_PHY. The ping-pong buffer is supported in command mode, provided the size of the packet defined in the header register is less than the size of each line buffer (768 *32 bits). If the size of the packet is greater than the size of the line buffer, the ping-pong mechanism cannot be used (both lines are used as a single line).

The ping-pong buffer status can be checked by the DSI_VCn_CTRL[14] PP_BUSY bit.

- When PP_BUSY equals 1, the ping-pong buffer is active and the line buffers are not ready to receive data; therefore, the user cannot update a new header.
- When PP_BUSY equals 0, at least one line buffer is empty; therefore, the user can update a new header. PP-BUSY is then set to 0x1. If both line buffers are empty, the user can write two headers, one following the other. PP_BUSY remains at 0x0 after the first header is written, and is set to 0x1 after second header is written.

An IRQ is available to allow software to update header on events. The IRQ is enabled by setting the DSI_VCn_IRQENABLE[8] PP_BUSY_CHANGE_IRQ bit to 0x1, and its status is accessible on the DSI_VCn_IRQSTATUS[8] PP_BUSY_CHANGE_IRQ bit.

15.4.3.3.3 Video + Command Modes

The two modes can be interlaced to send two DSI streams to two types of panels: Video or command types. The number of concurrent video stream is limited to a single one. The number of concurrent command mode streams is limited to 4 when there is no video stream and 3 otherwise. In case there is one DSI stream using video mode, the command mode pixels should be provided on the L4 interconnect only.

15.4.3.3.4 Burst Modes

- Frequency-burst mode The frequency-burst mode is used to reduce the high-speed (HS) period by increasing the clock frequency on the DSI link. It allows in some case, the power consumption reduction of the link. The non-HS period used typically to drive the main panel can be used to send data to the secondary panel or to allow feedback (acknowledge) from the primary and secondary panels. The DSI protocol engine needs to buffer a full line before sending the HS packets for the line. A double buffering mechanism is required to be able to send a line while the following one is being

received on the video port.

- Transparent-burst mode The transparent-burst mode is used by increasing the pixel clock frequency generated by the display controller with in addition an increase of the horizontal blanking period.

15.4.3.3.5 Interleaving Mode

Video mode can output command mode packets, which are provided to DSI through the L4 interconnect, during the blanking periods of the video stream sequence on the PPI link. These command mode packets can be programmed as high-speed packets or low-power packets.

During a video stream sequence on the PPI link, four types of gap exist:

- BLLP gap: Blanking period during VSA, VBP, and VFP lines
- HSA gap: Blanking period during VACT lines; always between HS and HE short packet
- HBP gap: Blanking period during VACT lines; always between HS/HE short packet and data pixel long packet
- HFP gap: Blanking period during VACT lines; always between data pixel long packet and the end of the current VACT line

To perform interleaving in a particular gap, video mode must be set to go into low-power state during the blanking gap. Each type of gap has separate configurable register bits that determine whether a blanking long packet will be sent or the link will go into low-power state during the gap on the PPI link. If low-power state is set during a gap, the DSI module performs interleaving during that period.

Two set of registers are available for:

- High-speed interleaving (when high-speed command mode packets must be sent during a video stream on the PPI link)
- Low-power interleaving (when low-power command mode packets must be sent during a video stream on the PPI link)

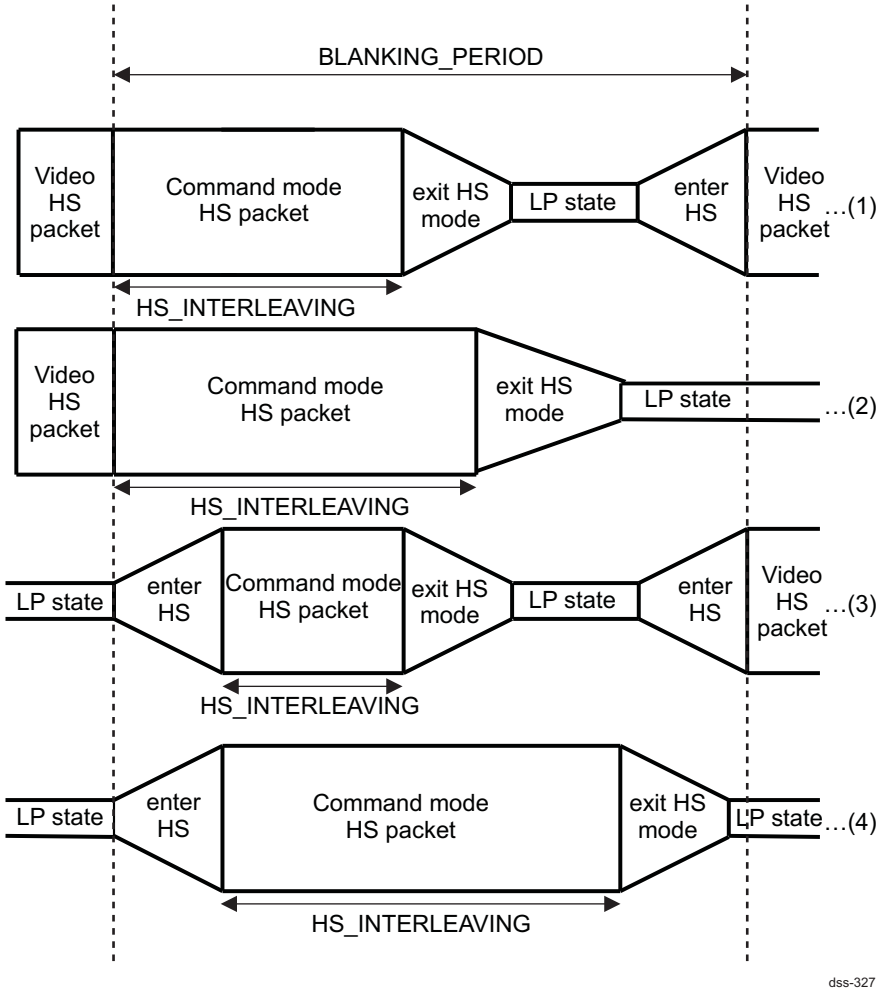
15.4.3.3.5.1 HS Command Mode Interleaving Programming Model

Figure 15-93 shows the various HS mode scenarios in interleaving mode during a blanking gap. For each type of blanking gap, a dedicated bit field determines the number of TxByteClkHS clock cycles used for interleaving in HS command mode packets.

- The BL_HS_INTERLEAVING[31:16] DSI_VM_TIMING6 bit field defines the number of TxByteClkHS clock cycles used to interleave HS command mode packets during a BLLP gap.
- The HBP_HS_INTERLEAVING[7:0] DSI_VM_TIMING4 bit field defines the number of TxByteClkHS clock cycles used to interleave HS command mode packets during an HBP gap.
- The HFP_HS_INTERLEAVING[15:8] DSI_VM_TIMING4 bit field defines the number of TxByteClkHS clock cycles used to interleave HS command mode packets during an HFP gap.
- The HSA_HS_INTERLEAVING[23:16] DSI_VM_TIMING4 bit field defines the number of TxByteClkHS clock cycles used to interleave HS command mode packets during an HSA gap.

These programmable values must be programmed to satisfy the timings for the clock and data lane to enter and exit HS mode latency. According to the scenario, different equations must be considered when calculating the register values.

Figure 15-93. HS Command Mode Interleaving



Note: For calculations and equations, the following abbreviations are used: EXIT_CLK_HS_MODE represents the exit HS mode latency for the clock lane. There is no dedicated register for this value but the programmer must know this value for further calculations.
EXIT_CLK_HS_MODE = $T_{CLK\text{-}TRAIL}$ + $TH_{S\text{-}EXIT}$

For the following equations, BLANKING_PERIOD represents the BLLP, HSA, HBP, or HFP blanking periods. The HS_INTERLEAVING period represents the maximal period HS command mode packets. Its value is set in the BL_HS_INTERLEAVING, HSA_HS_INTERLEAVING, HBP_HS_INTERLEAVING, or HFP_HS_INTERLEAVING registers, depending on the blanking type.

In each scenario, two calculations are present, depending on the value of ddr_clk_always_on.

- ddr_clk_always_on = 1: Clock lane is always active.
- ddr_clk_always_on = 0: Clock lane is activated only when there are HS packets to be sent on the PPI link.
- Scenario 1: The gap for interleaving starts and ends with a regular video stream HS packet.
  - ddr_clk_always_on = 1
    HS_INTERLEAVING = BLANKING_PERIOD – (EXIT_HS_MODE_LATENCY + max{ ENTER_HS_MODE_LATENCY, 2} + 1)
  - ddr_clk_always_on = 0
    HS_INTER1 = BLANKING_PERIOD – (EXIT_HS_MODE_LATENCY + max{ ENTER_HS_MODE_LATENCY, 2} + 1)

HS_INTER2 = BLANKING_PERIOD – (DDR_CLK_POST + EXIT_CLK_HS_MODE + DDR_CLK_PRE+ ENTER_HS_MODE_LATENCY + 1)
HS_INTERLEAVING = min{ HS_INTER1, HS_INTER2}

- Scenario 2: The gap for interleaving starts with a regular video stream HS packet and ends in LP state.
  - ddr_clk_always_on = 1
    HS_INTERLEAVING = BLANKING_PERIOD – (EXIT_HS_MODE_LATENCY + 3)
  - ddr_clk_always_on = 0
    HS_INTER1 = BLANKING_PERIOD – (EXIT_HS_MODE_LATENCY + 3)
    HS_INTER1 = BLANKING_PERIOD – (DDR_CLK_POST + EXIT_CLK_HS_MODE + 3)
    HS_INTERLEAVING = min{ HS_INTER1, HS_INTER2}
- Scenario 3: The gap for interleaving starts with the LP state and ends with a regular video stream HS packet.
  - ddr_clk_always_on = 1
    HS_INTERLEAVING = BLANKING_PERIOD – (ENTER_HS_MODE_LATENCY + EXIT_HS_MODE_LATENCY + max{ ENTER_HS_MODE_LATENCY, 2} + 1)
  - ddr_clk_always_on = 0
    HS_INTER1 = BLANKING_PERIOD – (DDR_CLK_PRE + ENTER_HS_MODE_LATENCY + EXIT_HS_MODE_LATENCY + max{ ENTER_HS_MODE_LATENCY, 2} + 1)
    HS_INTER2 = BLANKING_PERIOD – (DDR_CLK_PRE + ENTER_HS_MODE_LATENCY + DDR_CLK_POST + EXIT_CLK_HS_MODE + DDR_CLK_PRE+ ENTER_HS_MODE_LATENCY + 1)
    HS_INTERLEAVING = min{ HS_INTER1, HS_INTER2})
- Scenario 4: The gap for interleaving starts with the LP state and ends with a regular video stream HS packet.
  - ddr_clk_always_on = 1
    HS_INTERLEAVING = BLANKING_PERIOD – (ENTER_HS_MODE_LATENCY + EXIT_HS_MODE_LATENCY + 3)
  - ddr_clk_always_on = 0
    HS_INTER1 = BLANKING_PERIOD – (DDR_CLK_PRE+ ENTER_HS_MODE_LATENCY + EXIT_HS_MODE_LATENCY + 3)
    HS_INTER2 = BLANKING_PERIOD – (DDR_CLK_PRE+ ENTER_HS_MODE_LATENCY + DDR_CLK_POST + EXIT_CLK_HS_MODE + 1)
    HS_INTERLEAVING = min{ HS_INTER1, HS_INTER2}

15.4.3.3.5.2 LP Command Mode Interleaving Programming Model

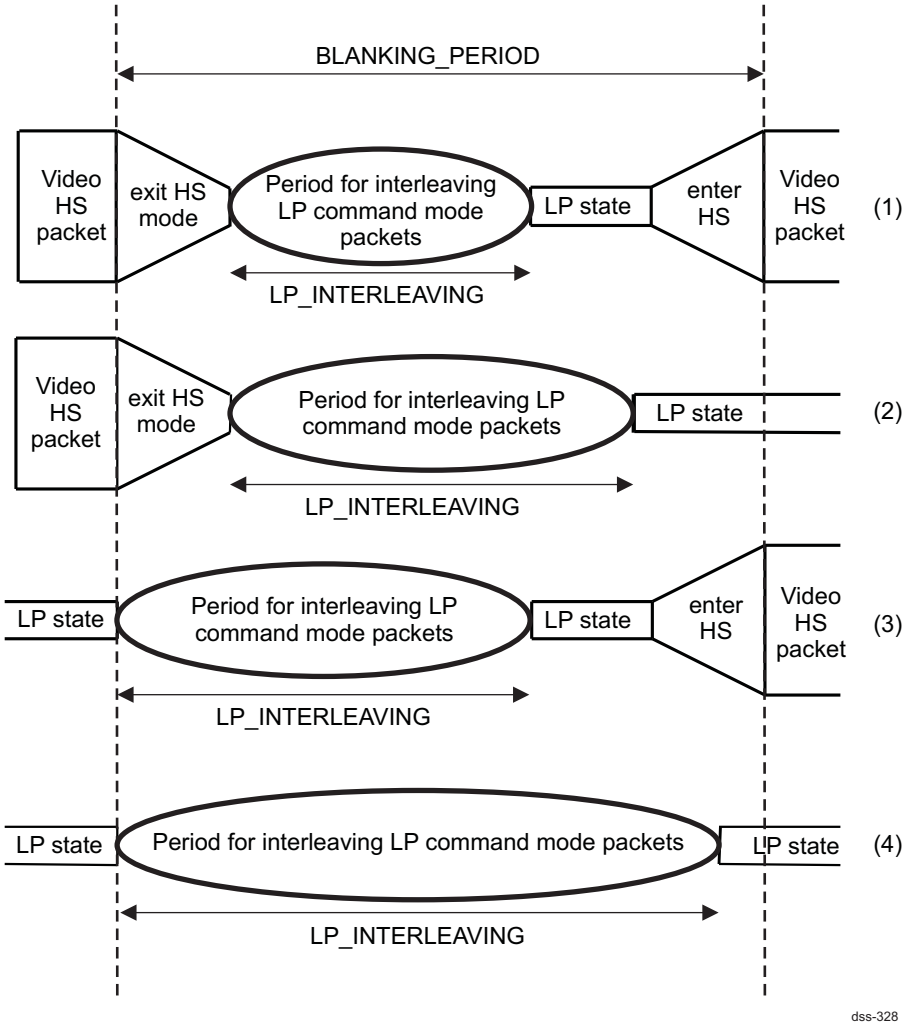
Figure 15-94 shows the various LP mode scenarios in interleaving mode during a blanking gap. For each type of blanking gap, a dedicated bit field determines the number of TxByteClkHS clock cycles used for interleaving in LP command mode packets.

- BL_LP_INTERLEAVING bit field DSI_VM_TIMING6[15:0] defines the number of TxByteClkHS clock cycles used to interleave the HS command mode packets during a BLLP gap.
- HBP_LP_INTERLEAVING bit field DSI_VM_TIMING5[7:0] defines the number of TxByteClkHS clock cycles used to interleave the HS command mode packets during an HBP gap.
- HFP_LP_INTERLEAVING bit field DSI_VM_TIMING5[15:8] defines the number of TxByteClkHS clock cycles used to interleave HS command mode packets during an HFP gap.
- HSA_LP_INTERLEAVING bit field DSI_VM_TIMING5[23:16] defines the number of TxByteClkHS clock cycles used to interleaveHS command mode packets during an HSA gap.

These programmable values must be programmed to satisfy the timings for clock and data lane enter and exit LP mode latency. Clock lane timings do not affect LP command mode interleaving, because the clock lane can be controlled separately, compared with the data lane high-speed and low-power mutually exclusive control. Clock lanes can be in high-speed mode while the data lanes are in high-speed data transfer mode, low-power data transfer mode, or in low-power state.

According to this scenario, different equations must be considered for calculating register values.

Figure 15-94. LP Command Mode Interleaving



For the following equations, BLANKING_PERIOD represents the BLLP, HSA, HBP, or HFP blanking periods. The LP_INTERLEAVING period represents the maximal period in LP command mode packets. Its value is set in the BL_LP_INTERLEAVING, HSA_LP_INTERLEAVING, HBP_LP_INTERLEAVING, or HFP_LP_INTERLEAVING registers, depending on the blanking type.

ALLOWED_HSBYTE_CLOCKS_FOR_LP represents the number of TxByteClkHS clock cycles during which LP interleaving can appear.

To calculate the LP_INTERLEAVING value:

1. Calculate how many TxByteClkHS clock cycles can be reserved for LP interleaving during the appropriate blanking video mode gap.
2. Calculate the LP_INTERLEAVING value according to the results of Step 1.

Step 1:

- Scenario 1: The gap for interleaving starts and ends with a regular video stream HS packet.
  ALLOWED_HSBYTE_CLOCKS_FOR_LP = BLANKING_PERIOD – (EXIT_HS_MODE_LATENCY + max{ ENTER_HS_MODE_LATENCY, 2} + 1)
- Scenario 2: The gap for interleaving starts with a regular video stream HS packet and ends in LP state.
  ALLOWED_HSBYTE_CLOCKS_FOR_LP = BLANKING_PERIOD – (EXIT_HS_MODE_LATENCY + 1)
- Scenario 3: The gap for interleaving starts with the LP state and ends with a regular video stream HS packet.
  ALLOWED_HSBYTE_CLOCKS_FOR_LP = BLANKING_PERIOD – (max{

ENTER_HS_MODE_LATENCY, 2} + 1)

- Scenario 4: The gap for interleaving starts with the LP state and ends with a regular video stream HS packet.

  ALLOWED_HSBYTE_CLOCKS_FOR_LP = BLANKING_PERIOD – 1

After finishing Step 1, the time period available for LP interleaving is known:

$T_{lp_available}$ = ALLOWED_HSBYTE_CLOCKS_FOR_LP *T TxByteClkHS

Step 2:

The resulting value must be programmed in the appropriate video mode register for LP interleaving.

$$LP_INTERLEAVING < \left[\frac{\frac{Tlp_available - 8*Thsbyte_clk - 5*Tdsif_clk}{Ttxclkesc}}{16} - 26 \right]$$

dss-E124

TxByteClkHS: Period of HS byte clock of DSI_PHY module

Tdsif_clk: Period of DSI functional clock

Ttxclkesc: Period of LP transmit escape clock

15.4.3.4 Power Management

The DSI protocol engine implements an handshake protocol on its L4 interconnect port with the PRCM. The DSI protocol engine provides a clock gating signal CIO_CLK_ICG to gate the L3 interface clock (L3_ICLK) provided by the PRCM to the DSI complex I/O. It allows reduction of the power consumption of the DSI complex I/O while the DSI link is not in used. To gate the L3_ICLK clock at DSI complex I/O level, set the DSS.DSI_CLK_CTRL[14] CIO_CLK_ICG bit to 1.

15.4.3.5 Serial Configuration Port (SCP) Interface

The SCP interface is used to transfer register values from the DSI protocol engine to the DSI PLL Control module and to the DSI complex I/O. It spends several cycles to serialize the data to be sent. Software users should take into account the delay in processing the transfer of the data from/to the slave port to/from the module.

15.4.3.5.1 Shadowing Register

The two first SCP registers for the DSI complex I/O address map should be implemented as shadow registers. The shadowing mechanism is enabled/disabled using the DSS.DSI_COMPLEXIO_CFG1[31] SHADOWING bit:

- When setting the DSS.DSI_COMPLEXIO_CFG1[31] SHADOWING bit to 1, the transfer of the values from the two first L4 interconnect port registers into the two first registers of the DSI complex I/O (DSS.DSI_PHY_CFG0 and DSS.DSI_PHY_CFG1) is done only when the DISPC_UPDATE_SYNC signal from the display controller is active and the DSS.DSI_COMPLEXIO_CFG1[30] GOBIT is set to 1. If there is no pending update for the two registers, when the DISPC_UPDATE_SYNC signal is asserted, the DSS.DSI_COMPLEXIO_CFG1[30] GObit is reset by hardware and there is no SCP transfer.
  - If there is only one register to update, only the corresponding new value is transferred. The second register in the DSI complex I/O is not updated. When the transfer is completed, the DSS.DSI_COMPLEXIO_CFG1[30] GOBIT is reset by hardware.
  - If the two registers need to be updated, the order of the transfer is first the register with lower address and then the second one. When the transfers are completed, the DSS.DSI_COMPLEXIO_CFG1[30] GOBIT is reset by hardware.

  When there is an on-going transfer (read or write) to any SCP register, the transfer should complete prior to start the update of shadowing registers.

- When unsetting the DSS.DSI_COMPLEXIO_CFG1[31] SHADOWING bit to 0, if the transfer into the two first DSI complex I/O registers has already started, it should be finished

Note: When reading the shadow registers, the local value stored in the DSI protocol engine is returned if the update is pending; otherwise, the values stored in the DSI complex I/O are returned.

15.4.3.5.2 Busy Signal

The signal SCPBusy indicates that there is still some activity using the SCPClk provided by the PRCM. The SCPClk clock is the DSS_L3_ICLK clock.

15.4.3.6 Power Control

The DSI protocol engine can control and send power commands for both DSI complex I/O and DSI PLL controller modules.

15.4.3.6.1 Complex I/O Power Control Commands

15.4.3.6.1.1 Complex I/O Power Control Commands

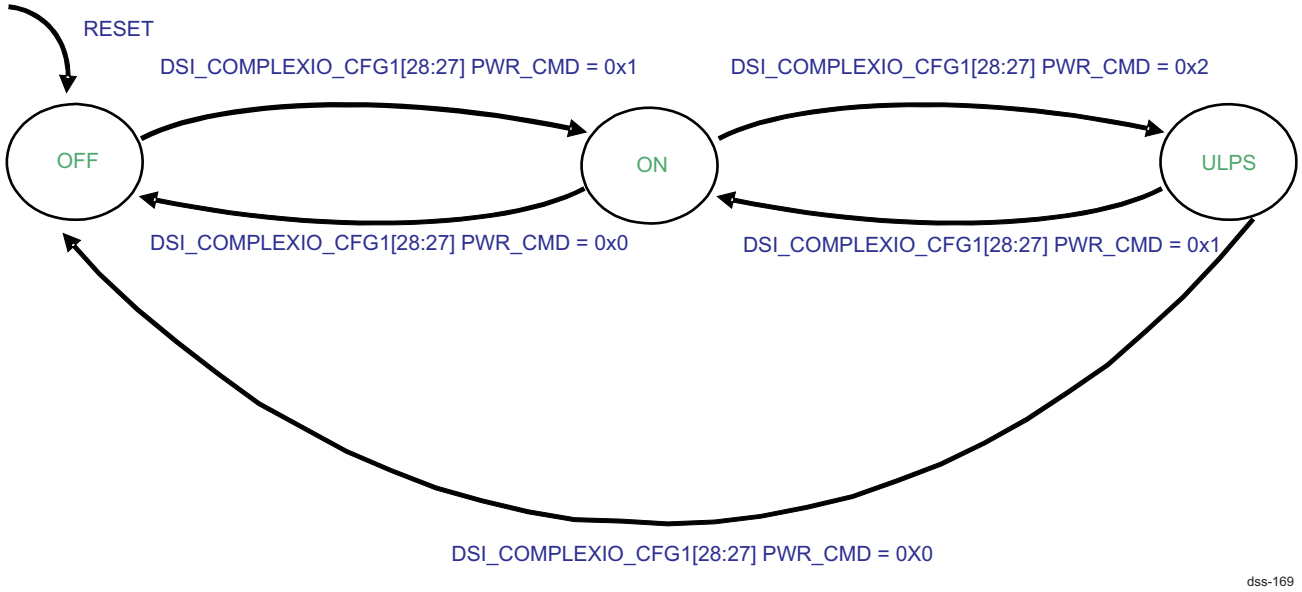
The DSI complex I/O can be set into three modes:

- OFF: In this power state, the complete DSI_PHY circuit is powered down. The internal LDO is OFF.
- ON: In this power state, the complete DSI_PHY circuit is powered on and functional.
- ULPS: In this power state, the ULPS exit detection circuit power switch is ON for the lanes which are in receive ULPS mode. For the lanes which are in transmit ULPS mode, the circuitry for weak pull-down is ON. The ultralow-power state should only be used when all the three lanes are in ULPS (transmit or receive).

15.4.3.6.1.2 Complex I/O Power FSM

Figure 15-95 describes the power control FSM to control the power state of the complex I/O.

Figure 15-95. Complex I/O Power FSM



The PwrCmdOff, PwrCmdUlp and PwrCmdOn commands control the state transition of the DSI complex I/O. Software users should set the DSS.DSI_COMPLEXIO_CFG1[28:27] PWR_CMD bit field to ask for a state change. The allowed transitions are: OFF -> ON and ON -> ULP and ULP -> OFF. The DSS.DSI_COMPLEXIO_CFG1[26:25] PWR_STATUS bit field gives a status on the current state of the DSI complex I/O.

CAUTION

- In automatic mode, the software should ensure that the DSI complex I/O in the ON mode (that is, ON command already sent) before sending requests to the complex I/O.
- In a command request to change to a state which is the current one (acknowledge has been received), the command is ignored (nothing is sent to the DSI complex I/O).
- To change state to ULP state, users should ensure that all the three ULPSActiveNot signals are low. The ULPSActiveNot_ALL0_IRQ interrupt can be used by software users to determine the state of the ULPSActiveNot signals. The change from ULP to ON state is required before starting the ULP status exit sequence (refer to Section 15.4.3.7.1 for details).

15.4.3.6.2 DSI PLL Power Control Commands

The DSI PLL controller module can be set into four modes:

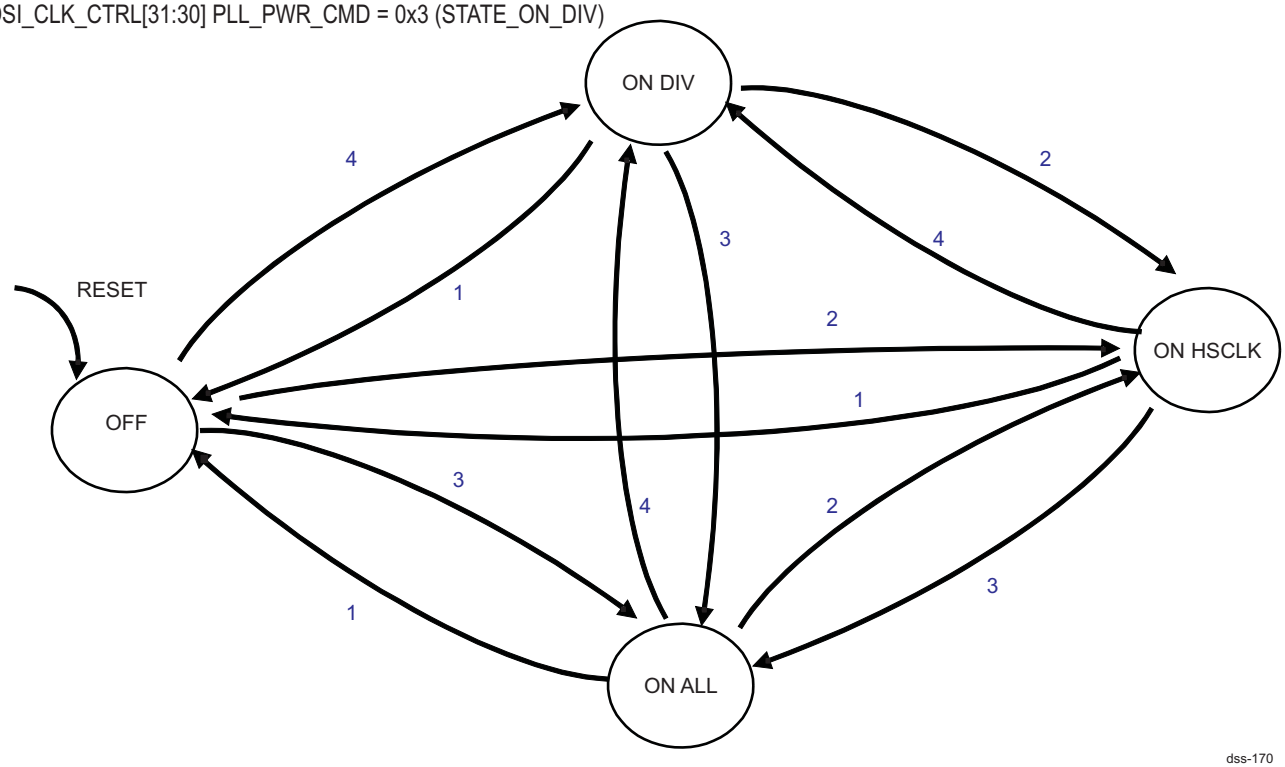
- OFF: The DSI PLL and HSDIVIDER are OFF.
- ON ALL: Both DSI PLL and HSDIVIDER are ON. The HS_CLK clock is provided to the DSI complex I/O and the second clock output is provided to the HSDIVIDER.
- ON HSCLK: The DSI PLL is ON. The HSDIVIDER is OFF. The HS_CLK clock is provided to the DSI complex I/O but the second clock output is not provided to the HSDIVIDER.
- ON DIV: Both DSI PLL and HSDIVIDER are ON. The HS_CLK clock is not provided to the DSI complex I/O but the second clock output is provided to the HSDIVIDER.

15.4.3.6.2.1 DSI-PLL Power FSM

Figure 15-96 shows the DSI PLL power FSM.

Figure 15-96. DSI PLL Power FSM

1 = DSI_CLK_CTRL[31:30] PLL_PWR_CMD = 0x0 (STATE_OFF)
2 = DSI_CLK_CTRL[31:30] PLL_PWR_CMD = 0x1 (STATE_ON_HSCLK)
3 = DSI_CLK_CTRL[31:30] PLL_PWR_CMD = 0x2 (STATE_ON_ALL)
4 = DSI_CLK_CTRL[31:30] PLL_PWR_CMD = 0x3 (STATE_ON_DIV)



The commands PLLPwrCmdOff, PLLPwrCmdOnAll, PLLPwrCmdOnDIV and PLLPwrCmdOnHSClk controls the state transition of the DSI PLL control module. Software users should set the DSS.DSI_CLK_CTRL[31:30] PLL_PWR_CMD bit field to ask for a state change. The DSS.DSI_CLK_CTRL[29:28] PLL_PWR_STATUS bit field gives a status on the current state of the DSI PLL controller.

Note: In a command requests to change to a state which is the current one (acknowledge has been received), the command is ignored (nothing is sent to the DSI PLL Control module).

All the DSI PLL power is controlled by the DSI protocol engine except the LDO power of the PLL and HSDIVIDER that can be controlled by the DSI PLL controller module. Indeed, the HSDIVIDER and PLL

SYSRESET signals can be forced by the DSI PLL controller module by setting DSS.DSI_PLL_CONTROL[4] DSI_HSDIV_SYSRESET and DSS.DSI_PLL_CONTROL[3] DSI_PLL_SYSRESET bits, respectively. By setting these bits to 1, the SYSRESET signal is forced active (module is forced to reset state). When these bits are set to 0 (reset value), the SYSRESET signals are controlled by the DSI PLL power FSM.

15.4.3.6.2.1.1 DSI-PLL HS Clock Signals

The DSIStopClk signal is provided to the DSI PLL control module. It indicates when the DSI Protocol engine does not need to use the high-speed transfer mode (HS mode) and PLL HS output (HS_CLK clock) can be stopped. The following conditions must also be met when DISPC_UPDATE_SYNC may be generated by the display controller, as that may also result in the PLL HS output being stopped.

When the interface is disabled (that is, DSS.DSI_CTRL[0] IF_EN bit set to 0), the signal DSIStopClk is asserted.

The assertion of the DSIStopClk depends on the following conditions:

- Clock lane TxRequestHS is deasserted (the DDR clock on the clock lane is not required anymore). The get TxRequestHS deassertion, all of the following conditions are required:
  - The DSS.DSI_CLK_CTRL[13] DDR_CLK_ALWAYS_ON bit must be reset to 0 and no HS data transfer should be on going or already scheduled
  - No VC active in video mode. No VC using the video mode is enabled; if the VC is enabled, the mode is command mode only (that is, DSS.DSI_VCn_CTRL[0] VC_EN bit set to 1 and DSS.DSI_VCn_CTRL[4] MODE bit set to 0)
  - No command mode requiring high-speed transfer (one or more VCs using command mode can be active)
  - Or DSS.DSI_CTRL[0] IF_EN bit reset to 0 (if all previous conditions are not required)

The deassertion of the DSIStopClk depends on one of the following conditions (the DSI interface is enabled by setting the DSS.DSI_CTRL[0] IF_EN bit to 1):

- Clock lane TxRequestHS must be asserted (the DDR clock on the clock lane is required anymore).
- One video mode VC active
- At least one VC in command mode requiring high-speed transfer
- The DSS.DSI_CLK_CTRL[13] DDR_CLK_ALWAYS_ON bit is set to 1 by software users (the DSS.DSI_CTRL[0] IF_EN bit should be reset to 0 for updating the DDR_CLK_ALWAYS_ON bit value)

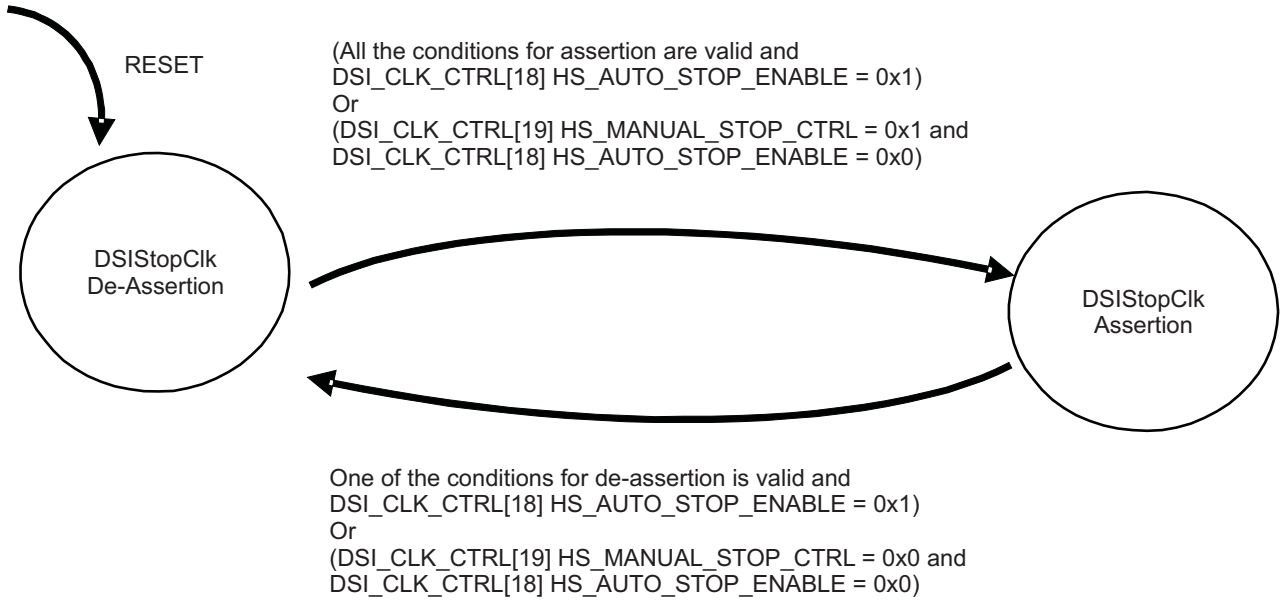
The automatic assertion/deassertion is enabled by using the DSS.DSI_CLK_CTRL[18] HS_AUTO_STOP_ENABLE bit.

The manual mode can be used by setting/resetting the DSS.DSI_CLK_CTRL[19] HS_MANUAL_STOP_CTRL bit to assert/deassert the DSIStopClk signal.

15.4.3.6.2.1.2 DSI-PLL HS Clock FSM

Figure 15-97 shows the DSI PLL HS clock FSM.

Figure 15-97. DSI PLL HS Clock FSM



When DSIStopClk is used there is a latency through other modules (DSI PLL controller and DSI_PHY) before TxByteClkHS is stopped. This latency must be accounted for to prevent any issue when DSIStopClk is deasserted soon after being asserted. This is done using a hardware timer programmed using the DSS.DSI_STOPCLK_TIMING[7:0] DSI_STOPCLK_LATENCY bit field. This timer is programmed in number of periods of the DSI Protocol functional clock (DSI_FCLK). At reset value, the timer is programmed with 0x80 (128) value.

CAUTION

The programmed value in the DSS.DSI_STOPCLK_TIMING[7:0] DSI_STOPCLK_LATENCY bit field must be greater than ((3 x L3_ICLK period) + (5 x CLKIN4DDR period))/(DSI_FCLK period).

15.4.3.7 Timers

Note: Among the timers described in this section, only the HS TX, LP RX and turnRequests timers generates interrupts immediately when the timer value is null. For ForceTxStopMode timer, it ends counting instantly and ForceTxStopMode is not asserted.

15.4.3.7.1 Twakeup Timer

The T_{WakeUp} timer is not implemented in the DSI protocol engine. The software must use a general-purpose (GP) timer to handle this. This timer is used for existing ULP status mode for the active lanes (clock and/or data lanes). The sequence to exit ULP state is:

1. Change the state of TxULPSExit for each lane to ACTIVE.
2. Wait for the interrupt indicating that all lanes with TxULPSExit active have acknowledged by asserting ULPSActiveNot. This is done by reading the DSS.DSI_COMPLEXIO_IRQSTATUS ULPSACTIVENOT_ALLi_IRQ bit fields (i = 0, 1).

3. Start the application wake-up timer (GP timer).
4. Wait for the time-out.
5. Change the TxUlpsClk signals to INACTIVE state for the clock lane and/or TxRequestEsc INACTIVE state for the data lane(s).

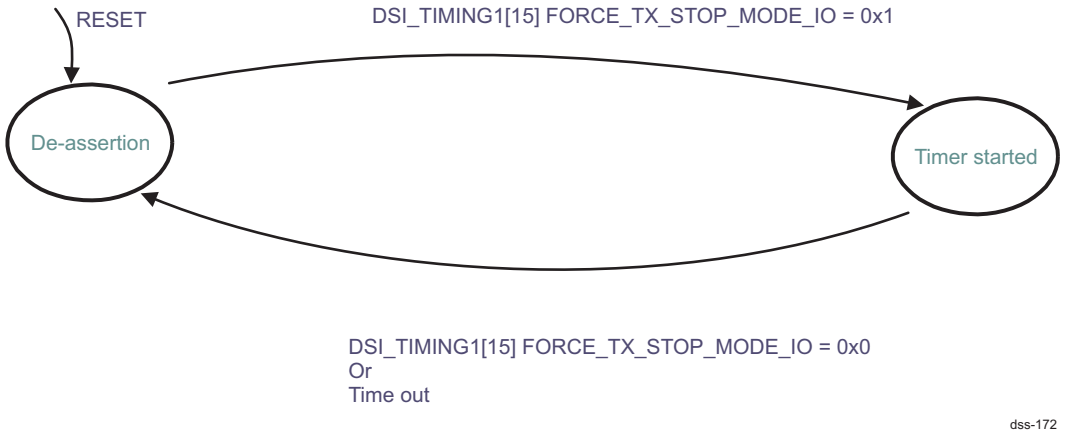
Note: The minimum time for the wake-up period is 1 ms.

To enter ULPS mode for clock lane, TxUlpsClk state should be change to active state. To enter ULPS mode for data lane, TxRequestEsc state should be changed to active state (TxUlpsEsc as well if it is not in active state already).

15.4.3.7.2 ForceTxStopMode FSM

The signal ForceTxStopMode is used at initialization time (DSI complex I/O). Figure 15-98 describes the ForceTxStopMode FSM to assert/deassert ForceTxStopMode signal.

Figure 15-98. ForceTxStopMode FSM



The DSI protocol engine asserts ForceTxStopMode by setting the DSS.DSI_TIMING1[15] FORCE_TX_STOP_MODE_IO bit to 1. Asserting the FORCE_TX_STOP_MODE_IO bit allows to initialize the lanes. The lanes are in the Stop State when ForceTxStopMode signal is high.

No data can be sent before the ForceTxStopMode signal is deasserted. The deassertion time is defined by the STOP_STATE_COUNTER_IO, Stop_State_x4_IO, Stop_State_x16_IO field DSI_TIMING1[15:0]. The FORCE_TX_STOP_MODE_IO bit is reset by hardware when the time is reached.

This bit can be reset by software.

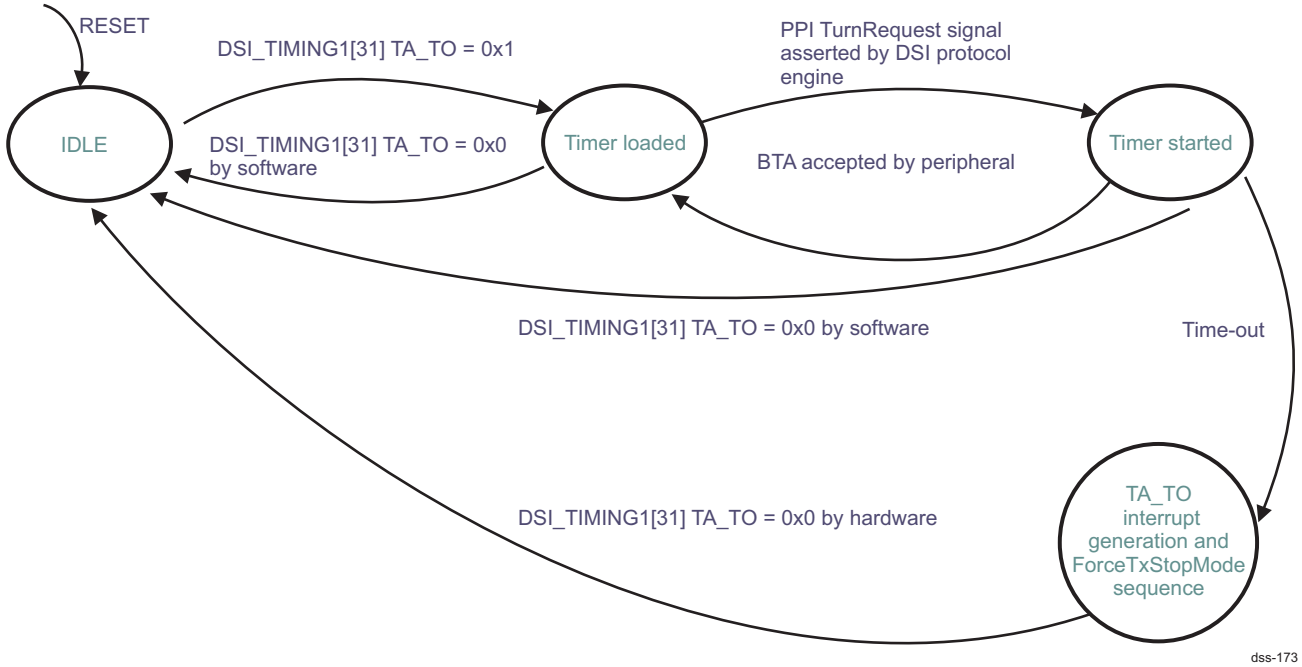
The calculation of the number of DSI_FCLK cycles assertion period is defined by:

Total period in DSI_FCLK cycles = DSI_TIMING1[12:0] STOP_STATE_COUNTER_IO x ((DSI_TIMING1[14] STOP_STATE_X16_IO x 15) + 1) x ((DSI_TIMING1[13] STOP_STATE_X4_IO x 3) + 1)

15.4.3.7.3 TurnRequest FSM

The signal TurnRequest is used to request turnaround. It is only valid for the data lane #1 since the other data lanes can not be used in the reverse direction to receive data from the DSI receiver. Figure 15-99 describes the TurnRequest FSM to assert/deassert TurnRequest signal.

Figure 15-99. TurnRequest FSM



The DSI protocol engine asserts TurnRequest signal during one TxClkEsc cycle when the turn-around is enabled. The DSS.DSI_TIMING1[31] TA_TO bit is set/reset by software to respectively enable/disable the timer for turnaround procedure failure. It can be reset by software or automatically by hardware when the time out occurs.

The timer is loaded with the value in number of DSI_FCLK cycles:

DSI_TIMING1[28:16] TA_TO_COUNTER x ((DSI_TIMING1[30] TA_TO_X16 x 15) + 1) x ((DSI_TIMING1[29] TA_TO_X8 x 7) + 1).

When the interrupt is generated, the hardware should automatically assert ForceTXStopMode in order for the DSI_PHY to drive LP-11 stop state. The ForceTXStopMode timer is used to define the minimum duration of LP-11 state. The Stop State can be longer if there is no activity.

The hardware resets the ForceTXStopMode bit, followed by an internal logic reset except all register values and TX FIFO content, then resets the DSS.DSI_CTRL[0] IF_EN bit. The software should take action to recover by resetting the peripheral, for example, if it is not responding. It should wait for DSS.DSI_TIMING1[15] FORCE_TX_STOP_MODE_IO and DSS.DSI_CTRL[0] IF_EN bits to be reset to 0 before starting the recovery sequence.

15.4.3.7.4 Peripheral Reset Timer

The peripheral reset timer is not implemented in the DSI protocol engine module. Such as the Twakeup timer, a general-purpose timer (GPTimer)should be used in case of reset of the peripheral to determine when the peripheral is ready again for operation.

15.4.3.7.5 HS TX Timer

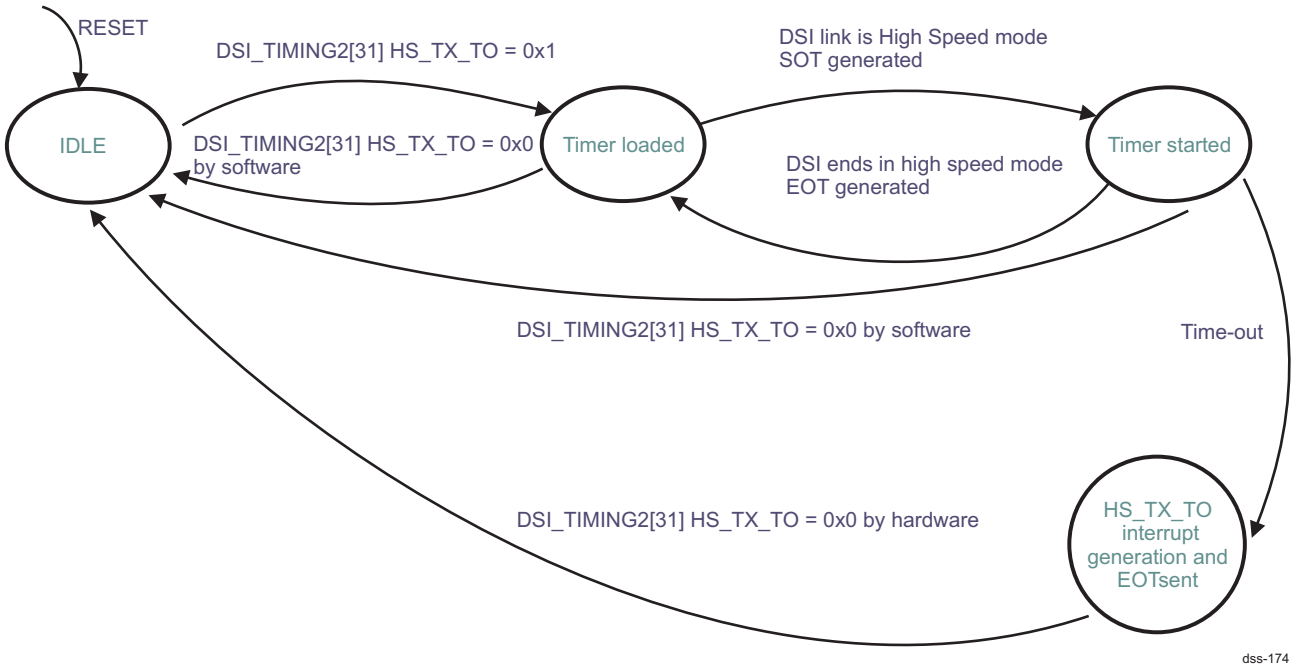
The HS TX timer is used to detect when the host has been in TX mode for too long. When time-out occurs, the EOT is forced. The timer is reloaded when a start of high speed transmission occurs. It is enabled/disabled by software through the DSS.DSI_TIMING2[31] HS_TX_TO bit. The interrupt HS_TX_TO_IRQ is generated when the timer expires. The DSS.DSI_IRQSTATUS[14] HS_TX_TO_IRQ bit is set to 1 when the HS TX time-out occurs.

The maximum time to be supported is 20 ms. It can be used to determine that at least once a frame in video mode, the HS mode is stopped to enter ULPS. Since the refresh rate can be up to 50 frames per second in video mode, the maximum time in HS is 20 ms.

The timer is loaded with the value in number of TxByteClkHS:

DSI TIMING2[28:16] HS_TX_TO_COUNTER x ((DSI TIMING2[30] HS_TX_TO_X16 x 15) +1) x ((DSI TIMING2[29] HS_TX_TO_X8 x 7) +1)

Figure 15-100. High-Speed TX Timer FSM



When the time-out occurs, the hardware should sent EOT request in order for the DSI complex I/O to drive LP-11 stop state. This is followed by the generation of the interrupt. The hardware will perform an internal logic reset including the TX FIFO content, but excluding the register values and then resets the DSS.DSI_CTRL[0] IF_EN bit.

The software should wait for the DSS.DSI_CTRL[0] IF_EN bit to be reset to 0 before taking any recovery action by resetting for example the peripheral if it is not responding.

15.4.3.7.6 LP RX Timer

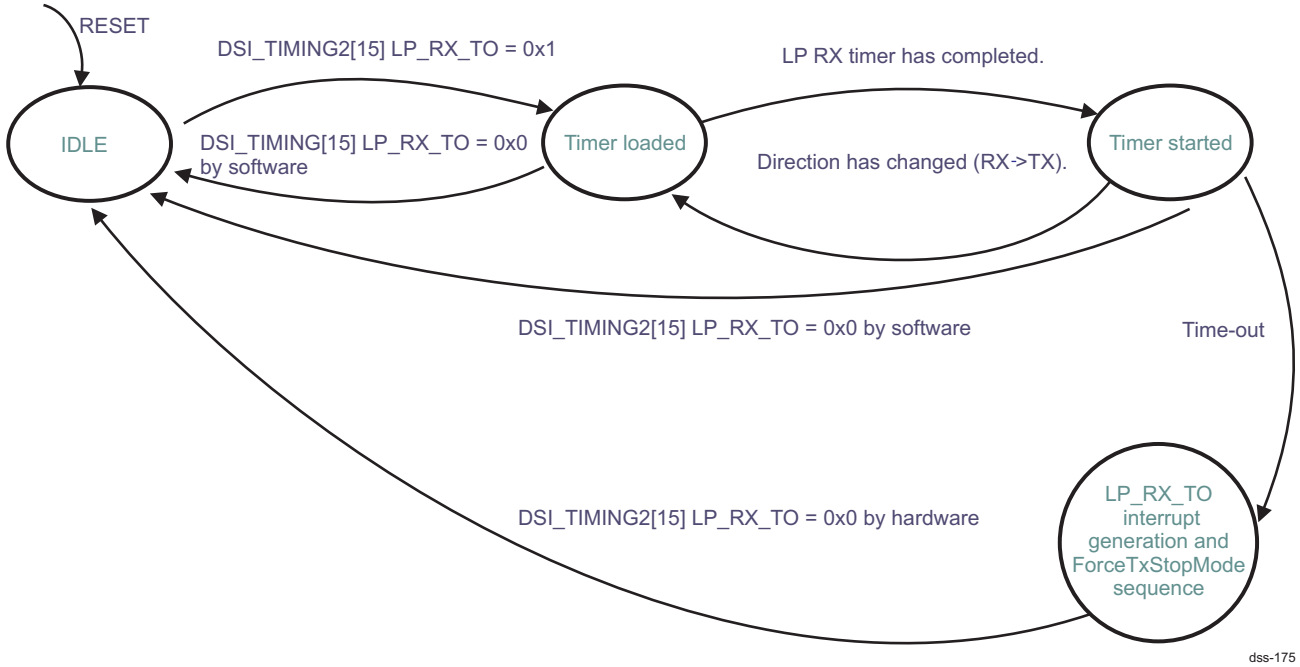
When the host is in Low power Receive mode after a bus turn-around, the LP RX timer is loaded. When the timer expires, the host requests the DSI complex I/O to drive LP-11. The interrupt LP_RX_TO_IRQ is generated when the timer expires. The DSS.DSI_IRQSTATUS[15] LP_RX_TO_IRQ bit is set to 1 when the LP RX time-out occurs.

The DSS.DSI_TIMING2[15] LP_RX_TO bit is set/reset by the software to respectively enable/disable the timer.

The timer is loaded with the value in number of DSI_FCLK cycles:

DSI_TIMING2[12:0] LP_RX_TO_COUNTER x ((DSI_TIMING2[14] LP_RX_TO_X16 x 15) + 1) x ((DSI_TIMING2[13] LP_RX_TO_X4 x 3) + 1)

Figure 15-101. Low-Power RX Timer FSM



When the interrupt is generated, the hardware should automatically reset the DSS.DSI_TIMING2[15] LP_RX_TO bit and then assert ForceTXStopMode in order for the DSI complex I/O to drive LP-11 stop state. The ForceTXStopMode timer is used to define the minimum duration of LP-11 state. The Stop State can be longer if there is no activity.

The hardware resets the ForceTXStopMode bit, followed by an internal logic reset except all register values and TX FIFO content, then resets the DSS.DSI_CTRL[0] IF_EN bit. The software should take action to recover by resetting the peripheral, for example, if it is not responding. It should wait for the DSS.DSI_TIMING1[15] FORCE_TX_STOP_MODE_IO and DSS.DSI_CTRL[0] IF_EN bits to be reset before starting the recovery sequence. The TX FIFO is not flushed (the FIFO is flushed only when DSS.DSI_VCn_CTRL[0] VC_EN is set to 1).

15.4.3.8 Bus Turnaround

The bus turn-around (BTA) is not automatically sent by default after each packet sent to the display(s). It is programmable independently for each VC ID. The VC can be enabled when DSS.DSI_VCn_CTRL[6] BTA_EN bit is set to 1 by software. The software should ensure that, when the BTA is sent to the peripheral, there is enough time allocated for the response and the BTA from the peripheral to host. When setting the DSS.DSI_VCn_CTRL[6] BTA_EN bit to 1, one BTA is sent manually to the peripheral. This manual mode can be used for packets in command or video mode.

CAUTION

The BTA should not be sent when the RX FIFO is not empty. Users should take care of emptying the RX FIFO before sending BTA to the peripheral. It is to ensure that when receiving new data from peripheral, all the allocated spaces for all the VCs are empty.

In automatic mode, the BTA is sent automatically at the end of short or long packets when respectively the DSS.DSI_VCn_CTRL[2] BTA_SHORT_EN or the DSS.DSI_VCn_CTRL[3] BTA_LONG_EN bits are set to 1.

Note: If the DSS.DSI_VCn_CTRL[2] BTA_SHORT_EN bit is enabled, users can still set the DSS.DSI_VCn_CTRL[6] BTA_EN bit. Only one BTA is sent to the peripheral and the DSS.DSI_VCn_CTRL[6] BTA_EN bit is reset by hardware.

If the DSS.DSI_VCn_CTRL[3] BTA_LONG_EN bit is enabled, users can still set the DSS.DSI_VCn_CTRL[6] BTA_EN bit. Only one BTA is sent to the peripheral and the DSS.DSI_VCn_CTRL[6] BTA_EN bit is reset by hardware.

If the DSS.DSI_VCn_CTRL[2] BTA_SHORT_EN and DSS.DSI_VCn_CTRL[3] BTA_LONG_EN bits are both enabled, users can still set the DSS.DSI_VCn_CTRL[6] BTA_EN bit to send a BTA. Only one BTA is sent and the DSS.DSI_VCn_CTRL[6] BTA_EN bit is reset by hardware.

As explained previously, two modes can be used for each VC ID:

- Automatic: After each packet, a bus turn-around is sent. To determine the size of the long packet, the protocol engine on the host side should read the word count defined in the header (in DSS.DSI_VCn_LONG_PACKET_HEADER register) and use it to determine the last data to be sent on the DSI link. For short packets, the size is always 4 bytes. Then the bus turn-around is sent to the peripheral. The word count is also used to determine how many bytes should be transferred from the 32-bit writes access to the payload register (DSS.DSI_VCn_LONG_PACKET_PAYLOAD register).
- Manual: In case of data transfer using the L4 interconnect port, while all data have been provided to the DSI protocol engine, users can select bus turn-around for the last packet provided to the L4 interconnect port only by setting the bus turn-around enable bit (DSS.DSI_VCn_CTRL[6] BTA_EN) or for last packets and following ones by setting the automatic mode; in case of data transfer using the video port, the bus turnaround enable bit (DSS.DSI_VCn_CTRL[6] BTA_EN) can be selected at any time during the transfer of the packet. In case of video mode packets (data and synchronization events) users can not determine when the BTA is sent relatively the video mode packets, so it is highly recommended to use manual BTA mode only for packets generated in command mode but it is possible to use BTA when for a VC in video mode. In case of data provided on the video port, an interrupt for end of packet transfer (PACKET_SENT_IRQ) is provided to indicate users when the packet has been completely sent by the DSI complex I/O. The PACKET_SENT_IRQ can be monitored in DSI_VCn_IRQSTATUS[2] PACKET_SENT_IRQ status bit. Users can request BTA even if the space allocated in the TX FIFO for the corresponding VC is empty. It can be sent later on even if there was no packet sent before BTA request. The DSS.DSI_VCn_CTRL[6] BTA_EN bit should be reset by hardware if the BTA request has been sent even if the automatic mode for this specific type of packets is enabled.

The bus turnaround is supported for video mode packets and for command mode packets. It is not possible to send BTA during the blanking periods of the video mode when HS blanking packets should be sent, that is, when one of the following bits is set to 1:

- DSS.DSI_CTRL[20] BLANKING_MODE
- DSS.DSI_CTRL[21] HFP_BLANKING_MODE
- DSS.DSI_CTRL[22] HBP_BLANKING_MODE
- DSS.DSI_CTRL[23] HSA_BLANKING_MODE

Therefore, in video mode, the BTA request is delayed until there is a blanking period without HS blanking packets.

TA Timer

When TurnRequest signal is asserted (always only for data lane #1), the TA_TO timer is started. If the direction signal is no changed according to the turn-around request, the TA_TO interrupt is generated. When the Direction signal is in output mode, any data on the input data bus should be ignored since the DSI is in transmission mode (data and triggers should be ignored). Refer toSection 15.4.3.7.3 for more details on the TA_TO timer.

15.4.3.9 PHY Triggers

The DSI protocol engine uses three triggers:

- Reset from host to display
- TE from display to host

- Acknowledge

These triggers are supported only for data lane #1.The reset trigger is associated with the signal TxTriggerEsc1[3]. The TE trigger is associated with the signal RxTriggerEsc1[2]. The acknowledge trigger is associated with the signal RxTriggerEsc1[1].

Note: The TX signals - TxTriggerEsc1[2], TxTriggerEsc1[1], and TxTriggerEsc1[0] - and the RX signals - RxTriggerEsc1[3], and RxTriggerEsc1[0] - are not used by DSI protocol engine.

15.4.3.9.1 Reset

The DSI protocol engine can use one of the triggers of the DSI_PHY to send a reset to the display. The DSS.DSI_CTRL[5] TRIGGER_RESET bit is set to 1 to send the trigger reset. When the software requires the trigger reset to be sent to the peripheral, the DSI protocol engine should reset its own logic but not the registers. The software can select between two reset modes:

- Immediate reset: All pending requests in TX FIFO not already taken into account for transfer scheduling, the RX FIFO requests, and the data from video port are ignored. Only the current transfer on DSI link and already scheduled ones are transmitted. All the other transfers are discarded.
- Synchronized reset: The mode is only valid if there is VC using the video mode and if it is active. The principle is to wait for the current video frame to be transferred on the link. Any data on VP after the current frame are ignored.

To select the reset mode, software users must program the DSS.DSI_CTRL[14] TRIGGER_RESET_MODE.

CAUTION

For both reset modes, the hardware should flush the FIFOs, synchronization buffers, and line buffers before resetting the DSS.DSI_CTRL[0] IF_EN bit.

15.4.3.9.2 Tearing Effect

The TE on the display is avoided by having synchronization information from the display. It is used only in command mode. In case of video mode, it is not functional. Users are responsible for selecting the command mode for the VC using the TE feature.

The software must set and send the appropriate sequence to receive the TE trigger from the peripheral. When receiving the TE trigger, the DSI protocol engine should generate the TE_TRIGGER_IRQ interrupt with TE event if the interrupt is enabled. To enable the interrupt, set to 1 the DSS.DSI_IRQENABLE[16] TE_TRIGGER_IRQ_EN bit. The DSS.DSI_IRQSTATUS[16] TE_TRIGGER_IRQ status bit indicates if the interrupt event has been generated.

One or multiple VCs can be synchronized using the same TE trigger. The DSS.DSI_VCn_TE[30] TE_EN bit should be set to indicate that the hardware should use the following TE trigger to start the transfer of the data from the related VC. This bit is reset when all the data have been sent to the peripheral. The DSS.DSI_VCn_TE[31] TE_START bit should be used when the automatic mode enabled by setting the DSS.DSI_VCn_TE[30] TE_EN bit is not used. It allows users to start the transfer manually based on application events or based on the TE trigger interrupt (TE_TRIGGER_IRQ).

The number of bytes to be transferred is defined by using the DSS.DSI_VCn_TE[15:0] TE_SIZE bit field. The TE_SIZE bit field is decremented for each payload byte (it does not include Check-sum) sent on the DSI link. The register content should not be modified by software during a transfer. The DSS.DSI_VCn_TE[15:0] TE_SIZE bit field should be set first to indicate that the following accesses to DSS.DSI_VCn_LONG_PACKET_HEADER register should be used for TE transfer.

The data can be provided from two sources (selection by setting the DSS.DSI_VCn_CTRL[1] SOURCE bit):

- L4 interconnect port using DMA request: The DMA request DSI_DMA_REQi (where i =0 to 3) to should be asserted only when TE trigger is received or when the DSS.DSI_VCn_TE[31] TE_START bit is set by user and should not be asserted anymore when all the bytes defined in DSS.DSI_VCn_TE[15:0] TE_SIZE bit field have been sent on the DSI link. The VC is associated with a DMA request (from DSI_DMA_REQ0 to DSI_DMA_REQ3) by programming the number in the DSS.DSI_VCn_CTRL[23:21] DMA_TX_REQ_NB bit field. The DSS.DSI_VCn_LONG_PACKET_PAYLOAD register is used to provide the number of bytes defined by the DSS.DSI_VCn_TE[15:0] TE_SIZE bit field (the check-sum value is not provided in the DSS.DSI_VCn_LONG_PACKET_PAYLOAD register). The size of the header is not taken into account in the number of bytes to transfer. The DSS.DSI_VCn_SHORT_PACKET_HEADER register is not used.
- Video port: The DMA request is not asserted. The data are captured in the line buffer using the STALL mechanism. In case there is no line buffer instantiated (that is, DSS.DSI_CTRL[13:12] LINE_BUFFER bit field set to 0), it is not possible to use the video port to provide data. The line buffer should be filled up according to the word count defined in the DSS.DSI_VCn_LONG_PACKET_HEADER register header. The value should be written before the TE trigger event is received or before the DSS.DSI_VCn_TE[31] TE_START bit is set to 1 by software. In case the total number of bytes defined by the DSS.DSI_VCn_TE[15:0] TE_SIZE bit field is not a multiple of the word count defined in the DSS.DSI_VCn_LONG_PACKET_HEADER register, all the packets have the same size defined by the WC of the header except the last transfer. The size of the last transfer is defined by the remaining bytes to send. Since the DSS.DSI_VCn_TE[15:0] TE_SIZE bit field is modified after each packet transfer, the size of the last packet is equal to the value of DSS.DSI_VCn_TE[15:0] TE_SIZE bit field just before the last transfer (the header and the payload check-sum sizes are not included in DSS.DSI_VCn_TE[15:0] TE_SIZE bit field).

When the transfer is completed, the value of the DSS.DSI_VCn_TE[15:0] TE_SIZE bit field is equal to 0. The software must ensure that the pending data in the TX FIFO for the corresponding VC using TE are related to TE transfer. Any data in the TX FIFO that should be sent before reception of TE trigger should be sent before TE. This is done by not enabling TE trigger until all data for the corresponding VC have been sent to the peripheral. The software can check that the space allocated for the VC in the TX FIFO is empty by reading the DSS.DSI_VCn_CTRL[5] TX_FIFO_NOT_EMPTY status bit.

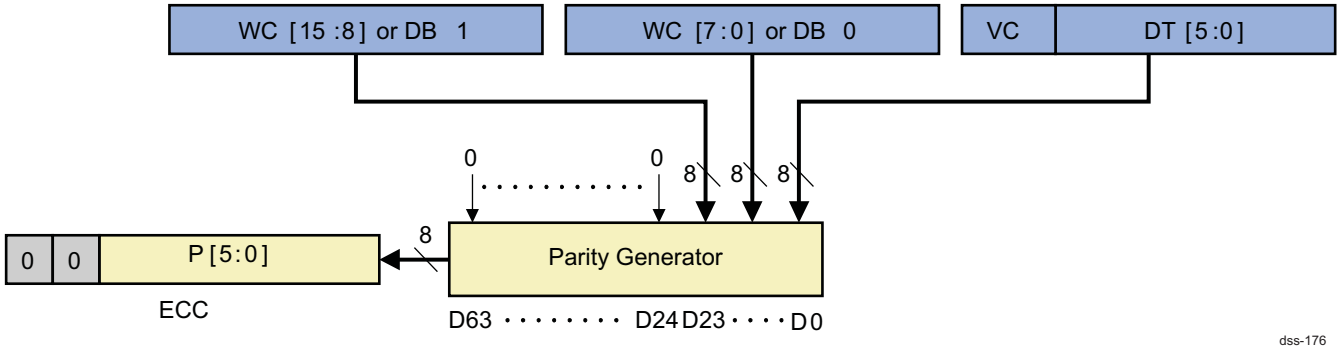
15.4.3.9.3 Acknowledge

The corresponding Acknowledge interrupt (ACK_TRIGGER_IRQ)should be generated upon reception of the acknowledge trigger. To enable the acknowledge interrupt, set the DSS.DSI_IRQENABLE[17] ACK_TRIGGER_IRQ_EN bit to 1. When the interrupt is generated, the DSS.DSI_IRQSTATUS[17] ACK_TRIGGER_IRQ status bit is set to 1.

15.4.3.10 ECC Generation

Note that the DSI protocol uses a four-byte packet header. Since ECC generation requires a fixed word length of 64-bits, the packet headers should be padded with additional bits to form a full eight-byte value for ECC generation and checking. The packet header less the ECC byte should occupy bits D[23:0] and the pad bits should occupy bits D[63:24]. All padding bits should be zero for the purpose of generating the ECC byte. ECC can be generated using a parallel approach as illustrated in Figure 15-102.

Figure 15-102. 64-Bit ECC Generation on TX Side



The ECC generation/check can be enabled and disabled by software. It is defined by a common bit for all the VCs:

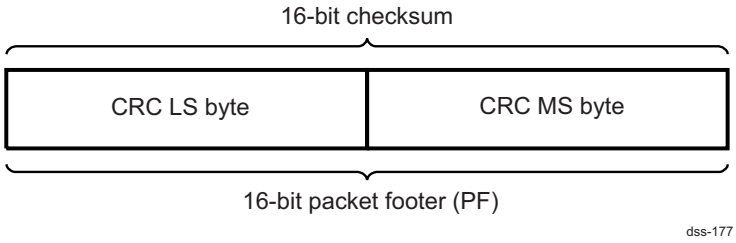
- The DSS.DSI_CTRL[2] ECC_RX_EN bit enables/disables the ECC generation in the receive direction.
- The DSS.DSI_VCn_CTRL[8] ECC_TX_EN bit enables/disables the ECC generation in the transmit direction

15.4.3.11 Checksum Generation for Long Packet Payloads

Long packets are comprised of a packet header protected by an ECC byte and a payload of 0 to 2^{16} - 1 bytes. To detect the errors during the transmission of long packets, a checksum is calculated over the payload portion of the data packet. Note that, for the special case of a zero-length payload, the 2-byte checksum is set to 0xFFFF. The checksum can only indicate the presence of one or more errors in the payload. Unlike ECC, the checksum does not enable error correction. For this reason, checksum calculation is not useful for some unidirectional DSI implementations since the peripheral has no way for reporting errors to the host processor. Checksum generation and transmission is mandatory for host processors sending long packets to peripherals. It is optional for peripherals transmitting long packets to the host processor. However, the format of long packets is fixed; the peripherals that do not support checksum generation should transmit two bytes having value 0x0000 in place of the checksum bytes when sending long packets to the host processor. The host processor should disable checksum checking for received long packets from peripherals that do not support checksum generation.

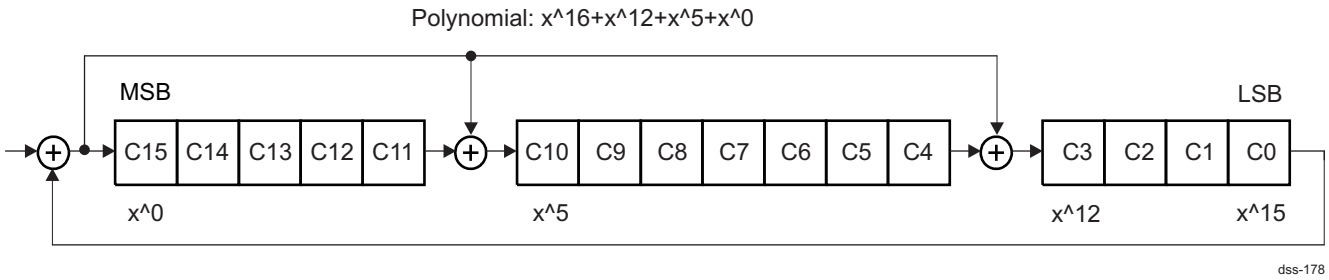
The checksum should be realized as a 16-bit CRC with a generator polynomial of x^16+x^12+x^5+x^0. The LS byte is sent first, followed by the MS byte. Note that within the byte, the LS bit is sent first.

Figure 15-103. Checksum Transmission



The CRC implementation is presented in Figure 15-104. The CRC shift register is initialized to 0xFFFF before packet data enters. Packet data not including the packet header then enters as a bitwise data stream from the left, LS bit first. Each bit is fed through the CRC shift register before it is passed to the output for transmission to the peripheral. After all bytes in the packet payload have passed through the CRC shift register, the shift register contains the checksum. C15 contains the checksums MSB and C0 the LSB of the 16-bit checksum. The checksum is then appended to the data stream and sent to the receiver.

Figure 15-104. 16 Bit CRC Generation Using a Shift Register



The check-sum generation/check can be enabled and disabled by software. It is defined by a common bit for all the VCs:

- The DSS.DSI_CTRL[1] CS_RX_EN bit enables/disables the check-sum generation in the receive direction.
- The DSS.DSI_VCn_CTRL[7] CS_TX_EN bit enables/disables the check-sum generation in the transmit direction

15.4.3.12 End of Transfer Packet

To allow the DSI protocol (rather than the DSI_PHY) at the display to detect the HS End Of Transfer (EOT), an EOT packet type is added. It is a fixed short packet (4 bytes) that is added at every HS-to-LP transition. This function is enabled by the DSI_CTRL[19] EOT_ENABLE bit.

The EOT packet has a fixed format:

- Data Type = DI [5:0] = 0b001000
- Virtual Channel = DI [7:6] = 0b00
- Payload Data [15:0] = 0x0F0F
- ECC [7:0] = 0x01

When more than one data lane is used, the bytes in the EOT packet are distributed across multiple lanes. EOT packet generation is supported only for the end of HS transmissions. No EOT packet is added at the end of LP transmissions. For LP reception, any EOT packet received is simply passed through the same as any other packet, but no internal decode or use is made of the EOT information.

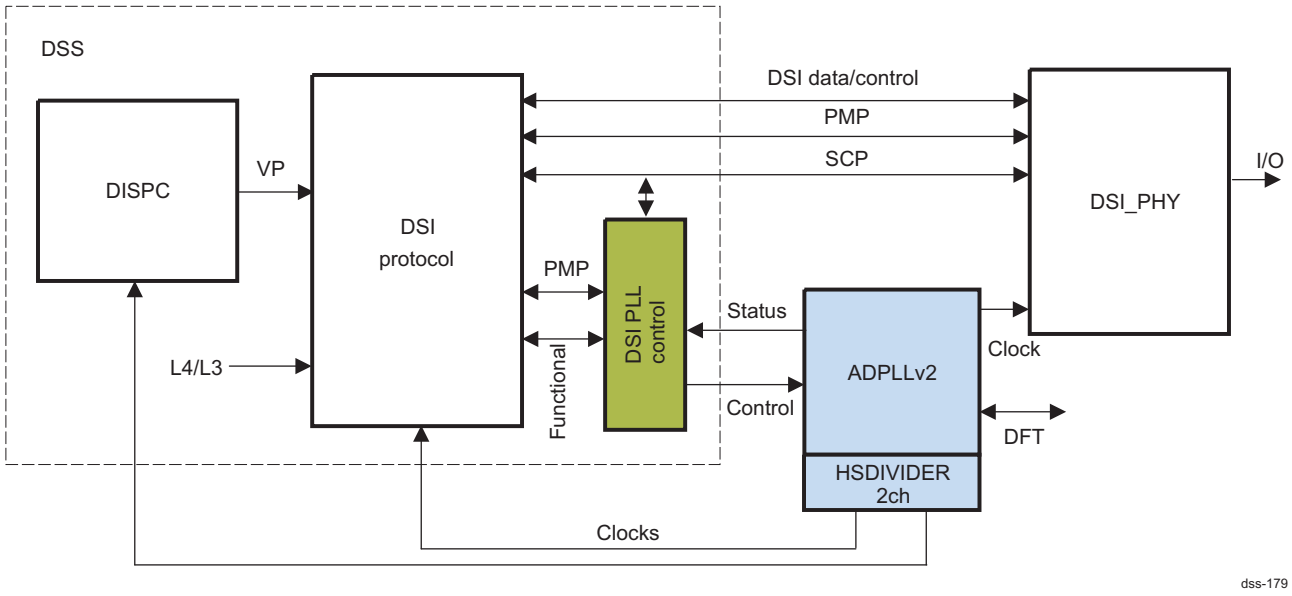
15.4.4 DSI PLL Controller Functionalities

15.4.4.1 DSI PLL Controller Overview

The DSI PLL controller module forms part of the display sub-system. Nevertheless, it uses the SCP (Serial Configuration Port) and PMP (Power Management Port) ports as the primary interfaces to the DSI protocol engine. The SCP interface is used to set the configuration of the DPLL and HSDIVIDER modules, primarily the various counter values. The PMP port is used to control the power state of the DPLL and HSDIVIDER modules. Figure 15-105 provides an overview of the DSI PLL controller module inside the display subsystem.

The DSI PLL is also used to generate the 74.25-MHz frequency used for HDTV applications.

Figure 15-105. DSI PLL Controller Overview



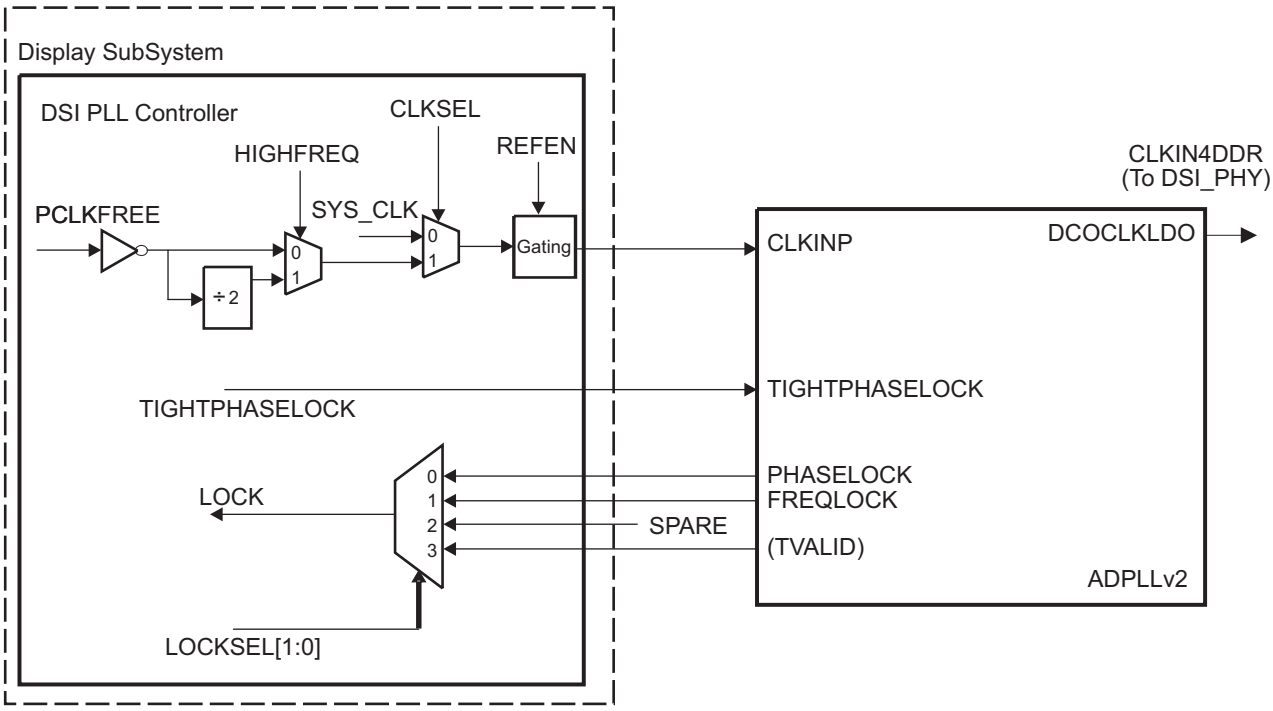
Note: The DSI PLL controller module does not have an interface to L4 interconnect. The programmable features are managed by registers mapped into the DSI protocol engine.

15.4.4.2 DSI PLL Controller Architecture

The DSI PLL is an ADPLLv2 module. The pixel clock (PCLK) frequency range is 2 to 65 MHz. This may be divided by 2. This is performed by setting the DSS.DSI_PLL_CONFIGURATION2[12] DSI_PLL_HIGHFREQ bit to 1.

Figure 15-106 shows the internal DSI PLL reference diagram.

Figure 15-106. DSI PLL Reference Diagram



Note: PCLK is inverted as the falling edge is the reference edge and ADPLLv2 uses positive edge as reference (F/F should be positive edge clock also)

dss-180

The DSI PLL clock output corresponds to the CLKIN4DDR clock of the DSI complex I/O module.

The DSI PLL reference clock is the DSI_PLL_REFCLK clock. Depending on the setting in DSS.DSI_PLL_CONFIGURATION2[11] DSI_PLL_CLKSEL bit, the reference clock can be either the DSS2_ALWON_FCLK provided by the PRCM or the PCLKFREE provided by the DISPC module.

15.4.4.3 DSI PLL Operations

The DSI PLL configuration signals operate according to Table 15-35. Table 15-35 indicates the operation when the PLL is not locked.

Table 15-35. DSI PLL Operation Modes When Not Locked

DSI PLL Operation Mode	Stop mode Low power[1]	Stop mode Fast Relock[1]	Idle bypass
Mode Description	**Output clocks stopped Lowest power standby**	**Output clocks stopped Fastest start-up time**	**Selects when PLL and HSDIVIDER bypass clocks are used**
DSS.DSI_PLL_CONFIGURATION2[0] DSI_PLL_IDLE	0	0	1
DSS.DSI_PLL_CONFIGURATION2[6] DSI_PLL_LOWCURRSTBY	1	0	1
DSS.DSI_PLL_CONFIGURATION1[0] DSI_PLL_STOPMODE	1	1	X

(1) Recommended

When locked, the PLL output frequency is: [(2 x REGM)/(REGN + 1)] x [CLKin(MHz)/(HIGHFREQ + 1)] where:

- M multiplier is programmed in DSS.DSI_PLL_CONFIGURATION1[18:8] DSI_PLL_REGM bit field.
- N divider is programmed in DSS.DSI_PLL_CONFIGURATION1[7:1] DSI_PLL_REGN bit field.
- HIGHFREQ divider by 2 is enabled by setting the DSS.DSI_PLL_CONFIGURATION2[12]

DSI_PLL_HIGHFREQ bit to 1.

15.4.4.4 DSI PLL Controller Shadowing Mechanism

The configuration registers are accessed through the DSI protocol engine register space using SCP port. This includes all the configuration signals and the returning status signals.

CAUTION

All writes must be 32-bit operations as the SCP always transfers 32 bits. Any 16-bit or 8-bit operations may lead to unpredictable errors.

A shadow mechanism is implemented for appropriate register values so that configurations may optionally be updated in synchronism with the display controller (DISPC) and DSI protocol engine. The front porch time from the DISPC indicates the time when making the update of the value. All the required updated values must have been written before this signal is asserted. Refer to Section 15.4.3.5.1 for more details.

15.4.4.5 Error Handling

The PLL lock and recalibration signals may be monitored to detect loss of lock or requirement to recalibrate (due to large temperature change since the last lock request):

- The DSS.DSI_PLL_STATUS[1] DSI_PLL_LOCK status bit gives the DSI PLL lock state.
- The DSS.DSI_PLL_STATUS[2] DSI_PLL_RECAL status bit informs if the PLL must be uncalibrated

These signals can also generate interrupts at DSI protocol engine level:

- The PLL_LOCK_IRQ interrupt indicates that the DSI PLL control module has sent a lock request to the DSI PLL. To monitor this event, read the DSS.DSI_IRQSTATUS[7] PLL_LOCK_IRQ bit. Set this bit to 1 to clear the status bit.
- The PLL_UNLOCK_IRQ interrupt indicates that the DSI PLL control module has sent an unlock request to the DSI PLL. To monitor this event, read the DSS.DSI_IRQSTATUS[8] PLL_UNLOCK_IRQ bit. Set this bit to 1 to clear the status bit.
- The PLL_RECAL_IRQ interrupt indicates that the DSI PLL control module has sent a recalibration request to the DSI PLL. To monitor this event, read the DSS.DSI_IRQSTATUS[9] PLL_RECAL_IRQ bit. Set this bit to 1 to clear the status bit.

15.4.5 DSI Complex I/O Functionalities

15.4.5.1 DSI Complex I/O Overview

DSI_PHY is a complex I/O with 3 unidirectional (HS) Lane Modules. This includes 2 data lane modules and 1 clock lane module. Each lane module has 2 data pads (DX, DY). These data pads are connected with a complementary lane module on the DSI receiver device using point to point interconnect.

Lane modules support high-speed burst mode. Forward direction and reverse direction escape modes are also supported. Escape modes maybe used for Low Power Data Transmission, among other things.

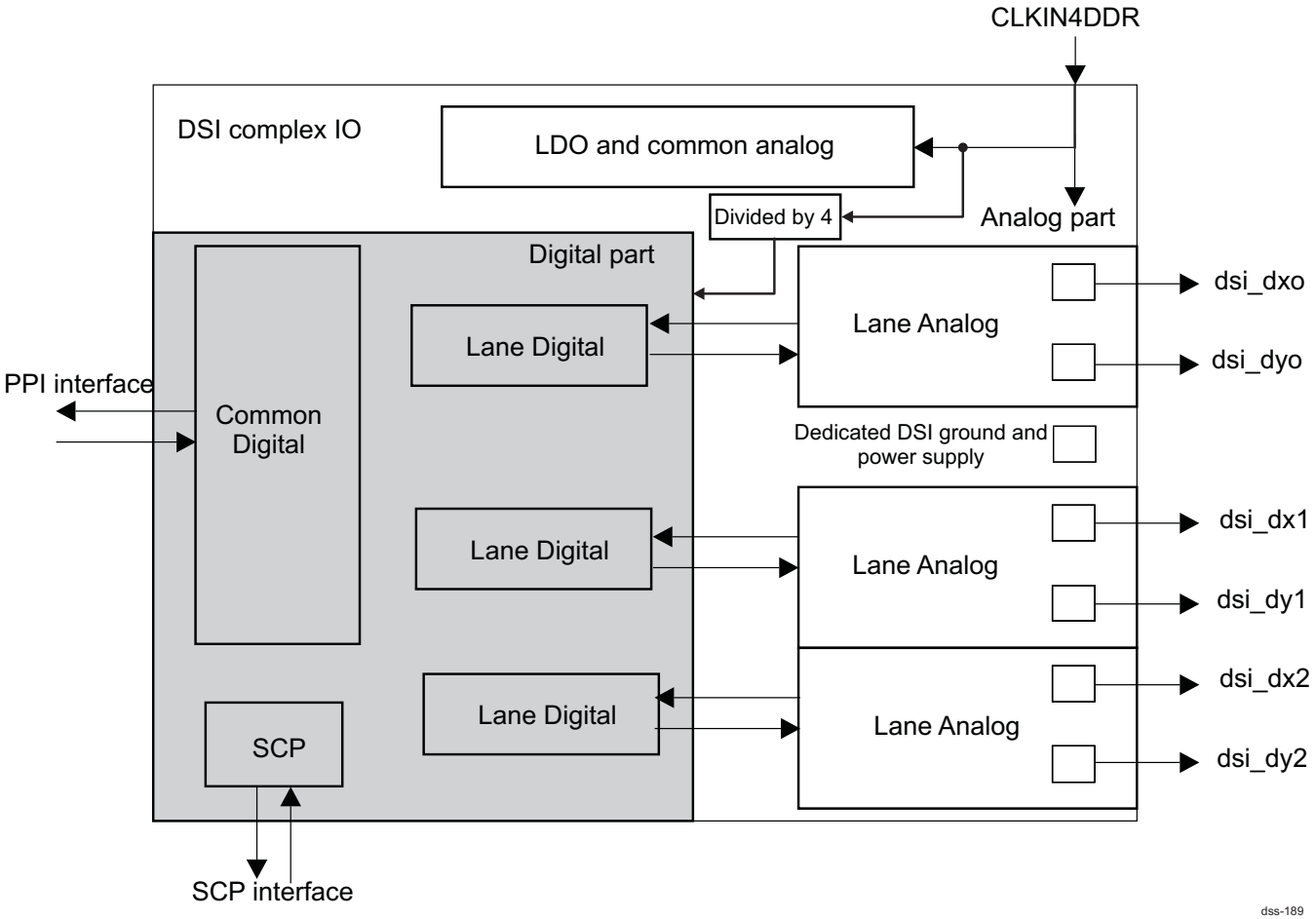
The maximum data rate supported in high-speed Mode is 800Mbps per data lane. The lane module function and position is configurable, that is, any lane module can be chosen as clock lane module, and DX/DY data pad for each lane module can be configured as either DP or DN pins defined by DSI_PHY spec.

DSI_PHY interacts with the higher layers of the DSI link through the PHY-Protocol Interface (PPI). DSI_PHY does not include a PLL; a high frequency clock input is expected in HS mode (CLKIN4DDR). DSI_PHY supports also Serial Configuration Protocol (SCP) to set various configuration and control registers. The DSI_PHY supports GPIO operation on each of the six data pins.

15.4.5.2 DSI Complex I/O Architecture

Figure 15-107 shows the top-level block diagram of DSI_PHY. It has 3 lane module functions, and some common analog and digital functions. Each lane module has an analog and a digital part. The digital lane circuitry implements the protocol interface of MIPI DSI_PHY. It interacts with the DSI protocol engine.

Figure 15-107. DSI Complex I/O Architecture



The lane module analog circuit converts the digital levels to the LP and HS level signals for the link. It also converts the analog state of the link into digital levels.

For high-speed signals, the lane module analog also does the last stage untiming of HS data to obtain optimum quadrature with the HS clock.

DSI_PHY has an internal LDO to generate 1.2V supply. This LDO requires an off-chip decoupling capacitor and hence a dedicated pin. DSI_PHY expects a high frequency clock input in HS mode.

The clock input is required to have four times the frequency of the DDR clock, that is, twice the HS data rate. GPIO functions are supported on each dsi_dxi and dsi_dyi.

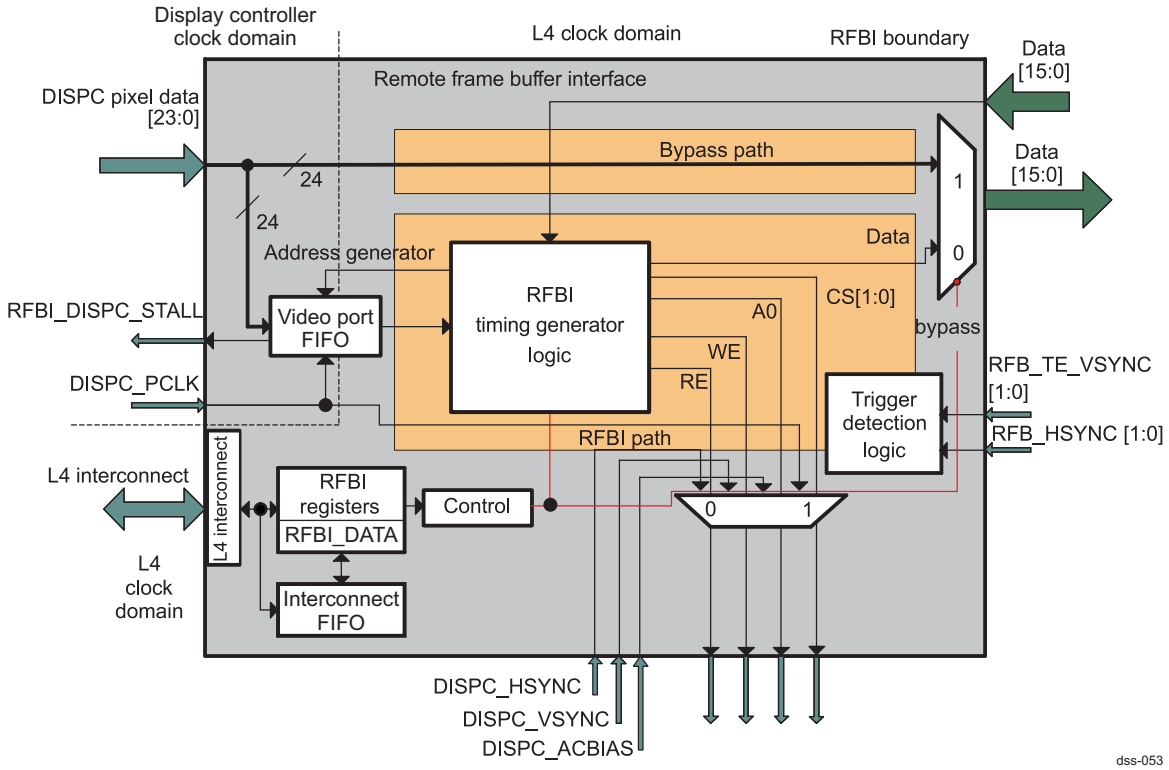
15.4.6 RFBI Functionalities

The RFBI module can capture the output pixel from the display controller and send the data to the RFB in the LCD panel. The application configures the RFBI module, sends commands, reads data, and configures the display controller to send data fetched from the system memory by the display controller DMA engine. The commands/data are sent using an 8-, 9-, 12-, or 16-bit parallel interface.

The display controller is configured to send the data in 12-, 16-, 18-, or 24-BPP format. In the video port FIFO, the encoded pixel values are in an LSB alignment independently of the endianness in system memory.

Figure 15-108 shows an overview of the RFBI architecture.

Figure 15-108. RFBI Architecture Overview



15.4.6.1 RFBI FIFO

The input video port FIFO receives data from the display controller at the pixel clock. The data in the video port FIFO are read by the RFBI and are sent to the LCD panel. The video port FIFO is 24 bits wide and each pixel in 12-, 16-, 18-, and 24-BPP format is stored in the video port FIFO using one 24-bit value aligned on the 24-bit LSB. Section 15.4.6.4, *Output Parallel Modes*, shows an example of an output configuration based on the interface width (16 bits) and the pixel format output (24 bits).

15.4.6.2 RFBI Interconnect FIFO

The interconnect FIFO receives the data from RFBI_DATA write requests to the L4 interconnect slave port. The data in the interconnect FIFO are read by the RFBI and sent to the LCD panel. The width of the interconnect FIFO is 32 bits. The size of the interconnect FIFO is 24 words of 32 bits (that is, 24 words of RFBI_DATA).

15.4.6.3 Input Pixel Formats

The supported pixel formats in the RFBI module are: RGB24-888, RGB18-666, RGB16-565, and RGB12-444 as output from the display controller and from the L4 (for writing parameters). In both cases, the pixels are formatted in accordance with the configuration of the output interfaces (multiple cycles).

15.4.6.4 Output Parallel Modes

The RFBI output modes are 8-, 9-, 12-, and 16-bit interfaces. Any mode can be selected regardless of the pixel format. Set the right configuration in the cycle registers to define a valid configuration for each output cycle.

The following example is an output configuration based on the 16-bit interface width and the 24-bit pixel format (i = 0 or 1) (see also Table 15-36):

- The DSS.RFBI_CONFIGi[10:9] CYCLEFORMAT bit field is set to 0x3 (three cycles for two pixels).
- The DSS.RFBI_DATA_CYCLE1_i register is set to 0x00000010 (16 bits from pixel 1 for the first cycle).
- The DSS.RFBI_DATA_CYCLE2_i register is set to 0x00080808 (8 bits from pixel 1 and pixel 2 and alignment of 8 bits from pixel 2 for the second cycle).
- The DSS.RFBI_DATA_CYCLE3_i register is set to 0x00100000 (16 bits from pixel 2 for the third cycle).

Table 15-36. 16-Bit Interface Configuration/24-Bit Mode

	24-Bit Mode		
	1st Cycle	**2nd Cycle**	**3rd Cycle**
Data[15]	R0[7]	B0[7]	G1[7]
Data[14]	R0[6]	B0[6]	G1[6]
Data[13]	R0[5]	B0[5]	G1[5]
Data[12]	R0[4]	B0[4]	G1[4]
Data[11]	R0[3]	B0[3]	G1[3]
Data[10]	R0[2]	B0[2]	G1[2]
Data[9]	R0[1]	B0[1]	G1[1]
Data[8]	R0[0]	B0[0]	G1[0]
Data[7]	G0[7]	R1[7]	B1[7]
Data[6]	G0[6]	R1[6]	B1[6]
Data[5]	G0[5]	R1[5]	B1[5]
Data[4]	G0[4]	R1[4]	B1[4]
Data[3]	G0[3]	R1[3]	B1[3]
Data[2]	G0[2]	R1[2]	B1[2]
Data[1]	G0[1]	R1[1]	B1[1]
Data[0]	G0[0]	R1[0]	B1[0]

15.4.6.5 Unmodified Bits

In a cycle, if every bit in the interface does not have a pixel value, the status of the unused bits can be programmed to be 0, 1, or the previous value (I/O power consumption optimization).

15.4.6.6 Bypass Mode

In bypass mode, the RFBI path is bypassed and the display controller data and signals are sent directly to the output interface of the RFBI.

15.4.6.7 Send Commands

The commands are written through the L4 interconnect and into the DSS.RFBI_CMD register. After a command is sent, another one can be accepted by the module and set. If the processing of a command is not complete, the MPU access to change the command stalls.

15.4.6.8 Read/Write

Depending on the status of A0, WE, and RE, the commands and display/parameter data are written to the panel (handled by the state-machine for the commands/parameter data and stored in memory for the display data), or the display data/status values are read from the LCD panel (status and display data in the LCD panel memory). The polarity of A0 (RFBI_A0 signal), WE (RFBI_WE signal), RE (RFBI_RE signal), and CSx (RFBI_CSx signal, with x = 0, 1) is programmable.

Table 15-37 describes the read/write function.

Table 15-37. Read/Write Function Description

A0 (RFBI_A0)	WE (RFBI_WE)	RE (RFBI_RE)	Function Description
1	0	1	Display data write, parameter data write
1	1	0	Display data read
0	1	0	Status read
0	0	1	Command data write

A minimum of RFBI_Cs cycle time, as defined in Table 15-38, is required to keep the RFBI_CSx signal asserted between write transfers of multiple pixels.

Table 15-38 indicates the minimum cycle time for RFBI_CSx, depending on the source of pixels (display controller or L4 interconnect slave port) and the cycle format (1pixel/cycle, 1 pixel/2 cycles, 1 pixel/3 cycles, or 2 pixels/3 cycles).

Table 15-38. Minimum Cycle Time for CSx/WE Always Asserted

RFBI Performance	RFBI_CONFIGi[10:9] CYCLEFORMAT	RFBI_CONFIGi[8:7] L4FORMAT	Minimum Cycle Time (in Number of L4 Cycles)
L4 interconnect	1 pixel/cycle	1 pixel	5
	1 pixel/2 cycles	1 pixel	4
	1 pixel/3 cycles	1 pixel	4
	2 pixels/3 cycles	1 pixel	6
	1 pixel/cycle	2 pixels	4
	1 pixel/2 cycles	2 pixels	4
	1 pixel/3 cycles	2 pixels	4
	2 pixels/3 cycles	2 pixels	6
Display Controller	1 pixel/cycle	N/A	4
	1 pixel/2 cycles	N/A	3
	1 pixel/3 cycles	N/A	3
	2 pixels/3 cycles	N/A	6

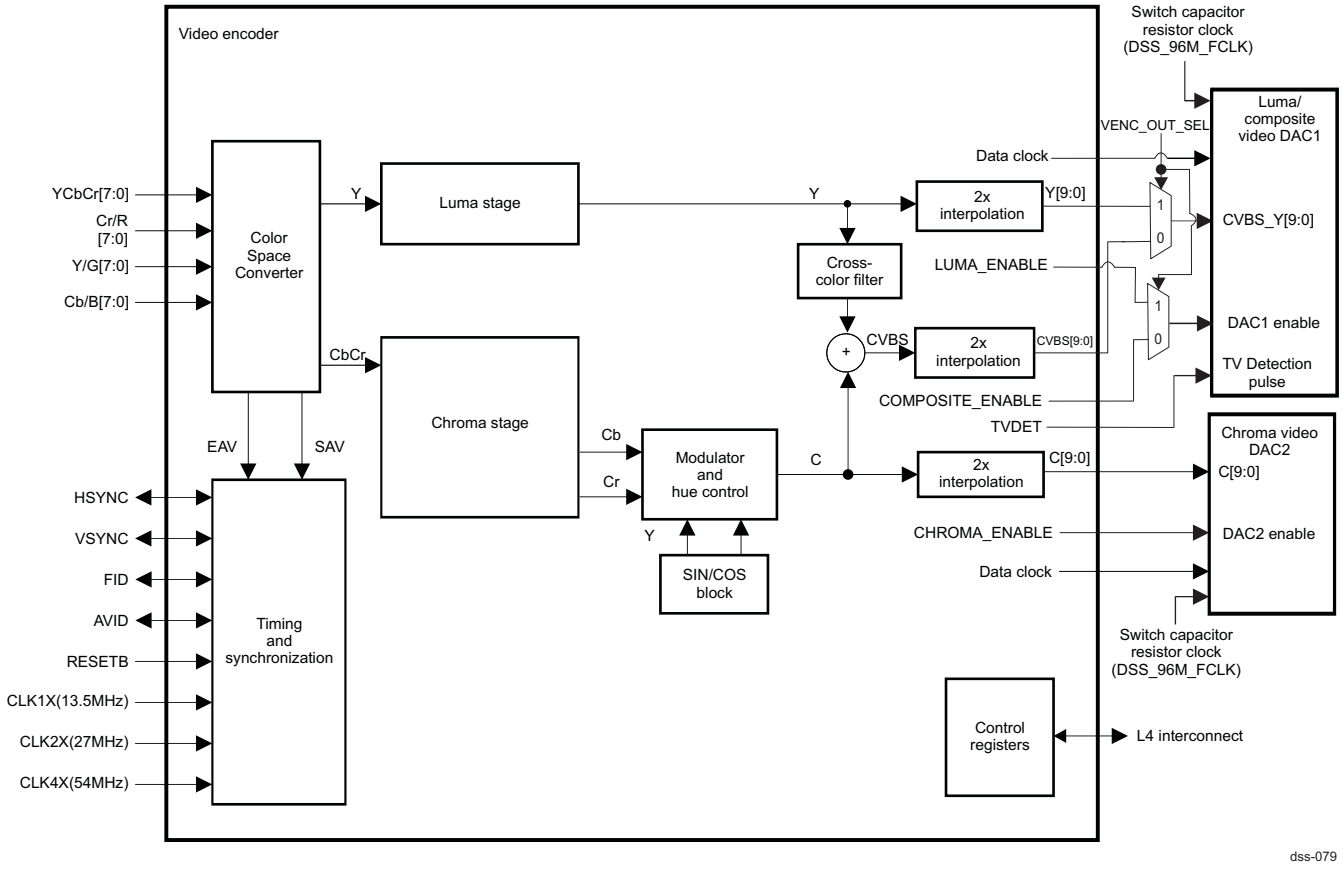
15.4.7 Video Encoder Functionalities

The input formats supported by the encoders are 24-bit 4:4:4 RGB. The encoder output is the DAC (for more information, see Section 15.2, *Display Subsystem Environment*). In the display subsystem, the input format from the display controller is always 24-bit RGB. The RGB-to-YCbCr color space converter converts the 24-bit RGB pixel data to 24-bit YCbCr data.

The remaining Cb and Cr color components enter the 2-to-1 chrominance decimation, which reduces by half the chrominance bandwidth and the amount of chrominance data. After the data manager, the encoder processes in 4:2:2 data path up to the 2x interpolation. A luma delay synchronizes luma to chrominance data.

Figure 15-109 shows an overview of the video encoder architecture.

Figure 15-109. Video Encoder Architecture Overview



Note: Output video mode can be either composite video (CVBS output) or Separate video (s-video: Luma and chroma outputs):

- Composite video: DAC1 is used
- Separate video (luma/chroma): Both DAC1 (luma) and DAC2 (chroma) are used

  The selection is programmed with DSS.DSS_CONTROL[6] VENC_OUT_SEL bit. Composite video is the default selection.

15.4.7.1 Test Pattern Generation

For diagnostic purposes, the data manager can be forced to output 100/100 color bar RGB/YCbCr data by setting the SVDS field VENC_F_CONTROL[7:6] register to 0x1.

Table 15-39. 100/100 Color Bar Table

COLOR	R	G	B	Y	Cb	Cr
White	255	255	255	235	128	128
Yellow	255	255	0	210	16	146
Cyan	0	255	255	170	166	16
Green	0	255	0	145	54	34
Magenta	255	0	255	106	202	222
Red	255	0	0	81	90	240
Blue	0	0	255	41	240	110
Black	0	0	0	16	128	128

15.4.7.2 Luma Stage

The luma stage includes a luma pipeline delay, luma shaping, 2x interpolation filter, and luma variable delay. The luma pipeline delay block is used to match luma path length to chroma path length. In the luma gain shaper, a programmable gain is first applied to the luminance data output. The luminance gain is defined by the DSS.VENC_GAIN_Y register. Horizontal sync, vertical sync, and setup insertion are then performed.

Black level and blank level are programmable through the DSS.VENC_BLACK_LEVEL and DSS.VENC_BLANK_LEVEL registers. All the transition edges of the luminance signal, such as sync edges and active video edges, are properly shaped and filtered to keep the bandwidth within the standards.

After all required components of the luminance signal are added, the resulting signal is low-passed and interpolated to 2x-pixel rate. This 2x interpolation simplifies the external analog reconstruction filter design and improves the signal-to-noise ratio.

15.4.7.3 Chroma Stage

The chroma stage includes a low-pass filter, first-stage 2x interpolation, chroma gain shaper, and second-stage 2x interpolation. A pair of programmable gains adjusts the time-multiplexed U/V signal. The gains for U and V are independently controlled by the DSS.VENC_GAIN_U and DSS.VENC_GAIN_V register bits.

15.4.7.4 Subcarrier and Burst Generation

The encoder uses a 32-bit subcarrier increment to synthesize the subcarrier. The value of the subcarrier increments required to generate the desired subcarrier frequency for NTSC and PAL format is found by:

$S_CARR = ROUND ([F_{sc}/F_{clkenc}] \times 2^{32})$

where:

F_{sc} = Frequency of the subcarrier

F_{clkenc} = Frequency of the internal video encoder

The DSS.VENC_S_CARR register controls the subcarrier frequency. The DSS.VENC_C_PHASE register controls the phase of the subcarrier. The phase of the color subcarrier is reset to DSS.VENC_C_PHASE. Table 15-40 presents the VENC_S_CARR register values depending the standard and pixel type used.

Table 15-40. VENC_S_CARR Register Recommended Values

Standard	Pixel Type	Subcarrier Frequency (Fsc) (MHz)	Fclkenc (MHz)	VENC_S_CARR register value (hexa)
NTSC-M, J	ITU-R601	3.579545	27	0x21F07C1F
PAL-M	ITU-R601	3.5756083125	27	0x21E6EFE3
PAL-B, D, G, H, I	ITU-R601	4.43361875	27	0x2A098ACB
NTSC-M, J	Square pixel	3.579545	24.5454	0x25555555
PAL-M	Square pixel	3.579561149	24.5454	0x1F15C01E
PAL-B, D, G, H, I	Square pixel	4.43361875	29.50	0x26798C0C

CAUTION

In square pixel mode, an external clock generator is needed to provide sampling frequencies (49.09 MHz for NTSC square pixel or 59 MHz for PAL square pixel).

The color subcarrier reset has four modes:

- No reset
- Reset every two lines
- Reset every two fields
- Reset every eight fields

The DSS.VENC_C_PHASE register can be used to adjust the SCH (subcarrier to horizontal sync phase). The DSS.VENC_BSTAMP_WSS_DATA[6:0] BSTAP bit field sets the amplitude of the color burst. The DSS.VENC_M_CONTROL[1] PAL bit enables phase alternation line encoding.

A phase switching subcarrier is generated to encode the chrominance signal when the DSS.VENC_M_CONTROL[1] PAL bit is set to 1. Otherwise, a normal subcarrier is generated. Phase alternation line refers to the encoding scheme in which the subcarrier alternates between two phases every scan line. Two possible alternation sequences are possible, and the DSS.VENC_M_CONTROL[5] PALPHS bit selects one of these sequences.

15.4.7.5 Closed Caption Encoding

The encoder can be programmed to encode closed-caption data and extended data in the selected line. The closed-caption data are sent to the encoder through the L4 interconnect. The data stream consists of 7-bit US-ASCII code and 1 odd-parity bit (see Table 15-41).

Table 15-41. Closed-Caption Data Format

MSB							LSB
Odd parity	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0

The standard service encodes closed caption in both fields; the extended service encodes closed caption in even fields. When set to 1, the DSS.VENC_L21_WC_CTL L21EN[0] bit enables closed-caption encoding in odd fields. When set to 1, the DSS.VENC_L21_WC_CTL L21EN[1] bit enables closed-caption encoding in even fields.

There is a 4-line offset between the value programmed in the DSS.VENC_LN_SEL[4:0] SLINE bit field and the scan line where closed caption is to be encoded. Program the value 0x11 to activate the closed caption on line 21. The DSS.VENC_LN_SEL[25:16] LN21_RUNIN bit field should be kept at reset value (0x10B). Four closed-caption data registers contain the data to be encoded. The DSS.VENC_LINE21[7:0] L21O and DSS.VENC_LINE21[15:8] L21O bit fields contain the first and the second bytes of closed-caption data to be encoded in the odd field. The DSS.VENC_LINE21[23:16] L21E and DSS.VENC_LINE21[31:24] L21E bit fields contain the first and the second bytes of data to be encoded in the even field.

Immediately after the closed-caption data is written to the registers, in either the odd field or even field, the corresponding closed-caption status bit (DSS.VENC_STATUS[4] CCE or DSS.VENC_STATUS[3] CCO) is reset to 0 to indicate that the closed-caption data is available in the closed-caption data registers and yet to be encoded.

Immediately after the closed-caption data is encoded, the DSS.VENC_STATUS[4] CCE bit or the DSS.VENC_STATUS[3] CCO bit is set to 1 to indicate that the closed-caption data has been encoded and is ready to accept new data. As seen in Figure 15-110, a null character is automatically inserted if the closed-caption data is not written to the closed-caption data registers in time for encoding.

The running clock frequency is controlled by the DSS.VENC_CC_CARR_WSS_CARR[15:0] FCC bit field which should be kept at reset value (0x2631) to get 5034960.5Hz (32xfline) for NTSC-601. The closed-caption running clock common frequencies are detailed in Table 15-42.

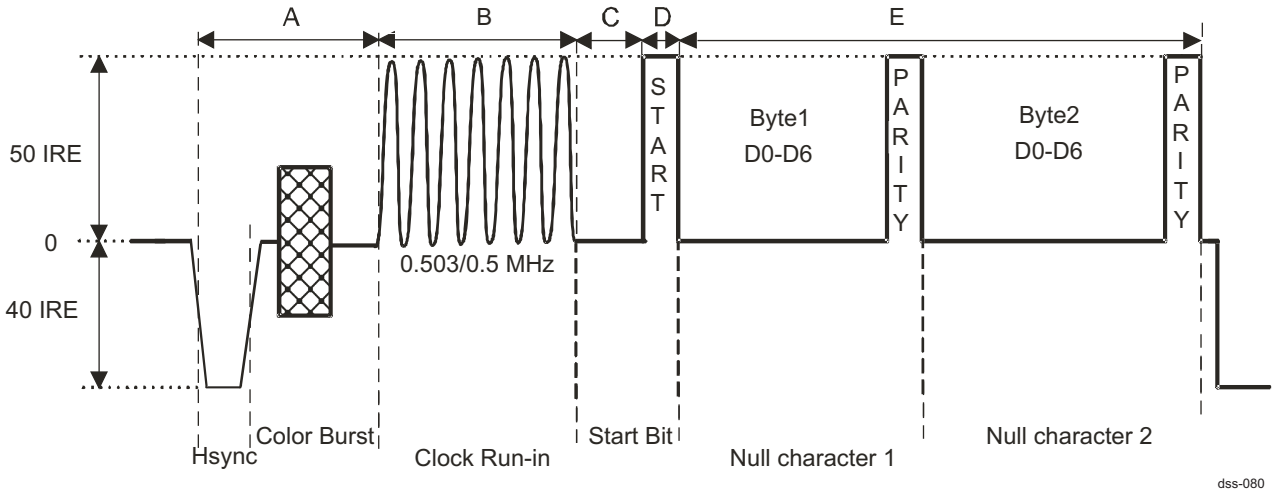
Table 15-42. Closed-Caption Run Clock Frequency Settings

	NTSC-601	PAL-601	NTSC Square Pixel	PAL Square Pixel
VENC_CC_CARR_WSS_CARR[15:0] FCC bit field value	0x2631	0x25ED	0x2A03	0x22B6

The closed-caption data is encoded in nonreturn-to-zero (NRZ) format. Additionally, the data translates to the IRE scale as follows: 0 = 0 IRE; 1 = 50 IRE.

Figure 15-110 shows the parameters of closed-caption line data implemented in different standards.

Figure 15-110. Closed Captioning Timing



Notes:

- The interval A is controlled by the DSS.VENC_LN_SEL[25:16] LN21_RUNIN bit field.
- The interval B is controlled by DSS.VENC_CC_CARR_WSS_CARR[15:0] FCC bit field.

Table 15-43. Closed-Caption Standard Timing Values

Intervals	Description	Timing Values for Encoding			Timing Values for Decoding		
		Minimal	Nominal	Maximal	Lower Bound	Nominal	Upper Bound
A	HSYNC to clock running	10.250 µs	10.500 µs	10.750 µs	10.000 µs	10.500 µs	11.000 µs
B	Clock running		12.910 µs			12.910 µs	
C	Clock running to third start bit		3.972 µs			3.972 µs	
D	Start bit		1.986 µs			1.986 µs	
E	Data characters		31.778 µs			31.778 µs	

Note: All timing values listed in Table 15-43 are measured from the mid-point (half amplitude) on all edges.

For a complete description of copy protection including CGMS-A, please refer to CEA-608-x standard

15.4.7.6 Wide-Screen Signaling (WSS) Encoding

The encoder can embed data, encoded in accordance with the IEC61880 and ITU-R 1119 data insertion standard, within the vertical blanking interval.

The encoder supports WSS data insertion on line 20 of every frame in the NTSC format. WSS data insertion is enabled by activating the DSS.VENC_L21_WC_CTL[14:13] EVEN_ODD_EN bit and by programming the VENC_BSTAMP_WSS_DATA[27:8] WSS_DATA bit field.

The running clock frequency is controlled by the DSS.VENC_CC_CARR_WSS_CARR[31:16] FWSS bit field. The wide-screen signaling running clock common frequencies are detailed in Table 15-44

Table 15-44. Wide-Screen Signaling Run Clock Frequency Settings

	NTSC-601	PAL-601	NTSC Square Pixel	PAL Square Pixel
VENC_CC_CARR_WSS_CARR[31:16] FWSS bit field value	0x043F	0x2F72	0x04AC	0x2B6D

To select the line where the WSS data are encoded, program the DSS.VENC_L21_WC_CTL[12:8] LINE bit field.

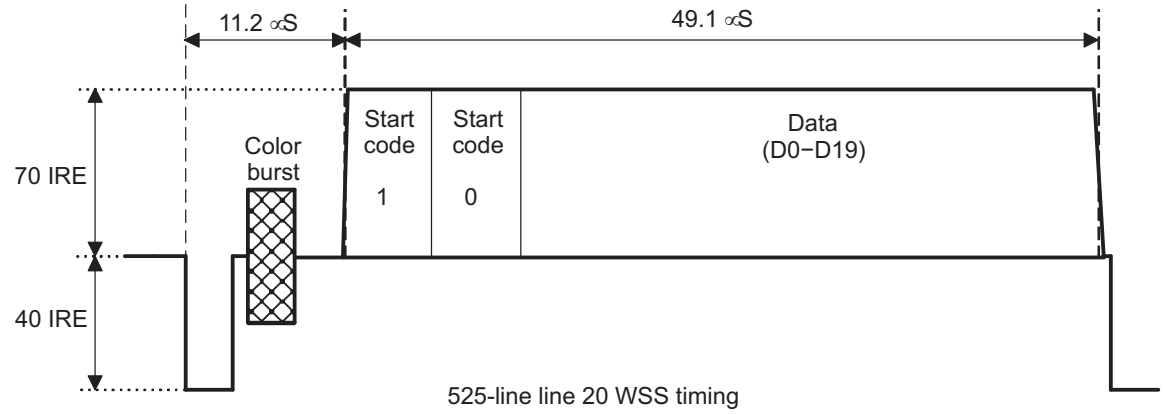
CAUTION

The setting of the LINE[4:0] bit field value depends on the video standard:

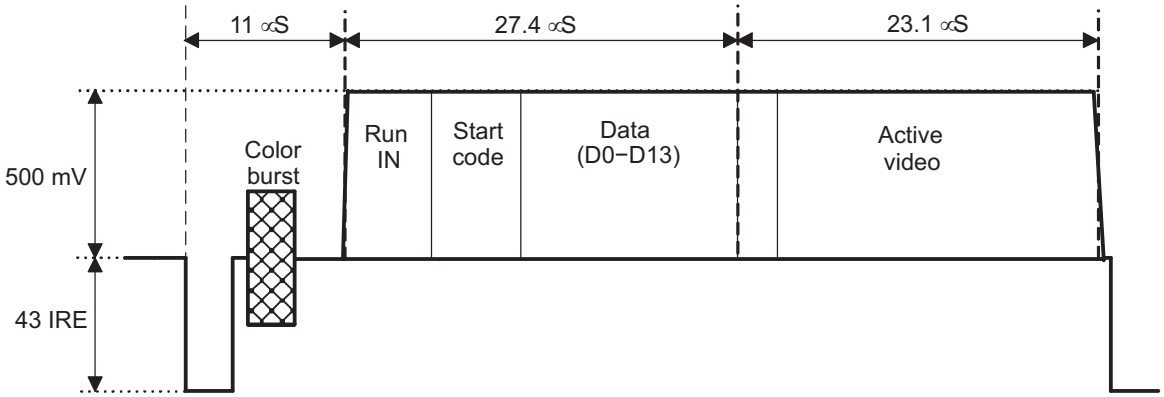
- PAL mode: There is an one line offset, so program the wanted line number - 1. The recommended value is line 0x16 + 1 = 0x17 (23rd line). Note that the default value is 0x14 + 1 = 0x15 (21st line).
- NTSC mode: There is a four line offset, so program the wanted line number - 4. The recommended value is line 0x10 + 4 = 0x14 (20th line). Note that the default value is 0x14 + 4 = 0x18 (24th line).

The WSS encoding block assumes that a full 10-bit video range is used to determine the 70 percent of peak-white amplitude of a logic-1 bit. The encoder also supports WSS data insertion on line 23 in the PAL format. Both waveforms are shown in Figure 15-111.

Figure 15-111. WSS Timing



525-line line 20 WSS timing



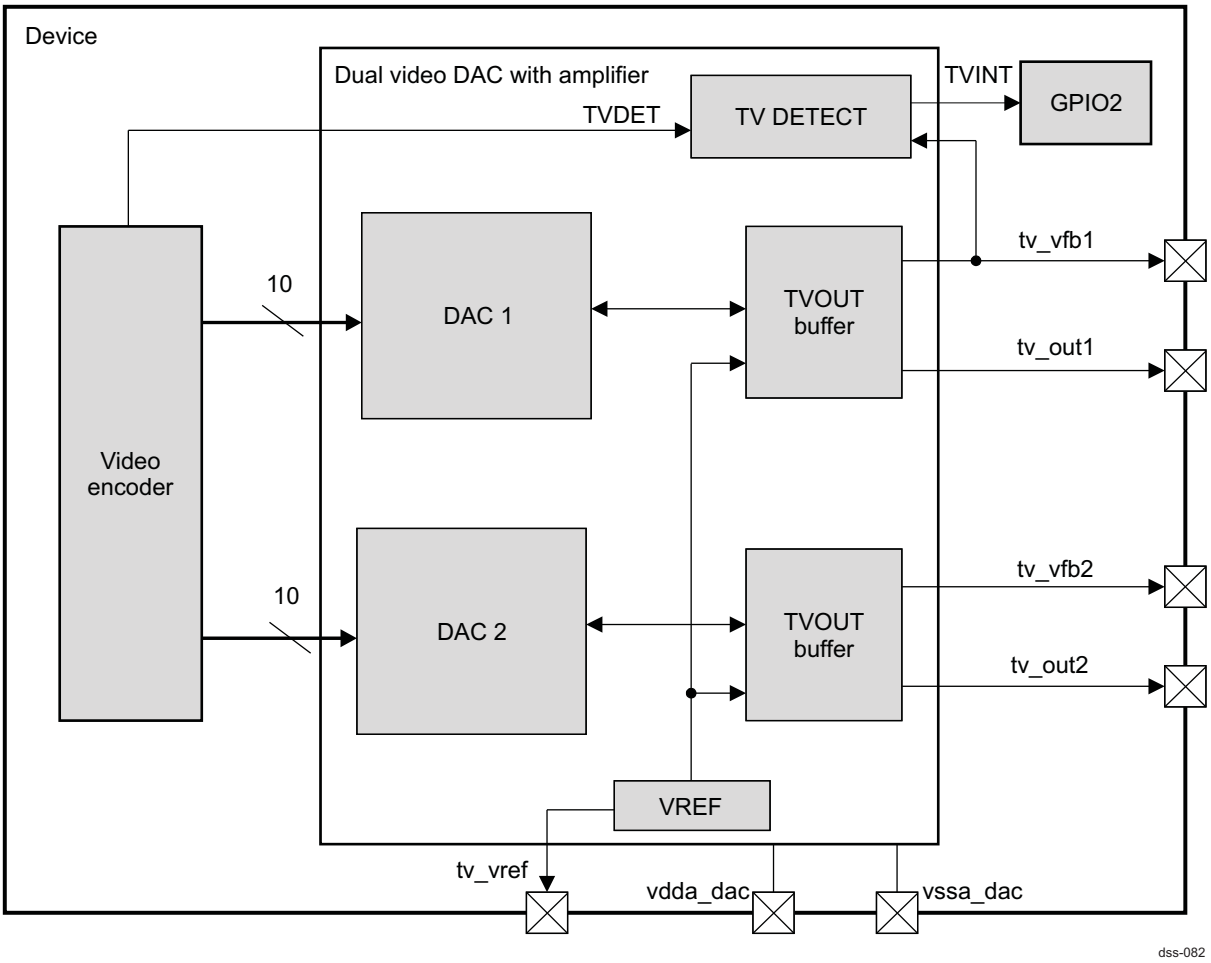
625-line line 23 WSS timing

dss-081

15.4.7.7 Video DAC Architecture

Figure 15-112 shows the architecture of the dual 10-bit video DAC.

Figure 15-112. Dual 10-Bit Video DAC Architecture



The display subsystem provides the necessary control signals to interface the memory frame buffer directly to external displays (TV sets). Two (one per channel) 10-bit current steering DAC is used to generate the video analog signal. The video mode DAC 1 (luma/composite) also includes the TV detection/disconnection and power-down mode features.

15.4.7.8 Video DC/AC Coupled TV Load

The dual 10-bit video DAC supports both dc-coupled and ac-coupled TV loads. The CONTROL.CONTROL_DEVCONF1[11] TVACEN bit is used to define which output coupling is used (0: Dc coupling; 1: Ac coupling). This bit is the first one to be programed according to the TV load on the PCB board.

Note: When dc coupling is used, note that there is a 385-mV dc offset at the TVOUT output (tv_out1 for DAC1 and tv_out2 for DAC2).

15.4.7.9 TV Detection/Disconnection Pulse Generation and Usage

15.4.7.9.1 TV Detection/Disconnection Pulse Generation

The TV detect block is an integral part of the dual video DAC module.

Notes:

- The TV detection/disconnection feature is supported only for VDAC1.
- The TV disconnection feature is recommended for power saving purpose. The TV detection/disconnection is only operational when video out is active. Therefore to detect cable connection automatically, it is necessary to periodically activate the video out to test for cable presence.

This block compares the output (tv_out1) to reference (tv_vref) voltages, to sense the condition of the load. To operate, the TV detect requires two digital signals, TVACEN and TVDET. The TVACEN signal indicates to both TVOUT buffer and the TV detect circuit if the load is ac or dc coupling to adjust accordingly. TVDET is a digital pulse at the frequency of the TV sync pulses. The operation of the circuit is based on the difference in voltage levels in the output of the buffer depending of the load status. The TV detect block compares the output against a couple of references and the result is latched at the start of every sync pulse. The status is read later in with the TVDET pulse rising edge.

The following registers are used to set the TV detection/disconnection pulse:

- The DSS.VENC_TVDETGP_INT_START_STOP_X register defines which pixels are used to start and stop the pulse inside their respective line.
- The DSS.VENC_TVDETGP_INT_START_STOP_Y register defines which lines are used to start and stop the pulse.
- The DSS.VENC_GEN_CTRL[0] EN bit enables or disables the TVDET pulse (0: Disable; 1: Enable).
- The DSS.VENC_GEN_CTRL[16] TVDP bit sets the TVDET pulse polarity (0: Active low; 1: Active high).

15.4.7.9.2 TV Detection Procedure

The TV detection procedure is the following:

1. Initial setup:
   - Program the DSS.VENC_TVDETGP_INT_START_STOP_X and DSS.VENC_TVDETGP_INT_START_STOP_Y registers to define on which pixels and lines, respectively, the TVDET pulse will start and stop.
   - Set the DSS.VENC_GEN_CTRL[16] TVDP bit to 1 (reset value) for a TVDET pulse active high polarity.
   - Set the DSS.VENC_GEN_CTRL[0] EN bit to 1 to enable the TVDET pulse generation.
2. The TVDET signal is set to low by hardware according to the settings in the DSS.VENC_TVDETGP_INT_START_STOP_X and DSS.VENC_TVDETGP_INT_START_STOP_Y registers.
3. Set the VENC_OUTPUT_CONTROL[0] LUMA_ENABLE bit (in s-video mode) or the VENC_OUTPUT_CONTROL[1] COMPOSITE_ENABLE (in composite video mode) to 1 to enable the video DAC1 output.
4. Power up the vdda_dac voltage for VDAC and the TVOUT buffer (repeat every TV field during the horizontal synchronization). This is software controlled through an I^2C interface connected to the power IC (TWL4030 device).
5. The TVDET pulse is set high by hardware according to the settings in the DSS.VENC_TVDETGP_INT_START_STOP_X and DSS.VENC_TVDETGP_INT_START_STOP_Y registers.
6. Check the TVINT output signal: When TVINT is set to 1, the load is connected.

CAUTION

- If ac coupling is selected, two TVDET pulses are required to set high the TVINT signal. Due to the internal logic of the video DAC1, the TVINT signal is generated after the next positive edge of the TVDET signal that happens during the next VSYNC timing.
- If dc coupling is selected, only one TVDET pulse is required to set high the TVINT signal.

15.4.7.9.3 TV Disconnection Procedure

The TV disconnection procedure is the following:

1. Initial setup:
   - Program the DSS.VENC_TVDETGP_INT_START_STOP_X and DSS.VENC_TVDETGP_INT_START_STOP_Y registers to define on which pixels and lines, respectively, the TVDET pulse will start and stop.
   - Set the DSS.VENC_GEN_CTRL[16] TVDP bit to 1 (reset value) for a TVDET pulse active high polarity.
   - Set the DSS.VENC_GEN_CTRL[0] EN bit to 1 to enable the TVDET pulse generation.
2. The TVDET signal is set to low by hardware according to the settings in the DSS.VENC_TVDETGP_INT_START_STOP_X and DSS.VENC_TVDETGP_INT_START_STOP_Y registers.
3. The TVDET pulse is set high by hardware according to the settings in the DSS.VENC_TVDETGP_INT_START_STOP_X and DSS.VENC_TVDETGP_INT_START_STOP_Y registers.
4. Check the TVINT output signal: When TVINT is reset to 0, the load is disconnected.
5. Reset the VENC_OUTPUT_CONTROL[0] LUMA_ENABLE bit (in s-video mode) or the VENC_OUTPUT_CONTROL[1] COMPOSITE_ENABLE (in composite video mode) to 0 to disable the video DAC1 output.
6. Power-down the vdda_dac voltage for VDAC and the TVOUT buffer (repeat every TV field during the horizontal synchronization). This is software controlled through an I^2C interface connected to the power IC (TWL4030 device).

CAUTION

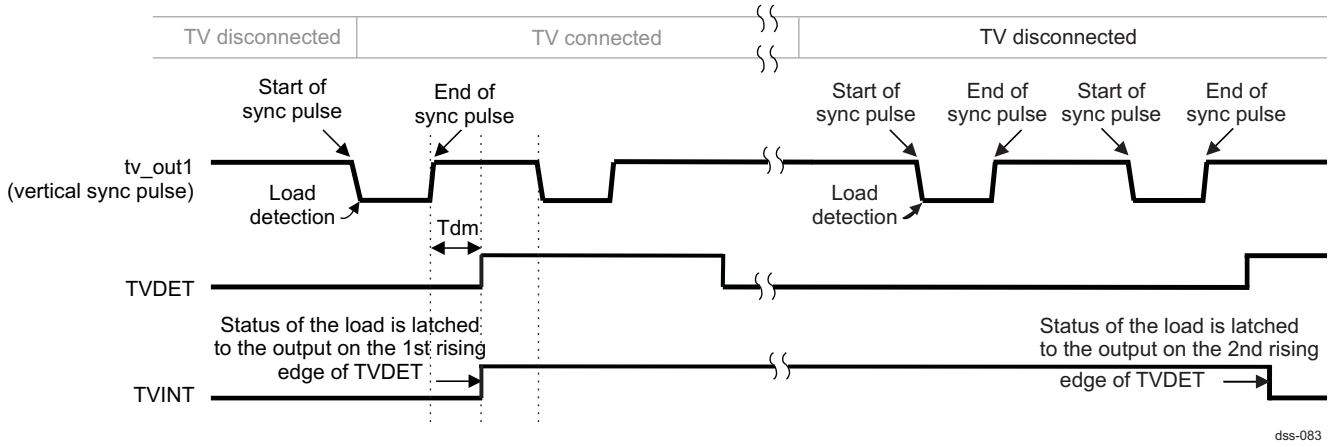
- If dc coupling is selected, two TVDET pulses are required to set low the TVINT signal. Due to the internal logic of the video DAC1, the TVINT signal is generated after the next positive edge of the TVDET signal that happens during the next VSYNC timing.
- If ac coupling is selected, only one TVDET pulse is required to set low the TVINT signal.

15.4.7.9.4 Recommended TV Detection/Disconnection Pulse Waveform

To enable the detection/disconnection of the load, the circuit requires that the TVDET pulse resembles the following waveform. As explained in Section 15.4.7.9.2, *TV Detection Procedure*, and in Section 15.4.7.9.3, *TV Disconnection Procedure* by using the video encoder registers, the TVDET pulse polarity, start and stop is programmable. The only critical parameter is Tdm, which should be longer than the delay through the DAC and TVOUT buffer, which is at least 750 ns.

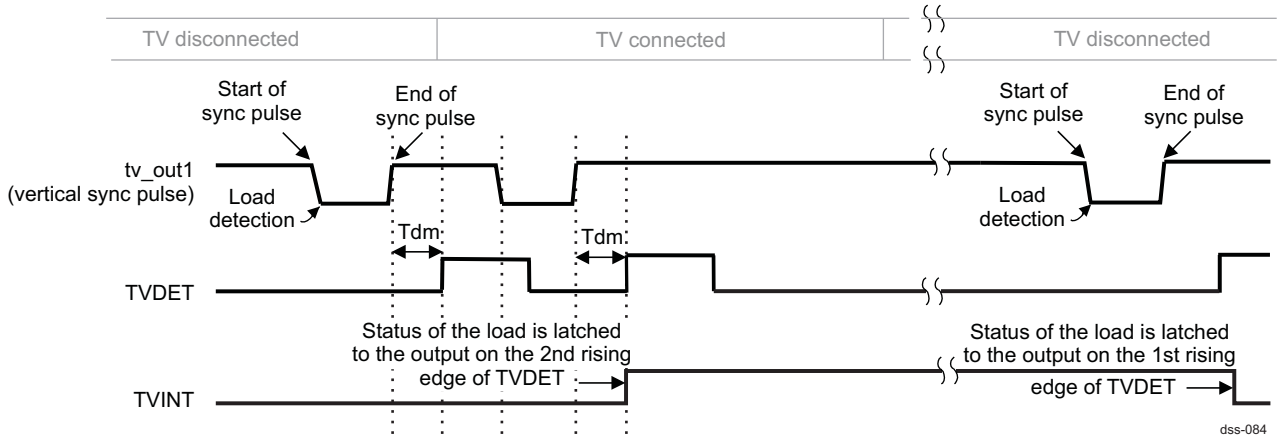
If dc-coupling is selected, the TVINT output signal for TV detection is latched at the rising edge of the first TVDET signal but the TVINT output signal for TV disconnection is latched after the next rising edge of the TVDET signal that happens during the next VSYNC timing. Figure 15-113 shows the waveforms for the dc-coupling TV detect pulse (TVDET) when load is connected and disconnected.

Figure 15-113. DC-Coupling TV Detect Waveforms for TV Connected and Disconnected



If ac-coupling is selected, the TVINT output signal for TV detection is latched after the next rising edge of the TVDET signal that happens during the next VSYNC timing but the TVINT output signal for TV disconnection is latched at the rising edge of the first TVDET signal. Figure 15-114 shows the waveforms for the ac-coupling TV detect pulse (TVDET) when load is connected and disconnected.

Figure 15-114. AC-Coupling TV Detect Waveforms for TV Connected and Disconnected



Notes:

- When setting the DSS.VENC_TVDETGP_INT_START_STOP_X register, software users must ensure that the TVDET signal is in the active area. To avoid any problem, the TVDET signal must not be longer than one line.
- The activation of the TVDET signal will not have a visual impact on the tv_out1 output signal.

15.4.7.9.5 TV Detection/Disconnection Usage

The TV-detection/TV-disconnection is based on the difference in voltage levels in the output of the TV buffer depending on the load status. The operation is slightly different for ac and for dc operation. For dc operation, the tv_out1 voltage is compared against a voltage reference (tv_vref) that makes the comparator trigger in each sync pulse while the load is connected. For ac operation, the tv_out1 voltage is compared against a voltage reference (tv_vref) that makes the comparator trigger in each sync pulse while the load is disconnected. In both cases, the TVINT output signal produces a logic 1 when a load is connected and a logic 0 when a load is disconnected.

For dc-coupling mode, see Figure 15-115 and for ac-coupling mode, see Figure 15-116

Note: Because the video DAC and the video encoder must be awake for connection detection, consider that the video DAC can take up to 10 µs to wake up.

Figure 15-115. GPIO Signal Waveform Proposal for TV Detection/Disconnection in DC-Coupling Mode

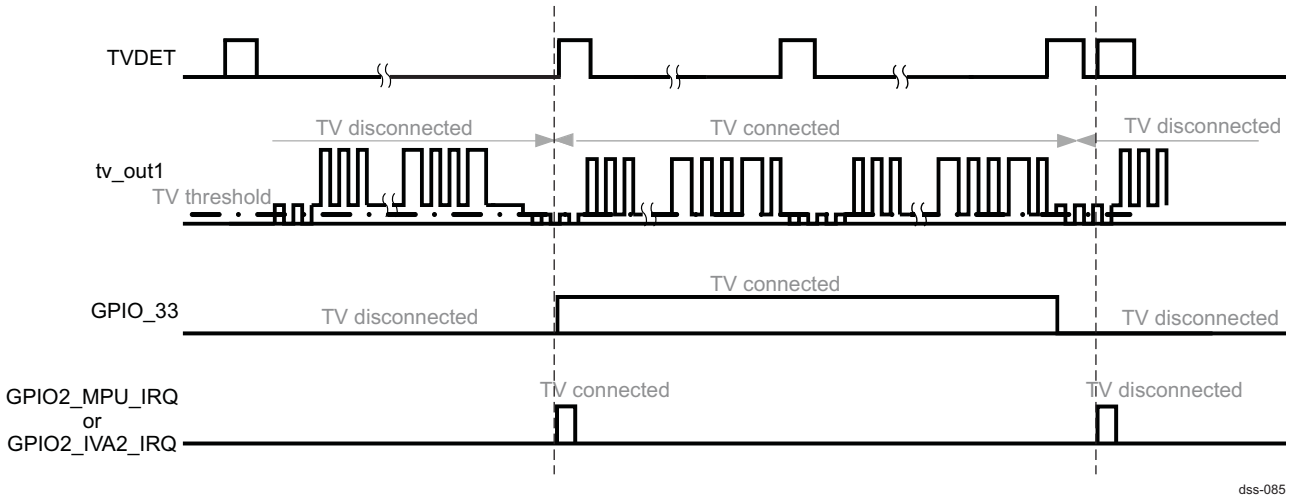
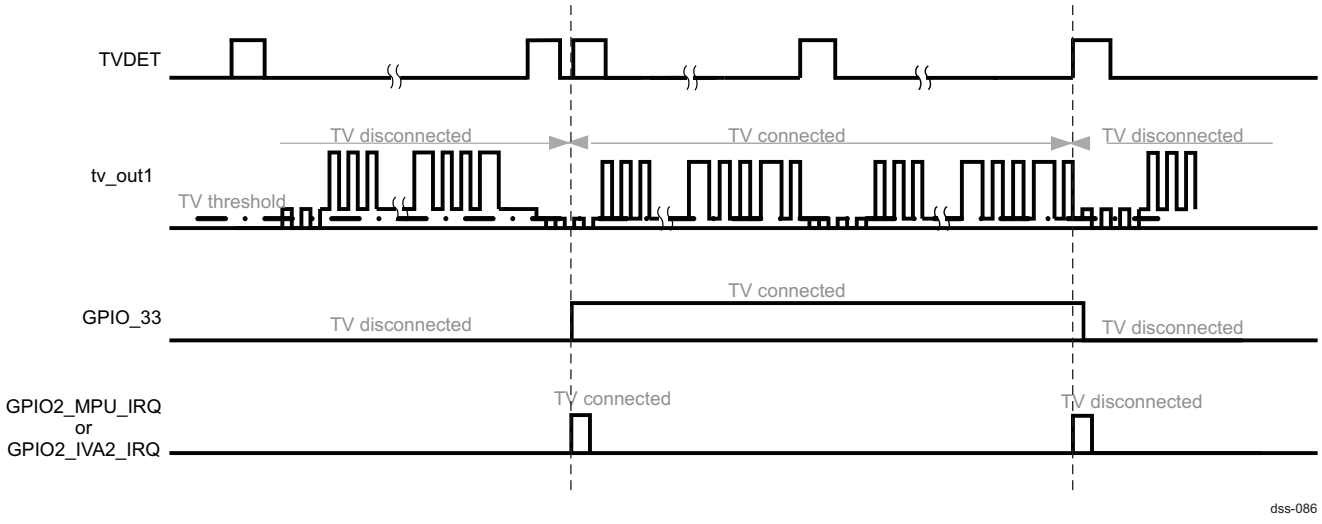


Figure 15-116. GPIO Signal Waveform Proposal for TV Detection/Disconnection in AC-Coupling Mode



15.4.7.10 Video DAC Bypass Mode

The dual 10-bit video DAC has a TVOUT buffer bypass mode that turns off the TVOUT buffer and redirects directly the DAC output to the VFB pins (tv_vfb1 for DAC1 and tv_vfb2 for DAC2).

This bypass mode is activated by setting the CONTROL.CONTROL_DEVCONF1[18] TVOUTBYPASS bit to 1. The reset value of the CONTROL.CONTROL_DEVCONF1[18] TVOUTBYPASS bit is 0 (that is, the TVOUT buffer is not bypassed).

Note: In bypass mode:

- the tv_out1 pin requires a RSET resistor connected to the ground. The typical value of the RSET resistor is 3.92 K.
- both tv_vfb1and tv_vfb2 pins require a RLOAD resistor connected to the ground. The typical value of the RLOAD resistor is 976.

CAUTION

- The TV detect feature is not available in bypass mode.
- In bypass mode, an external amplifier is needed on the tv_vbf1 and tv_vbf2 pins.

15.4.7.11 Video Dual-DAC Test Mode

The DAC can be tested for debug using either 10-bit external data or 10-bit internal register values directly connected to the DAC. See Figure 15-117 for video DAC test in composite video mode, and see Figure 15-118 for video DAC test in separate video mode.

Figure 15-117. DAC Test Mode in Composite Video Mode

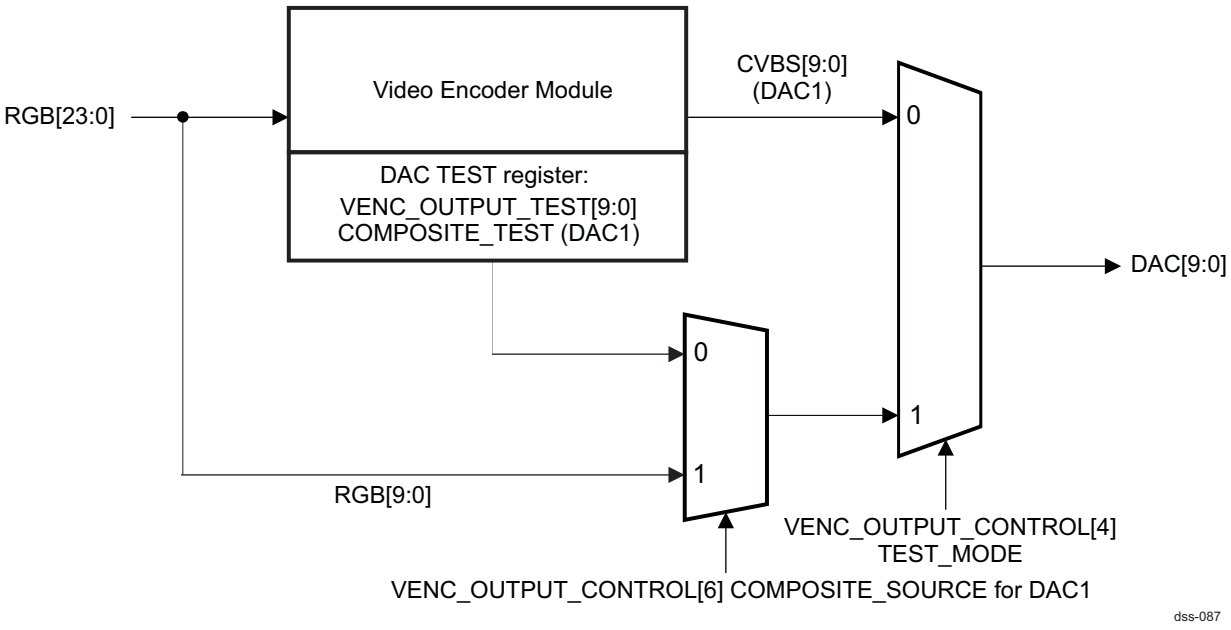
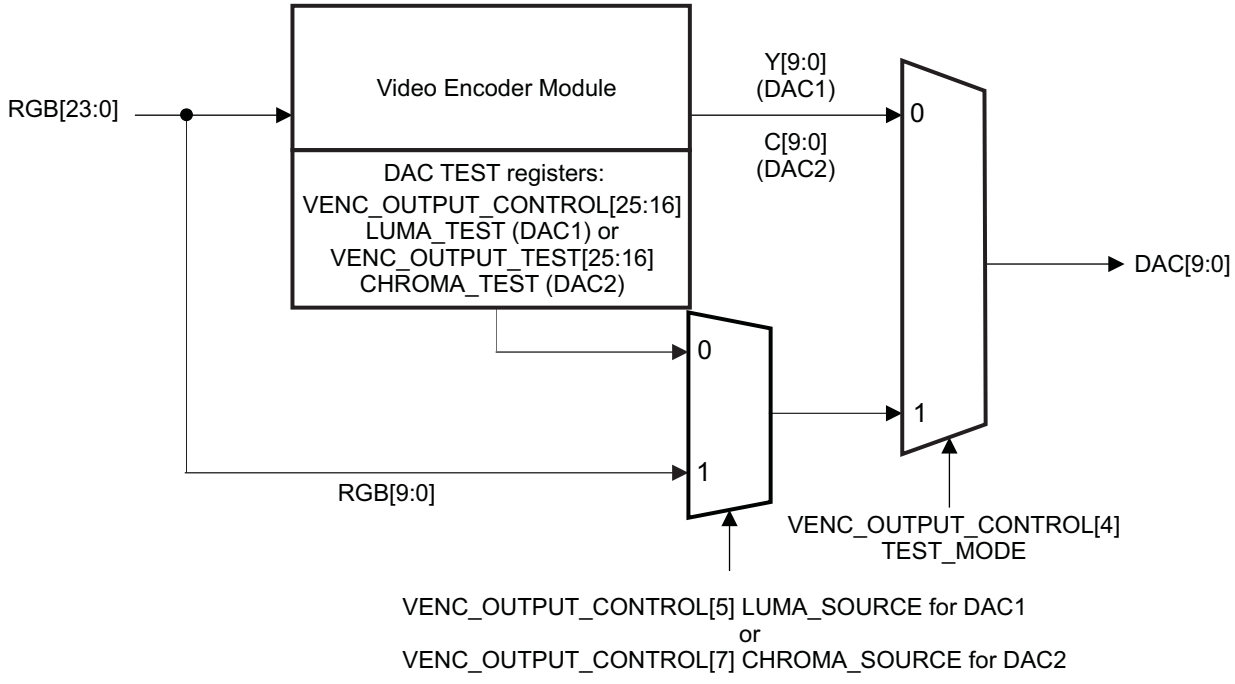


Figure 15-118. DAC Test Mode in Separate video Mode



- Use the DSS.VENC_OUTPUT_CONTROL[4] TEST_MODE bit to select between DAC normal mode (0x0) and DAC test mode (0x1).
- Use the DSS.VENC_OUTPUT_CONTROL[7] CHROMA_SOURCE bit for DAC2 and either the DSS.VENC_OUTPUT_CONTROL[6] COMPOSITE_SOURCE bit (in composite video mode) or the DSS.VENC_OUTPUT_CONTROL[5] LUMA_SOURCE bit (in s-video mode) for DAC1 to select the test mode:
  - 0x0: From the internal register DSS.VENC_OUTPUT_TEST[25:16] CHROMA_TEST bit field for DAC2 and either DSS.VENC_OUTPUT_TEST[9:0] COMPOSITE_TEST bit field (composite video) or DSS.VENC_OUTPUT_CONTROL[25:16] LUMA_TEST (s-video mode) for DAC1
  - 0x1: From the video port G[1:0], B[7:0]

Note: In the external data test mode (bypass mode), the display controller must provide the data (G[1:0], B[7:0]) externally. To do this, configure the video encoder to generate correct timing signals, without which the display controller cannot operate (even if the encoder core is bypassed from the data path perspective).

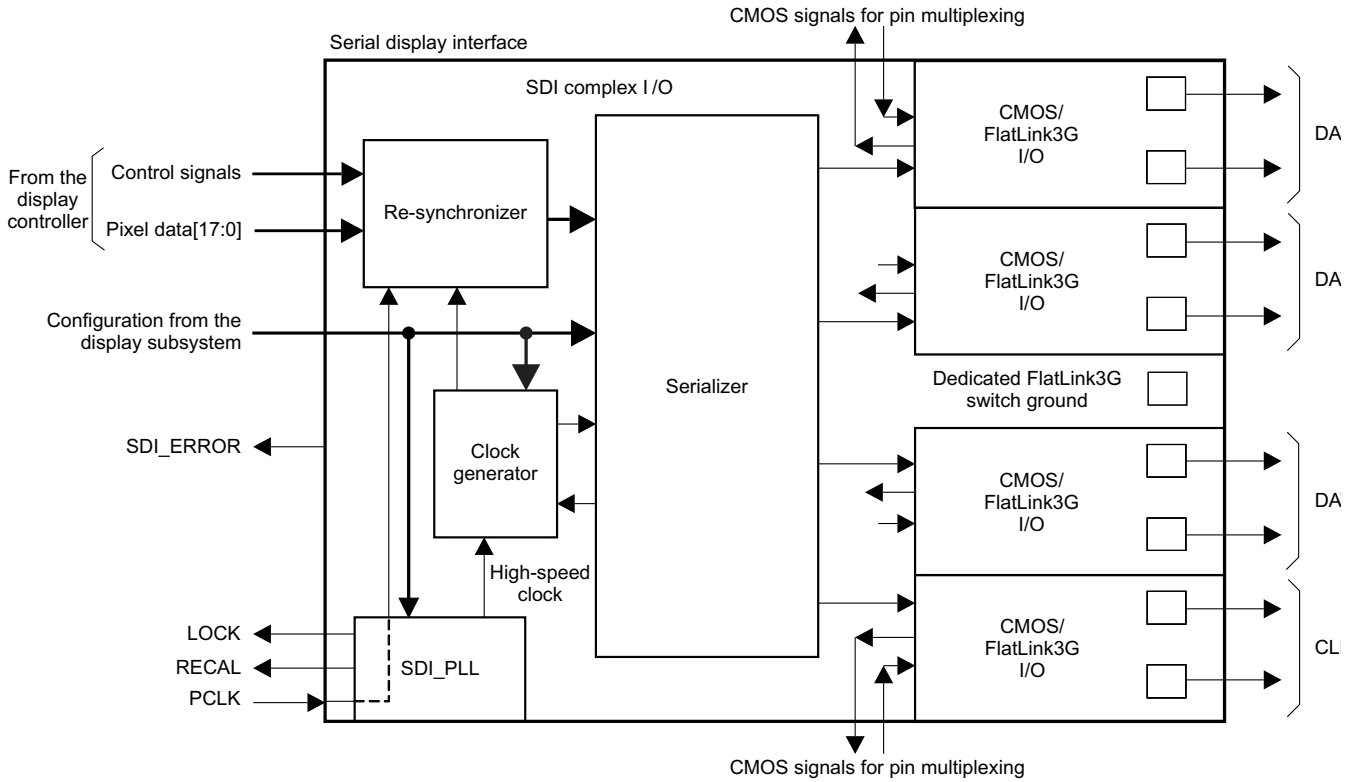
15.4.8 SDI Functionalities

The SDI interface has three main functions:

- A PLL (SDI_PLL) to multiply or divide the pixel clock frequency (PCLK) up to the required data rate. The pixel clock frequency range is 4-65 MHz. This clock may be divided by 2 above 32 MHz to meet the PLL input reference timing requirements.
- A specific IP block to output the data in the chosen serial format
- Output buffers to transmit the small swing differential signals

The last two functions are combined in the complex I/O module. Figure 15-119 shows the SDI architecture.

Figure 15-119. SDI Architecture Overview



Notes:

- The RECAL status indicates when the SDI_PLL must be recalibrated (0: No recalibration is required; 1: Recalibration is required).

  This status is connected to the GPIO_80 internal signal of the GPIO3 module. For additional information, see Chapter 25, *General-Purpose Interface*.
- The LOCK status indicates when the SDI_PLL is locked (0: PLL not locked; 1: PLL locked). This status is connected to the GPIO_81 internal signal of the GPIO3 module. For additional information, see Chapter 25, *General-Purpose Interface*.

For more information about lock and recalibration, see Chapter 4, *Power, Reset, and Clock Management*.

- The SDI_ERROR status indicated when the SDI module is in error mode (0: No error; 1: SDI in error). This status allows monitoring error conditions, such as inconsistent configuration or internal loss of synchronization. It is connected to the GPIO_82 internal signal of the GPIO3 module. For additional information, see Chapter 25, *General-Purpose Interface*.

15.5 Display Subsystem Basic Programming Model

This section describes how to configure the display subsystem for the desired functionalities and also describes the programming models of the display controller, the RFBI, the SDI and the video encoder.

The main configuration scenarios are:

- LCD panel support (bypass or RFBI mode)
  Configure the RFBI module (only if in RFBI mode; otherwise, the default values must remain), and then configure the display controller to the desired functionalities before the activities start.
- TV set support
  Configure the video encoder and then the display controller.
- Both LCD panel support (bypass or RFBI mode) and TV set support
  Configure the RFBI module (only if in RFBI mode; otherwise, leave the default values), configure the video encoder, and then configure the display controller.
- TI FlatLink 3G-compliant LCD panel support
  Configure the SDI, configure the video encoder, and then configure the display controller.

15.5.1 Display Subsystem Reset

The display subsystem can receive a software reset that is propagated through all of the submodules to initialize the subsystem. The following procedure describes a possible sequence:

1. If the LCD is on, stop the LCD by setting the DSS.DISPC_CONTROL[0] LCDENABLE bit to 0.
   a. Reset the frame done status bit by writing 1 in the DSS.DISPC_IRQSTATUS[0] FRAMEDONE bit.
   b. Wait until the DSS.DISPC_IRQSTATUS[0] FRAMEDONE bit is set to 1. This shows that the end of frame has taken place and the LCD stop is complete.
2. To take the display subsystem out of reset, all clocks related to the display subsystem must be enabled and the DPLL4 must be enabled. The following clocks must be enabled to take the display subsystem out of reset:
   - PRCM.CM_FCLKEN_DSS[0] EN_DSS1 bit set to 1
   - PRCM.CM_FCLKEN_DSS[1] EN_DSS2 bit set to 1
   - PRCM.CM_FCLKEN_DSS[2] EN_TV bit set to 1
   - PRCM.CM_ICLKEN_DSS[0] EN_DSS bit set to 1

   Once the clocks are enabled as shown, the display subsystem can be taken out of reset.
3. Write 1 in the DSS.DSS_SYSCONFIG[1] SOFTRESET bit to apply the soft reset to the subsystem.
4. Read the DSS.DSS_SYSSTATUS[0] RESETDONE bit. If this bit is 1, the reset sequence is complete; otherwise, read this bit again (the reset sequence is not completed).

15.5.2 Display Subsystem Configuration Phase

The display subsystem configuration phase is important to configure the data flow for using the LCD panel or the TV set. Use the following flow:

1. To configure the top level of the functional clock of the display controller clock, set the DSS.DSS_CONTROL[0] DSS_CLK_SWITCH bit.
2. To configure the top level of the video encoder, set the DSS.DSS_CONTROL[2] VENC_CLOCK_MODE bit and the DSS.DSS_CONTROL[3] VENC_CLOCK_X4 bit for TV set support.
3. To configure the top level of the DAC, set the DSS.DSS_CONTROL[4] DAC_DEMEN bit for TV set support (if required).
4. Configure the TI FlatLink 3G-compliant LCD panel through the SPI interface to use the serial interface display.
5. Configure the RFBI module and/or the SDI module and/or the video encoder as needed.
6. Configure the display controller.

15.5.3 Display Controller Basic Programming Model

Some display controller registers are termed *shadow registers*, which are associated with the digital output and/or the LCD output. A shadow register change has no direct effect on the configuration of the display controller unless the DSS.DISPC_CONTROL[5] GOLCD bit is set for the LCD output and/or the DSS.DISPC_CONTROL[6] GODIGITAL bit is set for the digital output.

In the case of the digital output, after programming the shadow registers, the DSS.DISPC_CONTROL[6] GODIGITAL bit must be set to 1. If this bit is not set, the configuration of the display controller will have no effect. This setting indicates that all display controller shadow registers are programmed and that hardware can update the internal registers at the external EVSYNC.

In the case of the LCD output, after programming the shadow registers, the DSS.DISPC_CONTROL[5] GOLCD bit must be set to 1. If this bit is not set, the configuration of the display controller will have no effect. This setting indicates that all display controller shadow registers are programmed and that hardware can update the internal registers at the VFP start period.

Before setting either the DSS.DISPC_CONTROL[5] GOLCD or DSS.DISPC_CONTROL[6] GODIGITAL bit, ensure that the bit is cleared.

Table 15-45 lists the shadow registers.

Table 15-45. Shadow Registers

Shadow Register Name	Updated on VFP Start Period (LCD output)	Updated on External VSYNC (Digital output)
DSS.DISPC_CONTROL	X(1)	X(1)
DSS.DISPC_CONFIG	X	X
DSS.DISPC_DEFAULT_COLOR_m (m = 0)	X	
DSS.DISPC_DEFAULT_COLOR_m (m = 1)		X
DSS.DISPC_TRANS_COLOR_m (m = 0)	X	
DSS.DISPC_TRANS_COLOR_m (m = 1)		X
DSS.DISPC_LINE_NUMBER	X	
DSS.DISPC_TIMING_H	X	
DSS.DISPC_TIMING_V	X	
DSS.DISPC_POL_FREQ	X	
DSS.DISPC_DIVISOR	X	
DSS.DISPC_SIZE_DIG		X
DSS.DISPC_SIZE_LCD	X	
DSS.DISPC_GFX_BAj (j = 0,1)	X	X
DSS.DISPC_GFX_POSITION	X	X
DSS.DISPC_GFX_SIZE	X	X
DSS.DISPC_GFX_ATTRIBUTES	X	X
DSS.DISPC_GFX_FIFO_THRESHOLD	X	X
DSS.DISPC_GFX_ROW_INC	X	X
DSS.DISPC_GFX_PIXEL_INC	X	X
DSS.DISPC_GFX_WINDOW_SKIP	X	X
DSS.DISPC_GFX_TABLE_BA	X	X
DSS.DISPC_GFX_PRELOAD	X	X
DSS.DISPC_CPR_COEF_R	X	
DSS.DISPC_CPR_COEF_G	X	
DSS.DISPC_CPR_COEF_B	X	
DSS.DISPC_VIDn_BAj (j= 0,1)	X	X
DSS.DISPC_VIDn_POSITION	X	X
DSS.DISPC_VIDn_SIZE	X	X

(1) Some of the register bit fields are shadow bits. For more information, see Section 15.7, *Display Subsystem Register Manual*.

Table 15-45. Shadow Registers (continued)

Shadow Register Name	Updated on VFP Start Period (LCD output)	Updated on External VSYNC (Digital output)
DSS.DISPC_VIDn_ATTRIBUTES	X	X
DSS.DISPC_VIDn_FIFO_THRESHOLD	X	X
DSS.DISPC_VIDn_ROW_INC	X	X
DSS.DISPC_VIDn_PIXEL_INC	X	X
DSS.DISPC_VIDn_FIR	X	X
DSS.DISPC_VIDn_PICTURE_SIZE	X	X
DSS.DISPC_VIDn_ACCUl (l = 0,1)	X	X
DSS.DISPC_VIDn_FIR_COEF_Hi (i = 0 to 7)	X	X
DSS.DISPC_VIDn_FIR_COEF_HVi (i = 0 to 7)	X	X
DSS.DISPC_VIDn_FIR_COEF_Vi (i = 0 to 7)	X	X
DSS.DISPC_VIDn_CONV_COEFi (i = 0 to 4)	X	X
DSS.DISPC_VIDn_PRELOAD	X	X
DSS.DISPC_DATA_CYCLEk (k = 0 to 3)	X	

15.5.3.1 Display Controller Configuration

The following registers define the display controller configuration:

- DSS.DISPC_SYSCONFIG
- DSS.DISPC_SYSSTATUS
- DSS.DISPC_IRQSTATUS
- DSS.DISPC_IRQENABLE

15.5.3.2 Graphics Layer Configuration

The graphics layer configuration is common to the LCD and the TV set.

15.5.3.2.1 Graphics DMA Registers

The following registers define the graphics DMA engine configuration:

- DSS.DISPC_CONTROL
- DSS.DISPC_GFX_BAj
- DSS.DISPC_GFX_ATTRIBUTES
- DSS.DISPC_GFX_ROW_INC
- DSS.DISPC_GFX_PIXEL_INC
- DSS.DISPC_GFX_FIFO_THRESHOLD
- DSS.DISPC_GFX_TABLE_BA

The following fields define the attributes of the graphics DMA engine:

- Graphics layer enable (DSS.DISPC_GFX_ATTRIBUTES[0] GFXENABLE bit): The default value of this bit at reset time is 0x0 (Disabled). The graphics DMA engine fetches encoded pixels from the system memory only when the graphics layer is enabled (a valid configuration is programmed for the graphics layer). The graphics window is present and the graphics pipeline is active.
- Burst size (DSS.DISPC_GFX_ATTRIBUTES[7:6] GFXBURSTSIZE field): The default burst size at reset time is 4 x 32 bytes. The possible values are 4 x 32, 8 x 32, and 16 x 32 bytes. The burst size is initialized at boot time by the software and never changes as long as the display controller is enabled. This field indicates the maximum burst size for the specific pipeline. In case of misalignment, the DMA engine may issue single and/or smaller burst requests because the burst size must be aligned to the burst boundary.
- Preload configuration (DSS.DISPC_GFX_ATTRIBUTES[11] GFXFIFOPRELOAD bit): The default

preload configuration uses the DSS.DISPC_GFX_PRELOAD register value (the reset value is 256 bytes) to define the number of bytes to be fetched from system memory into the display controller graphics FIFO. By programming the DSS.DISPC_GFX_ATTRIBUTES[11] GFXFIFOPRELOAD bit, software users select between preload register (with 256 bytes as the reset value) and the high threshold value for preload of the encoded pixels. For best performance, the configuration of thresholds is defined using the FIFO size (in bytes) minus 1 for the high threshold, and the FIFO size (in bytes) minus the burst size (in bytes) for the low threshold, which provides 960, 992, and 1008, respectively, for burst sizes 16x32, 8x32, and 4x32. Note also that the preload value is defined based on the following display types:

  – Active matrix (TFT) display: DSS.DISPC_GFX_PRELOAD[11:0] PRELOAD = 0x60 (value is 96)
  – Color passive matrix (STN) display: DSS.DISPC_GFX_PRELOAD[11:0] PRELOAD = 0x72 (value is 114)
  – Monochrome passive matrix (STN) display: DSS.DISPC_GFX_PRELOAD[11:0] PRELOAD = 0xE0 (value is 224)

- Base address of the graphics buffer in system memory (DSS.DISPC_GFX_BAj registers): The default value of these two registers at reset time is 0x0. The horizontal resolution is one pixel because the base address is aligned on a pixel size boundary. In case of 4 BPP, the resolution is two pixels; for 2 BPP, resolution is four pixels; for 1 BPP, resolution is eight pixels; and for RGB24 packed format, the resolution is four pixels. The vertical resolution is one line. The register DSS.DISPC_GFX_BA0 defines the base address of the even field; and DSS.DISPC_GFX_BA1 defines the base of the odd field in the case of an external synchronization and based on the value of the input signal DISPC_FID and the polarity. To improve system throughput, the base address should be aligned on the burst size boundary.
- Graphics FIFO threshold (DSS.DISPC_GFX_FIFO_THRESHOLD register): The low threshold (DSS.DISPC_GFX_FIFO_THRESHOLD[11:0] GFXFIFOLOWTHRESHOLD) and the high threshold (DSS.DISPC_GFX_FIFO_THRESHOLD[27:16] GFXFIFOHIGHTHRESHOLD) values define the FIFO DMA behavior. When the low level is reached, one or more requests are issued to the L3-based interconnect to fill up the FIFO to reach the high threshold. A request is issued as long as the FIFO has enough space available to accept a burst. The DMA engine then waits until the low level is reached to restart the requests. By setting the DSS.DISPC_CONFIG[14] FIFOMERGE bit to 1, users merge the three FIFOs (GFX, VID1, and VID2). In this case, the low threshold (the DSS.DISPC_GFX_FIFO_THRESHOLD[11:0] GFXFIFOLOWTHRESHOLD bit field) and the high threshold (DSS.DISPC_GFX_FIFO_THRESHOLD[27:16] GFXFIFOHIGHTHRESHOLD bit field) values must be programmed with a multiplier factor of three (3 x value). By default, the FIFOs are not merged (the DSS.DISPC_CONFIG[14] FIFOMERGE bit reset value is 0).
- Palette/gamma table used (DSS.DISPC_CONFIG[3] PALETTEGAMMATABLE bit): The bit indicates if the palette must be loaded before the graphics data for the following frame. The bit is set by software and reset by hardware.
- Base address of the palette/gamma table buffer in system memory (DSS.DISPC_GFX_TABLE_BA register): The default value of this register at reset time is 0x0. The base address is aligned on a 32-bit address. Depending on the pixel size of graphics data (1, 2, 4, or 8 BPP), 16 (1, 2, or 4 BPP), or 256 (8 BPP) x 32-bit values are loaded from system memory into the internal table memory. To load the table when using the memory as a gamma table, the graphics pipeline is enabled and then disabled by the software when the palette loaded interrupt is generated. The overlay manager ignores the graphics pipe when the table is used as a gamma table.

Note: In case of RGB16 format and optimization enabled, the base address is aligned on a 32-bit boundary and the number of bytes to skip is a multiple of 4 bytes.

- Graphics Priority (DSS.DISPC_GFX_ATTRIBUTES[14] GFXARBITRATION): The default value at reset time is 0x0. It is used to change between normal priority (value of 0) to high priority (value of 1) to change priority for the graphics channel vs. video channels. It can be used to give higher priority to the pipelines with real time constraint vs. non real time pipelines. For that is, pipelines associated to the LCD output in RFBI mode should have lower priority than pipelines associated to TV output.
- Graphics Self-Refresh (DSS.DISPC_GFX_ATTRIBUTES[15] GFXSELFREFRESH): The default value at reset time is 0x0. It is used to use the DMA FIFO without accessing the interconnect for multiple frames. Once, the data have been loaded to the DMA FIFO for displaying the frame, they are used for the following frames.

The sequence to activate the self-refresh is the following:

– Frame t: The bit field should be set at anytime during frame
– Frame t+1: Fetch of the data in the DMA FIFO and display of the frame
– Frame t+2: No access to the L3 interconnect, DMA FIFO uses to provide the pixels

The sequence to deactivate the self-refresh is the following:

– Frame t: No access to the L3 interconnect, DMA FIFO uses to provide the pixels, bit field can be changed at any time during the frame
– Frame t+1: Fetch of the data from system memory using the L3 interconnect

15.5.3.2.2 Graphics Layer Configuration Registers

The following registers define the graphics layer configuration:

- DSS.DISPC_CONFIG
- DSS.DISPC_GFX_POSITION
- DSS.DISPC_GFX_SIZE
- DSS.DISPC_GFX_ATTRIBUTES

The graphics layer is enabled/disabled by setting/resetting the DSS.DISPC_GFX_ATTRIBUTES[0] GFXENABLE bit. When the graphics layer is disabled, the graphics window does not exist on the screen and the graphics pipeline and DMA are inactive.

Set a valid configuration before enabling the graphics layer. After a register change, either the DSS.DISPC_CONTROL[6] GODIGITAL or DSS.DISPC_CONTROL[5] GOLCD bit must be set. The software must wait for the hardware to reset the bit before setting it. The software reset is not recommended because the application cannot ensure that the bit is reset before the hardware reset.

15.5.3.2.3 Graphics Window Attributes

The following fields define the attributes of the graphics window:

- Graphics format (DSS.DISPC_GFX_ATTRIBUTES[4:1] GFXFORMAT bit field): The default value of this bit field at reset time is 0x0 (BITMAP 1-BPP). The graphics format can be either: BITMAP1, BITMAP2, BITMAP4, or BITMAP8 (CLUT) or RGB12, RGB16, or RGB24 (true-color formats).
- Graphics window X-position (DSS.DISPC_GFX_POSITION[10:0] GFXPOSX bit field): The default value at reset time is 0x0. The window X-position is from 0 to 2047 columns. All integer values in the range [0:2047] are allowed.
- Graphics window Y-position (DSS.DISPC_GFX_POSITION[26:16] GFXPOSY bit field): The default value of this bit field at reset time is 0x0.The window Y-position is from 0 to 2047 rows. All integer values in the range [0:2047] are allowed.
- Graphics window width (DSS.DISPC_GFX_SIZE[10:0] GFXSIZEX bit field): The default value at reset time is 0x0 (1 pixel). The window width is from 1 to 2048 pixels. All integer values in the range [1:2048] are allowed for the following formats: 8 BPP, RGB12, RGB16, and RGB24. The width must be a multiple of eight pixels for 1 BPP, four pixels for 2 BPP, and two pixels for 4 BPP. The maximum bandwidth efficiency for accessing the pixels in system memory is reached when the width of the graphics window (in bytes) is a multiple of the graphics burst size defined in the DSS.DISPC_GFX_ATTRIBUTES[7:6] GFXBURSTSIZE bit field (in bytes).

Note: When the RGB24 packed format is selected, the width must be a multiple of 12 bytes when the DSS.DISPC_GFX_ROW_INC register is not 1. When DSS.DISPC_GFX_ROW_INC register is 1, the width can be any size from 1 to 2048 pixels.

The entire pixels of the graphics window must be inside the LCD screen. Depending on the width of the buffer to be displayed in the graphics layer and the position, the width should be adjusted by software to limit the right edge of the window inside the screen.

- Graphics window height (DSS.DISPC_GFX_SIZE[26:16] GFXSIZEY bit field): The default value at reset time is 0x0 (1 pixel). The window height is from 1 to 2048 pixels. All integer values in the range [1:2048] are allowed. The entire pixels of the graphics window must be inside the LCD screen. Depending on the height of the buffer to be displayed in the graphics layer and the position, the height

should be adjusted by software to limit the bottom edge of the window inside the screen

- Graphics data endianness (DSS.DISPC_GFX_ATTRIBUTES[10] GFXENDIANNESS bit): This bit indicates the endianness (little or big) of the graphics pixels. The default value at reset time is 0x0 (little endian).
- Graphics data nible mode (DSS.DISPC_GFX_ATTRIBUTES[9] GFXNIBBLEMODE bit): This bit indicates the nibble mode of the graphics pixels. The default value at reset time is 0x0 (Disable).
- Graphics replication logic enable (DSS.DISPC_GFX_ATTRIBUTES[5] GFXREPLICATIONENABLE bit): The default value at reset time is 0x0 (Disable). The encoded pixel data in RGB format (RGB16) can be extended to 24-bit format with or without replication of the MSB part to fill up the LSB due to the 24-bit left alignment. If the replication logic is turned off, the LSB part is filled up with 0s.
- Graphics window skip enable (DSS.DISPC_CONTROL[12] OVERLAYOPTIMIZATION bit): By setting/resetting the bit, the overlay optimization is enabled or disabled. Before enabling the overlay optimization, the DSS.DISPC_GFX_WINDOW_SKIP[31:0] GFXWINDOWSKIP bit field must be set according to the video1 and graphics windows overlap. The default value at reset time is 0x0 (Disable). When video1 is not present, the DSS.DISPC_GFX_WINDOW_SKIP[31:0] GFXWINDOWSKIP bit field should be reset. When the color key is used, the DSS.DISPC_GFX_WINDOW_SKIP[31:0] GFXWINDOWSKIP bit field should be reset.
- Graphics window skip (DSS.DISPC_GFX_WINDOW_SKIP[31:0] GFXWINDOWSKIP bit field): The bit field represents the number of bytes to skip while fetching the graphics-encoded pixels when reaching the beginning of the video window. The optimization allows fetching only the visible graphics pixels. The color key cannot be selected because the graphics pixels under the video window are not present. The default value at reset time is 0x0 (0 byte).

  Figure 15-120 through Figure 15-123 give examples of how to program the GFXWINDOWSKIP field for overlay optimization:

Figure 15-120. Overlay Optimization: Case 1

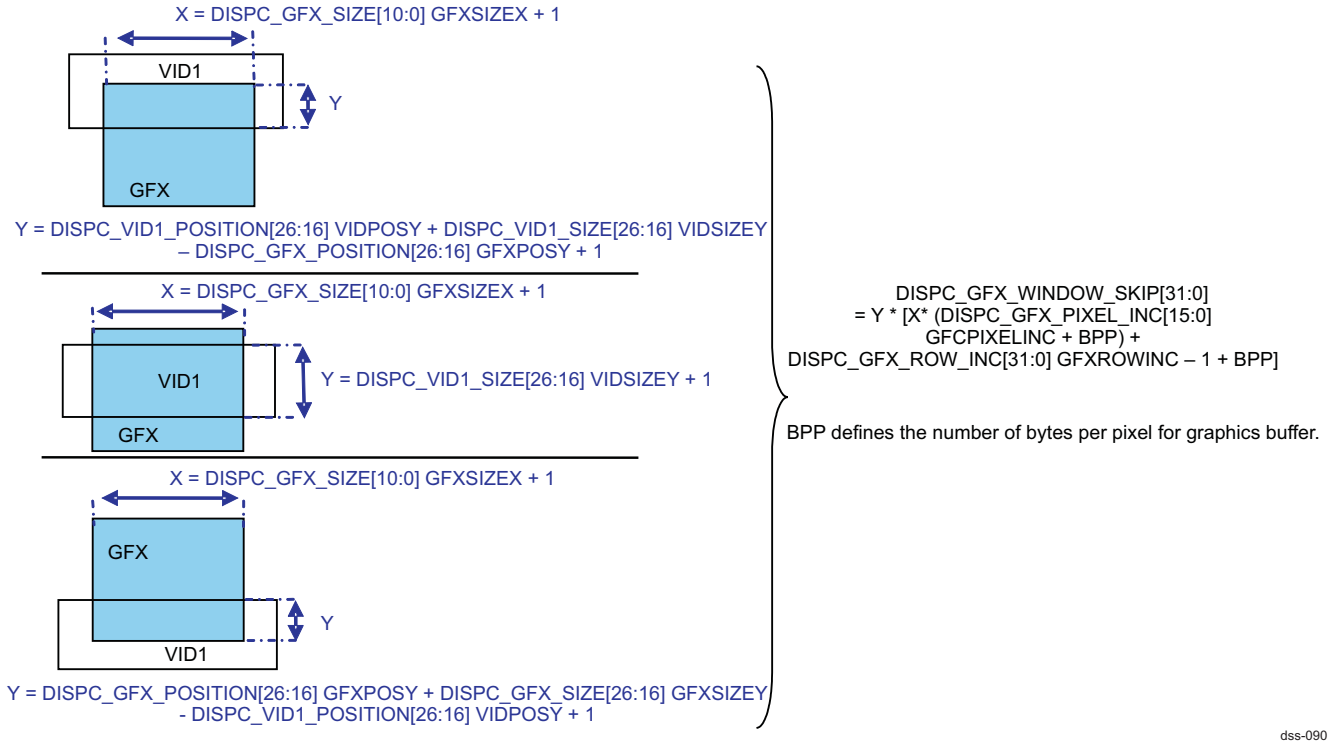
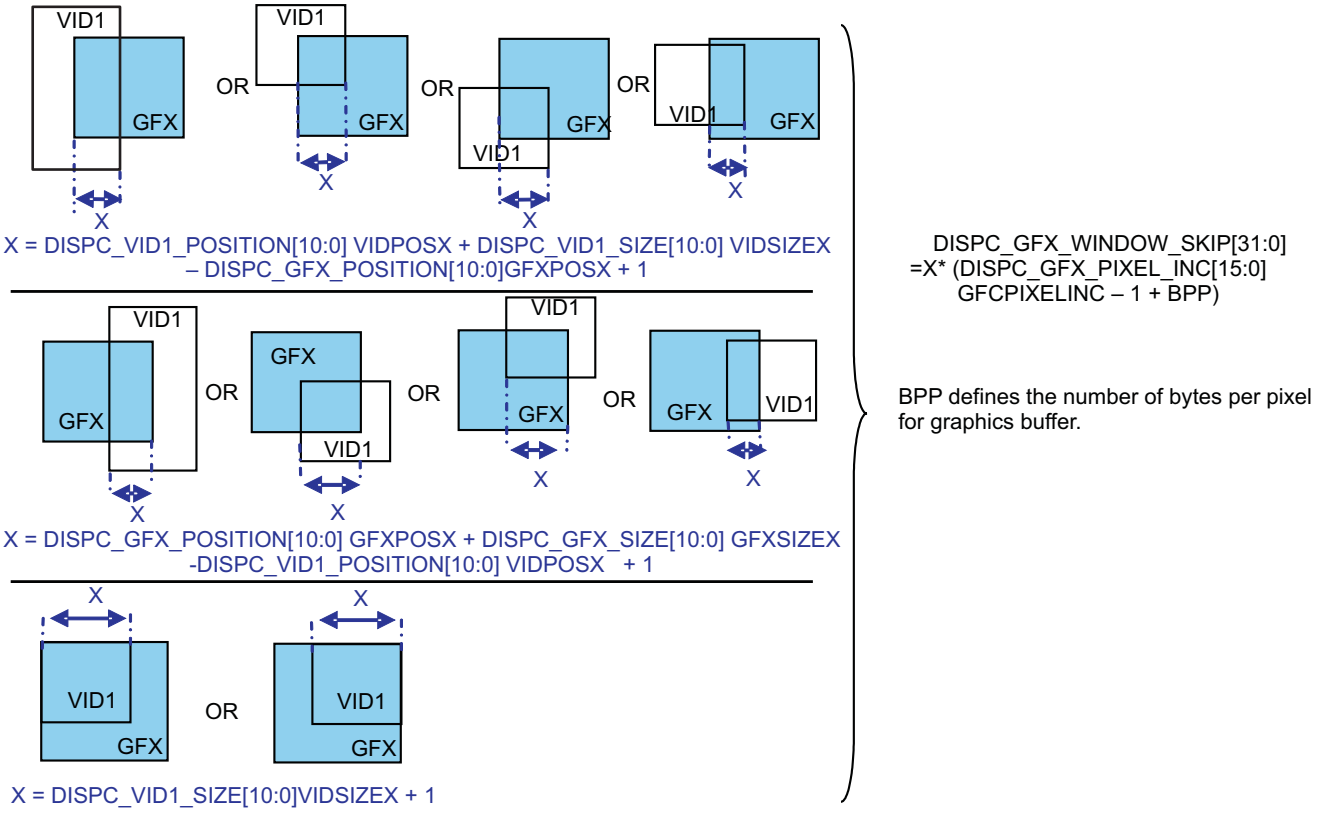
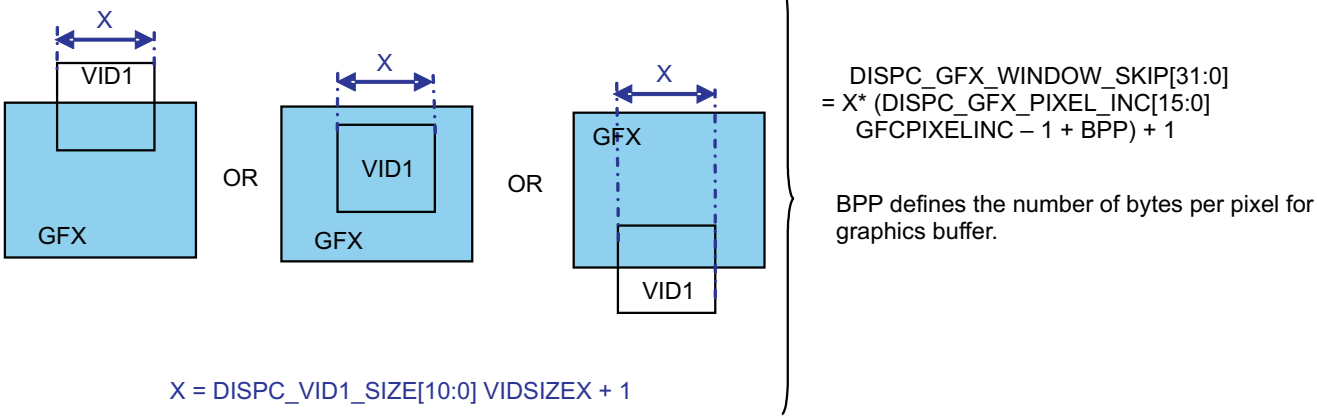


Figure 15-121. Overlay Optimization: Case 2



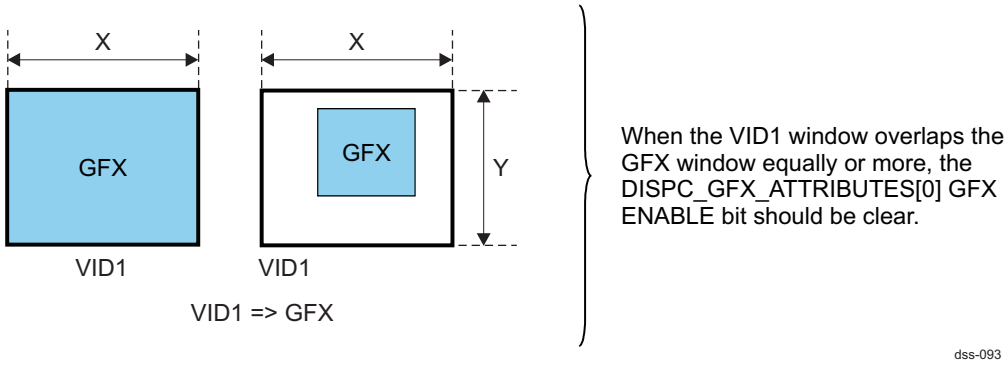
dss-091

Figure 15-122. Overlay Optimization: Case 3



dss-092

Figure 15-123. Overlay Optimization: Case 4



15.5.3.3 Video Layer Configuration

The video layer configuration is common to the LCD and the TV set.

15.5.3.3.1 Video DMA Registers

The following registers define the video DMA engine configuration:

- DSS.DISPC_CONTROL
- DSS.DISPC_VIDn_BA0
- DSS.DISPC_VIDn_ATTRIBUTES
- DSS.DISPC_VIDn_ROW_INC
- DSS.DISPC_VIDn_PIXEL_INC
- DSS.DISPC_VIDn_FIFO_THRESHOLD
- DSS.DISPC_VIDn_PICTURE_SIZE

The following fields define the attributes of the graphics DMA engine:

- Video layer enable (DSS.DISPC_VIDn_ATTRIBUTES[0] VIDENABLE bit): The default value of this bit at reset time is 0x0 (Disabled). The video DMA engine fetches encoded pixels from the system memory only when the video layer is enabled (a valid configuration is programmed for the video layer). The video window is present and the video pipeline is active.
- Burst size (DSS.DISPC_VIDn_ATTRIBUTES[15:14] VIDBURSTSIZE bit field): The default burst size at reset time is 4 x 32 bytes. The possible values are 4 x 32, 8 x 32, and 16 x 32 bytes. The burst size is initialized at boot time by the software and never changes as long as the display controller is enabled. This bit field indicates the maximum burst size for the specific pipeline. In case of misalignment, the DMA engine may issue single and/or smaller burst requests, because the burst size must be aligned to the burst boundary.
- Preload configuration (DSS.DISPC_VIDn_ATTRIBUTES[19] VIDFIFOPRELOAD bit): The default preload configuration uses the DSS.DISPC_VIDn_PRELOAD register value (the reset value is 256 bytes) to define the number of bytes to be fetched from system memory into the display controller graphics FIFO. By programming the DSS.DISPC_VIDn_ATTRIBUTES[19] VIDFIFOPRELOAD bit, software users select between preload register (with 256 bytes as the reset value) and the high threshold value for preload of the encoded pixels. For best performance, the configuration of thresholds is defined using the FIFO size (in bytes) minus 1 for the high threshold, and the FIFO size (in bytes) minus the burst size (in bytes) for the low threshold, which provides 960, 992, and 1008, respectively, for burst sizes 16x32, 8x32, and 4x32. Note also that the preload value is defined based on the following display types:
  - Active matrix (TFT) display: DSS.DISPC_VIDn_PRELOAD[11:0] PRELOAD = 0xB0 (value is 176)
  - Color passive matrix (STN) display: DSS.DISPC_VIDn_PRELOAD[11:0] PRELOAD = 0x110 (value is 272)
  - Monochrome passive matrix (STN) display: DSS.DISPC_VIDn_PRELOAD[11:0] PRELOAD = 0x1B0 (value is 432)

- Base address of the video buffer in system memory (DSS.DISPC_VIDn_BAj registers): The default value at reset time is 0x0. The horizontal resolution is one pixel because the base address is aligned on pixel size boundary. In case of YCbCr 4:2:2 formats, the resolution is 2 pixels. In case of RGB24 packed format, the resolution is 4 pixels. The vertical resolution is one line. The register DSS.DISPC_VIDn_BA0 defines the base address of the even field, and DSS.DISPC_VIDn_BA1 defines the base of the odd field in the case of an external synchronization and based on the value of the input signal DISPC_FID and the polarity. To improve system throughput, the base address should be aligned on the burst size boundary.
- Video FIFO threshold (DSS.DISPC_VIDn_FIFO_THRESHOLD register): The low threshold (DSS.DISPC_VIDn_FIFO_THRESHOLD[11:0] VIDFIFOLOWTHRESHOLD) and the high threshold (DSS.DISPC_VIDn_FIFO_THRESHOLD[27:16] VIDFIFOHIGHTHRESHOLD) values define the FIFO DMA behavior. When the low level is reached, one or more requests are issued to the L3-based interconnect to fill up the FIFO to reach the high threshold. A request is issued as long as the FIFO has enough space available to accept a burst. The DMA engine then waits until the low level is reached to restart the requests. By setting the DSS.DISPC_CONFIG[14] FIFOMERGE bit to 1, users merge the three FIFOs (GFX, VID1, and VID2). In this case, the low threshold (the DSS.DISPC_VIDn_FIFO_THRESHOLD[11:0] VIDFIFOLOWTHRESHOLD bit field with n corresponding to the active video channel 1 or 2) and the high threshold (DSS.DISPC_VIDn_FIFO_THRESHOLD[27:16] VIDFIFOHIGHTHRESHOLD bit field with n corresponding to the active video channel 1 or 2) values must be programmed with a multiplier factor of three (3 x value). By default, the FIFOs are not merged (the DSS.DISPC_CONFIG[14] FIFOMERGE bit reset value is 0).
- Video buffer width (DSS.DISPC_PICTURE_SIZE[10:0] VIDORGSIZEX): The default value at reset time is 0x0 (1 pixel). The buffer width in system memory is from 1 up to 2048 pixels. All the integer values in the range [1:2048] are allowed. Software users must program this bit field to the value minus 1.
- Video buffer height (DSS.DISPC_PICTURE_SIZE[26:16] VIDORGSIZEY): The default value at reset time is 0x0 (1 pixel). The buffer height in system memory is from 1 up to 2048 pixels. All the integer values in the range [1:2048] are allowed. Software users must program this field to the value minus 1.
- Video data endianness (DSS.DISPC_VIDn_ATTRIBUTES[17] VIDENDIANNESS bit, with n=1 or 2): This bit indicates the endianness (little or big) of the video pixels. The default value at reset time is 0x0 (little endian).

15.5.3.3.2 Video Configuration Register

The following shadow registers define video layer n (with n = 1 or 2) configuration:

- DSS.DISPC_CONFIG
- DSS.DISPC_VIDn_POSITION
- DSS.DISPC_VIDn_SIZE
- DSS.DISPC_VIDn_ATTRIBUTES
- DSS.DISPC_VIDn_FIR
- DSS.DISPC_VIDn_PICTURE_SIZE
- DSS.DISPC_VIDn_FIR_COEF_Hi (with i = 0 to 7)
- DSS.DISPC_VIDn_FIR_COEF_HVi (with i = 0 to 7)
- DSS.DISPC_VIDn_CONV_COEFi (with i = 0 to 4)
- The video layer n (with n = 1 or 2) is enabled/disabled by setting/resetting the DSS.DISPC_VIDn_ATTRIBUTES[0] VIDENABLE field. If the video layer is disabled, the video window does not exist on the screen and the whole video pipeline and DMA are inactive. Before enabling the video layer, a valid configuration must be set. After a register change, either the DSS.DISPC_CONTROL[6] GODIGITAL or DSS.DISPC_CONTROL[5] GOLCD bit must be set. The software must wait for the hardware to reset the bit before setting this bit. The software reset is not recommended because the application cannot ensure that the bit is reset before the hardware reset.

15.5.3.3.3 Video Window Attributes

The following fields define the attributes of video window n:

- Video format (DSS.DISPC_VIDn_ATTRIBUTES[4:1] VIDFORMAT bit field, with n = 1 or 2): The default value at reset time is 0x0 (BITMAP 1 BPP, nonsupported format by the video pipeline).The video format can be RGB16, RGB24, YUV2 4:2:2 co-DSS sited, and UYVY 4:2:2 co-sited.
- Video window X-position (DSS.DISPC_VIDn_POSITION[10:0] VIDPOSX bit field, with n = 1 or 2): The default value at reset time is 0x0 (first column starting on the left edge of the screen). The window X-position is from 0 to 2047 columns. All integer values in the range [0:2047] are allowed.
- Video window Y-position (DSS.DISPC_VIDn_POSITION[26:16] VIDPOSY bit field, with n = 1 or 2): The default value at reset time is 0x0 (first row starting at the top of the screen). The window Y-position is from 0 to 2047 rows. All integer values in the range [0:2047] are allowed.
- Video window width (DSS.DISPC_VIDn_SIZE[10:0] VIDSIZEX bit field, with n = 1 or 2): The default value at reset time is 0x0 (1 pixel). The window width is from 1 to 2048 pixels. All integer values in the range [1:2048] are allowed. The maximum bandwidth efficiency for accessing the pixels in system memory is reached when the width (in bytes) of the video window is a multiple of the video burst size defined in the DSS.DISPC_VIDn_ATTRIBUTES[15:14] VIDBURSTSIZE bit field (in bytes).

  Note: When the RGB24 packed format is selected, the width must be a multiple of 12 bytes when the DSS.DISPC_VIDn_ROW_INC register is not 1. When the DSS.DISPC_VIDn_ROW_INC register is 1, the width can be any size from 1 to 2048 pixels.

  The entire pixels of the video window must be inside the LCD screen. Depending on the width of the buffer to be displayed in the video layer and the position, the width should be adjusted by software to limit the right edge of the window inside the screen.

- Video window height (DSS.DISPC_VIDn_SIZE[26:16] VIDSIZEY bit field, with n = 1 or 2): The default value at reset time is 0x0 (1 pixel). The window height is from 1 to 2048 pixels. All integer values in the range [1:2048] are allowed. The entire pixels of the video window must be inside the LCD screen. Depending on the height of the buffer to be displayed in the video layer and the position, the height should be adjusted by software to limit the bottom edge of the window inside the screen.
- Video picture width in system memory (DSS.DISPC_VIDn_PICTURE_SIZE[10:0] VIDORGSIZEX bit field, with n = 1 or 2): The default value at reset time is 0x0 (1 pixel). The window width is from 1 to 2048 pixels. All integer values in the range [1:2048] are allowed with RGB16 and RGB24 video data. For YUV2 4:2:2 and UYVY 4:2:2 formats, the width must be a multiple of two pixels. The maximum bandwidth efficiency for accessing the pixels in system memory is reached when the width (in bytes) of the video picture is a multiple of the video burst size defined in the DSS.DISPC_VIDn_ATTRIBUTES[15:14] VIDBURSTSIZE bit field (in bytes).
- Video picture height in system memory (the DSS.DISPC_VIDn_PICTURE_SIZE[26:16] VIDORGSIZEY bit field, with n = 1 or 2): The default value at reset time is 0x0 (1 pixel). The window width is from 1 to 2048 pixels. All integer values in the range [1:2048] are allowed.
- Video Priority (DSS.DISPC_VIDn_ATTRIBUTES[23] VIDARBITRATION): The default value at reset time is 0x0. It is used to change between normal priority (value of 0) to high priority (value of 1) to change priority for the video channel vs. other channels. It can be used to give higher priority to the pipelines with real time constraint vs. non real time pipelines. For that is, pipelines associated to the LCD output in RFBI mode should have lower priority than pipelines associated to TV output.
- Video Self-Refresh (DSS.DISPC_VIDn_ATTRIBUTES[24] VIDSELFREFRESH): The default value at reset time is 0x0. It is used to use the DMA FIFO without accessing the interconnect for multiple frames. Once, the data have been loaded to the DMA FIFO for displaying the frame, they are used for the following frames.
  The sequence to activate the self-refresh is the following:
  - Frame t: The bit field should be set at anytime during frame
  - Frame t+1: Fetch of the data in the DMA FIFO and display of the frame
  - Frame t+2: No access to the L3 interconnect, DMA FIFO uses to provide the pixels

  The sequence to deactivate the self-refresh is the following:
  - Frame t: No access to the L3 interconnect, DMA FIFO uses to provide the pixels, bit field can be changed at any time during the frame

– Frame t+1: Fetch of the data from system memory using the L3 interconnect

15.5.3.3.4 Video Up-/Down-Sampling Configuration

The video horizontal up-/downsampling block for video pipeline n (with n = 1 or 2) is enabled/disabled by setting/resetting the DSS.DISPC_VIDn_ATTRIBUTES[5] VIDRESIZEENABLE bit.

The video vertical up-/downsampling block for video pipeline n is enabled/disabled by setting/resetting the DSS.DISPC_VIDn_ATTRIBUTES[6] VIDRESIZEENABLE bit.

Set a valid configuration before enabling the video up-/downsampling block.

Note: Vertical and horizontal downsampling are limited to a 1/4 resize factor.

After a register change, either the DSS.DISPC_CONTROL[6] GODIGITAL or DSS.DISPC_CONTROL[5] GOLCD bit must be set. The software must wait until the hardware resets this bit before setting it. The software reset is not recommended because the application cannot ensure that the bit is reset before the hardware reset.

The following fields define the configuration of the video up-/downsampling block for video pipeline n:

- Vertical up-/downsampling increment value (DSS.DISPC_VIDn_FIR[27:16] VIDFIRVINC bit field, with n = 1 or 2): The unsigned integer value range is [1:4096]. The software calculates the value using the following equation:

$$VIDFIRVINC[12:0] = 1024 \times \left(\frac{VIDORGSIZEY[10:0]}{VIDSIZEY[10:0]} \right)$$

dss-E093 (15-4)

Notes:

- If the VIDFIRVINC[11:0] bit field value is greater than 4096, it is clipped to 4096. If VIDSIZEY[10:0] equals 0x1, VIDSIZEY[10:0] is replaced by 0x2 in the previous equation.
- The VIDORGSIZEY[10:0] and VIDSIZEY[10:0] bit field values must be programmed with the value desired minus 1.

- Horizontal up-/downsampling increment value (DSS.DISPC_VIDn_FIR[11:0] VIDFIRHINC bit field, with n = 1 or 2): The unsigned integer value range is [1:4096]. The software calculates the value using the following equation:

$$VIDFIRHINC[12:0] = 1024 \times \left(\frac{VIDORGSIZEX[10:0]}{VIDSIZEX[10:0]} \right)$$

dss-E094 (15-5)

Notes:

- If the VIDFIRHINC[11:0] bit field value is greater than 4096, it is clipped to 4096. If VIDSIZEX[10:0] equals 1, VIDSIZEX[10:0] is replaced by 2 in the previous equation.
- The VIDORGSIZEX[10:0] and VIDSIZEX[10:0] bit field values must be programmed with the value desired minus 1.

- Vertical up-/downsampling accumulator value (DSS.DISPC_VIDn_ACCUl[25:16] VIDVERTICALACCU bit field): The unsigned integer value range is [0:1023]. The accumulator value indicates in which phase the vertical filtering starts. The value 0 indicates that 0 is the first phase used by the hardware to generate the first data (see Table 15-46).
- Vertical up-/downsampling line buffer configuration (DSS.DISPC_VIDn_ATTRIBUTES[22] VIDLINEBUFFERSPLIT bit): The default value at reset time is 0x0 (line buffers are not split). The backward compatibility is maintained versus OMAP2420 and OMAP2430 devices. When the bit field is set, each line buffer is split into two line buffers to be able to use six line buffers instead of three.
- Vertical up-/downsampling line buffer configuration (DSS.DISPC_VIDn_ATTRIBUTES[21] VIDVERTICALTAPS bit): The default value at reset time is 0x0 (3-tap configuration is used). If the bit is reset, the 3-tap configuration is used. The backward compatibility is maintained versus OMAP2420 and

OMAP2430 devices. When the bit is set, the 5-tap configuration is used and the DSS.DISPC_VIDn_ATTRIBUTES[22] VIDLINEBUFFERSPLIT bit must be set to 1.

- Vertical up-/downsampling line buffer configuration (DSS.DISPC_VIDn_ATTRIBUTES[20] VIDDMAOPTIMIZATION bit): The default value at set time is 0x0 (no optimization). If the bit is set, the DMA engine fetches two pixels for each 32-bit OCP request (RGB16 and YUV422) while doing 90- and 270-degree rotation. If the bit is clear, the DMA engine fetches one pixel for each 32-bit OCP request (RGB16 and YUV422) while doing 90- and 270-degree rotation. The width and height of picture should be even to use the optimization.

Note: If the 5-tap resizer is used for RGB16 and YUV422 picture formats, the width of the input picture must be a multiple of 2 pixels and more than 5 pixels. This leads to the following register configuration:

DISPC_VIDn_ATTRIBUTES[21] VIDVERTICALTAPS == 1

DISPC_VIDn_PICTURE_SIZE[10:0] VIDORGSIZEX > 4 and even

- Horizontal up-/downsampling accumulator value (DSS.DISPC_VIDn_ACCUI[9:0] VIDHORIZONTALACCU bit field): The unsigned integer value range is [0:1023]. The accumulator value indicates in which phase the horizontal filtering starts. The value 0 indicates that 0 is the first phase used by the hardware to generate the first data (see Table 15-46).

Table 15-46. Vertical/Horizontal Accumulator Phase

Accumulator Value	Phases f
0	0
128	1
256	2
384	3
512	4
640	5
768	6
896	7

- Vertical up-/downsampling coefficients (DSS.DISPC_VIDn_FIR_COEF_HVi registers, with n = 1 or 2, i = 0 to 7): The 3-tap vertical up-/downsampling coefficients are defined in these registers. There are eight registers for the eight phases with three coefficients for each, or a total of 24 programmable coefficients for the vertical up-/downsampling block. Each register contains two 8-bit signed coefficients and one 8-bit unsigned coefficient (the central one).

  In addition, there are 2-tap vertical up/downsampling coefficients defined in DSS.DISPC_VIDn_FIR_COEF_Vi registers. There are 8 registers for the 8 phases with 2 coefficients for each of them so a total of 16 programmable coefficients for the vertical up-/downsampling block used in addition of the 3-tap registers defined above. Each register contains two 8-bit signed coefficients. In case of 5-tap configuration, both sets of registers DSS.DISPC_VIDn_FIR_COEF_HVi and DSS.DISPC_VIDn_FIR_COEF_Vi are used. In case of 3-tap configuration, only one set of registers DSS.DISPC_VIDn_FIR_COEF_HVi is used.
- Horizontal up-/downsampling coefficients (DSS.DISPC_VIDn_FIR_COEF_Hi and DISPC_VIDn_FIR_COEF_HVi registers, with n = 1 or 2, i = 0 to 7): The DSS.DISPC_VIDn_FIR_COEF_Hi register and the DSS.DISPC_VIDn_FIR_COEF_HVi register define the 5-tap horizontal up-/downsampling coefficients. There are eight registers for the eight phases with five coefficients for each register, or a total of 40 programmable coefficients for the horizontal up-/downsampling block.

  Each DSS.DISPC_VIDn_FIR_COEF_Hi register contains three 8-bit signed coefficients and one 8-bit unsigned coefficient (the central one). Each DSS.DISPC_VIDn_FIR_COEF_HVi register contains one 8-bit signed coefficient.

  The programmable coefficient for the FIR up-/downsampling method must be adjusted based on application needs. For more details on scaling programming settings, see Section 15.6.1.

15.5.3.3.5 Video Color Space Conversion Configuration

The DSS.DISPC_VIDn_CONV_COEFi registers (with i = 0 to 4) has nine 11-bit coefficients defined for the programmable color space conversion block for video pipeline n (with n = 1 or 2).

The standard register coefficients are:

YCbCr-to-RGB Registers (VidFullRange=0)

$$\begin{bmatrix} R \\ G \\ B \end{bmatrix} = \frac{1}{256} * \begin{bmatrix} RY & RCr & RCb \\ GY & GCr & GCb \\ BY & BCr & BCb \end{bmatrix} * \begin{bmatrix} Y - 16 \\ Cr - 128 \\ Cb - 128 \end{bmatrix}$$

dssE095 (15-6)

YCbCr to RGB Registers (VidFullRange=1)

$$\begin{bmatrix} R \\ G \\ B \end{bmatrix} = \frac{1}{256} * \begin{bmatrix} RY & RCr & RCb \\ GY & GCr & GCb \\ BY & BCr & BCb \end{bmatrix} * \begin{bmatrix} Y \\ Cr - 128 \\ Cb - 128 \end{bmatrix}$$

dss-E096 (15-7)

Table 15-47 lists the color space conversion register values.

Table 15-47. Color Space Conversion Register Values

Coefficients	BT.601-5	BT.601-5 Range [0:255]	BT.709	BT.709 Range [0:255]
RY	298	256	298	256
RCr	409	351	459	394
RCb	0	0	0	0
GY	298	256	298	256
GCr	–208	–179	–137	–118
GCb	–100	–86	–55	–47
BY	298	256	298	256
BCr	0	0	0	0
BCb	517	443	541	465
VidFullRange	0	1	0	1

15.5.3.4 Rotation/Mirroring Display Subsystem Settings

This section describes rotation/mirroring settings. The device provides flexible mechanisms for an efficient implementation of rotation using the display-subsystem, its DMA engine, and the rotation engine of the SMS module. Depending on whether the image data is located in on-chip SRAM or external SDRAM, either a DMA rotation or a VRFB rotation is used. When configuring the rotation, the image data format (RGB or YUV) must also be taken into account. The video pipelines also perform the interpolation of YUV image data and the color conversion from YUV into RGB format for displaying the images on an LCD screen.

15.5.3.4.1 Image Data Formats

To understand the programming of the rotation mechanisms described underneath, the supported representations of the image data in memory must also be considered. Differences exist between the supported formats on the graphics and video pipelines. The graphics pipeline supports RGB and RGB with a color look-up table (CLUT), whereas the video pipelines support two versions of YUV 4:2:2 and RGB16. In the case of YUV, the interpolation and color conversion hardware in the video pipelines converts the image data to the RGB format suitable for the LCD screen.

The graphics pipeline supports:

- 1-, 2- ,4-, and 8-bits-per-pixel color look-up table
- 12-, 16-, and 24-bits-per-pixel RGB

The video pipelines support the following data formats (always 2 bytes/pixel):

- RGB16
- YUV2
- UYVY

For more information on the graphics data formats, please refer to Section 15.4.2.2, *Graphics Pipeline*.

For more information on the video data formats, please refer to Section 15.4.2.3, *Video Pipeline*.

In the video pipeline, the YUV 4:2:2 format is converted into a full YUV 4:4:4 format by interpolation of the chrominance values from neighboring pixels. After this interpolation is completed, the conversion to the RGB format (suitable for displaying the image on the LCD screen) is performed.

For more information on YUV 4:2:2 to RGB conversion, please refer to Section 15.4.2.3.2, *Color Space Conversion*.

15.5.3.4.2 Image Data from On-Chip SRAM

For image data located in the on-chip SRAM, the DSS DMA is used to perform 90-degree, 180-degree, and 270-degree rotation. This is done by using the double-indexed addressing mode of the DMA. This addressing mode allows a pixel and a row increment to be specified. These address increments are used after each pixel or each row (line), respectively. Figure 15-124 illustrates the principle steps for a 90-degree rotation.

Figure 15-124. 90° DMA Rotation Example

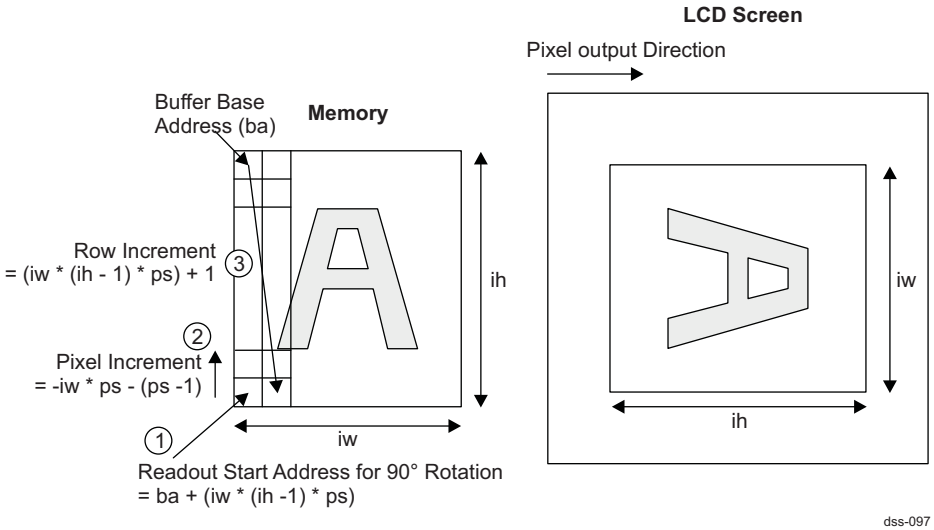


Table 15-48. 90-degree DMA Rotation Example Description

Parameter	Description	Additional parameters for formulas
ba	Buffer Base Address in Memory	
IW	Image Width in pixels	iw = IW-1
IH	Image Height in pixels	ih = IH-1
ps	Pixel Size (in bytes)	

Figure 15-124 shows how the image is stored in memory and how it is read out to achieve a 90-degree rotated orientation. The first pixel for the 0-degree orientation is located at the buffer base address (ba). If the image is to be shown on an LCD screen with a 90-degree rotation, the readout starts at the 90-degree base address (1). To proceed from one pixel to the next in the same line in the rotated orientation, the pixel increment (2) must be applied. At the end of each line of the rotated view, the row increment (3) is the offset to advance to the beginning of the next line in the memory buffer.

Hence, by setting the three DMA parameters (base address, pixel increment, and row increment), a 90-degree rotation can be achieved, as can 180-degree and 270-degree rotation. Each of the parameters can also be combined with an optional mirroring on the vertical axis.

Following there is a description the setup required to perform the rotation via the DSS DMA. This rotation mechanism is used when the image data is stored in internal SRAM.

15.5.3.4.2.1 Rotation/Mirroring Registers

To set up the rotation and/or mirroring, the following registers must be programmed:

- Graphics pipeline (GFX):
  - DSS.DISPC_GFX_BAj
  - DSS.DISPC_GFX_PIXEL_INC
  - DSS.DISPC_GFX_ROW_INC
  - DSS.DISPC_GFX_ATTRIBUTES
  - DSS.DISPC_GFX_POSITION
  - DSS.DISPC_GFX_SIZE
  - DSS.DISPC_GFX_FIFO_THRESHOLD
- Video pipelines (VID) 1 and 2:
  - DSS.DISPC_VIDn_BAj
  - DSS.DISPC_VIDn_PIXEL_INC
  - DSS.DISPC_VIDn_ROW_INC
  - DSS.DISPC_VIDn_ATTRIBUTES
  - DSS.DISPC_VIDn_POSITION
  - DSS.DISPC_VIDn_SIZE
  - DSS.DISPC_VIDn_CONV_COEF0 to DSS.DISPC_VIDn_CONV_COEF4
  - DSS.DISPC_VIDn_FIFO_THRESHOLD
- DSS.DISPC_xxx_BAj: These registers contain the base address of the image data at which the DSS DMA transfer of the image starts. The register values depend on the rotation chosen (see Table 15-49 for the formulas). When using the LCD interface, the registers DISPC_xxx_BA0 are used. The registers DISPC_xxx_BA1 are only used for the TV output.
- DSS.DISPC_xxx_PIXEL_INC[15:0]: This bit field contains the DMA addressing increment after each pixel. This bit field is used to perform the DSS DMA rotation (see Table 15-49 for the formula).

Note: When the RGB24 packet format is selected, the only valid value is 1.

- DSS.DISPC_xxx_ROW_INC[31:0]: This bit field contains the DMA addressing increment after each row (line) of pixels. This bit field is used to perform the DSS DMA rotation (see Table 15-49 for the formula).

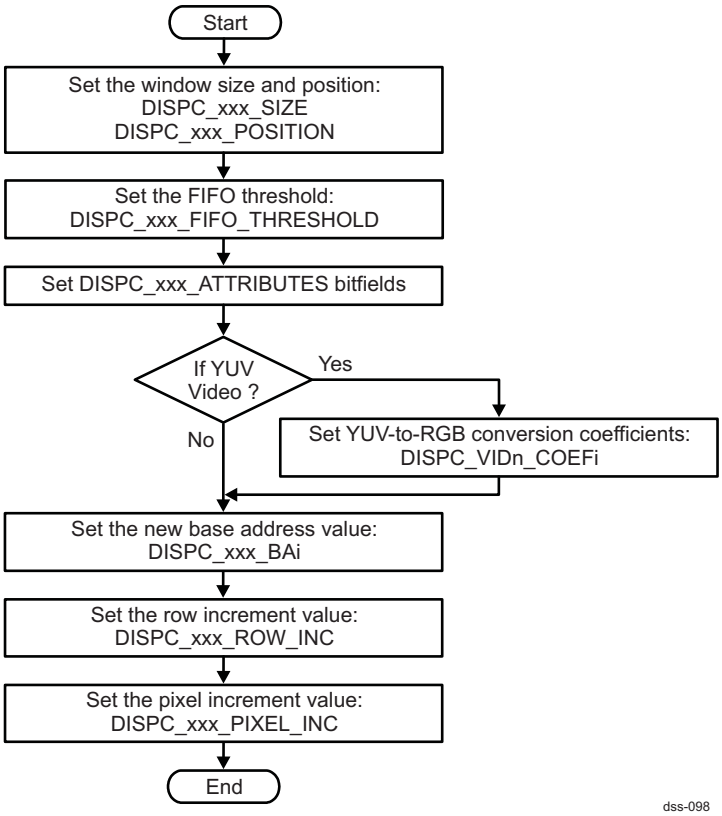
Note: When the RGB24 packet format is selected, the valid values are 1 and any value multiples of 12 bytes (4x32 bit). When the value is a multiple of 12 bytes, the width must be a multiple of 12 bytes. When the value is 1, the width can be any size from 1 to 2048 pixels.

- DSS.DISPC_xxx_ATTRIBUTES: These registers contain the main settings for the pipeline, such as the image data format, the rotation value, and the enable bit for the pipeline. The bit fields of these registers play a role in the rotation and in the image data format setup.
  - The following bit fields are used by the graphics pipeline to set up the image format:
    - GFXENABLE: Set this field to activate the hardware path in use.
    - GFXFORMAT: Use this field to specify the format of the graphic frame.
    - GFXROTATION: Set this field to the value corresponding to the rotation angle desired only if the frame contains RGB24 pixel data; otherwise, set it to 0x0 regardless of the degree of rotation.
    - GFXREPLICATIONENABLE: Use this bit to determine whether the encoded pixel data in RGB formats (RGB12 and RGB16) is extended to 24-bit format with or without replication of the MSB to fill the LSBs of each color component. If the replication logic is turned off, the LSB parts are filled with 0s. It is recommended to always enable this feature.
    - GFXCHANNELOUT: Set this field based on whether the frame is to be rendered on the LCD or on the TV set.
  - The following bit fields for the two video pipelines:
    - VIDENABLE: Set this field to activate the hardware path in use.
    - VIDFORMAT: Use this field to specify the format of the video frame (RGB16 or YUV422).
    - VIDCOLORCONVENABLE: If the video is in YUV422 format, set this field to enabled.
    - VIDROTATION: Set this field to the value corresponding to the rotation angle desired only if the frame contains non-RGB pixel data; otherwise, set it to 0x0 regardless of the degree of rotation. See Section 15.5.3.4.4 for more information.
    - VIDROWREPEATENABLE: Set this field to enabled only if the frame contains YUV pixel data and the rotation is 90-degree or 270-degree so that the row pixel data are fetched twice to extract both Y components. See Section 15.5.3.4.4 for more information.
    - VIDCHANNELOUT: Set this field based on whether the frame is to be rendered on the LCD or on the TV set.
- DSS.DISPC_xxx_POSITION: Use this register to configure the position of the window.
- DSS.DISPC_xxx_SIZE: Use this register to configure the size of the window.
- DSS.DISPC_VIDn_CONV_COEF0, DSS.DISPC_VIDn_CONV_COEF1, DSS.DISPC_VIDn_CONV_COEF2, DSS.DISPC_VIDn_CONV_COEF3, and DSS.DISPC_VIDn_CONV_COEF4: These registers contain the conversion coefficients required for YUV-to-RGB color conversion.
- DSS.DISPC_xxx_FIFO_THRESHOLD: Set the low threshold (DSS.DISPC_GFX_FIFO_THRESHOLD[11:0] GFXFIFOLOWTHRESHOLD) and the high threshold (DSS.DISPC_GFX_FIFO_THRESHOLD[27:16] GFXFIFOHIGHTHRESHOLD) values to define the FIFO DMA behavior. When the low level is reached, one or more requests are issued to the L3-based interconnect to fill up the FIFO to reach the high threshold. The DMA engine then waits until the low level is reached to restart the requests.

15.5.3.4.2.2 DMA Register Settings

To configure the display controller for rotation and/or mirroring, use the following settings:

Figure 15-125. Rotation/Mirroring Settings



Note: These registers are shadow registers. To take into account the new values, software users must set the DSS.DISPC_CONTROL[5] GOLCD bit to 1.

Table 15-49 details the base address, the row and pixel increment values to access the buffer in memory (contiguous pixels) except for the RGB24 packet format. Table 15-50 lists the rotation register settings for RGB24 packet format (only for the two video pipelines).

Table 15-49. DMA Rotation Register Settings

Rotation	Registers	GFX/VIDx (1)
0 degree	DSS.DISPC_xxx_BAj	ba
	DSS.DISPC_xxx_PIXEL_INC	1
	DSS.DISPC_xxx_ROW_INC	1
90 degrees	DSS.DISPC_xxx_BAj	ba + (iw x (ih-1) x ps)
	DSS.DISPC_xxx_PIXEL_INC	-(iw x ps) - 1
	DSS.DISPC_xxx_ROW_INC	(iw x (ih-1)) x ps + 1

(1)

- ba = start address of image buffer in memory
- iw = image width in pixels per row - 1 (for YUV: pixels per row divided by 2)
- ih = image height - 1 (number of rows)
- ps = pixel size in bytes (RGB: 2 bytes per pixel, YUV: 4 bytes per pixel)

See Table 15-48 for more information of these parameters.

Table 15-49. DMA Rotation Register Settings (continued)

Rotation	Registers	GFX/VIDx [1]
180 degrees	DSS.DISPC_xxx_BAj	ba + (iw x ih-1) x ps
	DSS.DISPC_xxx_PIXEL_INC	-2 x ps
	DSS.DISPC_xxx_ROW_INC	-2 x ps
270 degrees	DSS.DISPC_xxx_BAj	ba + (iw -1) x ps
	DSS.DISPC_xxx_PIXEL_INC	(iw -1) x ps + 1
	DSS.DISPC_xxx_ROW_INC	-(iw x (ih-1)) x ps - ps + 1

Note: In case of RGB16 format and optimization enabled, the base address is aligned on a 32-bit boundary and the number of bytes to skip is a multiple of 4 bytes.

Table 15-50. Video Rotation Register Settings (With RGB24 Packet Format)

Rotation	Registers (with n = 1 or 2)	SDRAM Direct Access [1]
0 degree	DSS.DISPC_VIDn_BAj	ba
	DSS.DISPC_VIDn_PIXEL_INC	1
	DSS.DISPC_VIDn_ROW_INC	1
180 degrees	DSS.DISPC_VIDn_BAj	ba+(iw*ih*3)-4
	DSS.DISPC_VIDn_PIXEL_INC	-7
	DSS.DISPC_VIDn_ROW_INC	-7

(1)

- ba = physical base address of the buffer in the system memory (top-left corner of the picture)
- iw = number of pixels per row - 1 (original picture in system memory) for BITMAP and RGB formats and number of pixels per row divided by 2 for YUB422 formats
- h = number of lines - 1 (original picture in system memory)
- ps = pixel size in bytes (BITMAP8: 1 byte; RGB16: 2 bytes; YUV422: 4 bytes)

See Table 15-48 for more information of these parameters.

The DMA rotation described above can be also combined with an optional mirroring on the vertical axis (see Figure 15-126). The only settings that must be changed to achieve this mirroring are the registers described above: DISPC_xxx_BAj, DISPC_xxx_PIXEL_INC, and DISPC_xxx_ROW_INC. Table 15-51 provides the DMA setup formulas for rotation with mirroring to access the buffer in memory (contiguous pixels) except for the RGB24 packet format.

Figure 15-126. 90° Rotation With Mirroring

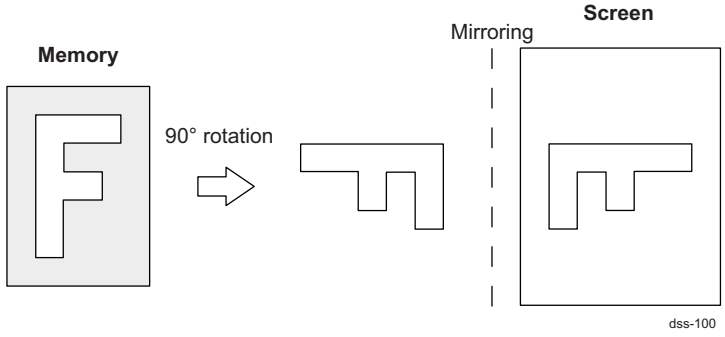


Table 15-51. Register Settings for DMA Rotation With Mirroring

Rotation	Registers	GFX/VIDx [1]
0 degree	DSS.DISPC_xxx_BAj	ba + (iw-1) x ps
	DSS.DISPC_xxx_PIXEL_INC	-2 x ps + 1
	DSS.DISPC_xxx_ROW_INC	2 x (iw-1) x ps + 1
90 degrees	DSS.DISPC_xxx_BAj	ba
	DSS.DISPC_xxx_PIXEL_INC	(iw-1) x ps
	DSS.DISPC_xxx_ROW_INC	(-iw x (ih-1)) x ps
180 degrees	DSS.DISPC_xxx_BAj	ba + iw x (ih-1) x ps
	DSS.DISPC_xxx_PIXEL_INC	1
	DSS.DISPC_xxx_ROW_INC	-2 x iw x ps
270 degrees	DSS.DISPC_xxx_BAj	ba + (iw x ih - 1) x ps
	DSS.DISPC_xxx_PIXEL_INC	(iw -1) x ps + 1
	DSS.DISPC_xxx_ROW_INC	- (iw x (ih-1)) x ps - ps + 1

(1)

- ba = start address of image buffer in memory
- iw = image width in pixels per row - 1 (for YUV: pixels per row divided by 2)
- ih = image height - 1 (number of rows)
- ps = pixel size in bytes (RGB: 2 bytes per pixel, YUV: 4 bytes per pixel)

See Table 15-48 for more information of these parameters.

15.5.3.4.3 Image Data from External SRAM

The device offers a rotation engine in the SMS called the VRFB. This rotation method must be used for maximum efficiency when the image data is located in SDRAM. To use the VRFB rotation, both the SMS module and the DSS DMA must be configured.

In the DSS, the same registers must be set up as described for DMA rotation (see Section 15.5.3.4.2, *Image Data from On-Chip SRAM*). The differences are in the setup of the following registers:

- DISPC_xxx_ROW_INC: This field contains the DMA addressing increment after each row (line) of pixels. This field is used to add the remaining delta at the end of each line, to reach the 2048-pixel line size of the VRFB (see Table 15-52 for the formula).
- DISPC_xxx_PIXEL_INC: This field contains the DMA addressing increment after each pixel. For VRFB rotation, this value is set always to 1 (see Table 15-52).

Table 15-52. VRFB Rotation - DMA Settings

Rotation	Registers	GFX/VIDx [1]
0 degree	DSS.DISPC_xxx_BAj	VBA0
	DSS.DISPC_xxx_PIXEL_INC	1
	DSS.DISPC_xxx_ROW_INC	(2048 - iw) x ps + 1
90 degrees	DSS.DISPC_xxx_BAj	VBA90 + offset
	DSS.DISPC_xxx_PIXEL_INC	1
	DSS.DISPC_xxx_ROW_INC	(2048 - ih) x ps + 1
180 degrees	DSS.DISPC_xxx_BAj	VBA180 + offset
	DSS.DISPC_xxx_PIXEL_INC	1
	DSS.DISPC_xxx_ROW_INC	(2048 - iw) x ps + 1

(1)

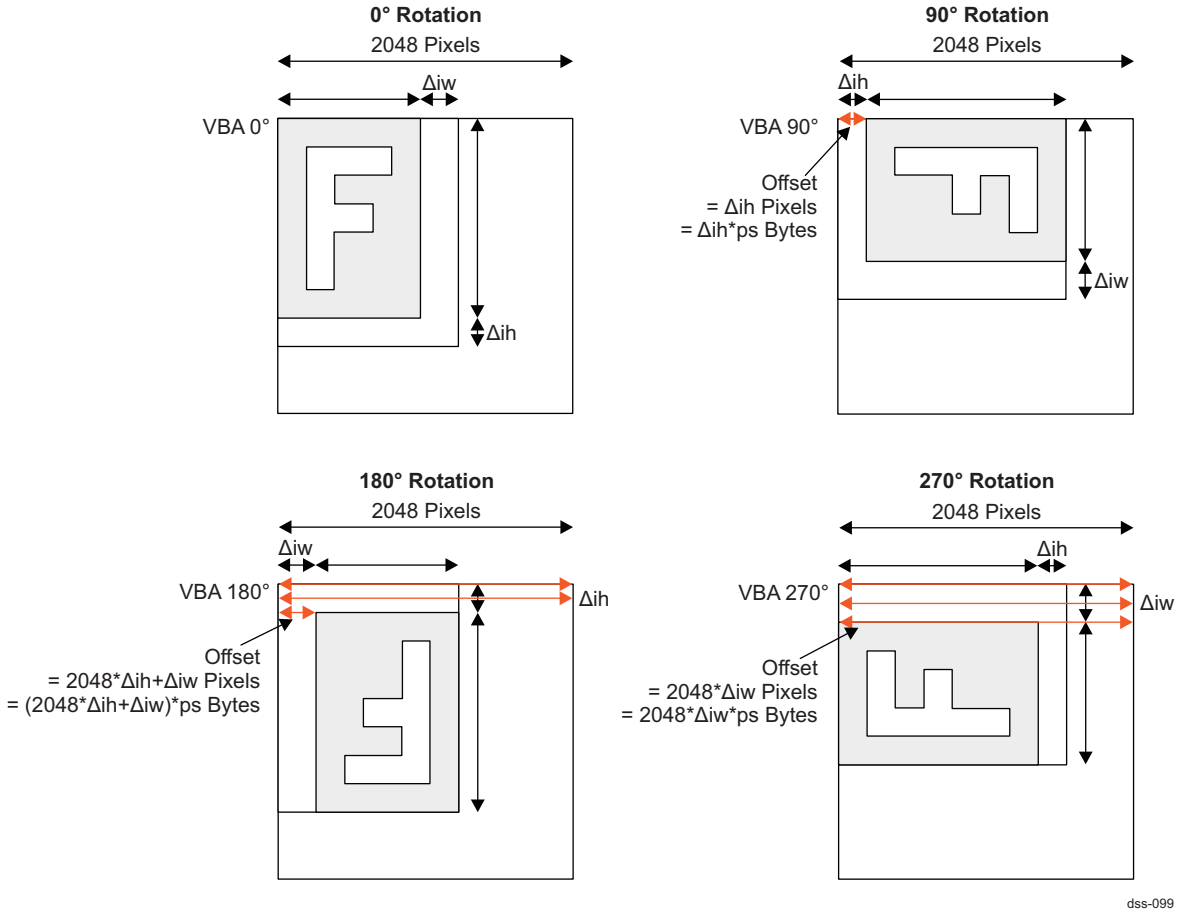
- VBAx = virtual address of the chosen VRFB context and orientation
- iw = image width (width in pixels for RGB, width in pixels divided by 2 for YUV)
- ih = image height
- ps = pixel size in bytes (2 bytes per pixel for RGB, 4 bytes per pixel for YUV)
- Offset = see below

Table 15-52. VRFB Rotation - DMA Settings (continued)

Rotation	Registers	GFX/VIDx [1]
270 degrees	DSS.DISPC_xxx_BAj	VBA270 + offset
	DSS.DISPC_xxx_PIXEL_INC	1
	DSS.DISPC_xxx_ROW_INC	(2048 - ih) x ps + 1

Figure 15-127 provides the offset values that must be added to the virtual base addresses for 90-degree, 180-degree, and 270-degree rotation. This offset is applicable only when the defined image size in the VRFB module is greater than the actual image size, because it must be a multiple of the page width and height. In the example discussed above, the image height was set to 256 lines, instead of 240, because the page height was 32 lines. This offset must be added to the virtual base addresses, because the VRFB module is not aware of the actual image size. Figure 15-127 illustrates why this occurs and how the offset is calculated.

Figure 15-127. Offset for VRFB Rotation



Δiw = Image width delta between the actual image width and the programmed image width because of the page width

Δih = Image height delta between the actual image height and the programmed image height because of the page height:

- Offset 90-degree: Δih pixels = Δih x ps bytes
- Offset 180-degree: 2048 x Δih+ Δiw pixels = 2048 x Δih x ps+ Δiw x ps bytes
- Offset 270-degree: 2048 x Δiw pixels = 2048 x Δiw x ps bytes

In the example given above, the delta in the image height is 256 – 240 = 16 lines (Δih = 16), whereas the exact value of the width can be programmed (Δiw = 0). In that case, the resulting offset values are:

- YUV:
  - Offset 90-degree: 16 x 4 bytes
  - Offset 180-degree: 2048 x 16 x 4 bytes
  - Offset 270-degree: 0 bytes
- RGB:
  - Offset 90-degree: 16 x 2 bytes
  - Offset 180-degree: 2048 x 16 x 2 bytes
  - Offset 270-degree: 0 bytes

As with DMA rotation, mirroring along the vertical axis can be added on top of the rotation when using the VRFB. This mirroring is achieved by combining an appropriate VRFB rotation with a readout of the VRFB by the DSS DMA, going backwards from the last line to the first line of the rotated image. Table 15-53 provides the DMA settings for VRFB rotation with mirroring.

Table 15-53. VRFB Rotation With Mirroring - DMA Settings

Rotation	Registers	GFX/VIDx (1)
0 degree	DSS.DISPC_xxx_BAj	VBA180 + 2048 x (ih - 1) x ps + offset
	DSS.DISPC_xxx_PIXEL_INC	1
	DSS.DISPC_xxx_ROW_INC	-(2048 + iw) x ps + 1
90 degrees	DSS.DISPC_xxx_BAj	VBA270 + 2048 x (iw - 1) x ps + offset
	DSS.DISPC_xxx_PIXEL_INC	1
	DSS.DISPC_xxx_ROW_INC	-(2048 + ih) x ps + 1
180 degrees	DSS.DISPC_xxx_BAj	VBA0 + 2048 x (ih - 1) x ps
	DSS.DISPC_xxx_PIXEL_INC	1
	DSS.DISPC_xxx_ROW_INC	-(2048 + iw) x ps + 1
270 degrees	DSS.DISPC_xxx_BAj	VBA90 + 2048 x (iw - 1) x ps + offset
	DSS.DISPC_xxx_PIXEL_INC	1
	DSS.DISPC_xxx_ROW_INC	-(2048 + ih) x ps + 1

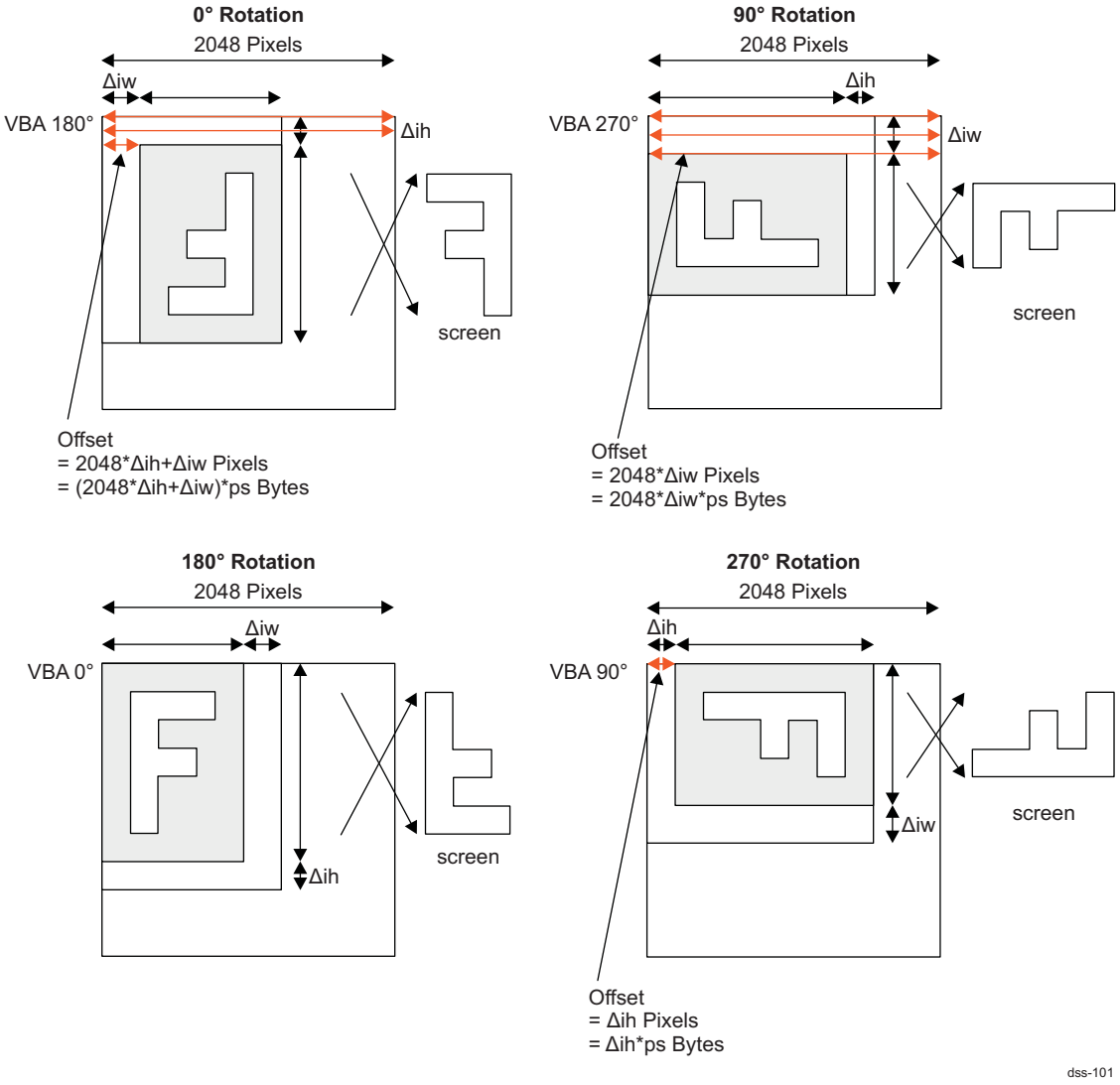
(1)

- VBAx = virtual address of the chosen VRFB context and orientation
- iw = image width (width in pixels for RGB, width in pixels divided by 2 for YUV)
- ih = image height
- ps = pixel size in bytes (2 bytes per pixel for RGB, 4 bytes per pixel for YUV)
- Offset = see below

Some offsets are required if there is a delta between the programmed image size in the VRFB and the actual image size. As shown in Figure 15-128, this applies to 0-degree rotation with mirroring (using VBA 180°), 90° rotation with mirroring (VBA 270-degree), and 270-degree rotation with mirroring (VBA 90-degree). The offsets are calculated in this manner:

- Offset 0-degree (mirroring): 2048 x Δih + Δiw pixels = 2048 x Δih x ps + Δiw x ps bytes
- Offset 90-degree (mirroring): 2048 x Δiw pixels = 2048 x Δiw x ps bytes
- Offset 270-degree (mirroring): Δih pixels = Δih x ps bytes

Figure 15-128. Offset for VRFB Rotation With Mirroring



15.5.3.4.4 Additional configuration when using YUV format

The rotation flag (DSS.DISPC_VIDn_ATTRIBUTES[13] VIDROTATION bit , with n = 1 or 2) and the repeat flag (DSS.DISPC_VIDn_ATTRIBUTES[18] VIDROWREPEATENABLE) indicate the rotation to apply to the video-encoded pixels from the SDRAM and SRAM. The 2D addressing mode is used, but even when accessing the SDRAM buffer, some post-processing must be performed on the YUV 4:2:2 data depending on the rotation. These bits are set only when the video format is not RGB; otherwise, the bit field is reset to 0. Table 15-54 and Table 15-55 describe the configuration of these registers.

Table 15-54. Video Rotation Register Settings (YUV Only)

Rotation	Registers (with n = 1 or 2)	SDRAM + Rotation engine
0 degree	DSS.DISPC_VIDn_ATTRIBUTES[13] VIDROTATION	0x0
	DSS.DISPC_VIDn_ATTRIBUTES[18] VIDROWREPEATENABLE	0x0
90 degrees	DSS.DISPC_VIDn_ATTRIBUTES[13] VIDROTATION	0x1
	DSS.DISPC_VIDn_ATTRIBUTES[18] VIDROWREPEATENABLE	0x1

Table 15-54. Video Rotation Register Settings (YUV Only) (continued)

Rotation	Registers (with n = 1 or 2)	SDRAM + Rotation engine
180 degrees	DSS.DISPC_VIDn_ATTRIBUTES[13] VIDROTATION	0x2
	DSS.DISPC_VIDn_ATTRIBUTES[18] VIDROWREPEATENABLE	0x0
270 degree	DSS.DISPC_VIDn_ATTRIBUTES[13] VIDROTATION	0x3
	DSS.DISPC_VIDn_ATTRIBUTES[18] VIDROWREPEATENABLE	0x1

Table 15-55. Video Rotation With Mirroring Register Settings (YUV only)

Rotation	Registers (with n = 1 or 2)	SDRAM + Rotation engine
0 degree	DSS.DISPC_VIDn_ATTRIBUTES[13] VIDROTATION	0x2
	DSS.DISPC_VIDn_ATTRIBUTES[18] VIDROWREPEATENABLE	0x0
90 degrees	DSS.DISPC_VIDn_ATTRIBUTES[13] VIDROTATION	0x1
	DSS.DISPC_VIDn_ATTRIBUTES[18] VIDROWREPEATENABLE	0x1
180 degrees	DSS.DISPC_VIDn_ATTRIBUTES[13] VIDROTATION	0x0
	DSS.DISPC_VIDn_ATTRIBUTES[18] VIDROWREPEATENABLE	0x0
270 degree	DSS.DISPC_VIDn_ATTRIBUTES[13] VIDROTATION	0x3
	DSS.DISPC_VIDn_ATTRIBUTES[18] VIDROWREPEATENABLE	0x1

Note: For YUV 4:2:2 video-encoded pixels, the hardware must always fetch a 32-bit value from the system memory to generate a YUV 4:4:4 value before YUV-to-RGB conversion.

- For 90- and 270-degree rotation, the missing chrominance samples for the odd pixels are generated by duplicating the chrominance samples of the previous even pixels.
- For 0- and 180-degree rotation, the missing chrominance samples for the odd pixels are generated by averaging the contiguous chrominance samples.

15.5.3.5 LCD-Specific Control Registers

The following registers define the LCD output configuration:

- DSS.DISPC_CONTROL
- DSS.DISPC_CONFIG
- DSS.DISPC_DEFAULT_COLOR_m (m=0)
- DSS.DISPC_TRANS_COLOR_m (m=0)
- DSS.DISPC_TIMING_H
- DSS.DISPC_TIMING_V
- DSS.DISPC_POL_FREQ
- DSS.DISPC_DIVISOR
- DSS.DISPC_SIZE_LCD
- DSS.DISPC_DATA_CYCLEk
- DSS.DISPC_CPR_COEF_R, DSS.DISPC_CPR_COEF_G, DSS.DISPC_CPR_COEF_B

Setting/resetting the DSS.DISPC_CONTROL[0] LCDENABLE bit enables/disables the LCD output. A valid configuration must be set before enabling the LCD output.

15.5.3.5.1 LCD Attributes

The following fields define the attributes of the panel connected to the display controller:

- Monochrome or color panel (the DSS.DISPC_CONTROL[2]MONOCOLOR bit)
- Passive Matrix or active Matrix panel (the DSS.DISPC_CONTROL[3] STNTFT bit)

- Color depth (the DSS.DISPC_CONTROL[9:8] TFTDATALINES bit field)
- Number of lines per panel (the DSS.DISPC_SIZE_LCD[26:16] LPP bit field)
- Number of pixels per line (the DSS.DISPC_SIZE_LCD[10:0] PPL bit field)
- 4- or 8-bit interface for Passive Matrix monochrome panel (the DSS.DISPC_CONTROL[4] M8B bit)

15.5.3.5.2 LCD Timings

The following bit fields define the timing generation of HSYNC/VSYNC:

- Horizontal front porch (the DSS.DISPC_TIMING_H[19:8] HFP bit field)
- Horizontal back porch (the DSS.DISPC_TIMING_H[31:20] HBP bit field)
- Horizontal synchronization pulse width (the DSS.DISPC_TIMING_H[7:0] HSW bit field)
- Vertical front porch (the DSS.DISPC_TIMING_V[19:8] VFP bit field)
- Vertical back porch (the DSS.DISPC_TIMING_V[31:20] VBP bit field)
- Vertical synchronization pulse width (the DSS.DISPC_TIMING_V[7:0] VSW bit field)
- On/Off control of HSYNC/VSYNC pixel clock (the DSS.DISPC_POL_FREQ[17] ONOFF bit)
- Program HSYNC/VSYNC rise or fall (the DSS.DISPC_POL_FREQ[16] RF bit)
- Invert HSYNC (the DSS.DISPC_POL_FREQ[13] IHS bit)
- Invert VSYNC (the DSS.DISPC_POL_FREQ[12] IVS bit)
- HSYNC gated (the DSS.DISPC_CONFIG[6] HSYNCGATED bit)
- VSYNC gated (the DSS.DISPC_CONFIG[7] VSYNCGATED bit)

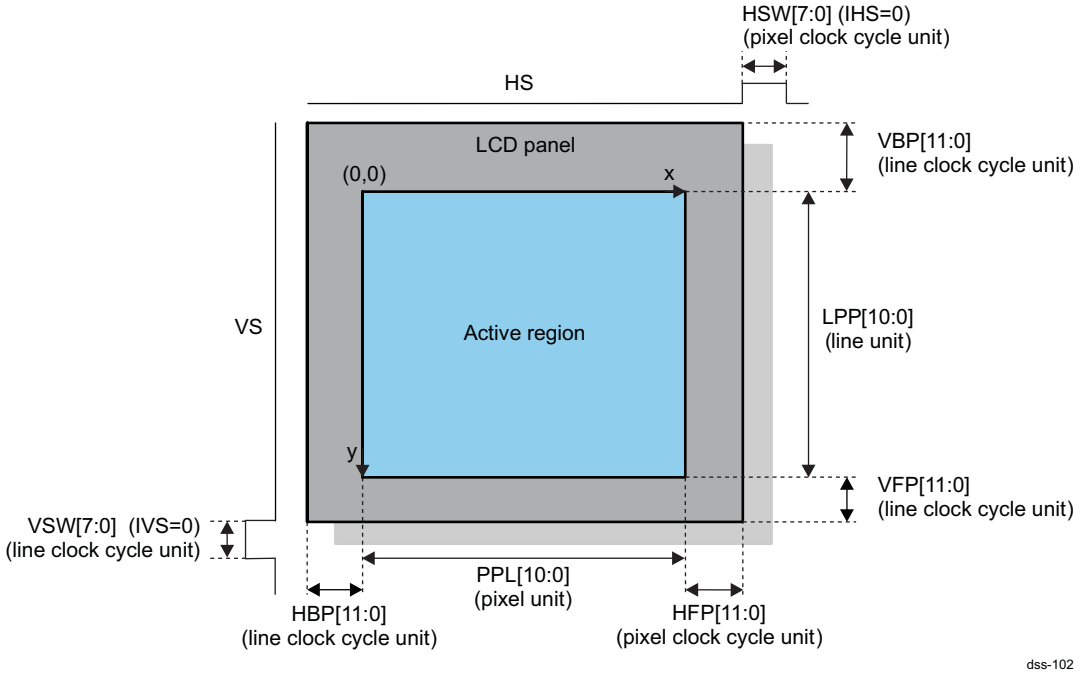
Table 15-56 describes the programming rules for LCD timing.

Table 15-56. Programming Rules

	No Downsampling	Downsampling H or V	Downsampling H + V
(HBP + HSW + HFP) * PCD	> 8	> 10	> 20

Figure 15-129 shows the timing values description in the case of an active matrix display.

Figure 15-129. Timing Values Description (Active Matrix Display)



The following bit fields define the timing generation of ac-bias (output enable in active matrix mode):

- Invert output enable (DSS.DISPC_POL_FREQ[15] IEO bit)
- ac-bias pin frequency (DSS.DISPC_POL_FREQ[7:0] ACB bit field)
- ac-bias pin transitions per interrupt (DSS.DISPC_POL_FREQ[11:8] ACBI bit field)
- ac-bias gated (DSS.DISPC_CONFIG[8] ACBIASGATED)

The following bit fields define the timing generation of the pixel clock:

- Pixel clock divisor (DSS.DISPC_DIVISOR[7:0] PCD bit field)
- Invert pixel clock (DSS.DISPC_POL_FREQ[14] IPC bit)
- Pixel clock gated (DSS.DISPC_CONFIG[5] PIXELCLOCKGATED bit)

The 8-bit pixel clock divider (the DSS.DISPC_DIVISOR[7:0] PCD bit field) selects the pixel clock frequency. This bit field generates a range of pixel clock frequencies from LC/1 to LC/255, where LC is the logic clock from the divided functional clock of the display controller by the DSS.DISPC_DIVISOR[23:16] LCD bit field.

The pixel clock is defined by the following equation:

Pixel Clock = (FunctionalClock/LCD[7:0])/PCD[7:0]

Table 15-57 through Table 15-60 show the pixel clock frequency limitations depending the panel type (active or passive matrix) and the mode (color or monochrome).

Table 15-57. Pixel Clock Frequency Limitations - RGB16 and YUV422 Active Matrix Display

Min PCD Values			Horizontal Resampling				
			Off	Up	1:1 - 1:2	1:2 - 1:3	1:3 - 1:4
Vertical Resampling	Off		2 (1)[(1)]	2 [(1)]	2	3	4
	Up		2 [(1)]	2 (1)[(1)]	2	3	4
	1:1 - 1:2	3-tap	2	2	4	6	8
		5-tap	PCDmin	PCDmin	PCDmin	PCDmin	PCDmin
	1:2 - 1:4		PCDmin	PCDmin	PCDmin	PCDmin	PCDmin

(1) The PCD value can be 1 in case all the data and synchronization signals are asserted and deasserted on the rising edge of the pixel clock.

Table 15-58. Pixel Clock Frequency Limitations - RGB16 and YUV422 Passive Matrix Display - Mono4

Min PCD Values			Horizontal Resampling				
			Off	Up	1:1 - 1:2	1:2 - 1:3	1:3 - 1:4
Vertical Resampling	Off		4	4	8	12	16
	Up		4	4	8	12	16
	1:1 - 1:2	3-tap	8	8	16	24	32
		5-tap	4xPCDmin	4xPCDmin	4xPCDmin	4xPCDmin	4xPCDmin
	1:2 - 1:4		4xPCDmin	4xPCDmin	4xPCDmin	4xPCDmin	4xPCDmin

Table 15-59. Pixel Clock Frequency Limitations - RGB16 and YUV422 Passive Matrix Display - Mono8

Min PCD Values			Horizontal Resampling				
			Off	Up	1:1 - 1:2	1:2 - 1:3	1:3 - 1:4
Vertical Resampling	Off		8	8	16	24	32
	Up		8	8	16	24	32
	1:1 - 1:2	3-tap	16	16	32	48	64
		5-tap	8xPCDmin	8xPCDmin	4xPCDmin	8xPCDmin	8xPCDmin
	1:2 - 1:4		8xPCDmin	8xPCDmin	4xPCDmin	8xPCDmin	8xPCDmin

Table 15-60. Pixel Clock Frequency Limitations - RGB16 and YUV422 Passive Matrix Display - Color

Min PCD Values			Horizontal Resampling				
			Off	Up	1:1 - 1:2	1:2 - 1:3	1:3 - 1:4
Vertical Resampling	Off		3	3	6	9	12
	Up		3	3	6	9	12
	1:1 - 1:2	3-tap	6	6	12	18	24
		5-tap	3xPCDmin	3xPCDmin	3xPCDmin	3xPCDmin	3xPCDmin
	1:2 - 1:4		3xPCDmin	3xPCDmin	3xPCDmin	3xPCDmin	3xPCDmin

Note: In case of RGB24 format, Figure 15-130 is still valid, except the PCDmin values which must be multiplied by two.

The PCDmin for vertical downsampling only is defined by the following equations:

Figure 15-130. PCDmin Formulas (V Down-Sampling Only)

$$h_ratio = \frac{DISPC_PPL_SIZE_LCD[10:0]PLL}{DISPC_VIDn_SIZE[10:0]VIDZSIZEX}$$

$$v_ratio = \frac{DISPC_VIDn_PICTURE_SIZE[10:0]VIDORGSIZEX}{DISPC_VIDn_SIZE[10:0]VIDSIZEX}$$

$$PCD\min = \frac{v_ratio}{2 \times h_ratio} \quad 1 < v_ratio \le 2$$

$$PCD\min = \max\left(\frac{v_ratio}{2 \times h_ratio}, \frac{v_ratio - 2}{2 \times (h_ratio - 1)}\right) \quad 2 < v_ratio \le 4$$

dss-E103

The PCDmin for horizontal downsampling only is defined by the following formula:

While downsampling by n, *PCDmin = n*

For H+V downsampling, the formula is the following:

PCDmin = max(PCDmin H only, PCDmin V only) as defined above

The refresh rate depends on the following parameters:

- Horizontal front porch (the DSS.DISPC_TIMING_H[19:8] HFP bit field)
- Horizontal back porch (the DSS.DISPC_TIMING_H[31:20] HBP bit field)
- Horizontal synchronization pulse width (the DSS.DISPC_TIMING_H[7:0] HSW bit field)
- Vertical front porch (the DSS.DISPC_TIMING_V[19:8] VFP bit field)
- Vertical back porch (the DSS.DISPC_TIMING_V[31:20] VBP bit field)
- Vertical synchronization pulse width (the DSS.DISPC_TIMING_V[7:0] VSW bit field)
- Number of lines per panel (the DSS.DISPC_SIZE[26:16] LPP bit field)
- Number of pixels per line (the DSS.DISPC_SIZE[10:0] PPL bit field)
- 4- or 8-bit interface for the passive matrix monochrome panel (the DSS.DISPC_CONTROL[4] M8B bit)

The following bit fields define the behavior of the internal blocks:

- Spatial/temporal dithering logic enabled (DSS.DISPC_CONTROL[7] SPATIALTEMPORALDITHERENABLE bit)
- Spatial/temporal dithering logic number of frames (DSS.DISPC_CONTROL[31:30] SPATIALTEMPORALDITHERFRAMES bit field). The default value of this bit field at reset time is 0x0, which is 1 frame only (spatial processing without temporal dithering). The possible values are 0x0 (one frame), 0x1 (two frames), and 0x2 (four frames). The number of frames is initialized before enabling the spatial/temporal dithering unit. The software must not change this bit field value while the spatial/temporal unit is enabled.

The following bit field defines the clock gating strategy:

- In active matrix mode, the pixel clock is always gated or only when valid data are present (the DSS.DISPC_CONFIG[0] PIXELGATED bit).

15.5.3.5.3 LCD Overlay

The following bit fields define the overlay attributes of the LCD output:

- Transparency color key (the DSS.DISPC_TRANS_COLOR0i register (i = 0))
- Transparency color key enable (the DSS.DISPC_CONFIG[10] TCKLCDENABLE bit)
- Transparency color key selection between the destination graphics transparency color key and the source video transparency color key (the DSS.DISPC_CONFIG[11] TCKLCDSELECTION bit)
- The default solid background color is defined in the DSS.DISPC_DEFAULT_COLOR_m[23:0] DEFAULTCOLOR bit field (i=0).
- Alpha blender Enable (DSS.DISPC_CONFIG[18] LCDALPHABLENDERENABLE)
- Global alpha blending values (DSS.DISPC_GLOBAL_ALPHA[23:16] VID2GLOBALALPHA and DSS.DISPC_GLOBAL_ALPHA[7:0] GFXGLOBALALPHA). The value 0xFF corresponds to 100% opaque and 0 to 100% transparent

Note: The destination graphics transparency color key is available only to the overlay with which the graphics pipeline is connected. The software must set the correct configuration of the LCD and digital overlays.

Note: When the alpha blender is enabled, the destination transparency color key is not available and the source transparency color key applies to the graphics pixels and not the video pixels.

When all of these fields are set to the appropriate values, set the DSS.DISPC_CONTROL[5] GOLCD bit to indicate that all shadow registers of the pipelines connected to the LCD output are latched by the hardware (only if the DSS.DISPC_CONTROL[0] LCDENABLE bit is already set to 1). If the LCD output is disabled, the new values will be updated when the DSS.DISPC_CONTROL[0] LCDENABLE bit will be set to 1.

15.5.3.5.4 LCD TDM

The following fields define the multiple cycle output configuration:

- First cycle (the DSS.DISPC_DATA_CYCLEk (k=0) register)
- Second cycle (the DSS.DISPC_DATA_CYCLEk (k=1) register)
- Third cycle (the DSS.DISPC_DATA_CYCLEk (k=2) register)
- Enable (the DSS.DISPC_CONTROL[20] TDMENABLE bit)
- Parallel mode (the DSS.DISPC_CONTROL[22:21] TDMPARALLEMODE field)
- Cycle format (the DSS.DISPC_CONTROL[24:23] TDMCYCLEFORMAT field)
- Unused bits (the DSS.DISPC_CONTROL[26:25] TDMUNUSEDBITS field)

When all of these bit fields are set to the appropriate values, set the DSS.DISPC_CONTROL[5] GOLCD bit to indicate that all shadow registers of the pipelines connected to the LCD output are latched by the hardware (only if the DSS.DISPC_CONTROL[0] LCDENABLE bit is already set to 1). If the LCD output is disabled, the new values will be updated when the DSS.DISPC_CONTROL[0] LCDENABLE bit will be set to 1.

15.5.3.5.5 LCD Spatial/Temporal Dithering

The following bit fields define the LCD spatial/temporal dithering configuration:

- Number of frames (the DSS.DISPC_CONTROL[31:30] SPATIALTEMPORALDITHERINGFRAMES bit field) with:
  - 0x0 Spatial only (default value)
  - 0x1 Spatial + Temporal over two frames

  - 0x2 Spatial + Temporal over four frames
  - 0x3 Reserved
- Enable (the DSS.DISPC_CONTROL[7] SPATIALTEMPORALDITHERENABLE bit)
  - 0x0 Disabled (default value)
  - 0x1 Enabled

When all of these bit fields are set to the appropriate values, set the DSS.DISPC_CONTROL[5] GOLCD bit to indicate that all shadow registers of the pipelines connected to the LCD output are latched by the hardware (only if the DSS.DISPC_CONTROL[0] LCDENABLE bit is already set to 1). If the LCD output is disabled, the new values will be updated when the DSS.DISPC_CONTROL[0] LCDENABLE bit will be set to 1.

15.5.3.5.6 LCD Color Phase Rotation

The following bit fields define the color phase rotation configuration:

- Enable (the DSS.DISPC_CONFIG[15] CPR bit)
  - 0x0 Disabled (default value)
  - 0x1 Enabled
- Red 10-bit signed coefficients used by the color phase rotation matrix (the DSS.DISPC_CPR_COEF_R register)
- Green 10-bit signed coefficients used by the color phase rotation matrix (the DSS.DISPC_CPR_COEF_G register)
- Blue 10-bit signed coefficients used by the color phase rotation matrix (the DSS.DISPC_CPR_COEF_B register)

The programmable color phase rotation block for the LCD output has nine 10-bit coefficients defined in the DSS.DISPC_CPR_COEF_R, DSS.DISPC_CPR_COEF_G, and DSS.DISPC_CPR_COEF_B, as described in Figure 15-131 through Figure 15-134.

Figure 15-131. Color Phase Rotation Matrix

$$\begin{bmatrix} Rout \\ Gout \\ Bout \end{bmatrix} = \frac{1}{256} * \left(\begin{bmatrix} RR & RG & RB \\ GR & GG & GB \\ BR & BG & BB \end{bmatrix} * \begin{bmatrix} Rin \\ Gin \\ Bin \end{bmatrix} \right)$$

dss-E105

Figure 15-132. Color Phase Rotation Matrix (R Component Only)

$$Riout = \frac{1}{256} * (RR * Rin + RG * Gin + RB * Bin)$$

dss-E106

Figure 15-133. Color Phase Rotation Matrix (G Component Only)

$$Gout = \frac{1}{256} * (GR * Rin + GG * Gin + GB * Bin)$$

dss-E107

Figure 15-134. Color Phase Rotation Matrix (B Component Only)

$$Bout = \frac{1}{256} * (BR * Rin + BG * Gin + BB * Bin)$$

dss-E104

When all of these bit fields are set to the appropriate values, set the DSS.DISPC_CONTROL[5] GOLCD bit to indicate that all shadow registers of the pipelines connected to the LCD output are latched by the hardware (only if the DSS.DISPC_CONTROL[0] LCDENABLE bit is already set to 1). If the LCD output is disabled, the new values will be updated when the DSS.DISPC_CONTROL[0] LCDENABLE bit will be set to 1.

15.5.3.5.6.1 Color Phase Rotation - Diagonal Matrix

The Color Phase Rotation feature is useful when using an LCD backlight that is not white. By using a correct configuration of the R, G and B coefficients of CPR, the color bias of the screen can be corrected. The following paragraphs give an example of configuration of the CPR feature.

The easiest example of CPR configuration is a diagonal matrix. This way, the output colors depends on one input color only. Figure 15-135 gives the example of a diagonal matrix and the corresponding equation of the output components.

Figure 15-135. Diagonal Matrix Configuration

$$\begin{bmatrix} Rout \\ Gout \\ Bout \end{bmatrix} = \frac{1}{256} * \left(\begin{bmatrix} RR & 0 & 0 \\ 0 & GG & 0 \\ 0 & 0 & BB \end{bmatrix} * \begin{bmatrix} Rin \\ Gin \\ Bin \end{bmatrix} \right)$$

$$\rightarrow \quad Rout = \frac{1}{256} * \left(RR * Rin \right)$$

$$\rightarrow \quad Gout = \frac{1}{256} * \left(GG * Gin \right)$$

$$\rightarrow \quad Bout = \frac{1}{256} * \left(BB * Bin \right)$$

dss-200

According to these 3 new equations, each output component only depends on the corresponding input color. The coefficients can easily be used to reduce the impact of a non-white backlight.

Let's take the example of a "blue" backlight. In this case, users have the feeling that a blue film has been added on the screen, and then each color seems to be "too much blue". The goal is then to reduce the "Blue" component and to keep the "Red" and "Green" ones unchanged. The following matrix can be used for a reduction by a half of the blue component. Figure 15-136 gives the corresponding matrix and equations for each component.

Figure 15-136. Example - Diagonal Matrix Configuration

$$\begin{bmatrix} Rout \\ Gout \\ Bout \end{bmatrix} = \frac{1}{256} * \left(\begin{bmatrix} 256 & 0 & 0 \\ 0 & 256 & 0 \\ 0 & 0 & 128 \end{bmatrix} * \begin{bmatrix} Rin \\ Gin \\ Bin \end{bmatrix} \right)$$

$$\rightarrow \quad Rout = \frac{1}{256} * \left(256 * Rin \right) \Rightarrow Rout = Rin$$

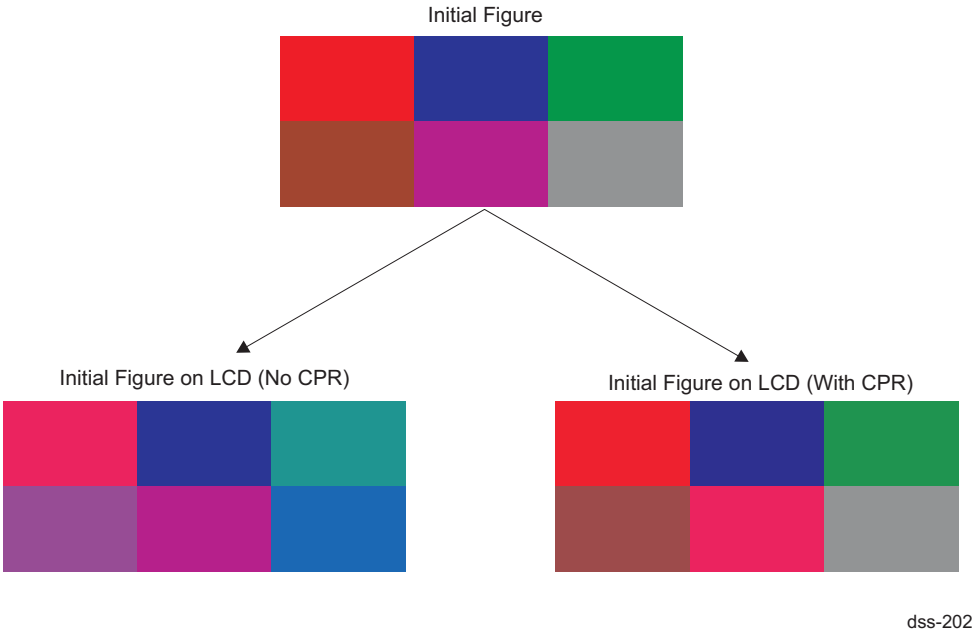
$$\rightarrow \quad Gout = \frac{1}{256} * \left(256 * Gin \right) \Rightarrow Gout = Gin$$

$$\rightarrow \quad Bout = \frac{1}{256} * \left(128 * Bin \right) \Rightarrow Bout = 0.5 * Bin$$

dss-201

Figure 15-137 shows the result of an image on a "blue" backlight screen with and without CPR.

Figure 15-137. Image With and Without CPR (Diagonal Matrix)



A drawback of this diagonal matric is that the color reduction is linear. The contrast is then different from the initial image. It is then necessary to use the 6 other coefficients of the CPR matrix to better correct a non-white backlight. The goal is to find the correct coefficients that remove the color offset added by the non-white backlight.

15.5.3.5.6.2 Color Phase Rotation - Standard Matrix

In the following example, the LCD backlight adds an offset of 128 (B_offset) to the Blue component. Figure 15-138 shows an example of matrix that reduces the offset of the screen, the corresponding equations and the resulting output colors.

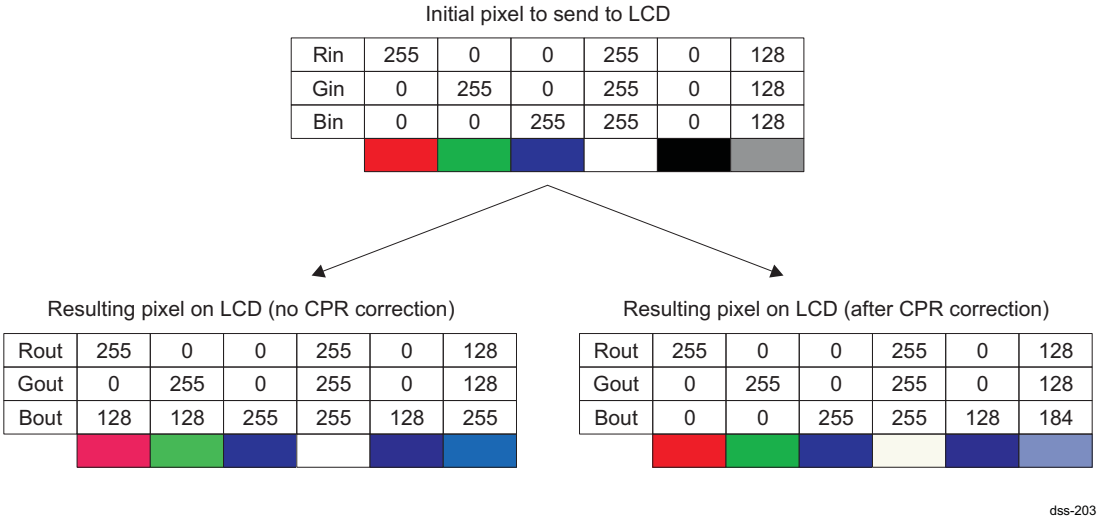
Figure 15-138. Example - Image With and Without CPR (Standard Matrix)

$$\begin{bmatrix} Rout \\ Gout \\ Bout \end{bmatrix} = \frac{1}{256} * \left(\begin{bmatrix} 256 & 0 & 0 \\ 0 & 256 & 0 \\ -129 & -129 & 370 \end{bmatrix} * \begin{bmatrix} Rin \\ Gin \\ Bin \end{bmatrix} \right)$$

$$\rightarrow Rout = \frac{1}{256} * \left(256 * Rin \right)$$

$$\rightarrow Gout = \frac{1}{256} * \left(256 * Gin \right)$$

$$\rightarrow Bout = \frac{1}{256} * \left(-129 * Rin + -129 * Gin + 370 * Bin \right) + B_offset$$



This CPR matrix gives inputs and outputs very close. However, black can not be corrected because of it's zero-components. No matter which coefficients are used in the matrix, the result will always be equal to the offset added by the LCD backlight.

15.5.3.6 TV Set-Specific Control Registers

The following registers define the digital output configuration:

- DSS.DISPC_CONTROL
- DSS.DISPC_CONFIG
- DSS.DISPC_DEFAULT_COLOR_m (m=1)
- DSS.DISPC_TRANS_COLOR_m (m=1)
- DSS.DISPC_SIZE_DIG

The digital output is enabled/disabled by setting/resetting the DSS.DISPC_CONTROL[1] DIGITALENABLE bit. A valid configuration must be set before the digital output can be enabled.

Perform the initialization sequence as follows:

1. Initialize the video encoder and the display controller configuration registers.
2. Set the DSS.DISPC_CONTROL[6] GODIGITAL bit and the DSS.DISPC_CONTROL[1] DIGITALENABLE bit to 1.
3. Wait for the first VSYNC pulse signal.
4. Clear the SYNCLOSTDIGITAL interrupt by setting the DSS.DISPC_IRQSTATUS[15] SYNCLOSTDIGITAL bit to 1.
5. Enable the SYNCLOSTDIGITAL interrupt by setting the DSS.DISPC_IRQENABLE[15]

SYNCLOSTDIGITAL bit to 1.

15.5.3.6.1 Digital Timings

The following bit fields define the timing information:

- Data hold time (the DSS.DISPC_CONTROL[19:17] HT bit field)
- Logic clock divisor (the DSS.DISPC_DIVISOR[23:16] LCD bit field)

The 8-bit pixel clock divider (DSS.DISPC_DIVISOR[23:16]) bit field is used to select the logic clock frequency. The LCD generates a range of pixel clock frequencies from FCK/1 to FCK/255, where FCK is the input functional clock of the display controller.

15.5.3.6.2 Digital Frame/Field Size

The following bit fields define the field size (frame if progressive mode):

- Number of lines per panel (the DSS.DISPC_SIZE_DIG[26:16] LPP bit field)
- Number of pixels per line (the DSS.DISPC_SIZE_DIG[10:0] PPL bit field)

15.5.3.6.3 Digital Overlay

The following bit fields define the overlay attributes of the digital output:

- Transparency color key (the DSS.DISPC_TRANS_COLOR_m register (m=1))
- Transparency color key enable (the DSS.DISPC_CONFIG[12] TCKDIGENABLE bit)
- Transparency color key selection between the destination graphics transparency color key and the source video transparency color key (the DSS.DISPC_CONFIG[13] TCKDIGSELECTION bit)
- The default solid background color is defined in the DSS.DISPC_DEFAULT_COLOR_m[23:0] DEFAULTCOLOR bit field (i=1).
- Alpha blender Enable (DSS.DISPC_CONFIG[19] TVALPHABLENDERENABLE)
- Global alpha blending values (DSS.DISPC_GLOBAL_ALPHA[23:16] VID2GLOBALALPHA and DSS.DISPC_GLOBAL_ALPHA[7:0] GFXGLOBALALPHA). The value 0xFF corresponds to 100% opaque and 0 to 100% transparent

Note: The destination graphics transparency color key is available only to the overlay with which the graphics pipeline is connected. The software must set the correct configuration of the LCD and digital overlays.

Note: When the alpha blender is enabled, the destination transparency color key is not available and the source transparency color key applies to the graphics pixels and not the video pixels.

When this bit field is set to the appropriate values, set the DSS.DISPC_CONTROL[6] GODIGITAL bit to indicate that all shadow registers of the pipelines connected to the digital output are latched by the hardware (only if the DSS.DISPC_CONTROL[1] DIGITALENABLE bit is already set to 1). If the digital output is disabled, the new values will be updated when the DSS.DISPC_CONTROL[1] DIGITALENABLE bit will be set to 1.

15.5.4 DSI Protocol Engine Basic Programming Model

This section describes the programming model of the DSI protocol engine.

15.5.4.1 Software Reset

The DSI protocol engine can be reset by software. This reset can be done for debug purposes or after a protocol error and has the same effect as the hardware reset. The DSI protocol engine can be reset by setting the DSS.DSI_SYSCONFIG[1] SOFT_RESET bit to 1. The software can monitor the DSS.DSI_SYSSTATUS[0] RESET_DONE status bit to wait for the completion of the reset procedure. If after 5 reads, the DSS.DSI_SYSSTATUS[0] RESET_DONE status bit still returns 0, it can be assumed that an error occurred during the reset stage.

Note: This software reset is optional as an hardware reset is always performed on the DSI protocol engine at device reset.

15.5.4.2 Power Management

The power management behavior of the DSI protocol engine is controlled by the DSS.DSI_SYSCONFIG register. This register completely controls the way the module interferes with the PRCM module. The DSS.DSI_SYSCONFIG[0] AUTO_IDLE bit should be set to 1 (default value) to enable automatic clock gating in the module.

15.5.4.3 Interrupts

There is a single interrupt request: DSI_IRQ. This interrupt line is merged with another interrupt line from the DISPC_IRQ in a single interrupt request DSS_IRQ. The DSI_IRQ events are generated only for the enabled VC(s). Two registers are used to enable and monitor the DSI interrupt events:

- DSS.DSI_IRQENABLE register: This register indicates the enabled/disabled event for the VCs. Each event for the VC is configured in the DSS.DSI_VCn_IRQENABLE register dedicated to the VC number. In addition, it includes one bit for the enable of error reporting for the complex I/O: Error signaling from the complex I/O: The interrupt is triggered when any error is received from the complex I/O (ErrSyncEsc[4:0], ErrEsc[4:0] (edge trigger interrupt), ErrControl[4:0] and ErrContentionLP0[4:0], ErrContentionLP1[4:0] from the complex I/O).
- DSS.DSI_IRQSTATUS register : The register DSI_IRQSTATUS flags which VC(s) is/have generated an interrupt. Based on the VC number, the register DSI_VCn_IRQSTATUS indicates the event generating the interrupt. In addition, it includes one bit for the status of error reporting for the complex I/O.

15.5.4.4 Global Register Controls

Prior to receive data from the DSI complex I/O, the DSI_PHY_SCP registers in the DSI complex I/O must be configured. Refer to Section 15.5.6 for more details. Table 15-61 details the register access width limitations for all the DSI modules.

Table 15-61. Register Access Width Limitations

Register Name	Register Access Width
All DSI complex I/O register (DSI_PHY_SCP)	32-bit only
All DSI PLL control module registers	32-bit only
DSI_VCn_LONG_PACKET_HEADER	32-bit only
DSI_VCn_SHORT_PACKET_HEADER	32-bit only
DSI_VCn_LONG_PACKET_PAYLOAD	16-bit, 32-bit
All others DSI protocol engine registers	8-bit, 16-bit and 32-bit

CAUTION

In case of different access width detailed in Table 15-61, an OCP error is generated in response to the write using SResp=ERR.

The DSI protocol engine is globally controlled by the DSS.DSI_CTRL register. The interface to the complex I/O is enabled by setting the DSS.DSI_CTRL[0] IF_EN bit. When the interface is disabled, it is possible to provide data to the TX FIFO and read pending data in the RX FIFO. When the DSS.DSI_CTRL[0] IF_EN bit is set to 1, the pending packets should be sent to the DSI complex I/O, the data transfer from the video port should be ignored only when received the next Vertical Sync Event which is received but not send to the DSI complex I/O.

When the DSS.DSI_CTRL[0] IF_EN bit is reset by software, the hardware should finish the transfer of the pending data in the TX FIFO and wait for response if BTA has been sent (Protocol engine is receive mode), then the hardware resets the DSS.DSI_CTRL[0] IF_EN bit. When using the video mode, the VC associated with the video port should be enabled prior to enable the interface according to the following sequence:

- DSS.DSI_CTRL[0] IF_EN bit is equal to 0
- Enable the VC associated with video mode by setting the DSS.DSI_VCn_CTRL[0] VC_EN
- Set the DSS.DSI_CTRL[0] IF_EN bit to 1

15.5.4.5 Virtual Channels

There is one set of registers for each VC. The attributes of the VC define the following characteristics:

- Transfer mode (DSS.DSI_VCn_CTRL[4] MODE bit):
  - Video mode
  - Command mode
- Data type
- Source (DSS.DSI_VCn_CTRL[1] SOURCE bit)
  - Video port
  - L4 interconnect port
- HS or LP forward transmission
- Automatic bus turn-around generation
  - Short packets (DSS.DSI_VCn_CTRL[2] BTA_SHORT_EN bit)
  - Long packets (DSS.DSI_VCn_CTRL[3] BTA_LONG_EN bit)
- DMA request configurations for RX and TX
  - DMA request number (DSS.DSI_VCn_CTRL[29:27] DMA_RX_REQ_NB bit field for RX FIFO and DSS.DSI_VCn_CTRL[23:21] DMA_TX_REQ_NB bit field for TX FIFO)
  - DMA threshold (DSS.DSI_VCn_CTRL[26:24] DMA_RX_THRESHOLD bit field for RX FIFO and DSS.DSI_VCn_CTRL[19:17] DMA_TX_THRESHOLD bit field for TX FIFO)
- Mode speed (DSS.DSI_VCn_CTRL[9] MODE_SPEED bit)
- ECC transmission (DSS.DSI_VCn_CTRL[8] ECC_TX_EN bit)
- CS transmission (DSS.DSI_VCn_CTRL[7] CS_TX_EN bit)

The VC ID not calculated by the DSI module but provided while writing into the registers DSI_VCn_SHORT_PACKET_HEADER and DSI_VCn_LONG_PACKET_HEADER.

15.5.4.6 Packets

The DSS.DSI_VCn_SHORT_PACKET_HEADER register is used to send only short packets (ECC can be calculated by hardware or by software user for debug purpose). The register is not used for video mode data since the short packets are generated by the hardware using the following information:

- synchronization events received on the video port (assertion/deassertion of the HSYNC and VSYNC input signals)

- DSS.DSI_CTRL[18] VP_HSYNC_END
- DSS.DSI_CTRL[17] VP_HSYNC_START
- DSS.DSI_CTRL[16] VP_VSYNC_END
- DSS.DSI_CTRL[15] VP_VSYNC_START
- DSS.DSI_CTRL[10] VP_HSYNC_POL
- DSS.DSI_CTRL[11] VP_VSYNC_POL
- DSS.DSI_VCn_CTRL[1] SOURCE

The DSS.DSI_VCn_LONG_PACKET_HEADER register is used to provide header for long packets (ECC is always calculated by hardware). The register is used for video mode and command mode. If the video mode is enabled for the VC, it is not possible to transfer concurrently (interleaved in a frame) data using long packets received on the video port and on the L4 interconnect port since the DSS.DSI_VCn_LONG_PACKET_HEADER register is used by the video mode. The register can be unprogrammed by users to send long packets received on the L4 interconnect port only when software users know that there is no expected data on the video port. The software should correctly program the register to send sequentially long packets in video and command modes.

The DSS.DSI_VCn_LONG_PACKET_PAYLOAD register is used to provide payload data for long packets (Check-sum is calculated by hardware when DSS.DSI_VCn_CTRL[7] CS_TX_EN is set to 1 otherwise the value 0x00 is used). The register is not used in video mode since payload data are provided by the video port. The software should ensure that the following sequence for write accesses to the header and payload registers (DSS.DSI_VCn_LONG_PACKET_HEADER and DSS.DSI_VCn_LONG_PACKET_PAYLOAD, respectively) is followed:

- A long packet header value with WC=0 written in DSS.DSI_VCn_LONG_PACKET_HEADER register can be followed by any access.
- A long packet header value with WC>0 written in DSS.DSI_VCn_LONG_PACKET_HEADER register should be followed by one or more writes to the DSS.DSI_VCn_LONG_PACKET_PAYLOAD register defined by the WC value before writing again to the same DSS.DSI_VCn_LONG_PACKET_HEADER register.

CAUTION

If this sequence is not followed, no error is generated. The access to other DSI registers during this sequence is allowed.

15.5.4.7 DSI Complex I/O

Prior to send/receive any data from the complex I/O, the DSI complex I/O timings should be set according to the display module timings. The DSI complex I/O pads must also be configured first. See Section 15.5.6, *DSI Complex I/O Basic Programming Model*.

15.5.4.8 Video Mode

The DSS.DSI_VM_TIMING1, DSS.DSI_VM_TIMING2, DSS.DSI_VM_TIMING3, DSS.DSI_VM_TIMING4, DSS.DSI_VM_TIMING5, DSS.DSI_VM_TIMING6, and DSS.DSI_VM_TIMING7 registers define the timings of the video mode.

The DSS.DSI_CTRL[20] BLANKING_MODE bit defines if the long blanking packets or LPS state are used during the blanking periods (except HFP, HBP, HSA defined by other bits) when there is no pending data in TX FIFO ready to be sent. The software should ensure that there is no data in the TX FIFO, no BTA, no RESET trigger sent, and the DSS.DSI_VCn_CTRL[9] MODE_SPEED bit is set to 1 (High-Speed mode) to keep the video mode transfer is HS mode during blanking periods (except for the last blanking period since it is required to go LPS at least once per frame).

The DSS.DSI_CTRL[21] HFP_BLANKING_MODE, DSS.DSI_CTRL[22] HBP_BLANKING_MODE and DSS.DSI_CTRL[23] HSA_BLANKING_MODE define if these blanking can send packets from the TX FIFO or should be kept in HS mode using only long blanking packets.

To ensure that the writes to the register DSS.DSI_VCn_LONG_PACKET_HEADER are correctly handled as header information for video mode long packets, the following registers should be programmed:

- DSS.DSI_VCn_CTRL[0] VC_EN bit set to 1
- DSS.DSI_VCn_CTRL[4] MODE bit set to 1
- DSS.DSI_VCn_LONG_PACKET_HEADER register access
- DSS.DSI_VCn_CTRL[0] VC_EN bit set to 1

Note: The DSS.DSI_VCn_CTRL[1] SOURCE and DSS.DSI_VCn_CTRL[9] MODE_SPEED bits are ignored by hardware when the video mode is selected (DSS.DSI_VCn_CTRL[4] MODE bit set to 1)

The interrupt events SYNC_LOST_IRQ and RESYNCHRONIZATION_IRQ indicates if the DSI protocol engine has not been able to resynchronize the video port timing to its own timing base or if it has been done. The RESYNCHRONIZATION_IRQ indicates software users that the video port works but the configuration of the timings for the display controller (DISPC) and for DSI Protocol engine may need to be modified to avoid the resynchronization to occur. The SYNC_LOST_IRQ and RESYNCHRONIZATION_IRQ events can be respectively monitored in DSS.DSI_IRQSTATUS[18] SYNC_LOST_IRQ and DSS.DSI_IRQSTATUS[5] RESYNCHRONIZATION_IRQ status bits.

The DSS.DSI_VM_TIMING2[27:24] WINDOW_SYNC bit field defines the synchronization period. The recommended value is 0x4 based on the implementation of the resynchronization scheme.

15.5.4.9 Video Port Data Bus

The DSS.DSI_CTRL[7:6] VP_DATA_BUS_WIDTH bit field is used to determine the width of the data bus on the video port. The supported formats are 16-bit, 18-bit and 24-bits.

15.5.4.10 Command Mode

15.5.4.10.1 Command Mode TX FIFO

The single TX FIFO is used on the L4 interconnect port to receive the data to be sent to the peripheral. The configuration of the FIFO for a specific VC should be done only when the VC is disabled.

Users should not enable the VC if there is still some pending data in the TX FIFO for the corresponding space allocated for the VC from previous active period. When the VC space in the TX FIFO is empty, the VC can be enabled.

For each VC, two dedicated DSS.DSI_VCn_LONG_PACKET_HEADER and DSS.DSI_VCn_LONG_PACKET_PAYLOAD registers are used to provide data for long packets. The register DSS.DSI_VCn_SHORT_PACKET_HEADER is used to provide data for short packets (32-bit long).

For each long packet, the DSS.DSI_VCn_LONG_PACKET_HEADER register should be written first and then the DSS.DSI_VCn_LONG_PACKET_PAYLOAD register. The only exception is when the word count defined in the header is equal to 0. In that case, it is not required to write into the payload register. For consecutive long packets, the header should be written into the DSS.DSI_VCn_LONG_PACKET_HEADER register even if the value remains the same.

The TX FIFO stores all the pending bytes to be sent to the peripheral(s). Multiple receivers can be addressed using the VC capability.

The 32-bit write requests only for each VC to the TX FIFO should be kept in order while sending the data to the DSI_PHY inside the VC requests. The only exception is in the case of last 32-bit write for the last bytes of the payload data since it could be 1, 2, 3, or 4 bytes.

Also in case the last transfer is a 32-bit write but the number of valid bytes is 1, 2, or 3 only (calculated using the header word count and the number of bytes are received for the payload), the hardware should store the 32-bit value into the TX FIFO but the invalid bytes are not sent, and are discarded.

When the word count defined in DSS.DSI_VCn_LONG_PACKET_HEADER register is not a multiple of the request threshold value defined in DSS.DSI_VCn_CTRL[19:17] DMA_TX_THRESHOLD bit field, 32-bit requests and/or bytes should be discarded by the hardware to store in FIFO only the exact number of valid bytes.

The DSI protocol module should be able to determine if the bytes in the TX FIFO correspond to a short or long packet without decoder the DT field. When the bytes are written into the DSS.DSI_VCn_SHORT_PACKET_HEADER, DSS.DSI_VCn_LONG_PACKET_HEADER, and DSS.DSI_VCn_LONG_PACKET_PAYLOAD registers, the hardware should store the information concerning long or short packet. A 1-bit flag should be used for each entry of the TX FIFO.

When the VC is disabled, the remaining bytes in the FIFO should be sent to the DSI link. The start event to send data to the DSI link one of the following events:

- All bytes have been received in the FIFO (header + payload).
- The space of the FIFO allocated for the VC is full.
- The space of the FIFO allocated for the VC is not enough to request more data using DMA request (threshold value bigger than space left in the TX FIFO for the VC).

Note: In case the video mode is active, the blanking period should be large enough to allow the transfer of the packet(s).

Sequential arbitration must be set (TX_FIFO_ARBITRATION[3] DSI_CTRL bit = 0x1) if only one VC is used to send multiple packets during the same blanking period.

When consecutive packets should be sent in HS mode, to ensure that there is no LP transition between them at least one of the following condition should be valid:

- Packets from the same VC
- Short packets or long packets with a payload size multiple of 4 bytes

To flush the FIFO (discard of the data) for some pending bytes, the software should change the allocated size of the chuck FIFO by:

- First disabling the VC resetting DSS.DSI_VCn_CTRL[0] VC_EN bit to 0
- Second change the size of the space of FIFO to 0 by writing the DSS.DSI_TX_FIFO_VC_SIZE VCn_FIFO_SIZE (n is the VC value between 0 and 3)

If there is an on-going packet transfer from the TX FIFO to DSI_PHY, the flush of the FIFO should stop immediately the transfer. Because the FIFO is accessible to users, it must ensure that there are no bytes left in the FIFO before starting the flush operation. But it can be required in dead-lock situation to flush the FIFO even if there are still bytes in the FIFO, in that case, the software should also take care of having a known state of the whole DSI protocol engine module (software reset may be required) To start of new transfer through the TX FIFO.

Users can check that there is no pending request before changing the size of the allocated FIFO for the VC by reading the relevant DSS.DSI_VCn_CTRL[15] VC_BUSY bit or by using the interrupt PACKET_SENT_IRQ: All the packets have been sent by counting the transferred requests to the L4 interconnect port and the number of requests sent to the DSI complex I/O. This interrupt can be monitoring by reading the DSS.DSI_VCn_IRQSTATUS[2] PACKET_SENT_IRQ status bit.

The DSS.DSI_CTRL[3] TX_FIFO_ARBITRATION bit defines if the arbitration scheme is:

- Round-robin between enabled VCs with pending ready requests (pending ready request means that all bytes for the packets are in the FIFO or the space of the FIFO for the VC is full) starting from the VC which has the least VC ID number.
- Sequential: All the pending ready requests for one VC are sent before moving to another VC. The condition of "space of the FIFO is full" should be evaluated after the end of each packet.

If users want to use sequential arbitration for all requests for all channels, a single VC should be used. (the VC ID defined in the header provided to the hardware using either the DSS.DSI_VCn_LONG_PACKET_HEADER or DSS.DSI_VCn_SHORT_PACKET_HEADER register is not used and not modified by the DSI protocol engine).

The register DSS.DSI_TX_FIFO_VC_SIZE defines the allocated number of 33-bit values for each VC in

the TX FIFO and the start address for each VC. The size of the space allocated in the TX FIFO defined by DSS.DSI_TX_FIFO_VC_SIZE VCn_FIFO_SIZE (n corresponds to the VC number n) bit fields should be a multiple of the threshold defined in DSS.DSI_VCn_CTRL[19:17] DMA_TX_THRESHOLD bit field. Only the enabled VCs should be taken into account. To change the size of the space of the memory allocated for a specific VC, the VC should be disabled by setting the DSS.DSI_VCn_CTRL[0] VC_EN bit to 0. The whole FIFO may not be used by all the VCs at a given time since a VC can be disabled to change one or multiple parameters. Software users are responsible for correctly configuring the start address and the size for each VC.

Table 15-62 indicates the corresponding values for the size of the space allocated in the FIFO.

Table 15-62. Virtual Channel TX FIFO Size Values

DSI_TX_FIFO_VC_SIZE.VCn_FIFO_SIZE[n = 0, 3]	Space Size (up to the size of the FIFO)
0	0 x 33 bits
1	32 x 33 bits
2	64 x 33 bits
3	96 x 33 bits
4	128 x 33 bits

The total size of TX FIFO is 128*33 bits. Therefore, the sum of all virtual channel FIFO allocation cannot exceed 128*33 bits.

Table 15-63 indicates the start address of the space in the FIFO.

Table 15-63. Virtual Channel TX FIFO Start Address

DSI_TX_FIFO_VC_SIZE.VCx_FIFO_ADD[x = 0, 2]	Start Address
0	0
1	32
2	64
3	96
4	128

CAUTION

There must be no overlap of different VCs spaces.

When the TX FIFO is full:

- the overflow interrupt (FIFO_TX_OVF_IRQ) is generated. To monitor this interrupt request, users can read the DSS.DSI_VCn_IRQSTATUS[3] FIFO_TX_OVF_IRQ status bit.
- there is no L4 interconnect error generated
- the commands are accepted but the data are not written into the FIFO

To ensure that all writes are correctly stored in the TX FIFO, the FIFO should not be full. The software must read the room in the space allocated for the VC in the TX FIFO by reading the DSS.DSI_TX_FIFO_VC_EMPTINESS register. In case there is no space allocated in the TX FIFO for the VC, the DSS.DSI_TX_FIFO_VC_EMPTINESS register indicates a value of 0 for the VC space emptiness.

When waiting to receive the first VSYNC event on the video port to start the video mode on DSI link no command data from TX FIFO should be sent on the interface. It is required to ensure that when receiving the VSYNC event, there is no on-going command mode transfer that could delay the start of video mode on the DSI link.

15.5.4.10.2 Command Mode RX FIFO

The RX FIFO is used to store the data received from the DSI complex I/O. The data are always packed in the RX FIFO (single or multiple packets receiving during a single or multiple BTA periods).

The read requests access to corresponding VC locations to transfer data for a specific VC. The logic managing the FIFO must be able to extract 32-bit values in-order for a specific VC upon read requests. The byte enable of the read access is ignored. Each read returns one 32-bit value from the RX FIFO. If the application accesses the RX FIFO should extract always 32-bit values. Only in the case of 1, 2, or 3 bytes are remaining in the RX FIFO.

The read requests (single or burst) can be less, equal or greater than the packet size. If the packet size is smaller than the read request, the following packet(s) is also transferred. If the packet size is longer than the read request, only part of the packet is transfer. In that case, the logic should keep the VC information to provide the rest of the data during the next read request(s).

The register DSS.DSI_RX_FIFO_VC_SIZE defines the allocated number of 33-bit values for each VC in the RX FIFO and the start address for each VC. Only the enabled VCs should be taken into account. to change the size of the space of the memory allocated for a specific VC, the VC should be disabled by resetting the DSS.DSI_VCn_CTRL[0] VC_EN bit to 0. The whole FIFO may not be used by the entire VC at a given time since a VC can be disabled to change one or multiple parameters. Software users are responsible for correctly configuring the start address and the size for each VC.

Table 15-64 indicates the corresponding values for the size of the space allocated in the FIFO:

Table 15-64. Virtual Channel RX FIFO Size Values

DSI_RX_FIFO_VC_SIZE.VCx_FIFO_SIZE[x = 0, 3]	Space Size (up to the size of the FIFO)
0	0 x 33 bits
1	32 x 33 bits
2	64 x 33 bits
3	96 x 33 bits
4	128 x 33 bits

The total size of RX FIFO is 128*33 bits. Therefore, the sum of all virtual channel FIFO allocation cannot exceed 128*33 bits.

Table 15-65 indicates the start address of the space in the FIFO.

Table 15-65. Virtual Channel RX FIFO Start Address

DSI_RX_FIFO_VC_SIZE.VCx_FIFO_ADD[x = 0, 2]	Start Address
0	0
1	32
2	64
3	96
4	128

CAUTION

There must be no overlap of different VCs spaces.

While reading the received bytes in the RX FIFO, only the DSS.DSI_VCn_SHORT_PACKET_HEADER register is used since the hardware does not keep track of the header position for long packets and start/end of each packet. The software must extract the information from the bytes read from the RX FIFO. There is no specific hardware to track the received bytes in the RX FIFO. The DSS.DSI_VCn_LONG_PACKET_HEADER and DSS.DSI_VCn_LONG_PACKET_PAYLOAD registers are not used.

The ECC is only used by the first header when receiving multiple packets during the same LP RX transfer from the peripheral since the DSI Protocol engine does not parse the header to identify the length of the packets. In case of multiple packets, the Check-sum does not be enabled since the hardware checks the check-sum considering a single packet. The ECC in the first header is used to correct and check the header. For the following headers in the same LP RX transfer, the hardware does not detect any header and can not check or/and detect errors in the headers of the packets except for the first packet.

When the RX FIFO is empty:

- there is no OCP error generated
- the commands are accepted and the data for the responses are 0s

15.5.4.10.3 Command Mode DMA Requests

The DMA requests (DSI_DMA_REQ) are used to allow automatic transfer by the system DMA or MPU (with less efficiency and through-put capability) from the DSI RX FIFO to the system memory and from the system memory to the DSI TX FIFO. Two independent DMA requests for RX FIFO and TX FIFO for the same VC are supported. The read and write accesses can use burst structure. The DSI protocol engine should access each write request in a burst without any IDLE between. For read OCP request in a burst to RX FIFO, the acceptance is sent immediately and the response is delayed by at least four L4 interface clock (DSS_L4_ICLK) cycles. In case of register reads, the response is returned in the first clock cycle.

The thresholds used for requests for the TX FIFO and RX FIFO are programmable by software. Users program the DSS.DSI_VCn_CTRL[19:17] DMA_TX_THRESHOLD and DSS.DSI_VCn_CTRL[26:24] DMA_RX_THRESHOLD bit fields for TX FIFO and RX FIFO respectively. The hardware asserts the DMA request based on the threshold value. The size of the space allocated in TX FIFO for each VC should be a multiple of DSS.DSI_VCn_CTRL[19:17] DMA_TX_THRESHOLD bit field value.

The only exception is in the case of the RX FIFO when the LP data transfer finishes and the threshold value is not reached. In that case the DMA request should be asserted. So the drain of the FIFO is supported in that configuration to empty the FIFO even if the number of data received is not a multiple of the threshold value.

In case of TX FIFO, if all the bytes defined by the word count field in the DSS.DSI_VCn_LONG_PACKET_HEADER header register have been received, the DMA request is not asserted anymore even during the last transfer less than DMA_TX_THRESHOLD number of bytes have been received because of the word count being not a multiple of the DMA_TX_THRESHOLD value.

In case of RX FIFO, while the DMA request is used to transfer the data from the RX FIFO to the system memory, the system DMA should be programmed to read the exact number of received bytes in the FIFO. If users do not know the size of the received bytes, the direct access of the RX FIFO through the DSS.DSI_VCn_SHORT_PACKET_HEADER register is performed until the DSS.DSI_VCn_CTRL[20] RX_FIFO_NOT_EMPTY bit goes to 0.

The use of each DMA request is programmable by software. The DSS.DSI_VCn_CTRL[23:21] DMA_TX_REQ_NB is dedicated to DMA request numbering for the TX FIFO. The DSS.DSI_VCn_CTRL[29:27] DMA_RX_REQ_NB is dedicated to DMA request numbering for the RX FIFO.

When the DMA request is used to indicate the number of 32-bit values ready in the RX FIFO or BTA has been received from peripheral indicating end of the transfer from peripheral to host for a transfer to the system memory, the DMA request corresponding to the VC ID is generated.

The system DMA transfers the number of 32-bit values defined in the threshold register or the exact number of bytes received from the peripheral (user should know the number of expected received bytes to program correctly the system DMA). When the system DMA transfers a multiple number of threshold value, the DSI protocol engine should send 0s for the data when there is no more received data in the RX FIFO for the VC. Software users are responsible for parsing the data and determine the valid bytes.

Software users can decide to determine the number of data received in the RX FIFO to read the information in the DSS.DSI_RX_FIFO_VC_FULLNESS register. Then the system DMA can be programmed to read the exact number of bytes from the RX FIFO. The BTA interrupt (BTA_IRQ) should be used to know when to read the number of received bytes. To monitor the BTA interrupt, the user should read the DSS.DSI_VCn_IRQSTATUS[5] BTA_IRQ status bit. The DMA request should not be selected until the system DMA is programmed with the correct number of data to read from RX FIFO.

If the RX FIFO space for the VC is expected to be overflow because the number of data to be received is greater than the space allocated for the VC, the previous programming model should be used. In place, the DMA request should be asserted as soon as the threshold is reached or when BTA is received.

When the DMA request is used to indicate the number of 33-bit entries empty in the TX FIFO for a transfer from the system memory, the DMA request corresponding to the VC ID is generated.

Note: To obtain best efficiency of the transfer the size of the request (read or write, single or burst) should be aligned with the threshold value.

Concurrent access using interlaced requests (read/write) to the TX and RX FIFO is supported for the same VC ID or different VC IDs.

15.5.4.11 Ultra-Low Power State

This section describes how to enter/exit to/from ultralow-power state (ULPS).

Note: The DSS.DSI_COMPLEXIO_CFG2 LANEx_ULPS_SIGy bits (x range is 1 to 3 corresponding to lane #1 to lane #3 and y range is 1 to 2) must be read back after writing to verify that the write operations are effective before proceeding to the next step. This is to take latency at low TxClkEsc frequencies into account.

15.5.4.11.1 Entering ULPS

To enter into ULPS for a clock lane, the following sequence is required:

1. Wait for DSS.DSI_COMPLEXIO_CFG2[16] HS_BUSY and DSS.DSI_COMPLEXIO_CFG2[17] LP_BUSY bits to be reset to 0 and ensure that the DSS.DSI_CLK_CTRL[13] DDR_CLK_ALWAYS_ON bit is 0.
2. TxUlpsClk state should change from inactive to active by setting the DSS.DSI_COMPLEXIO_CFG2 LANEx_ULPS_SIG2 (x range is 1 to 3 corresponding to lane #1 to lane #3) bit to 1.

To enter into ULPS for a data lane, the following sequence is required:

1. Wait for all TX_FIFOs for all VCs working in HS are empty, for video mode is not active, and for DSS.DSI_COMPLEXIO_CFG2[16] HS_BUSY bit is reset to 0 (in addition, for data lane #1, DSS.DSI_COMPLEXIO_CFG2[17] LP_BUSY bit is reset to 0)
2. TxRequestEsc state should change from inactive to active by setting the DSS.DSI_COMPLEXIO_CFG2. LANEx_ULPS_SIG2 (x range is 1 to 3 corresponding to lane #1 to lane #3) bit to 1.

Note: When the DSS.DSI_COMPLEXIO_CFG2 LANEx_ULPS_SIG2 and DSS.DSI_COMPLEXIO_CFG2 LANEx_ULPS_SIG1 bits are written to 0, they can be combined into one write. Both bits must be read back to confirm they are effective before proceeding.

15.5.4.11.2 Exiting ULPS

To exit from ULPS for a clock lane, the following sequence is required:

1. Change the state of TxUlpsExit for each lane to ACTIVE by setting the DSS.DSI_COMPLEXIO_CFG2 LANEx_ULPS_SIG1 (x range is 1 to 3 corresponding to lane #1 to lane #3) bit to 1.

2. Wait for the ULPSACTIVENOT_ALL1_IRQ interrupt indicating that all lanes with TxUlpsExit active have acknowledged by asserting UlpsActiveNot. This is performed by monitoring the DSS.DSI_COMPLEXIO_IRQSTATUS[31] ULPSACTIVENOT_ALL1_IRQ status bit.
3. Start the wake-up timer (GPTimer).
4. Wait for the time-out.
5. Change TxUlpsClk signals to INACTIVE state for the clock lane by resetting the DSS.DSI_COMPLEXIO_CFG2 LANEx_ULPS_SIG2 (x range is 1 to 3 corresponding to lane #1 to lane #3) bit to 0.
6. Reset the DSS.DSI_COMPLEXIO_CFG2 LANEx_ULPS_SIG1 (x range is 1 to 3 corresponding to lane #1 to lane #3) bit to 0.

Note: When the DSS.DSI_COMPLEXIO_CFG2 LANEx_ULPS_SIG2 and DSS.DSI_COMPLEXIO_CFG2 LANEx_ULPS_SIG1 bits are both being written to 0, they can be combined into one write. Both bits must be read back to confirm they are effective before proceeding.

To exit from ULPS for a clock lane, in case ComplexIO is in OFF state (the DSI protocol engine sends ComplexIO to OFF state by setting DSS.DSI_COMPLEXIO_CFG1[28:27] PWROFF = 0x0), the sequence is:

1. Change TxUlpsClk signals to INACTIVE state for the clock lane by resetting the DSS.DSI_COMPLEXIO_CFG2 LANEx_ULPS_SIG2 (x range is 1 to 3 corresponding to lane #1 to lane #3) bit to 0.
2. Change the state of TxUlpsExit for clock lane to INACTIVE state by resetting the DSS.DSI_COMPLEXIO_CFG2 LANEx_ULPS_SIG1 (x range is 1 to 3 corresponding to lane #1 to lane #3) bit to 0. This step is necessary only in case a PWROFF command (the command for power control of the complex I/O) is issued while the sequence for exiting is in progress (TxUlpsExit signal is already in ACTIVE state).

Note: When the DSS.DSI_COMPLEXIO_CFG2 LANEx_ULPS_SIG2 and DSS.DSI_COMPLEXIO_CFG2 LANEx_ULPS_SIG1 bits are both being written to 0, they can be combined into one write. Both bits must be read back to confirm they are effective before proceeding.

To exit from ULPS for a data lane, the following sequence is required:

1. Change the state of TxUlpsExit for each lane to ACTIVE by setting the DSS.DSI_COMPLEXIO_CFG2 LANEx_ULPS_SIG1 (x range is 1 to 3 corresponding to lane #1 to lane #3) bit to 1.
2. Wait for the ULPSACTIVENOT_ALL1_IRQ interrupt indicating that all lanes with TxUlpsExit active have acknowledged by asserting UlpsActiveNot. This is performed by monitoring the DSS.DSI_COMPLEXIO_IRQSTATUS[31] ULPSACTIVENOT_ALL1_IRQ status bit.
3. Start the application wake-up timer (GPtimer).
4. Wait for the time-out.
5. Change TxRequestEsc signals to INACTIVE state for the data lane by resetting the DSS.DSI_COMPLEXIO_CFG2 LANEx_ULPS_SIG2 (x range is 1 to 3 corresponding to lane #1 to lane #3) bit to 0.
6. Reset the DSS.DSI_COMPLEXIO_CFG2 LANEx_ULPS_SIG1 (x range is 1 to 3 corresponding to lane #1 to lane #3) bit to 0.

Note: When the DSS.DSI_COMPLEXIO_CFG2 LANEx_ULPS_SIG2 and DSS.DSI_COMPLEXIO_CFG2 LANEx_ULPS_SIG1 bits are both being written to 0, they can be combined into one write. Both bits must be read back to confirm they are effective before proceeding.

To exit from ULPS for a data lane, in case ComplexIO is in OFF state (the DSI protocol engine sends ComplexIO into OFF state by setting DSS.DSI_COMPLEXIO_CFG1[28:27] PWROFF = 0x0), the sequence is:

1. Change TxRequestEsc signals to INACTIVE state by resetting the DSS.DSI_COMPLEXIO_CFG2 LANEx_ULPS_SIG2 (x range is 1 to 3 corresponding to lane #1 to lane #3) bit to 0.
2. Change the state of TxUlpsExit to INACTIVE state by resetting the DSS.DSI_COMPLEXIO_CFG2 LANEx_ULPS_SIG1 (x range is 1 to 3 corresponding to lane #1 to lane #3) bit to 0. This step is necessary only in case a PWROFF command (the command for power control of the complex I/O) is issued while the sequence for exiting is in progress (TxUlpsExit signal is already in ACTIVE state).

Note: When the DSS.DSI_COMPLEXIO_CFG2 LANEx_ULPS_SIG2 and DSS.DSI_COMPLEXIO_CFG2 LANEx_ULPS_SIG1 bits are both being written to 0, they can be combined into one write. Both bits must be read back to confirm they are effective before proceeding.

When the sequence for entering/exiting into/from ULP state is started for specific lanes, users should wait for the completion of the sequence before changing the state of the same or other lanes.

15.5.4.12 DSI Programming Sequence Example

This section describes distinct configurations of the DSI protocol engine to support different type of traffics.

Note: When the VC is used to send video mode data from the video port, the DSS.DSI_VCn_CTRL[9] MODE_SPEED and the DSS.DSI_VCn_CTRL[1] SOURCE bits are ignored.

15.5.4.12.1 Video Mode Transfer

Description: One channel, video mode, no DMA requests, no bus turn-around.

1. Configure the display controller with the timing parameters.
2. Configure the DSS.DSI_VCn_CTRL register as follows:
   - SOURCE bit is ignored by hardware
   - BTA_LONG_EN bit is set to 0: No BTA on long packet
   - BTA_SHORT_EN bit is set to 0: No BTA on short packet
   - MODE bit set to 1: The video mode is selected
   - MODE_SPEED bit is ignored by hardware
3. Configure the DSS.DSI_VM_TIMING1 to DSS.DSI_VM_TIMING7 registers.
4. Set the ForceTxStopMode bit to 1 in the DSS.DSI_TIMING1 register.
5. Enable the channel by setting the DSS.DSI_VCn_CTRL[0] VC_EN bit to 1
6. Enable the module by setting the DSS.DSI_CTRL[0] IF_EN bit to 1.
7. Poll the ForceTxStopMode bit to 0 in the DSS.DSI_TIMING1 register.
8. Enable the LCD video output by setting the DSS.DISPC_CONTROL[0] LCDENABLE bit to 1

CAUTION

The restriction for stopping the video mode is that no frame should be sent by the display controller (DISPC) after disabling video mode in DSI.

15.5.4.12.2 Command Mode Transfer Example 1

CAUTION

In DSI command mode, the display controller must be configured in stall mode by setting the DSS.DISPC_CONTROL[11] STALLMODE bit to 1.

Description: One channel, command mode, no DMA requests, manual bus turn-around

1. Configure the DSS.DSI_VCn_CTRL register as follows:
   - SOURCE bit set to 0: The source is the L4 interconnect port
   - BTA_LONG_EN bit is set to 0: No automatic BTA on long packet
   - BTA_SHORT_EN bit is set to 0: No automatic BTA on short packet
   - MODE bit set to 0: The command mode is selected
2. Enable the packet sent interrupt by setting the DSS.DSI_VCn_IRQENABLE[2] PACKET_SENT_IRQ_EN bit to 1
3. Set the ForceTxStopMode bit to 1 in the DSS.DSI_TIMING1 register.
4. Enable the channel by setting the DSS.DSI_VCn_CTRL[0] VC_EN bit to 1
5. Enable the module by setting the DSS.DSI_CTRL[0] IF_EN bit to 1
6. Poll the ForceTxStopMode bit to 0 in the DSS.DSI_TIMING1 register.
7. For long packet:
   - Write the header value into the DSS.DSI_VCn_LONG_PACKET_HEADER register
   - Write the data into the DSS.DSI_VCn_LONG_PACKET_PAYLOAD register for the full payload. Repeat step until WC is reached (see DSI_VCn_LONG_PACKET_HEADER)

   For the short packet:
   - Send short packets through the L4 interconnect:Write the header value into the DSS.DSI_VCn_SHORT_PACKET_HEADER register. Repeat step for each short packet.
8. Send one or more packets through L4 interconnect:
   - Write the header value into DSS.DSI_VCn_LONG_PACKET_HEADER register
   - Write the data into DSS.DSI_VCn_LONG_PACKET_PAYLOAD register for the full payload.
   - Repeat step 5 for all the long packets
9. Send short packets through L4 interconnect:
   - Write the header value into DSS.DSI_VCn_SHORT_PACKET_HEADER register
   - Repeat step 6 for all the short packets
10. Interrupt routine: Wait for PACKET_SENT_IRQ interrupt generation by polling the DSS.DSI_VCn_IRQSTATUS[2] PACKET_SENT_IRQ status bit and notify the application software when received.
11. The applicative software forces the bus turn-around:
    - Wait until the PACKET_SENT_IRQ has happened as many times as the number of sent packets
    - Set the DSS.DSI_VCn_CTRL[6] BTA_EN bit to 1 to send manually a BTA
    - Wait until the DSS.DSI_VCn_CTRL[6] BTA_EN bit is reset to 0 by hardware
12. Receive the packets from the peripheral
    - Start polling the DSS.DSI_VCn_CTRL[20] RX_FIFO_NOT_EMPTY status bit
    - Whenever the RX_FIFO_NOT_EMPTY bit equals to 1, read one word in the RX FIFO

15.5.4.12.3 Command Mode Transfer Example 2

CAUTION

In DSI command mode, the display controller must be configured in stall mode by setting the DSS.DISPC_CONTROL[11] STALLMODE bit to 1.

Description: One channel, command mode, DMA request, automatic bus turn-around

1. Configure the DSS.DSI_VCn_CTRL register as follows:
   - SOURCE bit set to 0: The source is the L4 interconnect port
   - BTA_LONG_EN bit is set to 1: Automatic BTA on long packet
   - BTA_SHORT_EN bit is set to 1: Automatic BTA on short packet
   - MODE bit set to 0: The command mode is selected
2. Enable the packet sent interrupt by setting the DSS.DSI_VCn_IRQENABLE[2] PACKET_SENT_IRQ_EN bit to 1

3. Set the ForceTxStopMode bit to 1 in DSS.DSI_TIMING1 register.
4. Enable the channel by setting the DSS.DSI_VCn_CTRL[0] VC_EN bit to 1
5. Configure the TX FIFO threshold and DMA requests parameters:
   - Program the DSS.DSI_VCn_CTRL[19:17] DMA_TX_THRESHOLD bit field
   - Program the DSS.DSI_VCn_CTRL[23:21] DMA_TX_REQ_NB bit field
6. Program the system DMA to be ready to send data to the L4 interconnect port
7. Enable the module by setting the DSS.DSI_CTRL[0] IF_EN bit to 1
8. Poll the ForceTxStopMode bit to 0 in DSS.DSI_TIMING1 register.
9. Write the header value into DSS.DSI_VCn_LONG_PACKET_HEADER register
10. Interrupt routine: Wait for PACKET_SENT_IRQ interrupt generation by polling the DSS.DSI_VCn_IRQSTATUS[2] PACKET_SENT_IRQ status bit and notify the application software when received.
11. Receive the packets from the peripheral
    - Start polling the DSS.DSI_VCn_CTRL[20] RX_FIFO_NOT_EMPTY status bit
    - Whenever the RX_FIFO_NOT_EMPTY bit equals to 1, read one 32-bit word in the RX FIFO
12. Repeat the steps 7 for all long packets.

15.5.5 DSI PLL Controller Basic Programming Model

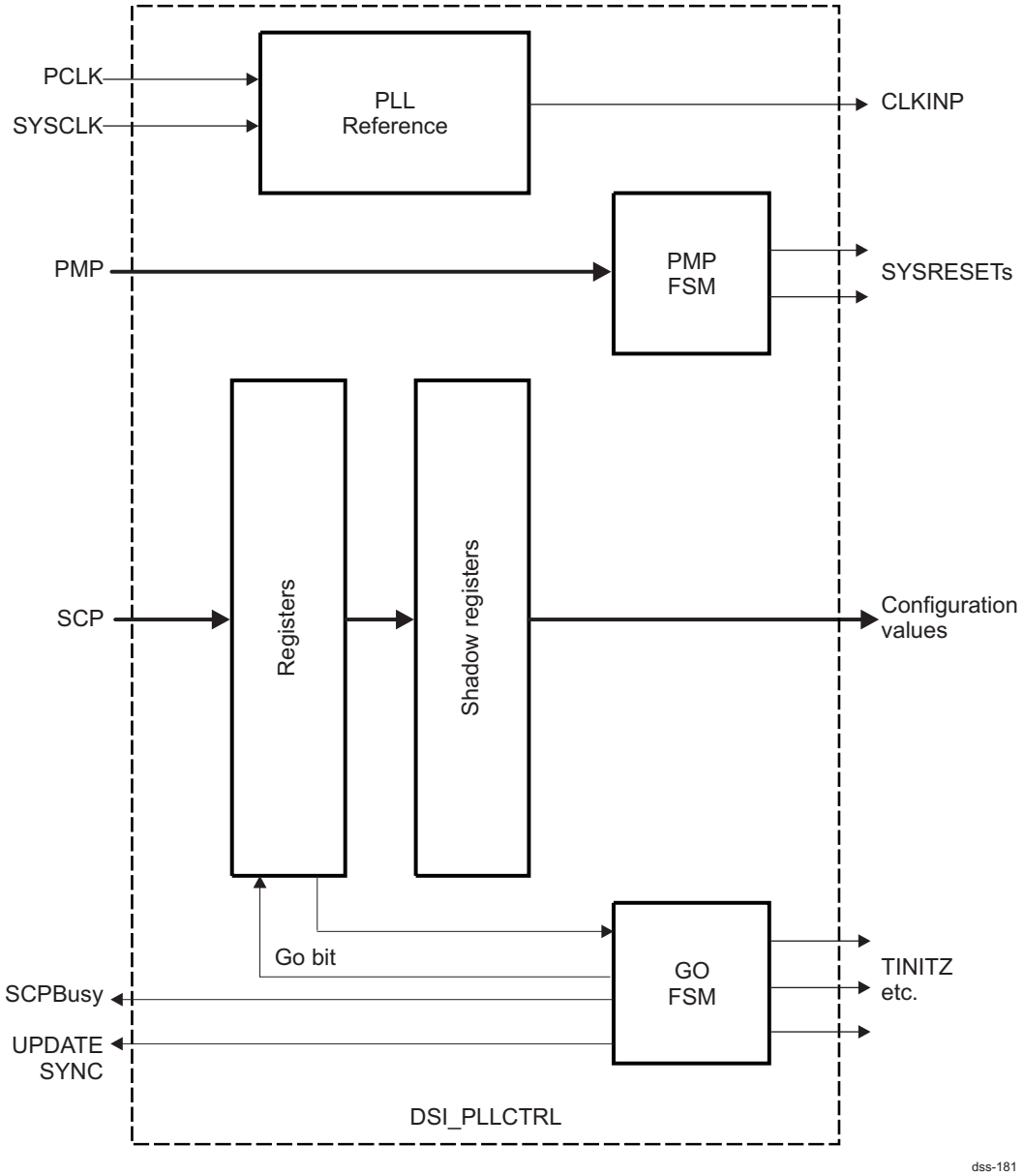
15.5.5.1 Software Reset

The DSI PLL control module does not have its own software reset. It is reset by the DSI protocol engine. Nevertheless, software users can monitor the reset status of the DSI PLL control module by reading the DSS.DSI_PLL_STATUS[0] DSI_PLLCTRL_RESET_DONE status bit.

15.5.5.2 DSI PLL Programming Blocks

Figure 15-139 shows the DSI PLL programming blocks.

Figure 15-139. DSI PLL Programming Blocks



15.5.5.3 DSI PLL Go Sequence

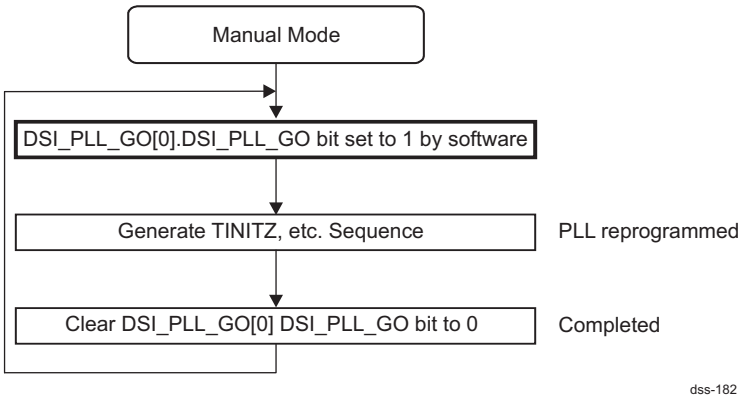
In Manual Mode (DSS.DSI_PLL_CONTROL[0] DSI_PLL_AUTOMODE bit set to 0), the DPLL requires a sequence on TINITZ, TENABLE and TENABLEDIV to update the configuration values and start the locking sequence.

Once all the configuration values have been programmed into the registers, the GO bit should be set. The appropriate sequence should then be sent on the TINITZ, TENABLE & TENABLEDIV pins, respecting the timing requirements of the ADPLLV2. The DSS.DSI_PLL_GO[0] DSI_PLL_GO bit will be cleared to 0 at the end of the sequence.

The TENABLEDIV signal is shared with the HSDIVIDER module, so that will be programmed at the same time. In this mode, the software should deassert CLKINEN by unsetting the DSS.DSI_PLL_CONFIGURATION2[14] DSI_PHY_CLKINEN to 0 and to assert HSDIVBYPASS correctly by setting the DSS.DSI_PLL_CONFIGURATION2[20] DSI_HSDIVBYPASS bit to 1 to prevent uncontrolled frequencies affecting the DSI_PHY and display subsystem during PLL locking. In manual mode the shadow register should be updated anyway so that valid values are present when later selecting automatic mode.

Figure 15-140 shows the DSI PLL Go flowchart in manual mode (DSS.DSI_PLL_CONTROL[0] DSI_PLL_AUTOMODE bit set to 0).

Figure 15-140. DSI PLL Go Sequence (Manual Mode)

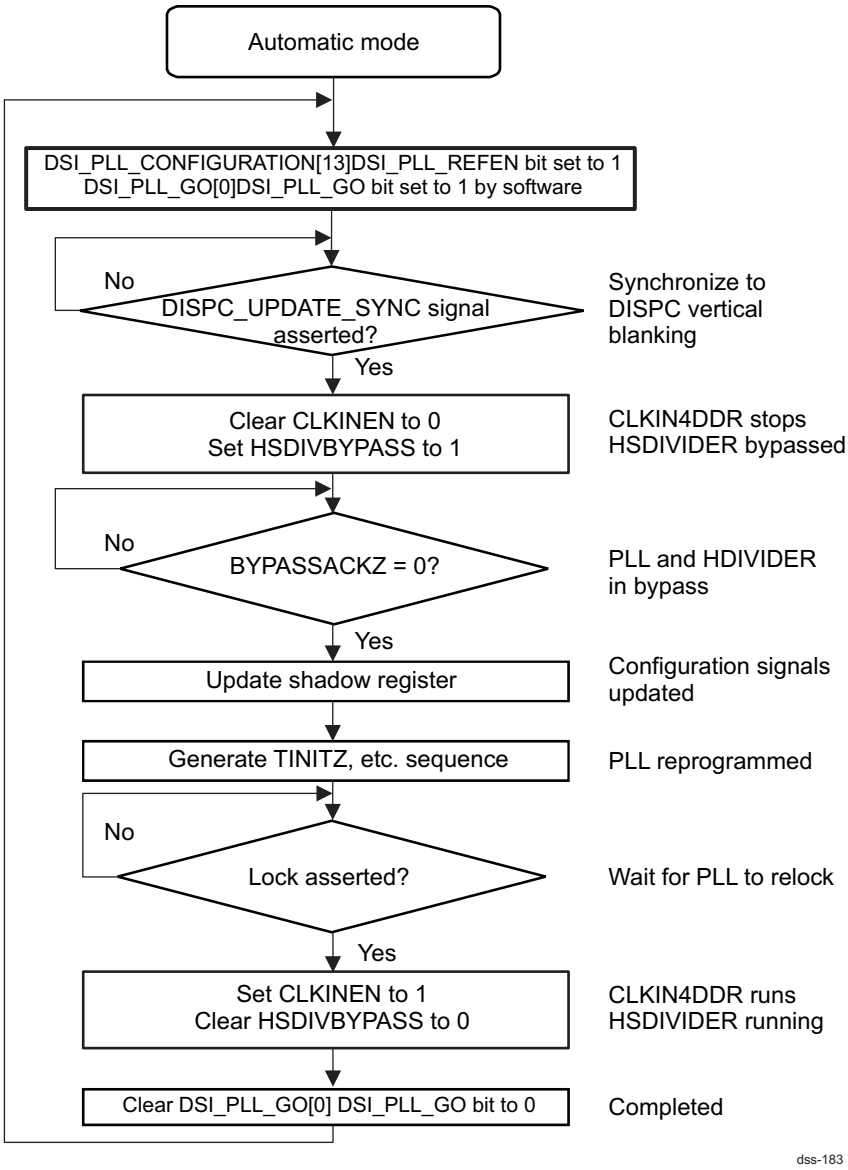


NOTE: All thick-outlined blocks show operations performed by software. Other blocks show operations performed by hardware.

In automatic Mode (DSS.DSI_PLL_CONTROL[0] DSI_PLL_AUTOMODE bit set to 1),the TINITZ, TENABLE and TENABLEDIV sequence and the update of the PLL configuration from the DSI_PLL_CONFIGURATION2 register will be deferred until the time of the front porch time signal sent by the DISPC module. This is intended to simplify the software to implement a configuration change (such as a frequency change to support a different link bandwidth). In this mode CLKINEN, HSDIVBYPASS and REFEN will be controlled automatically and the register value is overridden.

Figure 15-141 shows the DSI PLL Go flowchart in automatic mode (DSS.DSI_PLL_CONTROL[0] DSI_PLL_AUTOMODE bit set to 1).

Figure 15-141. DSI PLL Go Sequence (Automatic Mode)



NOTE: All thick-outlined blocks show operations performed by software. Other blocks show operations performed by hardware.

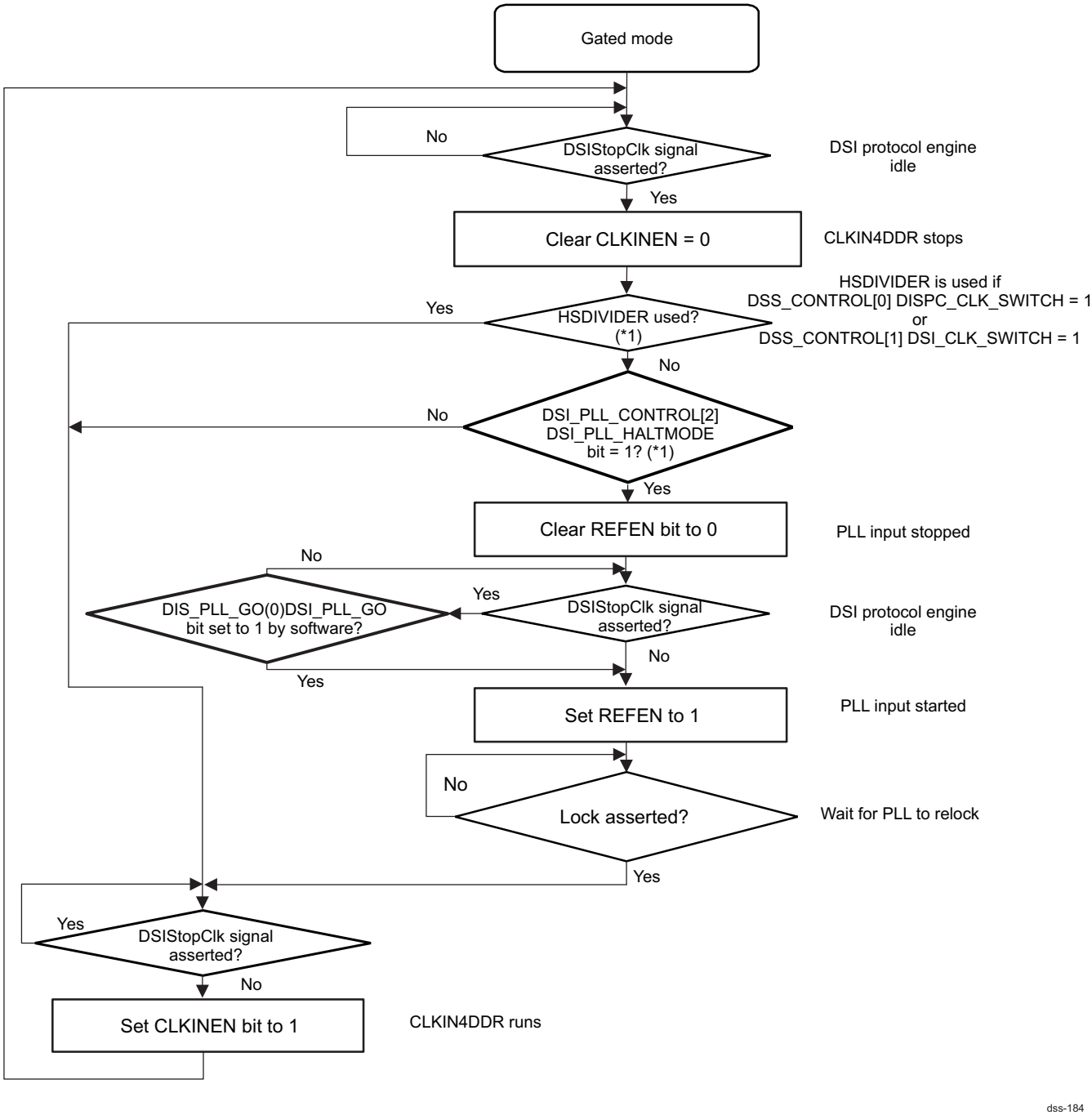
15.5.5.4 DSI PLL Clock Gating Sequence

Clock gating may be used to reduce system power consumption when the DSI protocol engine indicates that it does not need the clock. If the HSDIVIDER is not used, then the PLL can also be stopped (at the cost of additional unstarting latency).

The DSI protocol engine can verify when the PLL has unstarted by inspecting the LOCK signal (DSS.DSI_PLL_STATUS[1] DSI_PLL_LOCK status bit). Since TxByteClkHS is stopped when CLKIN4DDR is stopped, this should obviate the need for any explicit feedback that the clock has been unstarted in the other case. This flow chart should run even if the DSS.DSI_PLL_GO[0] DSI_PLL_GO bit has not been set.

Figure 15-142 shows the DSI PLL gated mode sequence.

Figure 15-142. Gated Mode Sequence



NOTE: All thick-outlined blocks show operations performed by software.

15.5.5.5 DSI PLL Lock Sequence

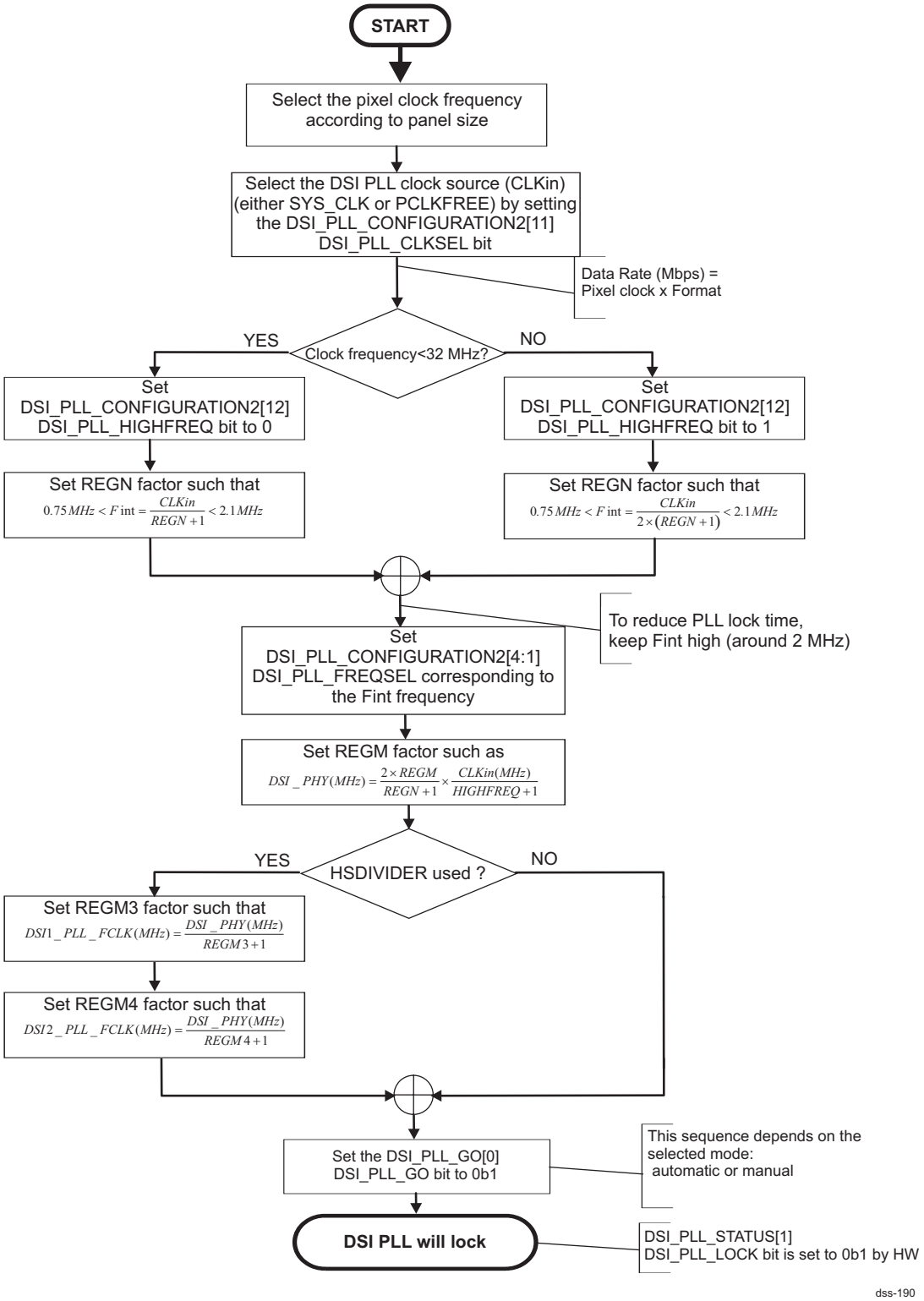
The DSI PLL (ADPLLv2) generates CLKIN4DDR. The HSDIVIDER generates two clocks: DSI1_PLL_FCLK connected to the display controller (DISPC) and the DSI2_PLL_FCLK connected to the DSI protocol engine. If these two clocks are not used, the HSDIVIDER functions are not required.

CLKIN4DDR is twice the data rate and four times the DSI output clock frequency. The DSI PLL factors need to be calculated based on the required input and output frequencies, keeping the PLL internal reference frequency in the appropriate range:

- REGM factor is programed by DSS.DSI_PLL_CONFIGURATION1[18:8] DSI_PLL_REGM bit field
- REGN factor is programed by DSS.DSI_PLL_CONFIGURATION1[7:1] DSI_PLL_REGN bit field
- REGM3 factor is programed by DSS.DSI_PLL_CONFIGURATION1[22:19] DSI_CLOCK_DIV bit field
- REGM4 factor is programed by DSS.DSI_PLL_CONFIGURATION1[26:23] DSIPROTO_CLOCK_DIV bit field

Figure 15-143 shows the programming sequence.

Figure 15-143. DSI PLL Programming Sequence



Notes:

- The REGM3 and REGM4 factors must according with the following conditions:
  - The DSI1_PLL_FCLK and DSI2_PLL_FCLK frequencies must be a multiple of the PCLK frequency (for proper settings of PCD and LCD factors in the DISPC)
  - The DSI1_PLL_FCLK and DSI2_PLL_FCLK frequencies must be lower than 173 MHz
- Most of the other DSI PLL programming values are available for software flexibility but it is not recommended to update the values in normal use. See Section 15.5.5.7 for details on DSI PLL recommended values.

DSI PLL programming examples:

- WVGA Display on one data pair: Pixel clock (PCLK) = 30 MHz with 18-BPP pixel format

The data rate is 30 x 18 = 540 Mbps on one data lane. Therefore, the frequency on the data lane is twice the data rate: 1080 MHz.

The frequency on the clock lane is 270 MHz (1080 divided by 4).

The SYS_CLK at 26 MHz is selected as the clock reference by setting the DSS.DSI_PLL_CONFIGURATION2[11] DSI_PLL_CLKSEL to 0b0.

Set the DSS.DSI_PLL_CONFIGURATION2[12] DSI_PLL_HIGHFREQ to 0b0 as PCLK is lower than 32 MHz.

Set Fint to 2 MHz as PLL internal reference frequency: Set REGN to 12 (divide by 13) by setting the DSS.DSI_PLL_CONFIGURATION1[7:1] DSI_PLL_REGN bit field to 0xC and set DSI_PLL_CONFIGURATION2[4:1] DSI_PLL_FREQSEL bit field to 0x7 (1.75 to 2.1 MHz range)

To get CLKIN4DDR to 1080 MHz, set the REGM factor to 270 by setting the DSS.DSI_PLL_CONFIGURATION1[18:8] DSI_PLL_REGM to 0x10E.

CLKIN4DDR = 2 x 270/13 x 26/1 = 1080 MHz

Since DSI1_PLL_FCLK and DSI2_PLL_FCLK (REGM3 and REGM4 factors) must be multiple of PCLK and also lower than 173 MHz, program these frequencies to 90 MHz by setting the REGM3 and REGM4 factors to 11 (divide by 12). This is done by setting the DSS.DSI_PLL_CONFIGURATION1[22:19] DSS_CLOCK_DIV bit field and DSS.DSI_PLL_CONFIGURATION1[26:23] DSIPROTO_CLOCK_DIV to 0xB:

DSI1_PLL_FCLK = DSI2_PLL_FCLK = 1080/12 = 90 MHz

- XGA Display on two data pairs: Pixel clock (PCLK) = 60 MHz with 16-BPP pixel format

  The data rate is (60 x 16)/2 = 480 Mbps on each data lane. Therefore, the frequency on the data lane is twice the data rate: 960 MHz.

  The frequency on the clock lane is 240 MHz (960 divided by 4).

  The SYS_CLK at 26 MHz is selected as the clock reference by setting the DSS.DSI_PLL_CONFIGURATION2[11] DSI_PLL_CLKSEL bit to 0b0.

  Set the DSS.DSI_PLL_CONFIGURATION2[12] DSI_PLL_HIGHFREQ bit to 0b0 as the source clock frequency (SYS_CLK in this example) is lower than 32 MHz.

  Set Fint to 2 MHz as PLL internal reference frequency: Set REGN to 12 (divide by 13) by setting the DSS.DSI_PLL_CONFIGURATION1[7:1] DSI_PLL_REGN bit field to 0xC and set DSI_PLL_CONFIGURATION2[4:1] DSI_PLL_FREQSEL bit field to 0x7 (1.75 to 2.1 MHz range)

  To get CLKIN4DDR to 960 MHz, set the REGM factor to 240 by setting the DSS.DSI_PLL_CONFIGURATION1[18:8] DSI_PLL_REGM to 0x0F0.

  CLKIN4DDR = 2 x 240/13 x 26/1 = 960 MHz

  Since DSI1_PLL_FCLK and DSI2_PLL_FCLK (REGM3 and REGM4 factors) must be multiple of PCLK and also lower than 173 MHz, program these frequencies to 120 MHz by setting the REGM3 and REGM4 factors to 7 (divide by 8). This is done by setting the DSS.DSI_PLL_CONFIGURATION1[22:19] DSS_CLOCK_DIV bit field and DSS.DSI_PLL_CONFIGURATION1[26:23] DSIPROTO_CLOCK_DIV to 0x7:

  DSI1_PLL_FCLK = DSI2_PLL_FCLK = 960/8 = 120 MHz

15.5.5.6 DSI PLL Error Handling

The PLL lock and recalibration signals may be monitored to detect loss of lock or requirement to recalibrate (due to large temperature change since the last lock request):

- The DSS.DSI_PLL_STATUS[1] DSI_PLL_LOCK status bit gives the DSI PLL lock state.
- The DSS.DSI_PLL_STATUS[2] DSI_PLL_RECAL status bit informs if the PLL must be uncalibrated

These signals can also generate interrupts at DSI protocol engine level:

-
  - DSS.DSI_IRQSTATUS[9], PLL_RECAL_IRQ bit
  - DSS.DSI_IRQSTATUS[8] PLL_UNLOCK_IRQ
  - DSS.DSI_IRQSTATUS[7] PLL_LOCK_IRQ
- The PLL_LOCK_IRQ interrupt indicates that the DSI PLL is locked. To monitor this event, read the DSS.DSI_IRQSTATUS[7] PLL_LOCK_IRQ bit. Set this bit to 1 to clear the status bit.
- The PLL_UNLOCK_IRQ interrupt indicates that the DSI PLL is unlocked. To monitor this event, read the DSS.DSI_IRQSTATUS[8] PLL_UNLOCK_IRQ bit. Set this bit to 1 to clear the status bit.
- The PLL_RECAL_IRQ interrupt indicates that the DSI PLL must be recalibrated. To monitor this event, read the DSS.DSI_IRQSTATUS[9] PLL_RECAL_IRQ bit. Set this bit to 1 to clear the status bit.

The PLL reference loss and limp status signals can also be monitored :

- The DSS.DSI_PLL_STATUS[3] DSI_PLL_LOSSREF status bit informs if the DSI PLL has lost the reference.
- The DSS.DSI_PLL_STATUS[4] DSI_PLL_LIMP status bit informs about the DSI PLL limp status.

15.5.5.7 DSI PLL Recommended Values

Table 15-66 shows the DSI PLL recommended values.

Table 15-66. Recommended Programming Values

Field Name	Value	Description
DSI_HSDIV_SYSRESET	0	Allow power FSM to control
DSI_PLL_SYSRESET	0	Allow power FSM to control
DSI_PLL_HALTMODE	-	See Section 15.5.5.4 for details
DSI_PLL_GATEMODE	-	See Section 15.5.5.4 for details
DSI_PLL_AUTOMODE	-	See Section 15.5.5.4 for details
DSI_PLL_GO	1->0	Write a 1 when PLL is to be (re-)locked with new parameters. This bit is cleared by hardware when the PLL request has completed
DSIPROTO_CLOCK_DIV	See (1)	DSI protocol engine clock divider
DSS_CLOCK_DIV	See (1)	DSS clock divider
DSI_PLL_REGM	See (1)	Feedback clock divider
DSI_PLL_REGN	See (1)	Reference clock divider
DSI_PLL_STOPMODE	1	Required to use GATEMODE bit
DSI_HSDIVBYPASS	0	PLL is controlling HSDIVIDER bypass
DSI_PROTO_CLOCK_PWDN	0	If PLL/HSDIVIDER is used as the DSI protocol clock source
DSI_PROTO_CLOCK_EN	1	If PLL/HSDIVIDER is used as the DSI protocol clock source
DSS_CLOCK_PWDN	0	If PLL/HSDIVIDER is used as the DSS clock source
DSS_CLOCK_EN	1	If PLL/HSDIVIDER is used as the DSS clock source

(1) The bit field value should be set according to the desired clock frequency.

Table 15-66. Recommended Programming Values (continued)

Field Name	Value	Description
DSI_BYPASSEN	0	To use PLL as the clock source. For small displays it may be possible to use the DSS functional clock, in which case this bit should be set to 1
DSI_PHY_CLKINEN	1	Enable CLKIN4DDR
DSI_PLL_REFEN	1	Enable PLL reference
DSI_PLL_HIGHFREQ	0/1	Set to 1 if the clock reference is higher than 32 MHz (21 MHz if DSS.DSI_PLL_CONFIGURATION1[7:1] DSI_PLL_REGN = 0)
DSI_PLL_CLKSEL	0/1	Set to 0 to use DSS2_ALWON_FCLK as the PLL reference or set to 1 to use PCLKFREE as the PLL reference, in this case the DISPC must be using DSS1_ALWON_FCLK.
DSI_PLL_LOCKSEL	0x0	Phase lock criteria to lock the PLL
DSI_PLL_DRIFTGUARDEN	0x0	The RECAL status/interrupt should be used to decide when to perform a PLL uncalibration No automatic uncalibration will be performed
DSI_PLL_TIGHTPHASELOCK	0	Normal criteria
DSI_LOWCURRSTDBY	0/1	Set to 0 for fast PLL unlock, but higher standby current Set to 1 for leakage level standby current, but longer unlock time
DSI_PLL_PLLLPMODE	0	Normal operation For smaller display sizes may be possible to set to 1
DSI_PLL_FREQSEL	See (1)	Must be set according to the PLL internal reference frequency (after N divider) See register descriptions for values
DSI_PLL_IDLE	0	PLL active

15.5.6 DSI Complex I/O Basic Programming Model

15.5.6.1 Software Reset

The clock domain using the TxByteClkHS from the DSI complex I/O has dedicated reset done information in the DSS.DSI_COMPLEXIO_CFG1[29] RESET_DONE bit. The DSS.DSI_SYSCONFIG[1] SOFT_RESET bit is used to reset the TxByteClkHS power domain. A dummy read using the SCP interface to any DSI_PHY register is required after DSI_PHY reset to complete the reset of the DSI complex I/O.

15.5.6.2 Reset-Done Bits

The DSI complex I/O has several clock domains. The reset status for each clock domain is provided in DSS.DSI_PHY_CFG5 register:

- DSS.DSI_PHY_CFG5[31] RESETDONETXBYTECLK bit : Reset done for the TXBYTECLK domain.
- DSS.DSI_PHY_CFG5[28:26] RESETDONETXCLKESCi bits: Reset done for the TXCLKESC domain for lane i (i between 0 and 2).
- DSS.DSI_PHY_CFG5[30] RESETDONESCPCLK bit: Reset done for the SCP clock domain. Software users must perform a dummy read on this bit to initiate the reset sequence of the SCP finite state machine. When the reset sequence is complete, the RESETDONESCPCLK signal goes high and software users can read again the DSS.DSI_PHY_CFG5[30] RESETDONESCPCLK bit to ensure that

the value is now 1.

Note: The software should not write in the DSI_PHY_SCP registers before the DSS.DSI_PHY_CFG5[30] RESETDONESCPCLK bit is set to 1.

- DSS.DSI_PHY_CFG5[29] RESETDONEPWRCLK bit: Reset done for the PWR clock domain. The reset sequence of the PWR finite state machine is complete when the RESETDONEPWRCLK signal goes high.

15.5.6.3 Pad Configuration

The number of lanes is configurable through the DSS.DSI_COMPLEXIO_CFG1 register.

It is not allowed to change on the fly the position (by modifying the DATAi_POSITION with i=1 or 2 and CLOCK_POSITION bit fields), P/N order (Positive/Negative order of the differential pair by modifying the DATAi_POL with i=1 or 2 and CLOCK_POL) or number of active data lanes (by modifying the DSS.DSI_COMPLEXIO_CFG1[10:8] DATA2_POSITION bit). To add or remove the lane #2, it is required to be in OFF mode for the DSI complex I/O.

The minimum requirement for the number of lanes is one clock lane and one data lane. Note that by default, the data lane 2 is not connected (the DSS.DSI_COMPLEXIO_CFG1[10:8] DATA2_POSITION bit reset value is 0).

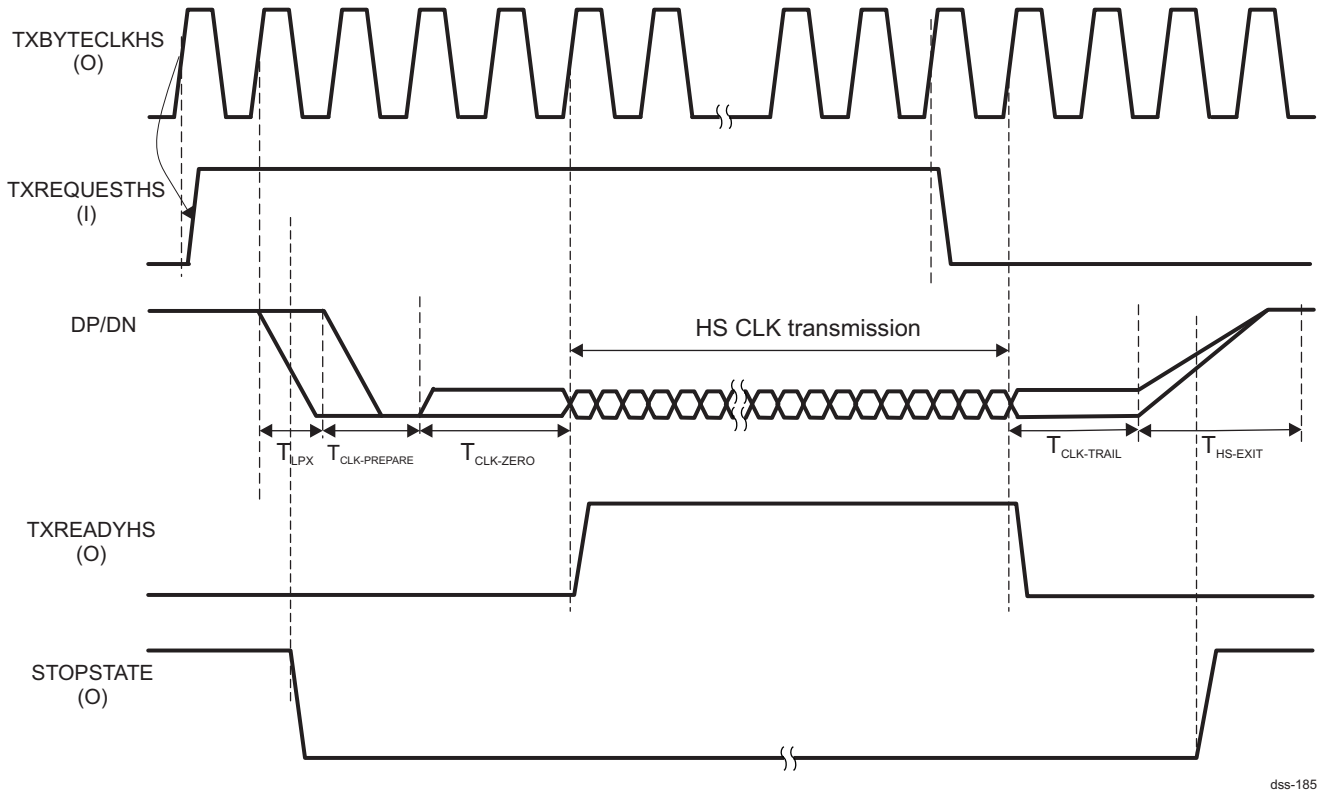
15.5.6.4 Display Timing Configuration

Depending on the CLKIN4DDR frequency settings programmed with the DSI PLL control module, software users must program accordingly the timing parameters in the DSI complex I/O registers.

15.5.6.4.1 High-Speed Clock Transmission

Figure 15-144 shows an example of high-speed Clock Transmission.

Figure 15-144. High-Speed Clock Transmission



TXByteClkHS is an output clock which is derived by dividing the CLKIN4DDR clock by 16.

To begin transmission, the protocol drives TXREQUESTHS high on a rising edge of TXByteClkHS. The PHY detects this signal on the next rising edge, following which it initiates the LP Start of Transmission (SoT) procedure.

During a high-speed Clock Transmission, these parameters are defined in multiples of CLKIN4DDR and programmed by the following register bit fields:

- TLPX timing is programmed by the DSS.DSI_PHY_CFG1[22:16] TLPX_HALF bit field.
- THS-PREPARE timing is programmed by the DSS.DSI_PHY_CFG0[31:24] THS_PREPARE bit field.
- TCLK-ZERO timing is programmed by the DSS.DSI_PHY_CFG1[7:0] TCLK_ZERO bit field.

. TCLK-ZERO is extended, if required, so that the entire LP SoT procedure lasts an integer number of TXByteClkHS cycles.

At the end of the SoT procedure, HS clock transmission begins. At the same time, TXREADYHS is made high.

To stop clock transmission, the protocol drives TXREQUESTHS low on a rising edge of TXByteClkHS. The DSI_PHY detects this change in TXREQUESTHS on the next edge and stops clock transmission. TXREADYHS is made low.

The DSI_PHY then goes through the LP End of Transmission (EoT) procedure. TCLK-TRAIL and THS-EXIT parameters are also multiples of CLKIN4DDR and programmed by the following register fields:

- TCLK-TRAIL timing is programmed by the DSS.DSI_PHY_CFG1[15:8] TCLK_TRAIL bit field.
- THS-EXIT timing is programmed by the DSS.DSI_PHY_CFG0[7:0] THS_EXIT bit field.

The DSI_PHY completes the SoT and EoT procedures, once begun, irrespective of any change in PPI signals. If TXREQUESTHS goes low during the SoT procedure, the PHY start the EoT procedure immediately after finishing the SoT procedure and no clock is transmitted.

STOPSTATE is high whenever the line is in LP-11 state, as determined by the outputs of the Low Power Receivers. This signal is not synchronized with TXByteClkHS.

It is requires that the high speed clock be present for some time before (TCLK-PRE) and some time after (TCLK-POST) high speed data transmission. The protocol must ensure that these timings are met by asserting and deasserting TXREQUESTHS appropriately.

The PHY ensures that the clock signal has a quadrature-phase with respect to a toggling bit sequence on any Data Lane, and a rising edge in the center of the first transmitted bit of every Data byte. These relations are not described in the timing diagram.

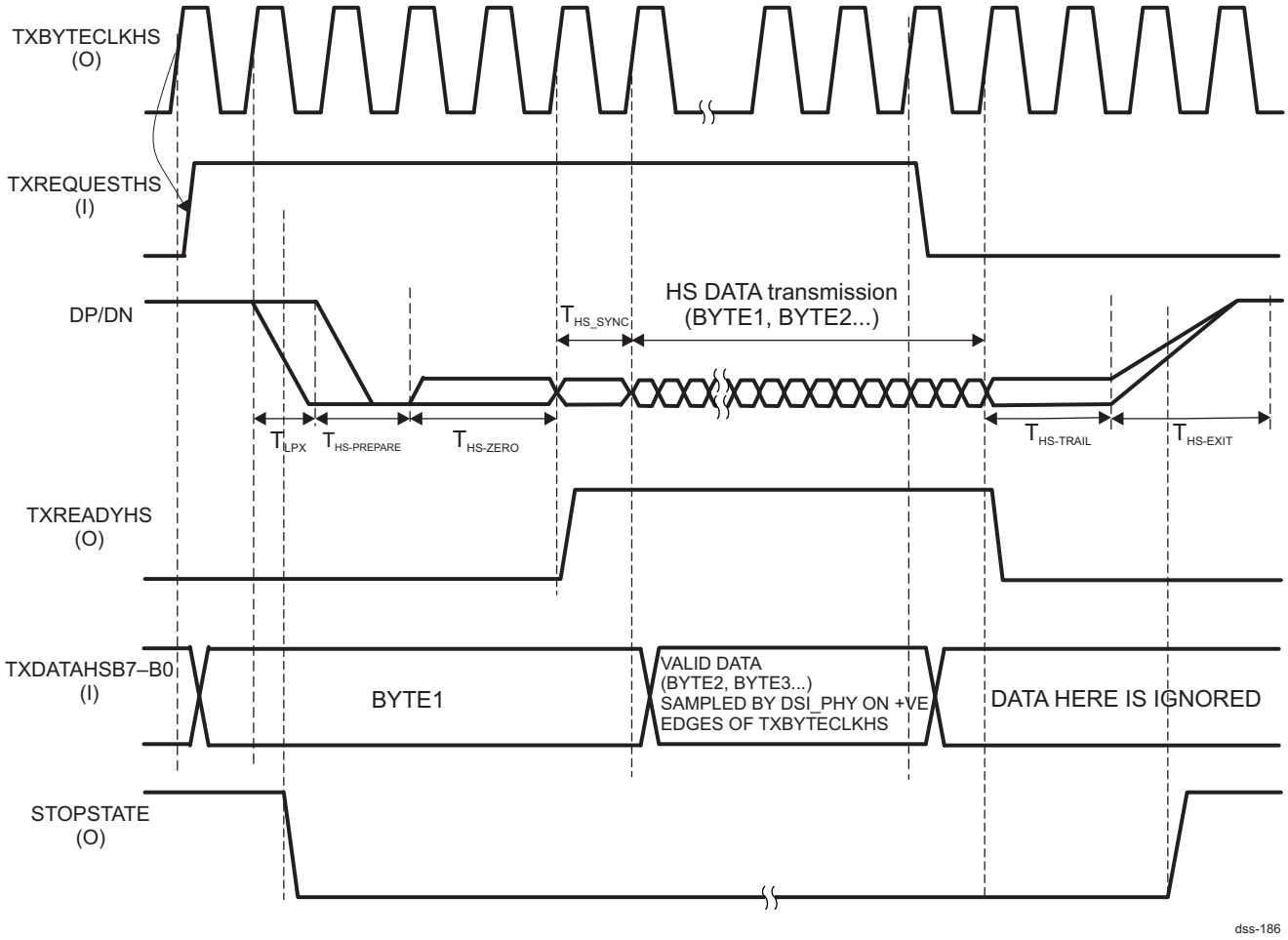
CLKIN4DDR can be shut off 300ns after the clock lane goes to STOPSTATE. Alternatively, CLKIN4DDR can be shut down after TCLK-Trail + THS-Exit + 2 Txbyteclk periods after TxRequestHS falling edge is received by DSI_PHY.

The DSI protocol engine should ensure that TXREQUESTESC, TXULPSCLK and TURNREQUEST are low whenever TXREQUESTHS is asserted.

15.5.6.4.2 High-Speed Data Transmission

Figure 15-145 shows an example of high-speed Data Transmission.

Figure 15-145. High-Speed Data Transmission



TXByteClkHS is an output clock which is derived by dividing CLKIN4DDR by 16.

To begin transmission, the protocol drives TXDATAHS with the first byte of data on a rising edge of TXByteClkHS. It also makes TXREQUESTHS high the same rising edge. The PHY detects TXREQUESTHS going high on the next rising edge of TXByteClkHS, following which it initiates the LP Start of Transmission (SoT) procedure.

During a high-speed Data Transmission, these timings are multiple of CLKIN4DDR and programmed by the following register bit fields:

- TLPX timing is programmed by the DSS.DSI_PHY_CFG1[22:16] TLPX_HALF bit field.
- THS-PREPARE + THS-ZERO timing is programmed by the DSS.DSI_PHY_CFG0[23:16] THS_PREPARE_THS_ZERO bit field.

. THS-ZERO will be extended, if required, so that the entire LP SoT procedure lasts an integer number of TXByteClkHS cycles. THS-SYNC corresponds to the length of the sync pattern which is eight high-speed bits.

Towards the end of the SoT procedure, the PHY makes TXREADYHS high on a positive edge of TXByteClkHS and then start accepting data from TXDATAHS from the next positive edge onwards. The protocol is expected to provide (new) valid data on TXDATAHS on every positive edge of TXByteClkHS if TXREADYHS is high.

At the end of the SoT procedure, HS data transmission begins. HS Data Transmission happens LSB first.

To stop data transmission, the protocol drives TXREQUESTHS low on a rising edge of TXByteClkHS. The PHY detects this change in TXREQUESTHS on the next edge and stops data transmission. TXREADYHS is made low and data on TXDATAHS, from that point, is ignored.

The PHY then goes through the LP End of Transmission (EoT) procedure. THS-TRAIL and THS-EXIT are also multiples of CLKIN4DDR and programmed by the following register bit fields:

- THS-TRAIL timing is programmed by the DSS.DSI_PHY_CFG0[15:8] THS_TRAIL bit field.
- THS-EXIT timing is programmed by the DSS.DSI_PHY_CFG0[7:0] THS_EXIT bit field.

The PHY completes the SoT and EoT procedures, once begun, irrespective of any change in PPI signals. If TXREQUESTHS goes low during the SoT procedure, the PHY start the EoT procedure immediately after finishing the SoT procedure and no data is transmitted.

STOPSTATE is high whenever the line is in LP-11 state, as determined by the outputs of the Low Power Receivers. This signal is not synchronized with TXByteClkHS.

The Protocol should ensure that TXREQUESTESC, TXULPSCLK and TURNREQUEST are low whenever TXREQUESTHS is asserted.

15.5.6.4.3 Other DSI_PHY Transmission and Reception

The timing of the following sequences defined in the DSI_PHY protocol cannot be programed by users:

- Low-power data transmission
- Escape mode trigger command transmission
- ULP state command transmission on data lanes
- ULP state transmission on clock lane
- Turn-around request in transmit mode
- Turn-around request in receive mode
- Low-power data in receive mode
- Low-power trigger in receive mode
- ULP state command on clock lane in receive mode
- ULP state command on data lane in receive mode

15.5.6.5 Error Handling

A dedicated register for the DSI complex I/O, DSS.DSI_COMPLEXIO_IRQSTATUS, indicates the state of each error provided by the DSI complex I/O error signals. The DSI_PHY reports the following errors:

- DSS.DSI_COMPLEXIO_IRQSTATUS[7:5] ERR_ESCi_IRQ: ERRESC is asserted if an unrecognized Escape entry command is received. This remains high until the next change in the line state.
- DSS.DSI_COMPLEXIO_IRQSTATUS[2:0] ERRSYNCESCi_IRQ: If the number of bits received during a low-power data transmission is not a multiple of eight when the transmission ends, ERRSYNCESC is made high and remains high until the next change in line state.
- DSS.DSI_COMPLEXIO_IRQSTATUS[12:10] ERRCONTROLi_IRQ: ERRCONTROL is asserted if an incorrect line state sequence is detected. For example, if a turn-around request or escape mode request is immediately followed by a Stop state instead of the required Bridge state, this signal is asserted and remains asserted until the next change in the line state.
- DSS.DSI_COMPLEXIO_IRQSTATUS ERRCONTENTIONLP0_i_IRQ: ERRCONTENTION0LPDX and ERRCONTENTION0LPDY are asserted when the Lane module detects a contention situation on lines DX and DY respectively while trying to drive the lines low. Contention is detected only if it lasts at least 50ns
- DSS.DSI_COMPLEXIO_IRQSTATUS ERRCONTENTIONLP1_i_IRQ: ERRCONTENTION1LPDX and ERRCONTENTION1LPDY are asserted when the Lane module detects a contention situation on lines DX and DY respectively while trying to drive the lines high. Contention is detected only if it lasts at least 50ns

The ULPSACTIVENOT signal goes low which indicates to the protocol that the PHY has entered ULP state. When all the ULPSActiveNot signals are low, the DSS.DSI_COMPLEXIO_IRQSTATUS[30] ULPSACTIVENOT_ALL0_IRQ event is generated. When all the ULPSActiveNot signals are high, the DSS.DSI_COMPLEXIO_IRQSTATUS[31] ULPSACTIVENOT_ALL1_IRQ event is generated.

When any of the events defined in DSS.DSI_COMPLEXIO_IRQSTATUS register happened, the DSS.DSI_IRQSTATUS[10] COMPLEXIO_ERR_IRQ bit is set to 1 at DSI protocol engine level.

The software must take appropriate action when receiving the interrupt indicating the error from the complex I/O. The action can be:

- Reset of the DSI protocol engine module
- Reset of the peripheral though reset trigger or directly driving the hardware reset pin of the display module
- Ignore the error

15.5.7 RFBI Basic Programming Model

The RFBI programming model must be used for LCD display support only.

15.5.7.1 DISPC Control Registers

The following DISPC registers are used in RFBI mode:

- The STALL mode is selected by setting the DSS.DISPC_CONTROL[11] STALLMODE bit. The DSS.DISPC_CONTROL[5] GOLCD bit should not be set to 1 but the display controller configuration (DMA engine, pipelines associated to the LCD output,..) should be set before enabling the LCD output by setting the DSS.DISPC_CONTROL[0] LCDENABLE bit to 1.
- To enable the hardware handcheck to avoid underflow, the DSS.DISPC_CONTROL[16] FIFOHANDCHECK should be set to 1. The reset value of this bit is 0. The handcheck applies to the pipelines connected to the LCD output. It should be disabled before resetting the DSS.DISPC_CONTROL[11] STALLMODE bit to 0. The new setting for the FIFO handcheck is used for the following frames.

Note: The LCD output is disabled at the end of the transfer of the frame. The software must reenable the LCD output to generate a new frame by setting the DSS.DISPC_CONTROL[0] LCDENABLE to 1. See Figure 15-146.

15.5.7.2 RFBI Control Registers

The following registers define the RFBI control registers:

- DSS.RFBI_CONTROL
- DSS.RFBI_PIXEL_CNT
- DSS.RFBI_LINE_NUMBER

15.5.7.2.1 High Threshold

The DSS.RFBI_CONTROL[6:5] HIGHTHRESHOLD bit field is used to define the threshold to be used for the generation of the DMA request to receive data into the interconnect FIFO (24×32 FIFO depth) through the address of the register RFBI_DATA. It should be the size of the burst. The supported values are 4x32, 8x32 and 16x32. The system DMA receives the DMA request and is in charge of providing the correct number of bytes. If the DSS.RFBI_CONTROL[7] DISABLE_DMA_REQ bit is reset, the DMA request is generated when there is enough room in the interconnect FIFO to accept the full burst. In case the RFBI receives writes L4 requests to the RFBI_DATA location when the interconnect FIFO is full, the request is not accepted. The RFBI waits for a free entry in the interconnect FIFO to accept the L4 request.

If the DSS.RFBI_CONTROL[7] DISABLE_DMA_REQ bit is set, the DMA request is not generated. The threshold value is ignored.

Note: Software users can access the RFBI_DATA location without using the DMA request and without programming the high threshold value (backward mode).

15.5.7.2.2 Bypass Mode

Setting the DSS.RFBI_CONTROL[1] BYPASSMODE bit directly outputs the LCD controller output to the LCD panel. Resetting this bit directs the MPU module to send commands/parameters and data from the input video port FIFO.

15.5.7.2.3 Enable

Setting/resetting the DSS.RFBI_CONTROL[0] ENABLE bit enables/disables the RFBI module. The hardware resets the enable bit after all of the pixels are sent to the panel. The DSS.RFBI_PIXEL_CNT[31:0] PIXELCNT bit field value defines the number of pixels to send to the LCD panel. When the transfer is finished, the configuration used can be modified.

Table 15-67. RFBI Behavior

RFBI_CONTROL[1] BYPASSMODE bit value	RFBI_CONTROL[0] ENABLE bit value	RFBI Behavior
0	0	L4 interconnect can write command/param/data and read data/status from the Remote Frame Buffer (RFB). L4 interconnect access can only be done to the CSx actually active
0	1	The DISPC sends pixels to the RFB.

The stall signal is asserted when the module is disabled. Through the L4 port, pixels can be sent to the LCD panel only when the pixel count has reach the value 0x0

Note: The LCD output is disabled at the end of the transfer of the frame. The software must reenable the LCD output to generate a new frame by setting the DSS.DISPC_CONTROL[0] LCDENABLE to 1. See Figure 15-146.

15.5.7.2.4 Configuration Selection

Setting the DSS.RFBI_CONTROL[3:2] CONFIGSELECT bit field selects the configuration number (1 or 0 if bits are set or reset). The registers associated with the configuration output the data to the LCD panel.

If both chip-selects are selected, the configuration for the first chip-select is used (except for the polarity of the RFBI_CS1 signal defined by the second configuration) and both devices connected to the CS signals are driven in parallel. In read mode, if both chip-selects are set, only RFBI_CS0 is asserted to read data from the device connected on RFBI_CS0. In write mode with two chip-selects selected, the RFBI can write to the two devices simultaneously.

15.5.7.2.5 ITE Bit

Set the DSS.RFBI_CONTROL[4] ITE bit to start capturing the data from the display controller. This bit has no effect if the trigger mode is set to external. The display controller must be configured in the STALL mode to account for the RFBI_DISPC_STALL signal. Setting the trigger mode to external (DSS.RFBI_CONFIGi[3:2] TRIGGERMODE bit field set to 0x1 or 0x2) causes the DSS.RFBI_CONTROL[4] ITE bit to be ignored. The corresponding chip-select must be selected when this bit is set by users.

The RFBI_DISPC_STALL signal is asserted when at least one of the following cases occur:

- Default status when no data to capture from the display controller
- High FIFO threshold reached

- End of the transfer (number of data to output)
- Reset of the RFBI module
- DSS.RFBI_CONTROL[0] ENABLE bit reset to 0x0

The RFBI_DISPC_STALL signal is deasserted when the DSS.RFBI_CONTROL[0] ENABLE bit is set to 0x1 and at least one of the following cases occur:

- Low FIFO threshold reached
- External TE occurs and the DSS.RFBI_CONFIGi[3:2] TRIGGERMODE bit field is set to 0x1 or 0x2 for automatic external trigger (start of the transfer, the FIFO pointers are reset, the FIFO is empty).
- DSS.RFBI_CONTROL[4] ITE bit set to 0x1 by users (start of the transfer, the FIFO pointers are reset, the FIFO is empty).

15.5.7.2.6 Number of Pixels to Transfer

Setting the DSS.RFBI_PIXEL_CNT[31:0] PIXELCNT bit field value directs the application to indicate the number of pixels to be transferred to the LCD panel. The value can be changed only when the DSS.RFBI_CONTROL[0] ENABLE is reset.

During the transfer, the hardware decrements the register when a pixel is sent to the remote frame buffer. When the DSS.RFBI_CONTROL[0] ENABLE bit is set and a new value is written in the DSS.RFBI_PIXEL_CNT register when the current value in the register is a non-zero (the remaining number of pixels to transfer), the ongoing transfer is aborted.

From the L4 interconnect side, if DSS.RFBI_CONFIGi[10:9] CYCLEFORMAT bit field is equal to 0x3 and DSS.RFBI_CONFIGi[8:7] L4FORMAT is equal to 0x0, an even number of write accesses to the data register should be performed before accessing any other register (CMD/PARAM/STATUS/READ).

When DSS.RFBI_CONFIGi[10:9] CYCLEFORMAT bit field is 0x3 - meaning that 2 pixels are sent over 3 cycles -, the number of pixels to be programmed in the DSS.RFBI_PIXEL_CNT[31:0] PIXELCNT bit field should be a multiple of 2. If another CYCLEFORMAT is used, the value for PIXELCNT can be odd or even. This constraint is valid for data provided on the L4 interconnect port and from the display controller.

If the DSS.RFBI_CONFIGi[10:9] CYCLEFORMAT bit field is equal to 0x3, DSS.RFBI_CONFIGi[8:7] L4FORMAT is equal to 0, and back to back register write is processed, the following registers should be written after the first data: RFBI_CMD, RFBI_PARAM, RFBI_READ, and RFBI_STATUS. The whole data transfer should first be performed before being able to write to any other registers (RFBI_CMD, RFBI_PARAM, RFBI_READ, and RFBI_STATUS).

15.5.7.2.7 Programmable Line Number

When the trigger mode is set to external trigger mode with HSYNC and VSYNC or the TE, the hardware resets the line counter when the VSYNC occurs and, after a programmable number of lines (the HSYNC pulse occurs for every line), the transfer to the LCD panel begins. When the programmable line number is 0, only the VSYNC pulse indicates the beginning of the transfer in both modes: HSYNC/VSYNC and TE (logical OR operation between HSYNC and VSYNC).

15.5.7.3 RFBI Configuration

The following registers define the RFBI configuration:

- DSS.RFBI_SYSCONFIG
- DSS.RFBI_SYSSTATUS
- DSS.RFBI_CONFIG0 (configuration 0) and DSS.RFBI_CONFIG1 (configuration 1)
- DSS.RFBI_VSYNC_WIDTH
- DSS.RFBI_HSYNC_WIDTH

The configuration register for one configuration can be accessed only when the configuration is not in use (based on the value of the RFBI_CONTROL[3:2] CONFIGSELECT bit field).

15.5.7.3.1 Parallel Mode

The DSS.RFBI_CONFIGi[1:0]PARALLELMODE bit field (with i = 0, 1) defines the width of the interface (8-, 9-, 12-, or 16-bit parallel).

15.5.7.3.2 Trigger Mode

Setting the DSS.RFBI_CONFIG[3:2]TRIGGERMODE bit field configures the trigger on the external TE signal (RFBI_TE_VSYNC), or external with VSYNC/HSYNC with the programmable number of HSYNCs to begin the transfer in both cases or the internal programmable DSS.RFBI_CONTROL[4] ITE bit.

15.5.7.3.3 VSYNC Pulse Width (Minimum Value)

The DSS.RFBI_VSYNC_WIDTH[15:0] MINVSYNCPULSEWIDTH bit field defines the minimum number of L4 clock cycles of the VSYNC pulse for detection on VSYNC. It allows differentiation between VSYNC and HSYNC, which are ORed on the same signal and is also used in the VSYNC/HSYNC mode on the two separate input lines.

- The VSYNC pulse width must be at least equal to two L4 cycles when HSYNC is not present.
- The VSYNC pulse width must be at least equal to four L4 cycles when HSYNC is present.

15.5.7.3.4 HSYNC Pulse Width (Minimum Value)

The DSS.RFBI_HSYNC_WIDTH[15:0] MINHSYNCPULSEWIDTH bit field defines the minimum number of L4 clock cycles of the HSYNC pulse for detection on HSYNC. It allows differentiation between VSYNC and HSYNC, which are ORed on the same signal, and is also used in the VSYNC/HSYNC mode on the separate two input lines. The HSYNC pulse width must always be at least equal to two L4 cycles to be detected.

15.5.7.3.5 Cycle Format

Setting the DSS.RFBI_CONFIGi[10:9] CYCLEFORMAT bit field (with i = 0, 1) defines which registers are used to format the data in the interconnect FIFO with the appropriate number of bits (starting from the LSB) and with the alignment on the interface as follows:

- DSS.RFBI_DATA_CYCLE_i (if DSS.RFBI_CONFIG[10:9] CYCLEFORMAT bit field = 00) only
  or
- DSS.RFBI_DATA_CYCLE1_i and DSS.RFBI_DATA_CYCLE2_i (if DSS.RFBI_CONFIG[10:9] CYCLEFORMAT bit field = 01)
  or
- DSS.RFBI_DATA_CYCLE1_i, DSS.RFBI_DATA_CYCLE2_i, and DSS.RFBI_DATA_CYCLE3_i (if DSS.RFBI_CONFIG[10:9] CYCLEFORMAT bit field = 10)

The data from the display controller and from the L4 interconnect are formatted based on the configuration of the DSS.RFBI_DATA_CYCLE_i registers.

15.5.7.3.6 Unused Bits

Based on the configuration, the undefined bits for each cycle are defined with the previous values of the bits at the same position in the previous cycle, 0s, or 1s (the unused bits can be at any position). The DSS.RFBI_CONFIGi[12:11] UNUSEDBITS bit field (with i = 0, 1) is used.

15.5.7.3.7 RFBI Timings

The timing registers for one configuration can be accessed only when the configuration is not in use (based on the value of the DSS.RFBI_CONTROL[3:2] CONFIGSELECT bit field). Granularity is defined using the DSS.RFBI_CONFIGi[4] TIMEGRANULARITY bit. This feature allows the extension of programmable ranges of timing parameters for the RFBI interface. Refer to Table 15-68 for the bits configuration values.

- Chip-select assertion/deassertion time

RFBI_A0 setup time to chip-select assertion is assured by the programmable chip-select assertion time from the start access time:
DSS.RFBI_ONOFF_TIMEi[3:0] CSONTIME bit field (with i = 0, 1).

The chip-select deassertion time from the start access time is programmable:
DSS.RFBI_ONOFF_TIMEi[9:4] CSOFFTIME bit field (with i = 0, 1)

CAUTION

Configuring DSS.RFBI_ONOFF_TIMEi[3:0] CSONTIME = DSS.RFBI_ONOFF_TIMEi[9:4] CSOFFTIME = 0 (with i = 0, 1) is not supported and must be avoided. This configuration creates contention on the bus and progressively damages the LCD panel.

- Chip-select pulse width

  The total chip-select pulse width is the time when write cycle time or read cycle time has completed and is programmable:
  DSS.RFBI_CYCLE_TIMEi[17:12] CSPULSEWIDTH bit field (with i = 0, 1)

  It applies on the read-to-write, write-to-read, read-to-read, and write-to-write access based on:

  – The DSS.RFBI_CYCLE_TIMEi [19] RRENABLE bit: Read-to-read access
  – The DSS.RFBI_CYCLE_TIMEi [20] WWENABLE bit: Write-to-write access
  – The DSS.RFBI_CYCLE_TIMEi [18] RWENABLE bit: Read-to-write access
  – The DSS.RFBI_CYCLE_TIMEi [21] WRENABLE bit: Write-to-read access

  By default, it applies to any access (read-to-read, read-to-write, write-to-read, write-to-write) when the chip-select changes.
- Access time

  The total access time is the time from when A0 becomes valid until data are sampled before deasserting the RE signal; access time is programmable:
  DSS.RFBI_CYCLE_TIMEi[27:22] ACCESSTIME bit field (with i = 0, 1)

  When reading the data on the bus, the data are sampled at the end of the access time, which occurs before the end of the read off time (DSS.RFBI_ONOFF_TIMEi[29:24] REOFFTIME, with i = 0, 1).
- Write enable cycle time

  The total write enable cycle time is the time from when A0 becomes valid until write cycle completion; the write enable cycle time is programmable:
  The DSS.RFBI_CYCLE_TIMEi[5:0] WECYCLETIME bit field (with i = 0, 1)
- Write enable assertion/deassertion time

  The WE assertion delay time from start access time is programmable:
  DSS.RFBI_ONOFF_TIMEi[13:10] WEONTIME bit field (with i = 0, 1)

  The WE deassertion delay time from the start access time is programmable:
  DSS.RFBI_ONOFF_TIMEi[19:14] WEOFFTIME bit field (with i = 0, 1)
- Read enable cycle

  The total read enable cycle time is the time when A0 becomes valid until read cycle completion; the read enable cycle time is programmable:
  The DSS.RFBI_CYCLE_TIMEi[11:6] RECYCLETIME bit field (with i = 0, 1)
- Read enable assertion/deassertion time

  The RE assertion delay time from the start access time is programmable:
  DSS.RFBI_ONOFF_TIMEi[23:20] REONTIME bit field (with i = 0, 1)

  The RE deassertion delay time from the start access time is programmable:
  DSS.RFBI_ONOFF_TIMEi[29:24] REOFFTIME bit field (with i = 0, 1)

At cycle time completion (read access or write access) all control signals (RFBI_CSi, RFBI_WE, and RFBI_RE, with i = 0, 1) are deasserted regardless of their deassertion time parameter values, if they are not deasserted already.

However, an exception to this forced deassertion exists when a pipelined request to the same chip-select or to a different chip-select is pending. Also, a control signal with deassertion time parameters equal to the cycle time parameter is not necessarily deasserted when a pipelined request to the same chip-select or different chip-select is pending. This prevents any unnecessary glitch transitions.

If no inactive cycles are required between successive accesses to the same chip-select (the DSS.RFBI_CYCLE_TIMEi[17:12] CSPULSEWIDTH bit field = 0, with i = 0, 1), and if assertion time parameters associated with the following access equal 0, the asserted control signals (RFBI_CSi, RFBI_WE, and RFBI_RE, with i = 0, 1) stay asserted. This is applicable to any read/write-to-read/write access combination.

Table 15-68 summarizes the configurations values for each timing bit.

Table 15-68. RFBI Timings Configuration

Configuration bits[1]	Granularity[2]	
	one	two
DSS.RFBI_ONOFF_TIMEi[3:0] CSONTIME	0 to 15	0 to 30
DSS.RFBI_ONOFF_TIMEi[9:4] CSOFFTIME	0 to 63	0 to 126
DSS.RFBI_CYCLE_TIMEi[17:12] CSPULSEWIDTH	0 to 63	0 to 126
DSS.RFBI_CYCLE_TIMEi[27:22] ACCESSTIME	0 to 63	0 to 126
DSS.RFBI_CYCLE_TIMEi[5:0] WECYCLETIME	0 to 63	0 to 126
DSS.RFBI_ONOFF_TIMEi[13:10] WEONTIME	0 to 15	0 to 30
DSS.RFBI_ONOFF_TIMEi[19:14] WEOFFTIME	0 to 63	0 to 126
DSS.RFBI_CYCLE_TIMEi[11:6] RECYCLETIME	0 to 63	0 to 126
DSS.RFBI_ONOFF_TIMEi[23:20] REONTIME	0 to 15	0 to 30
DSS.RFBI_ONOFF_TIMEi[29:24] REOFFTIME	0 to 63	0 to 126

(1) Where i = 0 or 1.
(2) Number of L4Clk cycles. The granularity can be configured using the DSS.RFBI_CONFIGi[4] TIMEGRANULARITY bit.

15.5.7.3.8 RFBI State-Machine

Referring to Table 15-37, the signals RFBI_A0, RFBI_RE, and RFBI_WE are asserted/deasserted based on the register accessed (DSS.RFBI_CMD, DSS.RFBI_PARAM, DSS.RFBI_DATA, DSS.RFBI_READ, and DSS.RFBI_STATUS). When the DSS.RFBI_SYSSTATUS[8] BUSY bit is set by hardware, any access to the registers is stalled, except for the RFBI_DATA register.

The DSS.RFBI_SYSSTATUS[9] BUSYRFBIDATA bit indicates whether there are still pending data in the interconnect FIFO associated with the register RFBI_DATA only.

- Command register

  Write a command at a time by writing in the DSS.RFBI_CMD register. If the previous command is not processed, the DSS.RFBI_SYSSTATUS[8] BUSY bit is set by hardware and the access to writing a new command is stalled.
- Parameter register

  Write a parameter at a time by writing in the DSS.RFBI_PARAM register.

  If the previous parameter is not processed, the DSS.RFBI_SYSSTATUS[8] BUSY bit is set by hardware and the access to writing a new parameter is stalled.
- Data register

  Write one or two pixels at a time by writing in the RFBI_DATA register (when DSS.RFBI_CONFIGi[10:9] CYCLEFORMAT = 0x3 with i = 0, 1, two pixels must be written contiguously, no other access to RFBI registers except DSS.RFBI_DATA is allowed).

  The pixels are formatted based on the specified cycle format. If two pixels are written into the 32-data register, the DSS.RFBI_CONFIGi[8:7] L4FORMAT bit field indicates the number of pixels for each L4 access to the register and the order of the pixels

  If the previous data are not processed, the DSS.RFBI_SYSSTATUS[8] BUSY bit is set by hardware and any access for writing new data is stalled. When the DSS.RFBI_SYSSTATUS[8] BUSY bit is reset by hardware, the access is not stalled.

- Read/status register

  Send through the command and parameter registers the correct information to receive data in the data or status register. The read data from the LCD panel is initiated by writing into the DSS.RFBI_READ or DSS.RFBI_STATUS registers. In this case, the DSS.RFBI_SYSSTATUS[8] BUSY bit is set until the data are available in the register.

  When the DSS.RFBI_SYSSTATUS[8] BUSY bit is set by hardware, the read or write access is stalled until the register is updated with a new value from the LCD panel. To avoid the stall, the software can poll the DSS.RFBI_SYSSTATUS[8] BUSY bit until it is reset by hardware. To receive the data, send the appropriate command/parameters.

15.5.7.3.9 RFBI Configuration Flow Charts

The RFBI configuration depends on the trigger mode used by the application. The available trigger modes are:

- Internal trigger mode when setting the DSS.RFBI_CONFIGi[3:2] TRIGGERMODE bit field to 0x0
- External trigger mode:
  - TE external trigger mode when setting the DSS.RFBI_CONFIGi[3:2] TRIGGERMODE bit field to 0x1
  - HSYNC/VSYNC external trigger mode when setting the DSS.RFBI_CONFIGi[3:2] TRIGGERMODE bit field to 0x2

Figure 15-146 gives an example of how to program and use the RFBI module:

Figure 15-146. How to Use RFBI

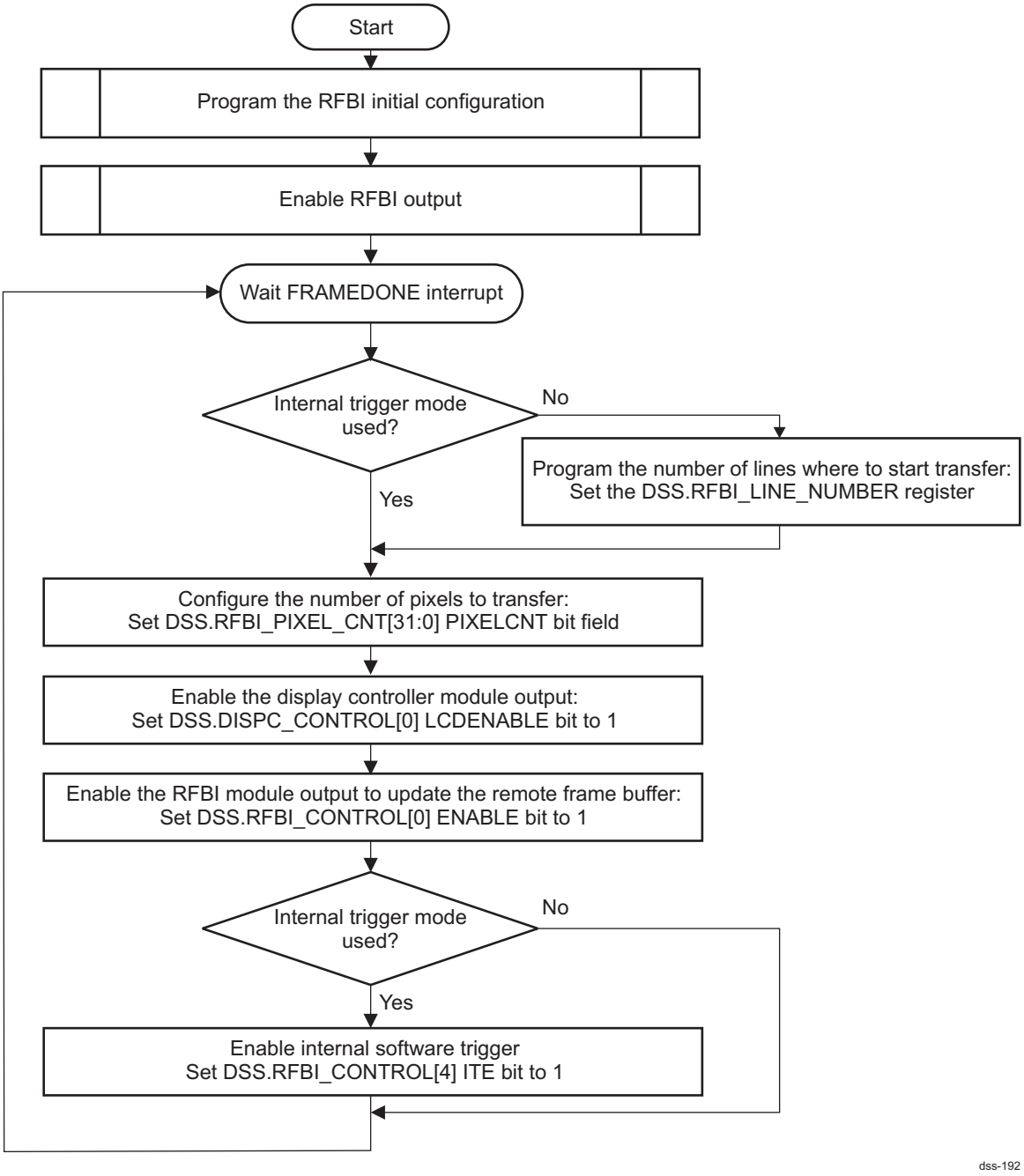


Figure 15-147 details how to configure the RFBI registers:

Figure 15-147. RFBI Initial Configuration

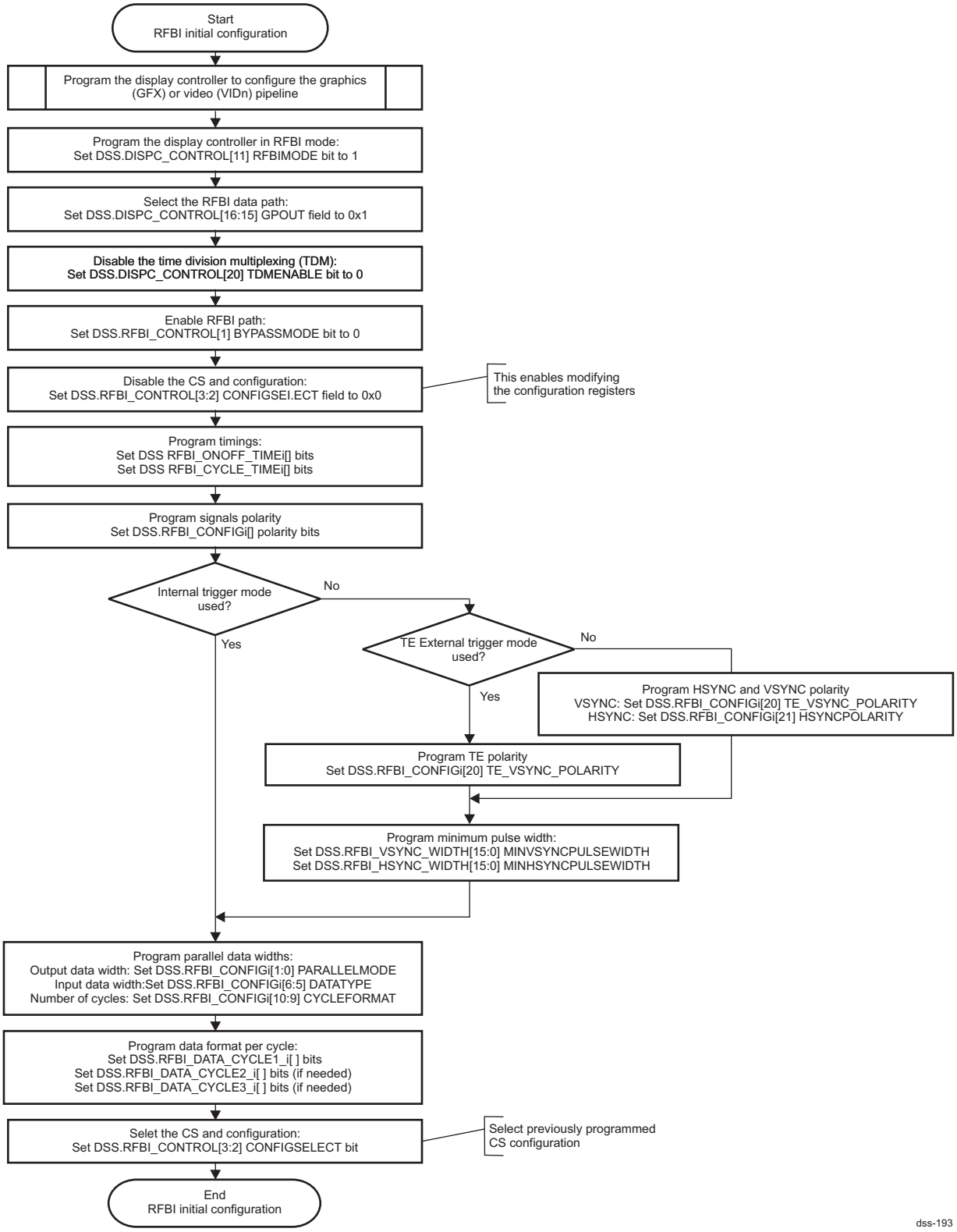
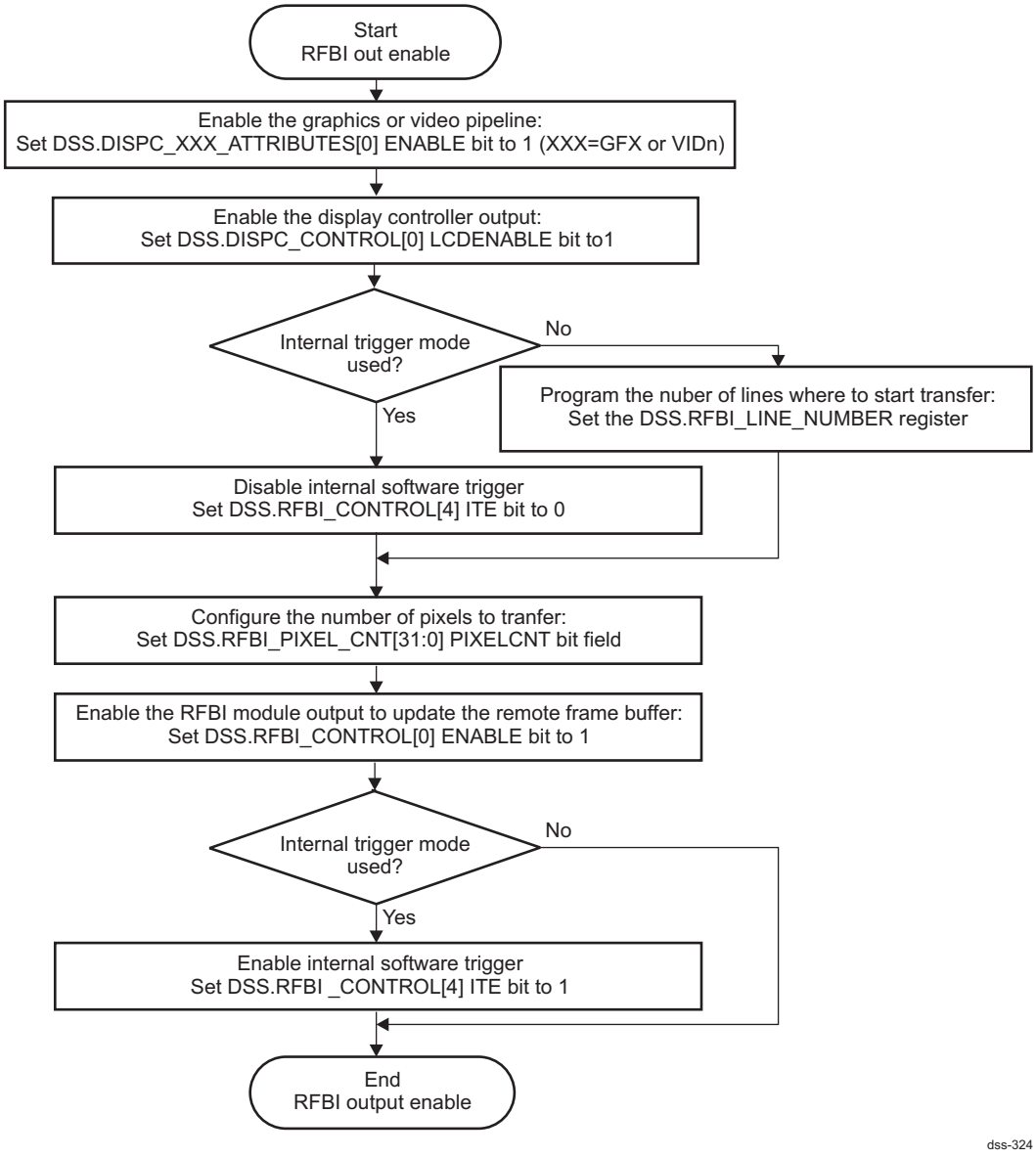


Figure 15-148 describes how to enable the RFBI module.

Figure 15-148. RFBI Output Enable



15.5.8 Video Encoder Basic Programming Model

15.5.8.1 Video Encoder Software Reset

By setting the DSS.VENC_F_CONTROL[8] RESET bit to 1, the video encoder is reset. This bit is automatically cleared by hardware when the reset is done.

Note: Before changing the standard (NTSC or PAL) and all the related registers, a software reset is required to properly initialize the VENC module.

15.5.8.2 Video DAC Settings

The video output format can be either:

- one composite video (CVBS) output signal with video DAC1 or
- two separated luma/chroma output signals with both video DAC1(luma analog signal) and DAC2 (chroma analog signal)

  Selection is performed with DSS.DSS_CONTROL[6] VENC_OUT_SEL bit:

  - When VENC_OUT_SEL bit is set to 0 (reset value), video DAC1 is selected for composite video (CVBS) signal
  - When VENC_OUT_SEL bit is set to 1, video DAC1 is selected for luma signal

  One of the first VENC settings is to enable the output of the video DACs. This setting depends on the video output format:

  - CVBS video output format: Enable video DAC1 composite output by setting the DSS.VENC_OUTPUT_CONTROL[1] COMPOSITE_ENABLE bit to 1
  - Separated luma/chroma output format: Enable video DAC1 luma output by setting the DSS.VENC_OUTPUT_CONTROL[0] LUMA_ENABLE bit to 1 and enable video DAC2 chroma output by setting the DSS.VENC_OUTPUT_CONTROL[2] CHROMA_ENABLE bit to 1

15.5.8.3 Video Encoder Programming Sequence

1. Set the DSS.VENC_F_CONTROL[8] RESET bit to 1 to perform a software reset of the VENC module.
2. Before any configuration change, save the DSS.DISPC_IRQENABLE register value (DSS interrupts context), and then disable the display subsystem interrupts by setting the DSS.DISPC_IRQENABLE register to 0x0000.
3. Configure the video encoder registers as described in Table 15-69, depending on the video standard used (PAL or NTSC). The DSS.VENC_F_CONTROL and DSS.VENC_SYNC_CTRL registers must be the last ones to be changed by software.
4. Set the DSS.DISPC_CONTROL[6] GODIGITAL bit and the DSS.DISPC_CONTROL[1] DIGITALENABLE bit to 1.
5. Wait for the first VSYNC pulse signal.
6. Clear the SYNCLOSTDIGITAL interrupt by setting the DSS.DISPC_IRQSTATUS[15] SYNCLOSTDIGITAL bit to 1.
7. Set the DSS.DISPC_IRQENABLE register to the value saved in step 2 (restore the DSS interrupts context).

15.5.8.4 Video Encoder Register Settings

For video encoder programming, see Table 15-69. This table lists the register values to use in standard applications. These values are validated programming values only for the TV display support (NTSC 601 and PAL 601 standards).

Table 15-69. Video Encoder Register Programming Values

Register Name	NTSC 601	PAL 601
VENC_F_CONTROL	0x00000000	0x00000000
VENC_VIDOUT_CTRL	0x00000001	0x00000001
VENC_SYNC_CTRL	0x00008040	0x00000040
VENC_LLEN	0x00000359	0x0000035F
VENC_FLENS	0x0000020C	0x00000270
VENC_HFLTR_CTRL	0x00000000	0x00000000
VENC_CC_CARR_WSS_CARR	0x043F2631	0x2F7225ED
VENC_C_PHASE	0x00000000	0x00000000
VENC_GAIN_U	0x00000102	0x00000111
VENC_GAIN_V	0x0000016C	0x00000181
VENC_GAIN_Y	0x0000012F	0x00000140

Table 15-69. Video Encoder Register Programming Values (continued)

Register Name	NTSC 601	PAL 601
VENC_BLACK_LEVEL	0x00000043	0x0000003B
VENC_BLANK_LEVEL	0x00000038	0x0000003B
VENC_X_COLOR	0x00000007	0x00000007
VENC_M_CONTROL	0x00000001	0x00000002
VENC_BSTAMP_WSS_DATA	0x00000038	0x0000003F
VENC_S_CARR	0x21F07C1F	0x2A098ACB
VENC_LINE21	0x00000000	0x00000000
VENC_LN_SEL	0x01310011	0x01290015
VENC_L21_WC_CTL	0x0000F003	0x0000F603
VENC_HTRIGGER_VTRIGGER	0x00000000	0x00000000
VENC_SAVID_EAVID	0x069300F4	0x06A70108
VENC_FLEN_FAL	0x0016020C	0x00180270
VENC_LAL_PHASE_RESET	0x00060107	0x00040135
VENC_HS_INT_START_STOP_X	0x008E0350	0x00880358
VENC_HS_EXT_START_STOP_X	0x000F0359	0x000F035F
VENC_VS_INT_START_X	0x01A00000	0x01A70000
VENC_VS_INT_STOP_X_VS_INT_START_Y	0x020701A0	0x000001A7
VENC_VS_INT_STOP_Y_VS_EXT_START_X	0x01AC0024	0x01AF0000
VENC_VS_EXT_STOP_X_VS_EXT_START_Y	0x020D01AC	0x000101AF
VENC_VS_EXT_STOP_Y	0x00000006	0x00000025
VENC_AVID_START_STOP_X	0x03480078	0x03530083
VENC_AVID_START_STOP_Y	0x02060024	0x026C002E
VENC_FID_INT_START_X_FID_INT_START_Y	0x0001008A	0x0001008A
VENC_FID_INT_OFFSET_Y_FID_EXT_START_X	0x01AC0106	0x002E0138
VENC_FID_EXT_START_Y_FID_EXT_OFFSET_Y	0x01060006	0x01380001
VENC_TVDETGP_INT_START_STOP_X	0x00140001	0x00140001
VENC_TVDETGP_INT_START_STOP_Y	0x00010001	0x00010001
VENC_GEN_CTRL	0x00F90000	0x00FF0000
VENC_OUTPUT_CONTROL	0x0000000A (composite video CVBS) 0x0000000D (split video S-video)	0x0000000A (composite video CVBS) 0x0000000D (split video S-video)
VENC_OUTPUT_TEST	0x00000000	0x00000000

Note: The following display controller registers must be programmed to the NTSC 601 video standard:

DSS.DISPC_SIZE_DIG[10:0] PPL = 720 - 1 = 719 = 0x2CF
DSS.DISPC_SIZE_DIG[26:16] LPP = (482/2) - 1 = 240 = 0xF0
DSS.DISPC_GFX_BA0 or DSS.DISPC_VIDn_BA0 = Base address of even bit field data
DSS.DISPC_GFX_BA1 or DSS.DISPC_VIDn_BA1 = Base address of odd bit field data

15.5.9 SDI Basic Programming Model

This section describes how to configure the SDI module for FlatLink3G-compliant LCD panel support.

Configure the SDI, and then configure the display controller.

15.5.9.1 SDI Configuration

Note: To use the SDI with the TI FlatLink3G display panel:

- Set the DSS.DISPC_CONTROL[3] STNTFT bit to 1 (active matrix display operation mode)
- Set the DSS.DISPC_CONTROL[9:8] TFTDATALINES bit field to 0x3 (24-bit data output)
- Set the DSS.DSS_SDI_CONTROL[1:0] SDI_BWSEL bit field to 0x2 (color depth is 24 bits)

15.5.9.1.1 SDI PLL Configuration

The programming of the PLL divisor must be consistent with the choice of the number of data pairs. The PLL divisor determines the multiplication factor from the pixel clock frequency to data rate.

Table 15-70 lists some example values of the PLL settings for TI FlatLink3G.

Table 15-70. PLL Divisor Example Values for TI FlatLink3G

		24-BPP data		
		1 data pair	2 data pairs	3 data pairs
Recommended values	Pixel clock frequency (MHz)	15	30	65
	DSS.DSS_PLL_CONTROL[16:11]			
	SDI_PLL_REGN bit field (N factor = SDI_PLL_REGN bit field value + 1)	0x7	0xE	0xF
	DSS.DSS_PLL_CONTROL[10:1]			
	SDI_PLL_REGM bit field (M factor)	0xF0	0xE1	0x140
	DDS.DSS_PLL_CONTROL[19]			
	SDI_PLL_HIGHFREQ	0	0	1
	Effective PLL divisor value (M/N)	30	15	10

For more details on SDI PLL programming settings, see Section 15.6.3.

15.5.9.1.2 Signal Features Configuration

For proper SDI operation, configure the following features in the display controller:

- Set the DSS.DISPC_POL_FREQ[16] RF bit to drive HSYNC and VSYNC on the rising edge of PCLK.
- Set the DSS.DISPC_POL_FREQ[17] ONOFF bit.
- Clear the DSS.DISPC_POL_FREQ[14] IPC bit to drive the pixel data on the rising edge of PCLK.
- Set the DSS.DISPC_CONTROL[29] LCDENABLEPOL bit to set the LCD-enable signal active high.

The following settings are not required by the SDI2 interface, but these polarities settings are the common polarities used by FlatLink3G displays:

- Set the DSS.DISPC_POL_FREQ[13] IHS bit to 1 to set HSYNC active low.
- Set the DSS.DISPC_POL_FREQ[12] IVS bit to 1 to set VSYNC active low.
- Clear the DSS.DISPC_POL_FREQ[15] IEO bit to set the data enable active high.

15.5.9.1.3 Number of Data Pairs

To select the number of data pairs used to transmit display data between the device and the display panel, set the DSS.DSS_SDI_CONTROL[3:2] SDI_PRSEL field.

Table 15-71 lists the different values for these bits based on the number of data pairs.

Table 15-71. SDI Pixel Data Format

	DSS.DSS_SDI_CONTROL[3:2] SDI_PRSEL
1 data pair (DATA1 pair)	00
2 data pairs (DATA1 and DATA2 pair)	01
3 data pairs (DATA1, DATA2, and DATA3 pair)	10

15.5.9.2 SDI Power-Management Programming Sequence

This section describes the software sequence for powering on and powering down the SDI module.

15.5.9.2.1 SDI Reset State

After DSS power domain reset or power-on reset, the SDI module is in low-power mode by default with the following configuration:

- Display SS pads multiplexing is in parallel mode and not in SDI mode:
  - CONTROL.CONTROL_PADCONF_DSS_DATA10[2:0]MUXMODE0 and CONTROL.CONTROL_PADCONF_DSS_DATA10[18:16]MUXMODE1 field values is 0x0: SDI_DATA1N/SDI_DATA1P signals are not mapped on dss_data10/dss_data11 pins.
  - CONTROL.CONTROL_PADCONF_DSS_DATA12[2:0]MUXMODE0 and CONTROL.CONTROL_PADCONF_DSS_DATA12[18:16]MUXMODE1 bit field values is 0x0: SDI_DATA2N/SDI_DATA2P signals are not mapped on dss_data12/dss_data13 pins.
  - CONTROL.CONTROL_PADCONF_DSS_DATA14[2:0]MUXMODE0 and CONTROL.CONTROL_PADCONF_DSS_DATA14[18:16]MUXMODE1 bit field values is 0x0: SDI_DATA3N/SDI_DATA3P signals are not mapped on dss_data14/dss_data15 pins.
  - CONTROL.CONTROL_PADCONF_DSS_DATA18[2:0]MUXMODE0 and CONTROL.CONTROL_PADCONF_DSS_DATA18[18:16]MUXMODE1 bit field values is 0x0: SDI_VSYNC/SDI_HSYNC signals are not mapped on dss_data18/dss_data19 pins.
  - CONTROL.CONTROL_PADCONF_DSS_DATA20[2:0]MUXMODE0 and CONTROL.CONTROL_PADCONF_DSS_DATA20[18:16]MUXMODE1 bit field values is 0x0: SDI_DEN/SDI_STP signals are not mapped on dss_data20/dss_data21 pins.
  - CONTROL.CONTROL_PADCONF_DSS_DATA22[2:0]MUXMODE0 and CONTROL.CONTROL_PADCONF_DSS_DATA22[18:16]MUXMODE1 bit field values is 0x0: SDI_CLKP/SDI_CLKN signals are not mapped on dss_data22/dss_data23 pins.
- SDI PLL is in reset mode: the DSS.DSS_PLL_CONTROL[18]SDI_PLL_SYSRESET bit value is 0x0.

15.5.9.2.2 SDI Power_On Sequence

When exiting OFF mode, to power-on the SDI module, software users must program the following sequence:

1. Program the SDI pads multiplexing configuration:
   a. Set the CONTROL.CONTROL_PADCONF_DSS_DATA10[2:0]MUXMODE0 bit field to 0x1 to select SDI_DATA1N signal on dss_data10 pin and CONTROL.CONTROL_PADCONF_DSS_DATA10[18:16]MUXMODE1 bit field to 0x1 to select SDI_DATA1P signal on dss_data11 pin.
   b. For 2-data and 3-data pair mode: set the CONTROL.CONTROL_PADCONF_DSS_DATA12[2:0]MUXMODE0 bit field to 0x1 to select SDI_DATA2N on dss_data12 pin and CONTROL.CONTROL_PADCONF_DSS_DATA12[18:16]MUXMODE1 bit field to 0x1 to select SDI_DATA2P signal on dss_data13 pin.
   c. For 3-data pair mode: set the CONTROL.CONTROL_PADCONF_DSS_DATA14[2:0]MUXMODE0 bit field to 0x1 to select SDI_DATA3N on dss_data14 pin and CONTROL.CONTROL_PADCONF_DSS_DATA14[18:16]MUXMODE1 bit field to 0x1 to select SDI_DATA3P signal on dss_data15 pin.

d. Set the CONTROL.CONTROL_PADCONF_DSS_DATA18[2:0]MUXMODE0 bit field to 0x1 to select SDI_VSYNC signal on dss_data18 pin and CONTROL.CONTROL_PADCONF_DSS_DATA18[18:16]MUXMODE1 bit field to 0x1 to select SDI_HSYNC signal on dss_data19 pin.
e. Set the CONTROL.CONTROL_PADCONF_DSS_DATA20[2:0]MUXMODE0 bit field to 0x1 to select SDI_DEN signal on dss_data20 pin and CONTROL.CONTROL_PADCONF_DSS_DATA20[18:16]MUXMODE1 bit field to 0x1 to select SDI_STP signal on dss_data21 pin.
f. Set the CONTROL.CONTROL_PADCONF_DSS_DATA22[2:0]MUXMODE0 bit field to 0x1 to select SDI_CLKP signal on dss_data22 pin and CONTROL.CONTROL_PADCONF_DSS_DATA22[18:16]MUXMODE1 bit field to 0x1 to select SDI_CLKN signal on dss_data23 pin.

2. Wait 1 ms.
3. Release the SDI PLL reset signal by setting the DSS.DSS_PLL_CONTROL[18]SDI_PLL_SYSRESET bit to 0x1.

Note: The same software sequence is needed when the DSS power domain is set to ON.

15.5.9.2.3 SDI Power-Down Sequence

To power down the SDI module, software users must program the following sequence:

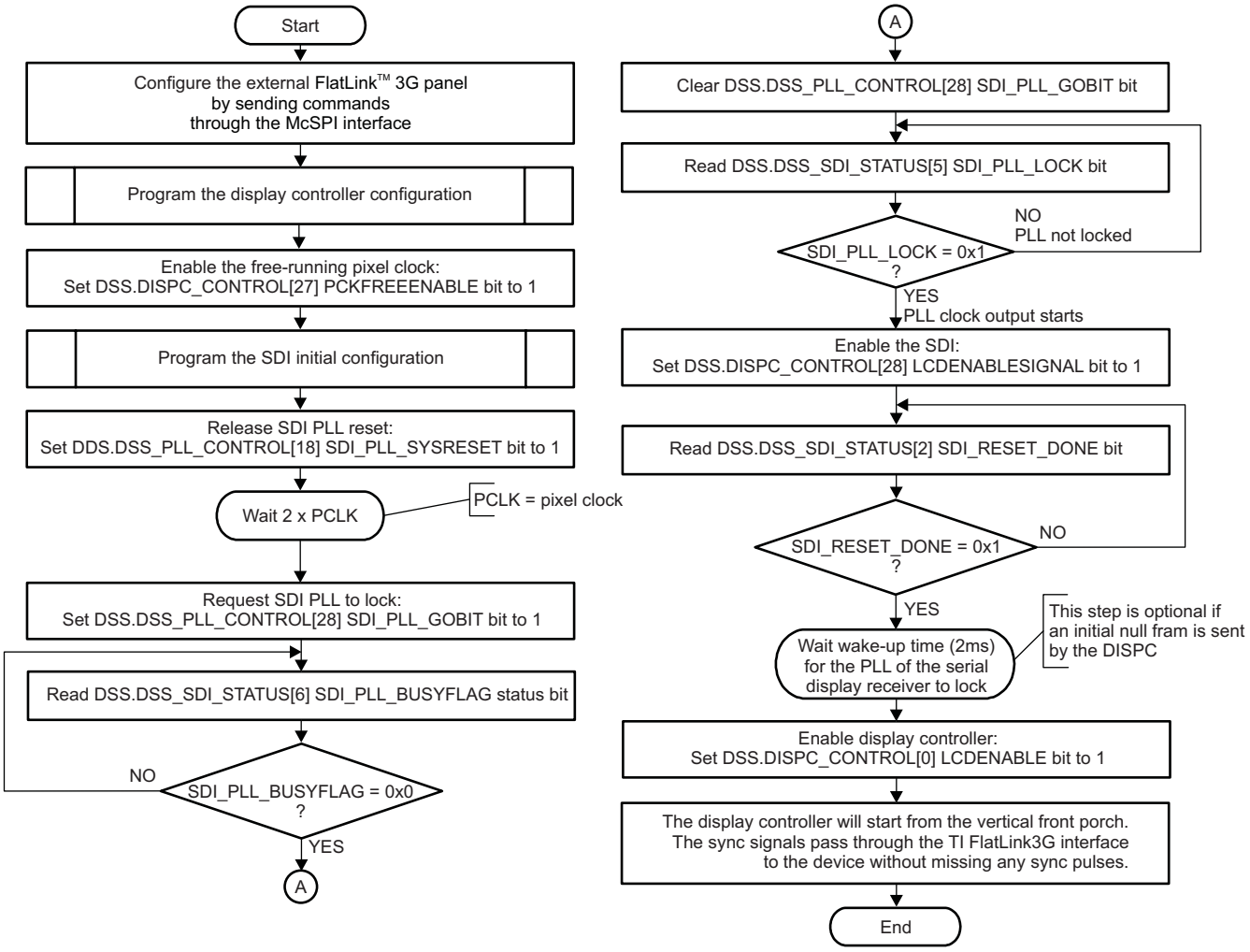
1. Power down the SDI PLL by setting the DSS.DSS_PLL_CONTROL[18]SDI_PLL_SYSRESET bit to 0x0.
2. Power-down the SDI pads:
   a. Set the CONTROL.CONTROL_PADCONF_DSS_DATA10[2:0]MUXMODE0 bit field to 0x0 to un-select SDI_DATA1N signal on dss_data10 pin and CONTROL.CONTROL_PADCONF_DSS_DATA10[18:16]MUXMODE1 bit field to 0x0 to un-select SDI_DATA1P signal on dss_data11 pin.
   b. For 2-data and 3-data pair mode: set the CONTROL.CONTROL_PADCONF_DSS_DATA12[2:0]MUXMODE0 bit field to 0x0 to un-select SDI_DATA2N on dss_data12 pin and CONTROL.CONTROL_PADCONF_DSS_DATA12[18:16]MUXMODE1 bit field to 0x0 to un-select SDI_DATA2P signal on dss_data13 pin.
   c. For 3-data pair mode: set the CONTROL.CONTROL_PADCONF_DSS_DATA14[2:0]MUXMODE0 bit field to 0x0 to un-select SDI_DATA3N on dss_data14 pin and CONTROL.CONTROL_PADCONF_DSS_DATA14[18:16]MUXMODE1 bit field to 0x0 to un-select SDI_DATA3P signal on dss_data15 pin.
   d. Set the CONTROL.CONTROL_PADCONF_DSS_DATA18[2:0]MUXMODE0 bit field to 0x0 to un-select SDI_VSYNC signal on dss_data18 pin and CONTROL.CONTROL_PADCONF_DSS_DATA18[18:16]MUXMODE1 bit field to 0x0 to un-select SDI_HSYNC signal on dss_data19 pin.
   e. Set the CONTROL.CONTROL_PADCONF_DSS_DATA20[2:0]MUXMODE0 field to 0x0 to un-select SDI_DEN signal on dss_data20 pin and CONTROL.CONTROL_PADCONF_DSS_DATA20[18:16]MUXMODE1 bit field to 0x0 to un-select SDI_STP signal on dss_data21 pin.
   f. Set the CONTROL.CONTROL_PADCONF_DSS_DATA22[2:0]MUXMODE0 bit field to 0x0 to un-select SDI_CLKP signal on dss_data22 pin and CONTROL.CONTROL_PADCONF_DSS_DATA22[18:16]MUXMODE1 bit field to 0x0 to un-select SDI_CLKN signal on dss_data23 pin.

Note: The same software sequence is needed to shut-down the DSS power domain.

15.5.9.3 SDI Start Sequence

Figure 15-149 describes the SDI start sequence.

Figure 15-149. SDI Start Sequence



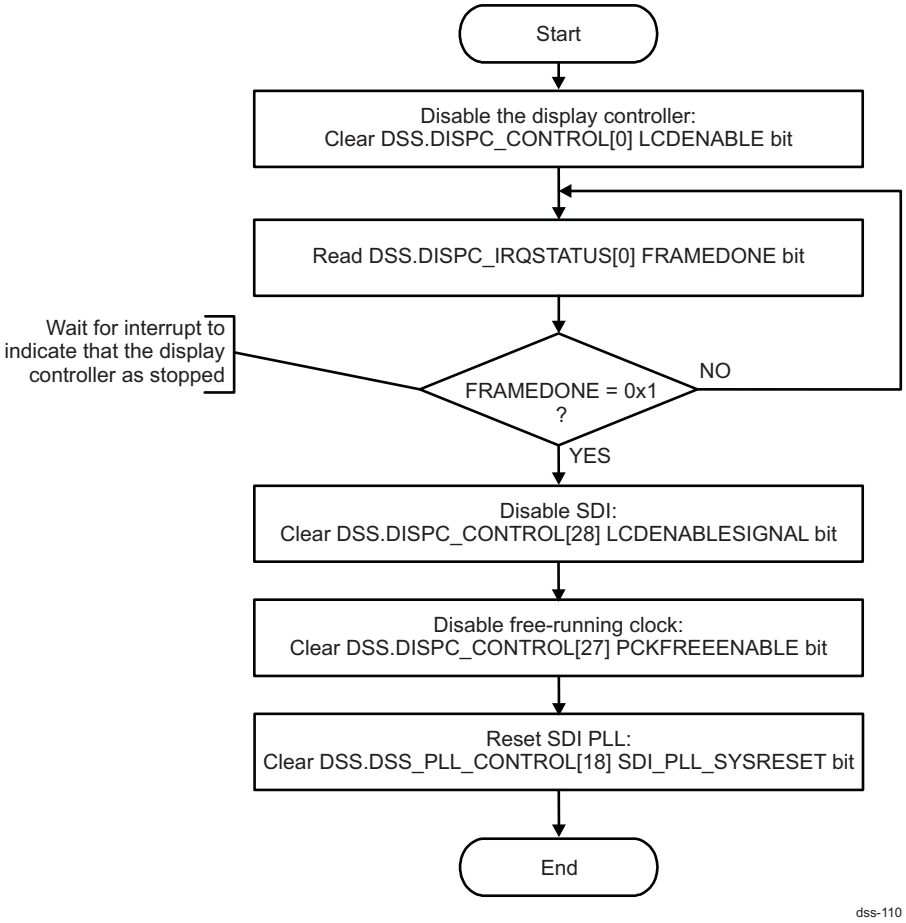
Note: The lock status of the SDI PLL is connected to the GPIO_81 signal of the GPIO3 module. To avoid polling, configure the GPIO3 module to generate an interrupt when the SDI PLL is locked (lock status is 1). For more information, see Chapter 25, *General-Purpose Interface*.

Note: TI provides a global solution with OMAP device associated to a programmable 27-bit serial display interface receiver, the SN65LVDS302 receiver. For more details on SN65LVDS302 device, see the TI public web at www.ti.com.

15.5.9.4 SDI Stop Sequence

Figure 15-150 describes the SDI stop sequence.

Figure 15-150. SDI Stop Sequence



15.5.9.5 Clock Source/Frequency Change Sequence

To change clock source or frequency, the simplest procedure is to follow the stop and start sequences documented above. If it is desired to change these without stopping the display a more rigorous procedure must be used as follows. The intent of this procedure is to perform the changes as quickly as possible during the vertical blanking interval and hence to minimize any visible display effect due to the increased blanking time.

15.5.9.5.1 Complete Sequence

Figure 15-151 and Figure 15-152 describe the clock source and frequency change sequence of the SDI.

Figure 15-151. SDI Clock Source/Frequency Change Sequence Part A

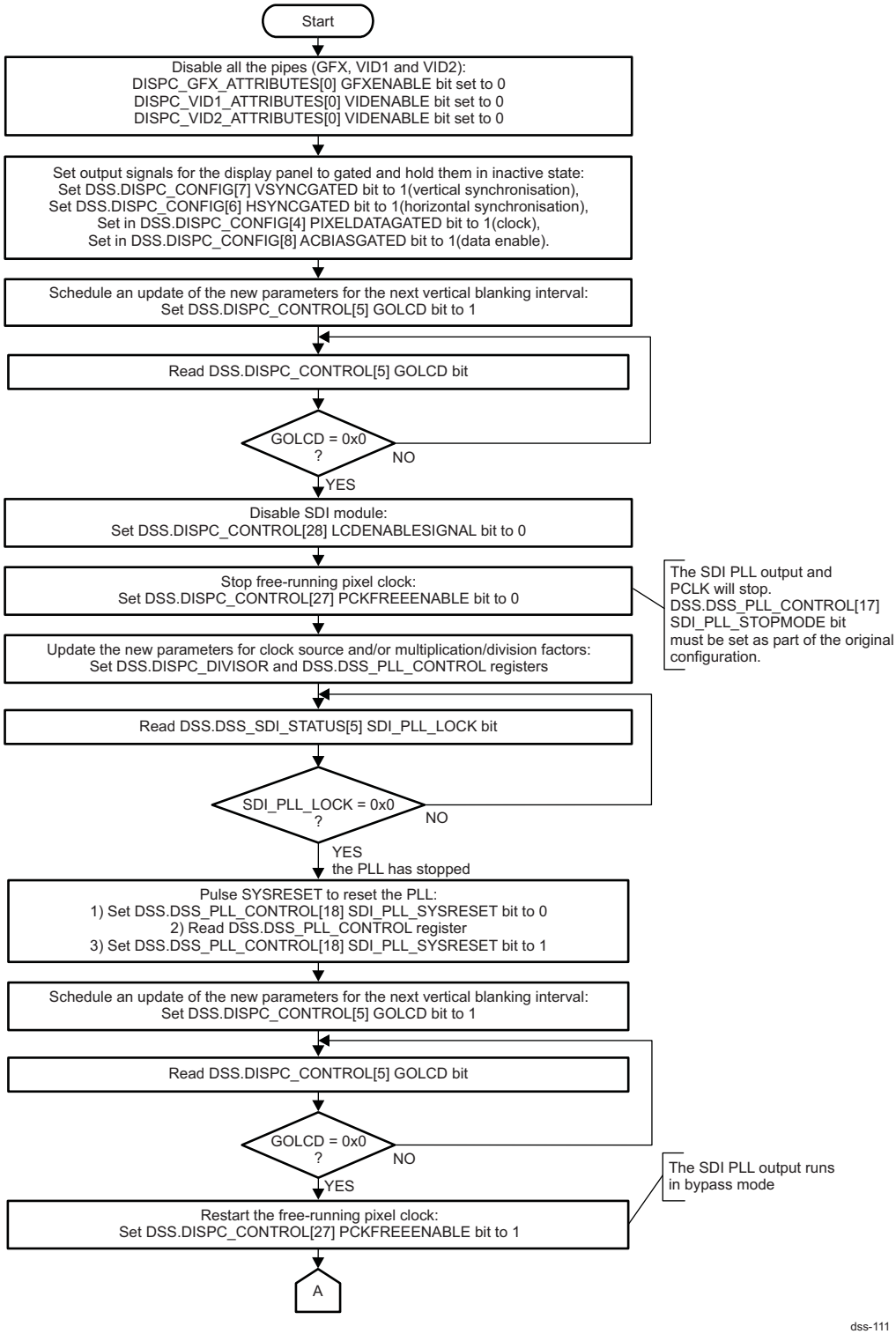
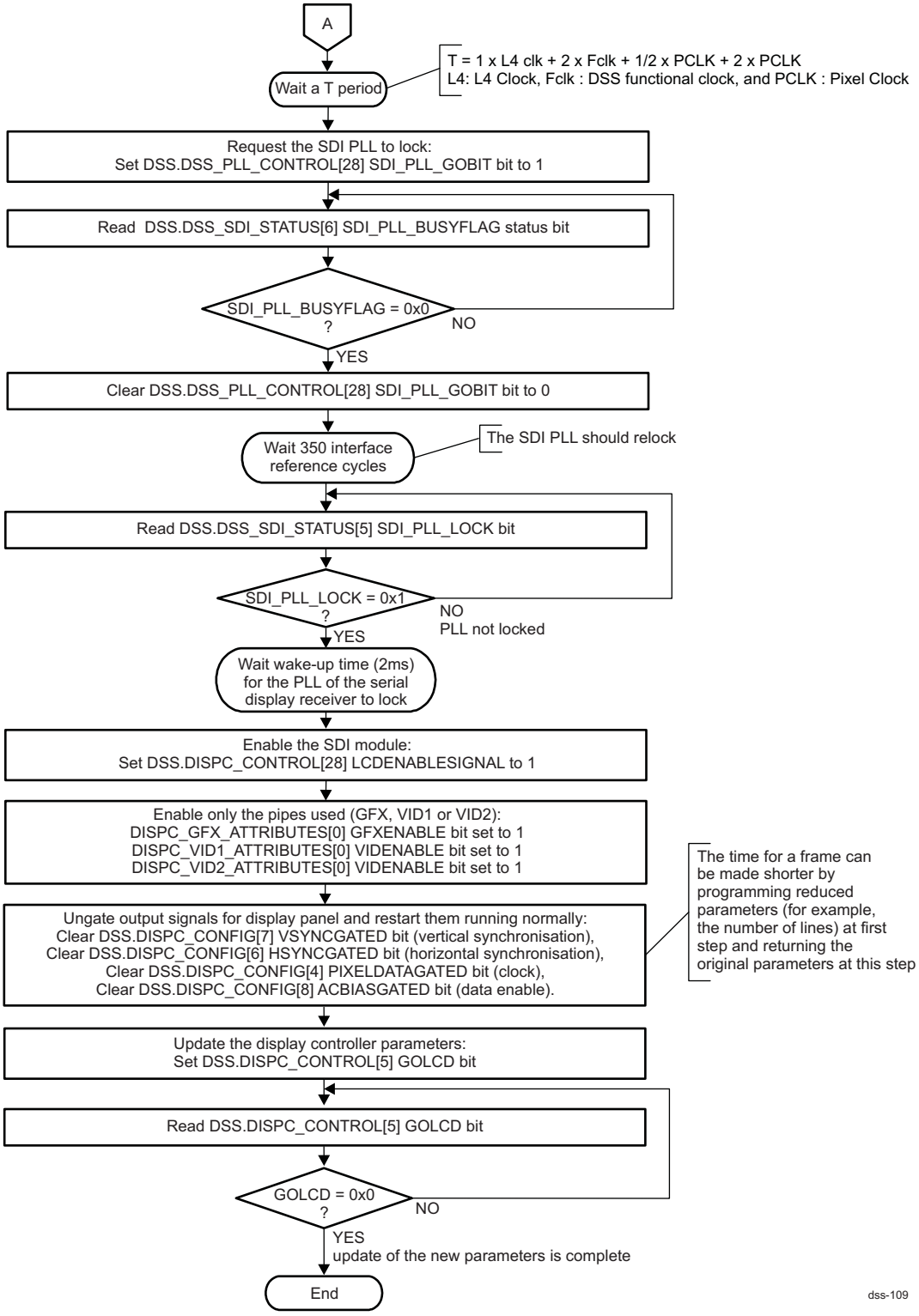


Figure 15-152. SDI Clock Source/Frequency Change Sequence Part B



Note: One interface reference cycle is defined as follows:

Pixel Clock Frequency (PCLK)/SDI_PLL_REGN[5:0]/SDI_PLL_HIGHFREQ

Assuming the DSS.DSS_PLL_CONTROL[25:22] SDI_PLL_FREQSEL bit field value is between 0xB and 0xF and the DSS.DSS_PLL_CONTROL[27:26] SDI_PLL_LOCKSEL bit field is set to 0x0.

Note: The lock status of the SDI PLL is connected to the GPIO_81 signal of the GPIO3 module. To avoid polling, configure the GPIO3 module to generate an interrupt when the SDI PLL is locked (lock status is 1). For more information, see Chapter 25, *General-Purpose Interface*.

Note: TI provides a global solution with OMAP device associated to a programmable 27-bit serial display interface receiver, the SN65LVDS302 receiver. For more details on SN65LVDS302 device, see the TI public web at www.ti.com.

15.5.9.5.2 Simplified Sequence When LCD and PCD Are Swapped

If the values in the DSS.DISPC_DIVISOR[23:16] LCD and DSS.DISPC_DIVISOR[7:0] PCD are exchanged (for example to reduce the internal display controller functional clock frequency while keeping the pixel clock frequency unchanged) with no other changes in the clock path, then there will be no discontinuity in the pixel clock. As a consequence the change will be transparent to the SDI module.

15.5.9.6 SDI Error Management

The DSS.DSS_SDI_STATUS[3] SDI_ERROR status bit indicates an internal buffer underflow/overflow. If such an error occurs, the SDI module must be disabled and enabled again to realign the buffer. This is done by setting the DSS.DISPC_CONTROL[28] LCDENABLESIGNAL bit to 0 (disabling) and to 1 (enabling). Ensure that the DSS.DSS_SDI_STATUS[2] SDI_RESET_DONE bit is set to 1. If this SDI error occurs during the debug phase, it is often due to programming incompatible values for MDIV, NDIV, and PDIV parameters for SDI PLL settings.

The SDI_ERROR internal signal is connected to the GPIO_82 internal signal of the GPIO3 module. Using the GPIO_82 signal, the GPIO3 bank generates two interrupts (GPIO3_MPU_IRQ and GPIO3_IVA2_IRQ) to alert the MPU or IVA2.2 subsystems, respectively, about any SDI error. For additional information, see Chapter 25, *General-Purpose Interface*.

15.6 Display Subsystem Use Cases and Tips

This section gives some generic use cases and tips for setting the modules of the display subsystem.

15.6.1 How to Configure the Scaling Unit in the DISPC Module

This section describes the scaling capability of the display controller (DISPC). The scaling unit is a part of the video pipeline is used when transferring pixels from the system memory (SDRAM or on-chip SRAM) to the LCD panel or the TV set. The scaling unit consists of two scaling blocks: The vertical scaling block followed by the horizontal scaling block. The input pixel format is RGB24. In case the pixel format in system memory is not RGB, the color space conversion unit in front of the scaling unit converts the YUV pixels into RGB pixels. The two scaling units are independent: Neither of them, only one, or both can be used simultaneously

15.6.1.1 Filtering

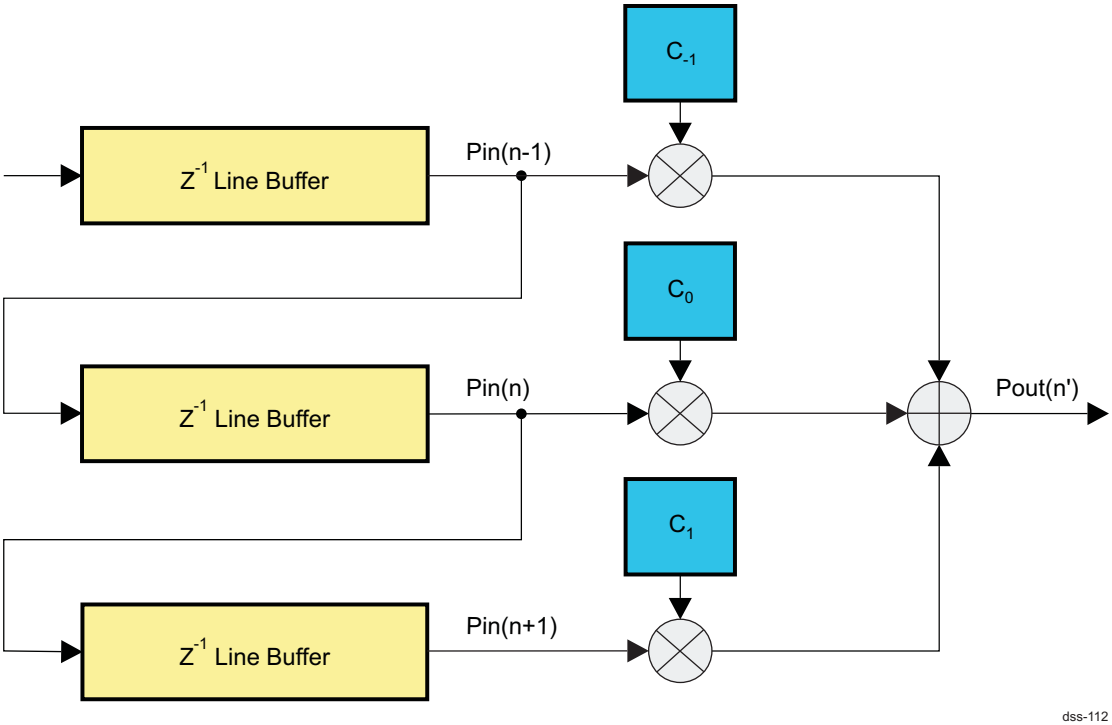
The scaling is used to down-scale, up-scale, or process the image while keeping the same size. It is applied independently horizontally and vertically. The same filtering applies for each color component (R, G, or B).

15.6.1.1.1 Vertical Filtering

The vertical filtering unit is based on a poly-phase rotation architecture with eight phases and three taps. That means that 24 coefficients are programmable

The vertical 3-tap filtering macro architecture is shown in Figure 15-153.

Figure 15-153. Vertical Filtering Macro Architecture (Three Taps)



For the 3-tap vertical up-/downsampling the equation is (with the example of R component):

$$Rout(n) = \left(\sum_{i=-1}^{i=1} Ci(\Phi) \times Rin(n+i) \right) >> 7 \quad \text{dss-E067} \qquad (15\text{-}8)$$

Legend:

Rout: R component output

Ci():Vertical FIR coefficients

Rin: R component input

The line (n+1) is older than line (n).

Note: If the 5-tap resizer is used for RGB16 and YUV422 picture formats, the width of the input picture msut be a multiple of 2 pixels and more than 5 pixels. This leads to the following register configuration:

DISPC_VIDn_ATTRIBUTES[21] VIDVERTICALTAPS == 1

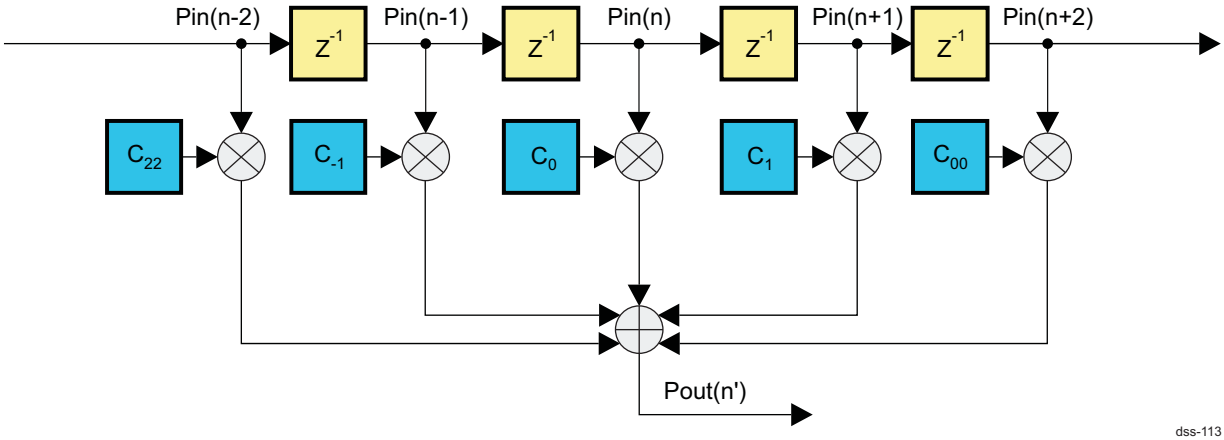
DISPC_VIDn_PICTURE_SIZE[10:0] VIDORGSIZEX > 4 and even

The programmable three coefficients of the poly-phase filters are signed 8-bit values (except for the central coefficient C_0(), which is unsigned).

The vertical filtering unit can be configured to support five taps.

The vertical 5-tap filtering macro architecture is shown in Figure 15-154.

Figure 15-154. Vertical Filtering Macro Architecture (Five Taps)



For the 5-tap vertical up/downsampling the equation is (with the example of R component):

$$Rout(n) = \left(\sum_{i=-2}^{i=2} Ci(\Phi) \times Rin(n+i) \right) >> 7 \quad \text{dss-E066} \qquad (15\text{-}9)$$

Legend:

Rout: R component output

Ci():Vertical FIR coefficients with $C_{+2}() = C_{00}()$ and $C_{-2}() = C_{22}()$

Rin: R component input

The line (n+1) is older than line (n).

The programmable five coefficients of the poly-phase filters are signed 8-bit values (except for the central coefficient C0(), which is unsigned).

In case of three taps, the memory lines are merged into three lines instead of six lines (one line is used as a cache line).

The first line is duplicated to fill up the two first lines (3-tap configuration) and the three first lines (5-tap configuration).

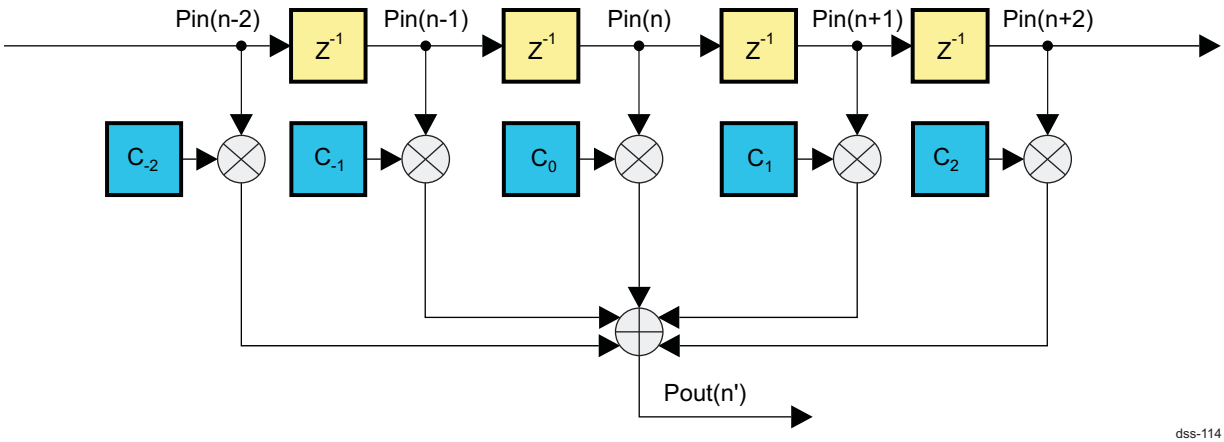
The last line is duplicated if the scaling logic requires loading of more lines and the last line has been reached

15.6.1.1.2 Horizontal Filtering

The horizontal filtering unit is based on a poly-phase rotation architecture with eight phases and five taps. That means that 40 coefficients are programmable.

The horizontal filtering macro architecture is shown in Figure 15-155.

Figure 15-155. Horizontal Filtering Macro Architecture (Five Taps)



For the 5-tap horizontal up-/downsampling, the equation is (with the example of R component):

$$Rout(n) = \left(\sum_{i=-3}^{i=3} Ci(\Phi) \times Rin(n+i) \right) >> 7 \quad \text{dss-E115} \qquad (15\text{-}10)$$

Legend:

Rout: R component output

Ci():Vertical FIR coefficients

Rin: R component input

The line (n+1) is older than line (n).

To horizontally and vertically filter the video layer, the phase is calculated separately. The programmable coefficients of the poly-phase filters are signed 8-bit values (except for the central coefficient C_0(), which is unsigned).

The first pixel is duplicated to fill up the three first pixel-buffers (5-tap configuration). The last pixel is duplicated if the scaling logic requires loading of more pixels and the last pixel has been reached

15.6.1.2 Scaling Algorithms

The up-/downsampling finite state machines (FSM) below are detailed in this section.

Figure 15-156 presents the vertical up-/downsampling FSM.

Figure 15-156. Vertical Up-/Down-Sampling Algorithm

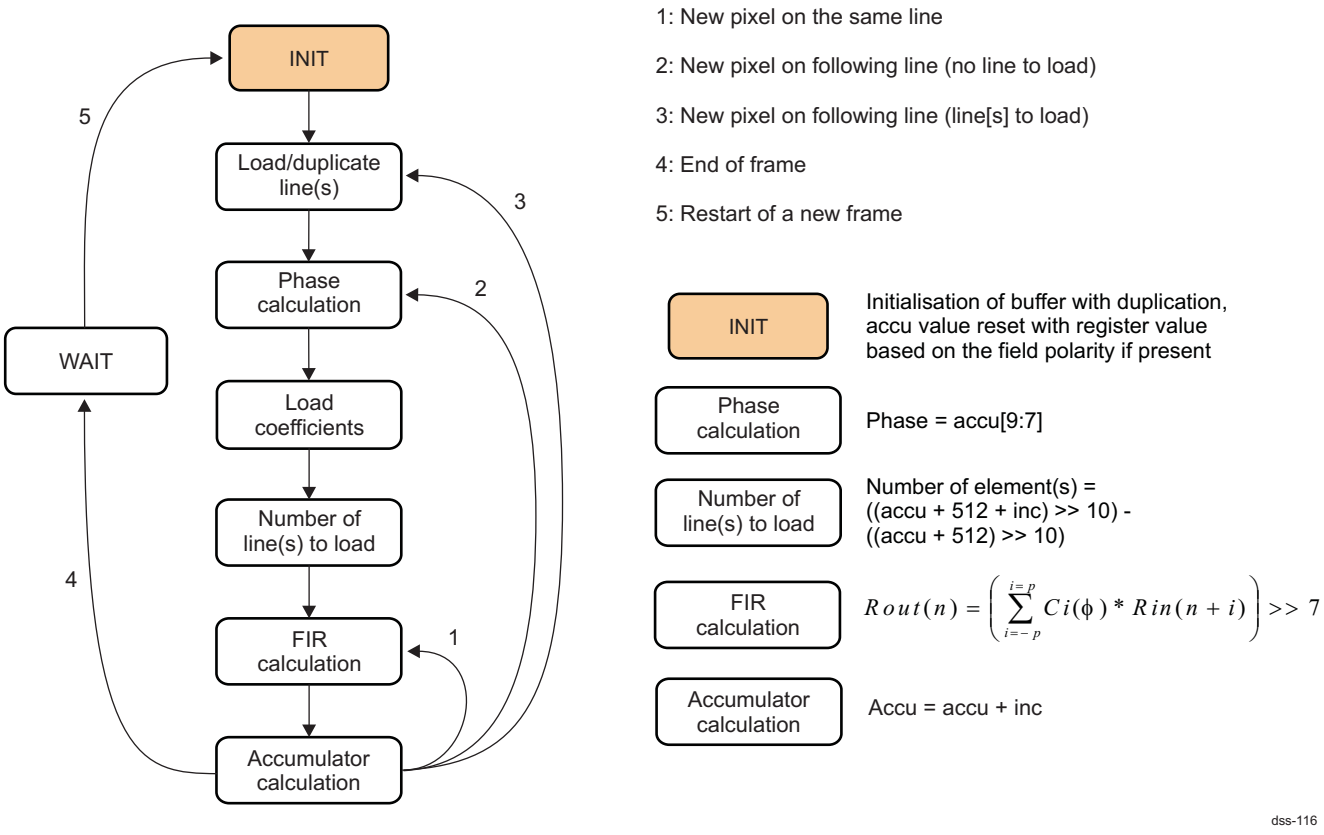
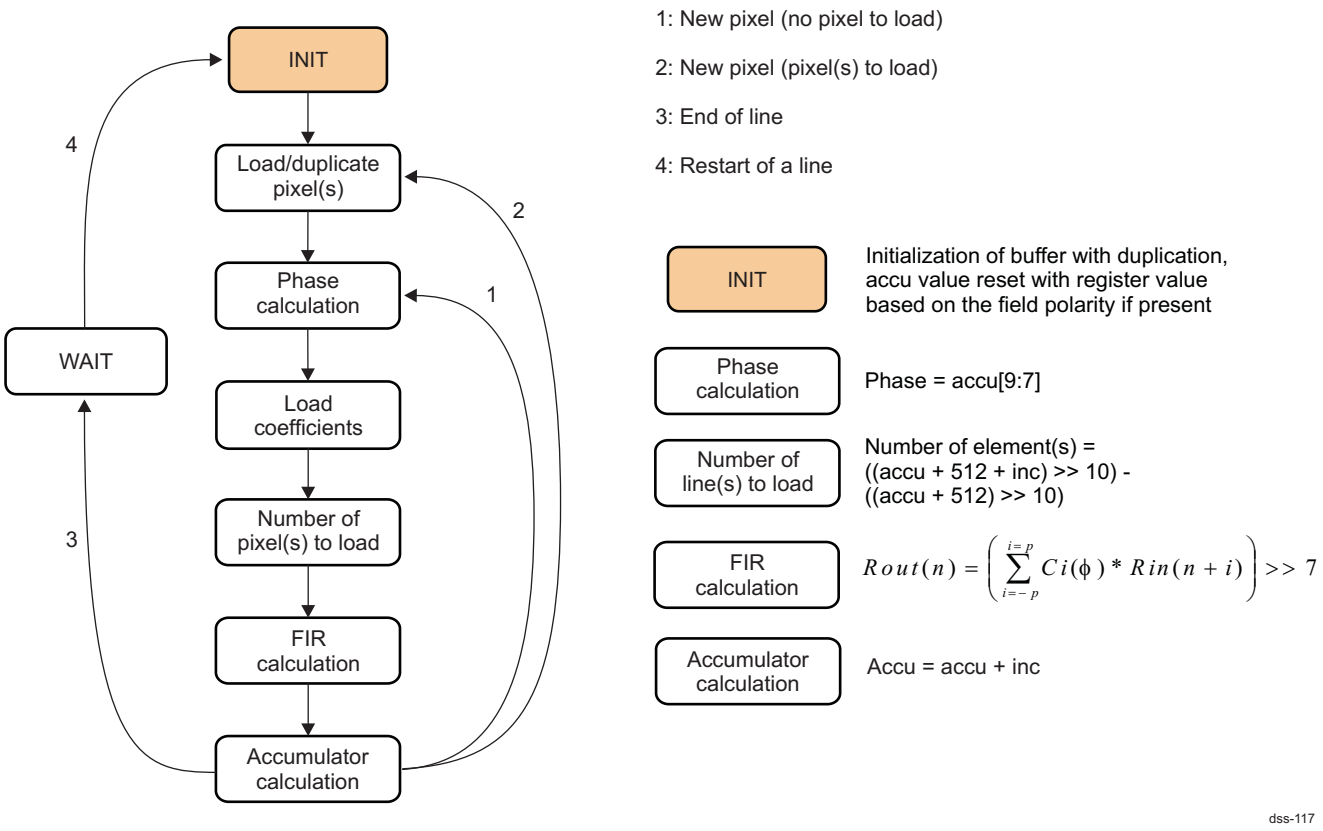


Figure 15-157 presents the horizontal up-/downsampling FSM.

Figure 15-157. Horizontal Up-/Down-Sampling Algorithm



15.6.1.3 Scaling Settings

Notes:

- In this section, the screen word refers to LCD panel or TV set.
- n indicates pipeline 0 or 1 because there are two video pipelines in the DISPC.

15.6.1.3.1 Register List

The following registers define the scaling registers for the video layer n configuration:

- DSS.DISPC_VIDn_BAj
- DSS.DISPC_VIDn_ATTRIBUTES
- DSS.DISPC_VIDn_FIR
- DSS.DISPC_VIDn_ACCUI
- DSS.DISPC_VIDn_FIR_COEF_Hi
- DSS.DISPC_VIDn_FIR_COEF_HVi
- DSS.DISPC_VIDn_FIR_COEF_Vi

Table 15-72 lists the registers for programming the vertical FIR coefficients (3-tap configuration).

Table 15-72. Vertical FIR Coefficients Corresponding Table (3-Tap Configuration)

$C_X()$	$VidFIRVC_X()$
$C_{-1}()$	$VidFIRVC_2()$

Table 15-72. Vertical FIR Coefficients Corresponding Table (3-Tap Configuration) (continued)

$C_X()$	$VidFIRVC_X()$
$C_0()$	$VidFIRVC_1()$
$C_1()$	$VidFIRVC_0()$

The corresponding registers for programming the vertical FIR coefficients (3-tap configuration) are:

- $VidFIRVC_2()$ = DSS.DISPC_VIDn_FIR_COEF_HVi[31:24] VIDFIRVC2
- $VidFIRVC_1()$ = DSS.DISPC_VIDn_FIR_COEF_HVi[23:16] VIDFIRVC1
- $VidFIRVC_0()$ = DSS.DISPC_VIDn_FIR_COEF_HVi[15:8] VIDFIRVC0

Table 15-73 lists the registers for programming the vertical FIR coefficients (5-tap configuration).

Table 15-73. Vertical FIR Coefficients Corresponding Table (5-Tap Configuration)

$C_X()$	$VidFIRVC_X()$
$C_{22}()$	$VidFIRVC_{22}()$
$C_{-1}()$	$VidFIRVC_2()$
$C_0()$	$VidFIRVC_1()$
$C_1()$	$VidFIRVC_0()$
$C_{00}()$	$VidFIRVC_{00}()$

The corresponding registers for programming the vertical FIR coefficients (5-tap configuration) are:

- $VidFIRVC_{22}()$ = DSS.DISPC_VIDn_FIR_COEF_Vi[15:8] VIDFIRVC22
- $VidFIRVC_2()$ = DSS.DISPC_VIDn_FIR_COEF_HVi[31:24] VIDFIRVC2
- $VidFIRVC_1()$ = DSS.DISPC_VIDn_FIR_COEF_HVi[23:16] VIDFIRVC1
- $VidFIRVC_0()$ = DSS.DISPC_VIDn_FIR_COEF_HVi[15:8] VIDFIRVC0
- $VidFIRVC_{00}()$ = DSS.DISPC_VIDn_FIR_COEF_Vi[7:0] VIDFIRVC00

Table 15-74 lists the registers for programming the horizontal FIR coefficients (5-tap configuration).

Table 15-74. Horizontal FIR Coefficients Corresponding Table (5-Tap Configuration)

$C_X()$	$VidFIRHC_X()$
$C_{-2}()$	$VidFIRHC_4()$
$C_{-1}()$	$VidFIRHC_3()$
$C_0()$	$VidFIRHC_2()$
$C_1()$	$VidFIRHC_1()$
$C_2()$	$VidFIRHC_0()$

The corresponding registers for programming the vertical FIR coefficients (3-tap configuration) are:

- $VidFIRHC_4()$ = DSS.DISPC_VIDn_FIR_COEF_HVi[7:0] VIDFIRHC4
- $VidFIRHC_3()$ = DSS.DISPC_VIDn_FIR_COEF_Hi[31:24] VIDFIRHC3
- $VidFIRHC_2()$ = DSS.DISPC_VIDn_FIR_COEF_Hi[23:16] VIDFIRHC2
- $VidFIRHC_1()$ = DSS.DISPC_VIDn_FIR_COEF_Hi[15:8] VIDFIRHC1
- $VidFIRHC_0()$ = DSS.DISPC_VIDn_FIR_COEF_Hi[7:0] VIDFIRHC0

15.6.1.3.2 Enabling

The video pipeline #n is enabled/disabled by setting/resetting the DSS.DISPC_VIDn_ATTRIBUTES[0].VIDENABLE bit. While the video pipeline is enabled/disabled, the video layer is visible/not visible on the screen (LCD panel or TV set).

The video up-/downsampling block for the video pipeline #n is programmed by setting the DSS.DISPC_VIDn_ATTRIBUTES[6:5] VIDRESIZEENABLE bit field:

- When the VIDRESIZEENABLE[1] bit is set to 1, the video vertical up-/downsampling block is enabled. When set to 0, the vertical resize processing is disabled.
- When the VIDRESIZEENABLE[0] bit is set to 1, the video horizontal up-/downsampling block is enabled. When set to 0, the horizontal resize processing is disabled.
- When the VIDRESIZEENABLE[1:0] is set to 0x3, both horizontal and vertical resize processing are enabled.

Notes:

- Set a valid configuration before enabling the video up-/downsampling block.
- Vertical and horizontal downsampling are limited to a 0.25 resize factor. When processing a down-scaling with a vertical factor between 0.5 and 0.25, a 5-tap filter configuration must be used. See Section 15.6.1.3.5 for more information concerning the filter coefficients.

15.6.1.3.3 Factor

The following register bit fields define the increment value of the video up-/downsampling block for video pipeline n:

- Vertical up-/downsampling increment value (DSS.DISPC_VIDn_FIR[27:16] VIDFIRVINC bit field, with n = 1 or 2): The unsigned integer value range is [1:4096]. The software calculates the value using the following equation:

$$VIDFIRVINC[11:0] = 1024 \times \frac{VIDORGSIZEY[10:0]}{VIDSIZEY[10:0]} \quad \text{dss-E118} \qquad (15\text{-}11)$$

Notes:

- If the VIDFIRVINC[11:0] bit field value is greater than 4096, it is clipped to 4096. If VIDSIZEY[10:0] equals 0x1, VIDSIZEY[10:0] is replaced by 0x2 in the previous equation.
- The VIDORGSIZEY[10:0] and VIDSIZEY[10:0] bit field values must be programmed with the value desired minus 1.

- Horizontal up-/downsampling increment value (the DSS.DISPC_VIDn_FIR[11:0] VIDFIRHINC bit field, with n = 1 or 2): The unsigned integer value range is [1:4096]. The software calculates the value using the following equation:

$$VIDFIRHINC[11:0] = 1024 \times \frac{VIDORGSIZEX[10:0]}{VIDSIZEX[10:0]} \quad \text{dss-E119} \qquad (15\text{-}12)$$

Notes:

- If the VIDFIRHINC[11:0] bit field value is greater than 4096, it is clipped to 4096. If VIDSIZEX[10:0] equals 1, VIDSIZEX[10:0] is replaced by 2 in the previous equation.
- The VIDORGSIZEX[10:0] and VIDSIZEX[10:0] bit field values must be programmed with the value desired minus 1.

15.6.1.3.4 Initial Phase

- Vertical up-/downsampling accumulator value DSS.DISPC_VIDn_ACCUI[25:16] VIDVERTICALACCU bit fields

  The unsigned integer value range is [0:1023]. The accumulator value indicates on which phase the vertical filtering starts. The value 0 indicates that the phase 0 is the first phase used by the hardware to generate the first data.
- Vertical up-/downsampling accumulator value DSS.DISPC_VIDn_ACCUI[9:0] VIDHORIZONTALACCU

bit fields

The unsigned integer value range is [0:1023]. The accumulator value indicates on which phase the horizontal filtering starts. The value 0 indicates that the phase 0 is the first phase used by the hardware to generate the first data

Table 15-75 lists the vertical/horizontal accumulator values and phases

Table 15-75. Vertical/Horizontal Accumulator Phase

Accumulator Value	Phases
0	0
128	1
256	2
384	3
512	4
640	5
768	6
896	7

Note: For LCD output, the initial phase is always 0 (horizontal and vertical.) For TV output, the vertical phases (odd and even) can be nonzero values.

15.6.1.3.5 Coefficients

- **Vertical up-/downsampling coefficients (DSS.DISPC_VIDn_FIR_COEF_HVi and DSS.DISPC_VIDn_FIR_COEF_V)**

  The 3-tap vertical up-/downsampling coefficients are defined in DSS.DISPC_VIDn_FIR_COEF_HVi registers. There are eight registers for the eight phases with three coefficients for each of them so a total of 24 programmable coefficients for the vertical up-/downsampling block. Each register contains two 8-bit signed coefficients and one 8-bit unsigned coefficient (central one).

  In addition, there are 2-tap vertical up-/downsampling coefficients defined in DSS.DISPC_VIDn_FIR_COEF_Vi registers. There are eight registers for the eight phases with two coefficients for each of them so a total of 16 programmable coefficients for the vertical up-/downsampling block used in addition of the 3-tap registers defined above. Each register contains two 8-bit signed coefficients (C_{22}() and C_{00}()).

  In case of 5-tap configuration, both sets of registers, DSS.DISPC_VIDn_FIR_COEF_HVi and DSS.DISPC_VIDn_FIR_COEF_V, are used. In case of 3-tap configuration, only one set of registers, DSS.DISPC_VIDn_FIR_COEF_HV, is used.

- **Horizontal up-/downsampling coefficients (DSS.DISPC_VIDn_FIR_COEF_Hi and DSS.DISPC_VIDn_FIR_COEF_HV)**

  The 5-tap horizontal up-/downsampling coefficients are defined in DSS.DISPC_VIDn_FIR_COEF_Hi and DSS.DISPC_VIDn_FIR_COEF_HVi registers. There are eight registers for the eight phases with five coefficients for each register, for a total of 40 programmable coefficients for the horizontal up-/downsampling block.

  Each DSS.DISPC_VIDn_FIR_COEF_Hi register contains three 8-bit signed coefficients and one 8-bit unsigned coefficient (central one), and each DSS.DISPC_VIDn_FIR_COEF_HVi contains one 8-bit signed coefficient.

Table 15-76 through Table 15-81 give the programmable coefficients for the FIR up/downsampling filters (Max-Fauque-Berthier method).

15.6.1.3.5.1 Up-Sampling

Table 15-76 gives the 24 coefficients to program the vertical upsampling (3-tap configuration).

Table 15-76. Up-Sampling Vertical Filter Coefficients (Three Taps)

Phases	VidFIRVC$_2$()	VidFIRVC$_1$()	VidFIRVC$_0$()
0	0	128	0
1	3	123	2
2	12	111	5
3	32	89	7
4	0	64	64
5	7	89	32
6	5	111	12
7	2	123	3

Table 15-77 gives the 40 coefficients to program the vertical upsampling (5-tap configuration).

Table 15-77. Up-Sampling Vertical Filter Coefficients (Five Taps)

Phases	VidFIRVC$_{22}$()	VidFIRVC$_2$()	VidFIRVC$_1$()	VidFIRVC$_0$()	VidFIRVC$_{00}$()
0	0	0	128	0	0
1	-1	13	124	-8	0
2	-2	30	112	-11	-1
3	-5	51	95	-11	-2
4	0	-9	73	73	-9
5	-2	-11	95	51	-5
6	-1	-11	112	30	-2
7	0	-8	124	13	-1

Table 15-78 gives the 40 coefficients to program the horizontal upsampling (5-tap configuration).

Table 15-78. Up-Sampling Horizontal Filter Coefficients (Five Taps)

Phases	VidFIRHC$_4$()	VidFIRHC$_3$()	VidFIRHC$_2$()	VidFIRHC$_1$()	VidFIRHC$_0$()
0	0	0	128	0	0
1	-1	13	124	-8	0
2	-2	30	112	-11	-1
3	-5	51	95	-11	-2
4	0	-9	73	73	-9
5	-2	-11	95	51	-5
6	-1	-11	112	30	-2
7	0	-8	124	13	-1

The upsampling coefficients register configuration (vertical three taps and horizontal five taps) is the following:

- DSS.DISPC_VIDn_FIR_COEF_H0 = 0x00800000
- DSS.DISPC_VIDn_FIR_COEF_HV0 = 0x00800000
- DSS.DISPC_VIDn_FIR_COEF_H1 = 0x0D7CF800
- DSS.DISPC_VIDn_FIR_COEF_HV1 = 0x037B02FF
- DSS.DISPC_VIDn_FIR_COEF_H2 = 0x1E70F5FF
- DSS.DISPC_VIDn_FIR_COEF_HV2 = 0x0C6F05FE
- DSS.DISPC_VIDn_FIR_COEF_H3 = 0x335FF5FE
- DSS.DISPC_VIDn_FIR_COEF_HV3 = 0x205907FB
- DSS.DISPC_VIDn_FIR_COEF_H4 = 0xF74949F7
- DSS.DISPC_VIDn_FIR_COEF_HV4 = 0x00404000
- DSS.DISPC_VIDn_FIR_COEF_H5 = 0xF55F33FB

- DSS.DISPC_VIDn_FIR_COEF_HV5 = 0x075920FE
- DSS.DISPC_VIDn_FIR_COEF_H6 = 0xF5701EFE
- DSS.DISPC_VIDn_FIR_COEF_HV6 = 0x056F0CFF
- DSS.DISPC_VIDn_FIR_COEF_H7 = 0xF87C0DFF
- DSS.DISPC_VIDn_FIR_COEF_HV7 = 0x027B0300

Note: In this case, the DSS.DISPC_VIDn_FIR_COEF_Vi registers are not used.

The upsampling coefficients register configuration (both vertical and horizontal five taps) is the following:

- DSS.DISPC_VIDn_FIR_COEF_H0 = 0x00800000
- DSS.DISPC_VIDn_FIR_COEF_HV0 = 0x00800000
- DSS.DISPC_VIDn_FIR_COEF_V0 = 0x00000000
- DSS.DISPC_VIDn_FIR_COEF_H1 = 0x0D7CF800
- DSS.DISPC_VIDn_FIR_COEF_HV1 = 0x0D7CF8FF
- DSS.DISPC_VIDn_FIR_COEF_V1 = 0x0000FF00
- DSS.DISPC_VIDn_FIR_COEF_H2 = 0x1E70F5FF
- DSS.DISPC_VIDn_FIR_COEF_HV2 = 0x1E70F5FE
- DSS.DISPC_VIDn_FIR_COEF_V2 = 0x0000FEFF
- DSS.DISPC_VIDn_FIR_COEF_H3 = 0x335FF5FE
- DSS.DISPC_VIDn_FIR_COEF_HV3 = 0x335FF5FB
- DSS.DISPC_VIDn_FIR_COEF_V3 = 0x0000FBFE
- DSS.DISPC_VIDn_FIR_COEF_H4 = 0xF74949F7
- DSS.DISPC_VIDn_FIR_COEF_HV4 = 0xF7404000
- DSS.DISPC_VIDn_FIR_COEF_V04 = 0x000000F7
- DSS.DISPC_VIDn_FIR_COEF_H5 = 0xF55F33FB
- DSS.DISPC_VIDn_FIR_COEF_HV5 = 0xF55F33FE
- DSS.DISPC_VIDn_FIR_COEF_V5 = 0x0000FEFB
- DSS.DISPC_VIDn_FIR_COEF_H6 = 0xF5701EFE
- DSS.DISPC_VIDn_FIR_COEF_HV6 = 0xF5701EFF
- DSS.DISPC_VIDn_FIR_COEF_V6 = 0x0000FFFE
- DSS.DISPC_VIDn_FIR_COEF_H7 = 0xF87C0DFF
- DSS.DISPC_VIDn_FIR_COEF_HV7 = 0xF87C0D00
- DSS.DISPC_VIDn_FIR_COEF_V7 = 0x000000FF

15.6.1.3.5.2 Down-Sampling

Table 15-79 gives the 24 coefficients to program the vertical downsampling (3-tap configuration).

Table 15-79. Down-Sampling Vertical Filter Coefficients (Three Taps)

Phases	$VidFIRVC_2()$	$VidFIRVC_1()$	$VidFIRVC_0()$
0	36	56	36
1	40	57	31
2	45	56	27
3	50	55	23
4	18	55	55
5	23	55	50
6	27	56	45
7	31	57	40

Table 15-80 gives the 40 coefficients to program the vertical downsampling (5-tap configuration).

Table 15-80. Down-Sampling Vertical Filter Coefficients (Five Taps)

Phases	VidFIRVC$_{22}$()	VidFIRVC$_{2}$()	VidFIRVC$_{1}$()	VidFIRVC$_{0}$()	VidFIRVC$_{00}$()
0	0	36	56	36	0
1	4	40	55	31	-2
2	8	44	54	27	-5
3	-12	48	53	22	-7
4	-9	17	52	51	17
5	-7	22	53	48	12
6	-5	27	54	44	8
7	-2	31	55	40	4

Table 15-81 gives the 40 coefficients to program the horizontal downsampling (5-tap configuration).

Table 15-81. Down-Sampling Horizontal Filter Coefficients (Five Taps)

Phases	VidFIRHC$_{4}$()	VidFIRHC$_{3}$()	VidFIRHC$_{2(}$)	VidFIRHC$_{1(}$)	VidFIRHC$_{0}$()
0	0	36	56	36	0
1	4	40	55	31	-2
2	8	44	54	27	-5
3	-12	48	53	22	-7
4	-9	17	52	51	17
5	-7	22	53	48	12
6	-5	27	54	44	8
7	-2	31	55	40	4

The downsampling coefficients register configuration (vertical three taps and horizontal five taps) is the following:

- DSS.DISPC_VIDn_FIR_COEF_H0 = 0x24382400
- DSS.DISPC_VIDn_FIR_COEF_HV0 = 0x24382400
- DSS.DISPC_VIDn_FIR_COEF_H1 = 0x28371FFE
- DSS.DISPC_VIDn_FIR_COEF_HV1 = 0x28391F04
- DSS.DISPC_VIDn_FIR_COEF_H2 = 0x2C361BFB
- DSS.DISPC_VIDn_FIR_COEF_HV2 = 0x2D381B08
- DSS.DISPC_VIDn_FIR_COEF_H3 = 0x303516F9
- DSS.DISPC_VIDn_FIR_COEF_HV3 = 0x3237170C
- DSS.DISPC_VIDn_FIR_COEF_H4 = 0x11343311
- DSS.DISPC_VIDn_FIR_COEF_HV4 = 0x123737F7
- DSS.DISPC_VIDn_FIR_COEF_H5 = 0x1635300C
- DSS.DISPC_VIDn_FIR_COEF_HV5 = 0x173732F9
- DSS.DISPC_VIDn_FIR_COEF_H6 = 0x1B362C08
- DSS.DISPC_VIDn_FIR_COEF_HV6 = 0x1B382DFB
- DSS.DISPC_VIDn_FIR_COEF_H7 = 0x1F372804
- DSS.DISPC_VIDn_FIR_COEF_HV7 = 0x1F3928FE

Notes:

- In this case, the DSS.DISPC_VIDn_FIR_COEF_Vi registers are not used.
- In this case, the downsampling factor must be higher than 1/2.

The downsampling coefficients register configuration (both the vertical and the horizontal five taps) is the following:

- DSS.DISPC_VIDn_FIR_COEF_H0 = 0x24382400
- DSS.DISPC_VIDn_FIR_COEF_HV0 = 0x24382400
- DSS.DISPC_VIDn_FIR_COEF_V0 = 0x00000000
- DSS.DISPC_VIDn_FIR_COEF_H1 = 0x28371FFE
- DSS.DISPC_VIDn_FIR_COEF_HV1 = 0x28371F04
- DSS.DISPC_VIDn_FIR_COEF_V1 = 0x000004FE
- DSS.DISPC_VIDn_FIR_COEF_H2 = 0x2C361BFB
- DSS.DISPC_VIDn_FIR_COEF_HV2 = 0x2C361B08
- DSS.DISPC_VIDn_FIR_COEF_V2 = 0x000008FB
- DSS.DISPC_VIDn_FIR_COEF_H3 = 0x303516F9
- DSS.DISPC_VIDn_FIR_COEF_HV3 = 0x3035160C
- DSS.DISPC_VIDn_FIR_COEF_V3 = 0x00000CF9
- DSS.DISPC_VIDn_FIR_COEF_H4 = 0x11343311
- DSS.DISPC_VIDn_FIR_COEF_HV4 = 0x113433F7
- DSS.DISPC_VIDn_FIR_COEF_V4 = 0x0000F711
- DSS.DISPC_VIDn_FIR_COEF_H5 = 0x1635300C
- DSS.DISPC_VIDn_FIR_COEF_HV5 = 0x163530F9
- DSS.DISPC_VIDn_FIR_COEF_V5 = 0x0000F90C
- DSS.DISPC_VIDn_FIR_COEF_H6 = 0x1B362C08
- DSS.DISPC_VIDn_FIR_COEF_HV6 = 0x1B362CFB
- DSS.DISPC_VIDn_FIR_COEF_V6 = 0x0000FB08
- DSS.DISPC_VIDn_FIR_COEF_H7 = 0x1F372804
- DSS.DISPC_VIDn_FIR_COEF_HV7 = 0x1F3728FE
- DSS.DISPC_VIDn_FIR_COEF_V7 = 0x0000FE04

Note: This configuration must be used for vertical downsampling factors between 1/2 and 1/4

15.6.2 Display Low-Power Refresh Settings

This section describes the display low-power refresh application on the device. The display subsystem remains active while saving power by putting unused power domains and unused modules into idle mode. This process can be expanded to include the screen saver mode in which the MPU subsystem wakes up to update the frame buffer and then returns to idle mode. On an OMAP platform, where power consumption is of high importance, the display modes must be configured properly to achieve optimal power savings

The display low-power refresh mode can be used in the following scenarios:

- During the period of time when there is no application running and the backlight turns off.
- Once the backlight turns off, the LCD display can be shut off or can be refreshed showing the time and date. The screen saver mode can be used to update the time every minute.

This section discusses the methodology for finding optimal power savings. These settings are detailed for a 16-bit, 240 x 320 pixel QVGA LCD.

15.6.2.1 Display Low-Power Refresh Overview

When the device is not in idle mode, meaning all clocks are on and the power is applied to all power domains, the following activity typically occurs with respect to the display subsystem:

- The MPU subsystem is processing
- The display subsystem DMA controller is moving data from the SDRAM frame buffer location to the display subsystem internal FIFO.

- The LCD data is being sent from the internal FIFO to the display panel.

When the MPU goes into idle mode, the following activity occurs:

- The display subsystem DMA controller remains active, moving data from the SDRAM frame buffer to the internal FIFO.
- The SDRAM will go in and out of self-refresh between transfers.
- The display subsystem internal FIFO will continue to send LCD data to the display panel.

This procedure is named as the display low-power refresh scenario.

Note: In the device, the display subsystem has its own power domain (the DSS power domain).

15.6.2.2 Display Subsystem Clock

15.6.2.2.1 Display Subsystem Clock Configuration

The display subsystem contains two possible functional clock sources, DSS functional clock 1 (DSS1_ALWON_FCLK) and DSI PLL functional clock 1 (DSI1_PLL_FCLK):

- DSS1_ALWON_FCLK is sourced from DPLL4 (DPLL4_ALWON_FCLK), with several multipliers available and is configured in the PRCM.CM_CLKSEL_DSS[4:0] CLKSEL_DSS1 bit field.
- DSI1_PLL_FCLK is one of the two output clocks from the DSI PLL.

The pixel clock is set as either DSS1_ALWON_FCLK or DSI1_PLL_FCLK by configuring the DSS.DSS_CONTROL[0] DISPC_CLK_SWITCH bit

Note: When the DSI PLL is in bypass mode, the DSI1_PLL_FCLK clock is the DSS2_ALWON_FCLK clock. This ensures the backward compatibility with OMAP2 devices.

The LCD logic clock is determined by the DSS.DISPC_DIVISOR[23:16] LCD bit field. This divisor is used on the DSS functional clock that is selected in the DSS_CONTROL register (either DSS1_ALWON_FCLK or DSS2_ALWON_FCLK). This LCD divisor selects the logical clock frequency which is used to clock the logic in the display subsystem. For some applications there is a required minimum logical clock frequency. The lower the logical clock frequency then the lower the power consumption.

The pixel clock is determined by setting the DSS.DISPC_DIVISOR[7:0] PCD bit field. This divisor is used on the LCD logic clock.

In the following example, the DPLL4 clock (DPLL4_ALWON_FCLK) is enabled and running at 266 MHz:

The PRCM.CM_CLKSEL2_PLL[18:8] PERIPH_DPLL_MULT bit field is set to 0x4 and the PRCM.CM_CLKSEL2_PLL[6:0] PERIPH_DPLL_DIV bit field is set to 0x1 (DPLL4 x 4/(1+1)):

DPLL4_ALWON_FCLKOUTX2= DPLL4_ALWON_FCLK(266MHz) x 4/2 = 532 MHz

Note: The DPLL4_ALWON_FCLOUTX2 clock is an internal clock in DPLL4 module after the DPLL_MULT and DPLL_DIV stages. The DPLL4_M4X2_CLK clock is one of the DPLL4 output clocks and is the clock source for DSS1_ALWON_FCLK.

The DSS.DSS_CONTROL[0] DSS_CLK_SWITCH bit is set to 0x0 to select DSS1_ALWON_FCLK as the display subsystem functional clock:

DSS1_ALWON_FCLK = DPLL4_M4X2_CLK

The PRCM.CM_CLKSEL_DSS[4:0] CLKSEL_DSS1 bit field is set to 0x08:

$$\text{DSS1_ALWON_FCLK} = \frac{\text{DPLL4_ALWON_FCLKOUTX2}\,(532\text{MHz})}{8} = 66.5\ \text{MHz} \quad \text{dss-E120} \qquad (15\text{-}13)$$

The DSS.DISPC_DIVISOR[23:16] LCD bit field is set to 0x01:

$$\text{LogicClock} = \frac{\text{DSS1_ALWON_FCLK(66.5MHz)}}{1} = 66.5 \text{ Mhz} \quad \text{dss-E121} \tag{15-14}$$

The DSS.DISPC_DIVISOR[6:0] PCD bit field is set to 0x0C:

$$\text{PixelClock} = \frac{\text{LogicClock(66.5MHz)}}{12} = 5.54 \text{ Mhz} \quad \text{dss-E122} \tag{15-15}$$

15.6.2.2.1.1 Pixel Clock Frequency Settings to Reduce Power Consumption

Power consumption is reduced when a low pixel clock frequency is used. If the clock frequency is set too low, however, the frames-per-second (FPS) are reduced. This can result in visible flickering on the screen each time the screen is refreshed. To avoid electrical polarization problems, refer to the appropriate LCD panel datasheet to determine the maximum range of pixel clock frequency variation. To save power, therefore, the pixel clock frequency during low-power mode must be set as low as possible, but high enough to eliminate visible flickering

15.6.2.2.1.2 Display Subsystem Divider Settings to Reduce Power Consumption

The pixel clock is determined by the DSS.DISPC_DIVISOR[7:0] PCD and DSS.DISPC_DIVISOR[23:16] LCD settings. In most cases, the LCD[7:0] bit field is set to 0x1 and the PCD[7:0] bit field is used as the main divider. To reduce power consumption, software users should investigate if the DSS.DISPC_DIVISOR[23:16] LCD bit field can be set to a value other than 0x1 and then decrease DSS.DISPC_DIVISOR[7:0] PCD bit field value. For example, if the desired pixel clock is 1.625 MHz with a 13-MHz functional clock, then this pixel clock can be achieved by setting DSS.DISPC_DIVISOR[23:16] LCD to 0x1 and DSS.DISPC_DIVISOR[7:0] PCD to 0x8. The same pixel clock can be achieved by setting DSS.DISPC_DIVISOR[23:16] LCD to 0x2 and DSS.DISPC_DIVISOR[7:0] PCD to 0x4.

15.6.2.2.2 Display Subsystem Clock Enable

To take the DSS out of reset, all DSS-related clocks must be enabled, and the DPLL4 clock must be enabled. After taking the DSS out of reset, these clocks can be disabled if they are not used. The following clocks must be enabled before the DSS can come out of reset:

- PRCM.CM_FCLKEN_DSS[0] EN_DSS1 = 0x1
- PRCM.CM_FCLKEN_DSS[1] EN_DSS2 = 0x1
- PRCM.CM_FCLKEN_DSS[2] EN_TV = 0x1
- PRCM.CM_ICLKEN_DSS[0] EN_DSS = 0x1
- PRCM.CM_CLKEN_PLL[18:16] EN_PERIPH_DPLL = 0x7

Once these clocks are enabled, the display subsystem can be taken out of reset.

The following sections explain the display low-power mode configuration options, which are determined by product requirements (LCD panel type).

15.6.2.3 DPLL4 in Low-Power Mode

For optimal power savings in low-power mode, DPLL4 can be put into low-power stop mode. Thus the DSS1_ALWON_FCLK clock cut when DPLL4 is in low-power stop mode. Software users must switch from DSS1_ALWON_FCLK to DSI1_PLL_FCLK by setting the DSS.DSS_CONTROL[0] DISPC_CLK_SWITCH bit to 0x1.

Note: Before switching to DSI1_PLL_FCLK, the DSI PLL and the HS divider must be programmed and the DSI PLL must be locked.

DPLL4 can be put in low-power stop mode in either of the following methods:

- Manually: When setting the PRCM.CM_CLKEN_PLL[18:16] EN_PERIPH_DPLL bit to 0x1.

- Automatically: When setting the PRCM.CM_AUTOIDLE_PLL[5:3] AUTO_PERIPH_DPLL bit field to 0x1. DPLL4 is automatically put in low-power stop mode when none of the 96- MHz and 54- MHz clocks are required anymore. DPLL4 is also restarted automatically.

Software users must remember to change the clock configuration after enabling DPLL4 when leaving low-power mode by setting the DSS.DSS_CONTROL[0] DISPC_CLK_SWITCH bit to 0x0. The DSS1_ALWON_FCLK clock will be selected.

Lock time must be considered before disabling the DSS1_ALWON_FCLK clock.

15.6.2.4 Autoidle and Smart Idle

15.6.2.4.1 Autoidle

To further save power consumption, the autoidle feature at the module can be enabled for the active modules. For example, the PRCM and the system control modules are active during this mode. By enabling the autoidle feature, the clocks at the module level are gated when they are not needed.

The RFBI, display controller, and L4 interfaces can internally gate their clocks to decrease power consumption if no transaction is present on the related bus. The following bits must be set to enable this functionality:

- DSS.DSS_SYSCONFIG[0] AUTOIDLE bit (1: Autoidle; 0: Clock free-running) for the display subsystem
- DSS.RFBI_SYSCONFIG[0] AUTOIDLE bit (1: Autoidle; 0: Clock free-running) for the RFBI
- DSS.DISPC_SYSCONFIG[0] AUTOIDLE bit (1: Autoidle; 0: Clock free-running) for the display controller
- DSS.DISPC_CONFIG[9] FUNCGATED bit (1: Functional clocks gated enabled, 0: Functional clocks gated disabled) for the display controller

15.6.2.4.2 Smart-Idle

The smart-idle feature can be enabled to allow the module to enter idle when the clocks are not needed. The smart-idle feature can be enabled for the display subsystem submodules to further save power consumption:

- Display subsystem: DSS.DSS_SYSCONFIG[4:3] SIDLEMODE
- Display controller: DSS.DISPC_SYSCONFIG[4:3] SIDLEMODE
- RFBI: DSS.RFBI_SYSCONFIG[4:3] SIDLEMODE

15.6.2.5 FIFO Thresholds

The display subsystem internal FIFO is used to move data to the LCD panel. This FIFO is filled by the display subsystem DMA controller. The DMA controller is triggered to start and stop based on two thresholds:

- DSS.DISPC_GFX_FIFO_THRESHOLD[11:0] GFXFIFOLOWTHRESHOLD
- DSS.DISPC_GFX_FIFO_THRESHOLD[27:16] GFXFIFOHIGHTHRESHOLD

When the level of the FIFO reaches the low threshold, the internal DMA controller begins to fill the FIFO With the data in the frame buffer. Once the amount of pixel data reaches the high threshold, the internal DMA controller stops.

15.6.2.5.1 FIFO Threshold Settings to Reduce Power Consumption

Power consumption is reduced by increasing the difference between the high and low FIFO threshold levels, thereby leaving the SDRAM in self-refresh for a longer period of time. To perform this reduction, consider the following:

- The low FIFO threshold level must be as low as possible, but not low enough to cause any underflow.
- The high FIFO threshold level must not exceed the FIFO size minus one burst. A value above this limit results in the DMA controller trying to fill the FIFO to a level that cannot be reached, which will increase power consumption.

- The difference between high and low FIFO threshold levels must not be less than one burst size. These settings do not reduce power consumption because the SDRAM never goes into self-refresh, but they will avoid underflow.

15.6.2.6 Vertical and Horizontal Timings

The vertical and horizontal timings and the pixel clock speed determine the number of frames updated per second. Figure 15-158 shows the timings for a 240 x 320 pixel QVGA LCD panel. If the pulse width (also called blanking) and the front porch parameters are increased, more setup time is added before the data is transferred. This additional time is beneficial for delaying the data transfer if the data is not ready because of bandwidth limitations. Care must be taken to determine the fps when modifying these parameters.

Use the following formula to determine the fps for a 240 x 320 QVGA LCD:

$$fps = \frac{1}{[(Hsw + 1) + (Hrp + 1) + 240 + (Hbp + 1)] \times [(Vsw + 1) + Vrp + 320 + Vbp)] \times (PCLK)}$$

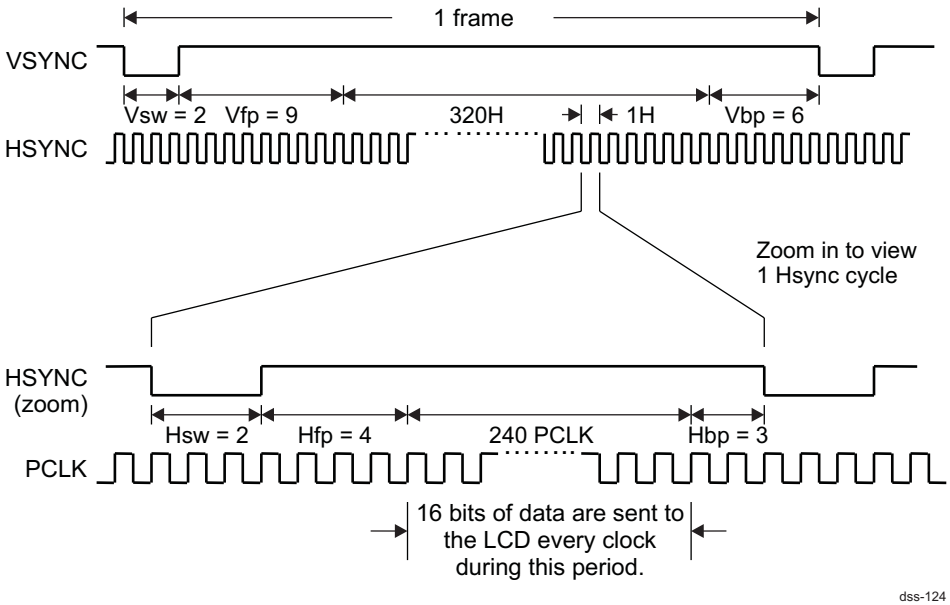
dss-E123 (15-16)

With:

- Hsw: DSS.DISPC_TIMING_H[7:0] HSW bit field value
- Hp: DSS.DISPC_TIMING_H[19:8] HFP bit field value
- Hbp: DSS.DISPC_TIMING_H[31:20] HBP bit field value
- Vsw: DSS.DISPC_TIMING_V[7:0] VSW bit field value
- Vp: DSS.DISPC_TIMING_V[19:8] VFP bit field value
- Vpb: DSS.DISPC_TIMING_V[31:20] VBP bit field value
- PCLK: Pixel clock period

The horizontal (Hsw) and vertical (Vsw) pulse widths and the horizontal front (Hp) and back (Hbp) porches are increased by 1 because the value is programmed as the desired value minus 1.

Figure 15-158. QVGA LCD Timings



The fps for the example of 6-MHz pixel clock with the setting shown in Figure 15-158 is as follows:

$$fps = \frac{1}{[(2 + 1) + 4 + 240 + 3] \times [(2 + 1) + 9 + 320 + 6] \times 166.67 \times 10^{-9}}$$

dss-E125 (15-17)

$$fps = 71Hz$$

15.6.2.6.1 Horizontal and Vertical Timing Settings to Reduce Power Consumption

The number of fps that the screen is refreshed is also determined by the vertical and horizontal timings. Consequently, longer timings between frames (blanking periods) reduce the fps and reduce average power consumption. Shorter blanking periods increase fps and increases power consumption. If the blanking between frames is too small, a FIFO underflow may occur.

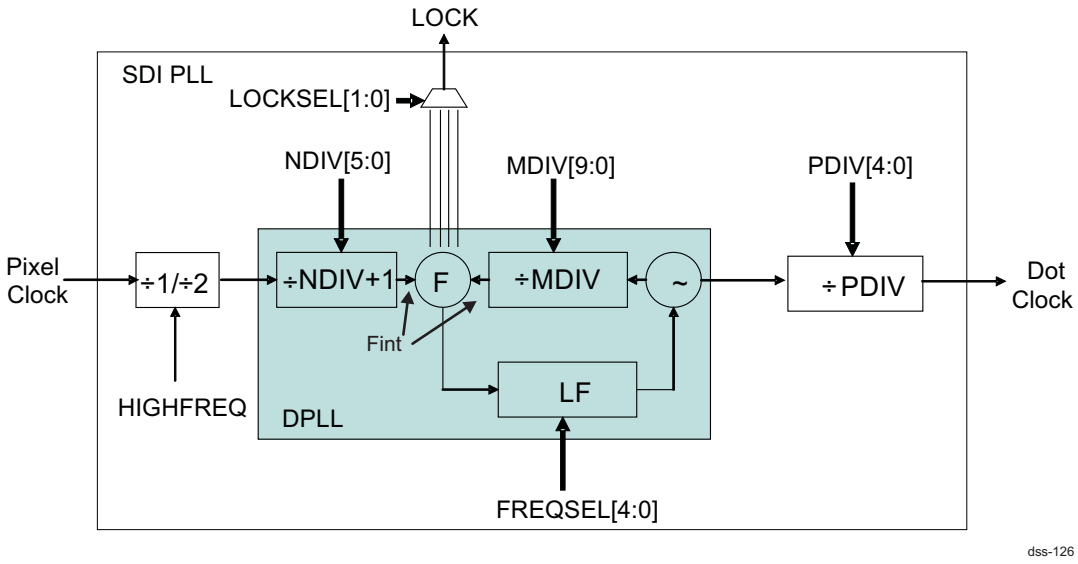
15.6.3 How to Configure the Serial Display Interface Module With FlatLink3G Protocol

This section describes the basic settings of Serial Display Interface (SDI) module and related Phase-Locked Loop (PLL).

15.6.3.1 SDI PLL Architecture

The SDI PLL architecture is based on the Digital Phase-Locked Loop (DPLL) that is described in Chapter 4, *Power, Reset, and Clock Management*. Figure 15-159 gives a functional description of the SDI PLL.

Figure 15-159. SDI PLL Architecture



Notes:

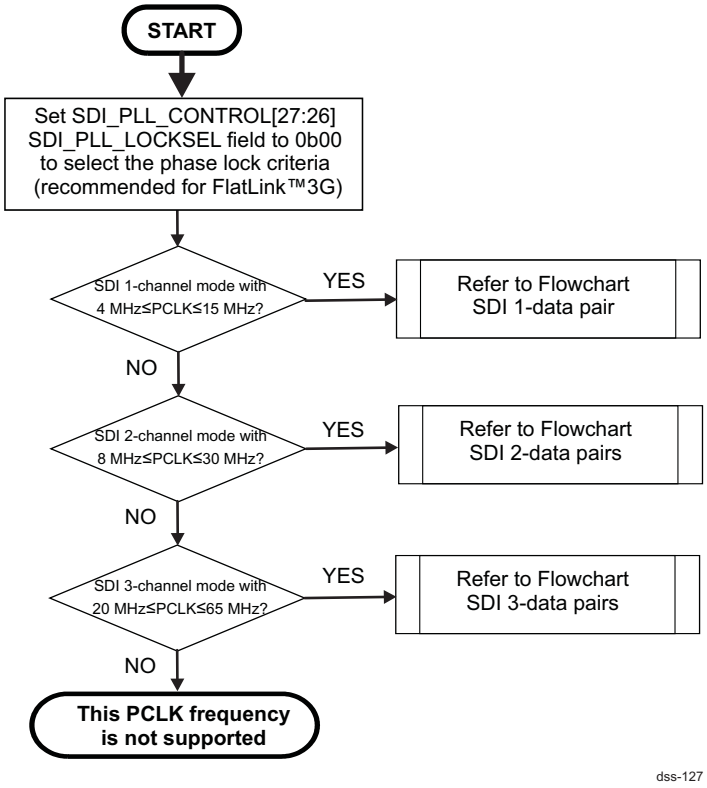
- FlatLink3G data rate is always at SCLK frequency.
- Software programmers must ensure that Pixel Clock = Dot Clock. If it is not the case the DSS.DSS_SDI_STATUS[3] SDI_ERROR status bit will be set indicating the pixel buffer overflow/underflow.

15.6.3.2 SDI PLL Configuration

The SDI PLL configuration depends on the pixel clock (PCLK), and on the number of data pairs in the FlatLink 3G protocol.

The following flowcharts detail the SDI PLL configuration from the PCLK frequency point of view. Figure 15-160 is the main flowchart. See Figure 15-161, Figure 15-162, and Figure 15-163 for details in PLL setting depending the PCLK input frequency.

Figure 15-160. Main Flowchart



Note: Compliance with the above ranges is required to meet the SN65LVDS302 receiver specifications.

Figure 15-161. Flowchart SDI 1-Data Pair

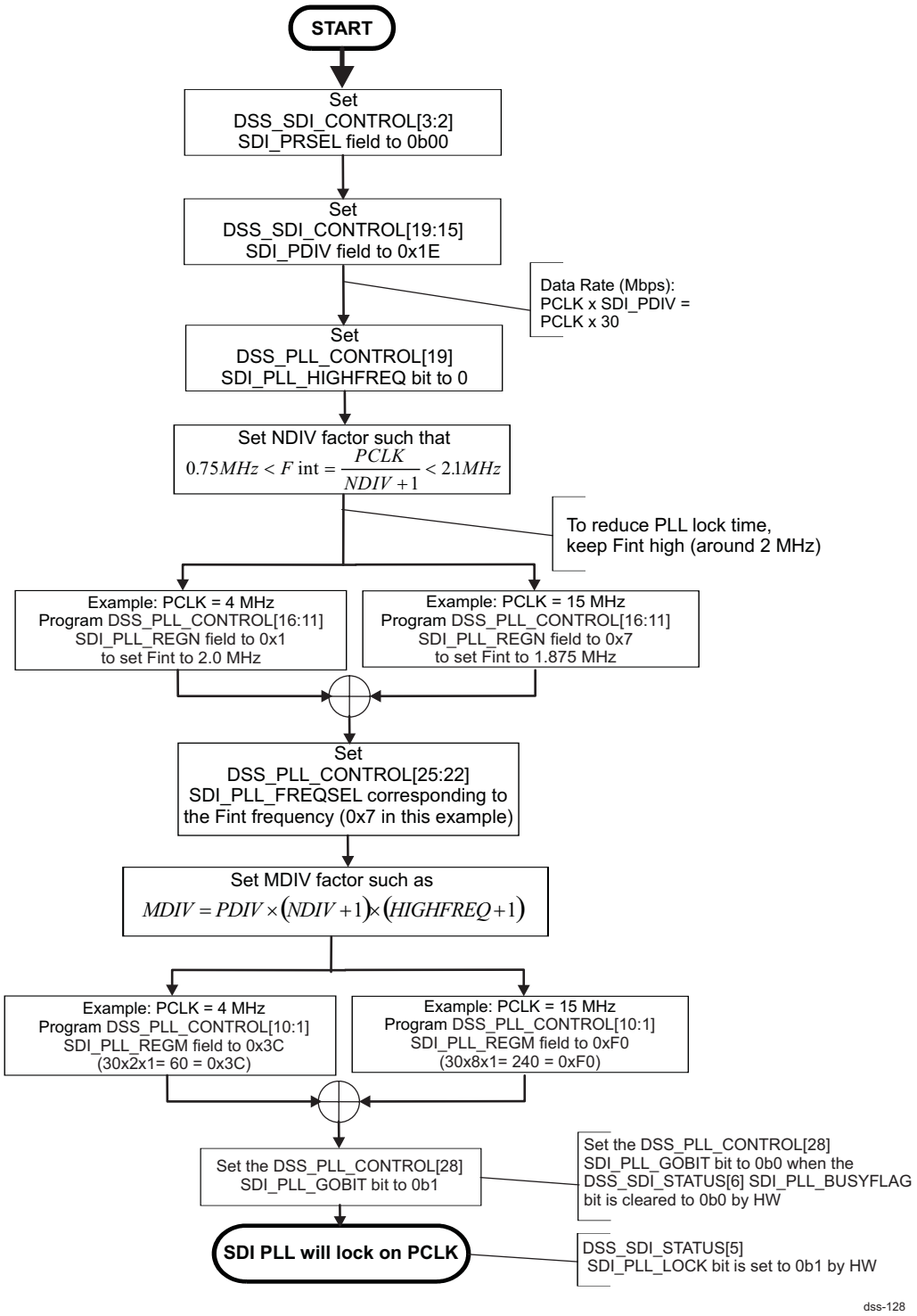


Figure 15-162. Flowchart SDI 2-Data Pairs

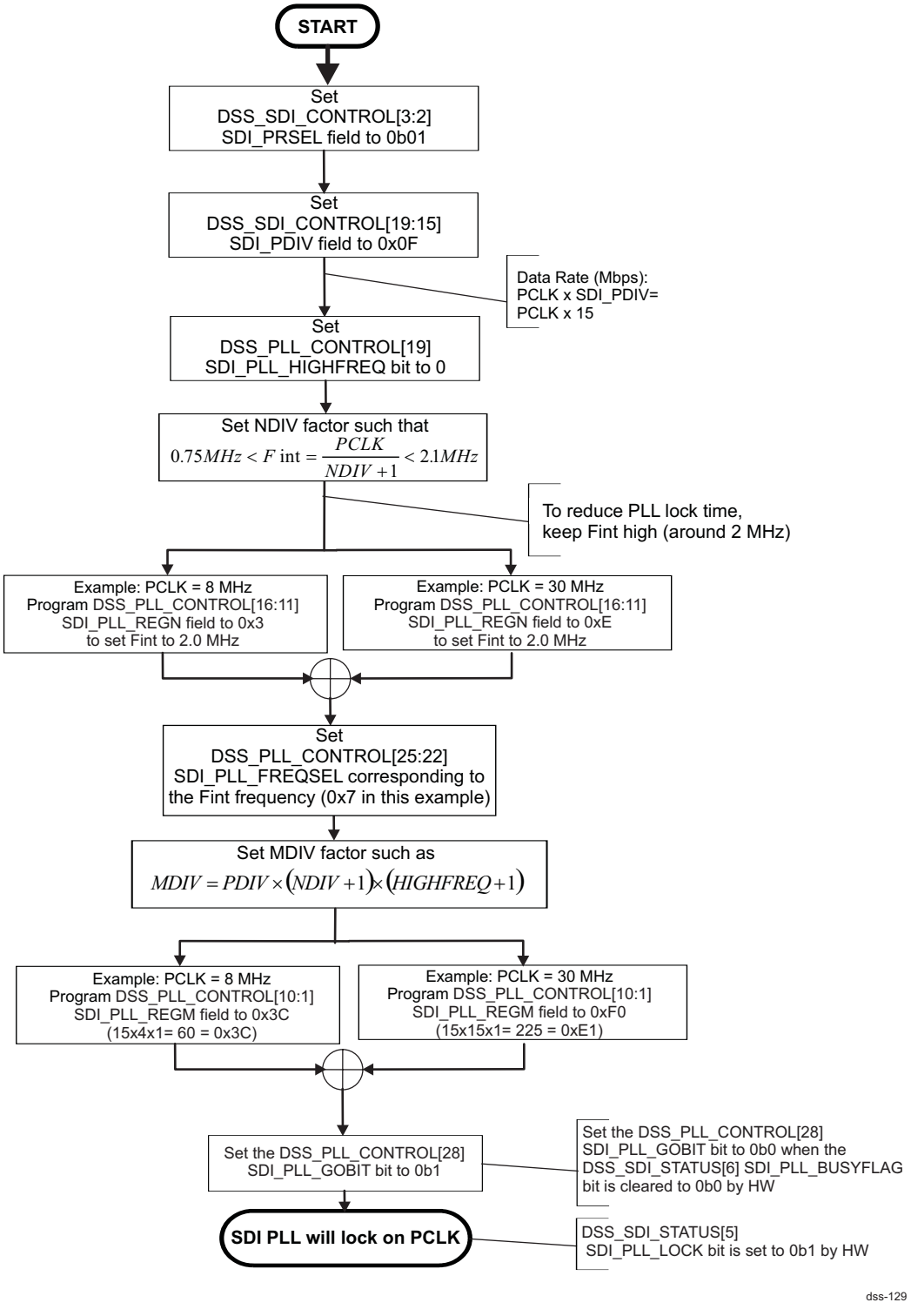
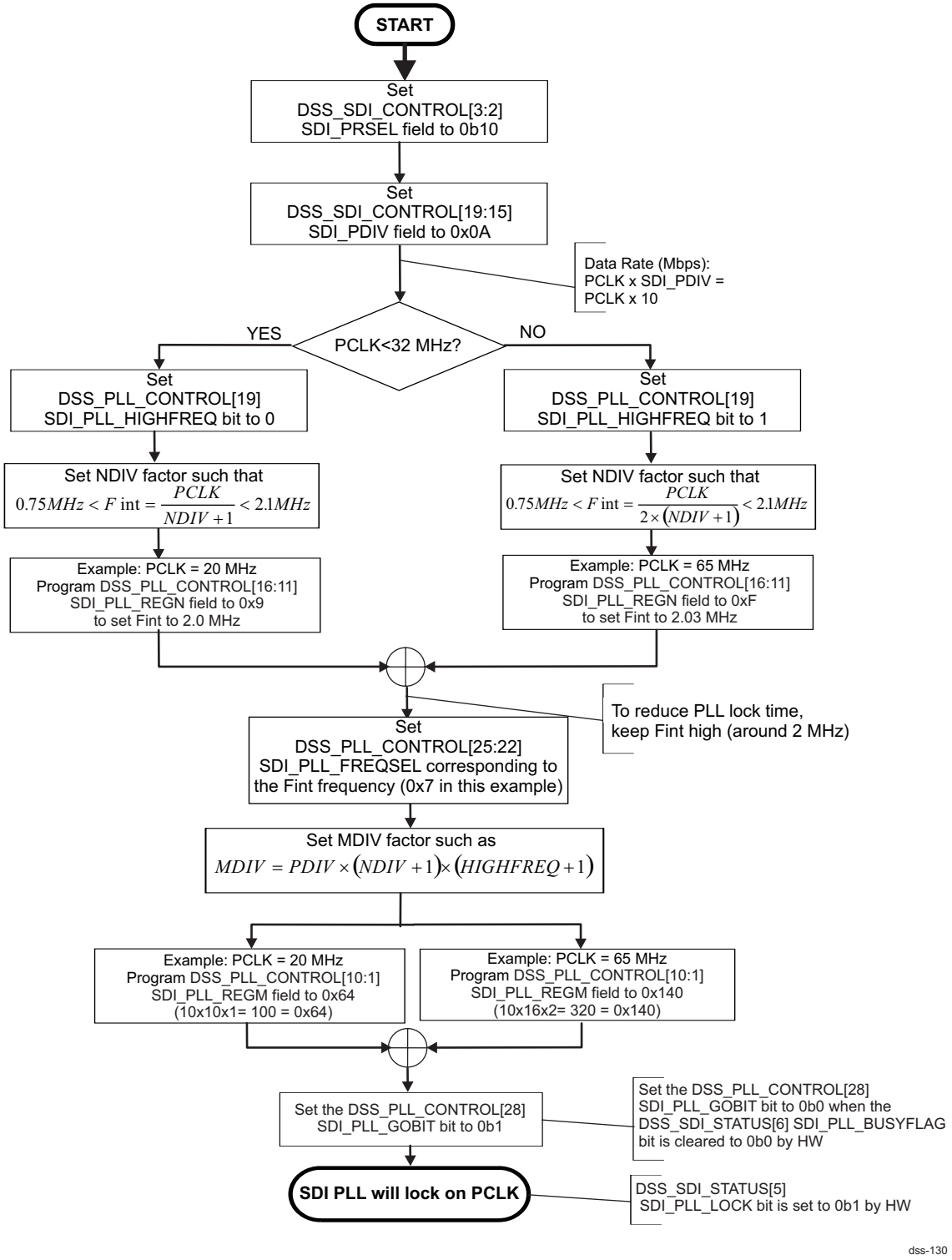


Figure 15-163. Flowchart SDI 3-Data Pairs



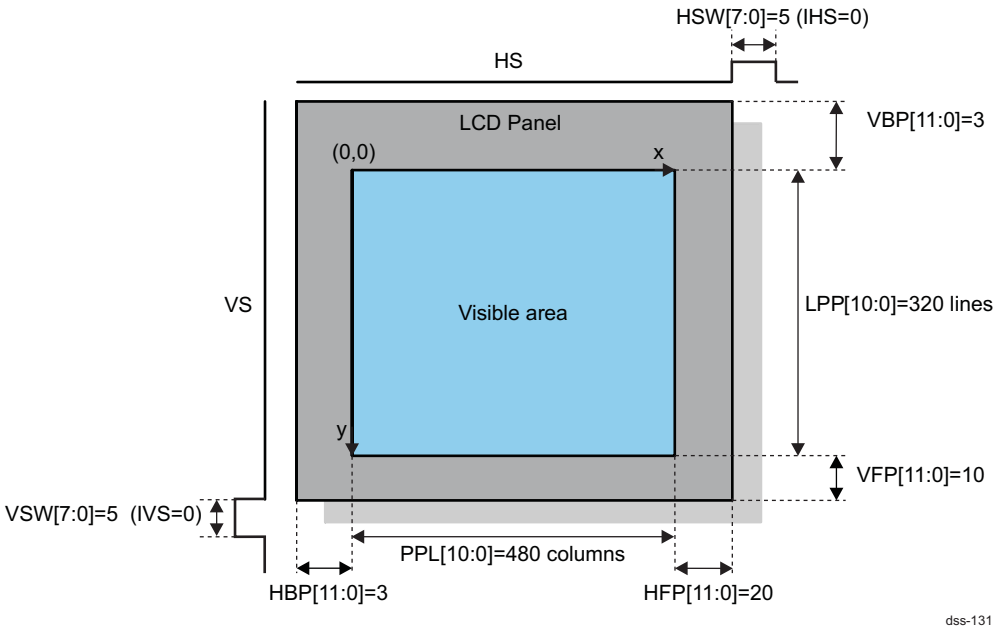
CAUTION

The PLL low-power mode (LP mode) can be activated with DSS.DSS_PLL_CONTROL[21] SDI_PLL_PLLLPMODE bit. By setting this bit to 1, software users reduce the power and increases the jitter. In LP mode, there is no effect on lock time; but in this mode, the SDI PLL allows only 0.75-2.1MHz range (DSS.DSS_PLL_CONTROL[25:22] SDI_PLL_FREQSEL bit field value between 0x3 and 0x7). If software users are using the range 7.5 MHz to 21 MHz (DSS.DSS_PLL_CONTROL[25:22] SDI_PLL_FREQSEL bit field value between 0xB and 0xF) in non-LP mode, and then switching to LP mode and setting the PLL in the range of 0.75 MHz to 2.1 MHz, there will be an effect on the lock time.

15.6.3.3 Application Example: HVGA Display

The following example details the SDI PLL settings for a half-VGA display with the following parameters shown in Figure 15-164.

Figure 15-164. HVGA Display



15.6.3.3.1 HVGA Display

- Display resolution: 320 x 480
- Frame refresh rate: 58.4 Hz
- Vertical visible pixel: 480 lines
- Vertical front porch: 20 lines
- Vertical sync: 5 lines
- Vertical back porch: 3 lines
- Horizontal visible pixel: 320 columns
- Horizontal front porch: 10 columns
- Horizontal sync: 5 columns
- Horizontal back porch: 3 columns

The calculation of the total number of pixel and Blanking overhead is as follows:

- Visible area pixel count: 480 x 320 = 153600 pixel
- Total frame pixel count: (480 + 20 + 5 + 3) x (320 + 10 + 5 + 3) = 171704 pixel
- Blanking overhead: (171704 153600)/153600 = 11.8%

This application requires the following SDI parameters:

- Pixel clock frequency: 171704 x 58.4 Hz = 10.03 MHz
- Serial data rate:
  - 1-channel mode: 10.03 MHz x 30 bit/channel = 300.9 Mbps
  - 2-channel mode: 10.03 MHz x 15 bit/channel = 150.45 Mbps

15.6.3.3.2 SDI PLL Settings for 1-Channel Mode:

- 1-data pair: Set DSS_SDI_CONTROL[3:2] SDI_PRSEL bit field to 0b00.
- Set the DSS_SDI_CONTROL[19:15] SDI_PDIV bit field to 0x1E.
- Set the DSS_PLL_CONTROL[19] SDI_PLL_HIGHFREQ bit to 0.
- Set the DSS_PLL_CONTROL[16:11] SDI_PLL_REGN bit field to 0x4. Thus Fint will be set to 2 MHz.
- Set the DSS_PLL_CONTROL[25:22] SDI-PLL_FREQSEL bit field to 0x7.
- Set DSS_PLL_CONTROL[10:1] SDI_PLL_REGM bit field to 0x96 (30x5x1 = 150 = 0x96).

15.6.3.3.3 SDI PLL Settings for 2-Channel Mode:

- 2-data pair: Set DSS_SDI_CONTROL[3:2] SDI_PRSEL bit field to 0b01.
- Set the DSS_SDI_CONTROL[19:15] SDI_PDIV bit field to 0x0F.
- Set the DSS_PLL_CONTROL[19] SDI_PLL_HIGHFREQ bit to 0.
- Set the DSS_PLL_CONTROL[16:11] SDI_PLL_REGN bit field to 0x4. Thus Fint will be set to 2 MHz.
- Set the DSS_PLL_CONTROL[25:22] SDI-PLL_FREQSEL bit field to 0x7.
- Set DSS_PLL_CONTROL[10:1] SDI_PLL_REGM bit field to 0x4B (15x5x1 = 75 = 0x4B).

15.6.4 How to Interface OMAP Device With SN65LVDS302 Receiver for an XGA Display Application

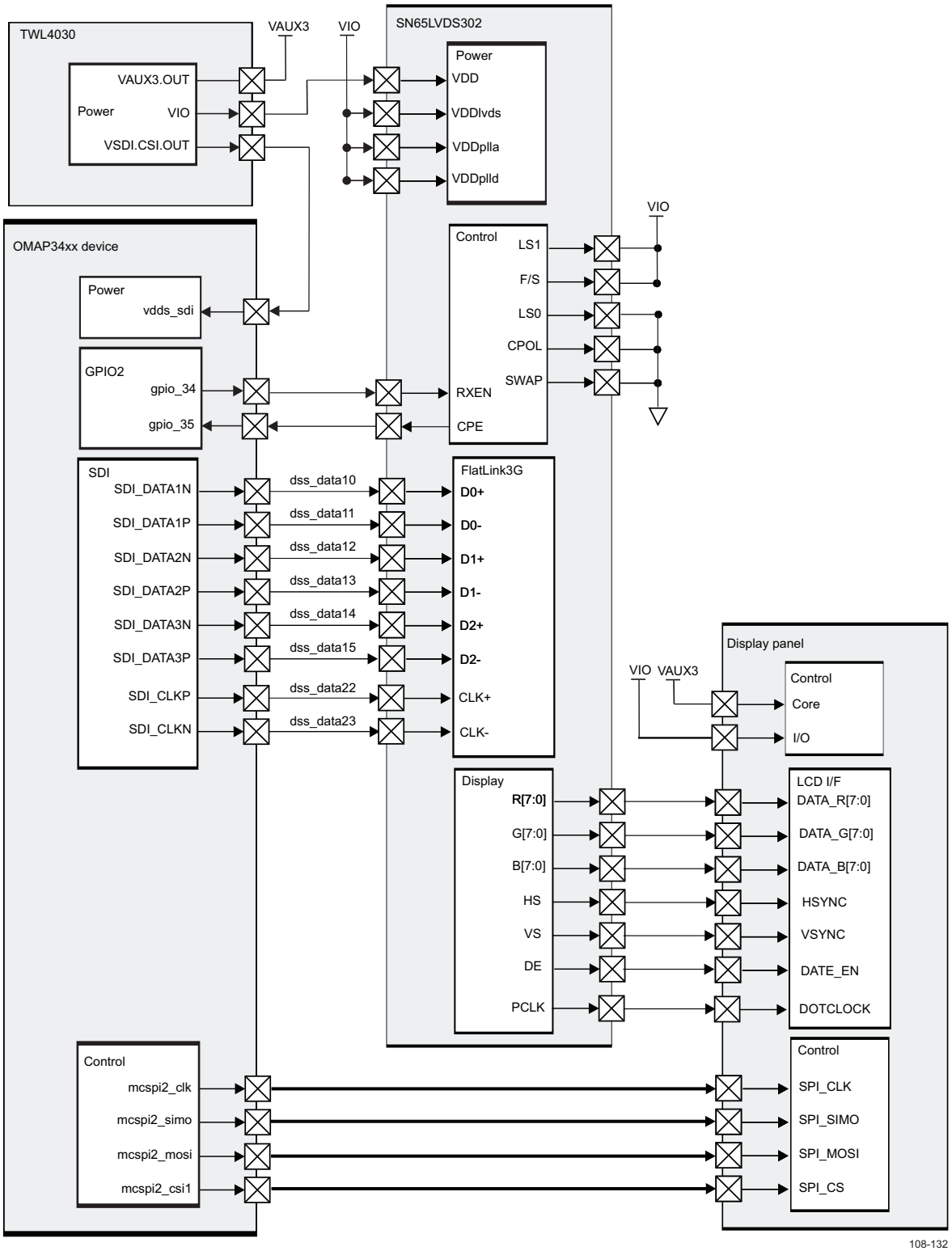
Texas Instruments Inc. provides a full FlatLink3G solution with the device associated to a SN65LVDS302 receiver. This section will also detail the hardware connections between these two devices and the basic software settings

This use case is based on the usage of a large (XGA) 16 Mega colors display requiring the three data channels and the complete output data bus to be used. Some adaptations may be adopted if the device is used in a less demanding use case

15.6.4.1 Hardware Connections

Figure 15-165 details the hardware connections with the device, the SN65LVDS302 receiver, the display panel and the power IC. The power IC can be the TWL4030 companion chip

Figure 15-165. Hardware Connections for FlatLink3G Application



Note: The SN65LVDS302 device was design-optimized for robustness to operate reliably in a dense environment with other digital switching ICs. In cell phone designs, the SN65LVDS302 often shares a power supply with the LCD display driver provided through the FPC interconnect. To minimize the power supply noise floor it is suggested to provide good decoupling near the SN65LVDS302 power pins. Using four capacitors (2x0.01F and 2x0.1F) was found to provide good performance. At the very least, it is recommended to install one 0.1F and one 0.01F capacitor near the SN65LVDS302. To avoid large current loops and trace inductance, the trace length between decoupling cap and IC power inputs pins must be minimized. Placing the capacitor underneath the SN65LVDS302 device on the bottom of the PCB is often a good choice

15.6.4.2 SN65LVDS302 Receiver Description

SN65LVDS302 is a serial interface receiver which converts the FlatlinkTM3G signals (3 sub-LVDS data pairs) into 27 signals (24 RGB bits, HS, VS and DE signals). It also provides a pixel clock (4 MHz to 65 MHz) which is formed from the sub-LVDS input clock pair, internally multiplied for data deserialization and then divided by same factor.

The device may operate in different modes, depending on the input data format. One, two or all of the three data pairs will be used, following the LS0 and LS1 configuration. The clock division rate will automatically be selected from 30 to 10. Pixel clock polarity can be chosen using the CPOL signal

The output bus order is configurable by SWAP pin configuration (R[7:0], G[7:0], B[7:0] or B[0:7], G[0:7], R[0:7]). Furthermore, the CMOS outputs rising/falling edges will be selected between slow or fast depending on F/S input to fit display timing constraints with regards to EMI and consumption consequences.

In the XGA display application detailed in Figure 7, the configuration is the following:

- LS1 = 1 and LS0 = 0: 3-data pairs mode is selected
- SWAP = 0: The data output format is R[7:0], G[7:0], B[7:0]
- F/S = 1: The CMOS bus time rise is selected in fast output
- CPOL = 0: The output clock polarity is rising edge

For more details on SN65LVDS302 receiver, please see the SN65LVDS302 data sheet (SLLS733)

The RXEN signal will enable the OMAP device to control SN65LVDS302 activity which will signal channel parity error through CPE signal. These two previous signals are connected to OMAP GPIO2 module

15.6.4.3 SDI Software Settings

15.6.4.3.1 SDI Configuration

To use the SDI with the TI FlatLink3G display panel:

- Set the DSS.DISPC_CONTROL[3] STNTFT bit to 0b1 (active matrix display operation mode)
- Set the DSS.DISPC_CONTROL[9:8] TFTDATALINES bit field to 0x3 (24-bit data output)
- Set the DSS.DSS_SDI_CONTROL[1:0] SDI_BWSEL bit field to 0x2 (color depth is 24 bits)

15.6.4.3.2 Signal Features Configuration

For proper SDI operation, configure the following features in the display controller:

- Set the DSS.DISPC_POL_FREQ[16] RF bit to 0b1 for driving HSYNC and VSYNC on the rising edge of PCLK.
- Set the DSS.DISPC_POL_FREQ[17] ONOFF bit to 0b1.
- Clear the DSS.DISPC_POL_FREQ[14] IPC bit to 0b0 for driving the pixel data on the rising edge of PCLK
- Set the DSS.DISPC_CONTROL[29] LCDENABLEPOL bit to 0b1 for setting the LCD-enable signal active high.

The following settings are not required by the SDI2 interface, but these polarities settings are the common polarities used by FlatLink3G displays:

- Clear the DSS.DISPC_POL_FREQ[13] IHS bit to 0b0 for setting HSYNC active high.
- Clear the DSS.DISPC_POL_FREQ[12] IVS bit to 0b0 for setting VSYNC active high.
- Clear the DSS.DISPC_POL_FREQ[15] IEO bit to 0b0 for setting the data enable active high.

15.6.4.3.3 SDI PLL Configuration

The SDI module settings for an XGA display application with 20% blanking overhead are detailed in this section.

- XGA display: 1024 x 768 = 786432 pixels
- 20% blanking overhead: 157286 pixels
- Visible are pixel count: 786432 + 157286 = 943718
- Frame Refresh Rate: 60 Hz
- Pixel Clock Frequency: 943718 x 60 = 56.6 MHz (in 20-65 MHz range for 3-data pairs).
- Serial Data rate: 3-channel mode : 56.6 MHz x 10 bit/channel = 566 Mbps

15.6.4.3.3.1 SDI PLL settings for 3-Channel Mode:

- 3-data pair: Set DSS_SDI_CONTROL[3:2] SDI_PRSEL bit field to 0b10
- Set the DSS_SDI_CONTROL[19:15] SDI_PDIV bit field to 0x0A
- Set the DSS_PLL_CONTROL[19] SDI_PLL_HIGHFREQ bit to 1 as PCLK>32 MHz
- Set the DSS_PLL_CONTROL[16:11] SDI_PLL_REGN bit field to 0xE. Thus Fint will be set to 2.02 MHz
- Set the DSS_PLL_CONTROL[25:22] SDI_PLL_FREQSEL bit field to 0x7
- Set DSS_PLL_CONTROL[10:1] SDI_PLL_REGM bit field to 0x12C (10x(14+1)x2 = 300 = 0x12C)

Set the SDI_PLL_CONTROL[27:26] SDI_PLL_LOCKSEL bit field to 0b00 to select the phase lock criteria to 0b00 (recommended for FlatLink3G mode)

15.6.4.4 SN65LVDS302 Receiver Settings

15.6.4.4.1 Receiver Power-Up

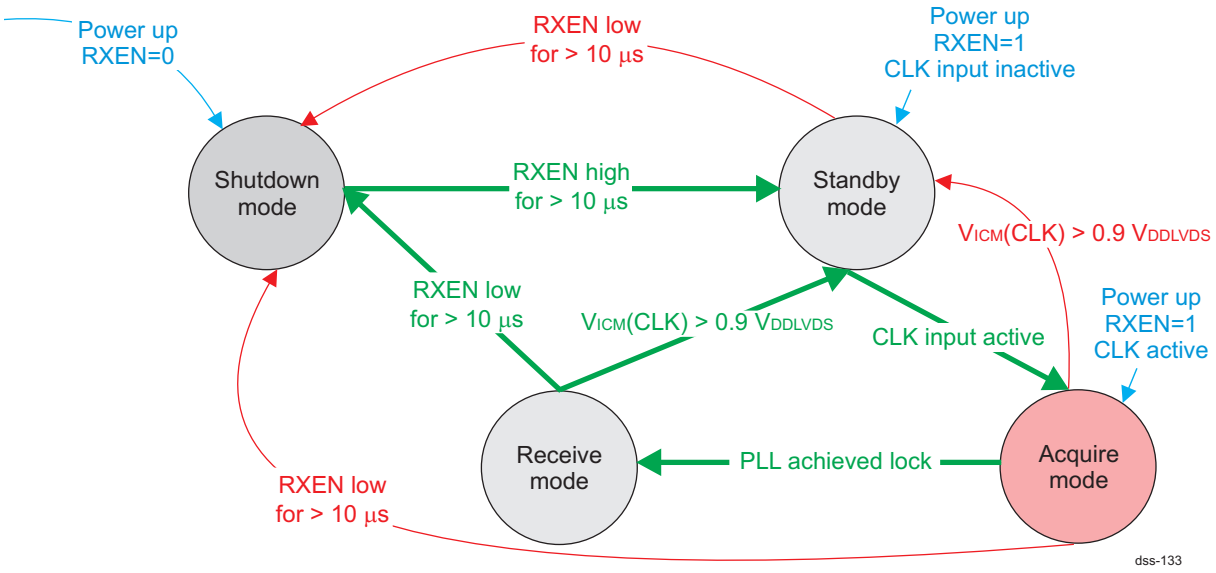
At power up, the receiver is enabled immediately if RXEN=1 (enters Acquire mode) and disabled if RXEN=0 (enters shutdown mode). The receiver may be turned off at any time by pulling RXEN low for at least 10 µs. This disables the CMOS drivers and turns off the receiver PLL, putting the device in shutdown mode. The RXEN signal is connected to the gpio_34 pin of the device. Thus the GPIO2 module can control the receiver power-up.

Note: The RXEN input incorporates glitch suppression logic to avoid unwanted switching. The input must be pulled continuously low for at least 10 µs to force the receiver to enter shutdown. The gpio_34 signal must be set continuously high by the OMAP device for at least 10 µs to activate the receiver. An input pulse shorter than 5 µs is interpreted as a glitch and is ignored.

15.6.4.4.2 Receiver Modes and Transitions

Figure 15-166 details the SN65LVDS302 receiver modes and transitions.

Figure 15-166. SN65LVDS302 Receiver Modes and Transitions



15.6.4.4.2.1 Modes Description

- Shutdown mode

  The SN65LVDS302 receiver may be put in Shutdown mode by holding the RXEN pin low. This turns off most of the receiver circuitry including the Sub-LVDS receivers, PLL, and deserializers. The Sub-LVDS input impedance remains 100- while any input signal becomes ignored. All outputs will hold a static output pattern: R[0:7]=G[0:7]=B[0:7]=VS=HS=high; DE=PCLK=low. The current draw in Shutdown mode will be nearly zero
- Standby mode

  The SN65LVDS302 receiver will enter the Standby mode when the SN65LVDS302 receiver is not in Shutdown mode but the clock input common mode range on the Sub-LVDS clock input shifts to VDDLVDS. The CLK input incorporates a pull-up circuitry. This circuit will shift the Sub-LVDS clock input common mode voltage to VDDLVDS in the absence of any input signal. All circuitry, except for the Sub-LVDS clock input Standby monitor will be shutdown. The Sub-LVDS input impedance remains 100- while any input signal on the data inputs D0, D1, and D2 becomes ignored. All outputs will hold a static output pattern: R[0:7]=G[0:7]=B[0:7]=VS=HS=high; DE=PCLK=low. The current draw in Standby mode will be very low.
- Active Modes

  The SN65LVDS302 receiver will be in active mode when the device is enabled by pulling RXEN high and VICM on the CLK input is driven below 0.9xVDD by an external input signal. Current draw in active mode will be dependent on operating frequency and the number of data transitions in the data payload
- Acquire Mode (PLL approaches lock)

  While the SN65LVDS302 receiver is enabled and a Sub-LVDS clock input present, the receiver PLL will pursue lock to the input clock. As long as the PLL is not locked the output data bus will hold a static output pattern: R[0:7]=G[0:7]=B[0:7]=VS=HS=high; DE=PCLK=low.
- Receive Mode

  When the receiver PLL achieves lock, the receiver enters the normal receive mode with the output bus presenting the deserialized data.

15.6.4.4.2.2 Mode Transitions

Table 15-82. SN65LVDS302 Receiver Mode Transitions

Mode Transition	Use Case	Transition Specifics
Shutdown to Standby	Drive RXEN high to enable receiver	• 1. RXEN high for longer than 10 µs • 2. Receiver enters Standby mode – a. R[0:7]=G[0:7]=B[0:7]=VS=HS remain high and DE=PCLK low – b. receiver turns on clock input monitor
Standby to acquire	Transmitter activity detected	• 1. CLK input monitor detects clock input activity • 2. Outputs remain static • 3. the receiver PLL becomes enabled
Acquire to Receive	Link is ready to receive data from OMAP SDI module.	• 1. PLL is active and achieves lock within 2 µs • 2. D1, D2, and D3 become active • 3. first Data word was recovered • 4. parallel output bus switches from static output pattern to output first valid data word
Receive to Standby	Receiver requested to enter Standby mode by input common mode voltage V_{ICM}>0.9 V_{DDLVDS} (that is, transmitter output clock turns high-impedance)	• 1. Transmitter disables outputs • 2. RX Input monitor detects V_{ICM}>0.9 V_{DDLVDS} • 3. R[0:7]=G[0:7]=B[0:7]=VS=HS transition to high and DE=PCLK to low on next falling PLL clock edge • 4. PLL shuts down • 5. Clock activity input monitor remains active
Receive/Standby to Shutdown	Turn off receiver	• 1. RXEN pulled low for longer than 10 µs • 2. Receiver immediately puts all outputs into high impedance • 3. Most receiver blocks are shut down for least power consumption

15.6.4.4.3 Parity Error Detection and Handling

The SN65LVDS302 receiver performs an error checking on the basis of a parity bit that is transmitted across the LVDS interface from the transmitting device. Once the SN65LVDS302 detects the presence of the clock and the receiver PLL has locked onto PCLK, then the parity is checked. The parity bit covers the 27-bit data payload (reserved bits are not covered in the parity calculations). Odd Parity bit signaling is used. The parity error is output on the output CPE pin:

- If the sum of the 27 data bits and the parity bit result in an odd number, the receive data are assumed to be valid and CPE output will be held low.
- If the sum equals an even number, parity error is declared and CPE output will indicate high for half a PCLK period.

The CPE output will be set with the data bit transition and cleared after 1/2 the data bit time. The receiver CPE output pin is connected to gpio_35 pin. The OMAP device can use the gpio_35 input pin to generate an IRQ, respectively GPIO2_MPU_IRQ for MPU subsystem and GPIO2_IVA2_IRQ for IVA2.2 subsystem. This will allow the OMAP device to count every detected parity error with a simple counter triggered by GPIO2 module interrupt request.

When a parity error is detected, the data on that PCLK cycle is not output. Instead, the last valid data from a previous PCLK cycle is repeated on the output bus. This is to prevent any bit error that may occur on the LVDS link from causing perturbations in VS, HS or DE that may be visually disruptive to the display panel.

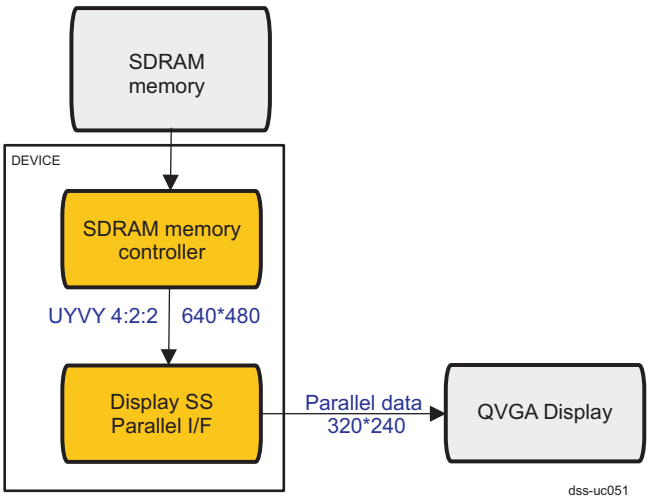
15.6.5 Camcorder Use Case: How to Configure the Display Subsystem When Connected With a QVGA LCD Panel

15.6.5.1 Overview

This section describes the configuration and used of a QVGA LCD panel through the parallel data interface (in RFBI by-pass mode).

The display subsystem reads each image from the SDRAM memory in the UYVY 4:2:2 format and converts it to a RGB18-666 pixel format to send to the QVGA LCD panel (see Figure 15-167).

Figure 15-167. Overview



The camcorder use case is configured as follows:

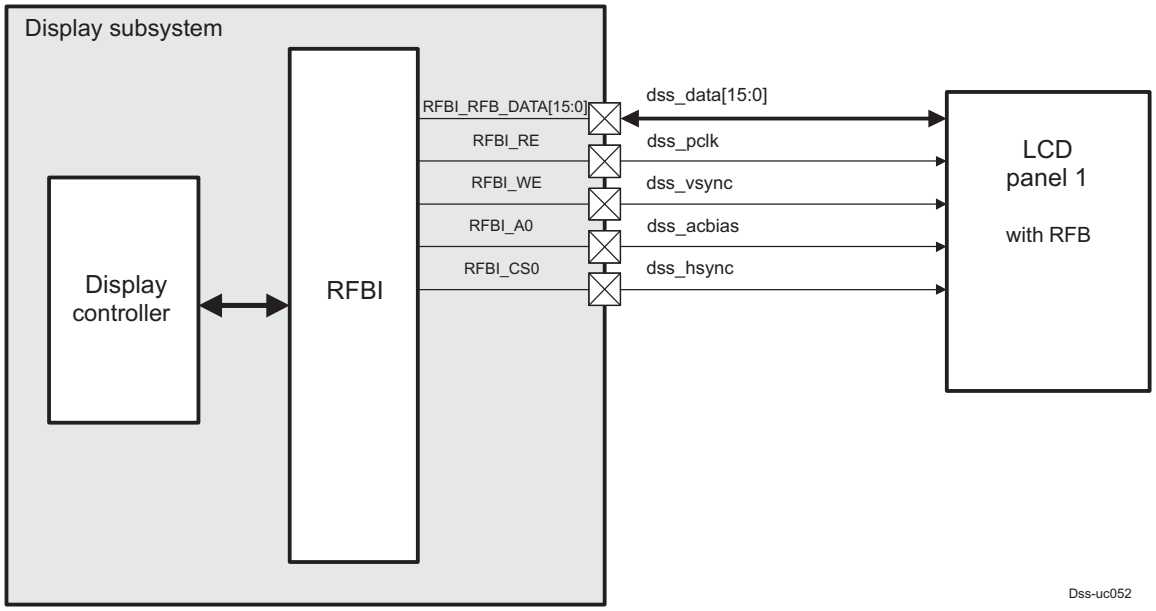
- Input Image: VGA 640*480 pixels , UYVY 4:2:2 format
- Output Image Pixel: QVGA 320*240 pixels , RGB18-666
- Pixel Clock (PCLK): 6 MHz to allow a video display of 60 frames per second
- No DMA requests to the system DMA, but the display subsystem uses its DMA controller to access the data into SDRAM
- VSYNC, SYNCLOST, VID1FIFOUNDERFLOW, and VID1ENDWINDOW interrupts used to monitor the video1 channel

15.6.5.2 Environment

In this use case, the display subsystem is connected to a QVGA LCD panel by the parallel interface.

Figure 15-168 details the OMAP device connections with the LCD panel.

Figure 15-168. Environment



15.6.5.2.1 LCD panel Features

The LCD panel includes the following features:

- Display format 320(H) x 240(V)
- Color mode: 18 bit/pixel R:5 G:6 B:5 (65 536 colors)

15.6.5.3 Data Path

In this use case, the display subsystem manages the following:

- Video stream (UYVY 4:2:2) between the SDRAM controller (SDRC) and the display controller through the internal DMA controller (L3 interconnect)
- Video stream (RGB16) between the display controller and the parallel module
- Video stream (parallel data) between the parallel module and the QVGA LCD panel

Figure 15-169 shows the data flow inside the display subsystem highlighted in red.

Figure 15-169. Display Subsystem Data Flow

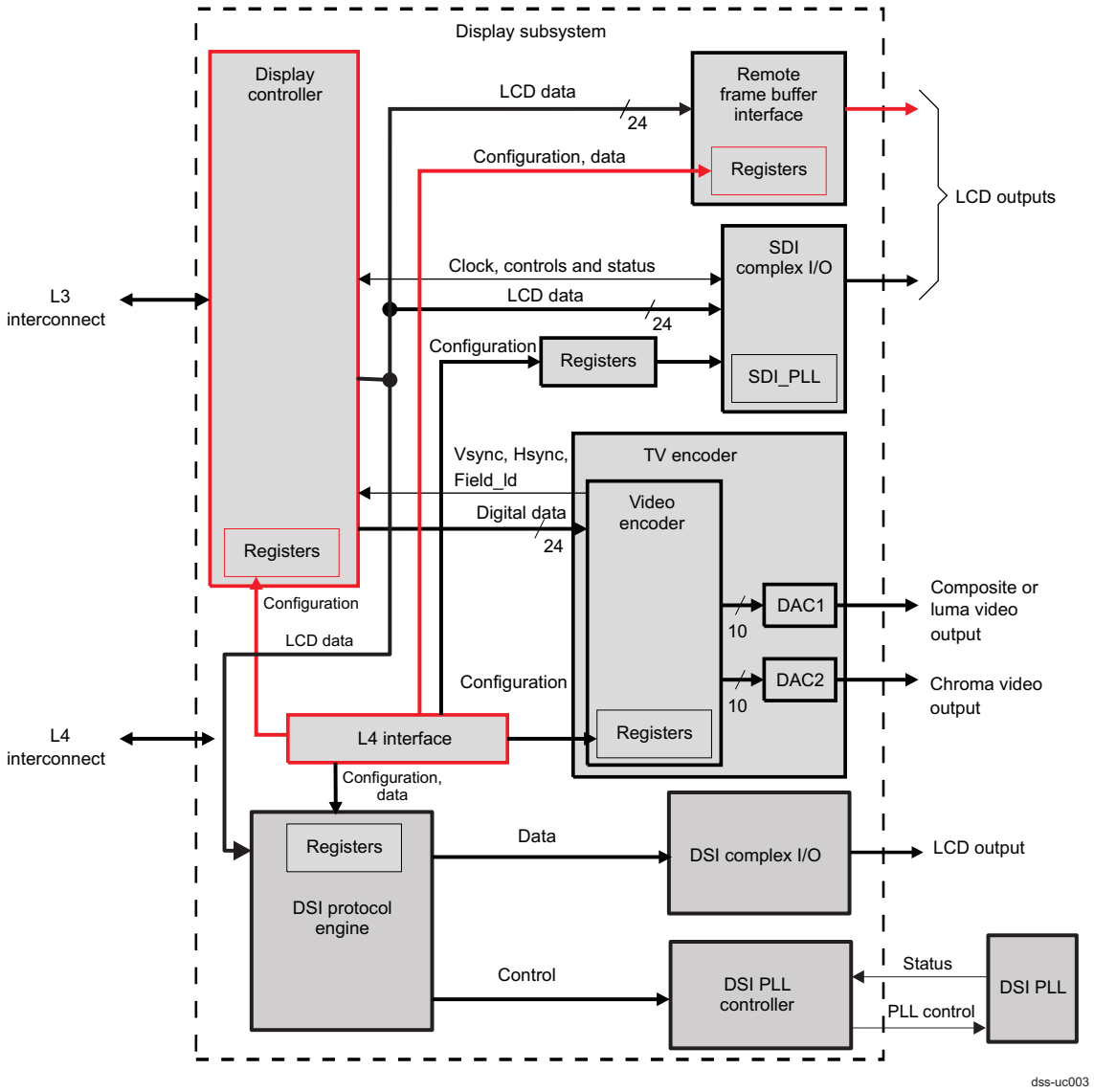
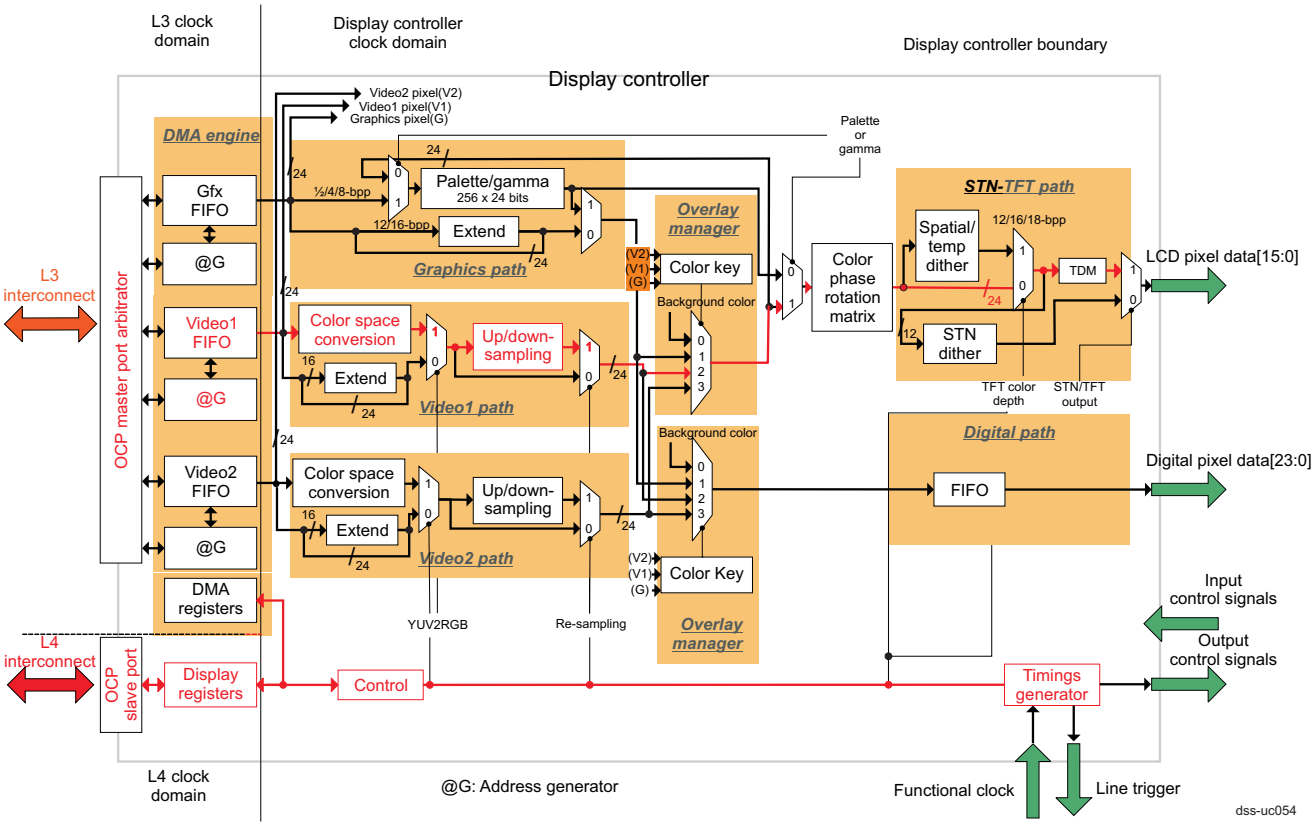


Figure 15-170 shows the data flow inside the display controller submodule highlighted in red.

Figure 15-170. Display Controller Data Flow



15.6.5.4 Programming Flow

This section details the programming flow of the display panel configuration for the camcorder use case.

The display panel is configured in 3 steps (see Figure 15-171).

Figure 15-171. Display Panel Configuration for the Camcorder Use Case

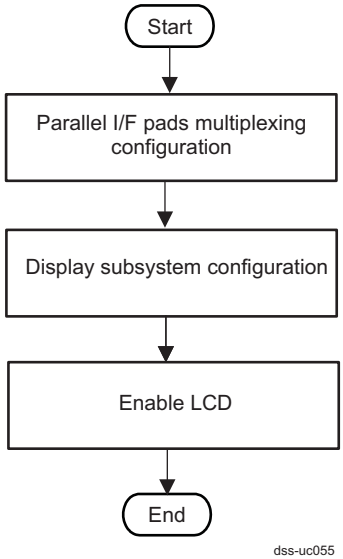
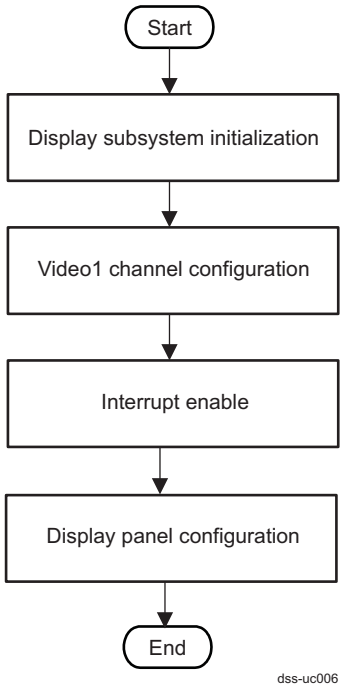


Figure 15-172 shows the programming flow of the display subsystem initial configuration, which consists of four steps:

Figure 15-172. Display Subsystem Configuration for Camcorder Use Case



15.6.5.4.1 Pads Multiplexing Configuration

All parallel pads are multiplexed in mode 1 on the dss_data pins:

Table 15-83 shows all the System Control Module registers.

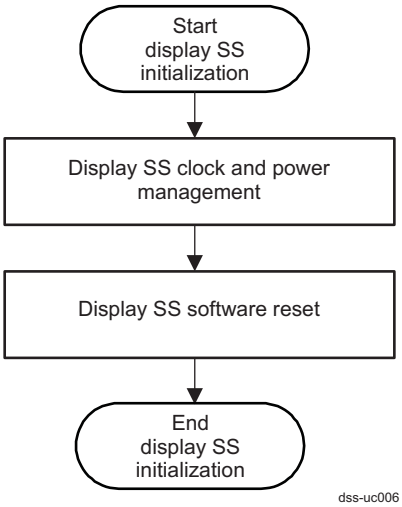
Table 15-83. Registers Print for QVGA LCD panel Pads Multiplexing Configuration

Register Name	Address	Value	Value Description
SCM.CONTROL_PADCONF_DSS_PCLK	0x4800 20D4	0x0110 0110	Configure dss_pclk and dss_hsync pads in mode 0
SCM.CONTROL_PADCONF_DSS_VSYNC	0x4800 20D8	0x0100 0110	Configure dss_vsync and dss_acbias pads in mode 0
SCM.CONTROL_PADCONF_DSS_DATA0	0x4800 20DC	0x0100 0100	Configure dss_data0 and dss_data1 pads in mode 0
SCM.CONTROL_PADCONF_DSS_DATA2	0x4800 20E0	0x0100 0100	Configure dss_data2 and dss_data3 pads in mode 0
SCM.CONTROL_PADCONF_DSS_DATA4	0x4800 20E4	0x0100 0100	Configure dss_data4 and dss_data5 pads in mode 0
SCM.CONTROL_PADCONF_DSS_DATA6	0x4800 20E8	0x0100 0100	Configure dss_data6 and dss_data7 pads in mode 0
SCM.CONTROL_PADCONF_DSS_DATA8	0x4800 20EC	0x0100 0100	Configure dss_data8 and dss_data9 pads in mode 0
SCM.CONTROL_PADCONF_DSS_DATA10	0x4800 20F0	0x0100 0100	Configure dss_data10 and dss_data11 pads in mode 0
SCM.CONTROL_PADCONF_DSS_DATA12	0x4800 20F4	0x0100 0100	Configure dss_data12 and dss_data13 pads in mode 0
SCM.CONTROL_PADCONF_DSS_DATA14	0x4800 20F8	0x0100 0100	Configure dss_data14 and dss_data15 pads in mode 0

15.6.5.4.2 Display Subsystem Initialization

Figure 15-173 details the display subsystem initialization.

Figure 15-173. Display Subsystem Initialization



15.6.5.4.2.1 Display Subsystem Clock and Power Management

The display subsystem clock (DSS1_ALWON_FCLK) is provided by the DPLL4 module in the PRCM (DPLL4_M4_X2). The software should set the following PRCM registers:

1. The DPLL4_ALWON_FCLKOUT clock is sourced by the system clock (SYS_CLK) at 26 MHz and calculated as follows:

$$DPLL4_ALWON_FCLKOUT = \frac{SYS_CLK(19.2\ MHz) \times 2 \times M\ (0xE1)}{N\ (0x9) + 1} = 864\ MHz$$

dss-uc020

   Set the M multiplier to 225 by setting the PRCM.CM_CLKSEL2_PLL[18:8] PERIPH_DPLL_MULT bit field to 0xE1.

   Set the N divider to 9 by setting the PRCM.CM_CLKSEL2_PLL[6:0] PERIPH_DPLL_DIV bit field to 0x09.
2. To follow the timing requirements for the use case, the DPLL4_M4_X2 clock frequency is calculated as follows:

$$DPLL4_M4X2_CLK = \frac{DPLL4_ALWON_FCLKOUT(864\,MHz)}{9} = 96\ MHz$$

dss-ucE007

   Set the DPLL4 ratio by programming to 0x9 the PRCM.CM_CLKSEL_DSS[4:0] CLKSEL_DSS1 bit field.
3. Enable the display subsystem functional clock (DSS1_ALWON_FCLK): Set the PRCM.CM_FCLKEN_DSS[0] EN_DSS1to 0x1.
4. Enable the display subsystem TV functional clock (DSS_TV_FCLK): Set the PRCM.CM_FCLKEN_DSS[2] EN_TV to 0x1.
5. Enable the display subsystem interface clock (DSS_L4_ICLK): Set the PRCM.CM_ICLKEN_DSS[0] EN_DSS to 0x1.

   Table 15-84 describes the registers from the PRCM to be configured for the display subsystem clock configuration step and the use case values.

Table 15-84. Registers Print for Display Subsytem Clock Management

Register Name	Address	Value	Value Description
PRCM.CM_FCLKEN_DSS	0x4800 4E00	0x0000 0005	Enable DSS1_LWON_FCLk and DSS_TV_FCLK
PRCM.CM_ICLKEN_DSS	0x4800 4E10	0x0000 0001	Enable DSS_L4_ICLK
PRCM.CM_CLKSEL2_PLL	0x4800 4D44	0x0000 E109	DPLL4 multiplier set to 0xE1 and divider set to 0x9
PRCM.CM_CLKSEL_DSS	0x4800 4E40	0x0001 0009	DPLL4 M4X2 divider set to 0x9

The display subsystem power management is programmed by the following PRCM registers:

1. Disable the autoidle mode: Set the PRCM.CM_AUTOIDLE_DSS register to 0x0.
2. Disable the DSS domain sleep dependency: Set the PRCM.CM_SLEEPDEP_DSS register to 0x0.
3. Disable the automatic transition between active and inactive: Set the PRMC.CM_CLKSTCTRL_DSS register to 0x0.

   Table 15-85 describes the registers from the PRCM to be configured for the display subsystem power management step and the use case values.

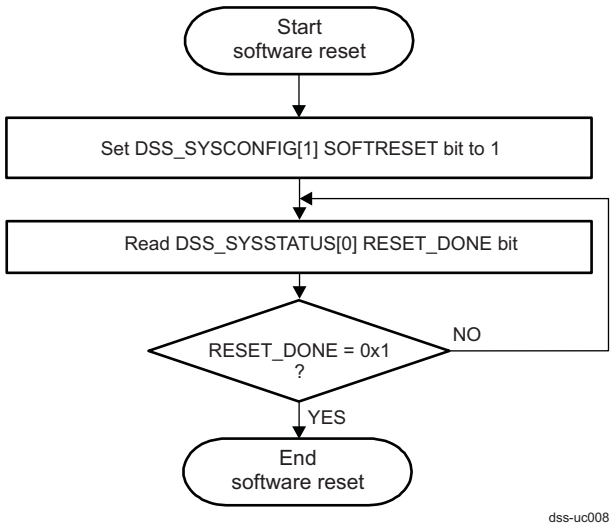
Table 15-85. Registers Print for Display Subsystem Power Management

Register Name	Address	Value	Value Description
PRCM.CM_AUTOIDLE_DSS	0x48004E30	0x0000 0000	Disable autoidle mode
PRCM.CM_SLEEPDEP_DSS	0x48004E44	0x0000 0000	Disable the DSS domain sleep dependency
PRCM.CM_CLKSTCTRL_DSS	0x48004E48	0x0000 0000	Disable the automatic clock state transition

15.6.5.4.2.2 Display Subsystem Software Reset

Perform a software reset as described in the following flowchart (see Figure 15-174).

Figure 15-174. Software Reset Flowchart



CAUTION

To update the RESETDONE status bit correctly, all interface and functional clocks, even for the TV output, must be provided to the display subsystem.

Table 15-86 describes the registers to be configured for the display subsystem software reset step and the use case values.

Table 15-86. Registers Print for Display Subsystem Software Reset

Register Name	Address	Value	Value Description
DSS_SYSCONFIG	0x4805 0010	0x0000 0002	Initiate a software reset. The DSS_SYSCONFIG[1] SOFTRESET is automatically reset by hardware
DSS_SYSSTATUS	0x4805 0014	0x0000 0001	The DSS_SYSSTATUS[0] RESETDONE is set to 1 when the reset sequence is done

15.6.5.4.3 Video1 Channel Configuration

The video1 channel is configured by programming the following steps:

1. Select the UYVY 4:2:2 format by programming the DSS.DISPC_VID1_ATTRIBUTES[4:1] VIDFORMAT bit field to 0xB.
2. Select the LCD data flow by programming the DSS.DISPC_VID1_ATTRIBUTES[16] VIDCHANNELOUT bit to 0x0.
3. Select the video DMA burst size to 16x32 by setting the DSS.DISPC_VID1_ATTRIBUTES[15:14] VIDBURSTSIZE to 0x2.
4. Program the image base address by setting the DSS.DISPC_VID1_BA0 register.
5. Program the FIFO thresholds as follows:

   To set the FIFO high threshold to 1023 bytes (maximum value allowed), set the DSS.DISPC_VID1_FIFO_THRESHOLD[27:16] VIDFIFOHIGHTHRESHOLD bit field to 0x3FF.

   Set the FIFO low threshold as described in the following equation:

$$\text{VIDFIFOLOWTHRESHOLD} = \text{VIDFIFOHIGHTHRESHOLD (0x3FF)} - \frac{\text{VIDBURSTSIZE(16x32)}}{8} = 959\ (0x3BF)$$

dss-ucE009

   Program the DSS.DISPC_VID1_FIFO_THRESHOLD[11:0] VIDFIFOLOWTHRESHOLD to 0x3BF.
6. Program the windows X-position to 0 (first column starting on the left edge of the screen) and the window Y-position to 0 (first row starting at the top of the screen): Set the DSS.DISPC_VID1_POSITION[10:0] VIDPOSX and the DSS.DISPC_VID1_POSITION[26:16] VIDPOSY registers to 0x0.
7. Program the window width to 320 by setting the DSS.DISPC_VID1_SIZE[10:0] VIDSIZEX to 0x13F.
8. Program the window height to 240 by setting the DSS.DISPC_VID1_SIZE[26:16] VIDSIZEY to 0xEF.
9. Program the picture width to 640 by setting the DSS.DISPC_VID1_PICTURE_SIZE[10:0] VIDORGSIZEX to 0x27F.
10. Program the picture height to 480 by setting the DSS.DISPC_VID1_PICTURE_SIZE[26:16] VIDORGSIZEY to 0x1DF.
11. Because the video format is in UYVY, software users must program the color conversion module as follows:

    To enable the color conversion module, set the DSS.DISPC_VID1_ATTRIBUTES[9] VIDCOLORCONVENABLE bit to 1.

    To select a limited range for color space conversion, set the DSS.DISPC_VID1_ATTRIBUTES[11] VIDFULLRANGE bit to 0.

    The DSS.DISPC_VID1_COEFi registers (with i = 0 to 4) have nine 11-bit coefficients defined for the programmable color space conversion block for video pipeline 1:

$$\begin{bmatrix} R \\ G \\ B \end{bmatrix} = \frac{1}{256} * \begin{bmatrix} RY & RCr & RCb \\ GY & GCr & GCb \\ BY & BCr & BCb \end{bmatrix} * \begin{bmatrix} Y - 16 \\ Cr - 128 \\ Cb - 128 \end{bmatrix}$$

dss-ucE010

The nine coefficient values for color space conversion from UYVY 4:2:2 to RGB are as follows:
RY = 298, RCb = 0, RCr = 409
GY = 298, GCb = (unsigned) (-11), GCr = (unsigned) (-208)
BY = 298, BCb = 517, BCr = 0

12. Configure the video1 channel for VRFB rotation settings as shown in Table 15-87:

Table 15-87. VRFB Rotation Configuration

Rotation	Registers	Value
0 degree	DSS.DISPC_VID1_BA0	VBA0
	DSS.DISPC_VID1_PIXEL_INC	1
	DSS.DISPC_VID1_ROW_INC	(2048 - iw) x ps + 1
90 degrees	DSS.DISPC_VID1_BA0	VBA90
	DSS.DISPC_VID1_PIXEL_INC	1
	DSS.DISPC_VID1_ROW_INC	(2048 - ih) x ps + 1
180 degrees	DSS.DISPC_VID1_BA0	VBA180
	DSS.DISPC_VID1_PIXEL_INC	1
	DSS.DISPC_VID1_ROW_INC	(2048 - iw) x ps + 1
270 degrees	DSS.DISPC_VID1_BA0	VBA270
	DSS.DISPC_VID1_PIXEL_INC	1
	DSS.DISPC_VID1_ROW_INC	(2048 - ih) x ps + 1

In the use case context, the input video format is UYVY for a VGA display (640 * 480). As a consequence, the VRFB configuration is as described in Table 15-88.

Table 15-88. VRFB Rotation Configuration for a VGA Display (UYVY format)

Rotation	Registers	Value
0 degree	DSS.DISPC_VID1_BA0	VBA0
	DSS.DISPC_VID1_PIXEL_INC	1
	DSS.DISPC_VID1_ROW_INC	6273
90 degrees	DSS.DISPC_VID1_BA0	VBA90
	DSS.DISPC_VID1_PIXEL_INC	1
	DSS.DISPC_VID1_ROW_INC	7233
180 degrees	DSS.DISPC_VID1_BA0	VBA180
	DSS.DISPC_VID1_PIXEL_INC	1
	DSS.DISPC_VID1_ROW_INC	6273
270 degrees	DSS.DISPC_VID1_BA0	VBA270
	DSS.DISPC_VID1_PIXEL_INC	1
	DSS.DISPC_VID1_ROW_INC	7233

For more information on VRFB module and settings, see the VRFB use case and tips section in Chapter 11, *Memory Subsystem*.

According to the rotation angle, program the DSS.DISPC_VID1_ATTRIBUTES[13] VIDROTATION and the DSS.DISPC_VID1_ATTRIBUTES[18] VIDROWREPEATENABLE bits as described in Table 15-89.

Table 15-89. Video Rotation Register Settings (UYVY Only)

Rotation	Registers Field Name	Registers Field Value
0 degree	DSS.DISPC_VID1_ATTRIBUTES[13] VIDROTATION	0x0
	DSS.DISPC_VID1_ATTRIBUTES[18] VIDROWREPEATENABLE	0x0
90 degrees	DSS.DISPC_VID1_ATTRIBUTES[13] VIDROTATION	0x1
	DSS.DISPC_VID1_ATTRIBUTES[18] VIDROWREPEATENABLE	0x1
180 degrees	DSS.DISPC_VID1_ATTRIBUTES[13] VIDROTATION	0x2
	DSS.DISPC_VID1_ATTRIBUTES[18] VIDROWREPEATENABLE	0x0
270 degrees	DSS.DISPC_VID1_ATTRIBUTES[13] VIDROTATION	0x3
	DSS.DISPC_VID1_ATTRIBUTES[18] VIDROWREPEATENABLE	0x1

13. Because the video size is in 640 * 480, the scaling unit must be used to downsample to a size of 320 * 240 using a 3 taps configuration.

    The video scaling block for the video pipeline 1 is set in downscaling for horizontal and vertical by setting the following bit field in DSS.DISPC_VIDn_ATTRIBUTES[6:5] :

    - VIDRESIZEENABLE bit field to 0x03
    - VIDHRESIZECONF bit field to 0x1
    - VIDVRESIZECONF bit field to 0x1

14. Program the downscaling of 2 by setting DSS.DISPC_VIDn_FIR to 0x0800 0800.

    All taps coefficients are programmed as in Table 15-90.

    Table 15-90 describes the registers to be configured for the video1 channel configuration step and the use case values.

Table 15-90. Registers Print for Video1 Channel Configuration

Register Name	Address	Value	Value Description
DISPC_VID1_ATTRIBUTES	0x4805 04CC	0x0000 8216 (for 0-degree rotation)	See the programming steps described above for details.
DISPC_VID1_BA0	0x4805 04BC	Base address value	Base address for an image in SDRAM. Depends on the rotation angle (see Table 15-88).
DISPC_VID1_FIFO_THRESHOLD	0x4805 04D0	0x03FF 03BF	FIFO high and low thresholds for DMA request generation
DISPC_VID1_POSITION	0x4805 04C4	0x0000 0000	No X-position and Y-position offsets
DISPC_VID1_SIZE	0x4805 04C8	0x00EF 013F	QVGA display:320*240
DISPC_VID1_PICTURE_SIZE	0x4805 04E4	0x01DF 027F	VGA display: 640*480
DISPC_VID1_CONV_COEF0	0x4805 0530	0x0199 012A	Color space conversion: RCR and RY coefficients
DISPC_VID1_CONV_COEF1	0x4805 0534	0x012A 0000	Color Space Conversion: GY and RCB coefficients
DISPC_VID1_CONV_COEF2	0x4805 0538	0x07F5 0730	Color Space Conversion: GCB and GCR coefficients
DISPC_VID1_CONV_COEF3	0x4805 053C	0x0000 012A	Color Space Conversion: BCR and BY coefficients
DISPC_VID1_CONV_COEF4	0x4805 0540	0x0000 0205	Color Space Conversion: BCB coefficient
DISPC_ROW_INC	0x4805 04D8	Image row increment	Image stride. Depends on the rotation angle (see Table 15-88).
DISPC_PIXEL_INC	0x4805 04DC	0x0000 0001	Pixel increment set to 1 because VRFB module is in charge of rotating the image
DISPC_VID1_FIR	0x4805 04E0	0x0800 0800	Vertical and horizontal ratio for downscaling

Table 15-90. Registers Print for Video1 Channel Configuration (continued)

DISPC_VID1_FIR_COEF_H_0	0x4805 04F0	0x24382400	Horizontal and vertical coefficients
DISPC_VID1_FIR_COEF_HV_0	0x4805 04F4	0x24382400	
DISPC_VID1_FIR_COEF_H_1	0x4805 04F8	0x28371FFE	
DISPC_VID1_FIR_COEF_HV_1	0x4805 04FC	0x28371F04	
DISPC_VID1_FIR_COEF_H_2	0x4805 0500	0x2C361BFB	
DISPC_VID1_FIR_COEF_HV_2	0x4805 0504	0x2C361B08	
DISPC_VID1_FIR_COEF_H_3	0x4805 0508	0x303516F9	
DISPC_VID1_FIR_COEF_HV_3	0x4805 050C	0x3035160C	
DISPC_VID1_FIR_COEF_H_4	0x4805 0510	0x11343311	
DISPC_VID1_FIR_COEF_HV_4	0x4805 0514	0x113433F7	
DISPC_VID1_FIR_COEF_H_5	0x4805 0518	0x1635300C	
DISPC_VID1_FIR_COEF_HV_5	0x4805 051C	0x163530F9	
DISPC_VID1_FIR_COEF_H_6	0x4805 0520	0x1B362C08	
DISPC_VID1_FIR_COEF_HV_6	0x4805 0524	0x1B362CFB	
DISPC_VID1_FIR_COEF_H_7	0x4805 0528	0x1F372804	
DISPC_VID1_FIR_COEF_HV_7	0x4805 052C	0x1F3728FE	
DISPC_VID1_FIR_COEF_V_0	0x4805 05E0	0x00000000	
DISPC_VID1_FIR_COEF_V_1	0x4805 05E4	0x000004FE	
DISPC_VID1_FIR_COEF_V_2	0x4805 05E8	0x000008FB	
DISPC_VID1_FIR_COEF_V_3	0x4805 05EC	0x00000CF9	
DISPC_VID1_FIR_COEF_V_4	0x4805 05F0	0x0000F711	
DISPC_VID1_FIR_COEF_V_5	0x4805 05F4	0x0000F90C	
DISPC_VID1_FIR_COEF_V_6	0x4805 05F8	0x0000FB08	
DISPC_VID1_FIR_COEF_V_7	0x4805 05FC	0x0000FE04	

15.6.5.4.4 Interrupts Enable

The interrupts enable stage is as follows:

1. To clear all display controller IRQs, set the DSS.DISPC_IRQSTATUS register to 0xFFFF.
2. To enable the VSYNC interrupt, set the DSS.DISPC_IRQENABLE[1] VSYNC bit to 1.
3. To enable the VID1FIFOUNDERFLOW interrupt, set the DSS.DISPC_IRQENABLE[10] VID1FIFOUNDERFLOW bit to 1.
4. To enable the ENDVID1WINDOW interrupt, set the DSS.DISPC_IRQENABLE[11] ENDVID1WINDOW bit to 1.
5. To enable the SYNCLOST interrupt, set the DSS.DISPC_IRQENABLE[14] SYNCLOST bit to 1.

Table 15-91 describes the registers to be configured for the interrupts enable step and the use case values.

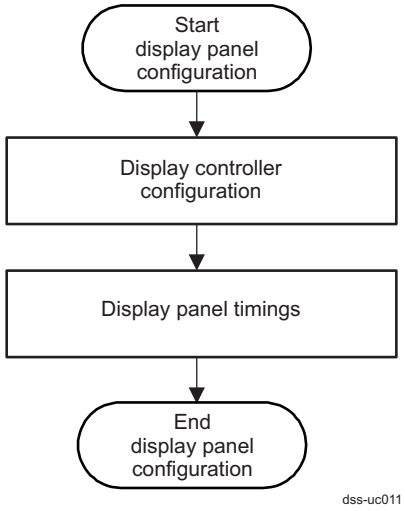
Table 15-91. Registers Print for Interrupts Enable

Register Name	Address	Value	Value Description
DISPC_IRQSTATUS	0x4805 0418	0xFFFF FFFF	Clear all DISPC interrupt requests. This register is automatically reset by hardware.
DISPC_IRQENABLE	0x4805 041C	0x0000 4C02	Enable the VSYNC, VID1FIFOUNDERFLOW, ENDVID1WINDOW, and SYNCLOST IRQs.

15.6.5.4.5 Display Panel Configuration

Figure 15-175 shows the structure of the display panel configuration stage:

Figure 15-175. Display Panel Configuration Flowchart



15.6.5.4.5.1 Display Controller Configuration

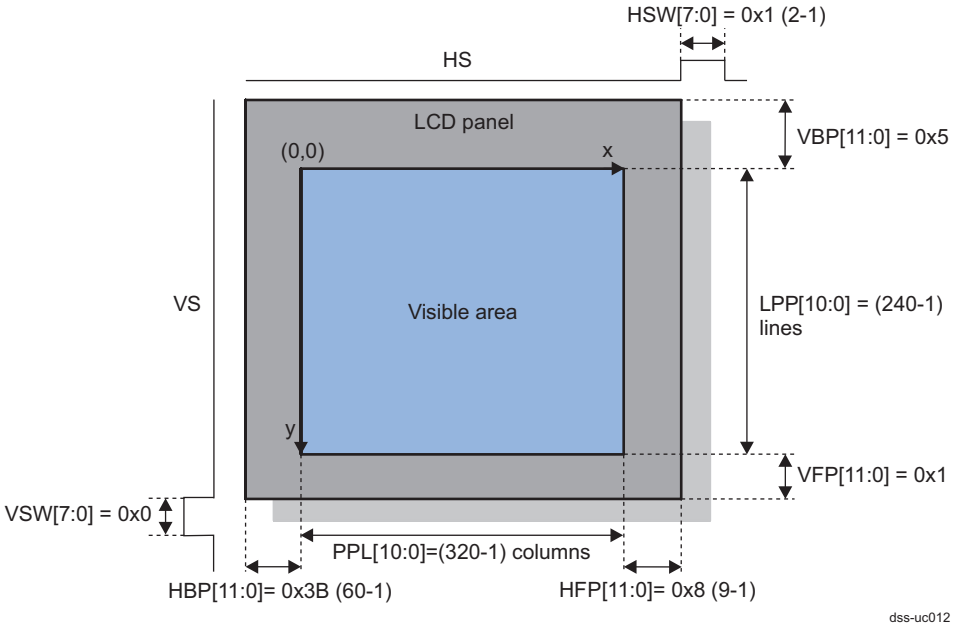
To interface the DISPC with the QVGA LCD panel (320*240), the following settings are required:

1. The QVGA LCD panel is an active matrix: set the DSS.DISPC_CONTROL[3] STNTFT bit to 1 (active matrix display operation mode).
2. Select 16-bit data output by setting the DSS.DISPC_CONTROL[9:8] TFTDATALINES bit field to 0x1.
3. Bypass the RFBI module by setting the DSS.DISPC_CONTROL[16:15] GPOUT[1:0] bit field to 0x3.
4. Select the color panel option by setting the DSS.DISPC_CONTROL[2]MONOCOLOR bit to 0x0.
5. Program the LCD enable signal active high by setting the DSS.DISPC_CONTROL[29] LCDENABLEPOL bit to 1.
6. Disable the free-running pixel clock by setting the DSS.DISPC_CONTROL[27] PCKFREEENABLE bit to 0.
7. Program the number of lines per panel to 240 by setting the DSS.DISPC_SIZE_LCD[26:16] LPP bit field to 0xEF (239 = 240 -1).
8. Program the number of pixels per line to 320 by setting the DSS.DISPC_SIZE_LCD[10:0] PPL bit field to 0x13F (319 = 320 -1).
9. The pallette module is bypassed: set the DSS.DISPC_CONFIG[2:1] LOADMODE bit field to 0x2 (frame data only).

15.6.5.4.5.2 Display Panel Timings

Figure 15-176 details the timing values description for the QVGA LCD panel.

Figure 15-176. QVGA LCD Panel Timings



The following steps define the timing generation of HSYNC/VSYNC signals based on the QVGA LCD panel specification:

1. Program the horizontal front porch to 9 by setting the DSS.DISPC_TIMING_H[19:8] HFP bit field to 0x8 (9-1).
2. Program the horizontal back porch to 60 by setting the DSS.DISPC_TIMING_H[31:20] HBP bit field to 0x3B (60-1).
3. Program the horizontal synchronization pulse width to 2 by setting the DSS.DISPC_TIMING_H[7:0] HSW bit field to 0x1 (2-1).
4. Program the vertical front porch to 1 by setting the DSS.DISPC_TIMING_V[19:8] VFP bit field to 1.
5. Program the vertical back porch to 5 by setting the DSS.DISPC_TIMING_V[31:20] VBP bit field to 5.
6. Program the vertical synchronization pulse width to 0 by setting the DSS.DISPC_TIMING_V[7:0] VSW bit field to 0.
7. For QVGA LCD panel, the line clock signal is active high and inactive low. As a consequence, invert HSYNC by setting the DSS.DISPC_POL_FREQ[13] IHS bit to 0.
8. For QVGA LCD panel, the frame clock signal is active high and inactive low. As a consequence, invert VSYNC by setting the DSS.DISPC_POL_FREQ[12] IVS bit to 0.
9. Drive HSYNC and VSYNC on the falling edge of PCLK by setting the DSS.DISPC_POL_FREQ[16] RF bit to 0 and the the DSS.DISPC_POL_FREQ[17] ONOFF bit to 0.
10. Drive the pixel data on the rising edge of PCLK by clearing the DSS.DISPC_POL_FREQ[14] IPC bit to 0.
11. Set the data enable active high by clearing the DSS.DISPC_POL_FREQ[15] IEO bit to 0.

The visible pixel count is 320 * 240 = 76 800 pixels.

The Total Frame Pixel Count (with the values given above) is (320+8+59+1) x (240+5+1+0) = 95 448 pixels.

The camcorder application displays 60 frame per second.

The pixel clock required for 60 fps is: 95 448 x 60 Hz = 5,72688 MHz.

On the other hand, the pixel clock is defined by the following equation:

$$PixelClock = \frac{Functional\ Clock}{LCD\ factor \times PCD\ factor}$$

dss-ucE013

To program the pixel clock to 5.72688 MHz, software users must set:

1. Set the DSS.DISPC_DIVISOR[23:16] LCD bit field to 0x2.

$$LogicClock = \frac{DSS1_ALWON_FCLK}{2} = 48\ Mhz$$

dss-ucE014

2. Set the DSS.DISPC_DIVISOR[7:0] PCD bit field to 0x8.

$$PixelClock = \frac{LogicClock}{8} = 6\ Mhz$$

dss-ucE015

Table 15-92 describes the registers to be configured for the display panel configuration step and the use case values.

Table 15-92. Registers Print for Display Panel Configuration

Register Name	Address	Value	Value Description
DISPC_CONTROL	0x4805 0440	0x0001 8108	See Section 15.6.5.4.5.1, *Display Controller Configuration*, for more details.
DISPC_SIZE_LCD	0x4805 047C	0x00EF 013F	QVGA display: 320 * 240
DISPC_CONFIG	0x4805 0444	0x0000 0004	Disable the palette
DISPC_TIMING_H	0x4805 0464	0x03B0 0801	Program horizontal timings
DISPC_TIMING_V	0x4805 0468	0x0050 0100	Program vertical timings
DISPC_POL_FREQ	0x4805 046C	0x0000 0000	Program the HSYNC and VSYNC signals for QVGA LCD panel
DISPC_DIVISOR	0x4805 0470	0x0002 0008	Set the divisors to program the PCLK frequency at 6 MHz.

15.6.5.4.6 LCD Enable

The LCD enable step is composed of the following stages:

1. To enable the video1 layer, set the DSS.DISPC_VID1_ATTRIBUTES[0] VIDENABLE bit to 1. The video DMA engine fetches encoded pixels from the system memory only when the video layer is enabled.
2. To enable the display controller output, set the DSS.DISPC_CONTROL[0] LCDENABLE bit to 1.
3. To account for all DISPC configuration done, set the DSS.DISPC_CONTROL[5] GOLCD bit to 1 to indicate that all shadow registers of the video1 pipeline connected to the LCD output are latched by the hardware. The hardware can now update the internal registers at the VFP start period and the image displayed on the panel.

Table 15-93 describes the registers to be configured for the LCD enable step and the use case values.

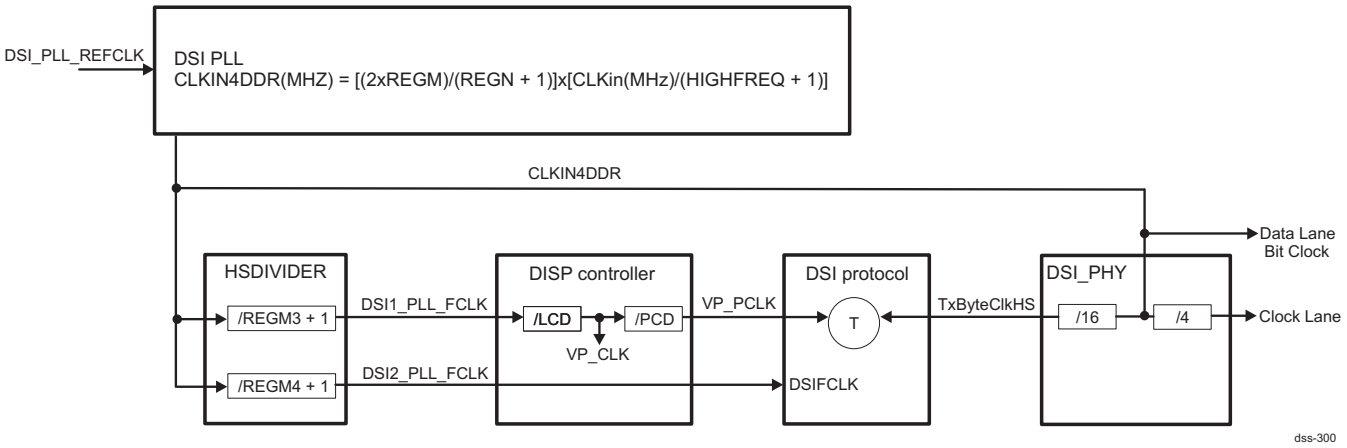
Table 15-93. Registers Print for LCD Enable

Register Name	Address	Value	Value Description
DISPC_CONTROL	0x4805 0440	0x0001 8129	Set DISPC_CONTROL[0] LCDENABLE and DISPC_CONTROL[5] GOLCD bits to 1.
DISPC_VID1_ATTRIBUTES	0x4805 04CC	0x0000 83F7	Enable the DISPC_VID1_ATTRIBUTES[0] VIDENABLE.

15.6.6 How to Configure the DSI PLL in Video Mode

Figure 15-177 shows a global overview of the DSI clock tree when used in video mode.

Figure 15-177. DSI Clock Tree in Video Mode



The settings of the DSI PLL registers can be summarized by 4 equations.

Equation 1

$$N * T_{VP_CLK} = T_L * T_{TxByteClkHS} \quad \text{dss-301} \qquad (15\text{-}18)$$

where

$T_L = T_{HS} + HSA_{DSI} + T_{HE} + HFP_{DSI} + (WC + 6)/NDL + HBP_{DSI}$

N is an integer.

NDL: Number of data lane

WC: Word count or payload in bytes

T_{HS} and T_{HE} are equal to 4/NDL and 0 if they are disabled.

HSA_{DSI} is HSA period in video mode.

T_{HS} is the length of HSYNC start short packet in number of byte clock cycles (TxByteClkHS).

T_{HE} is the length of HSYNC end short packet in number of byte clock cycles (TxByteClkHS).

HBP_{DSI} is HBP period in video mode.

HFP_{DSI} is HFP period in video mode.

Note: HSA_{DSI} timing is not used and does not have to be programmed when HE short packet is not generated.

Equation 2

$$R = T_{TxByteClkHS} / T_{VP_PCLK} \quad \text{dss-302} \qquad (15\text{-}19)$$

To synchronize DISPC and DSI Protocol Engine, users should follow the ratio T between TxByteClkHS and VP_PCLK as listed in Table 15-94.

Table 15-94. Ratio R

Number of data lanes	Pixel format	Ratio R
1	16-bits pixel	1/2

Table 15-94. Ratio R (continued)

Number of data lanes	Pixel format	Ratio R
1	18-bits pixel	4/9
1	24-bits pixel	1/3
2	16-bits pixel	1
2	18-bits pixel	8/9
2	24-bits pixel	2/3

All cases are covered by:

$F_{VP_PCLK} * bits_per_pixel = F_{TxByteClkHS} * NDL * 8$

dss-310

Equation 3

$$(HSA_{DISPC} + HFP_{DISPC} + PPL + HBP_{DISPC}) * T_{VP_PCLK} = (4/NDL + HFP_{DSI} + (WC + 6)/NDL + HBP_{DSI}) * T_{TxByteClkHS}$$ dss-303 (15-20)

Equation 4

$$HFP_{DSI} = ((HFP_{DISPC} * bits_per_pixel) / (NDL * 8)) - (2 / NDL)$$ dss-309 (15-21)

Example

The desired performances are:

- Clock lane at 150 MHz
- RGB24-888
- 1-data lane
- LCD size 480*640 with $HSA_{_DISP} = HFP_{_DISP} = HBP_{_DISP}$ =20, $VSA_{_DISP} = VFP_{_DISP} = VBP_{_DISP}$ =2

Step 1. **Determine REGM and REGN**

To obtain correct stability, Fint should be kept between 0.75 MHz and 2.1 MHz. In this case, Fint is maintained at 2 MHz. For more information, refer to the DSI PLL programming model.

$$REGN = (F_{DSI_PLL_REFCLK} / F_{int}) – 1$$

$$REGN = 12$$

$$REGM = (REGN+1)*F_{CLKIN4DDR} / (2 * F_{DSI_PLL_REFCLK})$$

$$REGM = 150$$ dss-318

Where DSS2_ALWON_FCLK= 26 MHz is used as a reference clock for $F_{DSI_PLL_REFCLK}$

Step 2. **Determine VP_PCLK and TxByteClkHS clocks.**

TxByteClkHS frequency is equal to 37.5 MHz. With ratio R equal to 1/3, VP_PCLK frequency is equal to 12,5 MHz. The frame rate can be estimated by:

$$Frame\ rate = F_{VP_PCLK} / (HSA_{DISPC} + HFP_{DISPC} + PPL + HBP_{DISPC}) * (VSA_{DISPC} + VFP_{DISPC} + LPP + VBP_{DISPC})$$

$$Frame\ rate = 12{,}5\ MHz / (540) * (646)$$

$$Frame\ rate = 35{,}83\ frame/sec$$ dss-323

Step 3. **Determine LCD, PCD and REGM3**

$$T_{CLKIN4DDR} = T_{TxByteClkHS} / 16 = T_{VP_PCLK} / ((REGM3 + 1) * LCD * PCD)$$

$$((REGM3 + 1) * LCD * PCD) = 16 * 3$$ dss-315

If LCD and PCD are set to 1 and 3 respectively, REGM3 is equal to 15.

Step 4. **Verify N as integer**

Firstly, TL must be determined

$$(HSA_{DISPC} + HFP_{DISPC} + PPL + HBP_{DISPC}) * T_{VP_PCLK} / T_{TxByteClkHS} = T_L = 1620$$ dss-316

From **Equation 1**,

$N*T_{VP_CLK} = T_L*T_{TxByteClkHS}$

$N= T_{TxByteclkHS}*T_L/T_{VP_CLK}= T_L/(R*PCD)$

N = 14580

N is an integer.

Step 5. **Determine HFP and HBP of the DSI protocol engine**

From **Equation 3**,

$$(HSA_{DISPC} + HFP_{DISPC} + PPL + HBP_{DISPC}) * T_{VP_PCLK} / T_{TxByteClkHS}) - (4/NDL + (WC + 6)/NDL) = HFP_{DSI} + HBP_{DSI}$$

$$HFP_{DSI} + HBP_{DSI} = 170$$

dss-317

From **Equation 4**,

$HFP_{DSI} = ((HFP_{DISPC} * bits_per_pixel) / (NDL * 8)) - (2 / NDL)$

$$HFP_{DSI} = 58$$

$$HBP_{DSI} = 170 - 58 = 112$$

dss-321

15.6.7 DSI Video Mode Using the DISPC Video Port

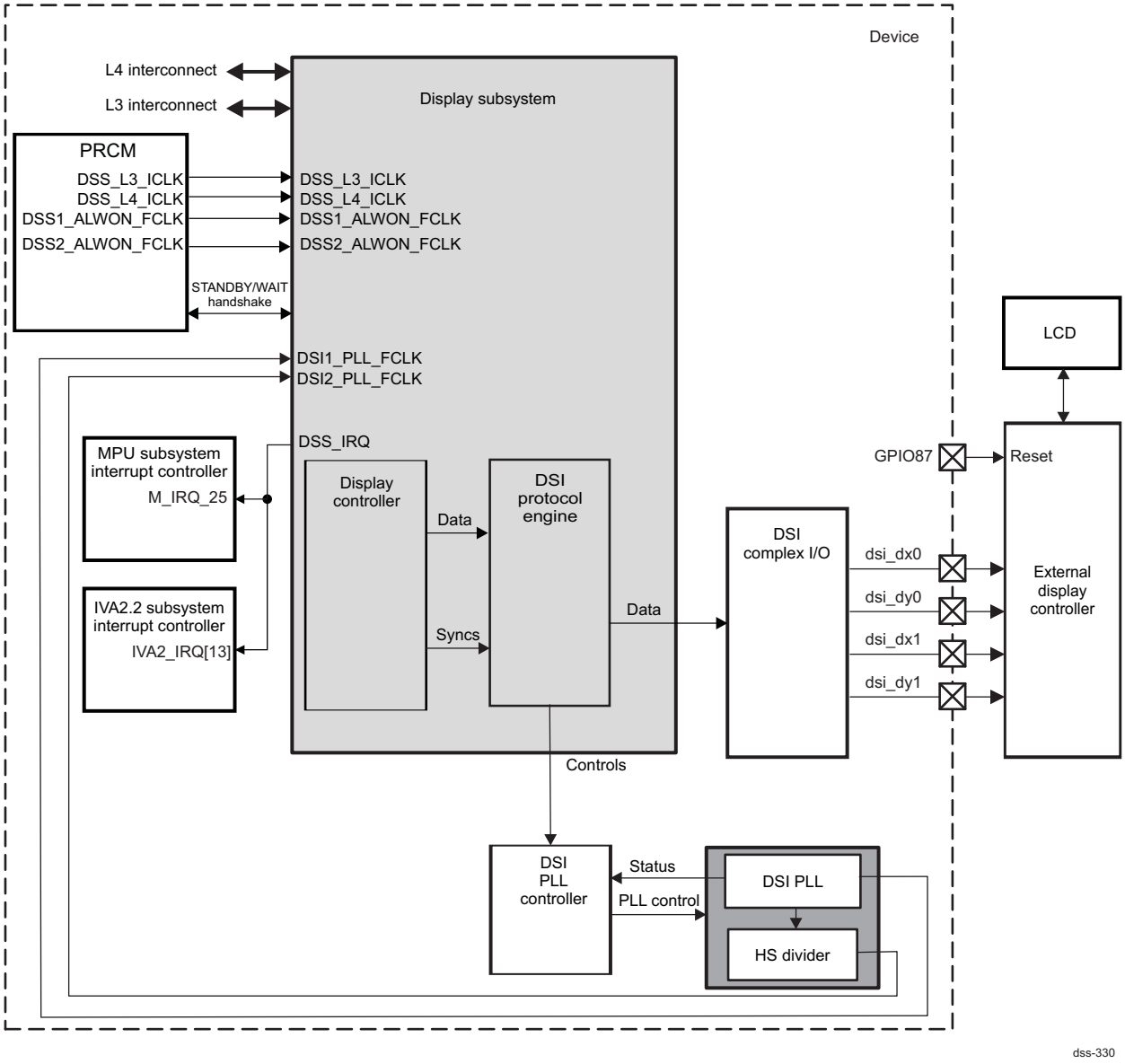
This section details the basic programming model of video mode using the DISPC video port.

The DSI interface is connected to an external MIPI display controller and the following parameters are used:

- 1 data lane: NDL = 1
- Clock lane at 150 MHz (DSI_DDR_CLK)
- LCD size is 640 × 480:
  - 480 pixels per line (PPL)
  - 680 lines per panel (LPP)
- Display controller input format: YUV
- Display controller output format: RGB888, that is 24 bits per pixel (BPP)
- Word Count: WC = 3 × PPL
- DSS2_ALWON_FLCK=26 MHz is used as a reference clock for DSI PLL
- Virtual channel 0 (VC0) is used for video mode.
- Interleaving is not used.
- It is assumed that all modules used in these programming models are in after-POR state.

Figure 15-178 is an overview of the connections in the display subsystem.

Figure 15-178. Overview



The steps listed in Table 15-95 are described in the following sections.

Table 15-95. Main Steps

Steps	Section
Configure DSS clocks	Section 15.6.7.1, *Display Subsystem Clock Configuration*
Configure the DSI and DSI PLL	Section 15.6.7.2, *Configure DSI, DSI PLL and Complex I/O*
Configure the external MIPI display controller	Section 15.6.7.3, *Initialization of the External MIPI Display Controller*
Configure the DISPC	Section 15.6.7.4, *Configure the DISPC*
Enable video mode using the DISPC video port	Section 15.6.7.5, *Enable Video Mode Using the DISPC Video Port*

The programming model must be followed in the order of the following sections.

15.6.7.1 Display Subsystem Clock Configuration

Table 15-96 lists the steps required to enable the clocks.

Table 15-96. PRCM Registers

Steps	Registers	Value
Set the divided DPLL value for DSS1.	CM_CLKSEL_DSS[4:0] CLKSEL_DSS1	0x9
Disable autoidle mode.	CM_AUTOIDLE_DSS[31:0]	0x0
Domain sleep is disabled.	CM_SLEEPDEP_DSS[31:0]	0x0
Automatic transition between active and inactive are disabled.	CM_CLKSTCTRL_DSS[31:0]	0x0
Enable DSS1, DSS2 and TV clock (DSS1_ALWON_FCLK, DSS2_ALWON_FCLK and TV_CLK). TV_CLK is only need for correct Reset.	CM_FCLKEN_DSS[31:0]	0x7
Enable the subsystem interface clock (DSS_L3_ICLK and DSS_L4_ICLK).	CM_ICLKEN_DSS[31:0]	0x1

15.6.7.2 Configure DSI, DSI PLL and Complex I/O

15.6.7.2.1 Reset DSI Modules

Table 15-97 lists the steps required to reset the DSI modules.

Table 15-97. Resets

Steps	Registers	Value
Reset IRQ status.	DSI_IRQSTATUS[31:0]	0x0
OCP and functional clock are maintained during wakeup, smart idle, and reset DSI.	DSI_SYSCONFIG[31:0]	0x312
Wait until RESET_DONE ≠ 0.	DSI_SYSSTATUS[0] RESET_DONE	Read 0x1

15.6.7.2.2 Set Up DSI DPLL

Table 15-98 lists the steps required to configure DSI PLL.

Table 15-98. DSI PLL Configuration Registers

Steps	Registers	Value
Turn on PLL and HSDIVIDER.	DSI_CLK_CTRL[31:30] PLL_PWR_CTRL	0x2
Wait until PLL_PWR_STATUS = 0x2.	DSI_CLK_CTRL[29:28] PLL_PWR_STATUS	Read 0x2
See the calculation following this table.	DSI_PLL_CONFIGURATION1[26:23] DSIPROTO_CLK_DIV	5
See the calculation following this table.	DSI_PLL_CONFIGURATION1[22:19] DSS_CLOCK_DIV	15
See the calculation following this table.	DSI_PLL_CONFIGURATION1[18:8] DSI_PLL_REGM	150
See calculation following this table.	DSI_PLL_CONFIGURATION1[7:1] DSI_PLL_REGN	12
Enable PLLStopMode.	DSI_PLL_CONFIGURATION1[0]	0x1
The DSI protocol engine clock divider, DSS clock divider, CLKIN4DDR, and PLL reference clock are enabled. PLL internal reference frequency is between 1.75 and 2.1MHz.	DSI_PLL_CONFIGURATION2[31:0]	0x5600E
Manual mode	DSI_PLL_CONTROL[31:0]	0x0

Table 15-98. DSI PLL Configuration Registers (continued)

Steps	Registers	Value
Request PLL locking sequence.	DSI_PLL_GO[0] DSI_PLL_GO	0x1
Read until DSI_PLL_GO = 0	DSI_PLL_GO[0] DSI_PLL_GO	Read 0x0
PLL is locked.	DSI_PLL_STATUS[1] DSI_PLL_LOCK	Read 0x1
Turn on PLL and HSDIVIDER; the DSI functional clock > 30-MHz sync is rising/rising; the DSIStopClk signal is automatically asserted/deasserted; the L3_ICLK clock to the DSI complex I/O is not gated; and LPCLKDIVISOR = 8	DSI_CLK_CTRL[31:0]	0x8024 4008

1. Calculate the divider value for the DSI protocol engine clock source:

   RegM4 = FCLKIN4DDR/DSI2_PLL_FCLK - 1

   FCLKIN4DDR = 4 x FCLKIN

   RegM4 = 5

2. Determine LCD, PCD, and REGM3:
   Calculate the divider value for DSS clock source: Same as Step 3.

$$RegM3 = ((BPP \times 2)/(DISPC_LCD \times DISPC_PCD \times NDL)) - 1$$

$$RegM3 = 15$$

dss-E127

3. Calculate N Divider for PLL:

$$FCLKIN4DDR = FCLKIN \times 4$$

$$RegN = (FDSI_PLL_REFCLK / F\,INt) - 1$$

$$FDSI_PLL_REFCLK = 26\ MHz\ (system\ clock)$$

$$Fint = 2\ MHz\ (reduce\ PLL\ lock\ time)$$

$$RegN = 12$$

dss-E128

4. Calculate M divider for PLL:

$$RegM = ((RegN + 1) \times (FCLKIN4DDR/(2 \times FDSI_PLL_REFCLK)))$$

$$FCLKIN4DDR = 4 \times 150\ MHz$$

$$RegM = 150$$

dss-E129

15.6.7.2.3 Switch to DSI PLL Clock Source

Select the DSI_PLL1 clock as the DISPC functional clock, and select the DSI_PLL2 clock as the DSI functional clock: Set DSS_CONTROL to 0x3.

15.6.7.2.4 Set Up DSI Protocol Engine

15.6.7.2.4.1 Set Up DSI Control Registers

Table 15-99 lists the steps to set up the DSI control registers. Table 15-100 lists the steps to set up the DSI complex I/O registers.

Table 15-99. DSI Control Registers

Steps	Registers	Value
Enable SYNCLOST event.	DSI_IRQENABLE[31:0]	0x4 0000
Set PACKET_SENT_IRQ_EN.	DSI_VC0_IRQENABLE[31:0]	0x4

Table 15-99. DSI Control Registers (continued)

Steps	Registers	Value
While the HSYNC START pulse is detected, the associated short packet HSYNC START is generated.	DSI_CTRL[17] VP_HSYNC_START	0x1
While the VSYNC START pulse is detected, the associated short packet VSYNC START is generated.	DSI_CTRL[15] VP_VSYNC_START	0x1
Set the trigger reset mode to *immediate.*	DSI_CTRL[14] TRIGGER_RESET_MODE	0x1
Activate two line buffers.	DSI_CTRL[13:12] LINE_BUFFER	0x2
HSYNC signal on the video port is active high.	DSI_CTRL[10] VP_HSYNC_POL	0x1
VSYNC on the video port is active high.	DSI_CTRL[11] VP_VSYNC_POL	0x1
Set the Data Enable signal on the video port as active high.	DSI_CTRL[9] VP_DE_POL	0x1
Set the size of the video port data bus for the RGB format: 0x2 for RGB 888.	DSI_CTRL[7:6] VP_DATA_BUS_WIDTH	0x2
VP_CLK_RATIO is not used if video port is used to provide data in video mode	DSI_CTRL[4] VP_CLK_RATIO	0x0
Set the arbitration scheme for granting the VC pending ready requests in the TX FIFO as Sequential Scheme.	DSI_CTRL[3] TX_FIFO_ARBITRATION	0x1
Enable the ECC check for the received header.	DSI_CTRL[2] ECC_RX_EN	0x1

Table 15-100. DSI Complex I/O Registers

Steps	Registers	Value
GOBIT, complex I/O power ON, select data, and clock position	DSI_COMPLEXIO_CFG1	0x4800 0032
Clear Reg	DSI_COMPLEXIO_IRQSTATUS	0xC3F39CE7
Disable IRQ	DSI_COMPLEXIO_IRQENABLE	0x0
Enable I/F	DSI_CTRL[0] IF_EN	0x1
Disable I/F	DSI_CTRL[0] IF_EN	0x0
Wait until IF_EN = 0	DSI_CTRL[0] IF_EN	Read 0x0
Enable Low Power clock	DSI_CLK_CTRL[20] LP_CLK_ENABLE	0x1
Reset is done.	DSI_COMPLEXIO_CFG1[29] RESET_DONE	Read 0x1
Power control is on.	DSI_COMPLEXIO_CFG1[26:25] PWR_STATUS	Read 0x1
Reset is complete.	DSI_SYSSTATUS[0] RESETDONE	Read 0x1

15.6.7.2.4.2 Configure DSI Timing and Virtual Channels

Table 15-101 lists the steps to configure DSI timing and the virtual channels.

Table 15-101. DSI Timing Registers

Steps	Registers	Value
STOP_STATE_COUNTER_IO = 0x999	DSI_TIMING1[31:0]	0x0000 0999
HS_TX_TO_X8, HS_TX_TO_COUNTER = 0x0FD2, LP_RX_TO_X16, LP_RX_TO_COUNTER = 0x00CD	DSI_TIMING2[31:0]	0x2FD2 40CD
(HSA<<24)|(HFP<<12)|HBP	DSI_VM_TIMING1[31:0]	0x0000 700A
(WINDOW_SYNC<<24)|(VSA<<16)|(DSI_VFP <<8)|VBP	DSI_VM_TIMING2[31:0]	0x0401 0101
(TL<<16)|DSI_VACT	DSI_VM_TIMING3[31:0]	0x05BB 0280

Table 15-101. DSI Timing Registers (continued)

Steps	Registers	Value
(ENTER_HS_MODE_LATENCY<<16)|EXIT_HS_MODE_LATENCY	DSI_VM_TIMING7[31:0]	0x0000 C000A
DDR_CLK_PRE<<8|DDR_CLK_POST	DSI_CLK_TIMING[31:0]	0x0000 0F0B
HS speed, ECC generation for the Transmit, Enable CheckSum generation for the Payload.	DSI_VC0_CTRL[31:0]	0x2080 0390

- Freq TxByteClkHS :

$$FHSB = FCLKIN4DDR / 16$$

$$FVPP = FCLKIN4DDR / ((RegM3 + 1) \times DISPC_LCD * DISPC_PCD)$$

$$FVP = FCLKIN4DDR / (RegM3 + 1)$$

dss-E130

- Length of the line in video mode in Nb of byte clock cycles (TxByteClkHS):

$$TL = FHSB / FVPP \times (DISPC_HSA + DISPC_HFP + PPL + DISPC_HBP)$$

$$TL1f = (BPP / (8 \times NDL)) \times (DISPC_HSA + DISPC_HFP + PPL + DISPC_HBP)$$

dss-E131

- Blanking periods (HBP + HFP) in DSI are calculated based on the following formula:

$$(DISPC_HSA + DISPC_HBP + PPL + DISPC_HFP) \times Fppi = (HS + HBP + ((WC + 6) / NDL) + HFP) \times Fvp$$

$$HBP + HFP = (TVPP / THSB) \times (DISPC_HSA + DISPC_HFP + PPL + DISPC_HBP) - (HS + (WC + 6) / NDL))$$

$$HBPplusHFP = (FHSB / FVPP) \times (DISPC_HSA + DISPC_HFP + PPL + DISPC_HBP) - (HS + WC + 6) / NDL$$

$$HBPplusHFPf = ((FHSB / FVPP) \times (DISPC_HSA + DISPC_HFP + PPL + DISPC_HBP)) - ((HS + WC + 6) / NDL)$$

$$HFP = (DISPC_HFP \times BPP) / (NDL \times 8) - (2 / NDL)$$

$$HBP = HBPplusHFP - HFP$$

dss-E132

15.6.7.2.5 Configure DSI_PHY

For the calculation of DSI_PHY timing, see Table 15-102. See Section 15.4.3.2, *Clock Requirements*, for details on timing calculation.

Table 15-102. Calculate DSI_PHY Timing

Steps	Registers	Value
See Section 15.4.3.2	DSI_PHY_CFG0[31:24] THS_PREPARE	ceil(70 ns/DDR clock period) + 2
	DSI_PHY_CFG0[23:16] THS_PREPARE_THS_ZERO	ceil(175 ns/DDR clock period) + 2
	DSI_PHY_CFG0[7:0] THS_EXIT	ceil(145 ns/DDR clock period)
	DSI_PHY_CFG0[15:8] THS_TRAIL	ceil(60 ns/DDR clock period) + 5
	DSI_PHY_CFG2[7:0] TCLK_PREPARE	ceil(65 ns/DDR clock period)
	DSI_PHY_CFG1[7:0] TCLK_ZERO	ceil(260 ns/DDR clock period)
	DSI_PHY_CFG1[15:0] TCLK_TRAIL	ceil(60 ns/DDR clock period) + 2
	DSI_PHY_CFG1[22:16] TLPX_HALF	ceil(26.5 ns/DDR clock period)

Note: Keep Reserved bits at reset value in the DSI_PHY_CFG1 and DSI_PHY_CFG2 registers.

15.6.7.2.6 Drive Stop State

Table 15-103 lists the steps to Drive Stop State.

Table 15-103. Drive Stop State

Steps	Registers	Value
Force TX stop mode	DSI_TIMING[15] FORCE_TX_STOP_MODE_IO	0x1
Wait until FORCE_TX_STOP_MODE_IO = 0	DSI_TIMING[15] FORCE_TX_STOP_MODE_IO	Read 0x0

15.6.7.3 Initialization of the External MIPI Display Controller

1. Wait for the external MIPI display controller initialization after power up.
   In this example, the external MIPI display controller is reset using GPIO 87.
2. Configure the external MIPI display controller.

15.6.7.4 Configure the DISPC

15.6.7.4.1 Reset DISPC

Table 15-104 lists the step sequence to reset the DISPC.

Table 15-104. Reset DISPC

Steps	Registers	Value
Reset the DISPC.	DISPC_SYSCONFIG[1] SOFTRESET	0x1
DISPC is reset.	DISPC_SYSCONFIG[0] RESETDONE	Read 0x1
No standby: MStandby is never asserted.	DISPC_SYSCONFIG [13:12] MIDLEMODE	0x1
Disable idle mode.	DISPC_SYSCONFIG [4:3] SIDLEMODE	0x1
Disable all interrupts.	DISPC_IRQENABLE[31:0]	0x0

15.6.7.4.2 Configure DISPC Timing, Window, and Color

Table 15-105 lists the steps to configure the DISPC registers. Table 15-106 lists the steps to configure the color space coefficient registers.

Table 15-107 lists the steps to configure DISPC_CONTROL.

Table 15-105. Configure DISPC Registers

Steps	Registers	Value
Set horizontal timings.	DISPC_TIMING_H	(HBP – 1<<20)|(HFP – 1<<8)|HSA – 1
Set vertical timing.	DISPC_TIMING_V	(HBP – 1<<20)|(HFP – 1<<8)|HSA – 1
Set LCD – PCD display controller and pixel clock divisor.	DISPC_DIVISOR	LDC = %1 PXLCLK = %4
Set the size of the LCD – 1.	DISPC_SIZE_LCD	(LPP – 1 + DSI_VFP)<<16|(PPL – 1)
Set default RGB value when there is no data.	DISPC_DEFAULT_COLOR0	0XFF

Table 15-105. Configure DISPC Registers (continued)

Steps	Registers	Value
Set video FIFO height and low threshold.	DISPC_VID1_FIFO_THRESHOLD	0xfc00c0
Set the X Y location of the upper left pixel (0 = 0) to the upper left corner.	DISPC_VID1_POSITION	0x0
Set size of the window.	DISPC_VID1_SIZE	(LPP – 1) <<16|(PPL – 1)
Set the size of the picture.	DISPC_VID1_PICTURE_SIZE	(LPP – 1) <<16|(PPL – 1)
Set the input address picture.	DISPC_VID1_BA0	0x–

Table 15-106. Configure Color Space Coefficient Registers

Comments	Registers	Value
RCR and RY coefficients	DISPC_VID1_CONV_COEF0	0X0199012A
GY and RCB coefficients	DISPC_VID1_CONV_COEF1	0X012A0000
GCB and GCR coefficients	DISPC_VID1_CONV_COEF2	0X079C0730
BCR and BY coefficients	DISPC_VID1_CONV_COEF3	0X0000012A
BCB coefficient	DISPC_VID1_CONV_COEF4	0X00000205

Table 15-107. Configure DISPC_CONTROL

Comments	Registers	Value
Enable pixel clock free-running.	DISPC_CONTROL[27] PCLKFREEENABLE	0x1
I/O pad mode selection: Bypass	DISPC_CONTROL[16] GPOUT1	0x1
I/O pad mode selection: Bypass	DISPC_CONTROL[15] GPOUT0	0x1
Normal mode selected	DISPC_CONTROL[11] STALLMODE	0x0
3 for RGB888	DISPC_CONTROL[9:8] TFTDATALINES	0x3
The hardware has finished updating the internal shadow registers of the pipeline(s) connected to the LCD output using the user values. The hardware resets the bit when the update completes.	DISPC_CONTROL[5] GOLCD	0x0
Active matrix display operation mode	DISPC_CONTROL[3] STNTFT	0x1
LCD interface is disabled.	DISPC_CONTROL[0] LCDENABLE	0x0

The input image is converted from UYVY into RGB (see Table 15-108) :

Table 15-108. Configure DISPC_VID1_ATTRIBUTES

Comments	Registers	Value
Row of VIDn will not be read twice.	DISPC_VID1_ATTRIBUTES[18] VIDROWREPEATENABLE	0x0
8 x 32-bit bursts	DISPC_VID1_ATTRIBUTES[15:14] VIDBURSTSIZE	0x1
Full range selected: Y is not modified before the color space conversion.	DISPC_VID1_ATTRIBUTES[11] VIDFULLRANGE	0x1
Enable color space conversion CbYCr to RGB.	DISPC_VID1_ATTRIBUTES[9] VIDCOLORCONVENABLE	0x1
UYVY	DISPC_VID1_ATTRIBUTES[4:1] VIDFORMAT	0xB
Video disabled	DISPC_VID1_ATTRIBUTES[0] VIDENABLE	0x0

15.6.7.5 Enable Video Mode Using the DISPC Video Port

Table 15-109 lists the steps to enable DISPC to send frames continuously.

Table 15-109. Enable DISPC

Steps	Registers	Value
Set up long packet header	DSI_VC0_LONG_PACKET_HEADER[31:0]	0x0005 A03E
Enable VC0.	DSI_VC0_CTRL[1] VC_EN	0x1
Enable IF.	DSI_CTRL[0] IF_EN	0x1
Wait until IF_EN ≠ 0.	DSI_CTRL[0] IF_EN	Read 0x0
Enable VID1.	DISPC_VID1_ATTRIBUTES[0] VIDENABLE	0x1
Enable LCD interface.	DISPC_CONTROL[0] LCDENABLE	0x1
Enable GOLCD.	DISPC_CONTROL[5] GOLCD	0x1
Wait until GOLCD = 0.	DISPC_CONTROL[5] GOLCD	Read 0x0

15.6.8 DSI Command Mode Using the DISPC Video Port

This section presents some generic use cases and tips for setting the modules of the display subsystem.

15.6.8.1 Display Subsystem Use Cases and Tips

This section explains the basic programming model of command mode using the DISPC video port.

The DSI interface is connected to an external MIPI DISPC using the following parameters:

- One data lane: NDL = 1
- Clock lane at 150 MHz (DSI_DDR_CLK)
- LCD size is 640 × 480:
  - 480 PPL
  - 680 LPP
- DISPC input format: YUV
- DISPC output format: RGB888 (24 BPP)
- Word Count: WC = 3 × PPL
- DSS2_ALWON_FLCK = 26 MHz used as a reference clock for DSI PLL
- Virtual channel 1 (VC1) used for command mode to configure the externel DISPC, and virtual channel 0 (VC0) used to send data
- Automatic TE used for synchronization
- Interleaving not used
- It is assumed that all modules used in these programming models are in after-POR state.

Figure 15-179 is an overview of the connections in the display subsystem.

Figure 15-179. Overview

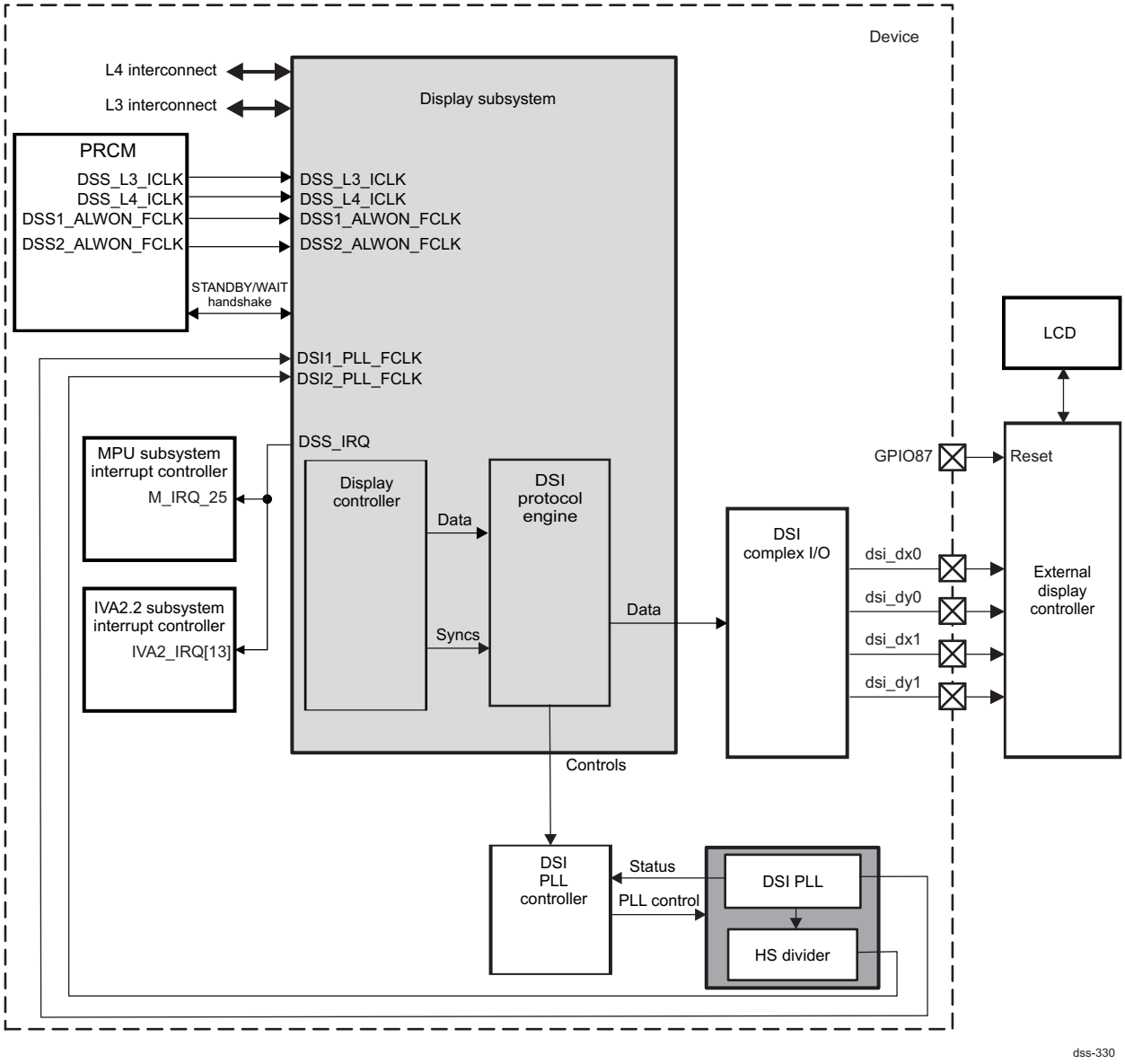


Table 15-110 lists the steps of the main sequence and the sections that describe them.

Table 15-110. Main Sequence

Steps	Register/Bit Field/Programming Model
Configure DSS clocks.	Section 15.6.8.1.1, *Configure DSS Clocks at the PRCM Module*
Configure the DSI protocol engine and DSI PLL.	Section 15.6.8.1.2, *Configure DSI Protocol Engine, DSI PLL, and Complex I/O*
Configure the external MIPI display controller.	Section 15.6.8.1.3, *Initialization of the External MIPI Display Controller*
Configure the DISPC.	Section 15.6.8.1.4, *Configure the DISPC*
Enable command mode using the DISPC video port.	Section 15.6.8.1.5, *Enable Command Mode Using the DISPC Video Port*
Send a frame data to LCD panel using automatic TE.	Section 15.6.8.1.6, *Send Frame Data to LCD Panel Using Automatic TE*

The programming model must follow the step sequence.

15.6.8.1.1 Configure DSS Clocks at the PRCM Module

Table 15-111 lists the steps required to enable the clocks.

Table 15-111. Configure DSS Clocks at the PRCM Module

Steps	Register/Bit Field/Programming Model	Value
Set the divided DPLL value for DSS1.	CM_CLKSEL_DSS[4:0] CLKSEL_DSS1	0x9
Disable autoidle mode.	CM_AUTOIDLE_DSS[0] AUTO_DSS	0x0
Domain sleep is disabled.	CM_SLEEPDEP_DSS[2:0] EN_IVA2, EN_MPU, EN_CORE	0x0
Disable automatic transition.	CM_CLKSTCTRL_DSS[1:0] CLKTRCTRL_DSS	0x0
Enable the DSS1_ALWON_FCLK, DSS2_ALWON_FCLK, and TV_CLK functional clocks.(1)	CM_FCLKEN_DSS[2:0] EN_TV, EN_DSS2, EN_DSS1	0x7
Enable the DSS_L3_ICLK and DSS_L4_ICLK interface clocks.	CM_ICLKEN_DSS[0] EN_DSS	0x1

(1) TV_CLK is required only for a correct reset.

15.6.8.1.2 Configure DSI Protocol Engine, DSI PLL, and Complex I/O

Table 15-112 lists the steps to configure the DSI protocol engine, DSI PLL, and complex I/O and the sections that describe the sequence.

Table 15-112. Configure DSI Protocol Engine, DSI PLL, and Complex I/O

Steps	Register/Bit Field/Programming Model
Reset DSI modules.	Section 15.6.8.1.2.1, *Reset DSI Modules*
Configure DSI DPLL.	Section 15.6.8.1.2.2, *Configure DSI PLL*
Switch to DSI PLL clock source.	Section 15.6.8.1.2.3, *Switch to DSI PLL Clock Source*
Configure DSI protocol engine.	Section 15.6.8.1.2.4, *Configure DSI Protocol Engine*
Configure DSI_PHY.	Section 15.6.8.1.2.5, *Configure DSI_PHY*
Drive stop state.	Section 15.6.8.1.2.6, *Drive Stop State*

15.6.8.1.2.1 Reset DSI Modules

Table 15-113 lists the steps required to reset the DSI modules.

Table 15-113. Reset DSI Modules

Steps	Register/Bit Field/Programming	Value
Clear DSI IRQ status.	DSI_IRQSTATUS[31:0]	0x0
Maintain interface and functional clock during wakeup.	DSI_SYSCONFIG[9:8] CLOCKACTIVITY	0x3
Enable smart-idle for power management.	DSI_SYSCONFIG[4:3] SIDLEMODE	0x2
Set DSI reset.	DSI_SYSCONFIG[1] SOFT_RESET	0x1
Wait until RESET_DONE = 1.	DSI_SYSSTATUS[0] RESET_DONE	

15.6.8.1.2.2 Configure DSI PLL

Table 15-114 lists the steps required to configure the DSI PLL.

Table 15-114. Configure DSI PLL

Steps	Register/Bit Field/Programming	Value
Enable PLL and HSDIVIDER.	DSI_CLK_CTRL[31:30] PLL_PWR_CMD	0x2
Wait until PLL_PWR_STATUS = 0x2.	DSI_CLK_CTRL[29:28] PLL_PWR_STATUS	
Set the REGM4 value (see).	DSI_PLL_CONFIGURATION1[26:23] DSIPROTO_CLK_DIV	5
Set the REGM3 value (see).	DSI_PLL_CONFIGURATION1[22:19] DSS_CLOCK_DIV	15
Set the REGN value (see).	DSI_PLL_CONFIGURATION1[7:1] DSI_PLL_REGN	12
Set the REGM value (see).	DSI_PLL_CONFIGURATION1[18:8] DSI_PLL_REGM	150
Enable PLL STOPMODE.	DSI_PLL_CONFIGURATION1[0] DSI_PLL_STOPMODE	0x1
Set the PLL internal reference frequency range between 1.75 and 2.1 MHz.	DSI_PLL_CONFIGURATION2[4:1] DSI_PLL_FREQSEL	0x7
Enable PLL reference clock control.	DSI_PLL_CONFIGURATION2[13] DSI_PLL_REFEN	0x1
Enable CLKIN4DDR control.	DSI_PLL_CONFIGURATION2[14] DSI_PHY_CLKINEN	0x1
Enable DSS clock divider.	DSI_PLL_CONFIGURATION2[16] DSS_CLOCK_EN	0x1
Enable DSI protocol engine clock divider.	DSI_PLL_CONFIGURATION2[18] DSI_PROTO_CLOCK_EN	0x1
Enable DSI configuration update with DISPC_UPDATE_SYNC.	DSI_PLL_CONTROL[0] DSI_PLL_AUTOMODE	0x0
Start PLL locking sequence.	DSI_PLL_GO[0] DSI_PLL_GO	0x1
Wait until DSI_PLL_GO = 0.	DSI_PLL_GO[0] DSI_PLL_GO	
Check whether PLL is locked.	DSI_PLL_STATUS[1] DSI_PLL_LOCK	0x1
Set the LP mode clock ratio.	DSI_CLK_CTRL[12:0] LP_CLK_DIVISOR	0x8
Set L3_ICLK clock to the DSI complex I/O to not gated.	DSI_CLK_CTRL[14] CIO_CLK_ICG	0x1
Enable the automatic assertion/deassertion of the DSIStopClk signal.	DSI_CLK_CTRL[18] HS_AUTO_STOP_ENABLE	0x1
Specify that the DSI functional clock is higher than 30 MHz with a synchronization rising/rising.	DSI_CLK_CTRL[21] LP_RX_SYNCHRO_ENABLE	0x1
Turn on PLL and HSDIVIDER.	DSI_CLK_CTRL[31:30] PLL_PWR_CMD	0x2

1. Calculate the divider value for the DSI protocol engine clock source:

$$RegM4 = FCLKIN4DDR / FDSI_PLL_REFCLK - 1$$

$$FCLKIN4DDR = 4 \times FCLKIN$$

$$RegM4 = 5$$

dss-E126

2. Determine LCD, PCD, and REGM3:
   Calculate the divider value for the DSS clock source: Same as Step 3.

$$RegM3 = ((BPP \times 2)/(DISPC_LCD \times DISPC_PCD \times NDL)) - 1$$

$$RegM3 = 15$$

dss-E127

3. Calculate N divider for PLL:

$$FCLKIN4DDR = FCLKIN \times 4$$

$$RegN = (FDSI_PLL_REFCLK / F\ INt) - 1$$

$$FDSI_PLL_REFCLK = 26\ MHz\ (system\ clock)$$

$$Fint = 2\ MHz\ (reduce\ PLL\ lock\ time)$$

$$RegN = 12$$

dss-E128

4. Calculate M divider for PLL:

$$RegM = ((RegN + 1) \times (FCLKIN4DDR / (2 \times FDSI_PLL_REFCLK)))$$

$$FCLKIN4DDR = 4 \times 150\ MHz$$

$$RegM = 150$$

dss-E129

15.6.8.1.2.3 Switch to DSI PLL Clock Source

Table 15-115 lists the sequence to switch the DSI and DISPC module clocks to DSI PLL clock source.

Table 15-115. Switch to DSI PLL Clock Source

Steps	Register/Bit Field/Programming	Value
Switch DISPC clock to DSI1_PLL_FCLK.	DSS_CONTROL[0] DISPC_CLK_SWITCH	0x1
Switch DSI clock to DSI2_PLL_FCLK.	DSS_CONTROL[1] DSI_CLK_SWITCH	0x1

15.6.8.1.2.4 Configure DSI Protocol Engine

15.6.8.1.2.4.1 Set Up DSI Control Registers

Table 15-116 lists the steps required to set up the DSI control registers. Table 15-117 lists the steps to set up the DSI complex I/O registers.

Table 15-116. DSI Control Registers

Steps	Register/Bit Field/Programming	Value
Enable SYNCLOST event.	DSI_IRQENABLE[18] SYNC_LOST_IRQ_EN	0x1
Enable IRQ to indicate that packet has been sent on VC1.	DSI_VC1_IRQENABLE[2] PACKET_SENT_IRQ	0x1
Enable IRQ to indicate that packet has been sent on VC0.	DSI_VC0_IRQENABLE[2] PACKET_SENT_IRQ	0x1
Set the trigger reset mode to *immediate.*	DSI_CTRL[14] TRIGGER_RESET_MODE	0x1
Activate the two line buffers.	DSI_CTRL[13:12] LINE_BUFFER	0x2
Set the size of the video port data bus to 24 bits (RGB 888).	DSI_CTRL[7:6] VP_DATA_BUS_WIDTH	0x2
Define the ratio between VP_CLK and VP_PCLK.	DSI_CTRL[4] VP_CLK_RATIO	0x1
Set the arbitration scheme for granting the VC pending ready requests in the TX FIFO as sequential scheme.	DSI_CTRL[3] TX_FIFO_ARBITRATION	0x1
Enable the ECC check for the received header.	DSI_CTRL[2] ECC_RX_EN	0x1

Table 15-117. DSI Complex I/O Registers

Steps	Register/Bit Field/Programming	Value
Determine the position of the clock lane.	DSI_COMPLEXIO_CFG1[2:0] CLOCK_POSITION	0x2
Determine the position of data 1 lane.	DSI_COMPLEXIO_CFG1[6:4] DATA1_POSITION	0x3
Turn on COMPLEXIO.	DSI_COMPLEXIO_CFG[28:27] PWR_CMD	0x1
Enable the synchronization of the shadow registers with DISPC_UPDATE_SYNC.	DSI_COMPLEXIO_CFG1[30] GOBIT	0x1
Clear all COMPLEXIO IRQ status.	DSI_COMPLEXIO_IRQSTATUS	0xC3F39CE7
Disable all COMPLEXIO IRQs.	DSI_COMPLEXIO_IRQENABLE	0x0
Enable interface.	DSI_CTRL[0] IF_EN	0x1
Disable interface.	DSI_CTRL[0] IF_EN	0x0
Wait until IF_EN = 0.	DSI_CTRL[0] IF_EN	
Enable the LP clock.	DSI_CLK_CTRL[20] LP_CLK_ENABLE	0x1

Table 15-117. DSI Complex I/O Registers (continued)

Steps	Register/Bit Field/Programming	Value
Check whether reset is complete.	DSI_COMPLEXIO_CFG1[29] RESET_DONE	0x1
Check whether power control is on.	DSI_COMPLEXIO_CFG1[26:25] PWR_STATUS	0x1
Check whether reset is complete.	DSI_SYSSTATUS[0] RESETDONE	0x1

15.6.8.1.2.4.2 Configure DSI Timing and Virtual Channels

Table 15-118 lists the steps to configure DSI timing and the virtual channels.

Table 15-118. DSI Timing Registers

Steps	Register/Bit Field/Programming	Value
Determine the number of DSI_FCLK clock cycles for the STOP-STATE counter.	DSI_TIMING1[12:0] STOP_STATE_COUNTER_IO	0x999
Disable the multiplication factor of 4 for the number of DSI_FCLK clock cycles for the STOP-STATE counter.	DSI_TIMING1[13] STOP_STATE_X4_IO	0x0
Disable the multiplication factor of 16 for the number of DSI_FCLK clock cycles for the STOP-STATE counter.	DSI_TIMING1[14] STOP_STATE_X16_IO	0x0
Clear turn-around timer settings.	DSI_TIMING1[30:16]	0x0000
Determine the number of DSI_FCLK clock cycles for the LP RX timer.	DSI_TIMING2[12:0] LP_RX_TO_COUNTER	0x0CD
Disable the multiplication factor of 4 for the number of DSI_FCLK clock cycles for the LP RX timer.	DSI_TIMING2[13] LP_RX_TO_X4	0x0
Enable the multiplication factor of 16 for the number of DSI_FCLK clock cycles for the LP RX timer.	DSI_TIMING2[14] LP_RX_TO_X16	0x1
Determine the number of TxByteClkHS clock cycles for the HS RX timer.	DSI_TIMING2[12:0] HS_TX_TO_COUNTER	0xFD2
Disable the multiplication factor of 8 for the number of TxByteClkHS clock cycles for the HS TX timer.	DSI_TIMING2[13] HS_TX_TO_X8	0x0
Enable the multiplication factor of 16 for the number of TxByteClkHS clock cycles for the HS TX timer.	DSI_TIMING2[14] HS_TX_TO_X16	0x1
DDR_CLK_PRE<<8|DDR_CLK_POST	DSI_CLK_TIMING[31:0]	0x0000 0F0B
Configuration of VC1		
Enable the checksum generation for the transmit payload.	DSI_VC1_CTRL[7] CS_TX_EN	0x1
Enable the ECC generation for the transmit header.	DSI_VC1_CTRL[8] ECC_TX_EN	0x1
Disable DMA request for TX FIFO.	DSI_VC1_CTRL[23:21] DMA_TX_REQ_NB	0x4
Disable DMA request for RX FIFO.	DSI_VC1_CTRL[29:27] DMA_RX_REQ_NB	0x4
Configuration of VC0		
Selects source of data and enable VP_STALL.	DSI_VC0_CTRL[1] SOURCE	0x1
Enable the checksum generation for the transmit payload.	DSI_VC0_CTRL[7] CS_TX_EN	0x1
Enable the ECC generation for the transmit header.	DSI_VC0_CTRL[8] ECC_TX_EN	0x1
Enable high-speed mode to send short and long packets to the peripheral.	DSI_VC0_CTRL[9] MODE_SPEED	0x1
Disable DMA request for TX FIFO.	DSI_VC0_CTRL[23:21] DMA_TX_REQ_NB	0x4
Disable DMA request for RX FIFO.	DSI_VC0_CTRL[29:27] DMA_RX_REQ_NB	0x4

Table 15-118. DSI Timing Registers (continued)

Steps	Register/Bit Field/Programming	Value
Configuration TX and RX FIFO		
Set size of the RX FIFO allocated for VC1 to 32 x 33 bits.	DSI_RX_FIFO_VC_SIZE[15:12] VC1_FIFO_SIZE	0x1
Set size of the TX and TX FIFO allocated for VC1 to 96 x 33 bits.	DSI_TX_FIFO_VC_SIZE[15:12] VC1_FIFO_SIZE	0x3

- Freq HSBYTE_CLK:

$$FHSB = FCLKIN4DDR / 16$$

$$FVPP = FCLKIN4DDR / ((RegM3 + 1) \times DISPC_LCD * DISPC_PCD)$$

$$FVP = FCLKIN4DDR / (RegM3 + 1)$$

dss-E130

- Length of the line in video mode in number of byte clock cycles (PPI clk):

$$TL = FHSB / FVPP \times (DISPC_HSA + DISPC_HFP + PPL + DISPC_HBP)$$

$$TL1f = (BPP / (8 \times NDL)) \times (DISPC_HSA + DISPC_HFP + PPL + DISPC_HBP)$$

dss-E131

- Blanking periods (HBP + HFP) in DSI are calculated based on the following formula:

$$(DISPC_HSA + DISPC_HBP + PPL + DISPC_HFP) \times Fppi = (HS + HBP + ((WC + 6) / NDL) + HFP) \times Fvp$$

$$HBP + HFP = (TVPP / THSB) \times (DISPC_HSA + DISPC_HFP + PPL + DISPC_HBP) - (HS + (WC + 6) / NDL))$$

$$HBPplusHFP = (FHSB / FVPP) \times (DISPC_HSA + DISPC_HFP + PPL + DISPC_HBP) - (HS + WC + 6) / NDL$$

$$HBPplusHFPf = ((FHSB / FVPP) \times (DISPC_HSA + DISPC_HFP + PPL + DISPC_HBP)) - ((HS + WC + 6) / NDL)$$

$$HFP = (DISPC_HFP \times BPP) / (NDL \times 8) - (2 / NDL)$$

$$HBP = HBPplusHFP - HFP$$

dss-E132

15.6.8.1.2.5 Configure DSI_PHY Timing

Table 15-119 summarizes DSI_PHY timing. For more details on timing calculation, see Section 15.4.3.2, *Clock Requirements*.

Table 15-119. Configure DSI_PHY Timing

Steps	Register/Bit Field/Programming	Value
Settings of the DSI protocol timing. For a complete description of timing specifications, see Section 15.4.3.2, *Clock Requirements*.	DSI_PHY_CFG0[31:24] THS_PREPARE	ceil(70 ns/DDR clock period) + 2
	DSI_PHY_CFG0[23:16] THS_PREPARE_THS_ZERO	ceil(175 ns/DDR clock period) + 2
	DSI_PHY_CFG0[7:0] THS_EXIT	ceil(145 ns/DDR clock period)
	DSI_PHY_CFG0[15:8] THS_TRAIL	ceil(60 ns/DDR clock period) + 5
	DSI_PHY_CFG2[7:0] TCLK_PREPARE	ceil(65 ns/DDR clock period)
	DSI_PHY_CFG1[7:0] TCLK_ZERO	ceil(260 ns/DDR clock period)
	DSI_PHY_CFG1[15:0] TCLK_TRAIL	ceil(60 ns/DDR clock period) + 2
	DSI_PHY_CFG1[22:16] TLPX_HALF	ceil(26.5 ns/DDR clock period)

Note: Keep Reserved bits at reset value in the DSI_PHY_CFG1 and DSI_PHY_CFG2 registers.

15.6.8.1.2.6 Drive Stop State

Table 15-120 lists the steps to drive the stop state.

Table 15-120. Drive Stop State

Steps	Register/Bit Field/Programming	Value
Force TX stop mode.	DSI_TIMING[15] FORCE_TX_STOP_MODE_IO	0x1
Wait until FORCE_TX_STOP_MODE_IO = 0.	DSI_TIMING[15] FORCE_TX_STOP_MODE_IO	

15.6.8.1.3 Initialization of the External MIPI LCD Controller

Table 15-121 lists the steps to initialize the external MIPI LCD controller.

Table 15-121. Initialization of the External MIPI LCD Controller

Steps	Register/Bit Field/Programming	Value
Reset the MIPI LCD controller using GPIO87.	–	0x1
Wait until initialization of the external MIPI LCD controller is finished after power up.	–	–
Configure the external MIPI LCD controller.	–	–

15.6.8.1.4 Configure the DISPC

15.6.8.1.4.1 Reset DISPC

Table 15-122 lists the steps to reset the DISPC.

Table 15-122. Reset DISPC

Steps	Register/Bit Field/Programming	Value
Reset the DISPC.	DISPC_SYSCONFIG[1] SOFTRESET	0x1
Wait until RESETDONE = 1.	DISPC_SYSCONFIG[0] RESETDONE	
Disable master interface power management.	DISPC_SYSCONFIG [13:12] MIDLEMODE	0x1
Disable slave interface power management.	DISPC_SYSCONFIG [4:3] SIDLEMODE	0x1
Disable all DISPC interrupts.	DISPC_IRQENABLE[31:0]	0x0

15.6.8.1.4.2 Configure DISPC Timing, Window, and Color

Table 15-123 lists the steps to configure the DISPC registers. Table 15-106 lists the steps to configure the color space coefficient registers.

Table 15-124 lists the steps to configure DISPC_CONTROL.

Table 15-123. Configure DISPC Registers

Steps	Register/Bit Field/Programming	Value
Configure LCD output		
Set the logic clock divisor (LCD).	DISPC_DIVISOR[23:16] LCD	0x1
Set the pixel clock divisor (PCD).	DISPC_DIVISOR[7:0] PCD	0x4
Set the number of lines on the LCD panel.	DISPC_SIZE_LCD[26:16] LPP	0x2A7
Set the number of PPL on the LCD panel.	DISPC_SIZE_LCD[10:0] PPL	0x1DF

Table 15-123. Configure DISPC Registers (continued)

Steps	Register/Bit Field/Programming	Value
Set solid background color.	DISPC_DEFAULT_COLOR0	0XFF
Configure VIDEO pipeline VID1.		
Set VID1 FIFO low threshold.	DISPC_VID1_FIFO_THRESHOLD[11:0] VIDFIFOLOWTHRESHOLD	0x0C0
Set VID1 FIFO high threshold.	DISPC_VID1_FIFO_THRESHOLD[27:16] VIDFIFOHIGHTHRESHOLD	0xFC0
Set the X position of the VID1 window.	DISPC_VID1_POSITION[10:0] VIDPOSX	0x0
Set the Y position of the VID1 window.	DISPC_VID1_POSITION[26:16] VIDPOSY	0x0
Set the number of lines of the VID1 window.	DISPC_VID1_SIZE[26:16] VIDSIZEY	0x2A7
Set the number of pixels of the VID1 window.	DISPC_VID1_SIZE[10:0] VIDSIZEX	0x1DF
Define the base address of the VID1 frame buffer.	DISPC_VID1_BA0	0x–

Table 15-124. Configure DISPC_CONTROL

Comments	Register/Bit Field/Programming	Value
Enable pixel clock free-running.	DISPC_CONTROL[27] PCLKFREEENABLE	0x1
Disable RFBI.	DISPC_CONTROL[16] GPOUT1	0x1
	DISPC_CONTROL[15] GPOUT0	0x1
Enable the stall mode.	DISPC_CONTROL[11] STALLMODE	0x1
Select size of DATALINES.	DISPC_CONTROL[9:8] TFTDATALINES	0x3
Select active matrix display operation mode.	DISPC_CONTROL[3] STNTFT	0x1
Disable LCD output interface.	DISPC_CONTROL[0] LCDENABLE	0x0
Update the internal DISPC registers.	DISPC_CONTROL[5] GOLCD	0x1

15.6.8.1.5 Enable Command Mode Using DISPC Video Port

Table 15-125 lists the steps to enable DISPC to send frames continuously. Two bus turn-arounds (BTA) must be generated (see the MIPI DSI specification for details):

- The first BTA gives bus possession to the display module.
- The second BTA obtains the TE trigger.

Table 15-125. Enable Command Mode and Automatic TE

Steps	Register/Bit Field/Programming	Value
Insert DCS write memory continue code.	DSI_CTRL[25] DCS_CMD_CODE	0x0
Enable automatic insertion of DCS command codes when data is sourced by the video port.	DSI_CTRL[24] DCS_CMD_ENABLE	0x1
Enable VC1.	DSI_VC1_CTRL[0] VC_EN	0x1
Enable VC0.	DSI_VC0_CTRL[0] VC_EN	0x1
Enable the interface.	DSI_CTRL[0] IF_EN	0x1
Wait until IF_EN = 1.	DSI_CTRL[0] IF_EN	
Send the sequence to receive theTE trigger from the peripheral. In this use case, code 0x35 + 1 parameter VC = 0, data type = 0x15, DCS write + 1 parameter.	DSI_VC1_SHORT_PACKET_HEADER[31:0] HEADER	0x0000 3515
Wait until PACKET_SENT_IRQ = 1.	DSI_VC1_IRQSTATUS[2] PACKET_SENT_IRQ	
Write 1 to clear PACKET_SENT_IRQ.	DSI_VC1_IRQSTATUS[2] PACKET_SENT_IRQ	0x1

15.6.8.1.6 Send Frame Data to LCD Panel Using Automatic TE

Table 15-126 summarizes the steps to send a frame data to the LCD panel using automatic TE.

Table 15-126. Send Frame Data to LCD Panel Using Automatic TE

Steps	Register/Bit Field/Programming	Value
Enable the transfer between DISPC and DSI. Reset after the transfer is done.	DISPC_CONTROL[0] LCDENABLE	0x1
Specify the number of bytes to send. When DCS insertions is used, word count (WC) must include this one DCS byte.	DSI_VC0_TE[23:0] TE_SIZE	(WC+1)*LPP
Set up long packet header. Send 0x39 DCS long write/write_LUT command packet used to send larger blocks of data to a display module that implements a DCS.	DSI_VC0_LONG_PACKET_HEADER[31:0] HEADER	(WC+1) << 8 + 0x39
Enable TE control.	DSI_VC0_TE[30] TE_EN	0x1
Wait until RX FIFO is empty, RX_FIFO_NOT_EMPTY = 0.	DSI_VC1_CTRL[20] RX_FIFO_NOT_EMPTY	0x0
Wait until TX FIFO is not full. TX_FIFO_FULL = 0.	DSI_VC1_CTRL[16] TX_FIFO_FULL	0x0
Enable first BTA to give bus possession to the display module.	DSI_VC1_CTRL[6] BTA_EN	0x1
Wait until BTA IRQ.	DSI_VC1_IRQSTATUS[5] BTA_IRQ	0x1
Write 1 to clear BTA IRQ.	DSI_VC1_IRQSTATUS[5] BTA_IRQ	0x1
Enable second BTA to get the TE trigger.	DSI_VC1_CTRL[6] BTA_EN	0x1
Wait until BTA IRQ.	DSI_VC1_IRQSTATUS[5] BTA_IRQ	0x1
Write 1 to clear BTA IRQ.	DSI_VC1_IRQSTATUS[5] BTA_IRQ	0x1
Wait until transfer is complete.	DSI_VC0_TE[30] TE_EN	Read 0x0

15.7 Display Subsystem Register Manual

CAUTION

- The DISS, DISPC, RFBI, and VENC registers have no register data width access restriction and can be accessed in 8-bit, 16-bit and 32-bit access..
- The DSI complex I/O and DSI PLL control module registers are limited to 32-bit data access; 16-bit and 8-bit data accesses are not allowed and can corrupt register content.
- The DSI protocol engine DSS.DSI_VCn_LONG_PACKET_HEADER and DSS.DSI_VCn_SHORT_PACKET_HEADER registers are limited to 32-bit data access; 16-bit and 8-bit data accesses are not allowed and can corrupt register content.
- The DSI protocol engine DSS.DSI_VCn_LONG_PACKET_PAYLOAD register is limited to 32-bit and 16-bit data access; 8-bit data accesses are not allowed and can corrupt register content.
- All other DSI protocol engine registers have no register data width access restriction and can be accessed in 8-bit, 16-bit and 32-bit access.

Table 15-127 summarizes the display subsystem instance.

Table 15-127. Instance Summary

Module Name	Base Address	Size
DSI Protocol Engine	0x4804 FC00	512 bytes
DSI_PHY	0x4804 FE00	64 bytes
DSI PLL Controller	0x4804 FF00	32 bytes
Display Subsystem	0x4805 0000	512 bytes
Display Controller	0x4805 0400	1K byte
Display Controller VID1	0x4805 0400	1K byte
Display Controller VID2	0x4805 0400	1K byte
RFBI	0x4805 0800	256 bytes
Video Encoder	0x4805 0C00	256 bytes

15.7.1 Display Subsystem Register Mapping Summary

Table 15-128 through Table 15-133 summarize the display subsystem register mapping.

Table 15-128. Display Subsystem Register Mapping Summary

Register Name	Type	Register Width (Bits)	Address Offset	Physical Address
DSS_REVISIONNUMBER	R	32	0x000	0x4805 0000
DSS_SYSCONFIG	RW	32	0x010	0x4805 0010
DSS_SYSSTATUS	R	32	0x014	0x4805 0014
DSS_IRQSTATUS	R	32	0x018	0x4805 0018
DSS_CONTROL	RW	32	0x040	0x4805 0040
DSS_SDI_CONTROL	RW	32	0x044	0x4805 0044
DSS_PLL_CONTROL	RW	32	0x048	0x4805 0048
DSS_SDI_STATUS	R	32	0x05C	0x4805 005C

Table 15-129. Display Controller Register Mapping Summary

Register Name	Type	Register Width (Bits)	Address Offset	Physical Address
DISPC_REVISION	R	32	0x000	0x4805 0400
DISPC_SYSCONFIG	RW	32	0x010	0x4805 0410
DISPC_SYSSTATUS	R	32	0x014	0x4805 0414
DISPC_IRQSTATUS	RW	32	0x018	0x4805 0418
DISPC_IRQENABLE	RW	32	0x01C	0x4805 041C
DISPC_CONTROL	RW	32	0x040	0x4805 0440
DISPC_CONFIG	RW	32	0x044	0x4805 0444
DISPC_DEFAULT_COLOR_m	RW	32	0x04C+(m * 0x04) (1)	0x4805 044C+(m * 0x04) (1)
DISPC_TRANS_COLOR_m	RW	32	0x054+(m * 0x04) (1)	0x4805 0454+(m * 0x04) (1)
DISPC_LINE_STATUS	R	32	0x05C	0x4805 045C
DISPC_LINE_NUMBER	RW	32	0x060	0x4805 0460
DISPC_TIMING_H	RW	32	0x064	0x4805 0464
DISPC_TIMING_V	RW	32	0x068	0x4805 0468
DISPC_POL_FREQ	RW	32	0x06C	0x4805 046C
DISPC_DIVISOR	RW	32	0x070	0x4805 0470
DISPC_GLOBAL_ALPHA	RW	32	0x074	0x4805 0474
DISPC_SIZE_DIG	RW	32	0x078	0x4805 0478
DISPC_SIZE_LCD	RW	32	0x07C	0x4805 047C
DISPC_GFX_BAj	RW	32	0x080+(j * 0x04) (2)	0x4805 0480+(j * 0x04) (2)
DISPC_GFX_POSITION	RW	32	0x088	0x4805 0488
DISPC_GFX_SIZE	RW	32	0x08C	0x4805 048C
DISPC_GFX_ATTRIBUTES	RW	32	0x0A0	0x4805 04A0
DISPC_GFX_FIFO_THRESHOLD	RW	32	0x0A4	0x4805 04A4
DISPC_GFX_FIFO_SIZE_STATUS	R	32	0x0A8	0x4805 04A8
DISPC_GFX_ROW_INC	RW	32	0x0AC	0x4805 04AC
DISPC_GFX_PIXEL_INC	RW	32	0x0B0	0x4805 04B0
DISPC_GFX_WINDOW_SKIP	RW	32	0x0B4	0x4805 04B4
DISPC_GFX_TABLE_BA	RW	32	0x0B8	0x4805 04B8
DISPC_DATA_CYCLEk	RW	32	0x1D4+(k * 0x04) (3)	0x4805 05D4+(k * 0x04) (3)
DISPC_CPR_COEF_R	RW	32	0x220	0x4805 0620
DISPC_CPR_COEF_G	RW	32	0x224	0x4805 0624
DISPC_CPR_COEF_B	RW	32	0x228	0x4805 0628
DISPC_GFX_PRELOAD	RW	32	0x22C	0x4805 062C

(1) m = 0 to 1
(2) j = 0 to 1
(3) k = 0 to 2

Table 15-130. Display Controller VID1 Register Summary

Register Name (n = 1 for VID1)	Type	Register Width (Bits)	Address Offset	Display Controller VID1 Physical Address
DISPC_VIDn_BAj	RW	32	0x0BC+((n–1)* 0x90) + (j * 0x04) (1)	0x4805 04BC + (j *0x04) (1)
DISPC_VIDn_POSITION	RW	32	0x0C4+((n–1)* 0x90)	0x4805 04C4
DISPC_VIDn_SIZE	RW	32	0x0C8+((n–1)* 0x90)	0x4805 04C8

(1) j = 0 to 1

Table 15-130. Display Controller VID1 Register Summary (continued)

Register Name (n = 1 for VID1)	Type	Register Width (Bits)	Address Offset	Display Controller VID1 Physical Address
DISPC_VIDn_ATTRIBUTES	RW	32	0x0CC+((n–1)* 0x90)	0x4805 04CC
DISPC_VIDn_FIFO_THRESHOLD	RW	32	0x0D0+((n–1)* 0x90)	0x4805 04D0
DISPC_VIDn_FIFO_SIZE_STATUS	R	32	0x0D4+((n–1)* 0x90)	0x4805 04D4
DISPC_VIDn_ROW_INC	RW	32	0x0D8+((n–1)* 0x90)	0x4805 04D8
DISPC_VIDn_PIXEL_INC	RW	32	0x0DC+((n–1)* 0x90)	0x4805 04DC
DISPC_VIDn_FIR	RW	32	0x0E0+((n–1)* 0x90)	0x4805 04E0
DISPC_VIDn_PICTURE_SIZE	RW	32	0x0E4+((n–1)* 0x90)	0x4805 04E4
DISPC_VIDn_ACCUl	RW	32	0x0E8 + ((n–1)* 0x90) + (l* 0x04) (2)	0x4805 04E8 + (l* 0x04) (2)
DISPC_VIDn_FIR_COEF_Hi	RW	32	0x0F0+ ((n–1)* 0x90) + (i* 0x08) (3)	0x4805 04F0 + (i* 0x08) (3)
DISPC_VIDn_FIR_COEF_HVi	RW	32	0x0F4+ ((n–1)* 0x90) + (i* 0x08) (3)	0x4805 04F4 + (i*0x08) (3)
DISPC_VIDn_CONV_COEF0	RW	32	0x130+((n–1)* 0x90)	0x4805 0530
DISPC_VIDn_CONV_COEF1	RW	32	0x134+((n–1)* 0x90)	0x4805 0534
DISPC_VIDn_CONV_COEF2	RW	32	0x138+((n–1)* 0x90)	0x4805 0538
DISPC_VIDn_CONV_COEF3	RW	32	0x13C+((n–1)* 0x90)	0x4805 053C
DISPC_VIDn_CONV_COEF4	RW	32	0x140+((n–1)* 0x90)	0x4805 0540
DISPC_VIDn_FIR_COEF_Vi	RW	32	0x1E0+ ((n–1)*0x20) + (i* 0x04) (3)	0x4805 05E0 + (i* 0x04) (3)
DISPC_VIDn_PRELOAD	RW	32	0x230+((n–1)* 0x04)	0x4805 0630

(2) l = 0 to 1
(3) i = 0 to 7

Table 15-131. Display Controller VID2 Register Summary

Register Name (n = 2 for VID2)	Type	Register Width (Bits)	Address Offset	Display Controller VID2 Physical Address
DISPC_VIDn_BAj	RW	32	0x0BC+((n–1)* 0x90) + (j * 0x04) (1)	0x4805 054C + (j *0x04) (1)
DISPC_VIDn_POSITION	RW	32	0x0C4+((n–1)* 0x90)	0x4805 0554
DISPC_VIDn_SIZE	RW	32	0x0C8+((n–1)* 0x90)	0x4805 0558
DISPC_VIDn_ATTRIBUTES	RW	32	0x0CC+((n–1)* 0x90)	0x4805 055C
DISPC_VIDn_FIFO_THRESHOLD	RW	32	0x0D0+((n–1)* 0x90)	0x4805 0560
DISPC_VIDn_FIFO_SIZE_STATUS	R	32	0x0D4+((n–1)* 0x90)	0x4805 0564
DISPC_VIDn_ROW_INC	RW	32	0x0D8+((n–1)* 0x90)	0x4805 0568
DISPC_VIDn_PIXEL_INC	RW	32	0x0DC+((n–1)* 0x90)	0x4805 056C
DISPC_VIDn_FIR	RW	32	0x0E0+((n–1)* 0x90)	0x4805 0570
DISPC_VIDn_PICTURE_SIZE	RW	32	0x0E4+((n–1)* 0x90)	0x4805 0574
DISPC_VIDn_ACCUl	RW	32	0x0E8 + ((n–1)* 0x90) + (l* 0x04) (2)	0x4805 0578 + (l* 0x04) (2)
DISPC_VIDn_FIR_COEF_Hi	RW	32	0x0F0+ ((n–1)* 0x90) + (i* 0x08) (3)	0x4805 0580 + (i* 0x08) (3)
DISPC_VIDn_FIR_COEF_HVi	RW	32	0x0F4+ ((n–1)* 0x90) + (i* 0x08) (3)	0x4805 0584 + (i*0x08) (3)
DISPC_VIDn_CONV_COEF0	RW	32	0x130+((n–1)* 0x90)	0x4805 05C0

(1) j = 0 to 1
(2) l = 0 to 1
(3) i = 0 to 7

Table 15-131. Display Controller VID2 Register Summary (continued)

Register Name (n = 2 for VID2)	Type	Register Width (Bits)	Address Offset	Display Controller VID2 Physical Address
DISPC_VIDn_CONV_COEF1	RW	32	0x134+((n–1)* 0x90)	0x4805 05C4
DISPC_VIDn_CONV_COEF2	RW	32	0x138+((n–1)* 0x90)	0x4805 05C8
DISPC_VIDn_CONV_COEF3	RW	32	0x13C+((n–1)* 0x90)	0x4805 05CC
DISPC_VIDn_CONV_COEF4	RW	32	0x140+((n–1)* 0x90)	0x4805 05D0
DISPC_VIDn_FIR_COEF_Vi	RW	32	0x1E0+ ((n–1)*0x20) + (i* 0x04) (3)	0x4805 0670 + (i* 0x04) (3)
DISPC_VIDn_PRELOAD	RW	32	0x230+((n–1)* 0x04)	0x4805 0634

Table 15-132. RFBI Register Mapping Summary

Register Name	Type	Register Width (Bits)	Address Offset	Physical Address
RFBI_REVISION	R	32	0x00	0x4805 0800
RFBI_SYSCONFIG	RW	32	0x10	0x4805 0810
RFBI_SYSSTATUS	R	32	0x14	0x4805 0814
RFBI_CONTROL	RW	32	0x40	0x4805 0840
RFBI_PIXEL_CNT	RW	32	0x44	0x4805 0844
RFBI_LINE_NUMBER	RW	32	0x48	0x4805 0848
RFBI_CMD	W	32	0x4C	0x4805 084C
RFBI_PARAM	W	32	0x50	0x4805 0850
RFBI_DATA	W	32	0x54	0x4805 0854
RFBI_READ	RW	32	0x58	0x4805 0858
RFBI_STATUS	RW	32	0x5C	0x4805 085C
RFBI_CONFIGi	RW	32	0x60+ (i* 0x18) (1)	0x4805 0860+ (i* 0x18) (1)
RFBI_ONOFF_TIMEi	RW	32	0x64+ (i* 0x18) (1)	0x4805 0864+ (i* 0x18) (1)
RFBI_CYCLE_TIMEi	RW	32	0x68+ (i* 0x18) (1)	0x4805 0868+ (i* 0x18) (1)
RFBI_DATA_CYCLE1_i	RW	32	0x6C+ (i* 0x18) (1)	0x4805 086C+ (i* 0x18) (1)
RFBI_DATA_CYCLE2_i	RW	32	0x70+ (i* 0x18) (1)	0x4805 0870+ (i* 0x18) (1)
RFBI_DATA_CYCLE3_i	RW	32	0x74+ (i* 0x18) (1)	0x4805 0874+ (i* 0x18) (1)
RFBI_VSYNC_WIDTH	RW	32	0x90	0x4805 0890
RFBI_HSYNC_WIDTH	RW	32	0x94	0x4805 0894

(1) i = 0 to 1

Table 15-133. Video Encoder Register Mapping Summary

Register Name	Type	Register Width (Bits)	Address Offset	Physical Address
VENC_REV_ID	R	32	0x00	0x4805 0C00
VENC_STATUS	R	32	0x04	0x4805 0C04
VENC_F_CONTROL	RW	32	0x08	0x4805 0C08
VENC_VIDOUT_CTRL	RW	32	0x10	0x4805 0C10
VENC_SYNC_CTRL	RW	32	0x14	0x4805 0C14
VENC_LLEN	RW	32	0x1C	0x4805 0C1C
VENC_FLENS	RW	32	0x20	0x4805 0C20
VENC_HFLTR_CTRL	RW	32	0x24	0x4805 0C24
VENC_CC_CARR_WSS_CARR	RW	32	0x28	0x4805 0C28
VENC_C_PHASE	RW	32	0x2C	0x4805 0C2C
VENC_GAIN_U	RW	32	0x30	0x4805 0C30

Table 15-133. Video Encoder Register Mapping Summary (continued)

Register Name	Type	Register Width (Bits)	Address Offset	Physical Address
VENC_GAIN_V	RW	32	0x34	0x4805 0C34
VENC_GAIN_Y	RW	32	0x38	0x4805 0C38
VENC_BLACK_LEVEL	RW	32	0x3C	0x4805 0C3C
VENC_BLANK_LEVEL	RW	32	0x40	0x4805 0C40
VENC_X_COLOR	RW	32	0x44	0x4805 0C44
VENC_M_CONTROL	RW	32	0x48	0x4805 0C48
VENC_BSTAMP_WSS_DATA	RW	32	0x4C	0x4805 0C4C
VENC_S_CARR	RW	32	0x50	0x4805 0C50
VENC_LINE21	RW	32	0x54	0x4805 0C54
VENC_LN_SEL	RW	32	0x58	0x4805 0C58
VENC_L21_WC_CTL	RW	32	0x5C	0x4805 0C5C
VENC_HTRIGGER_VTRIGGER	RW	32	0x60	0x4805 0C60
VENC_SAVID_EAVID	RW	32	0x64	0x4805 0C64
VENC_FLEN_FAL	RW	32	0x68	0x4805 0C68
VENC_LAL_PHASE_RESET	RW	32	0x6C	0x4805 0C6C
VENC_HS_INT_START_STOP_X	RW	32	0x70	0x4805 0C70
VENC_HS_EXT_START_STOP_X	RW	32	0x74	0x4805 0C74
VENC_VS_INT_START_X	RW	32	0x78	0x4805 0C78
VENC_VS_INT_STOP_X_VS_INT_START_Y	RW	32	0x7C	0x4805 0C7C
VENC_VS_INT_STOP_Y_VS_EXT_START_X	RW	32	0x80	0x4805 0C80
VENC_VS_EXT_STOP_X_VS_EXT_START_Y	RW	32	0x84	0x4805 0C84
VENC_VS_EXT_STOP_Y	RW	32	0x88	0x4805 0C88
VENC_AVID_START_STOP_X	RW	32	0x90	0x4805 0C90
VENC_AVID_START_STOP_Y	RW	32	0x94	0x4805 0C94
VENC_FID_INT_START_X_FID_INT_START_Y	RW	32	0xA0	0x4805 0CA0
VENC_FID_INT_OFFSET_Y_FID_EXT_START_X	RW	32	0xA4	0x4805 0CA4
VENC_FID_EXT_START_Y_FID_EXT_OFFSET_Y	RW	32	0xA8	0x4805 0CA8
VENC_TVDETGP_INT_START_STOP_X	RW	32	0xB0	0x4805 0CB0
VENC_TVDETGP_INT_START_STOP_Y	RW	32	0xB4	0x4805 0CB4
VENC_GEN_CTRL	RW	32	0xB8	0x4805 0CB8
VENC_OUTPUT_CONTROL	RW	32	0xC4	0x4805 0CC4
VENC_OUTPUT_TEST	RW	32	0xC8	0x4805 0CC8

Table 15-134. DSI Protocol Engine Register Mapping Summary

Register Name	Type	Register Width (Bits)	Address Offset	Physical Address
DSI_REVISION	R	32	0x000	0x4804 FC00
DSI_SYSCONFIG	RW	32	0x010	0x4804 FC10
DSI_SYSSTATUS	R	32	0x014	0x4804 FC14
DSI_IRQSTATUS	RW	32	0x018	0x4804 FC18
DSI_IRQENABLE	RW	32	0x01C	0x4804 FC1C
DSI_CTRL	RW	32	0x040	0x4804 FC40
DSI_COMPLEXIO_CFG1	RW	32	0x048	0x4804 FC48

Table 15-134. DSI Protocol Engine Register Mapping Summary (continued)

Register Name	Type	Register Width (Bits)	Address Offset	Physical Address
DSI_COMPLEXIO_IRQ STATUS	RW	32	0x04C	0x4804 FC4C
DSI_COMPLEXIO_IRQ ENABLE	RW	32	0x050	0x4804 FC50
DSI_CLK_CTRL	RW	32	0x054	0x4804 FC54
DSI_TIMING1	RW	32	0x058	0x4804 FC58
DSI_TIMING2	RW	32	0x05C	0x4804 FC5C
DSI_VM_TIMING1	RW	32	0x060	0x4804 FC60
DSI_VM_TIMING2	RW	32	0x064	0x4804 FC64
DSI_VM_TIMING3	RW	32	0x068	0x4804 FC68
DSI_CLK_TIMING	RW	32	0x06C	0x4804 FC6C
DSI_TX_FIFO_VC_SIZE	RW	32	0x070	0x4804 FC70
DSI_RX_FIFO_VC_SIZ E	RW	32	0x074	0x4804 FC74
DSI_COMPLEXIO_CFG 2	RW	32	0x078	0x4804 FC78
DSI_RX_FIFO_VC_FUL LNESS	R	32	0x07C	0x4804 FC7C
DSI_VM_TIMING4	RW	32	0x080	0x4804 FC80
DSI_TX_FIFO_VC_EMP TINESS	R	32	0x084	0x4804 FC84
DSI_VM_TIMING5	RW	32	0x088	0x4804 FC88
DSI_VM_TIMING6	RW	32	0x08C	0x4804 FC8C
DSI_VM_TIMING7	RW	32	0x090	0x4804 FC90
DSI_STOPCLK_TIMING	RW	32	0x094	0x4804 FC94
DSI_VCn_CTRL	RW	32	0x100+ (n* 0x20)(1)	0x4804 FD00+ (n* 0x20)(1)
DSI_VCn_TE	RW	32	0x104+ (n* 0x20)(1)	0x4804 FD04+ (n* 0x20)(1)
DSI_VCn_LONG_PACK ET_HEADER	W	32	0x108+ (n* 0x20)(1)	0x4804 FD08+ (n* 0x20)(1)
DSI_VCn_LONG_PACK ET_PAYLOAD	W	32	0x10C+ (n* 0x20)(1)	0x4804 FD0C+ (n* 0x20)(1)
DSI_VCn_SHORT_PAC KET_HEADER	RW	32	0x110+ (n* 0x20)(1)	0x4804 FD10+ (n* 0x20)(1)
DSI_VCn_IRQSTATUS	RW	32	0x118+ (n* 0x20)(1)	0x4804 FD18+ (n* 0x20)(1)
DSI_VCn_IRQENABLE	RW	32	0x11C+ (n* 0x20)(1)	0x4804 FD1C+ (n* 0x20)(1)

(1) n = 0 to 3

Table 15-135. DSI_PHY Register Mapping Summary

Register Name	Type	Register Width (Bits)	Address Offset	Physical Address
DSI_PHY_CFG0	RW	32	0x0000 0000	0x4804 FE00
DSI_PHY_CFG1	RW	32	0x0000 0004	0x4804 FE04
DSI_PHY_CFG2	RW	32	0x0000 0008	0x4804 FE08
DSI_PHY_CFG5	R	32	0x0000 0014	0x4804 FE14

Table 15-136. DSI PLL Controller Register Mapping Summary

Register Name	Type	Register Width (Bits)	Address Offset	Physical Address
DSI_PLL_CONTROL	RW	32	0x0000 0000	0x4804 FF00
DSI_PLL_STATUS	R	32	0x0000 0004	0x4804 FF04
DSI_PLL_GO	RW	32	0x0000 0008	0x4804 FF08
DSI_PLL_CONFIGURATION1	RW	32	0x0000 000C	0x4804 FF0C
DSI_PLL_CONFIGURATION2	RW	32	0x0000 0010	0x4804 FF10

15.7.2 Register Descriptions

15.7.2.1 Display Subsystem and SDI Registers

Table 15-137. DSS_REVISIONNUMBER

Address Offset	0x000		
Physical address	0x4805 0000	**Instance**	DISS
Description	This register contains the display subsystem revision number.		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																								REV							

Bits	Field Name	Description	Type	Reset
31:8	Reserved	Read returns 0.	R	0x000000
7:0	REV	Revision number [7:4] Major revision [3:0] Minor revision	R	TI internal data

Table 15-138. Register Call Summary for Register DSS_REVISIONNUMBER

Display Subsystem Register Manual

- Display Subsystem Register Mapping Summary: [0]

Table 15-139. DSS_SYSCONFIG

Address Offset	0x010		
Physical address	0x4805 0010	**Instance**	DISS
Description	This register controls the various parameters of the interconnect interface.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																											Reserved		Reserved		SOFTRESET	AUTOIDLE

Bits	Field Name	Description	Type	Reset
31:5	Reserved	Write 0s for future compatibility. Reads return zero.	RW	0x00000000
4:3	Reserved	Reserved. Keep at reset value.	RW	0x0
2	Reserved	Write 0s for future compatibility . Reads return zero.	RW	0

Bits	Field Name	Description	Type	Reset
1	SOFTRESET	Software reset. Set this bit to 1 to trigger a module reset. The bit is automatically reset by the hardware. During reads, it always returns 0. 0x0: Normal mode 0x1: The module is reset	RW	0
0	AUTOIDLE	Enable power management capability 0x0: OCP clock is free-running 0x1: Automatic OCP clock gating strategy is applied based on the OCP interface activity	RW	1

Table 15-140. Register Call Summary for Register DSS_SYSCONFIG

Display Subsystem Integration • Resets: [0] • Power Management: [1] • Interrupt Requests: [2]
Display Subsystem Basic Programming Model • Display Subsystem Reset: [3]
Display Subsystem Use Cases and Tips • Autoidle and Smart Idle: [4] [5] • Programming Flow: [6] [7]
Display Subsystem Register Manual • Display Subsystem Register Mapping Summary: [8]

Table 15-141. DSS_SYSSTATUS

Address Offset	0x014		
Physical address	0x4805 0014	**Instance**	DISS
Description	This register provides status information about the module.		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																															RESETDONE

Bits	Field Name	Description	Type	Reset
31:1	Reserved	Read returns 0.	R	0x00000000
0	RESETDONE	Internal reset monitoring Read 0x0: Internal module reset is ongoing. Read 0x1: Reset completed	R	1

Table 15-142. Register Call Summary for Register DSS_SYSSTATUS

Display Subsystem Integration • Resets: [0]
Display Subsystem Basic Programming Model • Display Subsystem Reset: [1]
Display Subsystem Use Cases and Tips • Programming Flow: [2] [3]
Display Subsystem Register Manual • Display Subsystem Register Mapping Summary: [4]

Table 15-143. DSS_IRQSTATUS

Address Offset	0x0000 0018		
Physical Address	0x4805 0018	**Instance**	DSS
Description	The register indicates the source of the interrupt and the status of the interrupt line.		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																														DSI_IRQ	DISPC_IRQ

Bits	Field Name	Description	Type	Reset
31:2	RESERVED	Reads returns 0.	R	0x00000000
1	DSI_IRQ	DSI interrupt status (related to DSI_IRQSTATUS) 0x0: DSI interrupt inactive 0x1: DSI interrupt active	R	0x0
0	DISPC_IRQ	DISPC interrupt status (related to DISPC_IRQSTATUS) 0x0: DISPC interrupt inactive 0x1: DISPC interrupt active	R	0x0

Table 15-144. Register Call Summary for Register DSS_IRQSTATUS

Display Subsystem Register Manual

- Display Subsystem Register Mapping Summary: [0]

Table 15-145. DSS_CONTROL

Address Offset	0x040		
Physical address	0x4805 0040	**Instance**	DISS
Description	This register contains the display subsystem control bits.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																									VENC_OUT_SEL	DAC_POWERDN_BGZ	DAC_DEMEN	VENC_CLOCK_4X_ ENABLE	VENC_CLOCK_MODE	DSI_CLK_SWITCH	DISPC_CLK_SWITCH

Bits	Field Name	Description	Type	Reset
31:7	Reserved	Reserved for future DAC use	RW	0x0000000
6	VENC_OUT_SEL	Video DAC1 input selection: 0x0: CVBS VENC output selected for composite video mode 0x1: Luminance VENC output selected for s-video mode	RW	0
5	DAC_POWERDN_BGZ	DAC Power-Down Control 0x0: DAC Power-Down Band Gap powered down 0x1: DAC Power-Down Band Gap powered up	RW	0

Bits	Field Name	Description	Type	Reset
4	DAC_DEMEN	DAC dynamic element matching enable 0x0: DAC Dynamic Element Matching Disabled 0x1: DAC Dynamic Element Matching Enabled	RW	0
3	VENC_CLOCK_4X_ ENABLE	VENC clock 4x enable 0x0: Disable 0x1: Enable	RW	0
2	VENC_CLOCK_MODE	VENC clock mode (0 for normal; 1 for square pixel) 0x0: Mode 0 0x1: Mode 1	RW	0
1	DSI_CLK_SWITCH	Selects the clock source for the DSI functional clock 0x0: DSS1_ALWON_FCLK clock is selected (from PRCM) 0x1: DSI2_PLL_FCLK clock is selected (from DSI PLL)	RW	0
0	DISPC_CLK_SWITCH	Selects the clock source for the DISPC functional clock 0x0: DSS1_ALWON_FCLK clock is selected (from PRCM) 0x1: DSI1_PLL_FCLK clock is selected (from DSI PLL)	RW	0

Table 15-146. Register Call Summary for Register DSS_CONTROL

Display Subsystem Environment • TV Display Support: [0] • Digital-to-Analog Converter: [1]
Display Subsystem Integration • Clocks: [2] [3] [4] [5]
Display Subsystem Functional Description • Video Encoder Functionalities: [6]
Display Subsystem Basic Programming Model • Display Subsystem Configuration Phase: [7] [8] [9] [10] • Video DAC Settings: [11]
Display Subsystem Use Cases and Tips • Display Subsystem Clock: [12] [13] [14] • DPLL4 in Low-Power Mode: [15] [16] • Configure DSI, DSI PLL and Complex I/O: [17] • Display Subsystem Use Cases and Tips: [18] [19]
Display Subsystem Register Manual • Display Subsystem Register Mapping Summary: [20]

Table 15-147. DSS_SDI_CONTROL

Address Offset	0x044		
Physical address	0x4805 0044	**Instance**	DISS
Description	This register contains the display subsystem control bits.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								SDI_PDIV								SDI_PHYLPMODE	SDI_RBITS			SDI_AUTOSTDBY	Reserved	Reserved						SDI_PRSEL		SDI_BWSEL	

Bits	Field Name	Description	Type	Reset
31:20	Reserved	Reserved. Write 0s for future compatibility. Read returns0.	RW	0x000
19:15	SDI_PDIV	Specifies the ratio of PLL output to pixel clock frequency	RW	0x00
14	SDI_PHYLPMODE	FlatLink3G output buffer low power option Disables the internal transmitter termination to reduce power. Requires reduced data and signal integrity verification 0x0: Standard mode 0x1: Low-power mode	RW	0
13:12	SDI_RBITS	FlatLink3G reserved bits F1 and F0	RW	0x0
11	SDI_AUTOSTDBY	FlatLink3G auto-standby 0x0: High-speed serial buffers only enabled when PLL is locked and SDI enable is active 0x1: High-speed serial buffers enabled while SDI enable is active	RW	0
10	Reserved	Must be programmed to the default value for SN65LVDS302 compatibility	RW	0
9:4	Reserved	Must be programmed to the default value	RW	0x0
3:2	SDI_PRSEL	Selection of the number of active data pairs 0x0: 1 pair: DATA0 0x1: 2 pairs: DATA0 and DATA1 0x2: 3 pairs: DATA0, DATA1, and DATA2	RW	0x0
1:0	SDI_BWSEL	Selects the color depth: must be programmed to 0x2 for FlatLink3G 0x0: Reserved 0x1: Reserved 0x2: Color depth is 24 bits.	RW	0x0

Table 15-148. Register Call Summary for Register DSS_SDI_CONTROL

Display Subsystem Basic Programming Model
- SDI Configuration: [0] [1] [2]

Display Subsystem Use Cases and Tips
- Application Example: HVGA Display: [3] [4] [5] [6]
- SDI Software Settings: [7] [8] [9]

Display Subsystem Register Manual
- Display Subsystem Register Mapping Summary: [10]

Table 15-149. DSS_PLL_CONTROL

Address Offset	0x048		
Physical address	0x4805 0048	**Instance**	DISS
Description	This register contains the display subsystem control bits.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved			SDI_PLL_GOBIT	SDI_PLL_LOCKSEL		SDI_PLL_FREQSEL				SDI_PLL_PLLLPMODE	SDI_PLL_LOWCURRSTBY	SDI_PLL_HIGHFREQ	SDI_PLL_SYSRESET	SDI_PLL_STOPMODE		SDI_PLL_REGN				SDI_PLL_REGM											SDI_PLL_IDLE

Bits	Field Name	Description	Type	Reset
31:29	Reserved	Reserved	RW	0x0
28	SDI_PLL_GOBIT	Requests PLL locking sequence. See the programming guide section for the use of this bit in conjunction with DSS_STATUS[6] SDI_PLL_BUSYFLAG 0: Inactive 1: Request PLL locking sequence	RW	0
27:26	SDI_PLL_LOCKSEL	Selects the lock criteria for PLL 0x0: Phase Lock (recommended setting for FlatLink3G) 0x1: Fine Phase Lock 0x2: Frequency Lock	RW	0x0
25:22	SDI_PLL_FREQSEL	PLL internal reference frequency (Fint) range selection 0x3: 0.75MHz to 1.0MHz 0x4: 1.0MHz to 1.25MHz 0x5: 1.25MHz to 1.5MHz 0x6: 1.5MHz to 1.75MHz 0x7: 1.75MHz to 2.1MHz 0xB: 7.5MHz to 10MHz 0xC: 10MHz to 12.5MHz 0xD: 12.5MHz to 15MHz 0xE: 15MHz to 17.5MHz 0xF: 17.5MHz to 21MHz Others: Reserved	RW	0x0
21	SDI_PLL_PLLLPMODE	Select the power/performance of the PLL 0x0: Full performance, minimized jitter 0x1: Reduced power, increased jitter	RW	0
20	SDI_PLL_LOWCURRSTBY	PLL Low Current Standby 0x0: Low Current Standby is not selected. 0x1: Low Current Standby is selected.	RW	0
19	SDI_PLL_HIGHFREQ	Enables a division of pixel clock by 2 before input of PLL. Required for pixel clock frequency above 32 MHz 0x0: Pixel clock is not divided. 0x1: Pixel clock is divided by 2.	RW	0
18	SDI_PLL_SYSRESET	SDI PLL reset bit. Active low default 0x0 : PLL under reset 0x1 : PLL operational	RW	0
17	SDI_PLL_STOPMODE	0xSDI PLL STOPMODE 0x0 : STOPMODE is not selected 0x1 : STOPMODE is selected	RW	0
16:11	SDI_PLL_REGN	SDI PLL REGN register	RW	0x00
10:1	SDI_PLL_REGM	SDI PLL REGM register	RW	0x000
0	SDI_PLL_IDLE	SDI PLL IDLE 0x0: IDLE is not selected 0x1: IDLE is selected	RW	0

Table 15-150. Register Call Summary for Register DSS_PLL_CONTROL

Display Subsystem Integration

- Power Management: [0] [1] [2]

Table 15-150. Register Call Summary for Register DSS_PLL_CONTROL (continued)

Display Subsystem Basic Programming Model • SDI Configuration: [3] [4] [5] • SDI Power-Management Programming Sequence: [6] [7] [8] • Clock Source/Frequency Change Sequence: [9] [10]
Display Subsystem Use Cases and Tips • SDI PLL Configuration: [11] [12] [13] • Application Example: HVGA Display: [14] [15] [16] [17] [18] [19] [20] [21] • SDI Software Settings: [22] [23] [24] [25]
Display Subsystem Register Manual • Display Subsystem Register Mapping Summary: [26] • Display Subsystem and SDI Registers: [27]

Table 15-151. DSS_SDI_STATUS

Address Offset	0x05C		
Physical address	0x4805 005C	**Instance**	DISS
Description	This register contains the display subsystem register.		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																							DSI_PLL_CLK2_STATUS	DSS_DSI_CLK1_STATUS	SDI_PLL_BUSYFLAG	SDI_PLL_LOCK	SDI_PLL_RECAL	SDI_ERROR	SDI_RESET_DONE	DSI_PLL_CLK1_STATUS	DSS_DISPC_CLK1_STATUS

Bits	Field Name	Description	Type	Reset
31:9	Reserved	Reserved	R	0x0000000
8	DSI_PLL_CLK2_ STATUS	DSI2_PLL_FCLK clock selection status (DSI mux) Indicates if the DSI protocol engine is running from the DSI2_PLL_FCLK clock Read 0: DSI2_PLL_FCLK is not selected (unused by DSI). Read 1: DSI2_PLL_FCLK is selected (used by DSI).	R	0
7	DSS_DSI_CLK1_ STATUS	DSS1_ALWON_FCLK clock selection status (DSI mux) Indicates if the DSI protocol engine is running from the DSS1_ALWON_FCLK clock Read 0: DSS1_ALWON_FCLK is not selected (unused by DSI). Read 1: DSS1_ALWON_FCLK is selected (used by DSI).	R	1
6	SDI_PLL_BUSYFLAG	PLL locking sequence status. See the programming guide section for the use of this bit in conjunction with DSS_PLL_CONTROL[28] SDI_PLL_GOBIT Read 0: PLL lock request has completed. Read 1: PLL lock request is in progress.	R	0
5	SDI_PLL_LOCK	SDI PLL lock status See the programming guide section for the use of this bit Read 0: PLL is not locked Read 1: PLL is locked	R	0
4	SDI_PLL_RECAL	SDI DPLL re-calibration status If this bit is active, the PLL must be recalibrated Read 0x0: Recalibration is not required. Read 0x1: Recalibration is required.	R	0

Bits	Field Name	Description	Type	Reset
3	SDI_ERROR	SDI error status bit See programming guide section for error recovery procedure Read 0x0: No error Read 0x1: Error condition required	R	0
2	SDI_RESET_DONE	SDI reset done status This status is delayed until the PLL output has started running Read 0x0: SDI reset is in progress Read 0x1: SDI reset has completed	R	0
1	DSI_PLL_CLK1_ STATUS	DSI1_PLL_FCLK clock selection status (DISPC mux) Indicates if the display controller is running from the DSI1_PLL_FCLK clock Read 0: DSI1_PLL_FCLK is not selected (unused by DISPC). Read 1: DSI1_PLL_FCLK is selected (used by DISPC).	R	0
0	DSS_DISPC_CLK1_ STATUS	DSS1_ALWON_FCLK clock selection status (DISPC mux) Indicates if the display controller is running from the DSS1_ALWON_FCLK clock Read 0: DSS1_ALWON_FCLK is not selected (unused by DISPC). Read 1: DSS1_ALWON_FCLK is selected (used by DISPC).	R	1

Table 15-152. Register Call Summary for Register DSS_SDI_STATUS

Display Subsystem Basic Programming Model • SDI Error Management: [0] [1]
Display Subsystem Use Cases and Tips • SDI PLL Architecture: [2]
Display Subsystem Register Manual • Display Subsystem Register Mapping Summary: [3]

15.7.2.2 Display Controller Registers

Table 15-153. DISPC_REVISION

Address Offset	0x000		
Physical address	0x4805 0400	**Instance**	DISC
Description	This register contains the IP revision code.		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																								REV							

Bits	Field Name	Description	Type	Reset
31:8	Reserved	Write 0s for future compatibility. Read returns 0.	R	0x000000
7:0	REV	IP revision [7:4] Major revision [3:0] Minor revision	R	TI internal data

Table 15-154. Register Call Summary for Register DISPC_REVISION

Display Subsystem Register Manual • Display Subsystem Register Mapping Summary: [0]

Table 15-155. DISPC_SYSCONFIG

Address Offset	0x010		
Physical address	0x4805 0410	**Instance**	DISC
Description	This register allows the control of various parameters of the interconnect interface.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																		MIDLEMODE		Reserved		CLOCKACTIVITY		Reserved			SIDLEMODE		ENWAKEUP	SOFTRESET	AUTOIDLE

Bits	Field Name	Description	Type	Reset
31:14	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0x00000
13:12	MIDLEMODE	Master interface power management, standby/waitcontrol	RW	0x0
		0x0: Force standby. MStandby is asserted only when the module is disabled.		
		0x1: No standby: MStandby is never asserted.		
		0x2: Smart Standby. MStandby is asserted based on the internal activity of the module.		
		0x3: Reserved		
11:10	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0x00
9:8	CLOCKACTIVITY	Clock activity during wakeup mode period	RW	0x0
		0x0: interface and functional clocks can be switched off.		
		0x1: Functional clocks can be switched off and interface clocks are maintained during wakeup period.		
		0x2: Interface clocks can be switched off and functional clocks are maintained during wakeup period.		
		0x3: Interface and functional clocks are maintained during wakeup period.		
7:5	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0x0
4:3	SIDLEMODE	Slave interface power management, idle req/ack control	RW	0x0
		0x0: Force idle. An idle request is acknowledged unconditionally.		
		0x1: No idle. An idle request is never acknowledged.		
		0x2: Smart idle. Idle request is acknowledged based on the internal activity of the module.		
		0x3: Reserved		
2	ENWAKEUP	Wakeup feature control	RW	0
		0x0: Wakeup is disabled.		
		0x1: Wakeup is enabled.		
1	SOFTRESET	Software reset. Set this bit to 1 to trigger a module reset. The bit is automatically reset by the hardware. During reads, it always returns 0.	RW	0
		0x0: Normal mode		
		0x1: The module is reset.		
0	AUTOIDLE	Internal interface clock gating strategy	RW	1
		0x0: Interface clock is free-running.		
		0x1: Automatic L3 and L4 interface clock gating strategy is applied based on interface activity.		

Table 15-156. Register Call Summary for Register DISPC_SYSCONFIG

Display Subsystem Integration
- Resets: [0]
- Power Management: [1] [2] [3] [4] [5] [6] [7] [8] [9] [10] [11]

Display Subsystem Basic Programming Model
- Display Controller Configuration: [12]

Display Subsystem Use Cases and Tips
- Autoidle and Smart Idle: [13] [14]
- Configure the DISPC: [15] [16] [17] [18]
- Display Subsystem Use Cases and Tips: [19] [20] [21] [22]

Display Subsystem Register Manual
- Display Subsystem Register Mapping Summary: [23]

Table 15-157. DISPC_SYSSTATUS

Address Offset	0x014		
Physical address	0x4805 0414	**Instance**	DISC
Description	This register provides status information about the module, excluding interrupt status information.		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																								Reserved							RESETDONE

Bits	Field Name	Description	Type	Reset
31:8	Reserved	Write 0s for future compatibility. Read returns 0.	R	0x000000
7:1	Reserved	Reserved. Read returns 0.	R	0x00
0	RESETDONE	Internal reset monitoring	R	0
		Read 0x0: Internal module reset is ongoing.		
		Read 0x1: Reset complete		

Table 15-158. Register Call Summary for Register DISPC_SYSSTATUS

Display Subsystem Basic Programming Model
- Display Controller Configuration: [0]

Display Subsystem Register Manual
- Display Subsystem Register Mapping Summary: [1]

Table 15-159. DISPC_IRQSTATUS

Address Offset	0x018		
Physical address	0x4805 0418	**Instance**	DISC
Description	This register regroups all the status of module internal events that generate an interrupt. A write of 1 to a given bit resets the bit.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																WAKEUP	SYNCLOSTDIGITAL	SYNCLOST	VID2ENDWINDOW	VID2FIFOUNDERFLOW	VID1ENDWINDOW	VID1FIFOUNDERFLOW	OCPERROR	PALETTEGAMMALOADING	GFXENDWINDOW	GFXFIFOUNDERFLOW	PROGRAMMEDLINENUMBER	ACBIASCOUNTSTATUS	EVSYNC_ODD	EVSYNC_EVEN	VSYNC	FRAMEDONE

Bits	Field Name	Description		Type	Reset
31:17	Reserved	Write 0s for future compatibility. Read returns 0		RW	0x0000
16	WAKEUP	Wakeup		RW	0
		Read 0x0:	Wakeup is false.		
		Write 0x0:	Wakeup status bit unchanged		
		Read 0x1:	Wakeup is true (pending).		
		Write 0x1:	Wakeup status bit reset		
15	SYNCLOSTDIGITAL	SyncLostDigital		RW	0
		Read 0x0:	SyncLostDigital is false.		
		Write 0x0:	SyncLostDigital status bit unchanged		
		Read 0x1:	SyncLostDigital is true (pending).		
		Write 0x1:	SyncLostDigital status bit reset		
14	SYNCLOST	SyncLost		RW	0
		Read 0x0:	SyncLost is false.		
		Write 0x0:	SyncLost status bit unchanged		
		Read 0x1:	SyncLost is true (pending).		
		Write 0x1:	SyncLost status bit reset		
13	VID2ENDWINDOW	Vid2EndWindow		RW	0
		Read 0x0:	Vid2EndWindow is false.		
		Write 0x0:	Vid2EndWindow status bit unchanged		
		Read 0x1:	Vid2EndWindow is true (pending).		
		Write 0x1:	Vid2EndWindow status bit reset		
12	VID2FIFOUNDERFLOW	Vid2FIFOUnderflow		RW	0
		Read 0x0:	Vid2FIFOUnderflow is false.		
		Write 0x0:	Vid2FIFOUnderflow status bit unchanged		
		Read 0x1:	Vid2FIFOUnderflow is true (pending).		
		Write 0x1:	Vid2FIFOUnderflow status bit reset		
11	VID1ENDWINDOW	Vid1EndWindow		RW	0
		Read 0x0:	Vid1EndWindow is false.		
		Write 0x0:	Vid1EndWindow status bit unchanged		
		Read 0x1:	Vid1EndWindow is true (pending).		
		Write 0x1:	Vid1EndWindow status bit reset		
10	VID1FIFOUNDERFLOW	Vid1FIFOUnderflow		RW	0
		Read 0x0:	Vid1FIFOUnderflow is false.		
		Write 0x0:	Vid1FIFOUnderflow status bit unchanged		
		Read 0x1:	Vid1FIFOUnderflow is true (pending).		
		Write 0x1:	Vid1FIFOUnderflow status bit reset		
9	OCPERROR	OCPError		RW	0
		Read 0x0:	OCPError is false.		

Bits	Field Name	Description		Type	Reset
		Write 0x0:	OCPError status bit unchanged		
		Read 0x1:	OCPError is true (pending).		
		Write 0x1:	OCPError status bit reset		
8	PALETTEGAMMA LOADING	PaletteGammaLoading		RW	0
		Read 0x0:	PaletteGammaLoading is false.		
		Write 0x0:	PaletteGammaLoading status bit unchanged		
		Read 0x1:	PaletteGammaLoading is true (pending).		
		Write 0x1:	PaletteGammaLoading status bit reset		
7	GFXENDWINDOW	GfxEndWindow		RW	0
		Read 0x0:	GfxEndWindow is false.		
		Write 0x0:	GfxEndWindow status bit unchanged		
		Read 0x1:	GfxEndWindow is true (pending).		
		Write 0x1:	GfxEndWindow status bit reset		
6	GFXFIFOUNDERFLOW	GfxFIFOUnderflow		RW	0
		Read 0x0:	GfxFIFOUnderflow is false.		
		Write 0x0:	GfxFIFOUnderflow status bit unchanged		
		Read 0x1:	GfxFIFOUnderflow is true (pending).		
		Write 0x1:	GfxFIFOUnderflow status bit reset		
5	PROGRAMMEDLINE NUMBER	ProgrammedLineNumber		RW	0
		Read 0x0:	ProgrammedLineNumber is false.		
		Write 0x0:	ProgrammedLineNumber status bit unchanged		
		Read 0x1:	ProgrammedLineNumber is true (pending).		
		Write 0x1:	ProgrammedLineNumber status bit reset		
4	ACBIASCOUNTSTATUS	ACBiasCountStatus		RW	0
		Read 0x0:	ACBiasCountStatus is false.		
		Write 0x0:	ACBiasCountStatus status bit unchanged		
		Read 0x1:	ACBiasCountStatus is true (pending).		
		Write 0x1:	ACBiasCountStatus status bit reset		
3	EVSYNC_ODD	EVSYNC_ODD		RW	0
		Read 0x0:	EVSYNC_ODD is false.		
		Write 0x0:	EVSYNC_ODD status bit unchanged		
		Read 0x1:	EVSYNC_ODD is true (pending).		
		Write 0x1:	EVSYNC_ODD status bit reset		
2	EVSYNC_EVEN	EVSYNC_EVEN		RW	0
		Read 0x0:	EVSYNC_EVEN is false.		
		Write 0x0:	EVSYNC_EVEN status bit unchanged		
		Read 0x1:	EVSYNC_EVEN is true (pending).		
		Write 0x1:	EVSYNC_EVEN status bit reset		
1	VSYNC	VSYNC		RW	0
		Read 0x0:	VSYNC is false.		
		Write 0x0:	VSYNC status bit unchanged		
		Read 0x1:	VSYNC is true (pending).		
		Write 0x1:	VSYNC status bit reset		
0	FRAMEDONE	FrameDone		RW	0
		Read 0x0:	FrameDone is false.		
		Write 0x0:	FrameDone status bit unchanged		
		Read 0x1:	FrameDone is true (pending).		
		Write 0x1:	FrameDone status bit reset		

Table 15-160. Register Call Summary for Register DISPC_IRQSTATUS

Display Subsystem Integration
- Interrupt Requests: [0] [1]

Display Subsystem Basic Programming Model
- Display Subsystem Reset: [2] [3]
- Display Controller Configuration: [4]
- TV Set-Specific Control Registers: [5]
- Video Encoder Programming Sequence: [6]

Display Subsystem Use Cases and Tips
- Programming Flow: [7] [8]

Display Subsystem Register Manual
- Display Subsystem Register Mapping Summary: [9]
- Display Subsystem and SDI Registers: [10]

Table 15-161. DISPC_IRQENABLE

Address Offset	0x01C		
Physical address	0x4805 041C	**Instance**	DISC
Description	This register allows the masking/unmasking of module internal interrupt sources, on an event-by-event basis.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																WAKEUP	SYNCLOSTDIGITAL	SYNCLOST	VID2ENDWINDOW	VID2FIFOUNDERFLOW	ENDVID1WINDOW	VID1FIFOUNDERFLOW	OCPERROR	PALETTEGAMMAMASK	GFXENDWINDOW	GFXFIFOUNDERFLOW	PROGRAMMEDLINENUMBER	ACBIASCOUNTSTATUS	EVSYNC_ODD	EVSYNC_EVEN	VSYNC	FRAMEMASK

Bits	Field Name	Description	Type	Reset
31:17	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0x0000
16	WAKEUP	Wakeup mask 0x0: Wakeup is masked. 0x1: Wakeup generates an interrupt when it occurs.	RW	0
15	SYNCLOSTDIGITAL	SyncLostDigital 0x0: SyncLostDigital is masked. 0x1: SyncLostDigital generates an interrupt when it occurs.	RW	0
14	SYNCLOST	SyncLost 0x0: SyncLost is masked. 0x1: SyncLost generates an interrupt when it occurs.	RW	0
13	VID2ENDWINDOW	Vid2EndWindow 0x0: Vid2EndWindow is masked. 0x1: Vid2EndWindow generates an interrupt when it occurs.	RW	0
12	VID2FIFOUNDERFLOW	Vid2FIFOUnderflow 0x0: Vid2FIFOUnderflow is masked. 0x1: Vid2FIFOUnderflow generates an interrupt when it occurs.	RW	0
11	ENDVID1WINDOW	EndVid1Window 0x0: EndVid1Window is masked.	RW	0

Bits	Field Name	Description	Type	Reset
		0x1: EndVid1Window generates an interrupt when it occurs.		
10	VID1FIFOUNDERFLOW	Vid1FIFOUnderflow	RW	0
		0x0: Vid1FIFOUnderflow is masked.		
		0x1: Vid1FIFOUnderflow generates an interrupt when it occurs.		
9	OCPERROR	OCPError	RW	0
		0x0: OCPError is masked.		
		0x1: OCPError generates an interrupt when it occurs.		
8	PALETTEGAMMAMASK	PaletteGammaMask	RW	0
		0x0: PaletteGammaMask is masked.		
		0x1: PaletteGammaMask generates an interrupt when it occurs.		
7	GFXENDWINDOW	GfxEndWindow	RW	0
		0x0: GfxEndWindow is masked.		
		0x1: GfxEndWindow generates an interrupt when it occurs.		
6	GFXFIFOUNDERFLOW	GfxFIFOUnderflow	RW	0
		0x0: GfxFIFOUnderflow is masked.		
		0x1: GfxFIFOUnderflow generates an interrupt when it occurs.		
5	PROGRAMMEDLINE NUMBER	ProgrammedLineNumber	RW	0
		0x0: ProgrammedLineNumber is masked.		
		0x1: ProgrammedLineNumber generates an interrupt when it occurs.		
4	ACBIASCOUNTSTATUS	ACBiasCountStatus	RW	0
		0x0: ACBiasCountStatus is masked.		
		0x1: ACBiasCountStatus generates an interrupt when it occurs.		
3	EVSYNC_ODD	EVSYNC_ODD	RW	0
		0x0: EVSYNC_ODD is masked.		
		0x1: EVSYNC_ODD generates an interrupt when it occurs.		
2	EVSYNC_EVEN	EVSYNC_EVEN	RW	0
		0x0: EVSYNC_EVEN is masked.		
		0x1: EVSYNC_EVEN generates an interrupt when it occurs.		
1	VSYNC	VSYNC	RW	0
		0x0: VSYNC is masked.		
		0x1: VSYNC generates an interrupt when it occurs.		
0	FRAMEMASK	FrameMask	RW	0
		0x0: FrameMask is masked.		
		0x1: FrameMask generates an interrupt when it occurs.		

Table 15-162. Register Call Summary for Register DISPC_IRQENABLE

Display Subsystem Integration

- Interrupt Requests: [0] [1]

Display Subsystem Basic Programming Model

- Display Controller Configuration: [2]
- TV Set-Specific Control Registers: [3]
- Video Encoder Programming Sequence: [4] [5] [6]

Table 15-162. Register Call Summary for Register DISPC_IRQENABLE (continued)

Display Subsystem Use Cases and Tips

- Programming Flow: [7] [8] [9] [10] [11]
- Configure the DISPC: [12]
- Display Subsystem Use Cases and Tips: [13]

Display Subsystem Register Manual

- Display Subsystem Register Mapping Summary: [14]

Table 15-163. DISPC_CONTROL

Address Offset	0x040		
Physical address	0x4805 0440	**Instance**	DISC
Description	The control register configures the display controller module.		
Type	RW		

31 30	29	28	27	26 25	24	23 22	21	20	19 18 17 16	15	14	13	12	11	10	9 8	7	6	5	4	3	2	1	0
SPATIALTEMPORALDITHERINGFRAMES	LCDENABLEPOL	LCDENABLESIGNAL	PCKFREEENABLE	TDMUNUSEDBITS	TDMCYCLEFORMAT	TDMPARALLELMODE	TDMENABLE	HT	GPOUT1	GPOUT0	GPIN1	GPIN0	OVERLAYOPTIMIZATION	STALLMODE	Reserved	TFTDATALINES	STDITHERENABLE	GODIGITAL	GOLCD	M8B	STNTFT	MONOCOLOR	DIGITALENABLE	LCDENABLE

Bits	Field Name	Description	Type	Reset
31:30	SPATIALTEMPORAL DITHERINGFRAMES	Spatial/Temporal dithering number of frames wr: VFP 0x0: Spatial only 0x1: Spatial and temporal over two frames 0x2: Spatial and temporal over four frames 0x3: Reserved	RW	0x0
29	LCDENABLEPOL	LCD Enable Signal Polarity 0x0: Active low 0x1: Active high	RW	0
28	LCDENABLESIGNAL	LCD Enable Signal: LCD interface active/inactive 0x0: Signal disabled 0x1: Signal enabled	RW	0
27	PCKFREEENABLE	Pixel clock free-running enabled/disabled 0x0: Clock disabled 0x1: Clock enabled	RW	0
26:25	TDMUNUSEDBITS	State of unused bits (TDM mode only) WR: VFP 0x0: Low level (0) 0x1: High level (1) 0x2: Unchanged from previous state 0x3: Reserved	RW	0x0

Bits	Field Name	Description	Type	Reset
24:23	TDMCYCLEFORMAT	Cycle format (TDM mode only) WR: VFP 0x0: 1 cycle for 1 pixel 0x1: 2 cycles for 1 pixel 0x2: 3 cycles for 1 pixel 0x3: 3 cycles for 2 pixels	RW	0x0
22:21	TDMPARALLELMODE	Output Interface width (TDM mode only) WR: VFP 0x0: 8-bit parallel output interface selected 0x1: 9-bit parallel output interface selected 0x2: 12-bit parallel output interface selected 0x3: 16-bit parallel output interface selected	RW	0x0
20	TDMENABLE	Enable the multiple cycle format (TDM mode used only for Active Matrix mode with the RFBI enable bit off). WR: VFP 0x0: TDM disabled 0x1: TDM enabled	RW	0
19:17	HT	Hold Time for digital output WR: EVSYNC Encoded value (from 0 to 7) holds time for digital output. The data will be held for (HT + 1) external digital clock periods.	RW	0x0
16	GPOUT1	General Purpose Output Signal 0x0: The GPout1 is reset. 0x1: The GPout1 is set.	RW	0
15	GPOUT0	General Purpose Output Signal 0x0: The GPout0 is reset. 0x1: The GPout0 is set.	RW	0
14	GPIN1	General Purpose Input Signal WR: VFP Read 0x0: The GPin1 has been reset. Read 0x1: The GPin1 has been set.	R	0
13	GPIN0	General Purpose Input Signal WR: VFP Read 0x0: The GPin0 has been reset. Read 0x1: The GPin0 has been set.	R	0
12	OVERLAYOPTIMIZATION	Overlay Optimization (available when graphics format is NOT is 1-, 2, and 4-BPP) WR: VFP or EVSYNC 0x0: Graphics data below video1 window fetched from memory or no overlap between graphics and video1 windows. 0x1: Graphics data below video1 window not fetched from memory.	RW	0
11	STALLMODE	Stall mode for the LCD output wr: VFP 0x0: Normal mode selected 0x1: Stall mode selected. The Display Controller sends the data without considering the VSYNC/HSYNC. The LCD output is disabled at the end of the transfer of the frame. The S/W has to re-enable the LCD output to generate a new frame. The stall mode is used in RFBI and DSI command modes.	RW	0
10	Reserved	Reserved for non-GP devices	RW	0

Bits	Field Name	Description	Type	Reset
9:8	TFTDATALINES	Number of lines of the LCD interface WR: VFP 0x0: 12-bit output aligned on the LSB of the pixel data interface 0x1: 16-bit output aligned on the LSB of the pixel data interface 0x2: 18-bit output aligned on the LSB of the pixel data interface 0x3: 24-bit output aligned on the LSB of the pixel data interface	RW	0x0
7	STDITHERENABLE	Spatial temporal dithering enable WR: VFP 0x0: Spatial/temporal dithering logic disabled 0x1: Spatial/temporal dithering logic enabled	RW	0
6	GODIGITAL	Digital GO Command 0x0: The hardware has finished updating the internal shadow registers of the pipeline(s) associated with the digital output using the user values. The hardware resets the bit when the update is completed. 0x1: Users have finished programming the shadow registers of the pipeline(s) associated with the digital output and the hardware can update the internal registers at the external VSYNC.	RW	0
5	GOLCD	LCD GO Command 0x0: The hardware has finished updating the internal shadow registers of the pipeline(s) connected to the LCD output using the user values. The hardware resets the bit when the update is completed. 0x1: Users have finished programming the shadow registers of the pipeline(s) associated with the LCD output and the hardware can update the internal registers at the VFP start period.	RW	0
4	M8B	Mono 8-bit mode WR: VFP 0x0: Pixel data [3:0] is used to output four pixel values to the panel at each pixel clock transition (only in Passive Mono 8-bit mode). 0x1: Pixel data [7:0] is used to output eight pixel values to the panel each pixel clock transition (only in Passive Mono 8-bit mode).	RW	0
3	STNTFT	LCD display type WR: VFP 0x0: Passive or Passive Matrix display operation enabled. Passive Matrix dither logic enabled. 0x1: Active Matrix display operation enabled. Passive Matrix Dither logic and output FIFO bypassed.	RW	0
2	MONOCOLOR	Monochrome/Color WR: VFP 0x0: Color operation enabled (Passive Matrix mode only) 0x1: Monochrome operation enabled (Passive Matrix mode only)	RW	0
1	DIGITALENABLE	Digital enable 0x0: Digital output disabled (at the end of the current field if interlace output when the bit is reset) 0x1: Digital output enabled	RW	0
0	LCDENABLE	LCD enable 0x0: LCD output disabled (at the end of the frame when the bit is reset) 0x1: LCD output enabled	RW	0

Table 15-164. Register Call Summary for Register DISPC_CONTROL

Display Subsystem Environment
- Parallel Interface: [0] [1] [2] [3]
- Video Port (VP) Interface: [4]

Display Subsystem Integration
- Clocks: [5] [6]
- Interrupt Requests: [7] [8]

Display Subsystem Functional Description
- Overlay Support: [9]
- Active/Passive Matrix Display Data Path: [10]
- DSI Transfer Modes: [11]

Display Subsystem Basic Programming Model
- Display Subsystem Reset: [12]
- Display Controller Basic Programming Model: [13] [14] [15] [16] [17] [18] [19]
- Graphics Layer Configuration: [20] [21] [22] [23]
- Video Layer Configuration: [24] [25] [26] [27] [28]
- Rotation/Mirroring Display Subsystem Settings: [29]
- LCD-Specific Control Registers: [30] [31] [32] [33] [34] [35] [36] [37] [38] [39] [40] [41] [42] [43] [44] [45] [46] [47] [48] [49] [50] [51] [52] [53] [54] [55] [56]
- TV Set-Specific Control Registers: [57] [58] [59] [60] [61] [62] [63] [64]
- DSI Programming Sequence Example: [65] [66] [67]
- DISPC Control Registers: [68] [69] [70] [71] [72] [73]
- RFBI Control Registers: [74]
- Video Encoder Programming Sequence: [75] [76]
- SDI Configuration: [77] [78] [79]
- SDI Error Management: [80]

Display Subsystem Use Cases and Tips
- SDI Software Settings: [81] [82] [83]
- Programming Flow: [84] [85] [86] [87] [88] [89] [90] [91] [92] [93] [94] [95]
- Configure the DISPC: [96] [97] [98] [99] [100] [101] [102] [103] [104]
- Enable Video Mode Using the DISPC Video Port: [105] [106] [107]
- Display Subsystem Use Cases and Tips: [108] [109] [110] [111] [112] [113] [114] [115] [116] [117]

Display Subsystem Register Manual
- Display Subsystem Register Mapping Summary: [118]
- Display Controller Registers: [119]

Table 15-165. DISPC_CONFIG

Address Offset	0x044		
Physical address	0x4805 0444	**Instance**	DISC
Description	This control register configures the display controller module. Shadow register, updated on VFP start period or EVSYNC		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED												TVALPHABLENDERENABLE	LCDALPHABLENDERENABLE	FIFOFILLING	FIFOHANDCHECK	CPR	FIFOMERGE	TCKDIGSELECTION	TCKDIGENABLE	TCKLCDSELECTION	TCKLCDENABLE	FUNCGATED	ACBIASGATED	VSYNCGATED	HSYNCGATED	PIXELCLOCKGATED	PIXELDATAGATED	PALETTEGAMMATABLE	LOADMODE		PIXELGATED

Bits	Field Name	Description	Type	Reset
31: 20	Reserved	Write 0s for future compatibility. Read returns 0	RW	0x00000
19	TVALPHABLENDER ENABLE	Selects the alpha blender (TV output) 0x0: Alpha blender is disabled. 0x1: The alpha blender is enabled.	RW	0
18	LCDALPHABLENDER ENABLE	Selects the alpha blender (LCD output) 0x0: Alpha blender is disabled. 0x1: The alpha blender is enabled.	RW	0
17	FIFOFILLING	Controls if the FIFO are refilled only when the LOW threshold is reached or if all FIFO are refilled when at least one of them reaches the LOW threshold. 0x0: Each FIFO is refilled when it reaches LOW threshold. 0x1: All FIFOs are refilled up to high threshold when at least one of them reaches the LOW threshold. (only active FIFOs should be considered and when reaching the end of the frame the FIFO goes to empty condition so no need to fill it again).	RW	0
16	FIFOHANDCHECK	Controls the handshake between FIFO and RFBI STALL to prevent from underflow. The bit should be set to 0 when the module is not in STALL mode. 0x0: Only the STALL signal from RFBI is used regardless of the FIFO fullness information to provide data to the RFBI module. 0x1: The STALL signal from RFBI is used in combination with the FIFO fullness information to provide data to the RFBI module only when it does not generated FIFO underflow.	RW	0
15	CPR	Color phase rotation control wr: VFP 0x0: Color phase rotation disabled 0x1: Color phase rotation enabled	RW	0
14	FIFOMERGE	FIFO merge control wr: EVSYNC or VFP 0x0: FIFO merge disabled Each FIFO is dedicated to one pipeline. 0x1: FIFO merge enabled All the FIFOS are merged into a single one to be used by the single active pipeline.	RW	0
13	TCKDIGSELECTION	Transparency color key selection (digital output) wr: EVSYNC 0x0: Graphics destination transparency color key selected in normal mode or graphics source transparency color key selected in alpha mode 0x1: Video source transparency color key selected in normal mode	RW	0
12	TCKDIGENABLE	Transparency color key enabled (digital output) wr: EVSYNC 0x0: Disable the transparency color key for digital output 0x1: Enable the transparency color key for digital output	RW	0
11	TCKLCDSELECTION	Transparency color key selection (LCD output) WR: VFP 0x0: Graphics destination transparency color key selected in normal mode or graphics source transparency color key selected in alpha mode 0x1: Video source transparency color key selected in normal mode	RW	0
10	TCKLCDENABLE	Transparency color key enabled (LCD output) WR: VFP *	RW	0

Bits	Field Name	Description	Type	Reset
		0x0: Disable the transparency color key for the LCD		
		0x1: Enable the transparency color key for the LCD		
9	FUNCGATED	Functional clocks gated enabled WR: immediate	RW	0
		0x0: Functional clocks gated disabled		
		0x1: Functional clocks gated enabled		
8	ACBIASGATED	ACBias Gated Enabled WR: VFP	RW	0
		0x0: AcBias Gated Disabled		
		0x1: AcBias Gated Enabled		
7	VSYNCGATED	VSYNC Gated Enabled WR: VFP	RW	0
		0x0: VSYNC Gated Disabled		
		0x1: VSYNC Gated Enabled		
6	HSYNCGATED	HSYNC Gated Enabled WR: VFP	RW	0
		0x0: HSYNC Gated Disabled		
		0x1: HSYNC Gated Enabled		
5	PIXELCLOCKGATED	Pixel Clock Gated Enabled WR: VFP	RW	0
		0x0: Pixel Clock Gated Disabled		
		0x1: Pixel Clock Gated Enabled		
4	PIXELDATAGATED	Pixel Data Gated Enabled WR: VFP	RW	0
		0x0: Pixel Data Gated Disabled		
		0x1: Pixel Data Gated Enabled		
3	PALETTEGAMMATABLE	Palette/Gamma Table selection WR: EVSYNC or VFP	RW	0
		0x0: LUT used as palette (only if graphics format is BITMAP1, 2, 4, and 8)		
		0x1: LUT used as gamma table (only if graphics format is NOT BITMAP1, 2, 4, and 8 or no graphics window present)		
2:1	LOADMODE	Loading Mode for the Palette/Gamma Table WR: EVSYNC or VFP	RW	0x0
		0x0: Palette/Gamma Table and data are loaded every frame.		
		0x1: Palette/Gamma Table to be loaded. Users set the bit when the palette/gamma table has to be loaded. H/W resets the bit when table has been loaded. (DISPC_GFX_ATTRIBUTES. GfxEnable has to be set to 1).		
		0x2: Frame data only loaded every frame		
		0x3: Palette/Gamma Table and frame data loaded on first frame then switch to 10 (H/W).		
0	PIXELGATED	Pixel Gated Enable (only for Active Matrix Display) WR: VFP	RW	0
		0x0: Pixel clock always toggles (only in Active Matrix mode)		
		0x1: Pixel clock only toggles when there is valid data to display. (only in Active Matrix mode)		

Table 15-166. Register Call Summary for Register DISPC_CONFIG

Display Subsystem Environment

- Parallel Interface: [0]

Table 15-166. Register Call Summary for Register DISPC_CONFIG (continued)

Display Subsystem Integration • Power Management: [1] [2]
Display Subsystem Functional Description • Overlay Support: [3] [4] [5] [6] [7] [8] [9] [10] [11] [12] [13] [14]
Display Subsystem Basic Programming Model • Display Controller Basic Programming Model: [15] • Graphics Layer Configuration: [16] [17] [18] [19] • Video Layer Configuration: [20] [21] [22] • LCD-Specific Control Registers: [23] [24] [25] [26] [27] [28] [29] [30] [31] [32] • TV Set-Specific Control Registers: [33] [34] [35] [36]
Display Subsystem Use Cases and Tips • Autoidle and Smart Idle: [37] • Programming Flow: [38] [39]
Display Subsystem Register Manual • Display Subsystem Register Mapping Summary: [40]

Table 15-167. DISPC_DEFAULT_COLOR_m

Address Offset	0x04C + m * 0x04	**Indexm**	m = 0 to 1
Physical address	0x4805 044C + m * 0x04	**Instance**	DISPC
Description	The control register allows to configure the default solid background color for the LCD (DISPC_DEFAULT_COLOR_0) and for 24-bit digital output (DISPC_DEFAULT_COLOR_1). Shadow register, updated on VFP start period for DISPC_DEFAULT_COLOR_0 and EVSYNC for DISPC_DEFAULT_COLOR_1		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								DEFAULTCOLOR																							

Bits	Field Name	Description	Type	Reset
31:24	Reserved	Write 0s for future compatibility. Read returns 0	RW	0x00
23:0	DEFAULTCOLOR	24-bit RGB color value to specify the default solid color to display when there is no data from the overlays.	RW	0x000000

Table 15-168. Register Call Summary for Register DISPC_DEFAULT_COLOR_m

Display Subsystem Basic Programming Model • Display Controller Basic Programming Model: [0] [1] • LCD-Specific Control Registers: [2] [3] • TV Set-Specific Control Registers: [4] [5]
Display Subsystem Register Manual • Display Subsystem Register Mapping Summary: [6]

Table 15-169. DISPC_TRANS_COLOR_m

Address Offset	0x054 + m * 0x04	**Index**	m = 0 to 1
Physical address	0x4805 0454 + m * 0x04	**Instance**	DISPC
Description	The register sets the transparency color value for the video/graphics overlays for the LCD output (DISPC_TRANS_COLOR_0) for 24-bit digital output(DISPC_TRANS_COLOR_1). Shadow register, updated on VFP start period for DISPC_TRANS_COLOR_0 and EVSYNC for DISPC_TRANS_COLOR_1		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								TRANSCOLORKEY																							

Bits	Field Name	Description	Type	Reset
31:24	Reserved	Write 0s for future compatibility. Read returns 0	RW	0x00
23:0	TRANSCOLORKEY	Transparency Color Key Value in RGB format [0] BITMAP 1 (CLUT), [23,1] set to 0s [1:0] BITMAP 2 (CLUT), [23,2] set to 0s [3:0] BITMAP 4 (CLUT), [23,4] set to 0s [7:0] BITMAP 8 (CLUT), [23,8] set to 0s [11:0] RGB 12, [23,12] set to 0s [15:0] RGB 16, [23,16] set to 0s [23:0] RGB 24	RW	0x000000

Table 15-170. Register Call Summary for Register DISPC_TRANS_COLOR_m

Display Subsystem Basic Programming Model

- Display Controller Basic Programming Model: [0] [1]
- LCD-Specific Control Registers: [2]
- TV Set-Specific Control Registers: [3] [4]

Display Subsystem Register Manual

- Display Subsystem Register Mapping Summary: [5]

Table 15-171. DISPC_LINE_STATUS

Address Offset	0x05C		
Physical address	0x4805 045C	**Instance**	DISC
Description	The control register indicates the current LCD panel display line number.		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																					LINENUMBER										

Bits	Field Name	Description	Type	Reset
31:11	Reserved	Write 0s for future compatibility. Read returns 0	R	0x000000
10:0	LINENUMBER	Current LCD panel line number Current display line number. The first active line has the value 0. During blanking lines the line number is not incremented.	R	0x7FF

Table 15-172. Register Call Summary for Register DISPC_LINE_STATUS

Display Subsystem Register Manual

- Display Subsystem Register Mapping Summary: [0]

Table 15-173. DISPC_LINE_NUMBER

Address Offset	0x060		
Physical address	0x4805 0460	**Instance**	DISC
Description	The control register indicates the LCD panel display line number for the interrupt and the DMA request. Shadow register, updated on VFP start period.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																					LINENUMBER										

Bits	Field Name	Description	Type	Reset
31:11	Reserved	Write 0s for future compatibility. Read returns 0	RW	0x000000
10:0	LINENUMBER	LCD panel line number programming LCD line number defines the line on which the programmable interrupt is generated and the DMA request occurs.	RW	0x000

Table 15-174. Register Call Summary for Register DISPC_LINE_NUMBER

Display Subsystem Integration
- DMA Requests: [0]

Display Subsystem Basic Programming Model
- Display Controller Basic Programming Model: [1]

Display Subsystem Register Manual
- Display Subsystem Register Mapping Summary: [2]

Table 15-175. DISPC_TIMING_H

Address Offset	0x064		
Physical address	0x4805 0464	**Instance**	DISC
Description	The register configures the timing logic for the HSYNC signal. Shadow register, updated on VFP start period		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
HBP																HFP								HSW							

Bits	Field Name	Description	Type	Reset
31:20	HBP	Horizontal Back Porch. Encoded value (from 1 to 4096) to specify the number of pixel clock periods to add to the beginning of a line transmission before the first set of pixels is output to the display (program to value minus 1).	RW	0x00
19:8	HFP	Horizontal front porch. Encoded value (from 1 to 4096) to specify the number of pixel clock periods to add to the end of a line transmission before line clock is asserted (program to value minus 1).	RW	0x00
7:0	HSW	Horizontal synchronization pulse width Encoded value (from 1 to 256) to specify the number of pixel clock periods to pulse the line clock at the end of each line (program to value minus 1).	RW	0x00

Table 15-176. Register Call Summary for Register DISPC_TIMING_H

Display Subsystem Environment
- Parallel Interface: [0] [1] [2]

Table 15-176. Register Call Summary for Register DISPC_TIMING_H (continued)

Display Subsystem Basic Programming Model • Display Controller Basic Programming Model: [3] • LCD-Specific Control Registers: [4] [5] [6] [7] [8] [9] [10]
Display Subsystem Use Cases and Tips • Vertical and Horizontal Timings: [11] [12] [13] • Programming Flow: [14] [15] [16] [17] • Configure the DISPC: [18]
Display Subsystem Register Manual • Display Subsystem Register Mapping Summary: [19]

Table 15-177. DISPC_TIMING_V

Address Offset	0x068		
Physical address	0x4805 0468	**Instance**	DISC
Description	The register configures the timing logic for the VSYNC signal. Shadow register, updated on VFP start period		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
VBP								VFP																VSW							

Bits	Field Name	Description	Type	Reset
31:20	VBP	Vertical back porch Encoded value (from 0 to 4095) to specify the number of line clock periods to add to the beginning of a frame before the first set of pixels is output to the display.	RW	0x00
19:8	VFP	Vertical front porch Encoded value (from 0 to 4095) to specify the number of line clock periods to add to the end of each frame.	RW	0x00
7:0	VSW	Vertical synchronization pulse width In active mode, encoded value (from 1 to 256) to specify the number of line clock periods (program to value minus one) to pulse the frame clock (VSYNC) pin at the end of each frame after the end of frame wait (VFP) period elapses. Frame clock uses as VSYNC signal in active mode. In passive mode, encoded value (from 1 to 256) to specify the number of extra line clock periods (program to value minus one) to insert after the vertical front porch (VFP) period has elapsed.	RW	0x00

Table 15-178. Register Call Summary for Register DISPC_TIMING_V

Display Subsystem Environment • Parallel Interface: [0] [1] [2]
Display Subsystem Basic Programming Model • Display Controller Basic Programming Model: [3] • LCD-Specific Control Registers: [4] [5] [6] [7] [8] [9] [10]
Display Subsystem Use Cases and Tips • Vertical and Horizontal Timings: [11] [12] [13] • Programming Flow: [14] [15] [16] [17] • Configure the DISPC: [18]
Display Subsystem Register Manual • Display Subsystem Register Mapping Summary: [19]

Table 15-179. DISPC_POL_FREQ

Address Offset	0x06C		
Physical address	0x4805 046C	**Instance**	DISC
Description	The register configures the signal configuration. Shadow register, updated on VFP start period		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														ONOFF	RF	IEO	IPC	IHS	IVS	ACBI				ACB							

Bits	Field Name	Description	Type	Reset
31:18	Reserved	Write 0s for future compatibility. Read returns 0	RW	0x0000
17	ONOFF	HSYNC/VSYNC Pixel clock Control On/Off 0x0: HSYNC and VSYNC are driven on opposite edges of pixel clock than pixel data 0x1: HSYNC and VSYNC are driven according to bit 16	RW	0
16	RF	Program HSYNC/VSYNC Rise or Fall 0x0: HSYNC and VSYNC are driven on falling edge of pixel clock (if bit 17 set to 1) 0x1: HSYNC and VSYNC are driven on the rising edge of pixel clock (if bit 17 set to 1)	RW	0
15	IEO	Invert output enable 0x0: Ac-bias is active high (active display mode) 0x1: Ac-bias is active low (active display mode)	RW	0
14	IPC	Invert pixel clock 0x0: Data is driven on the LCD data lines on the rising-edge of the pixel clock 0x1: Data is driven on the LCD data lines on the falling-edge of the pixel clock	RW	0
13	IHS	Invert HSYNC 0x0: Line clock pin is active high and inactive low 0x1: Line clock pin is active low and inactive high	RW	0
12	IVS	Invert VSYNC 0x0: Frame clock pin is active high and inactive low 0x1: Frame clock pin is active low and inactive high	RW	0
11:8	ACBI	AC-bias pin transitions per interrupt Value (from 0 to 15) used to specify the number of AC Bias pin transitions	RW	0x0
7:0	ACB	AC-bias pin frequency Value (from 0 to 255) used to specify the number of line clocks to count before transitioning the ac-bias pin. This pin is used to periodically invert the polarity of the power supply to prevent DC charge build-up within the display.	RW	0x00

Table 15-180. Register Call Summary for Register DISPC_POL_FREQ

Display Subsystem Environment

- Parallel Interface: [0] [1] [2] [3] [4] [5] [6] [7] [8] [9] [10] [11] [12] [13] [14] [15] [16] [17] [18] [19] [20] [21] [22] [23] [24] [25] [26] [27] [28] [29]
- Video Port (VP) Interface: [30]

Display Subsystem Basic Programming Model

- Display Controller Basic Programming Model: [31]
- LCD-Specific Control Registers: [32] [33] [34] [35] [36] [37] [38] [39] [40]
- SDI Configuration: [41] [42] [43] [44] [45] [46]

Table 15-180. Register Call Summary for Register DISPC_POL_FREQ (continued)

Display Subsystem Use Cases and Tips
• SDI Software Settings: [47] [48] [49] [50] [51] [52]
• Programming Flow: [53] [54] [55] [56] [57] [58] [59]
Display Subsystem Register Manual
• Display Subsystem Register Mapping Summary: [60]

Table 15-181. DISPC_DIVISOR

Address Offset	0x070		
Physical address	0x4805 0470	**Instance**	DISC
Description	The register configures the divisors. Shadow register, updated on VFP start period		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								LCD								Reserved								PCD							

Bits	Field Name	Description	Type	Reset
31:24	Reserved	Write 0s for future compatibility. Read returns 0	RW	0x00
23:16	LCD	Display Controller Logic Clock Divisor Value (from 1 to 255) to specify the frequency of the display controller logic clock based on the function clock. The value 0 is invalid.	RW	0x01
15:8	Reserved	Write 0s for future compatibility. Read returns 0	RW	0x00
7:0	PCD	Pixel Clock Divisor Value (from 1 to 255) to specify the frequency of the pixel clock based on the Logic clock which is the functional clock divided by LCD. The values 0 and 1 are invalid.	RW	0x02

Table 15-182. Register Call Summary for Register DISPC_DIVISOR

Display Subsystem Environment
• Video Port (VP) Interface: [0]
Display Subsystem Basic Programming Model
• Display Controller Basic Programming Model: [1]
• LCD-Specific Control Registers: [2] [3] [4] [5]
• TV Set-Specific Control Registers: [6] [7]
• Clock Source/Frequency Change Sequence: [8] [9]
Display Subsystem Use Cases and Tips
• Display Subsystem Clock: [10] [11] [12] [13] [14] [15] [16] [17] [18] [19] [20] [21]
• Programming Flow: [22] [23] [24]
• Configure the DISPC: [25]
• Display Subsystem Use Cases and Tips: [26] [27]
Display Subsystem Register Manual
• Display Subsystem Register Mapping Summary: [28]

Table 15-183. DISPC_GLOBAL_ALPHA

Address Offset	0x0000 0074		
Physical Address	0x4805 0474	**Instance**	DISC
Description	The register defines the global alpha value for the graphics and video 2 pipelines. Shadow register, updated on VFP start period or EVSYNC for each bit field depending on the association of the each pipeline with the LCD or TV output.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								VID2GLOBALALPHA								RESERVED								GFXGLOBALALPHA							

Bits	Field Name	Description	Type	Reset
31:24	RESERVED	Write 0s for future compatibility. Reads return 0	RW	0x00
23:16	VID2GLOBALALPHA	Global alpha value from 0 to 255. 0 corresponds to fully transparent and 255 to fully opaque.	RW	0x00
15:8	RESERVED	Write 0s for future compatibility. Reads return 0	RW	0x00
7:0	GFXGLOBALALPHA	Global alpha value from 0 to 255. 0 corresponds to fully transparent and 255 to fully opaque.	RW	0x00

Table 15-184. Register Call Summary for Register DISPC_GLOBAL_ALPHA

Display Subsystem Basic Programming Model
- LCD-Specific Control Registers: [0] [1]
- TV Set-Specific Control Registers: [2] [3]

Display Subsystem Register Manual
- Display Subsystem Register Mapping Summary: [4]

Table 15-185. DISPC_SIZE_DIG

Address Offset	0x078		
Physical address	0x4805 0478	**Instance**	DISC
Description	The register configures the size of the digital output field (interlace), frame (progressive) (horizontal and vertical). Shadow register, updated on EVSYNC.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								LPP								Reserved								PPL							

Bits	Field Name	Description	Type	Reset
31:27	Reserved	Write 0s for future compatibility. Read returns 0	RW	0x00
26:16	LPP	Lines per panel Encoded value (from 1 to 2048) to specify the number of lines per panel (program to value minus one).	RW	0x000
15:11	Reserved	Write 0s for future compatibility. Read returns 0	RW	0x00
10:0	PPL	Pixels per line Encoded value (from 1 to 2048) to specify the number of pixels contained within each line on the display (program to value minus one)	RW	0x000

Table 15-186. Register Call Summary for Register DISPC_SIZE_DIG

Display Subsystem Functional Description
- Video Pipeline: [0]

Display Subsystem Basic Programming Model
- Display Controller Basic Programming Model: [1]
- TV Set-Specific Control Registers: [2] [3] [4]
- Video Encoder Register Settings: [5] [6]

Display Subsystem Register Manual
- Display Subsystem Register Mapping Summary: [7]

Table 15-187. DISPC_SIZE_LCD

Address Offset	0x07C		
Physical address	0x4805 047C	**Instance**	DISC
Description	The register configures the panel size (horizontal and vertical). Shadow register, updated on VFP start period		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								LPP								Reserved								PPL							

Bits	Field Name	Description	Type	Reset
31:27	Reserved	Write 0s for future compatibility. Read returns 0	RW	0x00
26:16	LPP	Lines per panel Encoded value (from 1 to 2048) to specify the number of lines per panel (program to value minus one).	RW	0x000
15:11	Reserved	Write 0s for future compatibility. Read returns 0	RW	0x00
10:0	PPL	Pixels per line Encoded value (from 1 to 2048) to specify the number of pixels contains within each line on the display (program to value minus one). When running in normal mode (stall mode is bypassed by setting DSS.DISPC_CONTROL[11] STALLMODE =0) the line width must be set to a value multiple of 8 pixels (ex: PPL=0x7)	RW	0x000

Table 15-188. Register Call Summary for Register DISPC_SIZE_LCD

Display Subsystem Environment • Parallel Interface: [0] [1]
Display Subsystem Functional Description • Video Pipeline: [2]
Display Subsystem Basic Programming Model • Display Controller Basic Programming Model: [3] • LCD-Specific Control Registers: [4] [5] [6]
Display Subsystem Use Cases and Tips • Programming Flow: [7] [8] [9] • Configure the DISPC: [10] • Display Subsystem Use Cases and Tips: [11] [12]
Display Subsystem Register Manual • Display Subsystem Register Mapping Summary: [13]

Table 15-189. DISPC_GFX_BAj

Address Offset	0x080 + j * 0x04	**Index**	j = 0 to 1
Physical address	0x4805 0480+ j * 0x04	**Instance**	DISC
Description	The register configures the base address of the graphics buffer displayed in the graphics window (0 & 1 :for ping-pong mechanism with external trigger, based on the field polarity, 0 only used when graphics pipeline on the LCD output and 0 & 1 when on the 24-bit digital output). Shadow register, updated on VFP start period or EVSYNC.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GFXBA																															

Bits	Field Name	Description	Type	Reset
31:0	GFXBA	Graphics base address Base address of the graphics buffer (aligned on pixel size boundary) (in case 1-, 2-, and 4-BPP, byte alignment is required)	RW	0x00000000

Table 15-190. Register Call Summary for Register DISPC_GFX_BAj

Display Subsystem Basic Programming Model
- Display Controller Basic Programming Model: [0]
- Graphics Layer Configuration: [1] [2]
- Rotation/Mirroring Display Subsystem Settings: [3]

Display Subsystem Register Manual
- Display Subsystem Register Mapping Summary: [4]

Table 15-191. DISPC_GFX_POSITION

Address Offset	0x088		
Physical address	0x4805 0488	**Instance**	DISC
Description	The register configures the position of the graphics window. Shadow register, updated on VFP start period or EVSYNC.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								GFXPOSY								Reserved								GFXPOSX							

Bits	Field Name	Description	Type	Reset
31:27	Reserved	Write 0s for future compatibility. Read returns 0	RW	0x00
26:16	GFXPOSY	Y position of the graphics window. Encoded value (from 0 to 2047) to specify the Y position of the graphics window on the screen. The line at the top has the Y-position 0.	RW	0x000
15:11	Reserved	Write 0s for future compatibility. Read returns 0	RW	0x00
10:0	GFXPOSX	X position of the graphics window. Encoded value (from 0 to 2047) to specify the X position of the graphics window on the screen. The first pixel on the left of the screen has the X-position 0.	RW	0x000

Table 15-192. Register Call Summary for Register DISPC_GFX_POSITION

Display Subsystem Basic Programming Model
- Display Controller Basic Programming Model: [0]
- Graphics Layer Configuration: [1] [2] [3]
- Rotation/Mirroring Display Subsystem Settings: [4]

Display Subsystem Register Manual
- Display Subsystem Register Mapping Summary: [5]

Table 15-193. DISPC_GFX_SIZE

Address Offset	0x08C		
Physical address	0x4805 048C	**Instance**	DISC
Description	The register configures the size of the graphics window. Shadow register, updated on VFP start period or EVSYNC.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								GFXSIZEY								Reserved								GFXSIZEX							

Bits	Field Name	Description	Type	Reset
31:27	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0x00
26:16	GFXSIZEY	Number of lines of the graphics window. Encoded value (from 1 to 2048) to specify the number of lines of the graphics window (program to value minus one).	RW	0x000
15:11	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0x00
10:0	GFXSIZEX	Number of pixels of the graphics window. Encoded value (from 1 to 2048) to specify the number of pixels per line of the graphics window (program to value minus one).	RW	0x000

Table 15-194. Register Call Summary for Register DISPC_GFX_SIZE

Display Subsystem Basic Programming Model

- Display Controller Basic Programming Model: [0]
- Graphics Layer Configuration: [1] [2] [3]
- Rotation/Mirroring Display Subsystem Settings: [4]

Display Subsystem Register Manual

- Display Subsystem Register Mapping Summary: [5]

Table 15-195. DISPC_GFX_ATTRIBUTES

Address Offset	0x0A0		
Physical address	0x4805 04A0	**Instance**	DISC
Description	The register configures the graphics attributes. Shadow register, updated on VFP start period or EVSYNC.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																GFXSELFREFRESH	GFXARBITRATION	GFXROTATION		GFXFIFOPRELOAD	GFXENDIANNESS	GFXNIBBLEMODE	GFXCHANNELOUT	GFXBURSTSIZE		GFXREPLICATIONENABLE	GFXFORMAT				GFXENABLE

Bits	Field Name	Description	Type	Reset
31:16	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0x000000
15	GFXSELFREFRESH	Enables the self refresh of the graphics window from its own FIFO only. 0x0: The graphics pipeline accesses the interconnect to fetch data from the system memory	RW	0

Bits	Field Name	Description	Type	Reset
		0x1: The graphics pipeline does not need anymore to fetch data from memory. Only the graphics FIFO is used. It takes effect after the frame has been loaded in the FIFO		
14	GFXARBITRATION	Determines the priority of the graphics pipeline. The graphics pipeline is one of the high priority pipeline. The arbitration wheel gives always the priority first to the high priority pipelines using round-robin between them. When there is only normal priority pipelines sending requests, the round-robin applies between them. 0x0: The graphics pipeline is one of the normal priority pipeline. 0x1: The graphics pipeline is one of the high priority pipeline.	RW	0
13:12	GFXROTATION	Graphics rotation flag (used only in case of RGB24 packed format) 0x0: No rotation 0x1: Rotation by 90 degrees 0x2: Rotation by 180 degrees 0x3: Rotation by 270 degrees	RW	0x0
11	GFXFIFOPRELOAD	Graphics preload value 0x0: H/W prefetches pixels up to the preload value defined in the preload register. 0x1: H/W prefetches pixels up to high threshold value.	RW	0
10	GFXENDIANNESS	Graphics endianness 0x0: Little endian operation is selected. 0x1: Big endian operation is selected.	RW	0
9	GFXNIBBLEMODE	Graphics Nibble Mode (only for 1-, 2- and 4-BPP) 0x0: Nibble mode is disabled 0x1: Nibble mode is enabled	RW	0
8	GFXCHANNELOUT	Graphics Channel Out configuration wr: immediate 0x0: LCD output selected 0x1: 24-bit output selected	RW	0
7:6	GFXBURSTSIZE	Graphics DMA Burst Size 0x0: 4x32bit bursts 0x1: 8x32bit bursts 0x2: 16x32bit bursts 0x3: Reserved	RW	0x0
5	GFXREPLICATION ENABLE	GfxReplicationEnable 0x0: Disable Graphics replication logic 0x1: Enable Graphics replication logic	RW	0
4:1	GFXFORMAT	Graphics format; Other enums: Reserved (0x7, 0xA, 0xB and 0xF) 0x0: BITMAP 1 (CLUT) 0x1: BITMAP 2 (CLUT) 0x2: BITMAP 4 (CLUT) 0x3: BITMAP 8 (CLUT) 0x4: RGB 12 (un-packed in 16-bit container) 0x5: ARGB16 0x6: RGB 16 0x8: RGB 24 (un-packed in 32-bit container) 0x9: RGB 24 (packed in 24-bit container) 0xC: ARGB32 0xD: RGBA32	RW	0x0

Bits	Field Name	Description	Type	Reset
		0xE: RGBx 32 (24-bit RGB aligned on MSB of the 32-bit container)		
0	GFXENABLE	GfxEnable	RW	0
		0x0: Graphics disabled (graphics pipeline inactive and graphics window not present)		
		0x1: Graphics enabled (graphics pipeline active and graphics window present on the screen)		

Table 15-196. Register Call Summary for Register DISPC_GFX_ATTRIBUTES

Display Subsystem Basic Programming Model
- Display Controller Basic Programming Model: [0]
- Graphics Layer Configuration: [1] [2] [3] [4] [5] [6] [7] [8] [9] [10] [11] [12] [13] [14]
- Rotation/Mirroring Display Subsystem Settings: [15]

Display Subsystem Register Manual
- Display Subsystem Register Mapping Summary: [16]
- Display Controller Registers: [17]

Table 15-197. DISPC_GFX_FIFO_THRESHOLD

Address Offset	0x0A4		
Physical address	0x4805 04A4	**Instance**	DISC
Description	The register configures the graphics FIFO. Shadow register, updated on VFP start period or EVSYNC.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								GFXFIFOHIGHTHRESHOLD								Reserved								GFXFIFOLOWTHRESHOLD							

Bits	Field Name	Description	Type	Reset
31:28	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0x00
27:16	GFXFIFOHIGH THRESHOLD	Graphics FIFO High Threshold Number of bytes defining the threshold value.	RW	0x3FF
15:12	Reserved	Write 0s for future compatibility. Read returns 0	RW	0x00
11:0	GFXFIFOLOW THRESHOLD	Graphics FIFO Low Threshold Number of bytes defining the threshold value	RW	0x3C0

Table 15-198. Register Call Summary for Register DISPC_GFX_FIFO_THRESHOLD

Display Subsystem Basic Programming Model
- Display Controller Basic Programming Model: [0]
- Graphics Layer Configuration: [1] [2] [3] [4] [5] [6]
- Rotation/Mirroring Display Subsystem Settings: [7] [8] [9]

Display Subsystem Use Cases and Tips
- FIFO Thresholds: [10] [11]

Display Subsystem Register Manual
- Display Subsystem Register Mapping Summary: [12]

Table 15-199. DISPC_GFX_FIFO_SIZE_STATUS

Address Offset	0x0A8		
Physical address	0x4805 04A8	**Instance**	DISC
Description	This register defines the graphics FIFO size.		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																					GFXFIFOSIZE										

Bits	Field Name	Description	Type	Reset
31:11	Reserved	Write 0s for future compatibility. Read returns 0	R	0x000000
10:0	GFXFIFOSIZE	Graphics FIFO Size Number of bytes defining the FIFO value.	R	0x400

Table 15-200. Register Call Summary for Register DISPC_GFX_FIFO_SIZE_STATUS

Display Subsystem Register Manual

- Display Subsystem Register Mapping Summary: [0]

Table 15-201. DISPC_GFX_ROW_INC

Address Offset	0x0AC		
Physical address	0x4805 04AC	**Instance**	DISC
Description	The register configures the number of bytes to increment at the end of the row. Shadow register, updated on VFP start period or EVSYNC.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GFXROWINC																															

Bits	Field Name	Description	Type	Reset
31:0	GFXROWINC	Number of bytes to increment at the end of the row Encoded signed value (from -2^{31}- 1 to 2^{31}) to specify the number of bytes to increment at the end of the row in the graphics buffer. The value 0 is invalid. The value 1 means next pixel. The value 1+n*BPP means increment of n pixels. The value 1- (n+1)*BPP means decrement of n pixels.	RW	0x00000001

Table 15-202. Register Call Summary for Register DISPC_GFX_ROW_INC

Display Subsystem Basic Programming Model

- Display Controller Basic Programming Model: [0]
- Graphics Layer Configuration: [1] [2] [3]
- Rotation/Mirroring Display Subsystem Settings: [4]

Display Subsystem Register Manual

- Display Subsystem Register Mapping Summary: [5]

Table 15-203. DISPC_GFX_PIXEL_INC

Address Offset	0x0B0		
Physical address	0x4805 04B0	**Instance**	DISC
Description	The register configures the number of bytes to increment between two pixels. Shadow register, updated on VFP start period or EVSYNC.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																GFXPIXELINC															

Bits	Field Name	Description	Type	Reset
31:16	Reserved	Write 0s for future compatibility. Read returns 0	RW	0x0000
15:0	GFXPIXELINC	Number of bytes to increment between two pixels Encoded signed value (from -2^{15}- 1 to 2^{15}) to specify the number of bytes between two pixels in the graphics buffer. The value 0 is invalid. The value 1 means next pixel. The value 1+n*BPP means increment of n pixels. The value 1- (n+1)*BPP means decrement of n pixels.	RW	0x0001

Table 15-204. Register Call Summary for Register DISPC_GFX_PIXEL_INC

Display Subsystem Basic Programming Model
- Display Controller Basic Programming Model: [0]
- Graphics Layer Configuration: [1]
- Rotation/Mirroring Display Subsystem Settings: [2]

Display Subsystem Register Manual
- Display Subsystem Register Mapping Summary: [3]

Table 15-205. DISPC_GFX_WINDOW_SKIP

Address Offset	0x0B4		
Physical address	0x4805 04B4	**Instance**	DISC
Description	The register configures the number of bytes to skip during video window display. Shadow register, updated on VFP start period or EVSYNC.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GFXWINDOWSKIP																															

Bits	Field Name	Description	Type	Reset
31:0	GFXWINDOWSKIP	Number of bytes to skip during video window #1. Encoded signed value (from -2^{31}-1 to 2^{31}) to specify the number of bytes to skip in the graphics buffer when video window #1 is displayed on top of the graphics and no transparency color is enabled.	RW	0x00000000

Table 15-206. Register Call Summary for Register DISPC_GFX_WINDOW_SKIP

Display Subsystem Functional Description
- Overlay Support: [0]

Display Subsystem Basic Programming Model
- Display Controller Basic Programming Model: [1]
- Graphics Layer Configuration: [2] [3] [4] [5]

Display Subsystem Register Manual
- Display Subsystem Register Mapping Summary: [6]

Table 15-207. DISPC_GFX_TABLE_BA

Address Offset	0x0B8		
Physical address	0x4805 04B8	**Instance**	DISC
Description	The register configures the base address of the palette buffer or the gamma table buffer. Shadow register, updated on VFP start period or EVSYNC.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GFXTABLEBA																															

Bits	Field Name	Description	Type	Reset
31:0	GFXTABLEBA	Base address of the palette/gamma table buffer (24-bit entries in 32-bit containers, aligned on 32-bit boundary).	RW	0x00000000

Table 15-208. Register Call Summary for Register DISPC_GFX_TABLE_BA

Display Subsystem Basic Programming Model
- Display Controller Basic Programming Model: [0]
- Graphics Layer Configuration: [1] [2]

Display Subsystem Register Manual
- Display Subsystem Register Mapping Summary: [3]

Table 15-209. DISPC_VIDn_BAj

Address Offset	0x0BC + ((n–1)* 0x90) + (j* 0x04)	**Index**	n = 1 for VID1 or 2 for VID2 j = 0 to 1
Physical address	0x4805 04BC+ ((n–1)* 0x90) + (j* 0x04)	**Instance**	DISC
Description	The register configures the base address of the video buffer for video window #n(#j for ping-pong mechanism with external trigger, based on the field polarity: 0 for even field and 1 for odd field). Shadow register, updated on VFP start period or EVSYNC.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
VIDBA																															

Bits	Field Name	Description	Type	Reset
31:0	VIDBA	Video base address Base address of the video buffer (aligned on pixel size boundary)	RW	0x00000000

Table 15-210. Register Call Summary for Register DISPC_VIDn_BAj

Display Subsystem Basic Programming Model
- Display Controller Basic Programming Model: [0]
- Video Layer Configuration: [1]
- Rotation/Mirroring Display Subsystem Settings: [2] [3] [4]

Display Subsystem Use Cases and Tips
- Scaling Settings: [5]

Display Subsystem Register Manual
- Display Subsystem Register Mapping Summary: [6] [7]

Table 15-211. DISPC_VIDn_POSITION

Address Offset	0x0C4 + ((–1) *0x90)	**Index**	n = 1 for VID1 or 2 for VID2
Physical address	0x4805 04C4 + ((–1)* 0x90)	**Instance**	DISC
Description	The register configures the position of video window #n. Shadow register, updated on VFP start period or EVSYNC.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								VIDPOSY								Reserved								VIDPOSX							

Bits	Field Name	Description	Type	Reset
31:27	Reserved	Write 0s for future compatibility. Read returns 0	RW	0x00
26:16	VIDPOSY	Y position of video window #n Encoded value (from 0 to 2047) to specify the Y position of video window #n. The line at the top has the Y-position 0.	RW	0x000
15:11	Reserved	Write 0s for future compatibility. Read returns 0	RW	0x00
10:0	VIDPOSX	X position of video window #n Encoded value (from 0 to 2047) to specify the X position of video window #n. The first pixel on the left of the display screen has the X-position 0.	RW	0x000

Table 15-212. Register Call Summary for Register DISPC_VIDn_POSITION

Display Subsystem Basic Programming Model

- Display Controller Basic Programming Model: [0]
- Video Layer Configuration: [1] [2] [3]
- Rotation/Mirroring Display Subsystem Settings: [4]

Display Subsystem Register Manual

- Display Subsystem Register Mapping Summary: [5] [6]

Table 15-213. DISPC_VIDn_SIZE

Address Offset	0x0C8+((–1)* 0x90)	**Index**	n = 1 for VID1 or 2 for VID2
Physical address	0x4805 04C8+((–1)* 0x90)	**Instance**	DISC
Description	The register configures the size of video window #n. Shadow register, updated on VFP start period or EVSYNC.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								VIDSIZEY								Reserved								VIDSIZEX							

Bits	Field Name	Description	Type	Reset
31:27	Reserved	Write 0s for future compatibility. Read returns 0	RW	0x00
26:16	VIDSIZEY	Number of lines of video #n Encoded value (from 1 to 2048) to specify the number of lines of video window #n (program to value minus one).	RW	0x000
15:11	Reserved	Write 0s for future compatibility. Read returns 0	RW	0x00
10:0	VIDSIZEX	Number of pixels of video window #n Encoded value (from 1 to 2048) to specify the number of pixels of video window #n (program to value minus one).	RW	0x000

Table 15-214. Register Call Summary for Register DISPC_VIDn_SIZE

Display Subsystem Basic Programming Model

- Display Controller Basic Programming Model: [0]
- Video Layer Configuration: [1] [2] [3]
- Rotation/Mirroring Display Subsystem Settings: [4]

Display Subsystem Register Manual

- Display Subsystem Register Mapping Summary: [5] [6]

Table 15-215. DISPC_VIDn_ATTRIBUTES

Address Offset	0x0CC+ ((–1)* 0x90)	**Index**	n = 1 for VID1 or 2 for VID2
Physical address	0x4805 04CC+ ((–1)* 0x90)	**Instance**	DISC
Description	The register configures the attributes of video window #n such as format, resizeenable, shadow register, updated on VFP start period, or EVSYNC.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								VIDSELFREFRESH	VIDARBITRATION	VIDLINEBUFFERSPLIT	VIDVERTICALTAPS	VIDDMAOPTIMIZATION	VIDFIFOPRELOAD	VIDROWREPEATENABLE	VIDENDIANNESS	VIDCHANNELOUT	VIDBURSTSIZE		VIDROTATION		VIDFULLRANGE	VIDREPLICATIONENABLE	VIDCOLORCONVENABLE	VIDVRESIZECONF	VIDHRESIZECONF	VIDRESIZEENABLE		VIDFORMAT				VIDENABLE

Bits	Field Name	Description	Type	Reset
31: 25	RESERVED	Write 0s for future compatibility. Read returns 0.	RW	0x0000
24	VIDSELFREFRESH	Enables the self refresh of the video window from its own FIFO only. 0x0: The video pipeline accesses the interconnect to fetch data from the system memory 0x1: The video pipeline does not need anymore to fetch data from memory. Only the video FIFO is used. It takes effect after the frame has been loaded in the FIFO	RW	0
23	VIDARBITRATION	Determines the priority of the video pipeline. The video pipeline is one of the high priority pipeline. The arbitration wheel gives always the priority first to the high priority pipelines using round-robin between them. When there is only normal priority pipelines sending requests, the round-robin applies between them. 0x0: The video pipeline is one of the normal priority pipeline. 0x1: The video pipeline is one of the high priority pipeline.	RW	0
22	VIDLINEBUFFER SPLIT	Video vertical line buffer split 0x0: Vertical line buffers are not split. 0x1: Vertical line buffers are split into two.	RW	0
21	VIDVERTICALTAPS	Video vertical resize tap number 0x0: Three taps are used for the vertical filtering logic. The other two taps are not used. 0x1: Five taps are used for the vertical filtering logic.	RW	0
20	VIDDMAOPTI MIZATION	Video optimization in case of 0x0: The DMA engine fetches one pixel for each 32-bit OCP request (RGB16 and YUV422) while doing 90- and 270-degree rotation (accessing on-chip memory and off-chip memory).	RW	0

Bits	Field Name	Description	Type	Reset
		0x1: The DMA engine fetches two pixels for each 32-bit OCP request (RGB16 and YUV422) while doing 90- and 270-degree rotation (accessing on-chip memory and off-chip memory).		
19	VIDFIFOPRELOAD	Video preload value	RW	0
		0x0: H/W prefetches pixels up to the preload value defined in the preload register.		
		0x1: H/W prefetches pixels up to the high threshold value.		
18	VIDROWREPEAT ENABLE	Video Row Repeat (YUV case only when rotating 90 or 270-degree)	RW	0
		0x0: Row of VIDn won't be read twice.		
		0x1: The Row data are fetched twice to extract both the Y components		
17	VIDENDIANNESS	Video Endianness	RW	0
		0x0: Little endian operation is selected.		
		0x1: Big endian operation is selected.		
16	VIDCHANNELOUT	Video Channel Out configuration wr: Immediate	RW	0
		0x0: LCD output selected		
		0x1: 24 bit output selected		
15:14	VIDBURSTSIZE	Video DMA Burst Size	RW	0x0
		0x0: 4x32bit bursts		
		0x1: 8x32bit bursts		
		0x2: 16x32bit bursts		
		0x3: Reserved		
13:12	VIDROTATION	Video Rotation Flag	RW	0x0
		0x0: No rotation or VidFormat is RGB		
		0x1: Rotation by 90 degrees		
		0x2: Rotation by 180 degrees		
		0x3: Rotation by 270 degrees		
11	VIDFULLRANGE	VidFullRange	RW	0
		0x0: Limited range selected: 16 subtracted from Y before color space conversion		
		0x1: Full range selected: Y is not modified before the color space conversion		
10	VIDREPLICATION ENABLE	VidReplicationEnable	RW	0
		0x0: Disable Video replication logic		
		0x1: Enable Video replication logic		
9	VIDCOLORCONV ENABLE	VidColorConvEnable	RW	0
		0x0: Disable Color Space Conversion CbYCr to RGB		
		0x1: Enable Color Space Conversion CbYCr to RGB		
8	VIDVRESIZECONF	Video Vertical Resize Configuration	RW	0
		0x0: Up-sampling selected		
		0x1: Down-sampling selected		
7	VIDHRESIZECONF	Video Horizontal Resize Configuration	RW	0
		0x0: Up-sampling selected		
		0x1: Down-sampling selected		
6:5	VIDRESIZEENABLE	Video Resize Enable	RW	0x0
		0x0: Disable the resize processing		
		0x1: Enable the horizontal resize processing		

Bits	Field Name	Description	Type	Reset
		0x2: Enable the vertical resize processing		
		0x3: Enable both horizontal and vertical resize processing		
4:1	VIDFORMAT (Video 1 channel)	Video1 channel Format; Other enums: Reserved (all other values between 0x0 and 0x3, 0x5, 0x7, and between 0xC and 0xF)	RW	0x0
		0x4: RGB12 (16-bit container)		
		0x6: RGB 16		
		0x8: RGB 24 (unpacked in 32-bit container)		
		0x9: RGB 24 (packed in24-bit container)		
		0xA: YUV2 4:2:2 co-sited		
		0xB: UYVY 4:2:2 co-sited		
	VIDFORMAT (Video 2 channel)	Video2 channel Format; Other enums: Reserved (all other values: 0x0 and 0x3, 0x7, and 0xF)	RW	0x0
		0x4: RGB 12 (16-bit container)		
		0x5: ARGB 16		
		0x6: RGB 16		
		0x8: RGB 24 (un-packed in 32-bit container)		
		0x9: RGB 24 (packed in 24-bit container)		
		0xA: YUV2 4:2:2 co-sited		
		0xB: UYVY 4:2:2 co-sited		
		0xC: ARGB 32		
		0xD: RGBA 32		
		0xE: RGBx 32 (24-bit RGB aligned on MSB of the 32-bit container)		
0	VIDENABLE	VidEnable	RW	0
		0x0: Video disabled (video pipeline inactive and window not present)		
		0x1: Video enabled (video pipeline active and window present on the screen)		

Table 15-216. Register Call Summary for Register DISPC_VIDn_ATTRIBUTES

Display Subsystem Functional Description
- Overlay Support: [0]

Display Subsystem Basic Programming Model
- Display Controller Basic Programming Model: [1]
- Video Layer Configuration: [2] [3] [4] [5] [6] [7] [8] [9] [10] [11] [12] [13] [14] [15] [16] [17] [18] [19] [20]
- Rotation/Mirroring Display Subsystem Settings: [21] [22] [23] [24] [25] [26] [27] [28] [29] [30] [31] [32] [33] [34] [35] [36] [37] [38] [39]

Display Subsystem Use Cases and Tips
- Scaling Settings: [40] [41] [42]
- Programming Flow: [43]

Display Subsystem Register Manual
- Display Subsystem Register Mapping Summary: [44] [45]

Table 15-217. DISPC_VIDn_FIFO_THRESHOLD

Address Offset	0x0D0+ ((–1)* 0x90)	**Index**	n = 1 for VID1 or 2 for VID2
Physical address	0x4805 04D0+ ((–1)* 0x90)	**Instance**	DISC
Description	The register configures the video FIFO associated with video pipeline #n. Shadow register, updated on VFP start period or EVSYNC.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								VIDFIFOHIGHTHRESHOLD								Reserved								VIDFIFOLOWTHRESHOLD							

Bits	Field Name	Description	Type	Reset
31:28	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0x00
27:16	VIDFIFOHIGH THRESHOLD	Video FIFO high threshold Number of bytes defining the threshold value	RW	0x3FF
15:12	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0x00
11:0	VIDFIFOLOW THRESHOLD	Video FIFO low threshold Number of bytes defining the threshold value	RW	0x3C0

Table 15-218. Register Call Summary for Register DISPC_VIDn_FIFO_THRESHOLD

Display Subsystem Basic Programming Model

- Display Controller Basic Programming Model: [0]
- Video Layer Configuration: [1] [2] [3] [4] [5] [6]
- Rotation/Mirroring Display Subsystem Settings: [7]

Display Subsystem Register Manual

- Display Subsystem Register Mapping Summary: [8] [9]

Table 15-219. DISPC_VIDn_FIFO_SIZE_STATUS

Address Offset	0x0D4+ ((–1)* 0x90)	**Index**	n = 1 for VID1 or 2 for VID2
Physical address	0x4805 04D4+((–1)* 0x90)	**Instance**	DISC
Description	The register defines the video FIFO size for video pipeline #n.		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																								VIDFIFOSIZE							

Bits	Field Name	Description	Type	Reset
31:11	Reserved	Write 0s for future compatibility. Read returns 0.	R	0x000000
10:0	VIDFIFOSIZE	Video FIFO Size Number of bytes defining the FIFO value	R	0x400

Table 15-220. Register Call Summary for Register DISPC_VIDn_FIFO_SIZE_STATUS

Display Subsystem Register Manual

- Display Subsystem Register Mapping Summary: [0] [1]

Table 15-221. DISPC_VIDn_ROW_INC

Address Offset	0x0D8+ ((–1)* 0x90)	**Index**	n = 1 for VID1 or 2 for VID2
Physical address	0x4805 04D8+ ((–1)* 0x90)	**Instance**	DISC
Description	The register configures the number of bytes to increment at the end of the row for the buffer associated with video window #n. Shadow register, updated on VFP start period or EVSYNC.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
VIDROWINC																															

Bits	Field Name	Description	Type	Reset
31:0	VIDROWINC	Number of bytes to increment at the end of the row Encoded signed value (from -2^{31}- 1 to 2^{31}) to specify the number of bytes to increment at the end of the row in the video buffer. The value 0 is invalid. The value 1 means next pixel. The value 1+n*BPP means increment of n pixels. The value 1- (n+1)*BPP means decrement of n pixels.	RW	0x00000001

Table 15-222. Register Call Summary for Register DISPC_VIDn_ROW_INC

Display Subsystem Basic Programming Model

- Display Controller Basic Programming Model: [0]
- Video Layer Configuration: [1] [2] [3]
- Rotation/Mirroring Display Subsystem Settings: [4] [5] [6]

Display Subsystem Register Manual

- Display Subsystem Register Mapping Summary: [7] [8]

Table 15-223. DISPC_VIDn_PIXEL_INC

Address Offset	0x0DC+ ((–1)* 0x90)	**Index**	n = 1 for VID1 or 2 for VID2
Physical address	0x4805 04DC+ ((–1)* 0x90)	**Instance**	DISC
Description	The register configures the number of bytes to increment between two pixels for the buffer associated with video window #n. Shadow register, updated on VFP start period or EVSYNC.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																VIDPIXELINC															

Bits	Field Name	Description	Type	Reset
31:16	Reserved	Write 0s for future compatibility. Read returns 0	RW	0x0000
15:0	VIDPIXELINC	Number of bytes to increment at the end of the row Encoded signed value (from -2^{15}- 1 to 2^{15}) to specify the number of bytes between two pixels in the video buffer. The value 0 is invalid. The value 1 means next pixel. The value 1+n*BPP means increment of n pixels. The value 1- (n+1)*BPP means decrement of n pixels	RW	0x0001

Table 15-224. Register Call Summary for Register DISPC_VIDn_PIXEL_INC

Display Subsystem Basic Programming Model

- Display Controller Basic Programming Model: [0]
- Video Layer Configuration: [1]
- Rotation/Mirroring Display Subsystem Settings: [2] [3] [4]

Display Subsystem Register Manual

- Display Subsystem Register Mapping Summary: [5] [6]

Table 15-225. DISPC_VIDn_FIR

Address Offset	0x0E0+ ((–1)* 0x90)	**Index**	n = 1 for VID1 or 2 for VID2
Physical address	0x4805 04E0+((–1)* 0x90)	**Instance**	DISC
Description	The register configures the resize factors for horizontal and vertical up-/down-sampling of video window #n. Shadow register, updated on VFP start period or EVSYNC.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								VIDFIRVINC								Reserved								VIDFIRHINC							

Bits	Field Name	Description	Type	Reset
31:29	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0x0
28:16	VIDFIRVINC	Vertical increment of the up-/down-sampling filter Encoded value (from 1 to 4096). The value 0 is invalid. Values greater than 4096 are invalid.	RW	0x0000
15:13	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0x0
12:0	VIDFIRHINC	Horizontal increment of the up-/down-sampling filter Encoded value (from 1 to 4096). The value 0 is invalid. Values greater than 4096 are invalid.	RW	0x0000

Table 15-226. Register Call Summary for Register DISPC_VIDn_FIR

Display Subsystem Basic Programming Model
- Display Controller Basic Programming Model: [0]
- Video Layer Configuration: [1] [2] [3]

Display Subsystem Use Cases and Tips
- Scaling Settings: [4] [5] [6] [7] [8] [9] [10]
- Programming Flow: [11]

Display Subsystem Register Manual
- Display Subsystem Register Mapping Summary: [12] [13]

Table 15-227. DISPC_VIDn_PICTURE_SIZE

Address Offset	0x0E4+ ((–1)* 0x90)	**Index**	n = 1 for VID1 or 2 for VID2
Physical address	0x4805 04E4+ ((–1)* 0x90)	**Instance**	DISC
Description	The register configures the size of the video picture associated with video layer #nbefore up-/down-scaling. Shadow register, updated on VFP start period or EVSYNC.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								VIDORGSIZEY								Reserved								VIDORGSIZEX							

Bits	Field Name	Description	Type	Reset
31:27	Reserved	Write 0s for future compatibility. Read returns 0	RW	0x00
26:16	VIDORGSIZEY	Number of lines of the video picture Encoded value (from 1 to 2048) to specify the number of lines of the video picture in memory (program to value minus one).	RW	0x000
15:11	Reserved	Write 0s for future compatibility. Read returns 0	RW	0x00

Bits	Field Name	Description	Type	Reset
10:0	VIDORGSIZEX	Number of pixels of the video picture Encoded value (from 1 to 2048) to specify the number of pixels of the video picture in memory (program to value minus one). The size is limited to the size of the line buffer of the vertical sampling block in case the video picture is processed by the vertical filtering unit.(1)	RW	0x000

(1) For 5-tap RGB16 and YUV422 picture formats, the width of the input picture must be a multiple of 2 pixels and more than 5 pixels. This leads to the following register configuration:
- DISPC_VIDn_ATTRIBUTES[21] VIDVERTICALTAPS is set to 1.
- DISPC_VIDn_PICTURE_SIZE[10:0] VIDORGSIZEX must be even and more than 4.

Table 15-228. Register Call Summary for Register DISPC_VIDn_PICTURE_SIZE

Display Subsystem Basic Programming Model
- Display Controller Basic Programming Model: [0]
- Video Layer Configuration: [1] [2] [3] [4]

Display Subsystem Register Manual
- Display Subsystem Register Mapping Summary: [5] [6]

Table 15-229. DISPC_VIDn_ACCUI

Address Offset	0x0E8+ ((–1)* 0x90)+ (l*0x04)	**Index**	n = 1 for VID1 or 2 for VID2 l = 0 to 1
Physical address	0x4805 04E8 + ((n-1)*0x90)+ (l*0x04)	**Instance**	DISC
Description	The register configures the resize accumulator init values for horizontal and vertical up-/down-sampling of video window #n (#l for ping-pong mechanism with external trigger, based on the field polarity) Shadow register, updated on VFP start period or EVSYNC.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								VIDVERTICALACCU								Reserved								VIDHORIZONTALACCU							

Bits	Field Name	Description	Type	Reset
31:26	Reserved	Write 0s for future compatibility. Read returns 0	RW	0x00
25:16	VIDVERTICAL ACCU	Vertical initialization accu value. Encoded value (from 0 to 1023).	RW	0x000
15:10	Reserved	Write 0s for future compatibility. Read returns 0	RW	0x00
9:0	VIDHORIZONTAL ACCU	Horizontal initialization accu value. Encoded value (from 0 to 1023).	RW	0x000

Table 15-230. Register Call Summary for Register DISPC_VIDn_ACCUI

Display Subsystem Basic Programming Model
- Display Controller Basic Programming Model: [0]
- Video Layer Configuration: [1] [2]

Display Subsystem Use Cases and Tips
- Scaling Settings: [3] [4] [5]

Display Subsystem Register Manual
- Display Subsystem Register Mapping Summary: [6] [7]

Table 15-231. DISPC_VIDn_FIR_COEF_Hi

Address Offset	0x0F0+ ((–1)* 0x90) + (i* 0x08)	**Index**	n = 1 for VID1 or 2 for VID2 i = 0 to 7
Physical address	0x4805 04F0+ ((n-1)*0x90) + (i*0x08)	**Instance**	DISC
Description	The bank of registers configure the up-/down-scaling coefficients for the vertical and horizontal resize of the video picture associated with video window #n for the phases from 0 to 7. Shadow register, updated on VFP start period or EVSYNC.		
Type	RW		

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10 9 8	7 6 5 4 3 2 1 0
VIDFIRHC3	VIDFIRHC2	VIDFIRHC1	VIDFIRHC0

Bits	Field Name	Description	Type	Reset
31:24	VIDFIRHC3	Signed coefficient C3 for the horizontal up-/down-scaling with the phase n	RW	0x00
23:16	VIDFIRHC2	Unsigned coefficient C2 for the horizontal up-/down-scaling with the phase n	RW	0x00
15:8	VIDFIRHC1	Signed coefficient C1 for the horizontal up-/down-scaling with the phase n	RW	0x00
7:0	VIDFIRHC0	Signed coefficient C0 for the horizontal up-/down-scaling with the phase n	RW	0x00

Table 15-232. Register Call Summary for Register DISPC_VIDn_FIR_COEF_Hi

Display Subsystem Basic Programming Model
- Display Controller Basic Programming Model: [0]
- Video Layer Configuration: [1] [2] [3] [4]

Display Subsystem Use Cases and Tips
- Scaling Settings: [5] [6] [7] [8] [9] [10] [11] [12]

Display Subsystem Register Manual
- Display Subsystem Register Mapping Summary: [13] [14]

Table 15-233. DISPC_VIDn_FIR_COEF_HVi

Address Offset	0x0F4+ ((–1)* 0x90) + (i* 0x08)	**Index**	n = 1 for VID1 or 2 for VID2 i = 0 to 7
Physical address	0x4805 04F4+ ((–1)* 0x90) + (i* 0x08)	**Instance**	DISC
Description	The bank of registers configure the down/up-/down-scaling coefficients for the vertical and horizontal resize of the video picture associated with video window #n for the phases from 0 to 7. Shadow register, updated on VFP start period or EVSYNC.		
Type	RW		

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10 9 8	7 6 5 4 3 2 1 0
VIDFIRVC2	VIDFIRVC1	VIDFIRVC0	VIDFIRHC4

Bits	Field Name	Description	Type	Reset
31:24	VIDFIRVC2	Signed coefficient C2 for the vertical up-/down-scaling with the phase n	RW	0x00
23:16	VIDFIRVC1	Unsigned coefficient C1 for the vertical up-/down-scaling with the phase n	RW	0x00
15:8	VIDFIRVC0	Signed coefficient C0 for the vertical up-/down-scaling with the phase n	RW	0x00
7:0	VIDFIRHC4	Signed coefficient C4 for the horizontal up-/down-scaling with the phase n	RW	0x00

Table 15-234. Register Call Summary for Register DISPC_VIDn_FIR_COEF_HVi

Display Subsystem Basic Programming Model
- Display Controller Basic Programming Model: [0]
- Video Layer Configuration: [1] [2] [3] [4] [5] [6] [7]

Display Subsystem Use Cases and Tips
- Scaling Settings: [8] [9] [10] [11] [12] [13] [14] [15] [16] [17] [18] [19] [20]

Table 15-234. Register Call Summary for Register DISPC_VIDn_FIR_COEF_HVi (continued)

Display Subsystem Register Manual

- Display Subsystem Register Mapping Summary: [21] [22]

Table 15-235. DISPC_VIDn_CONV_COEF0

Address Offset	0x130+((–1)* 0x90)	**Index**	n = 1 for VID1 or 2 for VID2
Physical address	0x4805 0530+((–1)* 0x90)	**Instance**	DISC
Description	The register configures the color space conversion matrix coefficients for video pipeline #n. Shadow register, updated on VFP start period or EVSYNC.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								RCR								Reserved								RY							

Bits	Field Name	Description	Type	Reset
31:27	Reserved	Write 0s for future compatibility. Read returns 0	RW	0x00
26:16	RCR	RCr Coefficient Encoded signed value (from -1024 to 1023).	RW	0x000
15:11	Reserved	Write 0s for future compatibility. Read returns 0	RW	0x00
10:0	RY	RY Coefficient Encoded signed value (from -1024 to 1023).	RW	0x000

Table 15-236. Register Call Summary for Register DISPC_VIDn_CONV_COEF0

Display Subsystem Basic Programming Model

- Rotation/Mirroring Display Subsystem Settings: [0] [1]

Display Subsystem Register Manual

- Display Subsystem Register Mapping Summary: [2] [3]

Table 15-237. DISPC_VIDn_CONV_COEF1

Address Offset	0x134+((–1)* 0x90)	**Index**	n = 1 for VID1 or 2 for VID2
Physical address	0x4805 0534+ ((–1)* 0x90)	**Instance**	DISC
Description	The register configures the color space conversion matrix coefficients for video pipeline #n. Shadow register, updated on VFP start period or EVSYNC.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								GY								Reserved								RCB							

Bits	Field Name	Description	Type	Reset
31:27	Reserved	Write 0s for future compatibility. Read returns 0	RW	0x00
26:16	GY	GY Coefficient Encoded signed value (from -1024 to 1023).	RW	0x000
15:11	Reserved	Write 0s for future compatibility. Read returns 0	RW	0x00
10:0	RCB	RCb Coefficient Encoded signed value (from -1024 to 1023).	RW	0x000

Table 15-238. Register Call Summary for Register DISPC_VIDn_CONV_COEF1

Display Subsystem Basic Programming Model

- Rotation/Mirroring Display Subsystem Settings: [0]

Display Subsystem Register Manual

- Display Subsystem Register Mapping Summary: [1] [2]

Table 15-239. DISPC_VIDn_CONV_COEF2

Address Offset	0x138+ ((–1)* 0x90)	**Index**	n = 1 for VID1 or 2 for VID2
Physical address	0x4805 0538+ ((–1)* 0x90)	**Instance**	DISC
Description	The register configures the color space conversion matrix coefficients for video pipeline #n. Shadow register, updated on VFP start period or EVSYNC.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								GCB								Reserved								GCR							

Bits	Field Name	Description	Type	Reset
31:27	Reserved	Write 0s for future compatibility. Read returns 0	RW	0x00
26:16	GCB	GCb Coefficient Encoded signed value (from -1024 to 1023).	RW	0x000
15:11	Reserved	Write 0s for future compatibility. Read returns 0	RW	0x00
10:0	GCR	GCr Coefficient Encoded signed value (from -1024 to 1023).	RW	0x000

Table 15-240. Register Call Summary for Register DISPC_VIDn_CONV_COEF2

Display Subsystem Basic Programming Model

- Rotation/Mirroring Display Subsystem Settings: [0]

Display Subsystem Register Manual

- Display Subsystem Register Mapping Summary: [1] [2]

Table 15-241. DISPC_VIDn_CONV_COEF3

Address Offset	0x13C+ ((–1)* 0x90)	**Index**	n = 1 for VID1 or 2 for VID2
Physical address	0x4805 053C+ ((–1)* 0x90)	**Instance**	DISC
Description	The register configures the color space conversion matrix coefficients for video pipeline #n. Shadow register, updated on VFP start period or EVSYNC.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								BCR								Reserved								BY							

Bits	Field Name	Description	Type	Reset
31:27	Reserved	Write 0s for future compatibility. Read returns 0	RW	0x00
26:16	BCR	BCr coefficient Encoded signed value (from -1024 to 1023).	RW	0x000
15:11	Reserved	Write 0s for future compatibility. Read returns 0	RW	0x00
10:0	BY	BY coefficient Encoded signed value (from -1024 to 1023).	RW	0x000

Table 15-242. Register Call Summary for Register DISPC_VIDn_CONV_COEF3

Display Subsystem Basic Programming Model
- Rotation/Mirroring Display Subsystem Settings: [0]

Display Subsystem Register Manual
- Display Subsystem Register Mapping Summary: [1] [2]

Table 15-243. DISPC_VIDn_CONV_COEF4

Address Offset	0x140+ ((–1)* 0x90)	**Index**	n = 1 for VID1 or 2 for VID2
Physical address	0x4805 0540+ ((–1)* 0x90)	**Instance**	DISPC
Description	The register configures the color space conversion matrix coefficients for video pipeline #n. Shadow register, updated on VFP start period or EVSYNC.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																					BCB										

Bits	Field Name	Description	Type	Reset
31:11	Reserved	Write 0s for future compatibility. Read returns 0	RW	0x000000
10:0	BCB	BCb Coefficient Encoded signed value (from -1024 to 1023).	RW	0x000

Table 15-244. Register Call Summary for Register DISPC_VIDn_CONV_COEF4

Display Subsystem Basic Programming Model
- Rotation/Mirroring Display Subsystem Settings: [0] [1]

Display Subsystem Register Manual
- Display Subsystem Register Mapping Summary: [2] [3]

Table 15-245. DISPC_DATA_CYCLEk

Address Offset	0x1D4 + (k* 0x04)	**Index**	k = 0 to 2
Physical address	0x4805 05D4+ (k * 0x04)	**Instance**	DISPC
Description	The control register configures the output data format for ith (1st, 2nd or 3rd) cycle. Shadow register, updated on VFP start period.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved				BITALIGNMENTPIXEL2				Reserved			NBBITSPIXEL2					Reserved				BITALIGNMENTPIXEL1				Reserved			NBBITSPIXEL1				

Bits	Field Name	Description	Type	Reset
31:28	Reserved	Write 0s for future compatibility. Read returns 0	RW	0x0
27:24	BITALIGNMENT PIXEL2	Bit alignment Alignment of the bits from pixel#2 on the output interface	RW	0x0
23:21	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0x0
20:16	NBBITSPIXEL2	Number of bits Number of bits from the pixel #2 (value from 0 to 16 bits). The values from 17 to 31 are invalid.	RW	0x00

Bits	Field Name	Description	Type	Reset
15:12	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0x0
11:8	BITALIGNMENT PIXEL1	Bit alignment Alignment of the bits from pixel#1 on the output interface	RW	0x0
7:5	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0x0
4:0	NBBITSPIXEL1	Number of bits Number of bits from the pixel #1 (value from 0 to 16 bits). The values from 17 to 31 are invalid.	RW	0x00

Table 15-246. Register Call Summary for Register DISPC_DATA_CYCLEk

Display Subsystem Functional Description
- Active/Passive Matrix Display Data Path: [0] [1] [2]

Display Subsystem Basic Programming Model
- Display Controller Basic Programming Model: [3]
- LCD-Specific Control Registers: [4] [5] [6] [7]

Display Subsystem Register Manual
- Display Subsystem Register Mapping Summary: [8]

Table 15-247. DISPC_VIDn_FIR_COEF_Vi

Address Offset	0x1E0+ ((–1)* 0x20) + (i* 0x04)	**Index**	n = 1 for VID1 or 2 for VID2 i = 0 to 7
Physical address	0x4805 05E0+ ((n-1)*0x20) + (i* 0x04)	**Instance**	DISC
Description	This bank of registers configures the down/up/down-scaling coefficients for the vertical resize of the video picture associated with video window #n for phases 0 to 7. Shadow register, updated on VFP start period or EVSYNC.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																VIDFIRVC22								VIDFIRVC00							

Bits	Field Name	Description	Type	Reset
31:16	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0x0000
15:8	VIDFIRVC22	Signed coefficient C22 for vertical up/down-scaling with phase n	RW	0x00
7:0	VIDFIRVC00	Signed coefficient C00 for vertical up/down-scaling with phase n	RW	0x00

Table 15-248. Register Call Summary for Register DISPC_VIDn_FIR_COEF_Vi

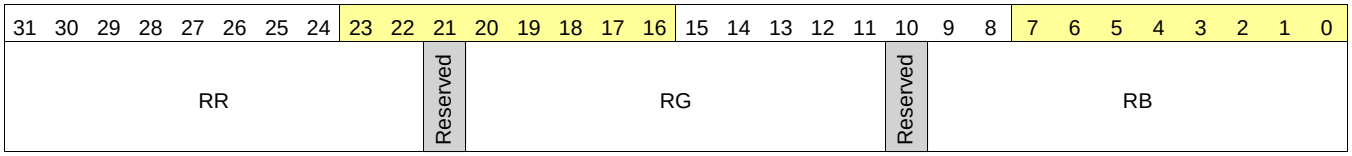
Display Subsystem Basic Programming Model
- Display Controller Basic Programming Model: [0]
- Video Layer Configuration: [1] [2]

Display Subsystem Use Cases and Tips
- Scaling Settings: [3] [4] [5] [6] [7] [8]

Display Subsystem Register Manual
- Display Subsystem Register Mapping Summary: [9] [10]

Table 15-249. DISPC_CPR_COEF_R

Address Offset	0x220		
Physical address	0x4805 0620	**Instance**	DISC
Description	This register configures the color phase rotation matrix coefficients for the red component. Shadow register, updated on VFP start period.		
Type	RW		



Bits	Field Name	Description	Type	Reset
31:22	RR	RR coefficient Encoded signed value (from -512 to 511)	RW	0x000
21	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0
20:11	RG	RG coefficient Encoded signed value (from -512 to 511)	RW	0x000
10	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0
9:0	RB	RB coefficient Encoded signed value (from -512 to 511)	RW	0x000

Table 15-250. Register Call Summary for Register DISPC_CPR_COEF_R

Display Subsystem Basic Programming Model

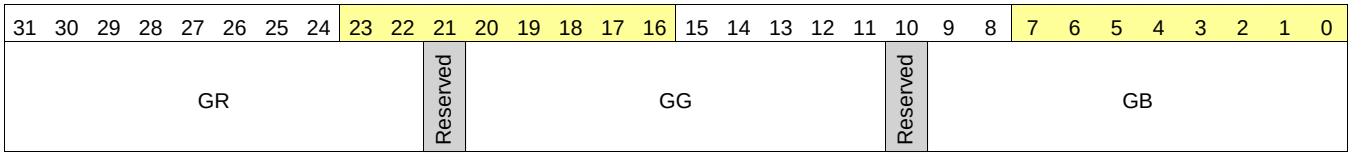
- Display Controller Basic Programming Model: [0]
- LCD-Specific Control Registers: [1] [2] [3]

Display Subsystem Register Manual

- Display Subsystem Register Mapping Summary: [4]

Table 15-251. DISPC_CPR_COEF_G

Address Offset	0x224		
Physical address	0x4805 0624	**Instance**	DISC
Description	This register configures the color phase rotation matrix coefficients for the green component. Shadow register, updated on VFP start period.		
Type	RW		



Bits	Field Name	Description	Type	Reset
31:22	GR	GR coefficient Encoded signed value (from -512 to 511)	RW	0x000
21	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0
20:11	GG	GG coefficient Encoded signed value (from -512 to 511)	RW	0x000
10	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0
9:0	GB	GB coefficient Encoded signed value (from -512 to 511)	RW	0x000

Table 15-252. Register Call Summary for Register DISPC_CPR_COEF_G

Display Subsystem Basic Programming Model

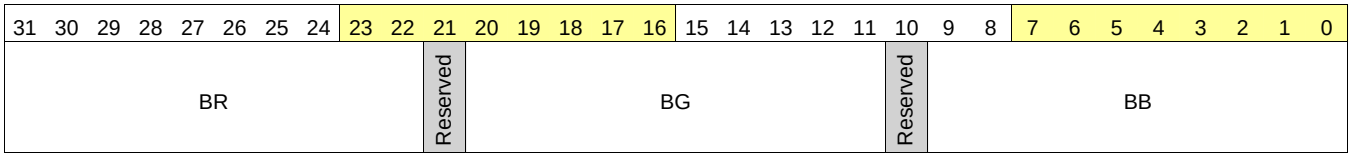
- Display Controller Basic Programming Model: [0]
- LCD-Specific Control Registers: [1] [2] [3]

Display Subsystem Register Manual

- Display Subsystem Register Mapping Summary: [4]

Table 15-253. DISPC_CPR_COEF_B

Address Offset	0x228		
Physical address	0x4805 0628	**Instance**	DISC
Description	This register configures the color phase rotation matrix coefficients for the blue component. Shadow register, updated on VFP start period.		
Type	RW		



Bits	Field Name	Description	Type	Reset
31:22	BR	BR coefficient Encoded signed value (from -512 to 511)	RW	0x000
21	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0
20:11	BG	BG coefficient Encoded signed value (from -512 to 511)	RW	0x000
10	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0
9:0	BB	BB coefficient Encoded signed value (from -512 to 511)	RW	0x000

Table 15-254. Register Call Summary for Register DISPC_CPR_COEF_B

Display Subsystem Basic Programming Model

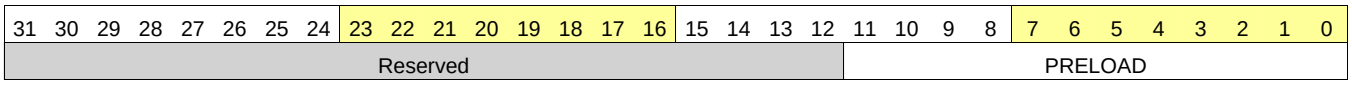
- Display Controller Basic Programming Model: [0]
- LCD-Specific Control Registers: [1] [2] [3]

Display Subsystem Register Manual

- Display Subsystem Register Mapping Summary: [4]

Table 15-255. DISPC_GFX_PRELOAD

Address Offset	0x22C		
Physical address	0x4805 062C	**Instance**	DISC
Description	This register configures the graphics FIFO. Shadow register, updated on VFP start period or EVSYNC.		
Type	RW		



Bits	Field Name	Description	Type	Reset
31:12	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0x00000
11:0	PRELOAD	Graphics preload value: Number of bytes defining the preload value. Constraint: Maximum value is (FIFO size - DMA burst size - 8) bytes	RW	0x100

Table 15-256. Register Call Summary for Register DISPC_GFX_PRELOAD

Display Subsystem Basic Programming Model • Display Controller Basic Programming Model: [0] • Graphics Layer Configuration: [1] [2] [3] [4]
Display Subsystem Register Manual • Display Subsystem Register Mapping Summary: [5]

Table 15-257. DISPC_VIDn_PRELOAD

Address Offset	0x230+ ((–1)* 0x04)	**Index**	n = 1 for VID1 or 2 for VID2
Physical address	0x4805 0630+ ((–1)* 0x04)	**Instance**	DISC
Description	This register configures the video FIFO. Shadow register, updated on VFP start period or EVSYNC.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																								PRELOAD							

Bits	Field Name	Description	Type	Reset
31:12	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0x00000
11:0	PRELOAD	Video preload value: Number of bytes defining the preload value. Constraint: Maximum value is (FIFO size - DMA burst size - 8) bytes	RW	0x100

Table 15-258. Register Call Summary for Register DISPC_VIDn_PRELOAD

Display Subsystem Basic Programming Model • Display Controller Basic Programming Model: [0] • Video Layer Configuration: [1] [2] [3] [4]
Display Subsystem Register Manual • Display Subsystem Register Mapping Summary: [5] [6]

15.7.2.3 RFBI Registers

Table 15-259. RFBI_REVISION

Address Offset	0x00		
Physical address	0x4805 0800	**Instance**	RFBI
Description	This register contains the IP revision code.		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																								REV							

Bits	Field Name	Description	Type	Reset
31:8	Reserved	Read returns 0.	R	0x000000
7:0	REV	IP revision [7:4] Major revision [3:0] Minor revision	R	TI internal data

Table 15-260. Register Call Summary for Register RFBI_REVISION

Display Subsystem Register Manual • Display Subsystem Register Mapping Summary: [0]

Table 15-261. RFBI_SYSCONFIG

Address Offset	0x10		
Physical address	0x4805 0810	**Instance**	RFBI
Description	This register allows control of various parameters of the interconnect interface.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																								Reserved	Reserved	SIDLEMODE			Reserved	SOFTRESET	AUTOIDLE

Bits	Field Name	Description	Type	Reset
31:7	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0x0000000
6	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0
5	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0
4:3	SIDLEMODE	Slave interface power management, Idle req/ack control 00: Force-idle: Idle request is acknowledged unconditionally. 01: No idle: An idle request is never acknowledged 10: Smart idle: Idle request is acknowledged based on the internal activity of the module. 11: Reserved	RW	0x0
2	Reserved	Write 0s for future compatibility Read returns 0	RW	0
1	SOFTRESET	Software reset Sets this bit to 1 to trigger a module reset. The bit is automatically reset by the hardware. During reads, it always returns 0. 0: Normal mode 1: The module is reset	RW	0
0	AUTOIDLE	Internal clock gating strategy (interconnectL4 and display controller clock) 0: Interconnect L4 clock and display controller clock are free-running. 1: Automatic clock gating strategy is applied for the interconnect L4 clock and display controller clock, based on the interconnect interface and internal activity.	RW	1

Table 15-262. Register Call Summary for Register RFBI_SYSCONFIG

Display Subsystem Integration
- Resets: [0]
- Power Management: [1] [2] [3] [4]

Display Subsystem Basic Programming Model
- RFBI Configuration: [5]

Display Subsystem Use Cases and Tips
- Autoidle and Smart Idle: [6] [7]

Display Subsystem Register Manual
- Display Subsystem Register Mapping Summary: [8]

Table 15-263. RFBI_SYSSTATUS

Address Offset	0x14		
Physical address	0x4805 0814	**Instance**	RFBI
Description	This register provides status information about the module, excluding the interrupt status information.		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																						BUSYRFBIDATA	BUSY	RESERVED							RESETDONE

Bits	Field Name	Description	Type	Reset
31: 10	Reserved	Reserved. Read returns 0	R	0x000000
9	BUSYRFBIDATA	Data are pending to be processed from interconnect FIFO. Read 0x0: No data pending Read 0x1: Some data are pending	R	0
8	BUSY	L4 Interface busy status bit Read 0x0: The access to the following register is not stalled: RFBI_CMD, RFBI_DATA, RFBI_STATUS, RFBI_PARAM, RFBI_READ. Read 0x1: The access to any of the following registers is stalled: RFBI_CMD, RFBI_DATA, RFBI_STATUS, RFBI_PARAM, RFBI_READ.	R	0
7:1	Reserved	Reserved. Read returns 0	R	0x00
0	RESETDONE	Internal reset monitoring 0: Internal module reset is on-going 1: Reset completed	R	1

Table 15-264. Register Call Summary for Register RFBI_SYSSTATUS

Display Subsystem Basic Programming Model

- RFBI Configuration: [0] [1] [2] [3] [4] [5] [6] [7] [8] [9]

Display Subsystem Register Manual

- Display Subsystem Register Mapping Summary: [10]

Table 15-265. RFBI_CONTROL

Address Offset	0x40		
Physical address	0x4805 0840	**Instance**	RFBI
Description	The control register allows configuration of the RFBI module.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																							SMART_DMA_REQ	DISABLE_DMA_REQ	HIGHTHRESHOLD			ITE	CONFIGSELECT		BYPASSMODE	ENABLE

Bits	Field Name	Description	Type	Reset
31: 9	Reserved	Write 0s for future compatibility Read returns 0	RW	0x0000000
8	SMART_DMA_REQ	Smart DMA request 0x0: The dmareq is asserted and de-asserted depending on the interconnect FIFO space even if MIdlereq is high in smart idle/no-idle mode and the entire burst gets error responses from the module. 0x1: The dmareq is de-asserted after 2 clk cycles if it has been asserted for more than or equal to 2 clk cycles and MIdlereq is high in smart idle or no idle mode. No more burst requests will be given even if the space is available in the interconnect FIFO.	RW	0x0
7	DISABLE_DMA_REQ	Disable DMA request 0x0: The dmareq is enabled and the signal is generated based on the space available and the request coming into the data register. 0x1: The dmareq is disabled and the signal is not generated at all based on space in the interconnect FIFO. It stays high until the DISABLE DMAREQ is high even if there is space in the interconnect FIFO to take requests.	RW	0x0
6:5	HIGHTHRESHOLD	Defines the interconnect FIFO high threshold used by HW to assert DMA request. Used only if data written to RFBI_DATA are sent using system DMA. 0x0: Size of the transfer of 4 words of 32-bit wide 0x1: Size of the transfer of 8 words of 32-bit wide 0x2: Size of the transfer of 16 words of 32-bit wide	RW	0x0
4	ITE	Internal Trigger 0: H/W waits for ITE bit to be set if in internal trigger mode for the configuration in use. 1: User sets the ITE bit to start the transfer, when H/W takes into account the bit, the H/W resets it.	RW	0
3:2	CONFIGSELECT	Select the CS and configuration 00: No CS selected 01: CS0 selected and configuration #0 10: CS1 selected and configuration #1 11: CS0 and CS1 both selected (only the configuration for CS0 is used)	RW	0x0
1	BYPASSMODE	Bypass Mode 0: The bypass mode not selected 1: The bypass mode is selected	RW	1
0	ENABLE	Enable/Disable flag 0: Disable the RFBI module 1: Enable the RFBI module	RW	0

Table 15-266. Register Call Summary for Register RFBI_CONTROL

Display Subsystem Environment
- Parallel Interface: [0] [1]

Display Subsystem Basic Programming Model
- RFBI Control Registers: [2] [3] [4] [5] [6] [7] [8] [9] [10] [11] [12] [13] [14] [15] [16] [17]
- RFBI Configuration: [18] [19] [20]

Display Subsystem Register Manual
- Display Subsystem Register Mapping Summary: [21]

Table 15-267. RFBI_PIXEL_CNT

Address Offset	0x44		
Physical address	0x4805 0844	**Instance**	RFBI
Description	The control register configures the RFBI pixel count value.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PIXELCNT																															

Bits	Field Name	Description	Type	Reset
31:0	PIXELCNT	Pixel counter value The S/W indicates the number of pixels to transfer to the LCD panel frame buffer. The value is set when the module is disabled. During the transfer the HW decrements the register when a pixel has been sent to the RFB.	RW	0x00000000

Table 15-268. Register Call Summary for Register RFBI_PIXEL_CNT

Display Subsystem Basic Programming Model
- RFBI Control Registers: [0] [1] [2] [3] [4]

Display Subsystem Register Manual
- Display Subsystem Register Mapping Summary: [5]

Table 15-269. RFBI_LINE_NUMBER

Address Offset	0x48		
Physical address	0x4805 0848	**Instance**	RFBI
Description	The control register configures the number of lines to synchronize the beginning of the transfer.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																					LINENUMBER										

Bits	Field Name	Description	Type	Reset
31:11	Reserved	Write 0s for future compatibility Read returns 0	RW	0x000000
10:0	LINENUMBER	Programmable line number Line number from 0 to 2^{11}-1. Number of HSYNC after the VSYNC occurs before the beginning of the transfer.	RW	0x000

Table 15-270. Register Call Summary for Register RFBI_LINE_NUMBER

Display Subsystem Basic Programming Model
- RFBI Control Registers: [0]

Display Subsystem Register Manual
- Display Subsystem Register Mapping Summary: [1]

Table 15-271. RFBI_CMD

Address Offset	0x4C		
Physical address	0x4805 084C	**Instance**	RFBI
Description	The control register configures the RFBI command		
Type	W		
Write Latency	1		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																CMD															

Bits	Field Name	Description	Type	Reset
31:16	Reserved	Write 0s for future compatibility Read returns 0	W	0x0000
15:0	CMD	Command Value 8/9/12/16 bit value depending on the parallel mode [7:0] 8-bit DT [8:0] 9-bit Data type [11:0] 12-bit Data type [15:0] 16-bit Data type	W	0x0000

Table 15-272. Register Call Summary for Register RFBI_CMD

Display Subsystem Functional Description
- Send Commands: [0]

Display Subsystem Basic Programming Model
- RFBI Control Registers: [1] [2]
- RFBI Configuration: [3] [4]

Display Subsystem Register Manual
- Display Subsystem Register Mapping Summary: [5]
- RFBI Registers: [6] [7]

Table 15-273. RFBI_PARAM

Address Offset	0x50		
Physical address	0x4805 0850	**Instance**	RFBI
Description	The control register configures the RFBI parameter.		
Type	W		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																PARAM															

Bits	Field Name	Description	Type	Reset
31:16	Reserved	Write 0s for future compatibility Read returns 0	W	0x0000
15:0	PARAM	Param Value 8/9/12/16 bit value depending on the parallel mode [7:0] 8-bit Data type [8:0] 9-bit Data type [11:0] 12-bit Data type [15:0] 16-bit Data type	W	0x0000

Table 15-274. Register Call Summary for Register RFBI_PARAM

Display Subsystem Basic Programming Model
- RFBI Control Registers: [0] [1]
- RFBI Configuration: [2] [3]

Table 15-274. Register Call Summary for Register RFBI_PARAM (continued)

Display Subsystem Register Manual • Display Subsystem Register Mapping Summary: [4] • RFBI Registers: [5] [6]

Table 15-275. RFBI_DATA

Address Offset	0x54		
Physical address	0x4805 0854	**Instance**	RFBI
Description	The control register configures the RFBI data.		
Type	W		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DATA																															

Bits	Field Name	Description	Type	Reset
31:0	DATA	Data value 12/16/18/24/2x16 bit value depending on the Data type [11:0] 12-bit Data type [15:0] 16-bit Data type [17:0] 18-bit Data type [23:0] 24-bit Data type [31:0] 2x16-bit Data type	W	0x00000000

Table 15-276. Register Call Summary for Register RFBI_DATA

Display Subsystem Overview • Display Subsystem Overview: [0]
Display Subsystem Functional Description • RFBI Interconnect FIFO: [1] [2]
Display Subsystem Basic Programming Model • RFBI Control Registers: [3] [4] [5] • RFBI Configuration: [6] [7] [8] [9] [10]
Display Subsystem Register Manual • Display Subsystem Register Mapping Summary: [11] • RFBI Registers: [12] [13] [14]

Table 15-277. RFBI_READ

Address Offset	0x58		
Physical address	0x4805 0858	**Instance**	RFBI
Description	The control register configures the RFBI read		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																READ															

Bits	Field Name	Description	Type	Reset
31:16	Reserved	Write 0s for future compatibility Read returns 0	RW	0x0000
15:0	READ	Read Value 8/9/12/16 bit value depending on the parallel mode [7:0] 8-bit Data type [8:0] 9-bit Data type [11:0] 12-bit Data type [15:0] 16-bit Data type	RW	0x0000

Table 15-278. Register Call Summary for Register RFBI_READ

Display Subsystem Basic Programming Model
- RFBI Control Registers: [0] [1]
- RFBI Configuration: [2] [3]

Display Subsystem Register Manual
- Display Subsystem Register Mapping Summary: [4]
- RFBI Registers: [5] [6]

Table 15-279. RFBI_STATUS

Address Offset	0x5C		
Physical address	0x4805 085C	**Instance**	RFBI
Description	The control register configures the RFBI status.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																STATUS															

Bits	Field Name	Description	Type	Reset
31:16	Reserved	Write 0s for future compatibility Read returns 0	RW	0x0000
15:0	STATUS	Status value 8/9/12/16 bit value depending on the parallel mode [7:0] 8-bit Data type [8:0] 9-bit Data type [11:0] 12-bit Data type [15:0] 16-bit Data type	RW	0x0000

Table 15-280. Register Call Summary for Register RFBI_STATUS

Display Subsystem Basic Programming Model
- RFBI Control Registers: [0] [1]
- RFBI Configuration: [2] [3]

Display Subsystem Register Manual
- Display Subsystem Register Mapping Summary: [4]
- RFBI Registers: [5] [6]

Table 15-281. RFBI_CONFIGi

Address Offset	0x60+ (i* 0x18)	**Index**	i = 0 to 1
Physical address	0x4805 0860+ (i* 0x18)	**Instance**	RFBI
Description	The control register allows configuration #I of the RFBI module.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved										HSYNCPOLARITY	TE_VSYNC_POLARITY	CSPOLARITY	WEPOLARITY	REPOLARITY	A0POLARITY	Reserved			UNUSEDBITS		CYCLEFORMAT		L4FORMAT		DATA TYPE		TIMEGRANULARITY		TRIGGERMODE		PARALLEL MODE

Bits	Field Name	Description	Type	Reset
31:22	Reserved	Write 0s for future compatibility Read returns 0	RW	0x000
21	HSYNCPOLARITY	HSYNC polarity 0: HSYNC active low 1: HSYNC active high	RW	1
20	TE_VSYNC_ POLARITY	TE or VSYNC Polarity 0: TE or VSYNC active low 1: TE or SYNC active high	RW	1
19	CSPOLARITY	CS Polarity 0: CS active low defined at reset time 1: CS active high defined at reset time	RW	0
18	WEPOLARITY	WE Polarity 0: WE active low 1: WE active high	RW	0
17	REPOLARITY	RE Polarity 0: RE active low 1: RE active high	RW	0
16	A0POLARITY	A0 Polarity 0: A0 active low 1: A0 active high	RW	1
15:13	Reserved	Write 0s for future compatibility Read returns 0	RW	0x0
12:11	UNUSEDBITS	State of unused bits 00: Low level (0) 01: High level (1) 10: Unchanged from previous state 11: Reserved	RW	0x0
10:9	CYCLEFORMAT	Cycle format 00: 1 cycle for 1 pixel 01: 2 cycles for 1 pixel 10: 3 cycles for 1 pixel 11: 3 cycles for 2 pixels	RW	0x0
8:7	L4FORMAT	L4 Write Access format 00: 1 pixel per L4 access to the register data 01: Reserved 10: 2 pixels per L4 access to the register data with 1st pixel at the position [15:0] 11: 2 pixels per L4 access to the register data with 1st pixel at the position [31:16]	RW	0x0
6:5	DATA TYPE	Data type from the display controller and L4 00: 12-bit 01: 16-bit 10: 18-bit 11: 24-bit	RW	0x0
4	TIMEGRANU LARITY	Multiplies signal timing latencies by two 0: x2 latencies disabled 1: x2 latencies enabled	RW	0
3:2	TRIGGERMODE	Trigger Mode 00: Internal trigger mode (ITE bit mode) 01: External trigger mode (TE signal) 10: External trigger mode (VSYNC/HSYNC signals) 11: Reserved	RW	0x0
1:0	PARALLELMODE	Parallel Mode 00: 8-bit parallel output interface selected 01: 9-bit parallel output interface selected 10: 12-bit parallel output interface selected 11: 16-bit parallel output interface selected	RW	0x0

Table 15-282. Register Call Summary for Register RFBI_CONFIGi

Display Subsystem Environment

- Parallel Interface: [0]

Table 15-282. Register Call Summary for Register RFBI_CONFIGi (continued)

Display Subsystem Functional Description • Output Parallel Modes: [1] • Read/Write: [2] [3]
Display Subsystem Basic Programming Model • RFBI Control Registers: [4] [5] [6] [7] [8] [9] [10] • RFBI Configuration: [11] [12] [13] [14] [15] [16] [17] [18] [19]
Display Subsystem Register Manual • Display Subsystem Register Mapping Summary: [20]

Table 15-283. RFBI_ONOFF_TIMEi

Address Offset	0x64+ (i* 0x18)	**Index**	i = 0 to 1
Physical address	0x4805 0864+ (i* 0x18)	**Instance**	RFBI
Description	The control register allows configuration of the RFBI timing.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved		REOFFTIME						REONTIME				WEOFFTIME				WEONTIME				CSOFFTIME				CSONTIME							

Bits	Field Name	Description	Type	Reset
31:30	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0x0
29:24	REOFFTIME	Read Enable deassertion time from start access time Number of L4Clk cycles	RW	0x00
23:20	REONTIME	Read Enable assertion time from start access time Number of L4Clk cycles	RW	0x0
19:14	WEOFFTIME	Write Enable deassertion time from start access time Number of L4Clk cycles	RW	0x00
13:10	WEONTIME	Write Enable assertion time from start access time Number of L4Clk cycles	RW	0x0
9:4	CSOFFTIME	CS deassertion time from start access time Number of L4Clk cycles	RW	0x00
3:0	CSONTIME	CS assertion time from start access time Number of L4Clk cycles	RW	0x0

Table 15-284. Register Call Summary for Register RFBI_ONOFF_TIMEi

Display Subsystem Environment • Parallel Interface: [0] [1] [2]
Display Subsystem Basic Programming Model • RFBI Configuration: [3] [4] [5] [6] [7] [8] [9] [10] [11] [12] [13] [14] [15] [16] [17]
Display Subsystem Register Manual • Display Subsystem Register Mapping Summary: [18]

Table 15-285. RFBI_CYCLE_TIMEi

Address Offset	0x68+ (i* 0x18)	**Index**	i = 0 to 1
Physical address	0x4805 0868+ (i* 0x18)	**Instance**	RFBI
Description	The control register allows configuration of the RFBI timing.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved				ACCESSTIME						WRENABLE	WWENABLE	RRENABLE	RWENABLE	CSPULSEWIDTH						RECYCLETIME						WECYCLETIME					

Bits	Field Name	Description	Type	Reset
31:28	Reserved	Write 0s for future compatibility Read returns 0	RW	0x0
27:22	ACCESSTIME	Access Time Number of L4Clk cycles	RW	0x00
21	WRENABLE	Write to Read Pulse Width Enable (same CS) 0: CSPulseWidth does not apply on Write to Read access 1: CSPulseWidth applies on Write to Read access	RW	0
20	WWENABLE	Write to Write Pulse Width Enable (same CS) 0: CSPulseWidth does not apply on Write to Write access 1: CSPulseWidth applies on Write to Write access	RW	0
19	RRENABLE	Read to Read Pulse Width Enable (same CS) 0: CSPulseWidth does not apply on Read to Read access 1: CSPulseWidth applies on Read to Read access	RW	0
18	RWENABLE	Read to Write Pulse Width Enable (same CS) 0: CSPulseWidth does not apply on Read to Write access 1: CSPulseWidth applies on Read to Write access	RW	0
17:12	CSPULSEWIDTH	CS Pulse Width Number of L4Clk cycles	RW	0x00
11:6	RECYCLETIME	RE Cycle Time Number of L4Clk cycles	RW	0x00
5:0	WECYCLETIME	WE Cycle Time Number of L4Clk cycles	RW	0x00

Table 15-286. Register Call Summary for Register RFBI_CYCLE_TIMEi

Display Subsystem Environment
- Parallel Interface: [0] [1] [2]

Display Subsystem Basic Programming Model
- RFBI Configuration: [3] [4] [5] [6] [7] [8] [9] [10] [11] [12] [13] [14] [15]

Display Subsystem Register Manual
- Display Subsystem Register Mapping Summary: [16]

Table 15-287. RFBI_DATA_CYCLE1_i

Address Offset	0x6C+ (i* 0x18)	**Index**	i = 0 to 1
Physical address	0x4805 086C+ (i* 0x18)	**Instance**	RFBI
Description	The control register configures the RFBI data format for 1st cycle.		
Type	RW		

31 30 29 28	27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10 9 8	7 6 5 4 3 2 1 0
Reserved	BITALIGNMENTPIXEL2	Reserved / NBBITSPIXEL2	Reserved / BITALIGNMENTPIXEL1	Reserved / NBBITSPIXEL1

Bits	Field Name	Description	Type	Reset
31:28	Reserved	Write 0s for future compatibility Read returns 0	RW	0x0
27:24	BITALIGNMENTPIXEL2	Bit alignment Alignment of the bits from pixel#2 on the output interface	RW	0x0
23:21	Reserved	Write 0s for future compatibility Read returns 0	RW	0x0
20:16	NBBITSPIXEL2	Number of bits Number of bits from the pixel #2 (value from 0 to16 bits). The values from 17 to 31 are invalid.	RW	0x00
15:12	Reserved	Write 0s for future compatibility Read returns 0	RW	0x0
11:8	BITALIGNMENTPIXEL1	Bit alignment Alignment of the bits from pixel#1 on the output interface	RW	0x0
7:5	Reserved	Write 0s for future compatibility Read returns 0	RW	0x0
4:0	NBBITSPIXEL1	Number of bits Number of bits from the pixel #1 (value from 0 to16 bits). The values from 17 to 31 are invalid.	RW	0x00

Table 15-288. Register Call Summary for Register RFBI_DATA_CYCLE1_i

Display Subsystem Functional Description
- Output Parallel Modes: [0]

Display Subsystem Basic Programming Model
- RFBI Configuration: [1] [2]

Display Subsystem Register Manual
- Display Subsystem Register Mapping Summary: [3]

Table 15-289. RFBI_DATA_CYCLE2_i

Address Offset	0x70+ (i* 0x18)	**Index**	i = 0 to 1
Physical address	0x4805 0870+ (i* 0x18)	**Instance**	RFBI
Description	The control register configures the RFBI data format for 2nd cycle.		
Type	RW		

31 30 29 28	27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10 9 8	7 6 5 4 3 2 1 0
Reserved	BITALIGNMENTPIXEL2	Reserved / NBBITSPIXEL2	Reserved / BITALIGNMENTPIXEL1	Reserved / NBBITSPIXEL1

Bits	Field Name	Description	Type	Reset
31:28	Reserved	Write 0s for future compatibility Read returns 0	RW	0x0
27:24	BITALIGNMENTPIXEL2	Bit alignment Alignment of the bits from pixel#2 on the output interface	RW	0x0
23:21	Reserved	Write 0s for future compatibility Read returns 0	RW	0x0
20:16	NBBITSPIXEL2	Number of bits Number of bits from the pixel #2 (value from 0 to16 bits). The values from 17 to 31 are invalid.	RW	0x00
15:12	Reserved	Write 0s for future compatibility Read returns 0	RW	0x0
11:8	BITALIGNMENTPIXEL1	Bit alignment Alignment of the bits from pixel#1 on the output interface	RW	0x0
7:5	Reserved	Write 0s for future compatibility Read returns 0	RW	0x0
4:0	NBBITSPIXEL1	Number of bits Number of bits from the pixel #1 (value from 0 to16 bits). The values from 17 to 31 are invalid.	RW	0x00

Table 15-290. Register Call Summary for Register RFBI_DATA_CYCLE2_i

Display Subsystem Functional Description
- Output Parallel Modes: [0]

Display Subsystem Basic Programming Model
- RFBI Configuration: [1] [2]

Display Subsystem Register Manual
- Display Subsystem Register Mapping Summary: [3]

Table 15-291. RFBI_DATA_CYCLE3_i

Address Offset	0x74+ (i* 0x18)	**Index**	i = 0 to 1
Physical address	0x4805 0874+ (i* 0x18)	**Instance**	RFBI
Description	The control register configures the RFBI data format for 3rd cycle.		
Type	RW		

31 30 29 28	27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10 9 8	7 6 5 4 3 2 1 0
Reserved	BITALIGNMENTPIXEL2	Reserved / NBBITSPIXEL2	Reserved / BITALIGNMENTPIXEL1	Reserved / NBBITSPIXEL1

Bits	Field Name	Description	Type	Reset
31:28	Reserved	Write 0s for future compatibility Read returns 0	RW	0x0
27:24	BITALIGNMENTPIXEL2	Bit alignment Alignment of the bits from pixel#2 on the output interface	RW	0x0
23:21	Reserved	Write 0s for future compatibility Read returns 0	RW	0x0
20:16	NBBITSPIXEL2	Number of bits Number of bits from the pixel #2 (value from 0 to16 bits). The values from 17 to 31 are invalid.	RW	0x00
15:12	Reserved	Write 0s for future compatibility Read returns 0	RW	0x0

Bits	Field Name	Description	Type	Reset
11:8	BITALIGNMENTPIXEL1	Bit alignment Alignment of the bits from pixel#1 on the output interface	RW	0x0
7:5	Reserved	Write 0s for future compatibility Read returns 0	RW	0x0
4:0	NBBITSPIXEL1	Number of bits Number of bits from the pixel #1 (value from 0 to16 bits). The values from 17 to 31 are invalid.	RW	0x00

Table 15-292. Register Call Summary for Register RFBI_DATA_CYCLE3_i

Display Subsystem Functional Description
- Output Parallel Modes: [0]

Display Subsystem Basic Programming Model
- RFBI Configuration: [1]

Display Subsystem Register Manual
- Display Subsystem Register Mapping Summary: [2]

Table 15-293. RFBI_VSYNC_WIDTH

Address Offset	0x90		
Physical address	0x4805 0890	**Instance**	RFBI
Description	The control register configures the RFBI VSYNC minimum pulse width		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																MINVSYNCPULSEWIDTH															

Bits	Field Name	Description	Type	Reset
31:16	Reserved	Write 0s for future compatibility Read returns 0	RW	0x0000
15:0	MINVSYNCPULSEWIDTH	Programmable min VSYNC pulse width Minimum VSYNC pulse width from 0 to 65535. Number of L4 clock cycles to determine when VSYNC pulse occurs. The values 0 and 1 are invalid.	RW	0x0000

Table 15-294. Register Call Summary for Register RFBI_VSYNC_WIDTH

Display Subsystem Environment
- Parallel Interface: [0]

Display Subsystem Basic Programming Model
- RFBI Configuration: [1] [2]

Display Subsystem Register Manual
- Display Subsystem Register Mapping Summary: [3]

Table 15-295. RFBI_HSYNC_WIDTH

Address Offset	0x94		
Physical address	0x4805 0894	**Instance**	RFBI
Description	The control register configures the RFBI HSYNC minimum pulse width.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																MINHSYNCPULSEWIDTH															

Bits	Field Name	Description	Type	Reset
31:16	Reserved	Write 0s for future compatibility Read returns 0	RW	0x0000
15:0	MINHSYNC PULSEWIDTH	Programmable min HSYNC pulse width Minimum HSYNC pulse width from 0 to 65535. Number of L4 clock cycles to determine when HSYNC pulse occurs. The values 0 and 1 are invalid.	RW	0x0000

Table 15-296. Register Call Summary for Register RFBI_HSYNC_WIDTH

Display Subsystem Environment • Parallel Interface: [0]
Display Subsystem Basic Programming Model • RFBI Configuration: [1] [2]
Display Subsystem Register Manual • Display Subsystem Register Mapping Summary: [3]

15.7.2.4 Video Encoder Registers

Table 15-297. VENC_REV_ID

Address Offset	0x00		
Physical address	0x4805 0C00	**Instance**	VENC
Description	Revision ID for the encoder		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																								REV_ID							

Bits	Field Name	Description	Type	Reset
31:8	Reserved	Reserved. Read returns 0s.	R	0x000000
7:0	REV_ID	This read-only register contains the revision ID for the encoder. The revision ID will identify different revisions of the IP.	R	TI internal data

Table 15-298. Register Call Summary for Register VENC_REV_ID

Display Subsystem Register Manual • Display Subsystem Register Mapping Summary: [0]

Table 15-299. VENC_STATUS

Address Offset	0x04		
Physical address	0x4805 0C04	**Instance**	VENC
Description	VENC_STATUS		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																											CCE	CCO	FSQ			

Bits	Field Name	Description	Type	Reset
31:5	Reserved	Reserved. Read returns 0s.	R	0x0000000
4	CCE	Closed caption status for even Field. This bit is set immediately after the data in registers LINE21_E0 and LINE21_E1 have been encoded to closed caption. This bit is reset when both of these registers are written.	R	0

Bits	Field Name	Description	Type	Reset
3	CCO	Closed Caption Status for Odd Field. This bit is set immediately after the data in registers LINE21_O0 and LINE21_O1 have been encoded to closed caption. This bit is reset when both of these registers are written.	R	0
2:0	FSQ	Field Sequence ID. For PAL, all three FSQ[2:0] are used whereas for NTSC only FSQ[1:0] is meaningful. Furthermore, FSQ[0] represents odd field when it is 0 and even field when it is 1. Read 0x0: Odd field Read 0x1: Even field	R	0x0

Table 15-300. Register Call Summary for Register VENC_STATUS

Display Subsystem Functional Description
- Closed Caption Encoding: [0] [1] [2] [3]

Display Subsystem Register Manual
- Display Subsystem Register Mapping Summary: [4]
- Video Encoder Registers: [5]

Table 15-301. VENC_F_CONTROL

Address Offset	0x08		
Physical address	0x4805 0C08	**Instance**	VENC
Description	This register specifies the input video source and format		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																							RESET	SVDS		RGBF	BCOLOR			FMT	

Bits	Field Name	Description	Type	Reset
31:9	Reserved	Reserved. Read returns 0s.	RW	0x000000
8	RESET	RESET the encoder 0x0: No effect 0x1: Reset the encoder, after reset, this bit is automatically set to zero.	RW	0
7:6	SVDS	Select Video Data Source. 0x0: Use external video source 0x1: Use internal Color BAR 0x2: Use background color 0x3: Reserved	RW	0x2
5	RGBF	RGB/YCrCb input coding range 0x0: The input RGB data are in binary format with coding range 0-255 The input YCrCb data are in binary format with coding range 0-255 0x1: The input RGB data are in binary format with coding range 16-235 The input YCrCb data are in binary format conforming to ITU-601 standard	RW	0
4:2	BCOLOR	Background color select 0x0: black 0x1: blue 0x2: red 0x3: magenta	RW	0x1

Bits	Field Name	Description	Type	Reset
		0x4: green		
		0x5: cyan		
		0x6: yellow		
		0x7: white		
1:0	FMT	These two bits specify the video input data stream format and timing	RW	0x3
		0x0: 24-bit 4:4:4 RGB		
		0x1: 24-bit 4:4:4		
		0x2: 16-bit 4:2:2		
		0x3: 8-bit ITU-R 656 4:2:2		

Table 15-302. Register Call Summary for Register VENC_F_CONTROL

Display Subsystem Functional Description
• Test Pattern Generation: [0]
Display Subsystem Basic Programming Model
• Video Encoder Software Reset: [1]
• Video Encoder Programming Sequence: [2] [3]
• Video Encoder Register Settings: [4]
Display Subsystem Register Manual
• Display Subsystem Register Mapping Summary: [5]

Table 15-303. VENC_VIDOUT_CTRL

Address Offset	0x10		
Physical address	0x4805 0C10	**Instance**	VENC
Description	Encoder output clock		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																															27_54

Bits	Field Name	Description	Type	Reset
31:1	Reserved	Reserved. Read returns 0s.	RW	0x00000000
0	27_54	Encoder output clock	RW	0
		0x0: 54 MHz, 4x oversampling		
		0x1: 27 MHz, 2x oversampling, the last 2x oversampling filter bypassed		

Table 15-304. Register Call Summary for Register VENC_VIDOUT_CTRL

Display Subsystem Basic Programming Model
• Video Encoder Register Settings: [0]
Display Subsystem Register Manual
• Display Subsystem Register Mapping Summary: [1]

Table 15-305. VENC_SYNC_CTRL

Address Offset	0x14		
Physical address	0x4805 0C14	**Instance**	VENC
Description	Sync Control Register		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																FREE	ESAV	IGNP	NBLNKS	VBLKM		HBLKM		Reserved	FID_POL	Reserved					

Bits	Field Name	Description	Type	Reset
31:16	Reserved	Reserved. Read returns 0s.	RW	0x0000
15	FREE	Free running	RW	1
		0x0: Free running disabled		
		0x1: Free running enabled. HSYNC and VSYNC are ignored		
14	ESAV	Enable to detect F and V bits only on EAV in ITU-R 656 input mode	RW	0
		0x0: Detection of F and V bits on both EAV and SAV		
		0x1: Detection of F and V bits only on EAV		
13	IGNP	Ignore protection bits in ITU-R 656 input mode	RW	0
		0x0: Protection bits are not ignored		
		0x1: Protection bits are ignored		
12	NBLNKS	Blank shaping	RW	0
		0x0: Blank shaping enabled		
		0x1: Blank shaping disabled		
11:10	VBLKM	Vertical blanking mode	RW	0x0
		0x0: Internal default blanking		
		0x1: Internal default blanking AND internal programmable blanking defined by VENC_FLEN_FAL[24:16]FAL and VENC_LAL_PHASE_RESET[8:0] LAL bit fields. Note: in this mode, the VENC_LAL_PHASE_RESET[16] SBLANK bit must be '0b1' to active the VBLKM functionality.		
		0x2: Reserved		
		0x3: Reserved		
9:8	HBLKM	Horizontal blanking mode	RW	0x0
		0x0: Internal default blanking		
		0x1: Internal programmable blanking defined by VENC_SAVID_EAVID[26:16] SAVID and VENC_SAVID_EAVID[10:0] EAVID bit fields.		
		0x2: External blanking defined by VENC_AVID_START_STOP_X and VENC_AVID_START_STOP_Y registers.		
		0x3: Reserved		
7	Reserved	Reserved. Read returns 0.	RW	0
6	FID_POL	FID output polarity	RW	0
		0x0: Odd field = 0 Even field = 1		
		0x1: Odd field = 1 Even field = 0		
5:0	Reserved	Reserved. Read returns 0.	RW	0x00

Table 15-306. Register Call Summary for Register VENC_SYNC_CTRL

Display Subsystem Basic Programming Model
- Video Encoder Programming Sequence: [0]
- Video Encoder Register Settings: [1]

Display Subsystem Register Manual
- Display Subsystem Register Mapping Summary: [2]
- Video Encoder Registers: [3]

Table 15-307. VENC_LLEN

Address Offset	0x1C		
Physical address	0x4805 0C1C	**Instance**	VENC
Description	VENC_LLEN		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																Reserved					LLEN										

Bits	Field Name	Description	Type	Reset
31:15	Reserved	Reserved. Read returns 0s.	RW	0x0000
14:11	Reserved	Reserved. Read returns 0s.	RW	0x0
10:0	LLEN	LLEN[10:0] Line length or total number of pixels in a scan line including active video and blanking. Total number of pixels in a scan line = LLEN **NOTE:** A write on the LLEN[10] is illegal.	RW	0x359

Table 15-308. Register Call Summary for Register VENC_LLEN

Display Subsystem Basic Programming Model
- Video Encoder Register Settings: [0]

Display Subsystem Register Manual
- Display Subsystem Register Mapping Summary: [1]
- Video Encoder Registers: [2] [3]

Table 15-309. VENC_FLENS

Address Offset	0x20		
Physical address	0x4805 0C20	**Instance**	VENC
Description	VENC_FLENS		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																					FLENS										

Bits	Field Name	Description	Type	Reset
31:11	Reserved	Reserved. Read returns 0s.	RW	0x000000
10:0	FLENS	The frame length or total number of lines in a frame including active video and blanking from the source image. Total number of lines in a frame from the source image = FLENS + 1	RW	0x20C

Table 15-310. Register Call Summary for Register VENC_FLENS

Display Subsystem Basic Programming Model
- Video Encoder Register Settings: [0]

Display Subsystem Register Manual
- Display Subsystem Register Mapping Summary: [1]
- Video Encoder Registers: [2]

Table 15-311. VENC_HFLTR_CTRL

Address Offset	0x24		
Physical address	0x4805 0C24	**Instance**	VENC
Description	VENC_HFLTR_CTRL		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																													CINTP	YINTP	

Bits	Field Name	Description	Type	Reset
31:3	Reserved	Reserved. Read returns 0s.	RW	0x00000000
2:1	CINTP	Chrominance interpolation filter control	RW	0x0
		0x0: The chrominance interpolation filter is enabled		
		0x1: The first section of the chrominance interpolation filter is bypassed		
		0x2: The second section of the chrominance interpolation filter is bypassed		
		0x3: Both sections of the filter are bypassed		
0	YINTP	Luminance interpolation filter control	RW	0
		0x0: The luminance interpolation filter is enabled		
		0x1: The luminance interpolation filter is bypassed		

Table 15-312. Register Call Summary for Register VENC_HFLTR_CTRL

Display Subsystem Basic Programming Model
- Video Encoder Register Settings: [0]

Display Subsystem Register Manual
- Display Subsystem Register Mapping Summary: [1]
- Video Encoder Registers: [2]

Table 15-313. VENC_CC_CARR_WSS_CARR

Address Offset	0x28		
Physical address	0x4805 0C28	**Instance**	VENC
Description	Frequency code control		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FWSS																FCC															

Bits	Field Name	Description	Type	Reset
31:16	FWSS	Wide screen signaling run-in code frequency control For common values for FWSS[15:0] bit field, refer to Table 15-44 Reset value is for NTSC-601standard	RW	0x043F
15:0	FCC	Close caption run-in code frequency control For common values for FCC[15:0] bit field refer to Table 15-42. Reset value is for NTSC-601 standard	RW	0x2631

Table 15-314. Register Call Summary for Register VENC_CC_CARR_WSS_CARR

Display Subsystem Functional Description
- Closed Caption Encoding: [0] [1] [2]
- Wide-Screen Signaling (WSS) Encoding: [3] [4]

Display Subsystem Basic Programming Model
- Video Encoder Register Settings: [5]

Display Subsystem Register Manual
- Display Subsystem Register Mapping Summary: [6]

Table 15-315. VENC_C_PHASE

Address Offset	0x2C		
Physical address	0x4805 0C2C	**Instance**	VENC
Description	VENC_C_PHASE		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																								CPHS							

Bits	Field Name	Description	Type	Reset
31:8	Reserved	Reserved. Read returns 0.	RW	0x000000
7:0	CPHS	Phase of the encoded video color subcarrier (including the color burst) relative to H-sync. The adjustable step is 360/256 degrees.	RW	0x00

Table 15-316. Register Call Summary for Register VENC_C_PHASE

Display Subsystem Functional Description
- Subcarrier and Burst Generation: [0] [1] [2]

Display Subsystem Basic Programming Model
- Video Encoder Register Settings: [3]

Display Subsystem Register Manual
- Display Subsystem Register Mapping Summary: [4]
- Video Encoder Registers: [5]

Table 15-317. VENC_GAIN_U

Address Offset	0x30		
Physical address	0x4805 0C30	**Instance**	VENC
Description	Gain control for Cb signal		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																								GU							

Bits	Field Name	Description	Type	Reset
31:9	Reserved	Reserved. Read returns 0s.	RW	0x000000
8:0	GU	Gain control for Cb signal. Following are typical programming examples for NTSC and PAL standards. NTSC with 7.5 IRE pedestal: WHITE - BLACK = 92.5 IRE GU = 0x102 NTSC with no pedestal: WHITE - BLACK = 100 IRE GU = 0x117 PAL with no pedestal: WHITE - BLACK = 100 IRE GU = 0x111	RW	0x102

Table 15-318. Register Call Summary for Register VENC_GAIN_U

Display Subsystem Functional Description • Chroma Stage: [0]
Display Subsystem Basic Programming Model • Video Encoder Register Settings: [1]
Display Subsystem Register Manual • Display Subsystem Register Mapping Summary: [2]

Table 15-319. VENC_GAIN_V

Address Offset	0x34		
Physical address	0x4805 0C34	**Instance**	VENC
Description	Gain control of Cr signal		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																								GV							

Bits	Field Name	Description	Type	Reset
31:9	Reserved	Reserved. Read returns 0s.	RW	0x000000
8:0	GV	Gain control of Cr signal. Following are typical programming examples for NTSC and PAL standards. NTSC with 7.5 IRE pedestal: WHITE - BLACK = 92.5 IRE GV = 0x16C NTSC with no pedestal: WHITE - BLACK = 100 IRE GV = 0x189 PAL with no pedestal: WHITE - BLACK = 100 IRE GV = 0x181	RW	0x16C

Table 15-320. Register Call Summary for Register VENC_GAIN_V

Display Subsystem Functional Description • Chroma Stage: [0]
Display Subsystem Basic Programming Model • Video Encoder Register Settings: [1]
Display Subsystem Register Manual • Display Subsystem Register Mapping Summary: [2]

Table 15-321. VENC_GAIN_Y

Address Offset	0x38		
Physical address	0x4805 0C38	**Instance**	VENC
Description	Gain control of Y signal		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																								GY							

Bits	Field Name	Description	Type	Reset
31:9	Reserved	Reserved. Read returns 0s.	RW	0x000000
8:0	GY	Gain control of Y signal. Following are typical programming examples for NTSC/PAL standards. NTSC with 7.5 IRE pedestal: WHITE - BLACK = 92.5 IRE GY = 0x12F NTSC with no pedestal: WHITE - BLACK = 100 IRE GY = 0x147 PAL with no pedestal: WHITE - BLACK = 100 IRE GY = 0x140	RW	0x12F

Table 15-322. Register Call Summary for Register VENC_GAIN_Y

Display Subsystem Functional Description • Luma Stage: [0]
Display Subsystem Basic Programming Model • Video Encoder Register Settings: [1]
Display Subsystem Register Manual • Display Subsystem Register Mapping Summary: [2]

Table 15-323. VENC_BLACK_LEVEL

Address Offset	0x3C		
Physical address	0x4805 0C3C	**Instance**	VENC
Description	Video Encoder BLACK LEVEL		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																								BLACK							

Bits	Field Name	Description	Type	Reset
31:7	Reserved	Reserved. Read returns 0.	RW	0x0000000
6:0	BLACK	Black level setting. Following are typical programming examples for NTSC/PAL standards. NTSC with 7.5 IRE pedestal: WHITE - BLACK = 92.5 IRE BLACK_LEVEL = 0x43 NTSC with no pedestal: WHITE - BLACK = 100 IRE BLACK_LEVEL = 0x38 PAL with no pedestal: WHITE - BLACK = 100 IRE BLACK_LEVEL = 0x3B	RW	0x43

Table 15-324. Register Call Summary for Register VENC_BLACK_LEVEL

Display Subsystem Functional Description • Luma Stage: [0]
Display Subsystem Basic Programming Model • Video Encoder Register Settings: [1]
Display Subsystem Register Manual • Display Subsystem Register Mapping Summary: [2]

Table 15-325. VENC_BLANK_LEVEL

Address Offset	0x40		
Physical address	0x4805 0C40	**Instance**	VENC
Description	Video Encoder BLANK LEVEL		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																								BLANK							

Bits	Field Name	Description	Type	Reset
31:7	Reserved	Reserved. Read returns 0s.	RW	0x0000000
6:0	BLANK	Blank level setting. Following are typical programming examples for NTSC/PAL standards. NTSC with 7.5 IRE pedestal: WHITE - BLACK = 92.5 IRE BLANK_LEVEL = 0x38 NTSC with no pedestal: WHITE - BLACK = 100 IRE BLANK_LEVEL = 0x38 PAL with no pedestal: WHITE - BLACK = 100 IRE BLANK_LEVEL = 0x3B	RW	0x38

Table 15-326. Register Call Summary for Register VENC_BLANK_LEVEL

Display Subsystem Functional Description • Luma Stage: [0]
Display Subsystem Basic Programming Model • Video Encoder Register Settings: [1]
Display Subsystem Register Manual • Display Subsystem Register Mapping Summary: [2]

Table 15-327. VENC_X_COLOR

Address Offset	0x44		
Physical address	0x4805 0C44	**Instance**	VENC
Description	Cross-Color Control Register		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																								XCE	Reserved		XCBW		LCD		

Bits	Field Name	Description	Type	Reset
31:7	Reserved	Reserved. Read returns 0s.	RW	0x0000000
6	XCE	Cross color reduction enable for composite video output. Cross color does not affect S-video output 0x0: Cross color reduction is disabled 0x1: Cross color is enabled	RW	0
5	Reserved	Reserved. Read returns 0.	RW	0
4:3	XCBW	Cross color reduction filter selection 0x0: The notch is at 32.8 % of the frequency of the encoding pixel clock 0x1: The notch is at 26.5 % of the frequency of the encoding pixel clock 0x2: The notch is at 30.0 % of the frequency of the encoding pixel clock 0x3: The notch is at 29.2 % of the frequency of the encoding pixel clock	RW	0x0
2:0	LCD	These three bits can be used for chroma channel delay compensation. Delay on Luma channel. 0x0: 0 0x1: 0.5 pixel clock period 0x2: 1.0 pixel clock period 0x3: 1.5 pixel clock period 0x4: -2.0 pixel clock period 0x5: -1.5 pixel clock period 0x6: -1.0 pixel clock period 0x7: -0.5 pixel clock period	RW	0x0

Table 15-328. Register Call Summary for Register VENC_X_COLOR

Display Subsystem Basic Programming Model • Video Encoder Register Settings: [0]
Display Subsystem Register Manual • Display Subsystem Register Mapping Summary: [1]

Table 15-329. VENC_M_CONTROL

Address Offset	0x48		
Physical address	0x4805 0C48	**Instance**	VENC
Description	VENC_M_CONTROL		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																								PALI	PALN	PALPHS	CBW			PAL	FFRQ

Bits	Field Name	Description	Type	Reset
31:8	Reserved	Reserved. Read returns 0s.	RW	0x0000000
7	PALI	PAL I enable	RW	0
		0x0: Normal operation		
		0x1: PAL I enable		
6	PALN	PAL N enable	RW	0
		0x0: Normal operation		
		0x1: PAL N enable		
5	PALPHS	PAL switch phase setting	RW	0
		0x0: PAL switch phase is nominal		
		0x1: PAL switch phase is inverted compared to nominal		
4:2	CBW	Chrominance lowpass filter bandwidth control	RW	0x0
		0x0: -6db at 21.8 % of encoding pixel clock frequency		
		0x1: -6db at 19.8 % of encoding pixel clock frequency		
		0x2: -6db at 18.0 % of encoding pixel clock frequency		
		0x3: Reserved		
		0x4: Reserved		
		0x5: -6db at 23.7 % of encoding pixel clock frequency		
		0x6: -6db at 26.8 % of encoding pixel clock frequency		
		0x7: Chrominance lowpass filter bypass		
1	PAL	Phase alternation line encoding selection	RW	0
		0x0: Phase alternation line encoding disabled		
		0x1: Phase alternation line encoding enabled		
0	FFRQ	The value of this field and the SQP bit in the VENC_BSTAMP_WSS_DATA[7] SQP bit control the number of horizontal pixels displayed per scan line	RW	1
		Mode: Configuration:		
		ITU-R 601 NTSC SQP = 0, FFRQ = 1, Number of pixels by line = 858		
		Square pixel NTSC SQP = 1, FFRQ = 1, Number of pixels by line = 780		
		ITU-R 601 PAL SQP = 0, FFRQ = 0, Number of pixels by line = 864		
		Square pixel PAL SQP = 1, FFRQ = 0, Number of pixels by line = 944		

Table 15-330. Register Call Summary for Register VENC_M_CONTROL

Display Subsystem Functional Description
- Subcarrier and Burst Generation: [0] [1] [2]

Display Subsystem Basic Programming Model
- Video Encoder Register Settings: [3]

Display Subsystem Register Manual
- Display Subsystem Register Mapping Summary: [4]
- Video Encoder Registers: [5] [6]

Table 15-331. VENC_BSTAMP_WSS_DATA

Address Offset	0x4C		
Physical address	0x4805 0C4C	**Instance**	VENC
Description	VENC BSTAMP and WSS_DATA		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								WSS_DATA																SQP	BSTAP							

Bits	Field Name	Description			Type	Reset
31:28	Reserved	Reserved. Read returns 0s.			RW	0x0
27:8	WSS_DATA	WSS data [19:0]: Wide Screen Signaling data			RW	0x00000
		NTSC:	WORD 0	WSS_D1, WSS_D0		
			WORD 1	WSS_D5, WSS_D4, WSS_D3, WSS_D2		
			WORD 2	WSS_D13, WSS_D12, WSS_D11, WSS_D10, WSS_D9, WSS_D8, WSS_D7, WSS_D6		
			CRC	WSS_D19, WSS_D18, WSS_D17, WSS_D16, WSS_D15, WSS_D14		
		PAL:	GROUP A	WSS_D3, WSS_D2, WSS_D1, WSS_D0		
			GROUP B	WSS_D7, WSS_D6, WSS_D5, WSS_D4		
			GROUP C	WSS_D10, WSS_D9, WSS_D8		
			GROUP D	WSS_D13, WSS_D12, WSS_D11		
7	SQP	Square-pixel sampling rate. Please refer to VENC_M_CONTROL[0] FFRQ bit description for programming information.			RW	0
		0x0:	ITU-R 601 sampling rate			
		0x1:	Square-pixel sampling rate			
6:0	BSTAP	Setting of amplitude of color burst.			RW	0x38

Table 15-332. Register Call Summary for Register VENC_BSTAMP_WSS_DATA

Display Subsystem Functional Description
- Subcarrier and Burst Generation: [0]
- Wide-Screen Signaling (WSS) Encoding: [1]

Display Subsystem Basic Programming Model
- Video Encoder Register Settings: [2]

Display Subsystem Register Manual
- Display Subsystem Register Mapping Summary: [3]
- Video Encoder Registers: [4]

Table 15-333. VENC_S_CARR

Address Offset	0x50		
Physical address	0x4805 0C50	**Instance**	VENC
Description	Color Subcarrier Frequency Registers.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FSC																															

Bits	Field Name	Description	Type	Reset
31:0	FSC	These four bytes' data program the color subcarrier frequency and are determined by the following formula. S_CARR = ROUND((Fsc/Fclkenc) * 2^{32}) Where: Fsc = Frequency of the subcarrier Fclkenc = Frequency of the internal video encoding clock = 2*LLEN *Fh LLEN = Number of pixels in a scan line. For LLEN setting, please refer to the description of VENC_LLEN register (offset 0x1C). Fh = Line frequency For typical setting of the FSC field, refer to Table 15-40.	RW	0x21F07C1F

Table 15-334. Register Call Summary for Register VENC_S_CARR

Display Subsystem Functional Description • Subcarrier and Burst Generation: [0] [1] [2]
Display Subsystem Basic Programming Model • Video Encoder Register Settings: [3]
Display Subsystem Register Manual • Display Subsystem Register Mapping Summary: [4]

Table 15-335. VENC_LINE21

Address Offset	0x54		
Physical address	0x4805 0C54	**Instance**	VENC
Description	VENC LINE 21		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
L21E																L21O															

Bits	Field Name	Description	Type	Reset
31:16	L21E	The two bytes of the venc_line21 data in the even field. For a complete field description, refer to CEA-608-x standard.	RW	0x0000
15:0	L21O	The two bytes of the closed caption data in the odd field. For a complete field description, refer to CEA-608-x standard.	RW	0x0000

Table 15-336. Register Call Summary for Register VENC_LINE21

Display Subsystem Functional Description • Closed Caption Encoding: [0] [1] [2] [3]
Display Subsystem Basic Programming Model • Video Encoder Register Settings: [4]
Display Subsystem Register Manual • Display Subsystem Register Mapping Summary: [5]

Table 15-337. VENC_LN_SEL

Address Offset	0x58		
Physical address	0x4805 0C58	**Instance**	VENC
Description	VENC_LN_SEL		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								LN21_RUNIN								Reserved								SLINE							

Bits	Field Name	Description	Type	Reset
31:26	Reserved	Reserved. Read returns 0s.	RW	0x00
25:16	LN21_RUNIN	The two bytes of the closed caption run in code position from the HSYNC.	RW	0x10B
15:5	Reserved	Reserved. Read returns 0s.	RW	0x000
4:0	SLINE	Selects the line where closed caption or extended service data are encoded. There is a four line offset, so program the wanted line number - 4. The default value is line 0x15 + 4 = 0x19 (25th line).	RW	0x15

Table 15-338. Register Call Summary for Register VENC_LN_SEL

Display Subsystem Functional Description • Closed Caption Encoding: [0] [1] [2]
Display Subsystem Basic Programming Model • Video Encoder Register Settings: [3]
Display Subsystem Register Manual • Display Subsystem Register Mapping Summary: [4] • Video Encoder Registers: [5]

Table 15-339. VENC_L21_WC_CTL

Address Offset	0x5C		
Physical address	0x4805 0C5C	**Instance**	VENC
Description	VENC L21 & WC_CTL registers		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																INV	EVEN_ODD_EN		LINE					Reserved						L21EN	

Bits	Field Name	Description	Type	Reset
31:16	Reserved	Reserved. Read returns 0s.	RW	0x0000
15	INV	WSS inverter 0x0: No effect 0x1: Invert WSS data	RW	0
14:13	EVEN_ODD_EN	This bit controls the WSS encoding. 0x0: WSS encoding OFF 0x1: Enables encoding in 1st field (odd field) 0x2: Enables encoding in 2nd field (even field) 0x3: Enables encoding in both fields	RW	0x0
12:8	LINE	Selects the line where WSS data are encoded. The LINE[4:0] bit field value depends on the video standard: PAL mode: There is an one line offset, so program the wanted line number - 1. The recommended value is line 0x16 + 1 = 0x17 (23rd line). NTSC mode: There is a four line offset, so program the wanted line number - 4. The recommended value is line 0x10 + 4 = 0x14 (20th line).	RW	0x14
7:2	Reserved	Reserved. Read returns 0s.	RW	0x00
1:0	L21EN	Those bits controls the Line21 closed caption encoding according to the mode. 0x0: Line21 encoding OFF 0x1: Enables encoding in 1st field (odd field) 0x2: Enables encoding in 2d field (even field)	RW	0x0

Bits	Field Name	Description	Type	Reset
		0x3: Enables encoding in both fields		

Table 15-340. Register Call Summary for Register VENC_L21_WC_CTL

Display Subsystem Functional Description • Closed Caption Encoding: [0] [1] • Wide-Screen Signaling (WSS) Encoding: [2] [3]
Display Subsystem Basic Programming Model • Video Encoder Register Settings: [4]
Display Subsystem Register Manual • Display Subsystem Register Mapping Summary: [5]

Table 15-341. VENC_HTRIGGER_VTRIGGER

Address Offset	0x60		
Physical address	0x4805 0C60	**Instance**	VENC
Description	VENC HTRIGGER and VTRIGGER		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								VTRIG								Reserved								HTRIG							

Bits	Field Name	Description	Type	Reset
31:26	Reserved	Reserved. Read returns 0s.	RW	0x00
25:16	VTRIG	Vertical trigger reference for VSYNC. These bits specify the phase between VSYNC input and the lines in a field. The VTRIG field is expressed in units of half-line.	RW	0x000
15:11	Reserved	Reserved. Read returns 0s.	RW	0x00
10:0	HTRIG	Horizontal trigger phase, which sets HSYNC. HTRIG is expressed in half-pixels or clk2x (27 MHz) periods	RW	0x000

Table 15-342. Register Call Summary for Register VENC_HTRIGGER_VTRIGGER

Display Subsystem Basic Programming Model • Video Encoder Register Settings: [0]
Display Subsystem Register Manual • Display Subsystem Register Mapping Summary: [1]

Table 15-343. VENC_SAVID_EAVID

Address Offset	0x64		
Physical address	0x4805 0C64	**Instance**	VENC
Description	VENC SAVID and EAVID		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								EAVID								Reserved								SAVID							

Bits	Field Name	Description	Type	Reset
31:27	Reserved	Reserved. Read returns 0s.	RW	0x00
26:16	EAVID	End of active video. These bits define the ending pixel position on a horizontal display line where active video will be displayed.	RW	0x693
15:11	Reserved	Reserved. Read returns 0s.	RW	0x00
10:0	SAVID	Start of active video. These bits define the starting pixel position on a horizontal line where active video will be displayed.	RW	0x0F4

Table 15-344. Register Call Summary for Register VENC_SAVID_EAVID

Display Subsystem Basic Programming Model
- Video Encoder Register Settings: [0]

Display Subsystem Register Manual
- Display Subsystem Register Mapping Summary: [1]
- Video Encoder Registers: [2] [3]

Table 15-345. VENC_FLEN_FAL

Address Offset	0x68		
Physical address	0x4805 0C68	**Instance**	VENC
Description	VENC FLEN and FAL		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								FAL								Reserved								FLEN							

Bits	Field Name	Description	Type	Reset
31:25	Reserved	Reserved. Read returns 0s.	RW	0x00
24:16	FAL	First Active Line of Field. These bits define the first active line of a field	RW	0x016
15:10	Reserved	Reserved. Read returns 0s.	RW	0x00
9:0	FLEN	Field length. These bits define the number of half_lines in each field. Length of field = (FLEN + 1) half_lines	RW	0x20C

Table 15-346. Register Call Summary for Register VENC_FLEN_FAL

Display Subsystem Basic Programming Model
- Video Encoder Register Settings: [0]

Display Subsystem Register Manual
- Display Subsystem Register Mapping Summary: [1]
- Video Encoder Registers: [2]

Table 15-347. VENC_LAL_PHASE_RESET

Address Offset	0x6C		
Physical address	0x4805 0C6C	**Instance**	VENC
Description	VENC LAL and PHASE_RESET		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved													PRES		SBLANK	Reserved								LAL							

Bits	Field Name	Description	Type	Reset
31:19	Reserved	Reserved. Read returns 0s.	RW	0x0000
18:17	PRES	Phase reset mode. 0x0: No reset 0x1: Reset every two lines 0x2: Reset every eight fields. Color subcarrier phase is reset to VENC_CPHASE[7:0] CPHS field value (offset 0x2C) upon reset 0x3: Reset every four fields. Color subcarrier phase is reset to VENC_CPHASE[7:0] CPHS bit field value (offset 0x2C) upon reset	RW	0x3
16	SBLANK	Data output enable	RW	0

Bits	Field Name	Description	Type	Reset
		0x0: No functionality 0x1: Enables the output of data when VENC_SYNC_CTRL[10] VBLMK bit is '0b1'.		
15:9	Reserved	Reserved. Read returns 0s.	RW	0x00
8:0	LAL	Last Active Line of Field. These bits define the last active line of a field. The LAL[8:0] bit field value must be set to a value lower than the active window.	RW	0x107

Table 15-348. Register Call Summary for Register VENC_LAL_PHASE_RESET

Display Subsystem Basic Programming Model
- Video Encoder Register Settings: [0]

Display Subsystem Register Manual
- Display Subsystem Register Mapping Summary: [1]
- Video Encoder Registers: [2] [3]

Table 15-349. VENC_HS_INT_START_STOP_X

Address Offset	0x70		
Physical address	0x4805 0C70	**Instance**	VENC
Description	VENC_HS_INT_START_STOP_X		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								HS_INT_STOP_X								Reserved								HS_INT_START_X							

Bits	Field Name	Description	Type	Reset
31:26	Reserved	Reserved. Read returns 0s.	RW	0x00
25:16	HS_INT_STOP_X	HSYNC internal stop. These bits define HSYNC internal stop pixel value	RW	0x07E
15:10	Reserved	Reserved. Read returns 0s.	RW	0x00
9:0	HS_INT_START_X	HSYNC internal start. These bits define HSYNCI NTERNAL start pixel value	RW	0x34E

Table 15-350. Register Call Summary for Register VENC_HS_INT_START_STOP_X

Display Subsystem Basic Programming Model
- Video Encoder Register Settings: [0]

Display Subsystem Register Manual
- Display Subsystem Register Mapping Summary: [1]
- Video Encoder Registers: [2]

Table 15-351. VENC_HS_EXT_START_STOP_X

Address Offset	0x74		
Physical address	0x4805 0C74	**Instance**	VENC
Description	VENC_HS_EXT_START_STOP_X		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								HS_EXT_STOP_X								Reserved								HS_EXT_START_X							

Bits	Field Name	Description	Type	Reset
31:26	Reserved	Reserved. Read returns 0s.	RW	0x00
25:16	HS_EXT_STOP_X	HSYNC external stop. These bits define HSYNC external stop pixel value	RW	0x00F
15:10	Reserved	Reserved. Read returns 0s.	RW	0x00
9:0	HS_EXT_START_X	HSYNC external start. These bits define HSYNC EXTERNAL start pixel value	RW	0x359

Table 15-352. Register Call Summary for Register VENC_HS_EXT_START_STOP_X

Display Subsystem Basic Programming Model
- Video Encoder Register Settings: [0]

Display Subsystem Register Manual
- Display Subsystem Register Mapping Summary: [1]
- Video Encoder Registers: [2]

Table 15-353. VENC_VS_INT_START_X

Address Offset	0x78		
Physical address	0x4805 0C78	**Instance**	VENC
Description	VENC_VS_INT_START_X		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								VS_INT_START_X								Reserved															

Bits	Field Name	Description	Type	Reset
31:26	Reserved	Reserved. Read returns 0s.	RW	0x00
25:16	VS_INT_START_X	VSYNC internal start. These bits define VSYNC internal start pixel value.	RW	0x1A0
15:0	Reserved	Reserved. Read returns 0s.	RW	0x0000

Table 15-354. Register Call Summary for Register VENC_VS_INT_START_X

Display Subsystem Basic Programming Model
- Video Encoder Register Settings: [0]

Display Subsystem Register Manual
- Display Subsystem Register Mapping Summary: [1]
- Video Encoder Registers: [2]

Table 15-355. VENC_VS_INT_STOP_X_VS_INT_START_Y

Address Offset	0x7C		
Physical address	0x4805 0C7C	**Instance**	VENC
Description	VENC VS_INT_STOP_X and VS_INT_START_Y		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								VS_INT_START_Y								Reserved								VS_INT_STOP_X							

Bits	Field Name	Description	Type	Reset
31:26	Reserved	Reserved. Read returns 0s.	RW	0x00
25:16	VS_INT_START_Y	VSYNC internal start. These bits define VSYNC INTERNAL start line value	RW	0x209
15:10	Reserved	Reserved. Read returns 0s.	RW	0x00

Bits	Field Name	Description	Type	Reset
9:0	VS_INT_STOP_X	VSYNC internal stop. These bits define VSYNC internal stop pixel value	RW	0x1A0

Table 15-356. Register Call Summary for Register VENC_VS_INT_STOP_X_VS_INT_START_Y

Display Subsystem Basic Programming Model
- Video Encoder Register Settings: [0]

Display Subsystem Register Manual
- Display Subsystem Register Mapping Summary: [1]

Table 15-357. VENC_VS_INT_STOP_Y_VS_EXT_START_X

Address Offset	0x80		
Physical address	0x4805 0C80	**Instance**	VENC
Description	VENC VS_INT_STOP_Y and VS_EXT_START_X		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								VS_EXT_START_X								Reserved								VS_INT_STOP_Y							

Bits	Field Name	Description	Type	Reset
31:26	Reserved	Reserved. Read returns 0s.	RW	0x00
25:16	VS_EXT_START_X	VSYNC external start. These bits define VSYNC external start pixel value.	RW	0x1AC
15:10	Reserved	Reserved. Read returns 0s.	RW	0x00
9:0	VS_INT_STOP_Y	VSYNC internal stop. These bits define VSYNC INTERNAL stop line value.	RW	0x022

Table 15-358. Register Call Summary for Register VENC_VS_INT_STOP_Y_VS_EXT_START_X

Display Subsystem Basic Programming Model
- Video Encoder Register Settings: [0]

Display Subsystem Register Manual
- Display Subsystem Register Mapping Summary: [1]

Table 15-359. VENC_VS_EXT_STOP_X_VS_EXT_START_Y

Address Offset	0x84		
Physical address	0x4805 0C84	**Instance**	VENC
Description	VENC VS_EXT_STOP_X and VS_EXT_START_Y		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								VS_EXT_START_Y								Reserved								VS_EXT_STOP_X							

Bits	Field Name	Description	Type	Reset
31:26	Reserved	Reserved. Read returns 0s.	RW	0x00
25:16	VS_EXT_START_Y	VSYNC external start. These bits define VSYNC EXTERNAL start line value.	RW	0x20D
15:10	Reserved	Reserved. Read returns 0s.	RW	0x00
9:0	VS_EXT_STOP_X	VSYNC external stop. These bits define VSYNC EXTERNAL stop pixel value.	RW	0x1AC

Table 15-360. Register Call Summary for Register VENC_VS_EXT_STOP_X_VS_EXT_START_Y

Display Subsystem Basic Programming Model
- Video Encoder Register Settings: [0]

Display Subsystem Register Manual
- Display Subsystem Register Mapping Summary: [1]

Table 15-361. VENC_VS_EXT_STOP_Y

Address Offset	0x88		
Physical address	0x4805 0C88	**Instance**	VENC
Description	VENC VS_EXT_STOP_Y		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																						VS_EXT_STOP_Y									

Bits	Field Name	Description	Type	Reset
31:10	Reserved	Reserved. Read returns 0s.	RW	0x000000
9:0	VS_EXT_STOP_Y	VSYNC external stop. These bits define VSYNC EXTERNAL stop line value.	RW	0x006

Table 15-362. Register Call Summary for Register VENC_VS_EXT_STOP_Y

Display Subsystem Basic Programming Model
- Video Encoder Register Settings: [0]

Display Subsystem Register Manual
- Display Subsystem Register Mapping Summary: [1]

Table 15-363. VENC_AVID_START_STOP_X

Address Offset	0x90		
Physical address	0x4805 0C90	**Instance**	VENC
Description	VENC_AVID_START_STOP_X		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								AVID_STOP_X								Reserved								AVID_START_X							

Bits	Field Name	Description	Type	Reset
31:26	Reserved	Reserved. Read returns 0s.	RW	0x00
25:16	AVID_STOP_X	AVID stop. These bits define AVID stop pixel value	RW	0x348
15:10	Reserved	Reserved. Read returns 0s.	RW	0x00
9:0	AVID_START_X	AVID start. These bits define AVID start pixel value	RW	0x078

Table 15-364. Register Call Summary for Register VENC_AVID_START_STOP_X

Display Subsystem Basic Programming Model
- Video Encoder Register Settings: [0]

Display Subsystem Register Manual
- Display Subsystem Register Mapping Summary: [1]
- Video Encoder Registers: [2] [3]

Table 15-365. VENC_AVID_START_STOP_Y

Address Offset	0x94		
Physical address	0x4805 0C94	**Instance**	VENC
Description	VENC_AVID_START_STOP_Y		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								AVID_STOP_Y								Reserved								AVID_START_Y							

Bits	Field Name	Description	Type	Reset
31:26	Reserved	Reserved. Read returns 0s.	RW	0x00
25:16	AVID_STOP_Y	AVID stop. These bits define AVID stop line value.	RW	0x206
15:10	Reserved	Reserved. Read returns 0s.	RW	0x00
9:0	AVID_START_Y	AVID start. These bits define AVID start line value	RW	0x026

Table 15-366. Register Call Summary for Register VENC_AVID_START_STOP_Y

Display Subsystem Basic Programming Model

- Video Encoder Register Settings: [0]

Display Subsystem Register Manual

- Display Subsystem Register Mapping Summary: [1]
- Video Encoder Registers: [2] [3]

Table 15-367. VENC_FID_INT_START_X_FID_INT_START_Y

Address Offset	0xA0		
Physical address	0x4805 0CA0	**Instance**	VENC
Description	VENC_FID_INT_START_X and FID_INT_START_Y		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								FID_INT_START_Y								Reserved								FID_INT_START_X							

Bits	Field Name	Description	Type	Reset
31:26	Reserved	Reserved. Read returns 0s.	RW	0x00
25:16	FID_INT_START_Y	FID internal start. These bits define FID internal start line value	RW	0x001
15:10	Reserved	Reserved. Read returns 0s.	RW	0x00
9:0	FID_INT_START_X	FID internal start. These bits define FID internal start pixel value	RW	0x08A

Table 15-368. Register Call Summary for Register VENC_FID_INT_START_X_FID_INT_START_Y

Display Subsystem Basic Programming Model

- Video Encoder Register Settings: [0]

Display Subsystem Register Manual

- Display Subsystem Register Mapping Summary: [1]

Table 15-369. VENC_FID_INT_OFFSET_Y_FID_EXT_START_X

Address Offset	0xA4		
Physical address	0x4805 0CA4	**Instance**	VENC
Description	VENC FID_INT_OFFSET_Y and FID_EXT_START_X		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								FID_EXT_START_X								Reserved								FID_INT_OFFSET_Y							

Bits	Field Name	Description	Type	Reset
31:26	Reserved	Reserved. Read returns 0s.	RW	0x00
25:16	FID_EXT_START_X	FID external start. These bits define FID external start pixel value	RW	0x1AC
15:10	Reserved	Reserved. Read returns 0s.	RW	0x00
9:0	FID_INT_OFFSET_Y	FID internal offset. These bits define FID internal offset linel value	RW	0x106

Table 15-370. Register Call Summary for Register VENC_FID_INT_OFFSET_Y_FID_EXT_START_X

Display Subsystem Basic Programming Model
- Video Encoder Register Settings: [0]

Display Subsystem Register Manual
- Display Subsystem Register Mapping Summary: [1]

Table 15-371. VENC_FID_EXT_START_Y_FID_EXT_OFFSET_Y

Address Offset	0xA8		
Physical address	0x4805 0CA8	**Instance**	VENC
Description	VENC FID_EXT_START_Y and FID_EXT_OFFSET_Y		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								FID_EXT_OFFSET_Y								Reserved								FID_EXT_START_Y							

Bits	Field Name	Description	Type	Reset
31:26	Reserved	Reserved. Read returns 0s.	RW	0x00
25:16	FID_EXT_OFFSET_Y	FID external offset. These bits define FID external offset line value	RW	0x106
15:10	Reserved	Reserved. Read returns 0s.	RW	0x00
9:0	FID_EXT_START_Y	FID external start. These bits define FID external start line value.	RW	0x006

Table 15-372. Register Call Summary for Register VENC_FID_EXT_START_Y_FID_EXT_OFFSET_Y

Display Subsystem Basic Programming Model
- Video Encoder Register Settings: [0]

Display Subsystem Register Manual
- Display Subsystem Register Mapping Summary: [1]

Table 15-373. VENC_TVDETGP_INT_START_STOP_X

Address Offset	0xB0		
Physical address	0x4805 0CB0	**Instance**	VENC
Description	TV Detection Start and Stop pixel values		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								TVDETGP_INT_STOP_X								Reserved								TVDETGP_INT_START_X							

Bits	Field Name	Description	Type	Reset
31:26	Reserved	Reserved. Read returns 0s.	RW	0x00
25:16	TVDETGP_INT_STOP_X	TVDETGP internal stop. These bits define TVDETGP internal stop pixel value.	RW	0x014
15:10	Reserved	Reserved. Read returns 0s.	RW	0x00
9:0	TVDETGP_INT_START_X	TVDETGP internal start. These bits define TVDETGP internal start pixel value	RW	0x001

Table 15-374. Register Call Summary for Register VENC_TVDETGP_INT_START_STOP_X

Display Subsystem Functional Description
- TV Detection/Disconnection Pulse Generation and Usage: [0] [1] [2] [3] [4] [5] [6] [7]

Display Subsystem Basic Programming Model
- Video Encoder Register Settings: [8]

Display Subsystem Register Manual
- Display Subsystem Register Mapping Summary: [9]

Table 15-375. VENC_TVDETGP_INT_START_STOP_Y

Address Offset	0xB4		
Physical address	0x4805 0CB4	**Instance**	VENC
Description	TV detection Start and Stop line values		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								TVDETGP_INT_STOP_Y								Reserved								TVDETGP_INT_START_Y							

Bits	Field Name	Description	Type	Reset
31:26	Reserved	Reserved. Read returns 0s.	RW	0x00
25:16	TVDETGP_INT_STOP_Y	TVDETGP internal stop. These bits define TVDETGP internal stop line value.	RW	0x001
15:10	Reserved	Reserved. Read returns 0s.	RW	0x00
9:0	TVDETGP_INT_START_Y	TVDETGP internal start. These bits define TVDETGP internal start line value.	RW	0x001

Table 15-376. Register Call Summary for Register VENC_TVDETGP_INT_START_STOP_Y

Display Subsystem Functional Description
- TV Detection/Disconnection Pulse Generation and Usage: [0] [1] [2] [3] [4] [5] [6]

Display Subsystem Basic Programming Model
- Video Encoder Register Settings: [7]

Display Subsystem Register Manual
- Display Subsystem Register Mapping Summary: [8]

Table 15-377. VENC_GEN_CTRL

Address Offset	0xB8		
Physical address	0x4805 0CB8	**Instance**	VENC
Description	TVDETGP enable and SYNC_POLARITY and UVPHASE_POL		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved					MS	656	CBAR	HIP	VIP	HEP	VEP	AVIDP	FIP	FEP	TVDP	Reserved															EN

Bits	Field Name	Description	Type	Reset
31:27	Reserved	Reserved. Read returns 0s.	RW	0x0000
26	MS	UVPHASE_POL MS mode UV phase	RW	0
		0x0: CbCr		
		0x1: CrCb		
25	656	UVPHASE_POL 656 input mode UV phase	RW	0
		0x0: CbCr		
		0x1: CrCb		
24	CBAR	UVPHASE_POL CBAR mode UV phase	RW	0
		0x0: CbCr		
		0x1: CrCb		
23	HIP	HSYNC internal polarity	RW	1
		0x0: Active low		
		0x1: Active high		
22	VIP	VSYNC internal polarity	RW	1
		0x0: Active low		
		0x1: Active high		
21	HEP	HSYNC external polarity	RW	1
		0x0: Active low		
		0x1: Active high		
20	VEP	VSYNC external polarity	RW	1
		0x0: Active low		
		0x1: Active high		
19	AVIDP	AVID polarity	RW	1
		0x0: Active low		
		0x1: Active high		
18	FIP	FID internal polarity	RW	1
		0x0: Active low		
		0x1: Active high		
17	FEP	FID external polarity	RW	1
		0x0: Active low		
		0x1: Active high		
16	TVDP	TVDETGP polarity	RW	1
		0x0: Active low		
		0x1: Active high		
15:1	Reserved	Reserved. Read returns 0s.	RW	0x00
0	EN	TVDETGP generation enable	RW	0
		0x0: Disabled		
		0x1: Enabled		

Table 15-378. Register Call Summary for Register VENC_GEN_CTRL

Display Subsystem Functional Description • TV Detection/Disconnection Pulse Generation and Usage: [0] [1] [2] [3] [4] [5]
Display Subsystem Basic Programming Model • Video Encoder Register Settings: [6]
Display Subsystem Register Manual • Display Subsystem Register Mapping Summary: [7]

Table 15-379. VENC_OUTPUT_CONTROL

Address Offset	0x0000 00C4		
Physical Address	0x4805 0CC4	**Instance**	VENC
Description	Output channel control register Also contains some test control features		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								LUMA_TEST								RESERVED								CHROMA_SOURCE	COMPOSITE_SOURCE	LUMA_SOURCE	TEST_MODE	VIDEO_INVERT	CHROMA_ENABLE	COMPOSITE_ENABLE	LUMA_ENABLE

Bits	Field Name	Description	Type	Reset
31:26	RESERVED	Reserved. Read returns 0s.	RW	0x00
25:16	LUMA_TEST	In test mode, DAC 1 input value (if s-video video mode is selected)	RW	0x000
15:8	RESERVED	Reserved. Read returns 0s.	RW	0x00
7	CHROMA_SOURCE	Source of chroma video data in test mode 0x0: Chroma test data comes from internal register OUTPUT_TEST[25:16] 0x1: Chroma test data comes from display controller video port G[1:0], B[7:0]	RW	0x0
6	COMPOSITE_ SOURCE	Source of composite video data in test mode 0x0: Composite test data comes from internal register OUTPUT_TEST[9:0] 0x1: Composite test data comes from display controller video port G[1:0], B[7:0]	RW	0x0
5	LUMA_SOURCE	Source of luminance video data in test mode 0x0: Luma test data comes from internal register OUTPUT_CONTROL[25:16] 0x1: Luma test data comes from display controller video port G[1:0], B[7:0]	RW	0x0
4	TEST_MODE	This enables the video DACs to be tested. The values sent to the DACs comes from a register for each output channel (Luma, Composite or Chroma) or from the display controller video port bits G[1:0], B[7:0], depending on the setting of the Source bits 0x0: Video outputs are in normal operation 0x1: Test mode. Video outputs are directly connected to either internal registers or the display controller video port.	RW	0x0

Bits	Field Name	Description	Type	Reset
3	VIDEO_INVERT	Controls the video output polarity. This may be used to correct for inversion in an external video amplifier. 0x0: Video outputs are inverted 0x1: Video outputs are normal polarity	RW	0x1
2	CHROMA_ENABLE	Enable the Chrominance output channel 0x0: Chroma output is disabled 0x1: Chroma output is enabled	RW	0x0
1	COMPOSITE_ ENABLE	Enable the Composite output channel 0x0: Composite output is disabled 0x1: Composite output is enabled	RW	0x0
0	LUMA_ENABLE	Enable the Luminance output channel 0x0: Luma output is disabled 0x1: Luma output is enabled	RW	0x0

Table 15-380. Register Call Summary for Register VENC_OUTPUT_CONTROL

Display Subsystem Environment
- TV Display Support: [0] [1]

Display Subsystem Functional Description
- TV Detection/Disconnection Pulse Generation and Usage: [2] [3] [4] [5]
- Video Dual-DAC Test Mode: [6] [7] [8] [9] [10]

Display Subsystem Basic Programming Model
- Video DAC Settings: [11] [12] [13]
- Video Encoder Register Settings: [14]

Display Subsystem Register Manual
- Display Subsystem Register Mapping Summary: [15]

Table 15-381. VENC_OUTPUT_TEST

Address Offset	0x0000 00C8		
Physical Address	0x4805 0CC8	**Instance**	VENC
Description	Test values for the Luma/Composite Video DAC1 (if composite video is selected) and the Chroma Video DAC2		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								CHROMA_TEST								RESERVED								COMPOSITE_TEST							

Bits	Field Name	Description	Type	Reset
31:26	RESERVED	Reserved. Read returns 0s.	RW	0x00
25:16	CHROMA_TEST	In test mode, DAC 2 input value	RW	0x000
15:10	RESERVED	Reserved. Read returns 0s.	RW	0x00
9:0	COMPOSITE_TEST	In test mode, DAC 1 input value (if composite video is selected)	RW	0x000

Table 15-382. Register Call Summary for Register VENC_OUTPUT_TEST

Display Subsystem Functional Description
- Video Dual-DAC Test Mode: [0] [1]

Display Subsystem Basic Programming Model
- Video Encoder Register Settings: [2]

Display Subsystem Register Manual
- Display Subsystem Register Mapping Summary: [3]

15.7.2.5 DSI Protocol Engine Registers

Table 15-383. DSI_REVISION

Address Offset	0x0000 0000		
Physical Address	0x4804 FC00	**Instance**	DSI_PROTOCOL_ENGINE
Description	MODULE REVISION This register contains the IP revision code in binary coded digital. For example, we have: 0x01 = revision 0.1 and 0x21 = revision 2.1		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																								REV							

Bits	Field Name	Description	Type	Reset
31:8	RESERVED	Write 0s for future compatibility. Reads returns 0.	R	0x000000
7:0	REV	IP revision [7:4] Major revision [3:0] Minor revision	R	TI internal data

Table 15-384. Register Call Summary for Register DSI_REVISION

Display Subsystem Register Manual

- Display Subsystem Register Mapping Summary: [0]

Table 15-385. DSI_SYSCONFIG

Address Offset	0x0000 0010		
Physical Address	0x4804 FC10	**Instance**	DSI_PROTOCOL_ENGINE
Description	SYSTEM CONFIGURATION REGISTER This register is the system configuration register.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																						RESERVED		CLOCKACTIVITY		RESERVED				SIDLEMODE		ENWAKEUP		SOFT_RESET		AUTO_IDLE	

Bits	Field Name	Description	Type	Reset
31:14	RESERVED	Write 0s for future compatibility. Reads returns 0.	RW	0x00000
13:10	RESERVED	Write 0s for future compatibility.	RW	0x0
9:8	CLOCKACTIVITY	Clocks activity during wake up mode period	RW	0x0
		0x0: Interface and Functional clocks can be switched off		
		0x1: Functional clocks can be switched off and Interface clocks are maintained during wake up period		
		0x2: Interface clocks can be switched off and Functional clocks are maintained during wake up period		
		0x3: Interface and Functional clocks are maintained during wake up period		
7:5	RESERVED	Write 0s for future compatibility. Reads returns 0.	RW	0x0

Bits	Field Name	Description	Type	Reset
4:3	SIDLEMODE	Slave interface power management, Idle req/ack control 0x0: Force-idle. An idle request is acknowledged unconditionally 0x1: No-idle. An idle request is never acknowledged 0x2: Smart-idle. Acknowledgement to an idle request is given based on the internal activity of the module. 0x3: Reserved	RW	0x2
2	ENWAKEUP	Wake-up mode enable bit 0x0: Wakeup is disabled. 0x1: Wakeup is enabled,	RW	0x0
1	SOFT_RESET	Software reset. Set the bit to 1 to trigger a module reset. The bit is automatically reset by the hw. During reads return 0. 0x0: Normal mode. 0x1: The module is reset	RW	0x0
0	AUTO_IDLE	Internal interface clock gating strategy 0x0: Interface clock is free-running. 0x1: Automatic Interface clock gating strategy is applied based on the module interface activity.	RW	0x1

Table 15-386. Register Call Summary for Register DSI_SYSCONFIG

Display Subsystem Integration
- Resets: [0]
- Power Management: [1] [2] [3] [4] [5] [6] [7]

Display Subsystem Basic Programming Model
- Software Reset: [8]
- Power Management: [9] [10]
- Software Reset: [11]

Display Subsystem Use Cases and Tips
- Configure DSI, DSI PLL and Complex I/O: [12]
- Display Subsystem Use Cases and Tips: [13] [14] [15]

Display Subsystem Register Manual
- Display Subsystem Register Mapping Summary: [16]

Table 15-387. DSI_SYSSTATUS

Address Offset	0x0000 0014		
Physical Address	0x4804 FC14	**Instance**	DSI_PROTOCOL_ENGINE
Description	SYSTEM STATUS REGISTER This register provides status information about the module, excluding the interrupt status register.		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																															RESET_DONE

Bits	Field Name	Description	Type	Reset
31:1	RESERVED	Reads returns 0.	R	0x00000000
0	RESET_DONE	Internal reset monitoring 0x0: Internal module reset is on going. 0x1: Reset completed.	R	0x1

Table 15-388. Register Call Summary for Register DSI_SYSSTATUS

Display Subsystem Basic Programming Model
- Software Reset: [0] [1]

Display Subsystem Use Cases and Tips
- Configure DSI, DSI PLL and Complex I/O: [2] [3]
- Display Subsystem Use Cases and Tips: [4] [5]

Display Subsystem Register Manual
- Display Subsystem Register Mapping Summary: [6]

Table 15-389. DSI_IRQSTATUS

Address Offset	0x0000 0018		
Physical Address	0x4804 FC18	**Instance**	DSI_PROTOCOL_ENGINE
Description	INTERRUPT STATUS REGISTER - All VCs + complex I/O + PLL This register associates one bit for each VC to determine which VC has generated the interrupt. The VC should be enabled for events to be generated on that VC. If the VC is disabled, the interrupt is not generated.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED											TA_TO_IRQ	LDO_POWER_GOOD_IRQ	SYNC_LOST_IRQ	ACK_TRIGGER_IRQ	TE_TRIGGER_IRQ	LP_RX_TO_IRQ	HS_TX_TO_IRQ	RESERVED	RESERVED		COMPLEXIO_ERR_IRQ	PLL_RECAL_IRQ	PLL_UNLOCK_IRQ	PLL_LOCK_IRQ	RESERVED	RESYNCHRONIZATION_IRQ	WAKEUP_IRQ	VIRTUAL_CHANNEL3_IRQ	VIRTUAL_CHANNEL2_IRQ	VIRTUAL_CHANNEL1_IRQ	VIRTUAL_CHANNEL0_IRQ

Bits	Field Name	Description	Type	Reset
31:21	RESERVED	Write 0s for future compatibility. Reads returns 0.	RW	0x000
20	TA_TO_IRQ	Turn-around Time out. 0x0: READS: Event is false. WRITES: Status bit unchanged. 0x1: READS: Event is true (pending). WRITES: Status bit is reset.	RW	0x0
19	LDO_POWER_GOOD_IRQ	Transition of the status signal LDOPWRGOOD from the DSI_PHY indicating a state change for the supply VDDALDODSIPLL from up to down or down to up. 0x0: READS: Event is false. WRITES: Status bit unchanged. 0x1: READS: Event is true (pending). WRITES: Status bit is reset.	RW	0x0
18	SYNC_LOST_IRQ	Synchronization with Video port is lost (Video mode only) 0x0: READS: Event is false. WRITES: Status bit unchanged. 0x1: READS: Event is true (pending). WRITES: Status bit is reset.	RW	0x0
17	ACK_TRIGGER_IRQ	Acknowledge Trigger 0x0: READS: Event is false. WRITES: Status bit unchanged. 0x1: READS: Event is true (pending). WRITES: Status bit is reset.	RW	0x0

Bits	Field Name	Description	Type	Reset
16	TE_TRIGGER_IRQ	Tearing Effect Trigger 0x0: READS: Event is false. WRITES: Status bit unchanged. 0x1: READS: Event is true (pending). WRITES: Status bit is reset.	RW	0x0
15	LP_RX_TO_IRQ	Interrupt for Low Power Rx Time out 0x0: READS: Event is false. WRITES: Status bit unchanged. 0x1: READS: Event is true (pending). WRITES: Status bit is reset.	RW	0x0
14	HS_TX_TO_IRQ	Interrupt for high-speed Tx Time out. 0x0: READS: Event is false. WRITES: Status bit unchanged. 0x1: READS: Event is true (pending). WRITES: Status bit is reset.	RW	0x0
13	RESERVED	Write 0s for future compatibility. Reads returns 0.	RW	0x0
12:11	RESERVED	Write 0s for future compatibility. Reads returns 0.	RW	0x0
10	COMPLEXIO_ ERR_IRQ	Error signaling from complex I/O: status of the complex I/O errors received from the complex I/O(events are defined in DSI_COMPLEXIO_IRQSTATUS). 0x0: READS: Event is false. 0x1: READS: Event is true (pending).	R	0x0
9	PLL_RECAL_IRQ	PLL recalibration event (assertion of recalibration signal from the DSI PLL Control module) 0x0: READS: Event is false. WRITES: Status bit unchanged. 0x1: READS: Event is true (pending). WRITES: Status bit is reset.	RW	0x0
8	PLL_UNLOCK_IRQ	PLL un-lock event (deassertion of lock signal from the DSI PLL Control module) 0x0: READS: Event is false. WRITES: Status bit unchanged. 0x1: READS: Event is true (pending). WRITES: Status bit is reset.	RW	0x0
7	PLL_LOCK_IRQ	PLL lock event (assertion of lock signal from the DSI PLL Control module) 0x0: READS: Event is false. WRITES: Status bit unchanged. 0x1: READS: Event is true (pending). WRITES: Status bit is reset.	RW	0x0
6	RESERVED	Write 0s for future compatibility. Reads returns 0.	RW	0x0
5	RESYNCHRONIZATION_ IRQ	Video mode resynchronization Indicates that the video port works but the configuration of the timings for the display controller (DISPC) and for DSI protocol engine may have to be modified to avoid the resynchronization to occur. 0x0: READS: Event is false. WRITES: Status bit unchanged. 0x1: READS: Event is true (pending). WRITES: Status bit is reset.	RW	0x0
4	WAKEUP_IRQ	Wakeup 0x0: READS: Event is false. WRITES: Status bit unchanged. 0x1: READS: Event is true (pending). WRITES: Status bit is reset.	RW	0x0

Bits	Field Name	Description	Type	Reset
3	VIRTUAL_CHANNEL3_IRQ	Virtual channel #3 Error signaling from DSI Virtual Channel3: Status of DSI Virtual Channel3 errors received from DSI Virtual Channel3 (events are defined in DSI_VC3_IRQSTATUS). 0x0: READS: Event is false. 0x1: READS: Event is true (pending).	R	0x0
2	VIRTUAL_CHANNEL2_IRQ	Virtual channel #2 Error signaling from DSI Virtual Channel2: Status of DSI Virtual Channel2 errors received from DSI Virtual Channel2 (events are defined in DSI_VC2_IRQSTATUS). 0x0: READS: Event is false. 0x1: READS: Event is true (pending).	R	0x0
1	VIRTUAL_CHANNEL1_IRQ	Virtual channel #1 Error signaling from DSI Virtual Channel1: Status of DSI Virtual Channel1 errors received from DSI Virtual Channel1 (events are defined in DSI_VC1_IRQSTATUS). 0x0: READS: Event is false. 0x1: READS: Event is true (pending).	R	0x0
0	VIRTUAL_CHANNEL0_IRQ	Virtual channel #0 Error signaling from DSI Virtual Channel0: Status of the DSI Virtual Channel0 errors received from DSI Virtual Channel0 (events are defined in DSI_VC0_IRQSTATUS). 0x0: READS: Event is false. 0x1: READS: Event is true (pending).	R	0x0

Table 15-390. Register Call Summary for Register DSI_IRQSTATUS

Display Subsystem Integration • Interrupt Requests: [0]
Display Subsystem Functional Description • Timers: [1] [2] • PHY Triggers: [3] [4] • Error Handling: [5] [6] [7]
Display Subsystem Basic Programming Model • Interrupts: [8] [9] • Video Mode: [10] [11] • DSI PLL Error Handling: [12] [13] [14] [15] [16] [17] • Error Handling: [18]
Display Subsystem Use Cases and Tips • Configure DSI, DSI PLL and Complex I/O: [19] • Display Subsystem Use Cases and Tips: [20]
Display Subsystem Register Manual • Display Subsystem Register Mapping Summary: [21] • Display Subsystem and SDI Registers: [22]

Table 15-391. DSI_IRQENABLE

Address Offset	0x0000 001C		
Physical Address	0x4804 FC1C	**Instance**	DSI_PROTOCOL_ENGINE
Description	INTERRUPT ENABLE REGISTER - This register associates one bit for each VC to enable/disable each VC individually.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED											TA_TO_IRQ_EN	LDO_POWER_GOOD_IRQ_EN	SYNC_LOST_IRQ_EN	ACK_TRIGGER_IRQ_EN	TE_TRIGGER_IRQ_EN	LP_RX_TO_IRQ_EN	HS_TX_TO_IRQ_EN	RESERVED	RESERVED	RESERVED	PLL_RECAL_IRQ_EN	PLL_UNLOCK_IRQ_EN	PLL_LOCK_IRQ_EN	RESERVED	RESYNCHRONIZATION_IRQ_EN	WAKEUP_IRQ_EN	RESERVED			

Bits	Field Name	Description	Type	Reset
31:21	RESERVED	Write 0s for future compatibility. Reads returns 0.	RW	0x000
20	TA_TO_IRQ_EN	Turn-around Time out. 0x0: Event is masked 0x1: Event generates an interrupt when it occurs	RW	0x0
19	LDO_POWER_GOOD_IRQ_EN	Transition of the status signal LDOPWRGOOD from the DSI_PHY indicating a state change for the supply VDDALDODSIPLL from up to down or down to up. 0x0: Event is masked 0x1: Event generates an interrupt when it occurs	RW	0x0
18	SYNC_LOST_IRQ_EN	Synchronization with Video port is lost (Video mode only) 0x0: Event is masked 0x1: Event generates an interrupt when it occurs	RW	0x0
17	ACK_TRIGGER_IRQ_EN	Acknowledge trigger 0x0: Event is masked 0x1: Event generates an interrupt when it occurs	RW	0x0
16	TE_TRIGGER_IRQ_EN	Tearing Effect trigger 0x0: Event is masked 0x1: Event generates an interrupt when it occurs	RW	0x0
15	LP_RX_TO_IRQ_EN	Interrupt for Low Power Rx Time out. 0x0: Event is masked 0x1: Event generates an interrupt when it occurs	RW	0x0
14	HS_TX_TO_IRQ_EN	Interrupt for high-speed Tx Time out. 0x0: Event is masked 0x1: Event generates an interrupt when it occurs	RW	0x0
13	RESERVED	Write 0s for future compatibility. Reads returns 0.	RW	0x0
12:11	RESERVED	Write 0s for future compatibility. Reads returns 0.	RW	0x0
10	RESERVED	Write 0s for future compatibility. Reads returns 0.	RW	0x0

Bits	Field Name	Description	Type	Reset
9	PLL_RECAL_IRQ_EN	PLL recalibration event (assertion of recalibration signal from the DSI PLL Control module) 0x0: Event is masked 0x1: Event generates an interrupt when it occurs	RW	0x0
8	PLL_UNLOCK_IRQ_EN	PLL un-lock event (deassertion of lock signal from the DSI PLL Control module) 0x0: Event is masked 0x1: Event generates an interrupt when it occurs	RW	0x0
7	PLL_LOCK_IRQ_EN	PLL lock event (assertion of lock signal from the DSI PLL Control module) 0x0: Event is masked 0x1: Event generates an interrupt when it occurs	RW	0x0
6	RESERVED	Write 0s for future compatibility. Reads returns 0.	RW	0x0
5	RESYNCHRONIZATION_IRQ_EN	Resynchronization 0x0: Event is masked 0x1: Event generates an interrupt when it occurs	RW	0x0
4	WAKEUP_IRQ_EN	Wakeup 0x0: Event is masked 0x1: Event generates an interrupt when it occurs	RW	0x0
3:0	RESERVED	Write 0s for future compatibility. Reads returns 0.	RW	0x0

Table 15-392. Register Call Summary for Register DSI_IRQENABLE

Display Subsystem Functional Description • PHY Triggers: [0] [1]
Display Subsystem Basic Programming Model • Interrupts: [2]
Display Subsystem Use Cases and Tips • Configure DSI, DSI PLL and Complex I/O: [3] • Display Subsystem Use Cases and Tips: [4]
Display Subsystem Register Manual • Display Subsystem Register Mapping Summary: [5]

Table 15-393. DSI_CTRL

Address Offset	0x0000 0040		
Physical Address	0x4804 FC40	**Instance**	DSI_PROTOCOL_ENGINE
Description	GLOBAL CONTROL REGISTER This register controls the DSI Protocol Engine module. This register should not be modified dynamically (except IF_EN bit field).		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED					RGB565_ORDER	DCS_CMD_CODE	DCS_CMD_ENABLE	HSA_BLANKING_MODE	HBP_BLANKING_MODE	HFP_BLANKING_MODE	BLANKING_MODE	EOT_ENABLE	VP_HSYNC_END	VP_HSYNC_START	VP_VSYNC_END	VP_VSYNC_START	TRIGGER_RESET_MODE	LINE_BUFFER		VP_VSYNC_POL	VP_HSYNC_POL	VP_DE_POL	VP_CLK_POL	VP_DATA_BUS_WIDTH		TRIGGER_RESET	VP_CLK_RATIO	TX_FIFO_ARBITRATION	ECC_RX_EN	CS_RX_EN	IF_EN

Bits	Field Name	Description	Type	Reset
31:27	RESERVED	Write 0s for future compatibility. Reads returns 0.	RW	0x00
26	RGB565_ORDER	Byte order for RBG565 command mode from video port 0x0: Unused 0x1: Byte order as for video mode	RW	0
25	DCS_CMD_CODE	DCS command code value to insert between header and video port data when enabled by DCS_CMD_ENABLE 0x0: DCS write memory continue code is inserted. 0x1: DCS write memory start code is inserted.	RW	0
24	DCS_CMD_ENABLE	Enables automatic insertion of DCS command codes when data is sourced by the video port. 0x0: DCS command code is not inserted when command mode traffic is coming from the Video Port. 0x1: DCS command code is inserted automatically when command mode traffic is coming from the Video Port.	RW	0
23	HSA_BLANKING_MODE	Blanking mode 0x0: Packets in TX FIFO are sent during HSA blanking period of video mode or LPS is used. 0x1: LONG BLANKING PACKETS only are used during HSA blanking period of video mode.	RW	0x0
22	HBP_BLANKING_MODE	Blanking mode 0x0: Packets in TX FIFO are sent during HBP blanking period of video mode or LPS is used. 0x1: LONG BLANKING PACKETS only are used during HBP blanking period of video mode.	RW	0x0
21	HFP_BLANKING_MODE	Blanking mode 0x0: Packets in TX FIFO are sent during HFP blanking period of video mode or LPS is used. 0x1: LONG BLANKING PACKETS only are used during HFP blanking period of video mode.	RW	0x0
20	BLANKING_MODE	Blanking mode 0x0: LPS is used during blanking periods of video mode (except HSA, HBP, HFP defined in HSA_BLANKING_MODE, HBP_BLANKING_MODE and HFP_BLANKING_MODE bit fields respectively) when there is no command mode data in TX FIFO ready to be sent. So blanking periods can be different during the frame depending on the TX FIFO. 0x1: LONG BLANKING PACKETS are used during blanking periods of video mode (except HSA, HBP, HFP defined in HSA_BLANKING_MODE, HBP_BLANKING_MODE and HFP_BLANKING_MODE bit fields respectively) regardless of the packets present in the TX FIFO ready to be sent	RW	0x0
19	EOT_ENABLE	Enable EOT packets at the end of HS transmission. 0x0: No EOT packets 0x1: EOT packet is sent at all HS to LP transitions	RW	0
18	VP_HSYNC_END	HSYNC end pulse. 0x0: Disabled. No HSYNC END short packet is generated. 0x1: Enabled. While the HSYNC END pulse is detected, the associated short packet HSYNC END is generated.	RW	0x0
17	VP_HSYNC_START	HSYNC start pulse. 0x0: Disabled. No HSYNC START short packet is generated. 0x1: Enabled. While the HSYNC start pulse is detected, the associated short packet HSYNC START is generated.	RW	0x0

Bits	Field Name	Description	Type	Reset
16	VP_VSYNC_END	VSYNC end pulse. 0x0: Disabled. No VSYNC END short packet is generated. 0x1: Enabled. While the VSYNC END pulse is detected, the associated short packet VSYNC END is generated.	RW	0x0
15	VP_VSYNC_START	VSYNC start pulse. 0x0: Disabled. No VSYNC START short packet is generated. 0x1: Enabled. While the VSYNC START pulse is detected, the associated short packet VSYNC START is generated.	RW	0x0
14	TRIGGER_RESET_MODE	Selection of the trigger reset mode 0x0: Synchronized: the mode is only valid if there is VC using the video mode and it is active. The principle is to wait for the current video frame to be transferred on the link. Any data received after the VSYNC are ignored. 0x1: Immediate: all pending requests in TX FIFO are taken into account for transfer scheduling, the RX FIFO is ignored, and the data from video port are ignored as soon as possible. Only the current transfer on DSI link and already scheduled ones are transmitted. All the other transfers are discarded.	RW	0x0
13:12	LINE_BUFFER	Number of line buffers to be used while receiving data on the video port. 0x0: No line buffer 0x1: 1 line buffer 0x2: 2 line buffers	RW	0x0
11	VP_VSYNC_POL	VP vertical synchronization signal polarity 0x0: VSYNC signal on the video port is active low. 0x1: VSYNC signal on the video port is active high.	RW	0x0
10	VP_HSYNC_POL	VP horizontal synchronization signal polarity 0x0: HSYNC signal on the video port is active low. 0x1: HSYNC signal on the video port is active high.	RW	0x0
9	VP_DE_POL	VP data enable signal polarity 0x0: DE signal on the video port is active low. 0x1: DE signal on the video port is active high.	RW	0x0
8	VP_CLK_POL	VP pixel clock polarity 0x0: The DSI Protocol Engine module captures the data on the VP on the pixel clock falling edge. The module connected to the VP must drive the data on the pixel clock rising edge. 0x1: The DSI Protocol Engine module captures the data on the VP on the pixel clock raising edge. The module connected to the VP must drive the data on the pixel clock falling edge.	RW	0x1
7:6	VP_DATA_BUS_WIDTH	Defines the size of the video port data bus 0x0: 16-bits data width (LSB of the 24-bit video port data bus) 0x1: 18-bits data width (LSB of the 24-bit video port data bus) 0x2: 24-bits data width (LSB of the 24-bit video port data bus)	RW	0x0
5	TRIGGER_RESET	Send the reset trigger to the peripheral. 0x0: READS: Reset trigger generation is completed. It is reset by HW when it is completed. WRITES: Cancellation of the request for Reset trigger generation (maybe too late since it is already on going) 0x1: READS: Generation of the reset trigger has been requested by user (could be on going but not completed yet). WRITES: Request for Reset trigger to be sent to the peripheral.	RW	0x0

Bits	Field Name	Description	Type	Reset
4	VP_CLK_RATIO	This bit indicates the clock ratio between VP_CLK and VP_PCLK. The clock VP_PCLK is generated from VP_CLK. It is divided down. The information is only used when the video port is used to provide data in command mode. In the case of video mode, it is not used. 0x0: The clock VP_PCLK is the clock VP_CLK divided by 2. The duty cycle of VP_PCLK is 50/50. 0x1: The clock VP_PCLK is the clock VP_CLK divided by 3 or more. The duty cycle of VP_PCLK is not 50/50 for odd ratio numbers (3,5,7,...).	RW	0x0
3	TX_FIFO_ ARBITRATION	Defines the arbitration scheme for granting the VC pending ready requests in the TX FIFO 0x0: Round-Robin Scheme is used 0x1: Sequential Scheme is used	RW	0x0
2	ECC_RX_EN	Enables the ECC check for the received header (short and long packets for all VC IDs). 0x0: Disabled 0x1: Enabled	RW	0x0
1	CS_RX_EN	Enables the checksum check for the received payload (long packet only for all VC IDs). 0x0: Disabled 0x1: Enabled	RW	0x0
0	IF_EN	Enables the module. When the module is disabled the signals from the complex I/O are gated (no updates of the interrupt status register). It is not possible to change the bit fields in the DSI_CTRL register, except IF_EN when it is enabled. All the other registers can be changed except the ones that require DSI_VCn_CTRL[0] VC_EN to be equal to 0 to be modified. 0x0: The interface is disabled. If one of the VC uses the video mode with the video port to receive the data, the DSI protocol engines is disabled when the next VSYNC is received and all the data in the FIFO for the other VCs in command mode are sent to the peripherals (if BTA_EN bit is enabled, the DSI protocol engine needs to wait for the response and BTA from the peripheral before disabling all the internal logic since an acknowledge is requested). 0x1: The interface is enabled immediately, the data acquisition on the video port starts on the next VSYNC (video mode) or first data received in the Slave port FIFO (command mode).	RW	0x0

Table 15-394. Register Call Summary for Register DSI_CTRL

Display Subsystem Environment

- Physical Layer: [0] [1] [2]
- Video Port (VP) Interface: [3] [4] [5] [6] [7] [8] [9] [10] [11]

Display Subsystem Functional Description

- Clock Requirements: [12]
- DSI Transfer Modes: [13] [14] [15] [16]
- Power Control: [17] [18] [19] [20]
- Timers: [21] [22] [23] [24] [25] [26]
- Bus Turnaround: [27] [28] [29] [30]
- PHY Triggers: [31] [32] [33] [34]
- ECC Generation: [35]
- Checksum Generation for Long Packet Payloads: [36]
- End of Transfer Packet: [37]

Table 15-394. Register Call Summary for Register DSI_CTRL (continued)

Display Subsystem Basic Programming Model

- Global Register Controls: [38] [39] [40] [41] [42] [43] [44]
- Packets: [45] [46] [47] [48] [49] [50]
- Video Mode: [51] [52] [53] [54]
- Video Port Data Bus: [55]
- Command Mode: [56]
- DSI Programming Sequence Example: [57] [58] [59]

Display Subsystem Use Cases and Tips

- Configure DSI, DSI PLL and Complex I/O: [60] [61] [62] [63] [64] [65] [66] [67] [68] [69] [70] [71] [72] [73]
- Enable Video Mode Using the DISPC Video Port: [74] [75]
- Display Subsystem Use Cases and Tips: [76] [77] [78] [79] [80] [81] [82] [83] [84] [85] [86] [87] [88]

Display Subsystem Register Manual

- Display Subsystem Register Mapping Summary: [89]
- DSI Protocol Engine Registers: [90] [91] [92] [93] [94] [95] [96] [97]

Table 15-395. DSI_COMPLEXIO_CFG1

Address Offset	0x0000 0048		
Physical Address	0x4804 FC48	**Instance**	DSI_PROTOCOL_ENGINE
Description	COMPLEXIO CONFIGURATION REGISTER for the complex I/O This register contains the lane configuration for the order and position of the lanes (clock and data) and the polarity order for the control of the PHY differential signals in addition to the control bit for the power FSM.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SHADOWING	GOBIT	RESET_DONE	PWR_CMD			PWR_STATUS		RESERVED		LDO_POWER_GOOD_STATE	USE_LDO_EXTERNAL	RESERVED		RESERVED		RESERVED		RESERVED		DATA2_POL		DATA2_POSITION		DATA1_POL		DATA1_POSITION		CLOCK_POL		CLOCK_POSITION	

Bits	Field Name	Description	Type	Reset
31	SHADOWING	Shadowing configuration. 0x0: Disabled. The writes to the DSIPHY_CFG0 and DSIPHY_CFG1 registers are done like the other SCP registers. 0x1: Enabled. The writes to the DSIPHY_CFG0 and DSIPHY_CFG1 registers are done only when the GO bit is set and when the signal DISPC_UPDATE_SYNC from the display controller module is active.	RW	0x0
30	GOBIT	Allows the synchronized update of the shadow registers when the signal DISPC_UPDATE_SYNC is active. 0x0: Resets the Gobit. The hardware has finished the update of the shadow SCP registers. The bit is reset by Hardware. The software can reset the bit in case users decide to abort it. There is no guarantee that the software reset is done before the transfer of the values to the complex I/O. 0x1: Set the Gobit. Only when the transfer of the new values for the two first registers is completed (2, 1, or 0 transfers are performed based on the number of registers to update), the GObit is reset. The DISPC_UPDATE_SYNC signal is used to synchronize the update. The bit must be set only when it is in reset state.	RW	0

Bits	Field Name	Description	Type	Reset
29	RESET_DONE	Internal reset monitoring of the power domain using TxByteClkHS from the complex I/O 0x0: Internal module reset is on going. 0x1: Reset completed.	R	1
28:27	PWR_CMD	Command for power control of the complex I/O 0x0: Command to change to OFF state 0x1: Command to change to ON state 0x2: Command to change to ultralow-power state	RW	0x0
26:25	PWR_STATUS	Status of the power control of the complex I/O 0x0: complex I/O in OFF state 0x1: complex I/O in ON state 0x2: complex I/O in ultralow-power state	R	0x0
24:22	RESERVED	Write 0s for future compatibility. Reads returns 0.	RW	0x0
21	LDO_POWER_GOOD_STATE	Indicates the state of the signal LDOPWRGOOD. VDDALDODSIPLL: 1.2-V power supply for the PLL. The voltage is supplied by the internal or external LDO. The interrupt LDO_POWER_GOOD_IRQ is generated when a transition is detected on the signal LDOPWRGOOD from the DSI_PHY. 0x0: VDDALDODSIPLL power supply is down 0x1: VDDALDODSIPLL power supply is up	R	0x0
20	USE_LDO_EXTERNAL	Select the external LDO for the DSI_PHY. 0x0: DSI_PHY internal LDO is used. 0x1: External LDO is used. DSI_PHY LDO is tri-stated.	RW	0x0
19	RESERVED	Write 0s for future compatibility. Reads returns 0.	RW	0x0
18:16	RESERVED	Write 0s for future compatibility. Reads returns 0.	RW	0x0
15	RESERVED	Write 0s for future compatibility. Reads returns 0.	RW	0x0
14:12	RESERVED	Write 0s for future compatibility. Reads returns 0.	RW	0x0
11	DATA2_POL	+/- differential pin order of DATA lane 2. 0x0: +/- pin order (dsi_dx=+ and dsi_dy=-) 0x1: -/+ pin order (dsi_dx=- and dsi_dy=+)	RW	0x0
10:8	DATA2_POSITION	Position and order of the DATA lane 2. 0x0: Not used/connected 0x1: Data lane 2 is at the position 1 (line 1). 0x2: Data lane 2 is at the position 2 (line 2). 0x3: Data lane 2 is at the position 3 (line 3). Other values: reserved	RW	0x0
7	DATA1_POL	+/- differential pin order of DATA lane 1 0x0: +/- pin order (dsi_dx=+ and dsi_dy=-) 0x1: -/+ pin order (dsi_dx=- and dsi_dy=+)	RW	0x0
6:4	DATA1_POSITION	Position and order of the DATA lane 1. 0x1: Data lane 1 is at the position 1 (line 1). 0x2: Data lane 1 is at the position 2 (line 2). 0x3: Data lane 1 is at the position 3 (line 3). Other values: reserved	RW	0x0
3	CLOCK_POL	+/- differential pin order of CLOCK lane. 0x0: +/- pin order (dsi_dx=+ and dsi_dy=-) 0x1: -/+ pin order (dsi_dx=- and dsi_dy=+)	RW	0x0

Bits	Field Name	Description	Type	Reset
2:0	CLOCK_POSITION	Position and order of the CLOCK lane. The clock lane is always present. 0x1: Clock lane is at the position 1 (line 1). 0x2: Clock lane is at the position 2 (line 2). 0x3: Clock lane is at the position 3 (line 3). Other values: reserved	RW	0x0

Table 15-396. Register Call Summary for Register DSI_COMPLEXIO_CFG1

Display Subsystem Environment
- Physical Layer: [0] [1]

Display Subsystem Functional Description
- DSI Protocol Architecture: [2]
- Serial Configuration Port (SCP) Interface: [3] [4] [5] [6] [7] [8] [9]
- Power Control: [10] [11]

Display Subsystem Basic Programming Model
- Ultra-Low Power State: [12] [13]
- Software Reset: [14]
- Pad Configuration: [15] [16] [17]

Display Subsystem Use Cases and Tips
- Configure DSI, DSI PLL and Complex I/O: [18] [19] [20]
- Display Subsystem Use Cases and Tips: [21] [22] [23] [24] [25]

Display Subsystem Register Manual
- Display Subsystem Register Mapping Summary: [26]

Table 15-397. DSI_COMPLEXIO_IRQSTATUS

Address Offset	0x0000 004C		
Physical Address	0x4804 FC4C	**Instance**	DSI_PROTOCOL_ENGINE
Description	INTERRUPT STATUS REGISTER - All errors from complex I/O		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ULPSACTIVENOT_ALL1_IRQ	ULPSACTIVENOT_ALL0_IRQ	RESERVED	RESERVED	RESERVED	RESERVED	ERRCONTENTIONLP1_3_IRQ	ERRCONTENTIONLP0_3_IRQ	ERRCONTENTIONLP1_2_IRQ	ERRCONTENTIONLP0_2_IRQ	ERRCONTENTIONLP1_1_IRQ	ERRCONTENTIONLP0_1_IRQ	RESERVED	RESERVED	STATEULPS3_IRQ	STATEULPS2_IRQ	STATEULPS1_IRQ	RESERVED	RESERVED	ERRCONTROL3_IRQ	ERRCONTROL2_IRQ	ERRCONTROL1_IRQ	RESERVED	RESERVED	ERRESC3_IRQ	ERRESC2_IRQ	ERRESC1_IRQ	RESERVED	RESERVED	ERRSYNCESC3_IRQ	ERRSYNCESC2_IRQ	ERRSYNCESC1_IRQ

Bits	Field Name	Description	Type	Reset
31	ULPSACTIVENOT_ ALL1_IRQ	All the ULPSActiveNOT signals corresponding to the lanes with TXULPSExit being high are high. 0x0: READS: Event is false. WRITES: Status bit unchanged. 0x1: READS: Event is true (pending). WRITES: Status bit is reset.	RW	0x0

Bits	Field Name	Description	Type	Reset
30	ULPSACTIVENOT_ALL0_IRQ	All signals ULPSActiveNOT are 0 0x0: READS: Event is false. WRITES: Status bit unchanged. 0x1: READS: Event is true (pending). WRITES: Status bit is reset.	RW	0x0
29	RESERVED	Write 0s for future compatibility. Reads returns 0.	RW	0x0
28	RESERVED	Write 0s for future compatibility. Reads returns 0.	RW	0x0
27	RESERVED	Write 0s for future compatibility. Reads returns 0.	RW	0x0
26	RESERVED	Write 0s for future compatibility. Reads returns 0.	RW	0x0
25	ERRCONTENTIONLP1_3_IRQ	Contention LP1 error for lane #3 0x0: READS: Event is false. WRITES: Status bit unchanged. 0x1: READS: Event is true (pending). WRITES: Status bit is reset.	RW	0x0
24	ERRCONTENTIONLP0_3_IRQ	Contention LP0 error for lane #3 0x0: READS: Event is false. WRITES: Status bit unchanged. 0x1: READS: Event is true (pending). WRITES: Status bit is reset.	RW	0x0
23	ERRCONTENTIONLP1_2_IRQ	Contention LP1 error for lane #2 0x0: READS: Event is false. WRITES: Status bit unchanged. 0x1: READS: Event is true (pending). WRITES: Status bit is reset.	RW	0x0
22	ERRCONTENTIONLP0_2_IRQ	Contention LP0 error for lane #2 0x0: READS: Event is false. WRITES: Status bit unchanged. 0x1: READS: Event is true (pending). WRITES: Status bit is reset.	RW	0x0
21	ERRCONTENTIONLP1_1_IRQ	Contention LP1 error for lane #1 0x0: READS: Event is false. WRITES: Status bit unchanged. 0x1: READS: Event is true (pending). WRITES: Status bit is reset.	RW	0x0
20	ERRCONTENTIONLP0_1_IRQ	Contention LP0 error for lane #1 0x0: READS: Event is false. WRITES: Status bit unchanged. 0x1: READS: Event is true (pending). WRITES: Status bit is reset.	RW	0x0
19	RESERVED	Write 0s for future compatibility. Reads returns 0.	RW	0x0
18	RESERVED	Write 0s for future compatibility. Reads returns 0.	RW	0x0
17	STATEULPS3_IRQ	Lane #3 in ultralow-power state 0x0: READS: Event is false. WRITES: Status bit unchanged. 0x1: READS: Event is true (pending). WRITES: Status bit is reset.	RW	0x0
16	STATEULPS2_IRQ	Lane #2 in ultralow-power state 0x0: READS: Event is false. WRITES: Status bit unchanged. 0x1: READS: Event is true (pending). WRITES: Status bit is reset.	RW	0x0

Bits	Field Name	Description	Type	Reset
15	STATEULPS1_IRQ	Lane #1 in ultralow-power state 0x0: READS: Event is false. WRITES: Status bit unchanged. 0x1: READS: Event is true (pending). WRITES: Status bit is reset.	RW	0x0
14	RESERVED	Write 0s for future compatibility. Reads returns 0.	RW	0x0
13	RESERVED	Write 0s for future compatibility. Reads returns 0.	RW	0x0
12	ERRCONTROL3_IRQ	Control error for lane #3 0x0: READS: Event is false. WRITES: Status bit unchanged. 0x1: READS: Event is true (pending). WRITES: Status bit is reset.	RW	0x0
11	ERRCONTROL2_IRQ	Control error for lane #2 0x0: READS: Event is false. WRITES: Status bit unchanged. 0x1: READS: Event is true (pending). WRITES: Status bit is reset.	RW	0x0
10	ERRCONTROL1_IRQ	Control error for lane #1 0x0: READS: Event is false. WRITES: Status bit unchanged. 0x1: READS: Event is true (pending). WRITES: Status bit is reset.	RW	0x0
9	RESERVED	Write 0s for future compatibility. Reads returns 0.	RW	0x0
8	RESERVED	Write 0s for future compatibility. Reads returns 0.	RW	0x0
7	ERRESC3_IRQ	Escape entry error for lane #3 0x0: READS: Event is false. WRITES: Status bit unchanged. 0x1: READS: Event is true (pending). WRITES: Status bit is reset.	RW	0x0
6	ERRESC2_IRQ	Escape entry error for lane #2 0x0: READS: Event is false. WRITES: Status bit unchanged. 0x1: READS: Event is true (pending). WRITES: Status bit is reset.	RW	0x0
5	ERRESC1_IRQ	Escape entry error for lane #1 0x0: READS: Event is false. WRITES: Status bit unchanged. 0x1: READS: Event is true (pending). WRITES: Status bit is reset.	RW	0x0
4	RESERVED	Write 0s for future compatibility. Reads returns 0.	RW	0x0
3	RESERVED	Write 0s for future compatibility. Reads returns 0.	RW	0x0
2	ERRSYNCESC3_IRQ	Low power Data transmission synchronization error for lane #3 0x0: READS: Event is false. WRITES: Status bit unchanged. 0x1: READS: Event is true (pending). WRITES: Status bit is reset.	RW	0x0
1	ERRSYNCESC2_IRQ	Low power Data transmission synchronization error for lane #2 0x0: READS: Event is false. WRITES: Status bit unchanged. 0x1: READS: Event is true (pending). WRITES: Status bit is reset.	RW	0x0

Bits	Field Name	Description	Type	Reset
0	ERRSYNCESC1_IRQ	Low power Data transmission synchronization error for lane #1 0x0: READS: Event is false. WRITES: Status bit unchanged. 0x1: READS: Event is true (pending). WRITES: Status bit is reset.	RW	0x0

Table 15-398. Register Call Summary for Register DSI_COMPLEXIO_IRQSTATUS

Display Subsystem Integration
- Interrupt Requests: [0] [1]

Display Subsystem Functional Description
- Timers: [2]

Display Subsystem Basic Programming Model
- Ultra-Low Power State: [3] [4]
- Error Handling: [5] [6] [7] [8] [9] [10] [11] [12] [13]

Display Subsystem Use Cases and Tips
- Configure DSI, DSI PLL and Complex I/O: [14]
- Display Subsystem Use Cases and Tips: [15]

Display Subsystem Register Manual
- Display Subsystem Register Mapping Summary: [16]
- DSI Protocol Engine Registers: [17]

Table 15-399. DSI_COMPLEXIO_IRQENABLE

Address Offset	0x0000 0050		
Physical Address	0x4804 FC50	**Instance**	DSI_PROTOCOL_ENGINE
Description	INTERRUPT ENABLE REGISTER - All errors from complex I/O		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ULPSACTIVENOT_ALL1_IRQ_EN	ULPSACTIVENOT_ALL0_IRQ_EN	RESERVED	RESERVED	RESERVED	RESERVED	ERRCONTENTIONLP1_3_IRQ_EN	ERRCONTENTIONLP0_3_IRQ_EN	ERRCONTENTIONLP1_2_IRQ_EN	ERRCONTENTIONLP0_2_IRQ_EN	ERRCONTENTIONLP1_1_IRQ_EN	ERRCONTENTIONLP0_1_IRQ_EN	RESERVED	RESERVED	STATEULPS3_IRQ_EN	STATEULPS2_IRQ_EN	STATEULPS1_IRQ_EN	RESERVED	RESERVED	ERRCONTROL3_IRQ_EN	ERRCONTROL2_IRQ_EN	ERRCONTROL1_IRQ_EN	RESERVED	RESERVED	ERRESC3_IRQ_EN	ERRESC2_IRQ_EN	ERRESC1_IRQ_EN	RESERVED	RESERVED	ERRSYNCESC3_IRQ_EN	ERRSYNCESC2_IRQ_EN	ERRSYNCESC1_IRQ_EN

Bits	Field Name	Description	Type	Reset
31	ULPSACTIVENOT_ ALL1_IRQ_EN	All the ULPSActiveNOT signals corresponding to the lanes with TXULPSExit being high are high. 0x0: Event is masked 0x1: Event generates an interrupt when it occurs	RW	0x0
30	ULPSACTIVENOT_ ALL0_IRQ_EN	All signals ULPSActiveNOT are 0 0x0: Event is masked 0x1: Event generates an interrupt when it occurs	RW	0x0
29	RESERVED	Write 0s for future compatibility. Reads returns 0.	RW	0x0
28	RESERVED	Write 0s for future compatibility. Reads returns 0.	RW	0x0

Bits	Field Name	Description	Type	Reset
27	RESERVED	Write 0s for future compatibility. Reads returns 0.	RW	0x0
26	RESERVED	Write 0s for future compatibility. Reads returns 0.	RW	0x0
25	ERRCONTENTIONLP1_3_IRQ_EN	Contention LP1 error for lane #3 0x0: Event is masked 0x1: Event generates an interrupt when it occurs	RW	0x0
24	ERRCONTENTIONLP0_3_IRQ_EN	Contention LP0 error for lane #3 0x0: Event is masked 0x1: Event generates an interrupt when it occurs	RW	0x0
23	ERRCONTENTIONLP1_2_IRQ_EN	Contention LP1 error for lane #2 0x0: Event is masked 0x1: Event generates an interrupt when it occurs	RW	0x0
22	ERRCONTENTIONLP0_2_IRQ_EN	Contention LP0 error for lane #2 0x0: Event is masked 0x1: Event generates an interrupt when it occurs	RW	0x0
21	ERRCONTENTIONLP1_1_IRQ_EN	Contention LP1 error for lane #1 0x0: Event is masked 0x1: Event generates an interrupt when it occurs	RW	0x0
20	ERRCONTENTIONLP0_1_IRQ_EN	Contention LP0 error for lane #1 0x0: Event is masked 0x1: Event generates an interrupt when it occurs	RW	0x0
19	RESERVED	Write 0s for future compatibility. Reads returns 0.	RW	0x0
18	RESERVED	Write 0s for future compatibility. Reads returns 0.	RW	0x0
17	STATEULPS3_IRQ_EN	Lane #3 in ultralow-power state 0x0: Event is masked 0x1: Event generates an interrupt when it occurs	RW	0x0
16	STATEULPS2_IRQ_EN	Lane #2 in ultralow-power state 0x0: Event is masked 0x1: Event generates an interrupt when it occurs	RW	0x0
15	STATEULPS1_IRQ_EN	Lane #1 in ultralow-power state 0x0: Event is masked 0x1: Event generates an interrupt when it occurs	RW	0x0
14	RESERVED	Write 0s for future compatibility. Reads returns 0.	RW	0x0
13	RESERVED	Write 0s for future compatibility. Reads returns 0.	RW	0x0
12	ERRCONTROL3_IRQ_EN	Control error for lane #3 0x0: Event is masked 0x1: Event generates an interrupt when it occurs	RW	0x0
11	ERRCONTROL2_IRQ_EN	Control error for lane #2 0x0: Event is masked 0x1: Event generates an interrupt when it occurs	RW	0x0
10	ERRCONTROL1_IRQ_EN	Control error for lane #1 0x0: Event is masked 0x1: Event generates an interrupt when it occurs	RW	0x0
9	RESERVED	Write 0s for future compatibility. Reads returns 0.	RW	0x0
8	RESERVED	Write 0s for future compatibility. Reads returns 0.	RW	0x0

Bits	Field Name	Description	Type	Reset
7	ERRESC3_IRQ_EN	Escape entry error for lane #3 0x0: Event is masked 0x1: Event generates an interrupt when it occurs	RW	0x0
6	ERRESC2_IRQ_EN	Escape entry error for lane #2 0x0: Event is masked 0x1: Event generates an interrupt when it occurs	RW	0x0
5	ERRESC1_IRQ_EN	Escape entry error for lane #1 0x0: Event is masked 0x1: Event generates an interrupt when it occurs	RW	0x0
4	RESERVED	Write 0s for future compatibility. Reads returns 0.	RW	0x0
3	RESERVED	Write 0s for future compatibility. Reads returns 0.	RW	0x0
2	ERRSYNCESC3_IRQ_EN	Low power Data transmission synchronization error for lane #3 0x0: Event is masked 0x1: Event generates an interrupt when it occurs	RW	0x0
1	ERRSYNCESC2_IRQ_EN	Low power Data transmission synchronization error for lane #2 0x0: Event is masked 0x1: Event generates an interrupt when it occurs	RW	0x0
0	ERRSYNCESC1_IRQ_EN	Low power Data transmission synchronization error for lane #1 0x0: Event is masked 0x1: Event generates an interrupt when it occurs	RW	0x0

Table 15-400. Register Call Summary for Register DSI_COMPLEXIO_IRQENABLE

Display Subsystem Use Cases and Tips
- Configure DSI, DSI PLL and Complex I/O: [0]
- Display Subsystem Use Cases and Tips: [1]

Display Subsystem Register Manual
- Display Subsystem Register Mapping Summary: [2]

Table 15-401. DSI_CLK_CTRL

Address Offset	0x0000 0054		
Physical Address	0x4804 FC54	**Instance**	DSI_PROTOCOL_ENGINE
Description	CLOCK CONTROL This register controls the CLOCK GENERATION. The register can be modified only when IF_EN is reset.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PLL_PWR_CMD		PLL_PWR_STATUS		RESERVED						LP_RX_SYNCHRO_ENABLE	LP_CLK_ENABLE	HS_MANUAL_STOP_CTRL	HS_AUTO_STOP_ENABLE	LP_CLK_NULL_PACKET_SIZE		LP_CLK_NULL_PACKET_ENABLE	CIO_CLK_ICG	DDR_CLK_ALWAYS_ON	LP_CLK_DIVISOR										

Bits	Field Name	Description	Type	Reset
31:30	PLL_PWR_CMD	Command for power control of the DSI PLL Control module 0x0: Command to change to OFF state 0x1: Command to change to ON state for PLL only (HSDIVISER is OFF) 0x2: Command to change to ON state for both PLL and HSDIVISER 0x3: Command to change to ON state for both PLL and HSDIVISER (no clock output to the DSI complex I/O)	RW	0x0
29:28	PLL_PWR_STATUS	Status of the power control of the DSI PLL Control module 0x0: DSI PLL Control module in OFF state 0x1: DSI PLL Control module in ON state for PLL only (HSDIVISER is OFF) 0x2: DSI PLL Control module in ON state for both PLL and HSDIVISER 0x3: DSI PLL Control module in ON state for both PLL and HSDIVISER (no clock output to the DSI complex I/O)	R	0x0
27:22	RESERVED	Write 0s for future compatibility. Reads returns 0.	RW	0x00
21	LP_RX_SYNCHRO_ ENABLE	Defines if the DSI functional clock is higher or lower than 30 MHz. The information is used to define synchronization to be used for RxValidEsc. 0x0: The DSI functional clock is equal or slower than 30 MHz. The synchronization is falling/rising. 0x1: The DSI functional clock is higher than 30 MHz. The synchronization is rising/rising.	RW	0x0
20	LP_CLK_ENABLE	Controls the gating of the TXCLKESC clock. 0x0: Disabled. The clock is not generated. The value of LP_CLK_DIVISOR[12:0] bit field is not used and does not have to be programmed. 0x1: Enabled. The clock is generated. The value of LP_CLK_DIVISOR[12:0] bit field is used and must be programmed.	RW	0x0
19	HS_MANUAL_STOP_ CTRL	In case HS_AUTO_STOP_ENABLE bit is set to 0 (reset value), the bit field allows manual control of the assertion/deassertion of the signal DSIStopClk by users. 0x0: DSIStopClk deassertion unconditionally. 0x1: DSIStopClk assertion unconditionally.	RW	0x0
18	HS_AUTO_STOP_ENABLE	Enables the automatic assertion/deassertion of DSIStopClk signal. 0x0: Auto mode disabled. 0x1: Auto mode enabled.	RW	0x0
17:16	LP_CLK_NULL_PACKET_ SIZE	Indicates the size of LP NULL Packets to be sent automatically when after the last LP packet transfer. It is used by the receiver to drain its internal pipeline. The valid values are from 0 to 3 bytes for the payload size.	RW	0x0
15	LP_CLK_NULL_PACKET_ ENABLE	Enables the generation of NULL packet in low speed. 0x0: Disabled. The NULL packet is not sent in LP mode after the last LP packet. 0x1: Enabled. The NULL packet is sent in LP mode after the last LP packet.	RW	0x0

Bits	Field Name	Description	Type	Reset
14	CIO_CLK_ICG	Controls the signal for gating the L3_ICLK clock provided to the complex I/O 0x0: Disabled. The L3_ICLK clock to the DSI complex I/O is gated. 0x1: Enabled. The L3_ICLK clock to the DSI complex I/O is not gated.	RW	0x0
13	DDR_CLK_ALWAYS_ON	Defines if the DDR clock is also sent when there is no HS packets sent to the peripheral (low power mode). So TXRequest for the clock lane is not de-asserted. 0x0: Disabled. The DDR clock is only provided when HS packets are sent. 0x1: Enabled. The DDR clock is always sent to the peripheral regardless of the state of the data lanes (HS or LP mode).	RW	0x0
12:0	LP_CLK_DIVISOR	Defines the ratio to be used for the generation of the low-power mode clock from DSI functional clock. The supported values are from 1 to 8191(the value 0 is invalid). The output frequency must be in the range between 20 MHz and 32 kHz.	RW	0x0001

Table 15-402. Register Call Summary for Register DSI_CLK_CTRL

Display Subsystem Environment
- Physical Layer: [0] [1]

Display Subsystem Functional Description
- Clock Requirements: [2] [3] [4] [5] [6] [7] [8]
- Power Management: [9]
- Power Control: [10] [11] [12] [13] [14] [15]

Display Subsystem Basic Programming Model
- Ultra-Low Power State: [16]

Display Subsystem Use Cases and Tips
- Configure DSI, DSI PLL and Complex I/O: [17] [18] [19] [20]
- Display Subsystem Use Cases and Tips: [21] [22] [23] [24] [25] [26] [27] [28]

Display Subsystem Register Manual
- Display Subsystem Register Mapping Summary: [29]
- DSI Protocol Engine Registers: [30] [31]

Table 15-403. DSI_TIMING1

Address Offset	0x0000 0058		
Physical Address	0x4804 FC58	**Instance**	DSI_PROTOCOL_ENGINE
Description	TIMING1 REGISTER This register controls the DSI Protocol Engine module timers. Any bit field can be modified while DSI_CTRL.IF_EN is set to 1. It is used to indicate the number of DSI_FCLK clock cycles for the timers FORCE_TX_STOP_TIMER and TA_TO_TIMER		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TA_TO	TA_TO_X16	TA_TO_X8	TA_TO_COUNTER													FORCE_TX_STOP_MODE_IO	STOP_STATE_X16_IO	STOP_STATE_X4_IO	STOP_STATE_COUNTER_IO												

Bits	Field Name	Description	Type	Reset
31	TA_TO	Enables the turn-around timer 0x0: Turn-around counter is disabled. 0x1: Turn-around counter is enabled (required to receive TA interrupt in case the turn-around procedure is not successful).	RW	0x0
30	TA_TO_X16	Multiplication factor for the number of DSI_FCLK clock cycles defined in TA_TO_COUNTER bit field 0x0: The number of DSI_FCLK clock cycles defined in TA_TO_COUNTER is multiplied by 1x 0x1: The number of DSI_FCLK clock cycles defined in TA_TO_COUNTER is multiplied by 16x	RW	0x1
29	TA_TO_X8	Multiplication factor for the number of DSI_FCLK clock cycles defined in TA_TO_COUNTER bit field 0x0: The number of DSI_FCLK clock cycles defined in TA_TO_COUNTER is multiplied by 1x 0x1: The number of DSI_FCLK clock cycles defined in TA_TO_COUNTER is multiplied by 8x	RW	0x1
28:16	TA_TO_COUNTER	Turn around counter. It indicates the number of DSI_FCLK clock cycles to wait for the change of the Direction PPI signal according to the TurnRequest signal The value is from 0 to 8191.	RW	0x1FFF
15	FORCE_TX_STOP_ MODE_IO	Control of ForceTxStopMode signal 0x0: Deassertion of ForceTxStopMode. The hardware reset the bit at the end of the ForceTXStopMode assertion. The SW can reset the bit to stop the assertion of the ForceTXStopMode signal prior to the completion of the period. 0x1: Assertion of ForceTxStopMode	RW	0x0
14	STOP_STATE_X16_IO	Multiplication factor for the number of DSI_FCLK clock cycles defined in STOP_STATE_COUNTER_IO bit field 0x0: The number of DSI_FCLK clock cycles defined in STOP_STATE _COUNTER_IO is multiplied by 1x 0x1: The number of DSI_FCLK clock cycles defined in STOP_STATE _COUNTER_IO is multiplied by 16x	RW	0x1
13	STOP_STATE_X4_IO	Multiplication factor for the number of DSI_FCLK clock cycles defined in STOP_STATE_COUNTER_IO bit field 0x0: The number of DSI_FCLK clock cycles defined in STOP_STATE _COUNTER_IO is multiplied by 1x 0x1: The number of DSI_FCLK clock cycles defined in STOP_STATE _COUNTER_IO is multiplied by 4x	RW	0x1
12:0	STOP_STATE_ COUNTER_IO	Stop state counter. It indicates the number of DSI_FCLK clock cycles to assert ForceTXStopMode signal. The value is from 0 to 8191.	RW	0x1FFF

Table 15-404. Register Call Summary for Register DSI_TIMING1

Display Subsystem Functional Description • Timers: [0] [1] [2] [3] [4] [5] [6] [7] [8] [9] [10]
Display Subsystem Basic Programming Model • DSI Programming Sequence Example: [11] [12] [13] [14] [15] [16]
Display Subsystem Use Cases and Tips • Configure DSI, DSI PLL and Complex I/O: [17] • Display Subsystem Use Cases and Tips: [18] [19] [20] [21]
Display Subsystem Register Manual • Display Subsystem Register Mapping Summary: [22]

Table 15-405. DSI_TIMING2

Address Offset	0x0000 005C		
Physical Address	0x4804 FC5C	**Instance**	DSI_PROTOCOL_ENGINE
Description	TIMING2 REGISTER This register controls the DSI Protocol Engine module timers. Any bit field can be modified while DSI_CTRL.IF_EN is set to 1. It is used to indicate the number of TxByteClkHS clock cycles for the timers HS_TX_TIMER and LP_RX_TIMER		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
HS_TX_TO	HS_TX_TO_X16	HS_TX_TO_X8	HS_TX_TO_COUNTER													LP_RX_TO	LP_RX_TO_X16	LP_RX_TO_X4	LP_RX_TO_COUNTER												

Bits	Field Name	Description	Type	Reset
31	HS_TX_TO	Enables the HS TX timer. 0x0: Turn-around counter is disabled. 0x1: Turn-around counter is enabled (required to receive TA interrupt in case the turn-around procedure is not successful).	RW	0x0
30	HS_TX_TO_X16	Multiplication factor for the number of TxByteClkHS functional clock cycles defined in HS_TX_COUNTER bit field 0x0: The number of TxByteClkHS functional clock cycles defined in HS_TX_TO_COUNTER is multiplied by 1x 0x1: The number of TxByteClkHS functional clock cycles defined in HS_TX_TO_COUNTER is multiplied by 16x	RW	0x1
29	HS_TX_TO_X8	Multiplication factor for the number of TxByteClkHS functional clock cycles defined in HS_TX_COUNTER bit 0x0: The number of TxByteClkHS functional clock cycles defined in HS_TX_TO_COUNTER is multiplied by 1x 0x1: The number of TxByteClkHS functional clock cycles defined in HS_TX_TO_COUNTER is multiplied by 8x	RW	0x1
28:16	HS_TX_TO_ COUNTER	HS_TX_TIMER counter. It indicates the number of TxByteClkHS function clock cycles for the HS TX timer. The value is from 0 to 8191.	RW	0x1FFF
15	LP_RX_TO	Enables the LP RX timer. 0x0: Turn-around counter is disabled. 0x1: Turn-around counter is enabled (required to receive TA interrupt in case the turn-around procedure is not successful).	RW	0x0
14	LP_RX_TO_X16	Multiplication factor for the number of DSI_FCLK clock cycles defined in LP_RX_COUNTER bit field 0x0: The number of DSI_FCLK clock cycles defined in LP_RX_TO_COUNTER is multiplied by 1x 0x1: The number of DSI_FCLK clock cycles defined in LP_RX_TO_COUNTER is multiplied by 16x	RW	0x1
13	LP_RX_TO_X4	Multiplication factor for the number of DSI_FCLK clock cycles defined in LP_RX_COUNTER bit 0x0: The number of DSI_FCLK clock cycles defined in LP_RX_TO_COUNTER is multiplied by 1x 0x1: The number of DSI_FCLK clock cycles defined in LP_RX_TO_COUNTER is multiplied by 4x	RW	0x1
12:0	LP_RX_TO_ COUNTER	LP_RX_TIMER counter. It indicates the number of DSI_FCLK clock cycles for the LP RX timer. The value is from 0 to 8191.	RW	0x1FFF

Table 15-406. Register Call Summary for Register DSI_TIMING2

Display Subsystem Functional Description
- Timers: [0] [1] [2] [3] [4] [5]

Display Subsystem Use Cases and Tips
- Configure DSI, DSI PLL and Complex I/O: [6]
- Display Subsystem Use Cases and Tips: [7] [8] [9] [10] [11] [12]

Display Subsystem Register Manual
- Display Subsystem Register Mapping Summary: [13]

Table 15-407. DSI_VM_TIMING1

Address Offset	0x0000 0060		
Physical Address	0x4804 FC60	**Instance**	DSI_PROTOCOL_ENGINE
Description	VIDEO MODE TIMING REGISTER This register defines the video mode timing.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
HSA								HFP								HBP															

Bits	Field Name	Description	Type	Reset
31:24	HSA	Defines the horizontal Sync active period used in video mode in number of byte clock cycles (TxByteClkHS) The supported values are from 0 to 255.	RW	0x00
23:12	HFP	Defines the horizontal front porch used in video mode in number of byte clock cycles (TxByteClkHS) The supported values are from 0 to 255	RW	0x000
11:0	HBP	Defines the horizontal back porch used in video mode in number of byte clock cycles (TxByteClkHS) The supported values are from 0 to 255	RW	0x000

Table 15-408. Register Call Summary for Register DSI_VM_TIMING1

Display Subsystem Basic Programming Model
- Video Mode: [0]
- DSI Programming Sequence Example: [1]

Display Subsystem Use Cases and Tips
- Configure DSI, DSI PLL and Complex I/O: [2]

Display Subsystem Register Manual
- Display Subsystem Register Mapping Summary: [3]

Table 15-409. DSI_VM_TIMING2

Address Offset	0x0000 0064		
Physical Address	0x4804 FC64	**Instance**	DSI_PROTOCOL_ENGINE
Description	VIDEO MODE TIMING REGISTER This register defines the video mode timing.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED				WINDOW_SYNC				VSA								VFP								VBP							

Bits	Field Name	Description	Type	Reset
31:28	RESERVED	Write 0s for future compatibility. Reads returns 0.	RW	0x0
27:24	WINDOW_SYNC	Number of TxByteClkHS clock cycles for the synchronization window. An interrupt for synchronization lost is generated when the received synchronization on video port is not inside the window. The DSI protocol engine does not change its own timings if the synch is inside the window. The valid values are from 0 to 15.	RW	0x0
23:16	VSA	Defines the vertical Sync active period used in video mode in number of lines. The supported values are from 0 to 255 It is used to generate the short packet for End of Vertical synchronization.	RW	0x00
15:8	VFP	Defines the vertical front porch used in video mode in number of lines. The supported values are from 0 to 255	RW	0x00
7:0	VBP	Defines the vertical back porch used in video mode in number of lines. The supported values are from 0 to 255	RW	0x00

Table 15-410. Register Call Summary for Register DSI_VM_TIMING2

Display Subsystem Environment • Video Port (VP) Interface: [0]
Display Subsystem Basic Programming Model • Video Mode: [1] [2]
Display Subsystem Use Cases and Tips • Configure DSI, DSI PLL and Complex I/O: [3]
Display Subsystem Register Manual • Display Subsystem Register Mapping Summary: [4]

Table 15-411. DSI_VM_TIMING3

Address Offset	0x0000 0068		
Physical Address	0x4804 FC68	**Instance**	DSI_PROTOCOL_ENGINE
Description	VIDEO MODE TIMING REGISTER This register defines the video mode timing.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TL																VACT															

Bits	Field Name	Description	Type	Reset
31:16	TL	Defines the number of length of the line in video mode in number of byte clock cycles (TxByteClkHS) The supported values are from 0 to 8192. The values from 8193 to 65535 are not supported.	RW	0x0000
15:0	VACT	Defines the number of active lines used in video mode. The supported values are from 0 to 65535	RW	0x0000

Table 15-412. Register Call Summary for Register DSI_VM_TIMING3

Display Subsystem Environment • Video Port (VP) Interface: [0]
Display Subsystem Basic Programming Model • Video Mode: [1]
Display Subsystem Use Cases and Tips • Configure DSI, DSI PLL and Complex I/O: [2]
Display Subsystem Register Manual • Display Subsystem Register Mapping Summary: [3]

Table 15-413. DSI_CLK_TIMING

Address Offset	0x0000 006C		
Physical Address	0x4804 FC6C	**Instance**	DSI_PROTOCOL_ENGINE
Description	CLOCK TIMING REGISTER This register controls the DSI Protocol Engine module timers. This register should not be modified while DSI_CTRL.IF_EN is set to 1.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																DDR_CLK_PRE								DDR_CLK_POST							

Bits	Field Name	Description	Type	Reset
31:16	RESERVED	Write 0s for future compatibility. Reads returns 0.	RW	0x0000
15:8	DDR_CLK_PRE	Indicates the number of TxByteClkHS cycles between the start of the DDR clock and the assertion of the data request signal. The values from 1 to 255 are valid. The value 0 is reserved. The value is not used if DSI_CLK_CTRL[13] DDR_CLK_ALWAYS_ON is set to 1 since the DDR clock is always present.	RW	0x01
7:0	DDR_CLK_POST	Indicates the number of TxByteClkHS cycles after the deassertion of the data request signal and the stop of the DDR clock. The values from 1 to 255 are valid. The value 0 is reserved. The value is not used if DSI_CLK_CTRL[13] DDR_CLK_ALWAYS_ON is set to 1 since the DDR clock is always present.	RW	0x01

Table 15-414. Register Call Summary for Register DSI_CLK_TIMING

Display Subsystem Functional Description

- Clock Requirements: [0] [1]

Display Subsystem Use Cases and Tips

- Configure DSI, DSI PLL and Complex I/O: [2]
- Display Subsystem Use Cases and Tips: [3]

Display Subsystem Register Manual

- Display Subsystem Register Mapping Summary: [4]

Table 15-415. DSI_TX_FIFO_VC_SIZE

Address Offset	0x0000 0070		
Physical Address	0x4804 FC70	**Instance**	DSI_PROTOCOL_ENGINE
Description	Defines the corresponding memory entries allocated for each VC. The VC must be disabled to allocate/un-allocate some entries in the TX FIFO.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
VC3_FIFO_SIZE				RESERVED			VC3_FIFO_ADD				VC2_FIFO_SIZE				RESERVED			VC2_FIFO_ADD				VC1_FIFO_SIZE				RESERVED			VC1_FIFO_ADD				VC0_FIFO_SIZE				RESERVED			VC0_FIFO_ADD

Bits	Field Name	Description	Type	Reset
31:28	VC3_FIFO_SIZE	Size of the FIFO allocated for VC 3. For a complete description, refer to Table 15-64.	RW	0x0
27	RESERVED	Write 0s for future compatibility. Reads returns 0.	RW	0x0
26:24	VC3_FIFO_ADD	Address of the space allocated in the FIFO for VC 3. For a complete description, refer to Table 15-65.	RW	0x0

Bits	Field Name	Description	Type	Reset
23:20	VC2_FIFO_SIZE	Size of the FIFO allocated for VC 2. For a complete description, refer to Table 15-64.	RW	0x0
19	RESERVED	Write 0s for future compatibility. Reads returns 0.	RW	0x0
18:16	VC2_FIFO_ADD	Address of the space allocated in the FIFO for VC 2. For a complete description, refer to Table 15-65.	RW	0x0
15:12	VC1_FIFO_SIZE	Size of the FIFO allocated for VC 1. For a complete description, refer to Table 15-64.	RW	0x0
11	RESERVED	Write 0s for future compatibility. Reads returns 0.	RW	0x0
10:8	VC1_FIFO_ADD	Address of the space allocated in the FIFO for VC 1.For a complete description, refer to Table 15-65.	RW	0x0
7:4	VC0_FIFO_SIZE	Size of the FIFO allocated for VC 0. For a complete description, refer to Table 15-64.	RW	0x0
3	RESERVED	Write 0s for future compatibility. Reads returns 0.	RW	0x0
2:0	VC0_FIFO_ADD	Address of the space allocated in the FIFO for VC 0. For a complete description, refer to Table 15-65.	RW	0x0

Table 15-416. Register Call Summary for Register DSI_TX_FIFO_VC_SIZE

Display Subsystem Basic Programming Model • Command Mode: [0] [1] [2] [3] [4]
Display Subsystem Use Cases and Tips • Display Subsystem Use Cases and Tips: [5]
Display Subsystem Register Manual • Display Subsystem Register Mapping Summary: [6]

Table 15-417. DSI_RX_FIFO_VC_SIZE

Address Offset	0x0000 0074		
Physical Address	0x4804 FC74	**Instance**	DSI_PROTOCOL_ENGINE
Description	Defines the corresponding memory entries allocated for each VC and the addresses. The VC must be disabled to allocate/un-allocate some entries in the RX FIFO.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
VC3_FIFO_SIZE				RESERVED				VC3_FIFO_ADD				VC2_FIFO_SIZE				RESERVED				VC2_FIFO_ADD				VC1_FIFO_SIZE				RESERVED			
VC1_FIFO_ADD				VC0_FIFO_SIZE				RESERVED				VC0_FIFO_ADD																			

Bits	Field Name	Description	Type	Reset
31:28	VC3_FIFO_SIZE	Size of the FIFO allocated for VC 3. For a complete description, refer to Table 15-64.	RW	0x0
27	RESERVED	Write 0s for future compatibility. Reads returns 0.	RW	0x0
26:24	VC3_FIFO_ADD	Address of the space allocated in the FIFO for VC 3.For a complete description, refer to Table 15-65.	RW	0x0
23:20	VC2_FIFO_SIZE	Size of the FIFO allocated for VC 2. For a complete description, refer to Table 15-64.	RW	0x0
19	RESERVED	Write 0s for future compatibility. Reads returns 0.	RW	0x0
18:16	VC2_FIFO_ADD	Address of the space allocated in the FIFO for VC 2. For a complete description, refer to Table 15-65.	RW	0x0

Bits	Field Name	Description	Type	Reset
15:12	VC1_FIFO_SIZE	Size of the FIFO allocated for VC 1. For a complete description, refer to Table 15-64.	RW	0x0
11	RESERVED	Write 0s for future compatibility. Reads returns 0.	RW	0x0
10:8	VC1_FIFO_ADD	Address of the space allocated in the FIFO for VC 1. For a complete description, refer to Table 15-65.	RW	0x0
7:4	VC0_FIFO_SIZE	Size of the FIFO allocated for VC 0. For a complete description, refer to Table 15-64.	RW	0x0
3	RESERVED	Write 0s for future compatibility. Reads returns 0.	RW	0x0
2:0	VC0_FIFO_ADD	Address of the space allocated in the FIFO for VC 0. For a complete description, refer to Table 15-65.	RW	0x0

Table 15-418. Register Call Summary for Register DSI_RX_FIFO_VC_SIZE

Display Subsystem Basic Programming Model • Command Mode: [0] [1] [2]
Display Subsystem Use Cases and Tips • Display Subsystem Use Cases and Tips: [3]
Display Subsystem Register Manual • Display Subsystem Register Mapping Summary: [4]

Table 15-419. DSI_COMPLEXIO_CFG2

Address Offset	0x0000 0078		
Physical Address	0x4804 FC78	**Instance**	DSI_PROTOCOL_ENGINE
Description	COMPLEXIO CONFIGURATION REGISTER for the complex I/O This register contains the lane configuration for the ULPS for each lane.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED														LP_BUSY	HS_BUSY	RESERVED						RESERVED	RESERVED	LANE3_ULPS_SIG2	LANE2_ULPS_SIG2	LANE1_ULPS_SIG2	RESERVED	RESERVED	LANE3_ULPS_SIG1	LANE2_ULPS_SIG1	LANE1_ULPS_SIG1

Bits	Field Name	Description	Type	Reset
31:18	RESERVED	Write 0s for future compatibility. Reads returns 0.	RW	0x0000
17	LP_BUSY	Indicates when there are still pending operations for VCs configured for LP mode. Forced to 1 when at least one VC is enabled and configured for LP mode. Read 0x0: LP logic is idle Read 0x1: LP logic is active	R	0x0
16	HS_BUSY	Indicates when there are still pending operations for VCs configured for HS mode. Forced to 1 when at least one VC is enabled and configured for HS mode Read 0x0: HS logic is idle Read 0x1: HS logic is active	R	0x0
15:10	RESERVED	Write 0s for future compatibility. Reads returns 0.	RW	0x00
9	RESERVED	Write 0s for future compatibility. Reads returns 0.	RW	0x0

Bits	Field Name	Description	Type	Reset
8	RESERVED	Write 0s for future compatibility. Reads returns 0.	RW	0x0
7	LANE3_ULPS_SIG2	Enables the ULPS for the lane #3. The HW should change the state of the lane to ULPS only when it is in stop state and there is no data pending inside the DSI protocol engine and the DSI protocol engine has control of the bus (BTA has not been sent). The state of the signal TxRequestEsc is change if lane #3 is a data lane. The state of the signal TxUlpsClk is change if lane #3 is a clock lane. There will be a latency depending on the frequency of TxClkExc. This bit should be read back to confirm a write has been effective. 0x0: READ: Inactive state effective WRITE: Request to change to inactive state 0x1: READ: Active state effective WRITE: Change request to active. If the lane is a data lane, TxRequestEsc is asserted and synchronously TxUlpsEsc is asserted for one period of TxClkEsc.	RW	0x0
6	LANE2_ULPS_SIG2	Enables the ULPS for the lane #2. The HW should change the state of the lane to ULPS only when it is in stop state and there is no data pending inside the DSI protocol engine and the DSI protocol engine has control of the bus (BTA has not been sent). The state of the signal TxRequestEsc is change if lane #2 is a data lane. The state of the signal TxUlpsClk is change if lane #2 is a clock lane. There will be a latency depending on the frequency of TxClkExc. This bit should be read back to confirm a write has been effective. 0x0: READ: Inactive state effective WRITE: Request to change to inactive state 0x1: READ: ACTIVE state effective WRITE: Change request to active. If the lane is a data lane, TxRequestEsc is asserted and synchronously TxUlpsEsc is asserted for one period of TxClkEsc.	RW	0x0
5	LANE1_ULPS_SIG2	Enables the ULPS for the lane #1. The HW should change the state of the lane to ULPS only when it is in stop state and there is no data pending inside the DSI protocol engine and the DSI protocol engine has control of the bus (BTA has not been sent). The state of the signal TxRequestEsc is change if lane #1 is a data lane. The state of the signal TxUlpsClk is change if lane #1 is a clock lane. There will be a latency depending on the frequency of TxClkExc. This bit should be read back to confirm a write has been effective. 0x0: READ: Inactive state effective WRITE: Request to change to inactive state 0x1: READ: ACTIVE state effective WRITE: Change request to active. If the lane is a data lane, TxRequestEsc is asserted and synchronously TxUlpsEsc is asserted for one period of TxClkEsc.	RW	0x0
4	RESERVED	Write 0s for future compatibility. Reads returns 0.	RW	0x0
3	RESERVED	Write 0s for future compatibility. Reads returns 0.	RW	0x0
2	LANE3_ULPS_SIG1	Enables the ULPS for the lane #3. The HW should change the state of the lane to ULPS only when it is in stop state and there is no data pending inside the DSI protocol engine and the DSI protocol engine has control of the bus (BTA has not been sent). The state of the signal TxULPSExit is changed if lane #3 is a clock lane. There will be a latency depending on the frequency of TxClkExc. This bit should be read back to confirm a write has been effective. 0x0: READ: Inactive state effective WRITE: Request to change to inactive state 0x1: READ: ACTIVE state effective WRITE: Change request to active. If the lane is a data lane, TxRequestEsc is asserted and synchronously TxUlpsEsc is asserted for one period of TxClkEsc.	RW	0x0

Bits	Field Name	Description	Type	Reset
1	LANE2_ULPS_SIG1	Enables the ULPS for the lane #2. The HW must change the state of the lane to ULPS only when it is in stop state and there is no data pending inside the DSI protocol engine and the DSI protocol engine has control of the bus (BTA has not been sent). The state of the signal TxULPSExit is changed if lane #3 is a clock lane. There will be a latency depending on the frequency of TxClkExc. This bit should be read back to confirm a write has been effective. 0x0: READ: Inactive state effective WRITE: Request to change to inactive state 0x1: READ: ACTIVE state effective WRITE: Change request to active. If the lane is a data lane, TxRequestEsc is asserted and synchronously TxUlpsEsc is asserted for one period of TxClkEsc.	RW	0x0
0	LANE1_ULPS_SIG1	Enables the ULPS for the lane #1. The HW must change the state of the lane to ULPS only when it is in stop state and there is no data pending inside the DSI protocol engine and the DSI protocol engine has control of the bus (BTA has not been sent). The state of the signal TxULPSExit is changed if lane #3 is a clock lane. There will be a latency depending on the frequency of TxClkExc. This bit should be read back to confirm a write has been effective. 0x0: READ: Inactive state effective WRITE: Request to change to inactive state 0x1: READ: ACTIVE state effective WRITE: Change request to active. If the lane is a data lane, TxRequestEsc is asserted and synchronously TxUlpsEsc is asserted for one period of TxClkEsc.	RW	0x0

Table 15-420. Register Call Summary for Register DSI_COMPLEXIO_CFG2

Display Subsystem Environment
- Physical Layer: [0]

Display Subsystem Basic Programming Model
- Ultra-Low Power State: [1] [2] [3] [4] [5] [6] [7] [8] [9] [10] [11] [12] [13] [14] [15] [16] [17] [18] [19] [20] [21] [22] [23] [24] [25] [26] [27]

Display Subsystem Register Manual
- Display Subsystem Register Mapping Summary: [28]

Table 15-421. DSI_RX_FIFO_VC_FULLNESS

Address Offset	0x0000 007C		
Physical Address	0x4804 FC7C	**Instance**	DSI_PROTOCOL_ENGINE
Description	Defines the fullness of each space allocated for each VC.		
Type	R		

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10 9 8	7 6 5 4 3 2 1 0
VC3_FIFO_FULLNESS	VC2_FIFO_FULLNESS	VC1_FIFO_FULLNESS	VC0_FIFO_FULLNESS

Bits	Field Name	Description	Type	Reset
31:24	VC3_FIFO_FULLNESS	Fullness of the FIFO allocated for VC 3.The valid values are from 0 to 127 corresponding to 1x33-bit,...up to 128x33-bit.	R	0x00
23:16	VC2_FIFO_FULLNESS	Fullness of the FIFO allocated for VC 2.The valid values are from 0 to 127 corresponding to 1x33-bit,...up to 128x33-bit.	R	0x00
15:8	VC1_FIFO_FULLNESS	Fullness of the FIFO allocated for VC 1.The valid values are from 0 to 127 corresponding to 1x33-bit,...up to 128x33-bit.	R	0x00
7:0	VC0_FIFO_FULLNESS	Fullness of the FIFO allocated for VC 0.The valid values are from 0 to 127 corresponding to 1x33-bit,...up to 128x33-bit.	R	0x00

Table 15-422. Register Call Summary for Register DSI_RX_FIFO_VC_FULLNESS

Display Subsystem Basic Programming Model
- Command Mode: [0]

Display Subsystem Register Manual
- Display Subsystem Register Mapping Summary: [1]

Table 15-423. DSI_VM_TIMING4

Address Offset	0x0000 0080		
Physical Address	0x4804 FC80	**Instance**	DSI_PROTOCOL_ENGINE
Description	VIDEO MODE TIMING REGISTER This register defines the video mode timing.		
Type	RW		

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10 9 8	7 6 5 4 3 2 1 0
RESERVED	HSA_HS_INTERLEAVING	HFP_HS_INTERLEAVING	HBP_HS_INTERLEAVING

Bits	Field Name	Description	Type	Reset
31:24	RESERVED	Write 0s for future compatibility. Reads returns 0.	RW	0x00
23:16	HSA_HS_INTERLEAVING	Defines the number of TxByteClkHS cycles that can be used for interleaving high-speed command mode packet into Video Mode stream during HSA blanking period. The supported values are from 0 to 255.	RW	0x00
15:8	HFP_HS_INTERLEAVING	Defines the number of TxByteClkHS cycles that can be used for interleaving high-speed command mode packet into Video Mode stream during HFP blanking period. The supported values are from 0 to 255	RW	0x00
7:0	HBP_HS_INTERLEAVING	Defines the number of TxByteClkHS cycles that can be used for interleaving high-speed command mode packet into Video Mode stream during HBP blanking period. The supported values are from 0 to 255	RW	0x00

Table 15-424. Register Call Summary for Register DSI_VM_TIMING4

Display Subsystem Functional Description
- DSI Transfer Modes: [0] [1] [2]

Display Subsystem Basic Programming Model
- Video Mode: [3]

Display Subsystem Register Manual
- Display Subsystem Register Mapping Summary: [4]

Table 15-425. DSI_TX_FIFO_VC_EMPTINESS

Address Offset	0x0000 0084		
Physical Address	0x4804 FC84	**Instance**	DSI_PROTOCOL_ENGINE
Description	Defines the emptiness of each space allocated for each VC.		
Type	R		

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10 9 8	7 6 5 4 3 2 1 0
VC3_FIFO_EMPTINESS	VC2_FIFO_EMPTINESS	VC1_FIFO_EMPTINESS	VC0_FIFO_EMPTINESS

Bits	Field Name	Description	Type	Reset
31:24	VC3_FIFO_EMPTINESS	Emptiness of the FIFO allocated for VC 3.The valid values are from 0 to 127 corresponding to 1x33-bit,...up to 128x33-bit.	R	0x00
23:16	VC2_FIFO_EMPTINESS	Emptiness of the FIFO allocated for VC 2.The valid values are from 0 to 127 corresponding to 1x33-bit,...up to 128x33-bit.	R	0x00

Bits	Field Name	Description	Type	Reset
15:8	VC1_FIFO_EMPTINESS	Emptiness of the FIFO allocated for VC 1.The valid values are from 0 to 127 corresponding to 1x33-bit,...up to 128x33-bit.	R	0x00
7:0	VC0_FIFO_EMPTINESS	Emptiness of the FIFO allocated for VC 0.The valid values are from 0 to 127 corresponding to 1x33-bit,...up to 128x33-bit.	R	0x00

Table 15-426. Register Call Summary for Register DSI_TX_FIFO_VC_EMPTINESS

Display Subsystem Basic Programming Model
- Command Mode: [0] [1]

Display Subsystem Register Manual
- Display Subsystem Register Mapping Summary: [2]

Table 15-427. DSI_VM_TIMING5

Address Offset	0x0000 0088		
Physical Address	0x4804 FC88	**Instance**	DSI_PROTOCOL_ENGINE
Description	VIDEO MODE TIMING REGISTER This register defines the video mode timing.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								HSA_LP_INTERLEAVING								HFP_LP_INTERLEAVING								HBP_LP_INTERLEAVING							

Bits	Field Name	Description	Type	Reset
31:24	RESERVED	Write 0s for future compatibility. Reads returns 0.	RW	0x00
23:16	HSA_LP_INTERLEAVING	Defines the number of bytes for Low Power command mode packets that can be sent on PPI link during HSA blanking period. The supported values are from 0 to 255.	RW	0x00
15:8	HFP_LP_INTERLEAVING	Defines the number of bytes for Low Power command mode packets that can be sent on PPI link during HFP blanking period. The supported values are from 0 to 255	RW	0x00
7:0	HBP_LP_INTERLEAVING	Defines the number of bytes for Low Power command mode packets that can be sent on PPI link during HBP blanking period. The supported values are from 0 to 255	RW	0x00

Table 15-428. Register Call Summary for Register DSI_VM_TIMING5

Display Subsystem Functional Description
- DSI Transfer Modes: [0] [1] [2]

Display Subsystem Basic Programming Model
- Video Mode: [3]

Display Subsystem Register Manual
- Display Subsystem Register Mapping Summary: [4]

Table 15-429. DSI_VM_TIMING6

Address Offset	0x0000 008C		
Physical Address	0x4804 FC8C	**Instance**	DSI_PROTOCOL_ENGINE
Description	VIDEO MODE TIMING REGISTER This register defines the video mode timing.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BL_HS_INTERLEAVING																BL_LP_INTERLEAVING															

Bits	Field Name	Description	Type	Reset
31:16	BL_HS_INTERLEAVING	Defines the number of TxByteClkHS clock cycles that can be used for interleaving high-speed command mode packet into Video Mode stream during blanking periods during VSA, VBP, VFP periods inside one video frame on PPI link. The supported values are from 0 to 65535.	RW	0x0000
15:0	BL_LP_INTERLEAVING	Defines the maximum number of bytes for Low Power command mode packets that can be sent on PPI link during blanking periods during VSA, VBP or VFP periods inside one video frame on PPI link. The supported values are from 0 to 65535	RW	0x0000

Table 15-430. Register Call Summary for Register DSI_VM_TIMING6

Display Subsystem Environment • Video Port (VP) Interface: [0]
Display Subsystem Functional Description • DSI Transfer Modes: [1] [2]
Display Subsystem Basic Programming Model • Video Mode: [3]
Display Subsystem Register Manual • Display Subsystem Register Mapping Summary: [4]

Table 15-431. DSI_VM_TIMING7

Address Offset	0x0000 0090		
Physical Address	0x4804 FC90	**Instance**	DSI_PROTOCOL_ENGINE
Description	Defines the maximum number of bytes of Low Power command mode packets that can be sent on PPI link during blanking periods during VSA, VBP or VFP periods inside one video frame on PPI link. The supported values are from 0 to 65535		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ENTER_HS_MODE_LATENCY																EXIT_HS_MODE_LATENCY															

Bits	Field Name	Description	Type	Reset
31:16	ENTER_HS_MODE_ LATENCY	Defines the number of TxByteClkHS clock cycles necessary for entering to HS mode. It corresponds to the delay in number of HS clock cycles from assertion of TxRequestHS signal to 1 until assertion of TxReadyHS signal to 1. The supported values are from 0 to 65535 .	RW	0x0000
15:0	EXIT_HS_MODE_ LATENCY	Defines the number of TxByteClkHS clock cycles necessary for exiting from HS mode. It corresponds to the maximum delay in number of TxByteClkHS clock cycles from deassertion of TxRequestHS signal until PPI link is in LP-11 state from which a new entrance to HS mode can be initiated which does not take more than ENTER_HS_MODE_LATENCY clock cycles. The supported values are from 0 to 65535	RW	0x0000

Table 15-432. Register Call Summary for Register DSI_VM_TIMING7

Display Subsystem Functional Description • Clock Requirements: [0] [1]
Display Subsystem Basic Programming Model • Video Mode: [2] • DSI Programming Sequence Example: [3]
Display Subsystem Use Cases and Tips • Configure DSI, DSI PLL and Complex I/O: [4]
Display Subsystem Register Manual • Display Subsystem Register Mapping Summary: [5]

Table 15-433. DSI_STOPCLK_TIMING

Address Offset	0x0000 0094		
Physical Address	0x4804 FC94	**Instance**	DSI_PROTOCOL_ENGINE
Description	Number of functional clock cycles to wait for TxByteClock to stop/start after change in DSIStopClk signal		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																								DSI_STOPCLK_LATENCY							

Bits	Field Name	Description	Type	Reset
31:8	RESERVED	Write 0s for future compatibility. Reads returns 0.	R	0x000000
7:0	DSI_STOPCLK_ LATENCY	Clock gating latency from DSI Protocol engine to TxByteClkHS	RW	0x80

Table 15-434. Register Call Summary for Register DSI_STOPCLK_TIMING

Display Subsystem Functional Description • Power Control: [0] [1]
Display Subsystem Register Manual • Display Subsystem Register Mapping Summary: [2]

Table 15-435. DSI_VCn_CTRL

Address Offset	0x0000 0100+ (n* 0x20)	**Index**	n = 0 to 3
Physical Address	0x4804 FD00+ (n* 0x20)	**Instance**	DSI_PROTOCOL_ENGINE
Description	CONTROL REGISTER - Virtual channel This register controls the VC.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								DMA_RX_REQ_NB				DMA_RX_THRESHOLD				DMA_TX_REQ_NB				RX_FIFO_NOT_EMPTY		DMA_TX_THRESHOLD				TX_FIFO_FULL	VC_BUSY	PP_BUSY	RESERVED			MODE_SPEED	ECC_TX_EN	CS_TX_EN	BTA_EN	TX_FIFO_NOT_EMPTY	MODE	BTA_LONG_EN	BTA_SHORT_EN	SOURCE	VC_EN

Bits	Field Name	Description	Type	Reset
31:30	RESERVED	Write 0s for future compatibility. Reads returns 0.	RW	0x0
29:27	DMA_RX_REQ_NB	Selection of the use of the DMA request (associated to the RX FIFO) 0x0: DSI_DMA_REQ0 is selected 0x1: DSI_DMA_REQ1 is selected 0x2: DSI_DMA_REQ2 is selected 0x3: DSI_DMA_REQ3 is selected 0x4: No DMA req selected	RW	0x0
26:24	DMA_RX_THRESHOLD	Defines the threshold value for the DMA request (associated to the RX FIFO) 0x0: 1x 32 bits 0x1: 2 x 32 bits 0x2: 4 x 32 bits 0x3: 8 x 32 bits 0x4: 16 x 32 bits 0x5: 32 x 32 bits	RW	0x0
23:21	DMA_TX_REQ_NB	Selection of the use of the DMA request (associated to the TX FIFO) 0x0: DSI_DMA_REQ0 is selected 0x1: DSI_DMA_REQ1 is selected 0x2: DSI_DMA_REQ2 is selected 0x3: DSI_DMA_REQ3 is selected 0x4: No DMA req selected	RW	0x0
20	RX_FIFO_NOT_EMPTY	FIFO status in command mode. Otherwise, this bit can be ignored. 0x0: The RX FIFO is empty (the FIFO does not contain any data for the VC) 0x1: The RX FIFO is not empty (the FIFO contains at least one byte for the VC)	R	0x0
19:17	DMA_TX_THRESHOLD	Defines the threshold value for the DMA request (associated to the TX FIFO) 0x0: 1x 32 bits 0x1: 2 x 32 bits 0x2: 4 x 32 bits 0x3: 8 x 32 bits 0x4: 16 x 32 bits 0x5: 32 x 32 bits	RW	0x0
16	TX_FIFO_FULL	FIFO status in command mode. Otherwise, this bit can be ignored. 0x0: The TX FIFO is not full (the FIFO can accept at least one more 32-bit value) 0x1: The TX FIFO is full	R	0x0
15	VC_BUSY	Indicates if previously scheduled activities (packets, BTA) are still being processed. Forced to 1 by hardware if VC is enabled. Software should check this bit is 0 before changing channel configuration. 0x0: No pending operations for this VC 0x1: Pending operations for this VC	R	0x0
14	PP_BUSY	Line buffer busy status. 0x0: Software is permitted to write a new header for VP command mode traffic. 0x1: Software is NOT permitted to write a new header for VP command mode traffic.	R	0

Bits	Field Name	Description	Type	Reset
13:10	RESERVED	Write 0s for future compatibility. Reads returns 0.	RW	0x00
9	MODE_SPEED	Selection of the mode. This bit is ignored by hardware when video mode is selected. 0x0: Low-power mode (CMOS) is used to send short and long packets to the peripheral. 0x1: High speed mode (SLVS) is used to send short and long packets to the peripheral.	RW	0x0
8	ECC_TX_EN	Enables the ECC generation for the transmit header (short and long packets). 0x0: Disabled 0x1: Enabled	RW	0x0
7	CS_TX_EN	Enables the checksum generation for the transmit payload (long packet only). 0x0: Disabled. The value 0x00 is used. 0x1: Enabled. The Check-sum value is calculated by HW.	RW	0x0
6	BTA_EN	Send the bus turn around to the peripheral. It can be used when the automatic mode is enabled (BTA_SHORT_EN=1 or/and BTA_LONG_EN=1). In that case only one BTA is sent to the peripheral. The manual mode allows users to define for which packets, the turn around is required for example getting acknowledge from the peripheral. 0x0: READS: BTA generation is completed. It is reset by HW when it is completed. WRITES: Cancellation of the BTA generation (not guarantee since it could already on going, must not be used). 0x1: READS: BTA generation has been requested by user (it could be on going but not completed). WRITES: Request for BTA generation.	RW	0x0
5	TX_FIFO_NOT_EMPTY	FIFO status 0x0: The TX FIFO is empty (the FIFO does not contain any data for the VC) 0x1: The TX FIFO is not empty (the FIFO contains at least one byte for the VC)	R	0x0
4	MODE	Selection of the mode 0x0: Command mode. 0x1: Video mode.	RW	0x0
3	BTA_LONG_EN	Enables the automatic bus turn-around after completion of each long packet transmission. 0x0: Disabled 0x1: Enabled	RW	0x0
2	BTA_SHORT_EN	Enables the automatic bus turn-around after completion of each short packet transmission. 0x0: Disabled 0x1: Enabled	RW	0x0

Bits	Field Name	Description	Type	Reset
1	SOURCE	Selection of the source between L4 interconnect slave port and video port. This bit is ignored by hardware when video mode is selected. 0x0: All the data are provided by the L4 interconnect slave port. Any transfer on the video port is ignored for this VC. 0x1: If MODE=VIDEO_MODE. any data received on the video port (pixels and enabled synchronization events using DSI_CTRL[17] VP_HSYNC_START, DSI_CTRL[18] VP_HSYNC_END, DSI_CTRL[15] VP_VSYNC_START, DSI_CTRL[16] VP_VSYNC_END,) are sent on the VC (only one VC can be associated with the video port, the software must ensure that no more than one VC is enabled with the video port as the main source for data). If MODE=COMMAND_MODE, the VP_STALL signal is used by the protocol engine to indicate when new data are required. The synchronization signals are not generated by the display controller. Regardless of the MODE, no data can be provided on the L4 interconnect slave port.	RW	0x0
0	VC_EN	Enables the VC. 0x0: Disabled. The VC must be disabled for any register change in the DSI_VCn_XXX registers the corresponding VC ID. 0x1: Enabled. No change is allowed to the VC registers (except for setting the bit fields/registers: DSI_VCn_CTRL[6] BTA_EN, DSI_VCn_TE[15:0] TE_SIZE, DSI_VCn_TE[31] TE_START, DSI_VCn_LONG_XXX, DSI_VCn_SHORT_XXX, DSI_VCn_IRQXXX registers).	RW	0x0

Table 15-436. Register Call Summary for Register DSI_VCn_CTRL

Display Subsystem Environment
- Protocol Layer: [0] [1] [2] [3]

Display Subsystem Functional Description
- DSI Transfer Modes: [4] [5] [6]
- Power Control: [7] [8]
- Timers: [9]
- Bus Turnaround: [10] [11] [12] [13] [14] [15] [16] [17] [18] [19] [20] [21] [22] [23] [24] [25] [26]
- PHY Triggers: [27] [28] [29]
- ECC Generation: [30]
- Checksum Generation for Long Packet Payloads: [31]

Display Subsystem Basic Programming Model
- Global Register Controls: [32]
- Virtual Channels: [33] [34] [35] [36] [37] [38] [39] [40] [41] [42] [43]
- Packets: [44] [45]
- Video Mode: [46] [47] [48] [49] [50] [51] [52]
- Command Mode: [53] [54] [55] [56] [57] [58] [59] [60] [61] [62] [63] [64]
- DSI Programming Sequence Example: [65] [66] [67] [68] [69] [70] [71] [72] [73] [74] [75] [76] [77] [78]

Display Subsystem Register Manual
- Display Subsystem Register Mapping Summary: [79]
- DSI Protocol Engine Registers: [80] [81]

Table 15-437. DSI_VCn_TE

Address Offset	0x0000 0104+ (n* 0x20)	**Index**	n = 0 to 3
Physical Address	0x4804 FD04+ (n* 0x20)	**Instance**	DSI_PROTOCOL_ENGINE
Description	CONTROL REGISTER - Virtual channel This register controls the tearing effect logic. It defines the size of the transfer when TE occurs and enables the automatic TE mode.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TE_START	TE_EN	RESERVED						TE_SIZE																							

Bits	Field Name	Description	Type	Reset
31	TE_START	Manual control of the start of the transfer. Users can use the TE interrupt to determine that the TE trigger has been received before setting the TE_START bit field. It is not mandatory to use the TE interrupt. 0x0: Indicates the end of the transfer. The bit can be used to cancel the transfer if not already started. The FIFO must be flushed by software to ensure it contains no remaining data. 0x1: Starts the transfer of the data. The size is defined in TE_SIZE. The bit field is set until the transfer completes. It is reset by hardware when the transfer completes.	RW	0
30	TE_EN	Tearing effect control 0x0: Disables the automatic transfer of the data using the TE trigger as a synchronization event. The interruption is used to know when the TE trigger is received. The hardware resets the bit field when the transfer completes(TE_SIZE=0). 0x1: Enables the automatic transfer of the data using the TE trigger as a synchronization event.	RW	0
29:24	RESERVED	Write 0s for future compatibility. Reads returns 0.	RW	0x0000
23:0	TE_SIZE	Defines the number of bytes (payload data excluding the check-sum) to be sent. Users must perform the write into the DSI_VCn_LONG_PACKET_HEADER register before sending data from the DSI_VCn_LONG_PACKET_PAYLOAD register. The register value is decremented for every byte of the DSI link that is sent. At the end of the transfer (TE_SIZE = 0), the TE_EN bit field is reset by hardware. The DMA_request is asserted when the trigger is received in order to receive data in the TX FIFO. It must not be used until all data (TE_SIZE) have been received in the FIFO.	RW	0x000000

Table 15-438. Register Call Summary for Register DSI_VCn_TE

Display Subsystem Functional Description

- PHY Triggers: [0] [1] [2] [3] [4] [5] [6] [7] [8] [9] [10] [11] [12] [13]

Display Subsystem Register Manual

- Display Subsystem Register Mapping Summary: [14]
- DSI Protocol Engine Registers: [15] [16]

Table 15-439. DSI_VCn_LONG_PACKET_HEADER

Address Offset	0x0000 0108+ (n* 0x20)	**Index**	n = 0 to 3
Physical Address	0x4804 FD08+ (n* 0x20)	**Instance**	DSI_PROTOCOL_ENGINE
Description	LONG PACKET HEADER INFORMATION–virtual channel. This register sets the 32-bit DATA_ID + Word count + ECC. The ECC is computed if ECC_TX_EN is set to 1. DATA_ID is located at bit[7:0]. WC is located at bit[23:8]. ECC is located at bit[31:24] (least-significant byte first and least significant bit first).		
Type	W		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
HEADER																															

Bits	Field Name	Description	Type	Reset
31:0	HEADER	Packet header information: DATA ID + DATA FIELD +ECC	W	0x00000000

Table 15-440. Register Call Summary for Register DSI_VCn_LONG_PACKET_HEADER

Display Subsystem Environment
- Video Port (VP) Interface: [0] [1]
- Protocol Layer: [2]

Display Subsystem Functional Description
- DSI Transfer Modes: [3] [4] [5] [6] [7] [8] [9]
- Bus Turnaround: [10]
- PHY Triggers: [11] [12] [13]

Display Subsystem Basic Programming Model
- Global Register Controls: [14]
- Virtual Channels: [15]
- Packets: [16] [17] [18] [19] [20] [21]
- Video Mode: [22] [23]
- Command Mode: [24] [25] [26] [27] [28] [29] [30] [31]
- DSI Programming Sequence Example: [32] [33] [34] [35]

Display Subsystem Register Manual
- Display Subsystem Register Manual: [36]
- Display Subsystem Register Mapping Summary: [37]
- DSI Protocol Engine Registers: [38] [39]

Table 15-441. DSI_VCn_LONG_PACKET_PAYLOAD

Address Offset	0x0000 010C+ (n* 0x20)	**Index**	n = 0 to 3
Physical Address	0x4804 FD0C+ (n* 0x20)	**Instance**	DSI_PROTOCOL_ENGINE
Description	LONG PACKET PAYLOAD INFORMATION–virtual channel. This register sets the payload information (excluding Check-sum). Hardware must capture the word count in the packet header (in DSI_VCn_LONG_PACKET_HEADER register) to determine the last valid data (the VC ID can be different from VC). Byte1 is bit[7:0]; Byte2 is bit[15:8]; Byte3 is bit[23:16]; Byte4 is bit[31:24]; and Byten is sent before Byten+1 (least-significant byte first and least significant bit first).		
Type	W		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PAYLOAD																															

Bits	Field Name	Description	Type	Reset
31:0	PAYLOAD	Packet payload information (excluding check-sum)	W	0x00000000

Table 15-442. Register Call Summary for Register DSI_VCn_LONG_PACKET_PAYLOAD

Display Subsystem Functional Description
- DSI Transfer Modes: [0] [1] [2] [3] [4]
- Bus Turnaround: [5]
- PHY Triggers: [6] [7]

Display Subsystem Basic Programming Model
- Global Register Controls: [8]
- Packets: [9] [10] [11]
- Command Mode: [12] [13] [14] [15]
- DSI Programming Sequence Example: [16] [17]

Display Subsystem Register Manual
- Display Subsystem Register Manual: [18]
- Display Subsystem Register Mapping Summary: [19]
- DSI Protocol Engine Registers: [20]

Table 15-443. DSI_VCn_SHORT_PACKET_HEADER

Address Offset	0x0000 0110+ (n* 0x20)	**Index**	n = 0 to 3
Physical Address	0x4804 FD10+ (n* 0x20)	**Instance**	DSI_PROTOCOL_ENGINE
Description	SHORT PACKET HEADER INFORMATION -Virtual channel This register sets the 24-bit DATA_ID + Short packet data field + ECC (the VC ID can be different than VC) DATA_ID is located at bit[7:0] short packet data field is located at bit[23:8] ECC is located at bit[31:24] (least-significant byte first and least significant bit first)		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
HEADER																															

Bits	Field Name	Description	Type	Reset
31:0	HEADER	WRITES: Packet header information: DATA ID + DATA FIELD +ECC written into the TX FIFO READS: 32-bit values read from the RX FIFO	RW	0x00000000

Table 15-444. Register Call Summary for Register DSI_VCn_SHORT_PACKET_HEADER

Display Subsystem Environment
- Protocol Layer: [0]

Display Subsystem Functional Description
- DSI Transfer Modes: [1] [2] [3]
- PHY Triggers: [4]

Display Subsystem Basic Programming Model
- Global Register Controls: [5]
- Virtual Channels: [6]
- Packets: [7]
- Command Mode: [8] [9] [10] [11] [12]
- DSI Programming Sequence Example: [13] [14]

Display Subsystem Register Manual
- Display Subsystem Register Manual: [15]
- Display Subsystem Register Mapping Summary: [16]

Table 15-445. DSI_VCn_IRQSTATUS

Address Offset	0x0000 0118+ (n* 0x20)	**Index**	n = 0 to 3
Physical Address	0x4804 FD18+ (n* 0x20)	**Instance**	DSI_PROTOCOL_ENGINE
Description	INTERRUPT STATUS REGISTER - Virtual channel This register regroups all the events related to the VC.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																							PP_BUSY_CHANGE_IRQ	FIFO_TX_UDF_IRQ	ECC_NO_CORRECTION_IRQ	BTA_IRQ	FIFO_RX_OVF_IRQ	FIFO_TX_OVF_IRQ	PACKET_SENT_IRQ	ECC_CORRECTION_IRQ	CS_IRQ

Bits	Field Name	Description	Type	Reset
31:9	RESERVED	Write 0s for future compatibility. Reads returns 0.	RW	0x000000
8	PP_BUSY_CHANGE_IRQ	Video port ping-pong buffer busy status. 0x0: READS: Event is false. WRITES: Status bit unchanged. 0x1: READS: Event is true (pending). WRITES: Status bit is reset.	RW	0
7	FIFO_TX_UDF_IRQ	FIFO underflow status. The FIFO used on the slave port for buffering the data received on the OCP slave port for the VC has underflowed which means that the data for the current packet have not been received in time since the transfer of the packet are already started (transfer started since the packet size is bigger than space allocated in the FIFO). 0x0: READS: Event is false. WRITES: Status bit unchanged. 0x1: READS: Event is true (pending). WRITES: Status bit is reset.	RW	0x0
6	ECC_NO_CORRECTION_IRQ	ECC error status (short and long packets). No correction of the header because of more than 1-bit error. 0x0: READS: Event is false. WRITES: Status bit unchanged. 0x1: READS: Event is true (pending). WRITES: Status bit is reset.	RW	0x0
5	BTA_IRQ	Virtual channel - BTA status. 0x0: READS: Event is false. WRITES: Status bit unchanged. 0x1: READS: Event is true (pending). WRITES: Status bit is reset.	RW	0x0
4	FIFO_RX_OVF_IRQ	FIFO overflow error status. The FIFO used on the slave port for buffering the data received on the DSI link for the VC has overflowed. 0x0: READS: Event is false. WRITES: Status bit unchanged. 0x1: READS: Event is true (pending). WRITES: Status bit is reset.	RW	0x0

Bits	Field Name	Description	Type	Reset
3	FIFO_TX_OVF_IRQ	FIFO overflow error status. The FIFO used on the slave port for buffering the data received on the OCP slave port for the VC has overflowed. 0x0: READS: Event is false. WRITES: Status bit unchanged. 0x1: READS: Event is true (pending). WRITES: Status bit is reset.	RW	0x0
2	PACKET_SENT_IRQ	Indicates that a packet has been sent. It is used when BTA manual mode is used. 0x0: READS: Event is false. WRITES: Status bit unchanged. 0x1: READS: Event is true (pending). WRITES: Status bit is reset.	RW	0x0
1	ECC_CORRECTION_IRQ	Virtual channel - ECC has been used to do the correction of the only 1-bit error status (short and long packet only). 0x0: READS: Event is false. WRITES: Status bit unchanged. 0x1: READS: Event is true (pending). WRITES: Status bit is reset.	RW	0x0
0	CS_IRQ	Virtual channel - Check-Sum mismatch status. 0x0: READS: Event is false. WRITES: Status bit unchanged. 0x1: READS: Event is true (pending). WRITES: Status bit is reset.	RW	0x0

Table 15-446. Register Call Summary for Register DSI_VCn_IRQSTATUS

Display Subsystem Integration
- Interrupt Requests: [0]

Display Subsystem Functional Description
- DSI Transfer Modes: [1]
- Bus Turnaround: [2]

Display Subsystem Basic Programming Model
- Interrupts: [3]
- Command Mode: [4] [5] [6]
- DSI Programming Sequence Example: [7] [8]

Display Subsystem Register Manual
- Display Subsystem Register Mapping Summary: [9]

Table 15-447. DSI_VCn_IRQENABLE

Address Offset	0x0000 011C+ (n* 0x20)	**Index**	n = 0 to 3
Physical Address	0x4804 FD1C + (n* 0x20)	**Instance**	DSI_PROTOCOL_ENGINE
Description	INTERRUPT ENABLE REGISTER - Virtual channel This register regroups all the events related to VC.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																							PP_BUSY_CHANGE_IRQ_EN	FIFO_TX_UDF_IRQ_EN	ECC_NO_CORRECTION_IRQ_EN	BTA_IRQ_EN	FIFO_RX_OVF_IRQ_EN	FIFO_TX_OVF_IRQ_EN	PACKET_SENT_IRQ_EN	ECC_CORRECTION_IRQ_EN	CS_IRQ_EN

Bits	Field Name	Description	Type	Reset
31:9	RESERVED	Write 0s for future compatibility. Reads returns 0.	RW	0x000000
8	PP_BUSY_CHANGE_IRQ_EN	Video port ping-pong buffer busy. 0x0: Event is masked 0x1: Event generates an interrupt when it occurs	RW	0
7	FIFO_TX_UDF_IRQ_EN	FIFO underflow enable. The FIFO used on the slave port for buffering the data received on the OCP slave port for the VC has underflowed which means that the data for the current packet have not been received in time since the transfer of the packet are already started (transfer started since the packet size is bigger than space allocated in the FIFO). 0x0: Event is masked 0x1: Event generates an interrupt when it occurs	RW	0x0
6	ECC_NO_CORRECTION_IRQ_EN	ECC error (short and long packets). No correction of the header because of more than 1-bit error. 0x0: Event is masked 0x1: Event generates an interrupt when it occurs	RW	0x0
5	BTA_IRQ_EN	Virtual channel -Bus turn around reception 0x0: Event is masked 0x1: Event generates an interrupt when it occurs	RW	0x0
4	FIFO_RX_OVF_IRQ_EN	FIFO overflow enable. The FIFO used on the slave port for buffering the data received on the DSI link for the VC has overflowed. 0x0: Event is masked 0x1: Event generates an interrupt when it occurs	RW	0x0
3	FIFO_TX_OVF_IRQ_EN	FIFO overflow enable. The FIFO used on the slave port for buffering the data received on the OCP slave port for the VC has overflowed. 0x0: Event is masked 0x1: Event generates an interrupt when it occurs	RW	0x0
2	PACKET_SENT_IRQ_EN	Indicates that a packet has been sent. It is used when BTA manual mode is used. 0x0: Event is masked 0x1: Event generates an interrupt when it occurs	RW	0x0
1	ECC_CORRECTION_IRQ_EN	Virtual channel - ECC has been used to correct the only 1-bit error (short and long packet). 0x0: Event is masked 0x1: Event generates an interrupt when it occurs	RW	0x0
0	CS_IRQ_EN	Virtual channel - Check-Sum of the payload mismatch detection 0x0: Event is masked 0x1: Event generates an interrupt when it occurs	RW	0x0

Table 15-448. Register Call Summary for Register DSI_VCn_IRQENABLE

Display Subsystem Functional Description • DSI Transfer Modes: [0]
Display Subsystem Basic Programming Model • Interrupts: [1] • DSI Programming Sequence Example: [2] [3]
Display Subsystem Register Manual • Display Subsystem Register Mapping Summary: [4]

15.7.2.6 DSI complex I/O Registers

Table 15-449. DSI_PHY_CFG0

Address Offset	0x0000 0000		
Physical Address	0x4804 FE00	**Instance**	DSI_PHY_SCP
Description	Configuration register for HS mode timings		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
THS_PREPARE								THS_PREPARE_THS_ZERO								THS_TRAIL								THS_EXIT							

Bits	Field Name	Description	Type	Reset
31:24	THS_PREPARE	THS-PREPARE timing parameter in multiples of DDR clock period. DDR clock = CLKIN4DDR/4. Programmed value = CEIL(70 ns/DDR clock period) + 2	RW	0x1A
23:16	THS_PREPARE_ THS_ZERO	THS-PREPARE + THS-ZERO timing parameter in multiples of DDR clock period. DDR clock = CLKIN4DDR/4. Programmed value = ceil(175 ns/DDR clock period) + 2	RW	0x3c
15:8	THS_TRAIL	THS-TRAIL timing parameter in multiples of DDR clock period. DDR clock = CLKIN4DDR/4. Programmed value = ceil(60 ns/DDR clock period) + 5	RW	0x1A
7:0	THS_EXIT	Ths-exit timing parameter in multiples of DDR clock period. DDR clock = CLKIN4DDR/4). Programmed value = ceil(145 ns/DDR clock period).	RW	0x28

Table 15-450. Register Call Summary for Register DSI_PHY_CFG0

Display Subsystem Functional Description • Clock Requirements: [0] [1] [2] [3] [4] • Serial Configuration Port (SCP) Interface: [5]
Display Subsystem Basic Programming Model • Display Timing Configuration: [6] [7] [8] [9] [10]
Display Subsystem Use Cases and Tips • Configure DSI, DSI PLL and Complex I/O: [11] [12] [13] [14] • Display Subsystem Use Cases and Tips: [15] [16] [17] [18]
Display Subsystem Register Manual • Display Subsystem Register Mapping Summary: [19]

Table 15-451. DSI_PHY_CFG1

Address Offset	0x0000 0004		
Physical Address	0x4804 FE04	**Instance**	DSI_PHY_SCP
Description	Configuration register for LP mode and HS mode timings		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED			RESERVED		RESERVED			RESERVED	TLPX_HALF							TCLK_TRAIL								TCLK_ZERO							

Bits	Field Name	Description	Type	Reset
31:29	RESERVED	Reserved. Keep at reset value	RW	0x2
28:27	RESERVED	Reserved. Read returns zero. Write only zero for future compatibility	RW	0x0
26:24	RESERVED	Reserved. Keep at reset value	RW	0x2
23	RESERVED	Reserved. Read returns zero. Write only zero for future compatibility	RW	0x0
22:16	TLPX_HALF	(TLPX)/2 timing parameter in multiples of DDR clock frequency. DDR clock = CLKIN4DDR/4. Programmed value = ceil(25ns/DDR clock period) Default value : 10 (for 400 MHz) Note: TLPX is used to define the length of LP-01 state in HS Start of Transmission sequences on clock and data lanes. For all other purposes TLPX is defined by the period of TxLPEsc clock.	RW	0x0A
15:8	TCLK_TRAIL	TCLK-TRAIL timing parameter in multiples of DDR clock frequency. DDR clock = CLKIN4DDR/4. Programmed value = ceil(60 ns/DDR clock period) + 2	RW	0x18
7:0	TCLK_ZERO	TCLK-ZERO timing parameter in multiples of DDR clock period. DDR clock = CLKIN4DDR/4. Programmed value = ceil(260 ns/DDR clock period)	RW	0x75

Table 15-452. Register Call Summary for Register DSI_PHY_CFG1

Display Subsystem Functional Description
- Clock Requirements: [0] [1] [2] [3]
- Serial Configuration Port (SCP) Interface: [4]

Display Subsystem Basic Programming Model
- Display Timing Configuration: [5] [6] [7] [8]

Display Subsystem Use Cases and Tips
- Configure DSI, DSI PLL and Complex I/O: [9] [10] [11] [12]
- Display Subsystem Use Cases and Tips: [13] [14] [15] [16]

Display Subsystem Register Manual
- Display Subsystem Register Mapping Summary: [17]

Table 15-453. DSI_PHY_CFG2

Address Offset	0x0000 0008		
Physical Address	0x4804 FE08	**Instance**	DSI_PHY_SCP
Description	Sync pattern and reserved bits		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																								TCLK_PREPARE							

Bits	Field Name	Description	Type	Reset
31:8	RESERVED	Reserved. Do not modify the reset value.	RW	0xB80000
7:0	TCLK_PREPARE	TCLK-PREPARE timing parameter in multiples of DDR clock period. DDR clock = CLKIN4DDR/4. Programmed value = ceil(65 ns/DDR clock period)	RW	0x1B

Table 15-454. Register Call Summary for Register DSI_PHY_CFG2

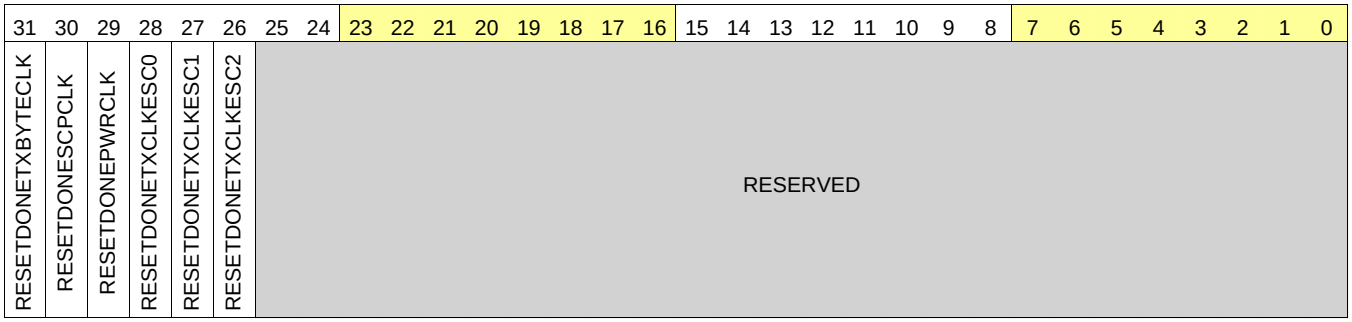
Display Subsystem Functional Description
- Clock Requirements: [0]

Display Subsystem Use Cases and Tips
- Configure DSI, DSI PLL and Complex I/O: [1] [2]
- Display Subsystem Use Cases and Tips: [3] [4]

Display Subsystem Register Manual
- Display Subsystem Register Mapping Summary: [5]

Table 15-455. DSI_PHY_CFG5

Address Offset	0x0000 0014		
Physical Address	0x4804 FE14	**Instance**	DSI_PHY_SCP
Description	Reset done bits		
Type	R		



Bits	Field Name	Description	Type	Reset
31	RESETDONETXBYTECLK	RESETDONETXBYTECLK 0x0: No reset 0x1: Reset done for the TXBYTECLK domain	R	0x0
30	RESETDONESCPCLK	RESETDONESCPCLK 0x0: No reset 0x1: Reset done for the SCP clock domain	R	0x0
29	RESETDONEPWRCLK	RESETDONEPWRCLK 0x0: No reset 0x1: Reset done for the PWR clock domain	R	0x0
28	RESETDONETXCLKESC0	RESETDONETXCLKESC0 0x0: No reset 0x1: Reset done for the TXCLKESC domain for lane 0	R	0x0
27	RESETDONETXCLKESC1	RESETDONETXCLKESC1 0x0: No reset 0x1: Reset done for the TXCLKESC domain for lane 1	R	0x0
26	RESETDONETXCLKESC2	RESETDONETXCLKESC2 0x0: No reset 0x1: Reset done for the TXCLKESC domain for lane 2	R	0x0
25:0	RESERVED	Read-Only register. Read returns zero	R	0x0000000

Table 15-456. Register Call Summary for Register DSI_PHY_CFG5

Display Subsystem Basic Programming Model • Reset-Done Bits: [0] [1] [2] [3] [4] [5] [6]
Display Subsystem Register Manual • Display Subsystem Register Mapping Summary: [7]

15.7.2.7 DSI PLL Control Module Registers

Table 15-457. DSI_PLL_CONTROL

Address Offset	0x0000 0000		
Physical Address	0x4804 FF00	**Instance**	DSI_PLL_CTRL_SCP
Description	This register controls the PLL reset/power and modes		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																											DSI_HSDIV_SYSRESET	DSI_PLL_SYSRESET	DSI_PLL_HALTMODE	DSI_PLL_GATEMODE	DSI_PLL_AUTOMODE

Bits	Field Name	Description	Type	Reset
31:5	RESERVED	Reserved. Write only zero for future compatibility. Reads return zero.	R	0x0000000
4	DSI_HSDIV_SYSRESET	Force HSDIVIDER SYSRESET 0x0: HSDIVIDER SYSRESET controlled by power FSM 0x1: HSDIVIDER SYSRESET forced active	RW	0x0
3	DSI_PLL_SYSRESET	Force ADPLLV2 SYSRESET 0x0: PLL SYSRESET controlled by power FSM 0x1: PLL SYSRESET forced active	RW	0x0
2	DSI_PLL_HALTMODE	Allow PLL to be halted if no activity 0x0: PLL will not be halted 0x1: PLL will be halted based on activity	RW	0x0
1	DSI_PLL_GATEMODE	Allow PLL clock gating for power saving 0x0: CLKIN4DDR on 0x1: CLKIN4DDR gated by DSI Protocol Engine activity	RW	0x0
0	DSI_PLL_AUTOMODE	Automatic update mode. If this bit is set then the configuration updates will be synchronized to DISPC_UPDATE_SYNC. If this bit is clear configuration updates will be done immediately. 0x0: Manual mode 0x1: Automatic mode	RW	0x0

Table 15-458. Register Call Summary for Register DSI_PLL_CONTROL

Display Subsystem Functional Description • Power Control: [0] [1]
Display Subsystem Basic Programming Model • DSI PLL Go Sequence: [2] [3] [4] [5]

Table 15-458. Register Call Summary for Register DSI_PLL_CONTROL (continued)

Display Subsystem Use Cases and Tips

- Configure DSI, DSI PLL and Complex I/O: [6]
- Display Subsystem Use Cases and Tips: [7]

Display Subsystem Register Manual

- Display Subsystem Register Mapping Summary: [8]

Table 15-459. DSI_PLL_STATUS

Address Offset	0x0000 0004		
Physical Address	0x4804 FF04	**Instance**	DSI_PLL_CTRL_SCP
Description	This register contains the status information		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																						DSI_BYPASSACKZ	DSIPROTO_CLOCK_ACK	DSS_CLOCK_ACK	DSI_PLL_BYPASS	DSI_PLL_HIGHJITTER	DSI_PLL_LIMP	DSI_PLL_LOSSREF	DSI_PLL_RECAL	DSI_PLL_LOCK	DSI_PLLCTRL_RESET_DONE

Bits	Field Name	Description	Type	Reset
31:10	RESERVED	Reserved. Reads return zero.	R	0x000000
9	DSI_BYPASSACKZ	State of bypass mode on PHY and HSDIVIDER 0x0: DSI_PHY and HSDIVIDER have switched to using the bypass clocks. 0x1: PLL outputs are still being used by DSI_PHY or HSDIVIDER	R	0x0
8	DSIPROTO_ CLOCK_ACK	Acknowledge for enable of DSI Protocol Engine clock Verify the status before selecting this source in the DSI Protocol Engine clock mux 0x0: DSI Protocol Engine clock inactive 0x1: DSI Protocol Engine clock active	R	0x0
7	DSS_CLOCK_ACK	Acknowledge for enable of DSS clock Verify the status before selecting this source in the DSS clock multiplexer 0x0: DSS clock inactive 0x1: DSS clock active	R	0x0
6	DSI_PLL_BYPASS	DSI PLL Bypass status 0x0: PLL not bypassing 0x1: PLL bypass	R	0x0
5	DSI_PLL_HIGHJITTER	DSI PLL High Jitter status 0x0: PLL in normal jitter condition 0x1: PLL in high jitter condition: Phase error > 24% (TIGHTPHASELOCK = 0) Phase error > 12% (TIGHTPHASELOCK = 1)	R	0x0
4	DSI_PLL_LIMP	DSI PLL Limp status 0x0: LIMP mode inactive 0x1: LIMP mode active	R	0x0

Bits	Field Name	Description	Type	Reset
3	DSI_PLL_LOSSREF	DSI PLL Reference Loss status 0x0: Reference input active 0x1: Reference input inactive	R	0x0
2	DSI_PLL_RECAL	DSI PLL re-calibration status If this bit is active, the PLL must be recalibrated 0x0: Recalibration is not required 0x1: Recalibration is required	R	0x0
1	DSI_PLL_LOCK	DSI PLL Lock status See the programming guide for the use of this bit 0x0: PLL is not locked 0x1: PLL is locked	R	0x0
0	DSI_PLLCTRL_RESET_DONE	DSI PLL Controller reset done status 0x0: Reset is in progress 0x1: Reset has completed	R	0x0

Table 15-460. Register Call Summary for Register DSI_PLL_STATUS

Display Subsystem Functional Description
- Error Handling: [0] [1]

Display Subsystem Basic Programming Model
- Software Reset: [2]
- DSI PLL Clock Gating Sequence: [3]
- DSI PLL Error Handling: [4] [5] [6] [7]

Display Subsystem Use Cases and Tips
- Configure DSI, DSI PLL and Complex I/O: [8]
- Display Subsystem Use Cases and Tips: [9]

Display Subsystem Register Manual
- Display Subsystem Register Mapping Summary: [10]

Table 15-461. DSI_PLL_GO

Address Offset	0x0000 0008		
Physical Address	0x4804 FF08	**Instance**	DSI_PLL_CTRL_SCP
Description	This register contains the GO bit		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																															DSI_PLL_GO

Bits	Field Name	Description	Type	Reset
31:1	RESERVED	Reserved. Write only zero for future compatibility. Reads return zero.	R	0x00000000
0	DSI_PLL_GO	Request (re-)locking sequence of the PLL. If the AutoMode bit is set, then this will be deferred until DISPC_UPDATE_SYNC goes active 0x0: No pending action 0x1: Request PLL (re-)locking/locking pending	RW	0x0

Table 15-462. Register Call Summary for Register DSI_PLL_GO

Display Subsystem Basic Programming Model
- DSI PLL Go Sequence: [0] [1]
- DSI PLL Clock Gating Sequence: [2] [3]
- DSI PLL Recommended Values: [4]

Display Subsystem Use Cases and Tips
- Configure DSI, DSI PLL and Complex I/O: [5] [6] [7] [8] [9]
- Display Subsystem Use Cases and Tips: [10] [11] [12] [13] [14]

Display Subsystem Register Manual
- Display Subsystem Register Mapping Summary: [15]
- DSI PLL Control Module Registers: [16] [17]

Table 15-463. DSI_PLL_CONFIGURATION1

Address Offset	0x0000 000C		
Physical Address	0x4804 FF0C	**Instance**	DSI_PLL_CTRL_SCP
Description	This register contains the latched PLL and HSDIVDER configuration bits		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED					DSIPROTO_CLOCK_DIV			DSS_CLOCK_DIV					DSI_PLL_REGM												DSI_PLL_REGN							DSI_PLL_STOPMODE

Bits	Field Name	Description	Type	Reset
31:27	RESERVED	Reserved. Write only zero for future compatibility. Reads return zero.	R	0x00
26:23	DSIPROTO_CLOCK_DIV	Divider value for DSI Protocol Engine clock source REGM4	RW	0x0
22:19	DSS_CLOCK_DIV	Divider value for DSS clock source REGM3	RW	0x0
18:8	DSI_PLL_REGM	M Divider for PLL	RW	0x000
7:1	DSI_PLL_REGN	N Divider for PLL (Reference)	RW	0x00
0	DSI_PLL_STOPMODE	DSI PLL STOPMODE 0x0: STOPMODE is not selected 0x1: STOPMODE is selected	RW	0x0

Table 15-464. Register Call Summary for Register DSI_PLL_CONFIGURATION1

Display Subsystem Functional Description
- DSI PLL Operations: [0] [1] [2]

Display Subsystem Basic Programming Model
- DSI PLL Lock Sequence: [3] [4] [5] [6] [7] [8] [9] [10] [11] [12] [13] [14]
- DSI PLL Recommended Values: [15]

Display Subsystem Use Cases and Tips
- Configure DSI, DSI PLL and Complex I/O: [16] [17] [18] [19] [20]
- Display Subsystem Use Cases and Tips: [21] [22] [23] [24] [25]

Display Subsystem Register Manual
- Display Subsystem Register Mapping Summary: [26]

Table 15-465. DSI_PLL_CONFIGURATION2

Address Offset	0x0000 0010		
Physical Address	0x4804 FF10	**Instance**	DSI_PLL_CTRL_SCP
Description	This register contains the unlatched PLL and HSDIVDER configuration bits These bits are "shadowed" when automatic mode is selected		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED											DSI_HSDIVBYPASS	DSI_PROTO_CLOCK_PWDN	DSI_PROTO_CLOCK_EN	DSS_CLOCK_PWDN	DSS_CLOCK_EN	DSI_BYPASSEN	DSI_PHY_CLKINEN	DSI_PLL_REFEN	DSI_PLL_HIGHFREQ	DSI_PLL_CLKSEL	DSI_PLL_LOCKSEL		DSI_PLL_DRIFTGUARDEN	DSI_PLL_TIGHTPHASELOCK	DSI_PLL_LOWCURRSTBY	DSI_PLL_PLLLPMODE	DSI_PLL_FREQSEL				DSI_PLL_IDLE

Bits	Field Name	Description	Type	Reset
31:21	RESERVED	Reserved. Write only zero for future compatibility. Reads return zero.	R	0x000
20	DSI_HSDIVBYPASS	Forces HSDIVIDER to bypass mode 0x0: HSDIVIDER in normal operation. Bypass controlled by PLL. 0x1: HSDIVIDER is forced to bypass mode.	RW	0x0
19	DSI_PROTO_CLOCK_PWDN	Power down for DSI Protocol Engine clock source 0x0: DSI Protocol Engine clock divider is active. 0x1: DSI Protocol Engine clock divider is powered down.	RW	0x0
18	DSI_PROTO_CLOCK_EN	Enable for DSI Protocol Engine clock source 0x0: DSI Protocol Engine clock divider is disabled. 0x1: DSI Protocol Engine clock divider is enabled.	RW	0x0
17	DSS_CLOCK_PWDN	Power down for DSS clock source 0x0: DSS clock divider is active. 0x1: DSS clock divider is powered down.	RW	0x0
16	DSS_CLOCK_EN	Enable for DSS clock source 0x0: DSS clock divider is disabled. 0x1: DSS clock divider is enabled.	RW	0x0
15	DSI_BYPASSEN	Selects DSS functional clock as CLKIN4DDR source 0x0: PLL controls CLKIN4DDR source: PLL DCO if PLL is locked DSS functional clock if not locked. 0x1: Force DSS functional clock to be used as CLKIN4DDR source	RW	0x0
14	DSI_PHY_CLKINEN	CLKIN4DDR control 0x0: CLKIN4DDR is disabled. 0x1:CLKIN4DDR is enabled.	RW	0x0
13	DSI_PLL_REFEN	PLL reference clock control 0x0: PLL reference clock disabled. 0x1: PLL reference clock enabled.	RW	0x0
12	DSI_PLL_HIGHFREQ	Enables a division of pixel clock by 2 before input to the PLL Required for pixel clock frequencies above 32 MHz (21 MHZ if N = 0) 0x0: Pixel clock is not divided. 0x1: Pixel clock is divided by 2.	RW	0x0

Bits	Field Name	Description	Type	Reset
11	DSI_PLL_CLKSEL	Reference clock selection 0x0: Selects DSS2_ALWON_FCLK as PLL reference clock 0x1: Selects Pixel Clock (PCLKFREE) as PLL reference clock	RW	0x0
10:9	DSI_PLL_LOCKSEL	Selects the lock criteria for the PLL 0x0: Phase Lock criteria depends on setting of DSI_PLL_TIGHTPHASELOCK bit 0x1: Frequency lock 0x2: Spare	RW	0x0
8	DSI_PLL_ DRIFTGUARDEN	DSI PLL DRIFTGUARDEN 0x0: Only RECAL flag is asserted in case of temperature drift. The programmer should take appropriate action. 0x1: Temperature drift will initiate automatic recalibration. RECAL flag will be asserted while this is taking place.	RW	0x0
7	DSI_PLL_ TIGHTPHASELOCK	DSI PLL Phase Lock criteria If this bit is set, the phase lock tolerance is reduced 0x0: Normal phase lock criteria Phase error lower than 6.4 % 0x1: Tightened phase lock criteria Phase error lower than 3.2 %	RW	0x0
6	DSI_PLL_ LOWCURRSTBY	PLL LOW CURRENT STANDBY 0x0: LOWCURRSTBY is not selected. 0x1: LOWCURRSTBY is selected.	RW	0x0
5	DSI_PLL_ PLLLPMODE	Select the power/performance of the PLL 0x0: Full performance, minimized jitter 0x1: Reduced power, increased jitter	RW	0x0
4:1	DSI_PLL_ FREQSEL	PLL internal reference frequency range selection 0x3: 0.75MHz to 1.0MHz 0x4: 1.0MHz to 1.25MHz 0x5: 1.25MHz to 1.5MHz 0x6: 1.5MHz to 1.75MHz 0x7: 1.75MHz to 2.1MHz 0xB: 7.5MHz to 10MHz 0xC: 10MHz to 12.5MHz 0xD: 12.5MHz to 15MHz 0xE: 15MHz to 17.5MHz 0xF: 17.5MHz to 21MHz	RW	0x0
0	DSI_PLL_IDLE	DSI PLL IDLE: 0x0: IDLE is not selected. 0x1: IDLE is selected.	RW	0x0

Table 15-466. Register Call Summary for Register DSI_PLL_CONFIGURATION2

Display Subsystem Integration
- Clocks: [0] [1]

Display Subsystem Functional Description
- DSI PLL Controller Architecture: [2] [3]
- DSI PLL Operations: [4] [5] [6]

Display Subsystem Basic Programming Model
- DSI PLL Go Sequence: [7] [8] [9]
- DSI PLL Lock Sequence: [10] [11] [12] [13] [14] [15]

Display Subsystem Use Cases and Tips
- Configure DSI, DSI PLL and Complex I/O: [16]
- Display Subsystem Use Cases and Tips: [17] [18] [19] [20] [21]

Table 15-466. Register Call Summary for Register DSI_PLL_CONFIGURATION2 (continued)

Display Subsystem Register Manual • Display Subsystem Register Mapping Summary: [22]

Timers

This chapter describes the timer modules for the OMAP34xx multimedia device.

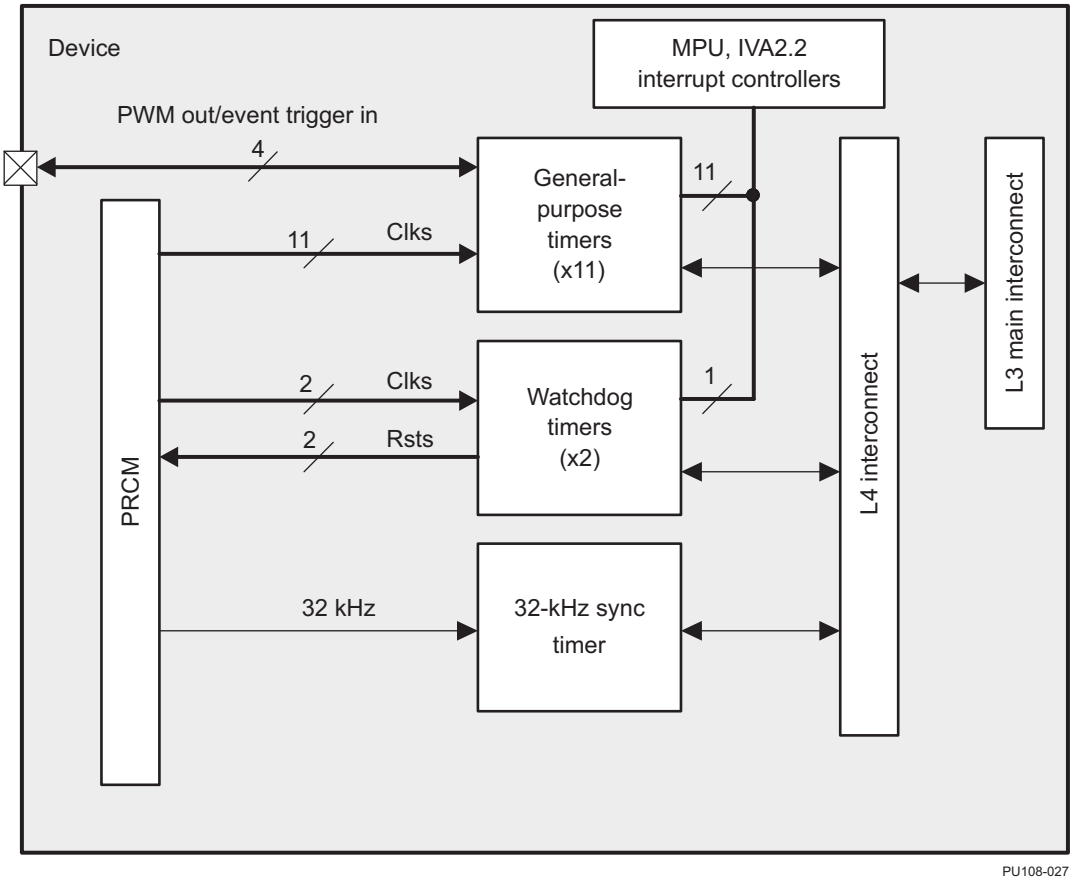
Topic **Page**

16.1 Timers Overview 2558
16.2 General-Purpose Timers 2559
16.3 General-Purpose Timers Register Manual 2580
16.4 Watchdog Timers 2602
16.5 Watchdog Timer Register Manual 2611
16.6 32-kHz Synchronized Timer 2619
16.7 32-kHz Sync Timer Register Manual 2621

16.1 Timers Overview

The device includes several types of timers used by the system software, including 11 general-purpose timers (GP timers), 2 watchdog timers (WDTs), a 32-kHz synchronized timer. Figure 16-1 shows the counters in the device in a high-level block diagram.

Figure 16-1. Timers



The 2 WDTs are clocked with 32-kHz clocks. The 32-kHz synchronized timer, which is reset only at power up, provides the operating system with a stable timing source that stores the relative time since the last power cycle of the product. Finally, 11 GP timers, which are useful simply as basic timers, are included to generate time-stamp-based interrupts to the system software or to use as a source of pulse-width modulation (PWM) signals.

16.2 General-Purpose Timers

16.2.1 GP Timers Overview

The device has 11 GP timers: GPTIMER1 through GPTIMER11.

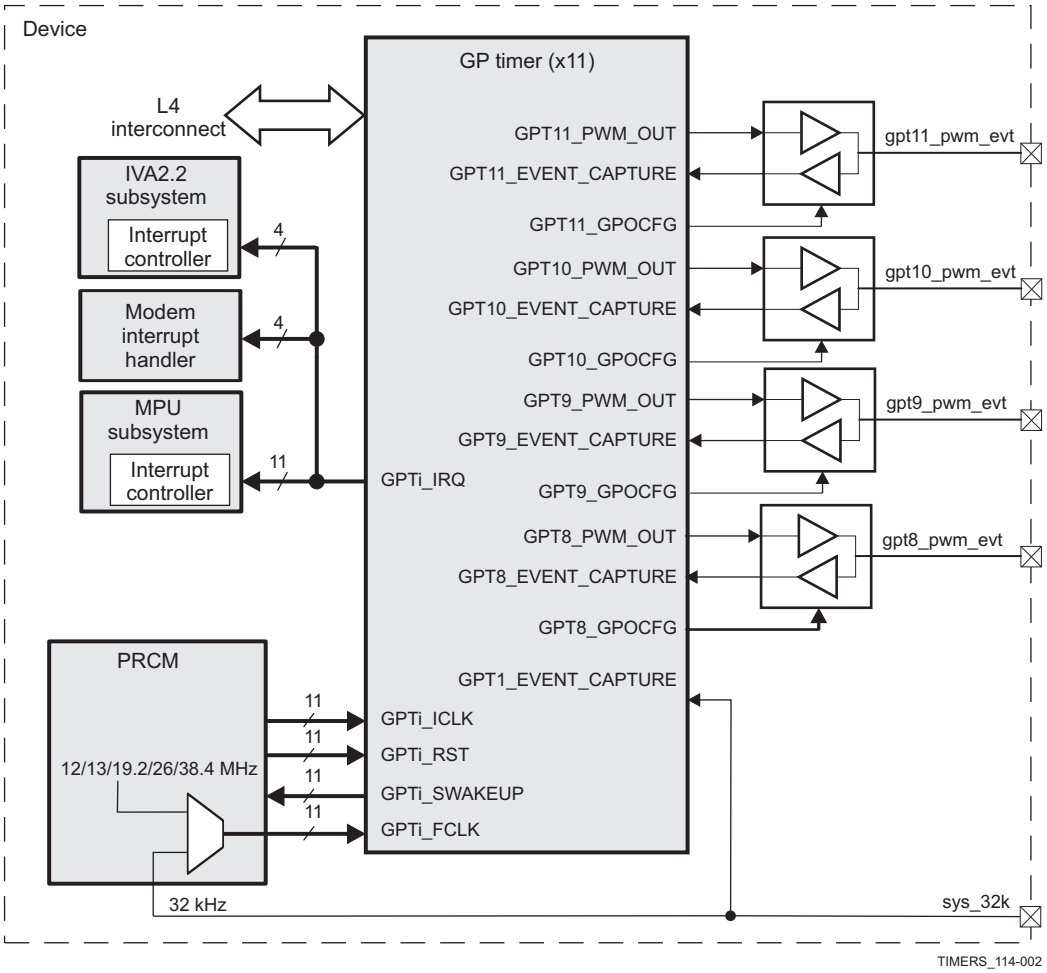
Each timer can be clocked from either the system clock (12, 13, 16.8, 19.2, 26, or 38.4 MHz) or the 32-kHz clock. The selection of the clock source is made at the power, reset, clock management (PRCM) module level. For more information, see Section 16.2.3.1, *Clocking, Reset, and Power-Management Scheme*.

GPTIMER1 has its GPT1_EVENT_CAPTURE pin tied to the 32-kHz clock and can be used to gauge the system clock input; it detects its frequency among 12, 13, 16.8, 19.2, 26, or 38.4 MHz.

Each timer can provide an interrupt to the microprocessor unit (MPU) subsystem. In addition, GPTIMER5 through GPTIMER8 also have interrupts connected to the IVA2.2 subsystem.

GPTIMER1, GPTIMER2, and GPTIMER10 include specific functions to generate accurate tick interrupts to the operating system. GPTIMER8 through GPTIMER11 are connected to external pins by their PWM output or their event capture input pin (for external timer triggering). Figure 16-2 shows an overview of the GP timers.

Figure 16-2. GP Timers Overview



16.2.1.1 GP Timers Features

The following are the main features of the GP timers controllers:

- L4 slave interface support:

  - 32-bit data bus width
  - 32-/16-bit access supported
  - 8-bit access not supported
  - 10-bit address bus width
  - Burst mode not supported
  - Write nonposted transaction mode supported
- Interrupts generated on overflow, compare, and capture
- Free-running 32-bit upward counter
- Compare and capture modes
- Autoreload mode
- Start/stop mode
- Programmable divider clock source (2^n with n = [0:8])
- Dedicated input trigger for capture mode and dedicated output trigger/PWM signal
- Dedicated output signal for general-purpose using GPTi_GPOCFG signal
- On-the-fly read/write register (while counting)
- 1-ms tick with 32,768 Hz functional clock generated (only GPTIMER1, GPTIMER2, and GPTIMER10)

16.2.2 GP Timers Environment

16.2.2.1 GP Timers External System Interface

Four of the 11 GP timers can send or receive stimulus to/from the external (off-chip) system. In the device, however, only GPTIMER8 through GPTIMER11 are configured to output a PWM pulse or receive an external event signal used as a trigger to capture the current timer count. GPTIMER1 is also configured to receive an event trigger input (GPT1_EVENT_CAPTURE) tied to the internal 32-kHz clock. This event signal gauges the system clock input; it detects its frequency among 12, 13, 16.8, 19.2, 26, or 38.4 MHz.

Figure 16-3 shows the external system interface for the GP timers, and Table 16-1 describes the GP timer inputs and outputs.

Figure 16-3. GP Timers External System Interface

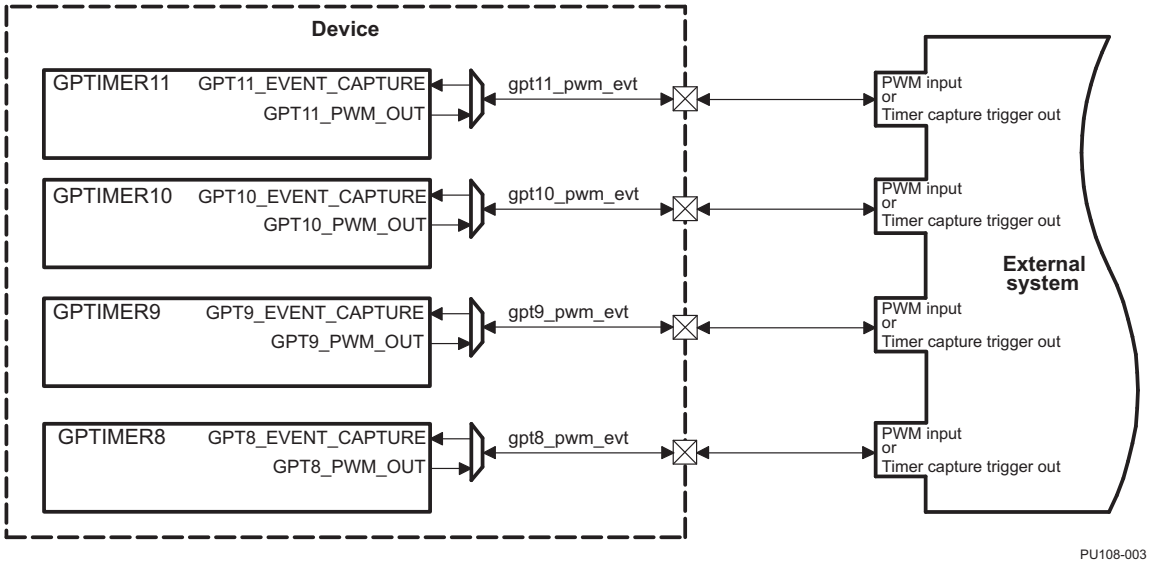


Table 16-1. Input/Output Description

Pin Name	Type [1]	Reset Value	Signal Name	Description
gpt8_pwm_evt	I/O	0	GPT8_EVENT_CAPTURE GPT8_PWM_OUT	GPTIMER8 trigger input/ PWM output
gpt9_pwm_evt	I/O	0	GPT9_EVENT_CAPTURE GPT9_PWM_OUT	GPTIMER9 trigger input/ PWM output
gpt10_pwm_evt	I/O	0	GPT10_EVENT_CAPTURE GPT10_PWM_OUT	GPTIMER10 trigger input/ PWM output
gpt11_pwm_evt	I/O	0	GPT11_EVENT_CAPTURE GPT11_PWM_OUT	GPTIMER11 trigger input/ PWM output

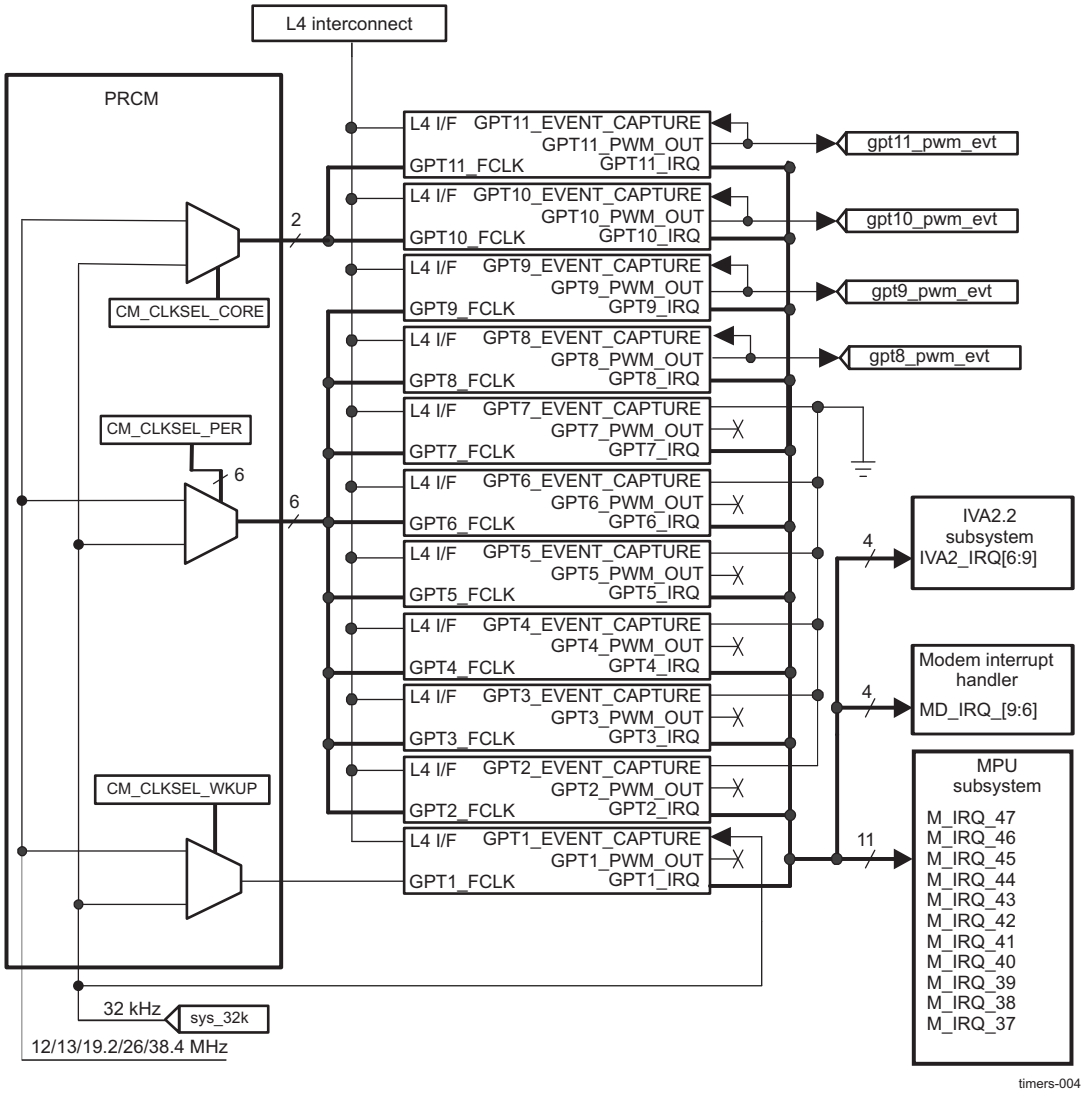
(1) When configured for that function; I = input, O = output

Note: The event trigger input (GPTi_EVENT_CAPTURE) for GPTIMER2 through GPTIMER7 is internally tied low, and the PWM output (GPTi_PWM_OUT) is not connected.

16.2.3 GP Timers Integration

Figure 16-4 shows the GP timer integration in the device.

Figure 16-4. GP Timer Integration



16.2.3.1 Clocking, Reset, and Power-Management Scheme

16.2.3.1.1 Clock Management

There are two clock domains in the GP timers:

- Functional clock domain: GPTi_FCLK is the GP timer functional clock. It is used to clock the GP timer internal logic.
- Interface clock domain: GPTi_ICLK is the GP timer interface clock. It is used to synchronize the GP timer L4 port to the L4 interconnect. All accesses from the interconnect are synchronous to GPTi_ICLK.

Table 16-2 lists the source clocks for each GP timer in the device. For more information on clock control and domains, see Chapter 4, *Power, Reset, and Clock Management*.

Table 16-2. Clock, Power, and Reset Domains for GP Timers

Timer	Interface Clock	Functional Clock	Power Domain
GPTIMER1	WKUP_L4_ICLK	GPT1_FCLK	WKUP
GPTIMER2 through GPTIMER9	PER_L4_ICLK	GPTx_ALWON_FCLK (x = 2 through 9)	PER
GPTIMER10 and 11	CORE_L4_ICLK	GPTx_FCLK (x = 10 or 11)	CORE

GP timers can be selected as the source of GPTi_FCLK (32-kHz clock or 12, 13, 16.8, 19.2, 26, or 38.4 MHz) at the PRCM level in the corresponding registers. For more information, see Chapter 4, *Power, Reset, and Clock Management*. Table 16-3 lists the GP timer PRCM clock selection bits.

Table 16-3. GP Timer PRCM Clock Selection Bits

Name	Associated PRCM Clock Output	Selection Bit
GPT1_FCLK	GPT1_FCLK	PRCM.CM_CLKSEL_WKUP[0] CLKSEL_GPT1
GPT[2:9]_FCLK	GPT[2:9]_ALWON_FCLK	PRCM.CM_CLKSEL_PER [0:7] CLKSEL_GPT[2:9]
GPT10_FCLK	GPT10_FCLK	PRCM.CM_CLKSEL_CORE [6] CLKSEL_GPT10
GPT11_FCLK	GPT11_FCLK	PRCM.CM_CLKSEL_CORE [7] CLKSEL_GPT11

From a global system power-management perspective, when one or both of the GP timer clocks is no longer required, the GP timers can be deactivated at the PRCM level in the corresponding registers. Table 16-4 lists the GP timer PRCM clock control bits.

Table 16-4. GP Timer PRCM Clock Control Bits

Name	Associated PRCM Clock Output	Enable Bit	Autoidle Bit
GPT1_FCLK	GPT1_FCLK	PRCM.CM_FCLKEN_WKUP[0] EN_GPT1 bit	N/A
GPT[2:9]_FCLK	GPT[2:9]_ALWON_FCLK	PRCM.CM_FCLKEN_PER [3:10] EN_GPT[2:9] bit	N/A
GPT10_FCLK	GPT10_FCLK	PRCM.CM_FCLKEN1_CORE [11] EN_GPT10 bit	N/A
GPT11_FCLK	GPT11_FCLK	PRCM.CM_FCLKEN1_CORE [12] EN_GPT11 bit	N/A
GPT1_ICLK	WKUP_L4_ICLK	PRCM.CM_ICLKEN_WKUP[0] EN_GPT1 bit	PRCM.CM_AUTOIDLE_WKUP[0] AUTO_GPT1 bit
GPT[2:9]_ICLK	PER_L4_ICLK	PRCM.CM_ICLKEN_PER[3:10] EN_GPT[2:9] bit	PRCM.CM_AUTOIDLE_PER[3:10] AUTO_GPT[2:9] bit
GPT10_ICLK	CORE_L4_ICLK	PRCM.CM_ICLKEN1_CORE[11] EN_GPT10 bit	PRCM.CM_AUTOIDLE1_CORE[11] AUTO_GPT10 bit
GPT11_ICLK		PRCM.CM_ICLKEN1_CORE[12] EN_GPT11 bit	PRCM.CM_AUTOIDLE1_CORE[12] AUTO_GPT11 bit

Notes:

- The PRCM function clock outputs are gated at the PRCM level, assuming all the modules that share it are disabled in the corresponding register. Disabling GP timers is a necessary but not sufficient action. The clock is effectively out when all PRCM conditions are fulfilled. For the other actions to be taken, see Chapter 4, *Power, Reset, and Clock Management*.
- The PRCM interface clock outputs are gated at the PRCM level, assuming all the modules that share it are disabled in the corresponding register. Disabling GP timers is a necessary but not sufficient action. The clock is effectively out when all PRCM conditions are fulfilled. For the other actions to be taken, see Chapter 4, *Power, Reset, and Clock Management*.
- The PRCM AUTOIDLE bits are used to link/unlink the GP timers from the clock domain transitions related to the GPTi_ICLK clocks.
- For further details about source clock gating and domain transitions, see Chapter 4, *Power, Reset, and Clock Management.*

GP timer modules also have an internal bit, GPTi.TIOCP_CFG[0] AUTOIDLE. The GP timer AUTOIDLE bit is used to apply an internal interface clock gating strategy.

At the PRCM level, when all conditions to shut off the PRCM functional or interface output clocks are met (see Chapter 4, *Power, Reset, and Clock Management*, for details), the PRCM automatically launches a hardware handshake protocol to ensure the GP timer is ready to have its clocks switched off. Namely, the PRCM asserts an IDLE request to the GP timer.

Although this handshake is a hardware function and is out of software control, the way the GP timer acknowledges the PRCM IDLE request is configurable through the GPTi.TIOCP_CFG[4:3] IDLEMODE bit. Table 16-5 lists the IDLEMODE settings and the related acknowledgement modes.

Table 16-5. IDLEMODE Settings

IDLEMODE Value	Selected Mode	Description
00	Force-idle	The GP timer acknowledges unconditionally the IDLE request from the PRCM module, regardless of its internal operations. This mode must be used carefully, because it does not prevent the loss of data when the clock is switched off.
01	No-idle	The GP timer never acknowledges an IDLE request from the PRCM module. This mode is safe from a module point of view, because it ensures that the clocks remain active. It is not efficient from a power-saving perspective, however, because it does not allow the PRCM output clock to be shut off and thus the power domain to be set to a lower power state.
10	Smart-idle	The GP timer acknowledges the IDLE request, basing its decision on its internal activity. The acknowledge signal is asserted only when all pending transactions and IRQ requests are treated. This is the best approach for efficient system power management.
11	Reserved	

When configured in smart-idle mode, the GP timer also offers an additional granularity on GPTi_FCLK and GPTi_ICLK gating. The GPTi.TIOCP_CFG[9:8] CLOCKACTIVITY bit field is used to determine which clock will be shut down (GPTi_FCLK, GPTi_ICLK, neither of them, or both of them).

The CLOCKACTIVITY setting is used internally to the GP timer to determine the part of the module on which the conditions to acknowledge the PRCM IDLE request will be tested. For example, if GPTi_FCLK is not to be shut down on a PRCM IDLE request, the GP timer considers only GPTi_ICLK and the associated pending activities before acknowledging the request.

Some GP timer features are associated with GPTi_ICLK and others are associated with GPTi_FCLK. Using the CLOCKACTIVITY setting along with the smart-idle mode ensures that the features associated with the clock that will remain active are always enabled, even if the GP timer acknowledges an IDLE request.

Table 16-6 lists the CLOCKACTIVITY settings and the associated features.

Table 16-6. CLOCKACTIVITY Settings

CLOCKACTIVITY Value	GPTi_ICLK Effects	GPTi_FCLK Effects	Description	Associated Features
00	OFF	OFF	Both GPTi_ICLK and GPTi_FCLK are considered for generating the acknowledge. This setting also means both GPTi_FCLK and GPTi_ICLK are likely to be shut down on PRCM IDLE request.	The idle acknowledge signal is asserted when there are no pending activities on the functional clock domain (improved latency in assertion of idle acknowledge). The wake-up capability of the GP timer is disabled.
01	ON	OFF	GPTi_ICLK is not shut down on PRCM IDLE request; only GPTi_FCLK is affected.	
10	OFF	ON	GPTi_FCLK is not shut down on PRCM IDLE request; only GPTi_ICLK is affected.	The idle acknowledge signal is asserted when there are no pending activities on the interface clock domain, without evaluating the pending activities on the functional clock domain (the GP timer enters into sleep mode, and if a pending interrupt event is finished during idle mode, the wake-up signal is asserted). The wake-up signal is enabled.
11	ON	ON	None of the clocks are shut down. Therefore, the GP timer can potentially acknowledge the IDLE request without checking the internal functionalities linked to its clocks.	

CAUTION

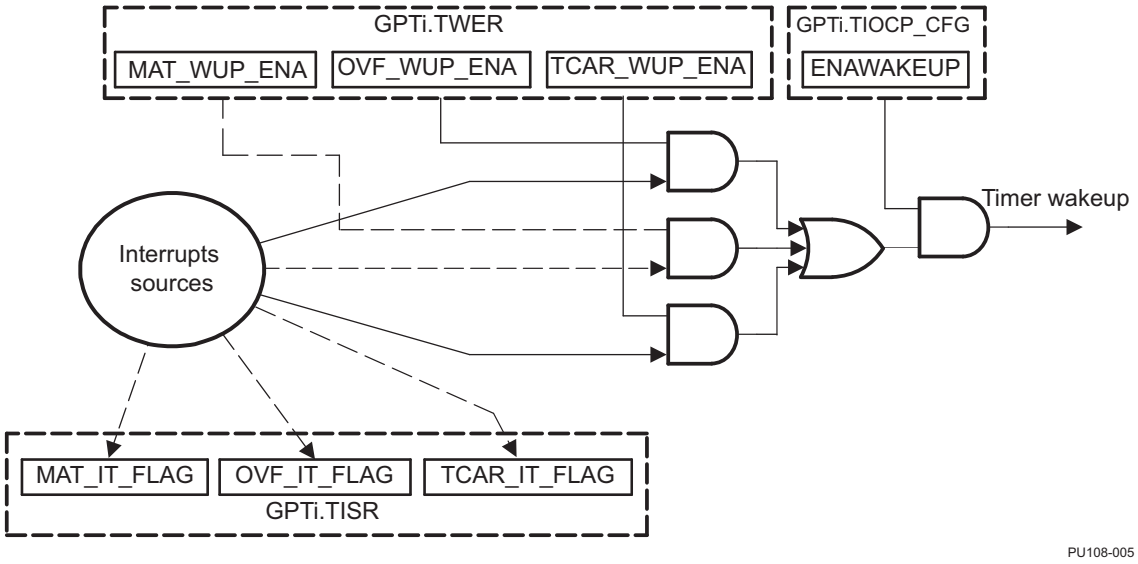
The PRCM module does not have any hardware means to read the CLOCKACTIVITY settings. The software must ensure consistent programming between the GP timer CLOCKACTIVITY and the PRCM functional clock and interface clock control bits. If the GP timer is disabled in both CM_FCLKEN and CM_ICLKEN PRCM registers while CLOCKACTIVITY is set to 11, nothing prevents the PRCM module from asserting its IDLE request, which is acknowledged regardless of the features associated with the GP timer clocks. This can lead to unpredictable behavior.

16.2.3.1.2 Wake-Up Capability

If the GPTi.TIOCP_CFG[4:3] IDLEMODE bit field sets the smart-idle mode, the timer module evaluates its internal capability to have the interface clock switched off. When there is no more internal activity (no pending interrupt sources: match, overflow, or timer capture events), the idle acknowledge signal is asserted and the timer enters into sleep mode, ready to issue a wake-up request. This wake-up request is sent only if the GPTi.TIOCP_CFG[2] ENAWAKEUP bit enables the timer wake-up capability.

Figure 16-5 shows the wake-up request generation. For more information on the GP timer clock control, see Section 16.2.3.1.1, *Clock Management*.

Figure 16-5. Wake-Up Request Generation



The timer wake-up enable register (GPTi.TWER) allows masking the expected source of the wake-up event that generates a wake-up request. The GPTi.TWER register is programmed synchronously with the interface clock before the PRCM module sends an idle mode request. The expected source of the wake-up event is an overflow (GPTi.TCRR), a timer match (the compare result of GPTi.TCRR and GPTi.TMAR matches the counter value), and a timer capture (an external pulse transition of the correct polarity is detected on the GPTi_EVENT_CAPTURE).

When the wake-up event is issued, the associated interrupt status bit is set in the timer status register (GPTi.TISR). The pending wake-up event is reset when the set status bit is overwritten by 1.

Note: The status bit must be reset to re-enter idle mode.

16.2.3.1.3 Reset and Power Management

GPTIMER1 belongs to the WKUP power domain. As part of that domain, this GP timer is sensitive to a WKUP_RST signal issued by the PRCM module. For further details about the WKUP power domain implementation and the WKUP_RST signal, see Chapter 4, *Power, Reset, and Clock Management*.

GPTIMER2 through GPTIMER9 belong to the PER power domain. As part of that domain, these GP timers are sensitive to a PER_RST signal issued by the PRCM module. For further details about the PER power domain implementation and the PER_RST signal, see Chapter 4, *Power, Reset, Clock Management*.

GPTIMER10 and GPTIMER11 belong to the CORE power domain. As part of that domain, these GP timers are sensitive to a CORE_RST signal issued by the PRCM module. For further details about the CORE power domain implementation and the CORE_RST signal, see Chapter 4, *Power, Reset, and Clock Management*.

16.2.3.2 Software Reset

Two bit fields (GPTi.TIOCP_CFG[1] SOFTRESET and GPTi.TSICR[1] SFT) can generate a software reset of the GP timer. For both of these bits, all read accesses return 0.

The GPTi.TIOCP_CFG[1] SOFTRESET bit allows resetting the functional and interface domains. The GPTi.TSICR[1] SFT bit allows resetting the functional part of the GP timer.

Before accessing or using the GP timer, the local host must ensure that both internal resets are released by reading the GPTi.TISTAT[0] RESETDONE bit. This bit field monitors the internal reset status.

16.2.3.3 GP Timer Interrupts

Table 16-7 lists the interrupt mapping from the 12 GP timers to the three internal processors.

Table 16-7. Timer Interrupt Names and Processor IRQ Mapping

Timer	Interrupt Name	Mapping	Comments
GPTIMER 1	GPT1_IRQ	M_IRQ_37	GPTIMER1 interrupt to MPU
GPTIMER2	GPT2_IRQ	M_IRQ_38	GPTIMER2 interrupt to MPU
GPTIMER3	GPT3_IRQ	M_IRQ_39	GPTIMER3 interrupt to MPU
GPTIMER4	GPT4_IRQ	M_IRQ_40	GPTIMER4 interrupt to MPU
GPTIMER5	GPT5_IRQ	M_IRQ_41	GPTIMER5 interrupt to MPU
		IVA2_IRQ[6]	GPTIMER5 interrupt to IVA2.2
GPTIMER6	GPT6_IRQ	M_IRQ_42	GPTIMER6 interrupt to MPU
		IVA2_IRQ[7]	GPTIMER6 interrupt to IVA2.2
		MD_IRQ_6	GPTIMER6 interrupt to modem subsystem (D2D)
GPTIMER7	GPT7_IRQ	M_IRQ_43	GPTIMER7 interrupt to MPU
		IVA2_IRQ[8]	GPTIMER7 interrupt to IVA2.2
		MD_IRQ_7	GPTIMER7 interrupt to modem subsystem (D2D)
GPTIMER8	GPT8_IRQ	M_IRQ_44	GPTIMER8 interrupt to MPU
		IVA2_IRQ[9]	GPTIMER8 interrupt to IVA2.2
		MD_IRQ_8	GPTIMER8 interrupt to modem subsystem (D2D)
GPTIMER9	GPT9_IRQ	M_IRQ_45	GPTIMER9 interrupt to MPU
		MD_IRQ_9	GPTIMER9 interrupt to modem subsystem (D2D)
GPTIMER10	GPT10_IRQ	M_IRQ_46	GPTIMER10 interrupt to MPU
GPTIMER11	GPT11_IRQ	M_IRQ_47	GPTIMER11 interrupt to MPU

The timer can issue an overflow interrupt, a timer match interrupt, and a timer capture interrupt. Each internal interrupt source can be independently enabled/disabled in the interrupt enable register (GPTi.TIER). When the interrupt event is issued, the associated interrupt status bit is set in the timer status register (GPTi.TISR). The pending interrupt event is reset when the set status bit is overwritten by a1.

16.2.4 GP Timers Functional Description

Each GP timer contains a free-running upward counter with autoreload capability on overflow. The timer counter can be read and written on-the-fly (while counting). Each GP timer includes compare logic to allow an interrupt event on a programmable counter matching value. A dedicated output signal can be pulsed or toggled on either an overflow or a match event. This offers time-stamp trigger signaling or PWM signal sources. A dedicated input signal can be used to trigger an automatic timer counter capture or an interrupt event on a programmable input signal transition type. A programmable clock divider (prescaler) allows reduction of the timer input clock frequency. All internal timer interrupt sources are merged into one module interrupt line and one wake-up line. Each internal interrupt source can be independently enabled/disabled with a dedicated bit of the GPTi.TIER register for the interrupt features and a dedicated bit of the GPTi.TWER register for the wakeup. In addition, GPTIMER1, GPTIMER2, and GPTIMER10 have implemented a mechanism to generate an accurate tick interrupt.

For each GP timer implemented in the device, there are two possible clock sources:

- 32-kHz clock
- System clock

Selection of the input clock source is done in the registers in the PRCM configuration (see Section 16.2.1, *GP Timer Overview*).

Each GP timer supports three functional modes:

- Timer mode
- Capture mode
- Compare mode

By default, after core reset, the capture and compare modes are disabled.

16.2.4.1 GP Timers Block Diagram

Figure 16-6 shows the block diagram of common GP timers, and Figure 16-7 shows the block diagram of GP timers with 1-ms tick generation module.

Figure 16-6. Block Diagram of GPTIMER3 through GPTIMER9 and GPTIMER11

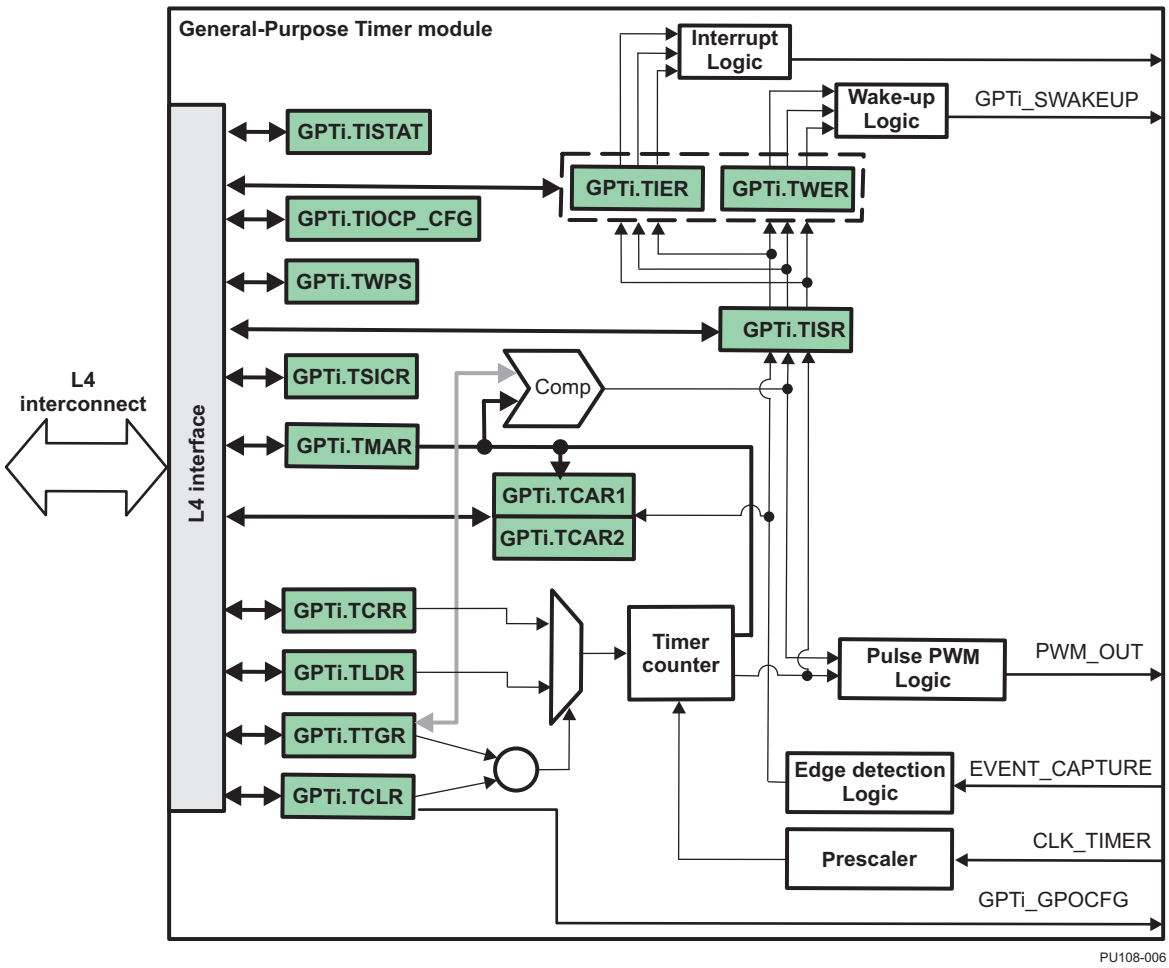
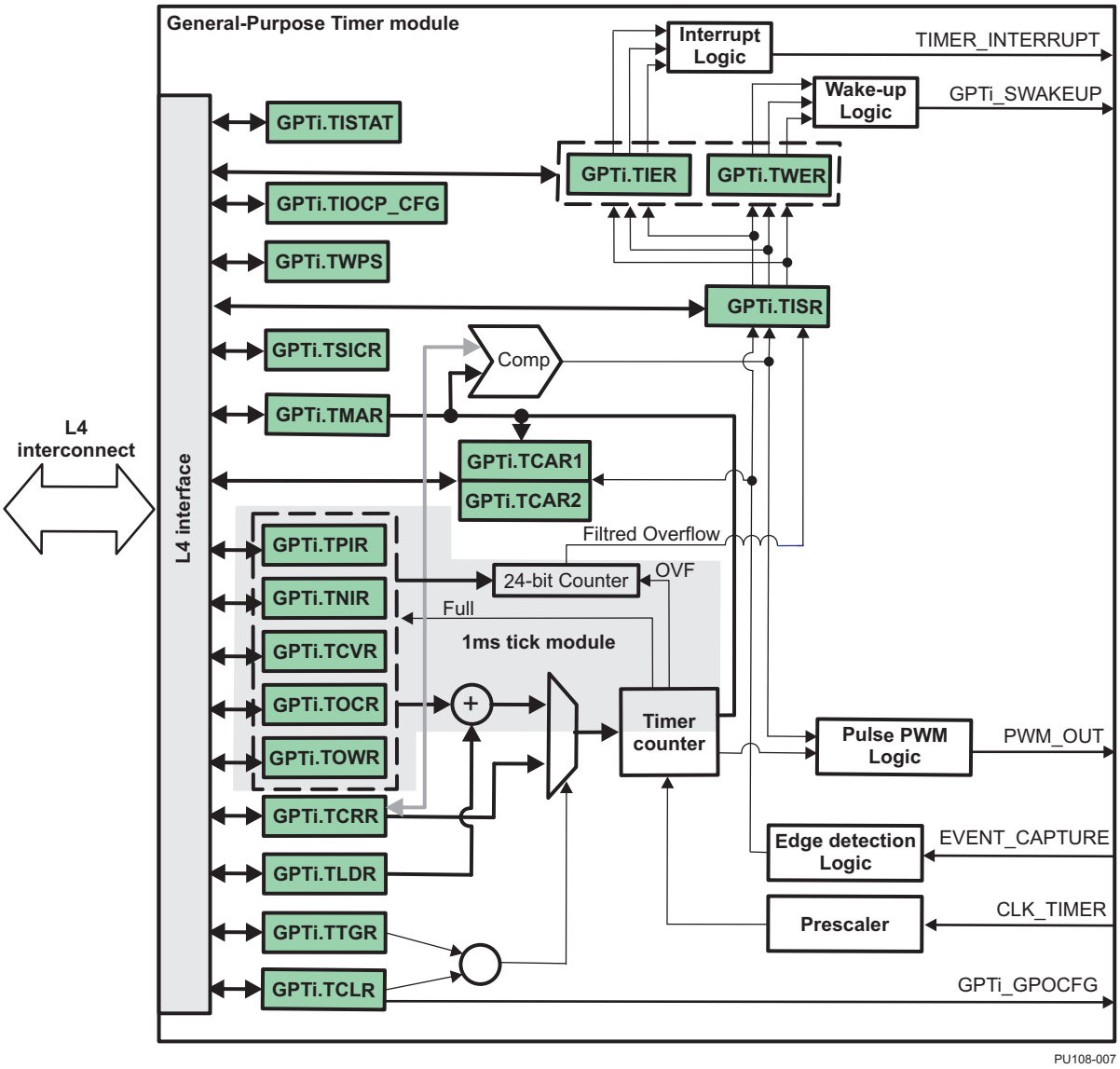


Figure 16-7. Block Diagram of GPTIMER1, GPTIMER2, and GPTIMER10



16.2.4.2 Timer Mode Functionality

The timer is an upward counter that can be started and stopped at any time through the timer control register (GPTi.TCLR[0] ST bit). The timer counter register (GPTi.TCRR) can be loaded when stopped or on-the-fly (while counting). GPTn.TCRR can be loaded directly by a GPTi.TCRR write access with a new timer value. The GPTi.TCRR register can also be loaded with the value held in the timer load register GPTi.TLDR by a trigger register (GPTi.TTGR) write access. The GPTi.TCRR loading is done regardless of the GPTi.TTGR written value. The timer counter register GPTi.TCRR value can be read when stopped or captured on-the-fly by a GPTi.TCRR read access. The timer is stopped and the counter value is set to 0 when the module reset is asserted. The timer is maintained at stop after the reset is released.

In one-shot mode (the GPTi.TCLR[1] AR bit set to 0), the counter is stopped after counting overflow occurs (the counter value remains at 0).

When the autoreload mode is enabled (the GPTi.TCLR[1] AR bit set to 1), the GPTi.TCRR register is reloaded with the timer load register (GPTi.TLDR) value after a counting overflow occurs.

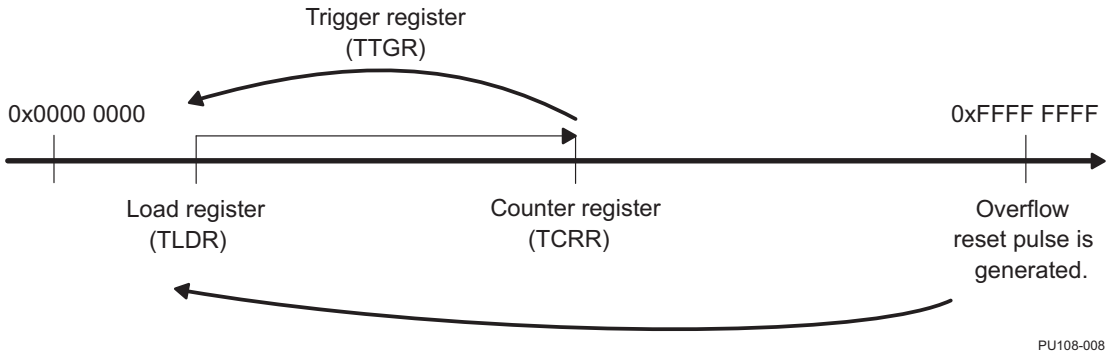
CAUTION

Do not put the overflow value (0xFFFFFFFF) in the GPTi.TLDR register because it can lead to undesired results.

An interrupt can be issued on overflow if the overflow interrupt enable bit is set in the timer interrupt enable register (GPTi.TIER[1] OVF_IT_ENA bit set to 1), the interrupt is enabled after 10 * GPTi.ICLK clock cycles. A dedicated output pin (timer PWM) can be programmed in GPTi.TCLR[12] through GPTi.TCLR[11:10] (PT and TRG bits) to generate one positive pulse (prescaler duration) or to invert the current value (toggle mode) when an overflow occurs. The GPTi.TCLR[12] PT bit selects pulse/toggle modulation (GPTi.TCLR[11:10] TRG bit field select trigger mode).

Figure 16-8 shows the GPTi.TCRR timing value.

Figure 16-8. GPTi.TCRR Timing Value



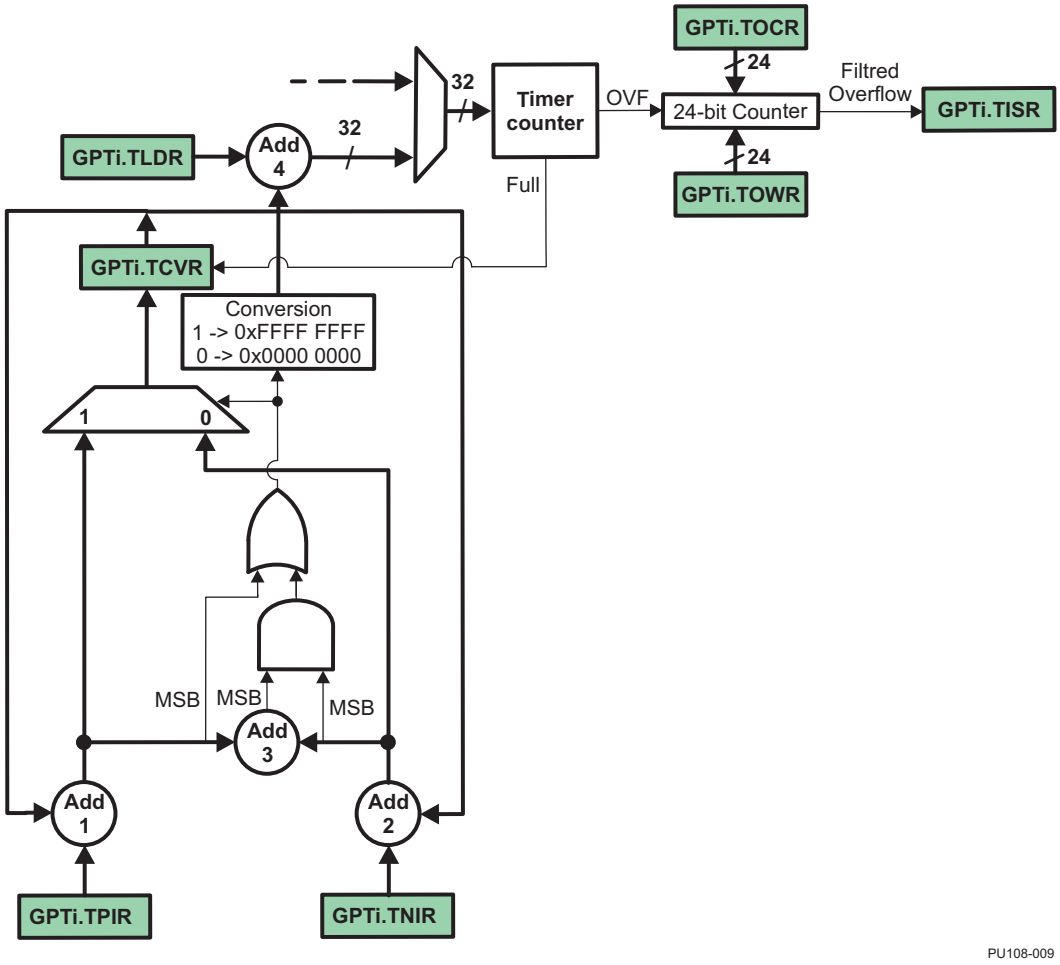
16.2.4.2.1 1-ms Tick Generation (Only GPTIMER1, GPTIMER2, and GPTIMER10)

Because the timer input clock is 32,768 Hz, the interrupt period is not exactly 1 ms. If the clock counts up to 32, it obtains a 0.977-ms period; if it counts up to 33, it obtains a 1.007-ms period. For large granularity, the error is cumulative and can generate important deviations to the standard value.

To minimize the error between a true 1-ms tick and the tick generated by the 32,768 Hz timer, the sequencing of periods less than 1 ms and periods greater than 1 ms must be shuffled. An additional 1-ms block is used to correct this error. Refer to Figure 16-9.

In this implementation, the increment sequencing is automatically managed by the timer to minimize the error. The user must define only the value of the timer positive increment register (GPTi.TPIR[31:0] POSITIVE_INC_VALUE bit field) and the timer negative increment register (GPTi.TNIR[31:0] NEGATIVE_INC_VALUE bit field). An automatic adaptation mechanism is used to simplify the programming model.

Figure 16-9. Block Diagram of the 1-ms Tick Module



The GPTi.TPIR, GPTi.TNIR, and GPTi.TCVR registers and adders Add1, Add2, and Add3 are used to define whether the next value loaded in the timer counter register (GPTi.TCRR[31:0] TIMER_COUNTER bit field) is the value of the GPTi.TLDR[31:0] LOAD_VALUE bit field (period less than 1 ms) or the value of GPTi.TLDR[31:0] LOAD_VALUE - 1 (period greater than 1 ms).

Table 16-8 lists the value loaded in the GPTi.TCRR register according to the sign of the result of Add1, Add2, and Add3.

MSB = 0: Positive value, MSB = 1: Negative value

Table 16-8. Value Loaded in GPTi.TCRR to Generate 1-ms Tick

Add1 MSB	Add2 MSB	Add3 MSB	Value of GPTi.TCRR Register
0	0	0	GPTi.TLDR[31:0] LOAD_VALUE bit field
0	0	1	GPTi.TLDR[31:0] LOAD_VALUE bit field
0	1	0	GPTi.TLDR[31:0] LOAD_VALUE bit field
0	1	1	GPTi.TLDR[31:0] LOAD_VALUE - 1
1	0	0	N/A
1	0	1	N/A
1	1	0	GPTi.TLDR[31:0] LOAD_VALUE - 1
1	1	1	GPTi.TLDR[31:0] LOAD_VALUE - 1

The values of the GPTi.TPIR and GPTi.TNIR registers are calculated using the following formula:

- Positive increment value = ((INTEGER[Fclk * Ttick] + 1) * 1e6) - (Fclk * Ttick * 1e6)
- Negative increment value = (INTEGER[Fclk * Ttick] * 1e6) - (Fclk * Ttick * 1e6)

Note: Fclk clock frequency (kHz)

Ttick tick period (ms)

The timer overflow counter register (GPTi.TOCR) and the timer overflow wrapping register (GPTi.TOWR) are used to filter interrupts. When the timer overflows, it increments the 24-bit TOCR register. When the 24-bit TOCR register values match the value in the 24-bit TOWR register and the timer overflow is asserted, the TOCR register is reset and an interrupt is generated to the TISR register.

With the conversion block in reset state (the positive increment register, negative increment register, and counter value register are zeroed), the programming model and the behavior of GPTIMER1, GPTIMER2, and GPTIMER10 remain unchanged.

For 1-ms tick with a 32,768-Hz clock:

- GPTi.TPIR[31:0] POSITIVE_INC_VALUE = 232000
- GPTi.TNIR[31:0] NEGATIVE_INC_VALUE = -768000
- GPTi.TLDR[31:0] LOAD_VALUE = 0xFFFFFFE0

Note: Any value of the tick period can be generated with the appropriate value of the GPTi.TPIR, GPTi.TNIR, and GPTi.TLDR registers.

By default, the GPTi.TPIR, GPTi.TNIR, GPTi.TCVR, GPTi.TOCR, and GPTi.TOWR registers and the associated logic are in reset mode (all 0s) and have no action on the programming model.

16.2.4.3 Capture Mode Functionality

When a transition is detected on the module input pin (EVENT_CAPTURE), the timer value in the GPTi.TCRR register can be captured and saved in the GPTi.TCAR1 or GPTi.TCAR2 register function of the mode selected in the GPTi.TCLR[13] CAPT_MODE bit. The edge detection circuitry monitors transitions on the input pin (EVENT_CAPTURE).

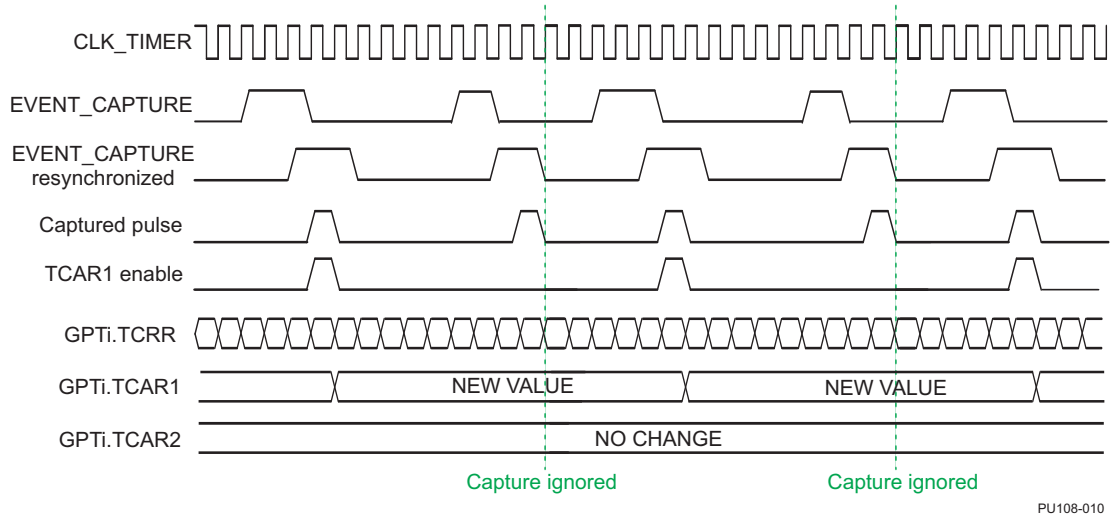
The rising edge, falling edge, or both, can be selected in the GPTi.TCLR[9:8] TCM field to trigger the timer counter capture. The module sets the GPTi.TISR[2] TCAR_IT_FLAG bit when an active edge is detected, and at the same time, the counter value GPTi.TCRR is stored in timer capture register GPTi.TCAR1 or GPTi.TCAR2, as follows:

- If the GPTi.TCLR[13] CAPT_MODE bit is 0, then on the first enabled capture event the value of the counter register is saved in the GPTi.TCAR1 register, and all the next events are ignored (no update on the GPTi.TCAR1 register and no interrupt triggering) until the detection logic is reset or the GPTi.TISR[2] TCAR_IT_FLAG bit is cleared by writing 1 in it.
- If the GPTi.TCLR[13] CAPT_MODE bit is 1, then on the first enabled capture event the value of the counter register is saved in the GPTi.TCAR1 register, and on the second enabled capture event, the value of the counter register is saved in the GPTi.TCAR2 register. If a capture interrupt is enabled, the interrupt triggers on the second event capture. All other events are ignored (no update on GPTi.TCAR1/GPTi.TCAR2 and no interrupt triggering) until the detection logic is reset or GPTi.TISR[2] TCAR_IT_FLAG bit is cleared by writing 1 in it. This mechanism is useful for period calculation of a clock, if that clock is connected to the EVENT_CAPTURE input pin.

The edge detection logic is reset (a new capture is enabled) when the active capture interrupt is servedthe GPTi.TISR[2] TCAR_IT_FLAG bit (previously 1) is cleared by writing 1 to it or when the edge detection mode bits (the GPTi.TCLR[9:8] TCM field) are changed from no-capture mode detection to any other mode. The timer functional clock (input to prescaler) is used to sample the input pin (EVENT_CAPTURE). An input negative or positive pulse can be detected when the pulse time is greater than the functional clock period. An interrupt is issued on edge detection if the capture interrupt enable bit is set in the GPTi.TIER[2] TCAR_IT_ENA bit. See the examples in Figure 16-10 and Figure 16-11.

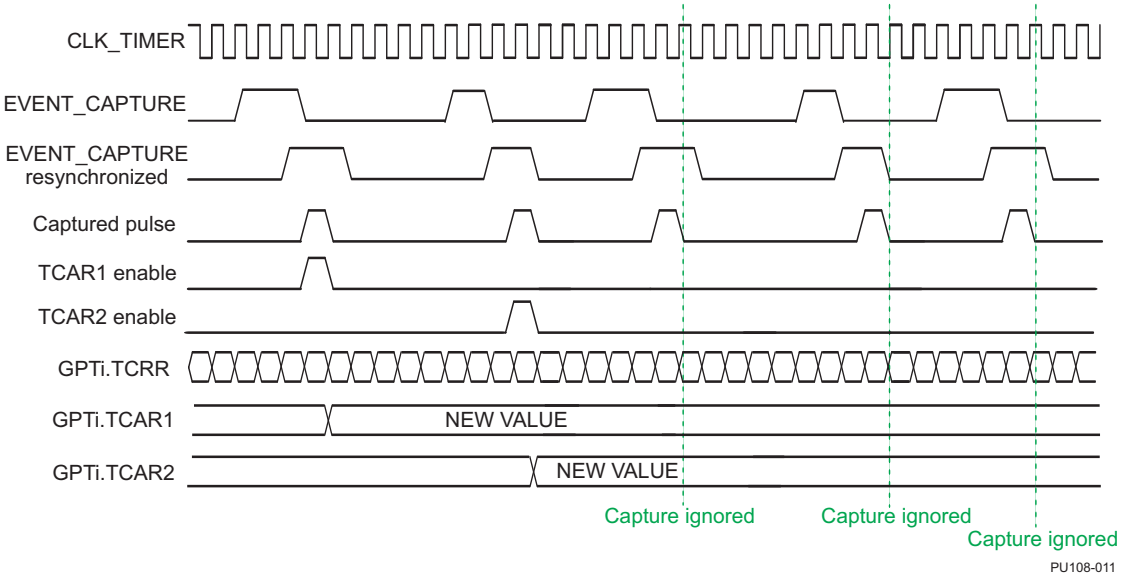
In Figure 16-10, the GPTi.TCLR[9:8] TCM value is 0b01, and GPTi.TCLR[13] CAPT_MODE is 0. Only the rising edge of EVENT_CAPTURE triggers a capture in the GPTi.TCAR1 and GPTi.TCAR2 registers, and only the GPTi.TCAR1 register updates.

Figure 16-10. Capture Wave Example for GPTi.TCLR[13] CAPT_MODE = 0



In Figure 16-11, the GPTi.TCLR[9:8] TCM value is 0b01, and GPTi.TCLR[13] CAPT_MODE is 1. Only the rising edge of EVENT_CAPTURE triggers a capture in the GPTi.TCAR1 register on the first enabled event, and the GPTi.TCAR2 register updates on the second enabled event.

Figure 16-11. Capture Wave Example for GPTi.TCLR[13] CAPT_MODE = 1



16.2.4.4 Compare Mode Functionality

When the compare enable register GPTi.TCLR[6] CE bit is set to 1, the timer value (GPTi.TCRR[31:0] TIMER_COUNTER field) is continuously compared to the value held in the timer match register (GPTi.TMAR). The GPTi.TMAR[31:0] COMPARE_VALUE value can be loaded at any time (timer counting or stopped). When the GPTi.TCRR and the GPTi.TMAR values match, an interrupt is issued, if the GPTi.TIER[0] MAT_IT_ENA bit is set.

The dedicated output pin (timer PWM) can be programmed in the GPTi.TCLR[12] PT bit through the GPTi.TCLR[11:10] TRG field to generate one positive pulse (timer clock duration) or to invert the current value (toggle mode) when an overflow or a match occurs.

16.2.4.5 Prescaler Functionality

A prescaler can be used to divide the timer counter input clock frequency. The prescaler is enabled when the GPTi.TCLR[5] PRE bit is set. The GPTi.TCLR[4:2] PTV field sets the second prescaler ratio. The prescaler counter is reset when the timer counter is stopped or reloaded on-the-fly.

Table 16-9 lists the prescaler/timer reload values versus contexts.

Table 16-9. Prescaler/Timer Reload Values Versus Contexts

Context	Prescaler	Timer Counter
Overflow (when autoreload is on)	Reset	GPTi.TLDR[31:0]
TCRR write	Reset	GPTi.TCRR[31:0]
TTGR write	Reset	GPTi.TLDR[31:0]
Stop	Reset	Frozen

16.2.4.6 Pulse-Width Modulation

The timer can be configured to provide a programmable PWM output. The timer PWM output pin can be configured to toggle on an event. The GPTi.TCLR[11:10] TRG field determines on which register value the PWM pin toggles. Either overflow alone or both overflow and match can be selected to toggle the timer PWM pin when a compare condition occurs.

CAUTION

In toggle mode when GPTi.TCLR[11:10] TRG = 0x2 (overflow and match), the first event that will toggle the PWM line is an overflow event. If a match event occurs first, it will not toggle the PWM line. Figure 16-13 illustrates those.

The GPTi.TCLR[7] SCPWM bit can be programmed to set or clear the timer PWM output signal only while the counter is stopped or the trigger is off. This allows setting the output pin to a known state before modulation starts. Modulation synchronously stops when the GPTi.TCLR[11:10] TRG field is cleared and overflow occurs. This allows fixing a deterministic state of the output pin when modulation stops.

In Figure 16-12, the internal overflow pulse is set each time (0xFFFF FFFF - GPTi.TLDR[31:0] LOAD_VALUE + 1) the value is reached, and the internal match pulse is set when the counter reaches the GPTi.TMAR register value. According to the value of the GPTi.TCLR[12] PT and GPTi.TCLR[11:10] TRG bits, the timer provides pulse or PWM event on the output pin (timer PWM).

The GPTi.TLDR and GPTi.TMAR registers must keep values smaller than the overflow value (0xFFFF FFFF) by at least two units. In case the PWM trigger events are both overflow and match, the difference between the values kept in the GPTi.TMAR register and the value in the GPTi.TLDR register must be at least two units. When match event is used, the compare mode GPTi.TCLR[6] CE bit must be set.

In Figure 16-12, the GPTi.TCLR[7] SCPWM bit is set to 0. In Figure 16-13, the GPTi.TCLR[7] SCPWM bit is set to 1. To obtain the desired wave form, start the counter at 0xFFFF FFFE value (to ensure an overflow first) or adjust the line polarity (GPTi.TCLR[7] SCPWM bit).

Figure 16-12. Timing Diagram of PWM With GPTi.TCLR[7] SCPWM Bit = 0

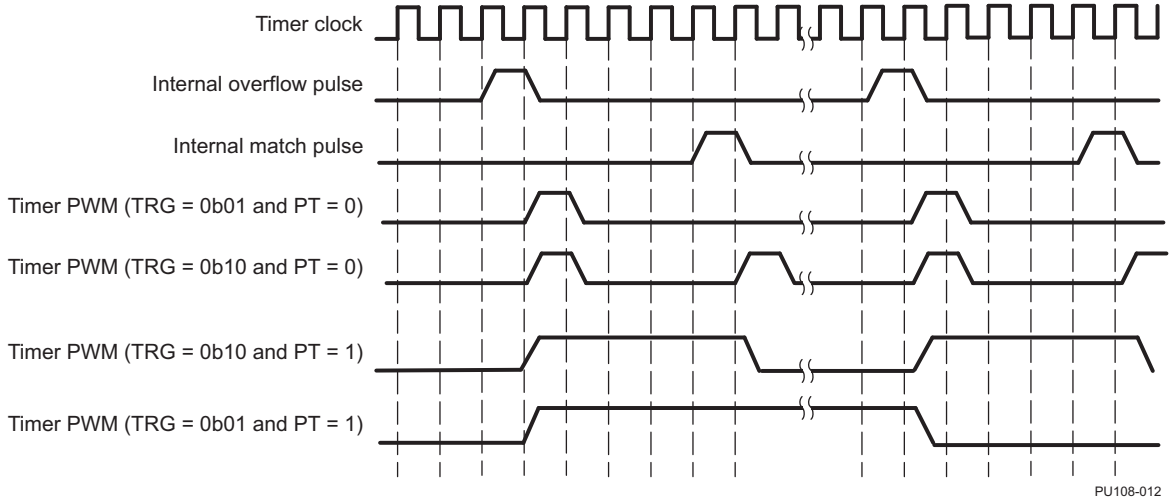
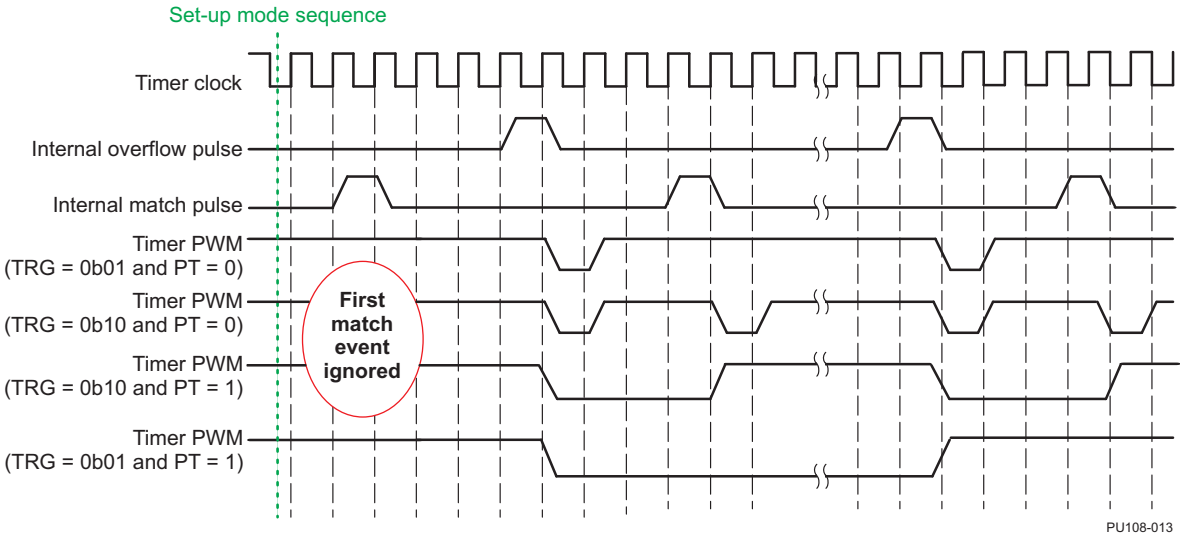


Figure 16-13. Timing Diagram of PWM With GPTi.TCLR[7] SCPWM Bit = 1



16.2.4.7 Timer Counting Rate

The timer rate is defined by the following values:

- Value of the prescaler fields (GPTi.TCLR[5] PRE bit and GPTi.TCLR[4:2] PTV field)
- Value loaded into the timer load register (GPTi.TLDR)

Table 16-10 lists prescaler clock ratio values.

Table 16-10. Prescaler Clock Ratio Values

GPTi.TCLR[5] PRE	GPTi.TCLR[4:2] PTV	Divisor (PS)
0	X	1
1	0	2
1	1	4
1	2	8
1	3	16
1	4	32

Table 16-10. Prescaler Clock Ratio Values (continued)

GPTi.TCLR[5] PRE	GPTi.TCLR[4:2] PTV	Divisor (PS)
1	5	64
1	6	128
1	7	256

Thus, the timer overflow-rate is expressed as:

OVF_Rate = (0xFFFF FFFF - GPTn.TLDR + 1) * (timer-functional clock period) * PS

With (timer-functional clock period) = 1 / (timer-functional clock frequency) and PS = $2^{(PTV + 1)}$ if prescaler is enabled, or PS = 1 if prescaler is disabled.

CAUTION

Internal resynchronization causes any write to the GPTn.TCLR[1] ST bit to have some latency before the register is updated:

2.5 * functional clock cycles write_GPTn.TCLR_latency 3.5 * functional clock cycles

Remember to take this latency into account whenever the timer must be started or stopped by a software change to the GPTn.TCLR[1] ST bit.

CAUTION

- In the non-PWM mode, GTPi.TLDR must be maintained at less than or equal to 0xFFFF FFFE.
- In the PWM mode, GTPi.TLDR must be maintained at less than or equal to 0xFFFF FFFD.

For example, with a timer clock input of 32 kHz and a GPTn.TCLR[5] PRE field equal to 0, the timer output period is as listed in Table 16-11.

Table 16-11. Value and Corresponding Interrupt Period

GPTi.TLDR[31:0] LOAD_VALUE	Interrupt Period
0x0000 0000	39 h
0xFFFF 0000	2.1 s
0xFFFF FFF0	524 µs
0xFFFF FFFE	65.5 µs

16.2.5 Timer Under Emulation

During emulation mode, the timer continues to run according to the value of the GPTi.TIOCP_CFG[5] EMUFREE bit.

If the GPTi.TIOCP_CFG[5] EMUFREE bit is set to 1, timer execution is not stopped in emulation mode and the interrupt is still generated when overflow or match is reached.

If the GPTi.TIOCP_CFG[5] EMUFREE bit is set to 0, the prescaler and timer are frozen and both resume on exit from emulation mode. The asynchronous external input pin (gpti_pwm_evt, with i=[9:12]) is internally synchronized on two timer-clock rising edges.

16.2.6 Accessing GP Timer Registers

All accesses are nonposted until software reconfiguration.

All registers are 32 bits wide, accessible through the L4 interface with 16-bit or 32-bit access (read/write).

Any 16-bit write access must be least-significant bit (LSB) first, and the second write access must be most-significant bit (MSB). Write operations to the GP timer registers (GPTi.TIDR, GPTi.TIOCP_CFG, GPTi.TISR, GPTi.TIER, GPTi.TWER, and GPTi.TSICR) can skip the MSB access if it is not necessary to update the 16 MSBs of the register.

Write operations to any functional register (GPTi.TCLR, GPTi.TCRR, GPTi.TLDR, GPTi.TTGR, and GPTi.TMAR, and GPTi.TPIR, GPTi.TNIR, GPTi.TCVR , GPTi.TOCR , and GPTi.TOWR for GPTIMER1, GPTIMER2, and GPTIMER10) must be complete (the MSB must be written even if the MSB data is not used).

16.2.6.1 Writing to Timer Registers

The host uses the L4 interface to write the following registers synchronously with the timer interface clock:

- GPTi.TLDR
- GPTi.TCRR
- GPTi.TIER
- GPTi.TISR
- GPTi.TCLR
- GPTi.TIOCP_CFG
- GPTi.TWER
- GPTi.TTGR
- GPTi.TSICR
- GPTi.TMAR

GPTIMER1, GPTIMER2, and GPTIMER10 also have the following registers:

- GPTi.TPIR
- GPTi.TNIR
- GPTi.TCVR
- GPTi.TOCR
- GPTi.TOWR

In 16-bit access mode, the 16 LSBs must be written before writing to the 16 MSBs.

16.2.6.1.1 Write Posting Synchronization Mode

This mode is used if the GPTi.TSICR[2] POSTED bit is set to 1.

This mode uses a posted write scheme to update any internal register (GPTi.TCLR, GPTi.TCRR, GPTi.TLDR, GPTi.TTGR, GPTi.TMAR, and GPTi.TPIR, GPTi.TNIR, GPTi.TCVR, GPTi.TOCR, and GPTi.TOWR for GPTIMER1, GPTIMER2, and GPTIMER10). Therefore, the write transaction is immediately acknowledged on the L4 interface, although the effective write operation occurs later, because of a resynchronization in the timer clock domain. The advantage is that neither the interconnect nor the device that requested the write transaction is stalled.

For each register, a status bit is provided in the timer write-posted status register GPTi.TWPS. In this mode, it is mandatory that the software checks this status bit prior to any write access. In case a write is attempted to a register with a previous access pending, the previous access is discarded without notice.

The timer module updates the timer counter register value synchronously with the L4 clock. Consequently, any read access to the timer counter register GPTi.TCRR does not add any resynchronization latency; the current value is always available.

If a write access is pending for a register, reading from this register does not yield a correct result. Software synchronization must be used to avoid incorrect results.

The drawback of this automatic update mechanism is that it assumes a given relationship between the timer interface frequency and the timer clock frequency.

Functional frequency range: freq(timer clock) < freq(L4 interface clock)/4

16.2.6.1.2 Write Nonposting Synchronization Mode

This mode is used if the GPTi.TSICR[2] POSTED bit is set to 0.

This mode uses a nonposted write scheme to update any internal register. Therefore, the write transaction is not acknowledged on the L4 interface until the effective write operation occurs after the resynchronization in the timer functional clock domain. The drawback is that both the interconnect and the device that requested the write transaction are stalled during this period.

The same full resynchronization scheme is used for a read transaction, and the same stall period applies. A register read following a write to the same register is always coherent.

This mode is functional regardless of the ratio between the L4 interface frequency and the timer clock frequency.

16.2.6.2 Reading From Timer Counter Registers

In 16-bit access mode, reading the 16 LSBs from the timer counter registers (GPTi.TCRR, GPTi.TCAR1, and GPTi.TCAR2) captures the current timer counter value. This must be followed by reading the 16MSBs.

IVA2.2 subsystem 16-bit accesses can be interleaved with MPU subsystem 32-bit accesses.

Note: LSB/MSB accesses cannot be interleaved (that is, the sequence LSB register 1, LSB register 2, MSB register 1, MSB register 2 is not supported).

16.3 General-Purpose Timers Register Manual

16.3.1 GP Timer Register Map

16.3.1.1 Instance Summary

Table 16-12 lists the base address and block size for the GP timer module instances. All timers are memory mapped to the L4 peripheral bus memory space.

Table 16-12. GP Timer Instance Summary

Module Name	Base Address	Size
GPTIMER1	0x4831 8000	4K bytes
GPTIMER2	0x4903 2000	4K bytes
GPTIMER3	0x4903 4000	4K bytes
GPTIMER4	0x4903 6000	4K bytes
GPTIMER5	0x4903 8000	4K bytes
GPTIMER6	0x4903 A000	4K bytes
GPTIMER7	0x4903 C000	4K bytes
GPTIMER8	0x4903 E000	4K bytes
GPTIMER9	0x4904 0000	4K bytes
GPTIMER10	0x4808 6000	4K bytes
GPTIMER11	0x4808 8000	4K bytes

16.3.2 GP Timer Register Mapping Summary

CAUTION

The GP timer registers are limited to 32-bit and 16-bit data accesses; 8-bit access is not allowed and can corrupt the register content.

Table 16-13 through Table 16-15 provide the register summary and associated offset addresses for the 11 GP timer internal registers. (Example: The physical address for the TCLR register of GPTIMER8 is 0x4903 E024.)

Table 16-13. GPTIMER1 to GPTIMER4 Register Summary

Register Name	Type	Register Width (Bits)	Address Offset	Physical Address (GPTIMER1)	Physical Address (GPTIMER2)	Physical Address (GPTIMER3)	Physical Address (GPTIMER4)
TIDR	R	32	0x000	0x4831 8000	0x4903 2000	0x4903 4000	0x4903 6000
TIOCP_CFG	RW	32	0x010	0x4831 8010	0x4903 2010	0x4903 4010	0x4903 6010
TISTAT	R	32	0x014	0x4831 8014	0x4903 2014	0x4903 4014	0x4903 6014
TISR	RW	32	0x018	0x4831 8018	0x4903 2018	0x4903 4018	0x4903 6018
TIER	RW	32	0x01C	0x4831 801C	0x4903 201C	0x4903 401C	0x4903 601C
TWER	RW	32	0x020	0x4831 8020	0x4903 2020	0x4903 4020	0x4903 6020
TCLR	RW	32	0x024	0x4831 8024	0x4903 2024	0x4903 4024	0x4903 6024
TCRR	RW	32	0x028	0x4831 8028	0x4903 2028	0x4903 4028	0x4903 6028
TLDR	RW	32	0x02C	0x4831 802C	0x4903 202C	0x4903 402C	0x4903 602C
TTGR	RW	32	0x030	0x4831 8030	0x4903 2030	0x4903 4030	0x4903 6030
TWPS	R	32	0x034	0x4831 8034	0x4903 2034	0x4903 4034	0x4903 6034
TMAR	RW	32	0x038	0x4831 8038	0x4903 2038	0x4903 4038	0x4903 6038
TCAR1	R	32	0x03C	0x4831 803C	0x4903 203C	0x4903 403C	0x4903 603C
TSICR	RW	32	0x040	0x4831 8040	0x4903 2040	0x4903 4040	0x4903 6040
TCAR2	R	32	0x044	0x4831 8044	0x4903 2044	0x4903 4044	0x4903 6044
TPIR	RW	32	0x048	0x4831 8048	0x4903 2048	-	-
TNIR	RW	32	0x04C	0x4831 804C	0x4903 204C	-	-
TCVR	RW	32	0x050	0x4831 8050	0x4903 2050	-	-
TOCR	RW	32	0x054	0x4831 8054	0x4903 2054	-	-
TOWR	RW	32	0x058	0x4831 8058	0x4903 2058	-	-

Table 16-14. GPTIMER5 to GPTIMER8 Register Summary

Register Name	Type	Register Width (Bits)	Address Offset	Physical Address (GPTIMER5)	Physical Address (GPTIMER6)	Physical Address (GPTIMER7)	Physical Address (GPTIMER8)
TIDR	R	32	0x000	0x4903 8000	0x4903 A000	0x4903 C000	0x4903 E000
TIOCP_CFG	RW	32	0x010	0x4903 8010	0x4903 A010	0x4903 C010	0x4903 E010
TISTAT	R	32	0x014	0x4903 8014	0x4903 A014	0x4903 C014	0x4903 E014
TISR	RW	32	0x018	0x4903 8018	0x4903 A018	0x4903 C018	0x4903 E018
TIER	RW	32	0x01C	0x4903 801C	0x4903 A01C	0x4903 C01C	0x4903 E01C
TWER	RW	32	0x020	0x4903 8020	0x4903 A020	0x4903 C020	0x4903 E020
TCLR	RW	32	0x024	0x4903 8024	0x4903 A024	0x4903 C024	0x4903 E024
TCRR	RW	32	0x028	0x4903 8028	0x4903 A028	0x4903 C028	0x4903 E028
TLDR	RW	32	0x02C	0x4903 802C	0x4903 A02C	0x4903 C02C	0x4903 E02C
TTGR	RW	32	0x030	0x4903 8030	0x4903 A030	0x4903 C030	0x4903 E030
TWPS	R	32	0x034	0x4903 8034	0x4903 A034	0x4903 C034	0x4903 E034
TMAR	RW	32	0x038	0x4903 8038	0x4903 A038	0x4903 C038	0x4903 E038
TCAR1	R	32	0x03C	0x4903 803C	0x4903 A03C	0x4903 C03C	0x4903 E03C
TSICR	RW	32	0x040	0x4903 8040	0x4903 A040	0x4903 C040	0x4903 E040
TCAR2	R	32	0x044	0x4903 8044	0x4903 A044	0x4903 C044	0x4903 E044

Table 16-15. GPTIMER9 to GPTIMER11 Register Summary

Register Name	Type	Register Width (Bits)	Address Offset	Physical Address (GPTIMER9)	Physical Address (GPTIMER10)	Physical Address (GPTIMER11)
TIDR	R	32	0x000	0x4904 0000	0x4808 6000	0x4808 8000
TIOCP_CFG	RW	32	0x010	0x4904 0010	0x4808 6010	0x4808 8010
TISTAT	R	32	0x014	0x4904 0014	0x4808 6014	0x4808 8014
TISR	RW	32	0x018	0x4904 0018	0x4808 6018	0x4808 8018
TIER	RW	32	0x01C	0x4904 001C	0x4808 601C	0x4808 801C
TWER	RW	32	0x020	0x4904 0020	0x4808 6020	0x4808 8020
TCLR	RW	32	0x024	0x4904 0024	0x4808 6024	0x4808 8024
TCRR	RW	32	0x028	0x4904 0028	0x4808 6028	0x4808 8028
TLDR	RW	32	0x02C	0x4904 002C	0x4808 602C	0x4808 802C
TTGR	RW	32	0x030	0x4904 0030	0x4808 6030	0x4808 8030
TWPS	R	32	0x034	0x4904 0034	0x4808 6034	0x4808 8034
TMAR	RW	32	0x038	0x4904 0038	0x4808 6038	0x4808 8038
TCAR1	R	32	0x03C	0x4904 003C	0x4808 603C	0x4808 803C
TSICR	RW	32	0x040	0x4904 0040	0x4808 6040	0x4808 8040
TCAR2	R	32	0x044	0x4904 0044	0x4808 6044	0x4808 8044
TPIR	RW	32	0x048	-	0x4808 6048	-
TNIR	RW	32	0x04C	-	0x4808 604C	-
TCVR	RW	32	0x050	-	0x4808 6050	-
TOCR	RW	32	0x054	-	0x4808 6054	-
TOWR	RW	32	0x058	-	0x4808 6058	-

16.3.3 GP Timer Register Descriptions

Table 16-16 through Table 16-54 describe the GP timer register bits.

Table 16-16. TIDR

Address Offset	0x000		
Physical Address	0x4831 8000	**Instance**	GPT1
	0x4903 2000		GPT2
	0x4903 4000		GPT3
	0x4903 6000		GPT4
	0x4903 8000		GPT5
	0x4903 A000		GPT6
	0x4903 C000		GPT7
	0x4903 E000		GPT8
	0x4904 0000		GPT9
	0x4808 6000		GPT10
	0x4808 8000		GPT11
Description	This register contains the IP revision code.		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																								TID_REV							

Bits	Field Name	Description	Type	Reset
31:8	Reserved	Reads return 0.	R	0x000000
7:0	TID_REV	IP revision [7:4] Major revision [3:0] Minor revision Examples: 0x10 for 1.0, 0x21 for 2.1	R	See (1)

(1) TI internal data

Table 16-17. Register Call Summary for Register TIDR

General-Purpose Timers • Accessing GP Timer Registers: [0]
General-Purpose Timers Register Manual • GP Timer Register Mapping Summary: [1] [2] [3]

Table 16-18. TIOCP_CFG

Address Offset	0x010		
Physical Address	0x4831 8010	**Instance**	GPT1
	0x4903 2010		GPT2
	0x4903 4010		GPT3
	0x4903 6010		GPT4
	0x4903 8010		GPT5
	0x4903 A010		GPT6
	0x4903 C010		GPT7
	0x4903 E010		GPT8
	0x4904 0010		GPT9
	0x4808 6010		GPT10
	0x4808 8010		GPT11
Description	This register controls the various parameters of the GP timer L4 interface.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																						CLOCKACTIVITY		Reserved		EMUFREE		IDLEMODE		ENAWAKEUP	SOFTRESET	AUTOIDLE

Bits	Field Name	DESCRIPTION	Type	Reset
31:10	Reserved	Write 0s for future compatibility. Reads return 0.	R	0x0000000
9:8	CLOCKACTIVITY	Clock activity during wakeup mode period: 0x0: L4 interface and Functional clocks can be switched off. 0x1: L4 interface clock is maintained during wake-up period; Functional clock can be switched off. 0x2: L4 interface clock can be switched off; Functional clock is maintained during wake-up period. 0x3: L4 interface and Functional clocks are maintained during wake-up period.	RW	0x0
7:6	Reserved	Write 0s for future compatibility. Reads return 0.	R	0x0

Bits	Field Name	DESCRIPTION	Type	Reset
5	EMUFREE	Emulation mode	RW	0
		0x0: Timer counter frozen in emulation		
		0x1: Timer counter free-running in emulation		
4:3	IDLEMODE	Power management, req/ack control	RW	0x0
		0x0: Force-idle. An idle request is acknowledged unconditionally.		
		0x1: No-idle. An idle request is never acknowledged.		
		0x2: Smart-idle. Acknowledgement to an idle request is given based on the internal activity of the module.		
		0x3: Reserved. Do not use.		
2	ENAWAKEUP	Wake-up feature global control	RW	0
		0x0: No wake-up line assertion in idle mode		
		0x1: Wake-up line assertion enabled in smart-idle mode		
1	SOFTRESET	Software reset. This bit is automatically reset by the hardware. During reads, it always returns 0.	RW	0
		0x0: Normal mode		
		0x1: The module is reset.		
0	AUTOIDLE	Internal L4 interface clock gating strategy		0
		0x0: L4 interface clock is free-running.		
		0x1: Automatic L4 interface clock gating strategy is applied, based on the L4 interface activity.		

Table 16-19. Register Call Summary for Register TIOCP_CFG

General-Purpose Timers

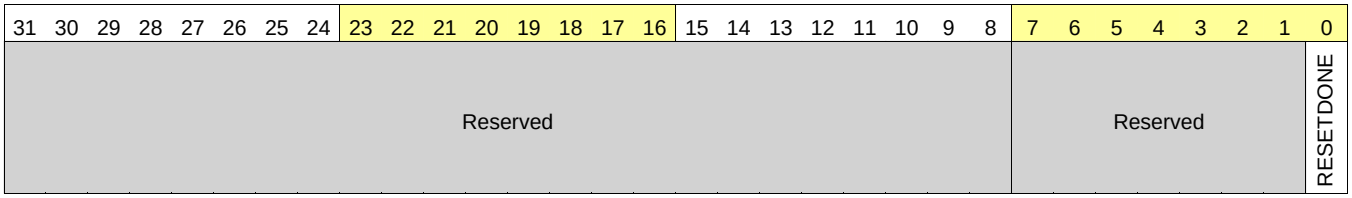
- Clock Management: [0] [1] [2]
- Wake-Up Capability: [3] [4]
- Software Reset: [5] [6]
- Timer Under Emulation: [7] [8] [9]
- Accessing GP Timer Registers: [10]
- Writing to Timer Registers: [11]

General-Purpose Timers Register Manual

- GP Timer Register Mapping Summary: [12] [13] [14]

Table 16-20. TISTAT

Address Offset	0x014		
Physical Address	0x4831 8014	**Instance**	GPT1
	0x4903 2014		GPT2
	0x4903 4014		GPT3
	0x4903 6014		GPT4
	0x4903 8014		GPT5
	0x4903 A014		GPT6
	0x4903 C014		GPT7
	0x4903 E014		GPT8
	0x4904 0014		GPT9
	0x4808 6014		GPT10
	0x4808 8014		GPT11
Description	This register provides status information about the module, excluding the interrupt status information.		
Type	R		



Bits	Field Name	DESCRIPTION	Type	Reset
31:8	Reserved	Reads return 0.	R	0x000000
7:1	Reserved	Reads return 0.	R	0x00
0	RESETDONE	Internal reset monitoring	R	0
		0x0: Internal module reset is ongoing.		
		0x1: Reset completed		

Table 16-21. Register Call Summary for Register TISTAT

General-Purpose Timers
- Software Reset: [0]

General-Purpose Timers Register Manual
- GP Timer Register Mapping Summary: [1] [2] [3]

Table 16-22. TISR

Address Offset	0x018		
Physical Address	0x4831 8018	**Instance**	GPT1
	0x4903 2018		GPT2
	0x4903 4018		GPT3
	0x4903 6018		GPT4
	0x4903 8018		GPT5
	0x4903 A018		GPT6
	0x4903 C018		GPT7
	0x4903 E018		GPT8
	0x4904 0018		GPT9
	0x4808 6018		GPT10
	0x4808 8018		GPT11
Description	This register shows which interrupt events are pending inside the module.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																													TCAR_IT_FLAG	OVF_IT_FLAG	MAT_IT_FLAG

Bits	Field Name	Description		Type	Reset
31:3	Reserved	Reads return 0.		R	0x00000000
2	TCAR_IT_FLAG	Pending capture interrupt status		RW	0
		Read 0x0:	No capture interrupt event pending		
		Write 0x0:	Status unchanged		
		Read 0x1:	Capture interrupt event pending		
		Write 0x1:	Status bit cleared		
1	OVF_IT_FLAG	Pending overflow interrupt status		RW	0
		Read 0x0:	No overflow interrupt pending		
		Write 0x0:	Status unchanged		
		Read 0x1:	Overflow interrupt pending		
		Write 0x1:	Status bit cleared		
0	MAT_IT_FLAG	Pending match interrupt status		RW	0
		Read 0x0:	No match interrupt pending		
		Write 0x0:	Status unchanged		
		Read 0x1:	Match interrupt pending		
		Write 0x1:	Status bit cleared		

Table 16-23. Register Call Summary for Register TISR

General-Purpose Timers

- Wake-Up Capability: [0]
- GP Timer Interrupts: [1]
- 1-ms Tick Generation (Only GPTIMER1, GPTIMER2, and GPTIMER10): [2]
- Capture Mode Functionality: [3] [4] [5] [6]
- Accessing GP Timer Registers: [7]
- Writing to Timer Registers: [8]

General-Purpose Timers Register Manual

- GP Timer Register Mapping Summary: [9] [10] [11]

Table 16-24. TIER

Address Offset	0x01C		
Physical Address	0x4831 801C	**Instance**	GPT1
	0x4903 201C		GPT2
	0x4903 401C		GPT3
	0x4903 601C		GPT4
	0x4903 801C		GPT5
	0x4903 A01C		GPT6
	0x4903 C01C		GPT7
	0x4903 E01C		GPT8
	0x4904 001C		GPT9
	0x4808 601C		GPT10
	0x4808 801C		GPT11
Description	This register controls (enable/disable) the interrupt events.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																													TCAR_IT_ENA	OVF_IT_ENA	MAT_IT_ENA

Bits	Field Name	Description	Type	Reset
31:3	Reserved	Reads return 0.	R	0x00000000
2	TCAR_IT_ENA	Enable capture interrupt	RW	0
		0x0: Disable capture interrupt.		
		0x1: Enable capture interrupt.		
1	OVF_IT_ENA	Enable overflow interrupt	RW	0
		0x0: Disable overflow interrupt.		
		0x1: Enable overflow interrupt.		
0	MAT_IT_ENA	Enable match interrupt	RW	0
		0x0: Disable match interrupt.		
		0x1: Enable match interrupt.		

Table 16-25. Register Call Summary for Register TIER

General-Purpose Timers

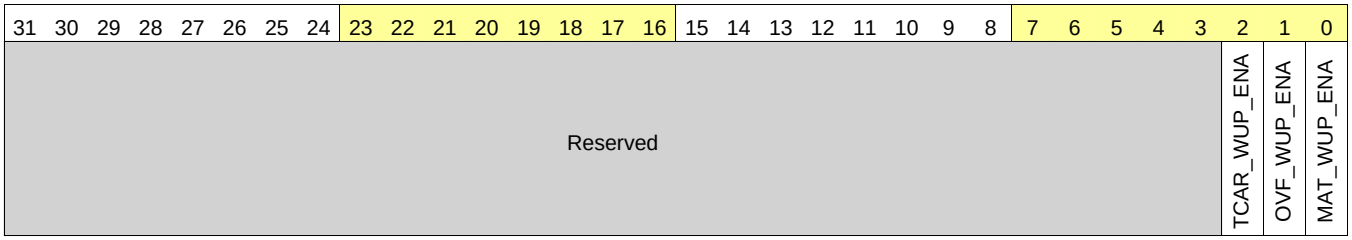
- GP Timer Interrupts: [0]
- GP Timers Functional Description: [1]
- Timer Mode Functionality: [2]
- Capture Mode Functionality: [3]
- Compare Mode Functionality: [4]
- Accessing GP Timer Registers: [5]
- Writing to Timer Registers: [6]

General-Purpose Timers Register Manual

- GP Timer Register Mapping Summary: [7] [8] [9]

Table 16-26. TWER

Address Offset	0x020		
Physical Address	0x4831 8020	**Instance**	GPT1
	0x4903 2020		GPT2
	0x4903 4020		GPT3
	0x4903 6020		GPT4
	0x4903 8020		GPT5
	0x4903 A020		GPT6
	0x4903 C020		GPT7
	0x4903 E020		GPT8
	0x4904 0020		GPT9
	0x4808 6020		GPT10
	0x4808 8020		GPT11
Description	This register controls (enable/disable) the wake-up feature on specific interrupt events.		
Type	RW		



Bits	Field Name	Description	Type	Reset
31:3	Reserved	Reads return 0	R	0x00000000
2	TCAR_WUP_ENA	Enable capture wake-up	RW	0
		0x0: Disable capture wake-up.		
		0x1: Enable capture wake-up.		
1	OVF_WUP_ENA	Enable overflow wake-up	RW	0
		0x0: Disable overflow wake-up.		
		0x1: Enable overflow wake-up.		
0	MAT_WUP_ENA	Enable match wake-up	RW	0
		0x0: Disable match wake-up.		
		0x1: Enable match wake-up.		

Table 16-27. Register Call Summary for Register TWER

General-Purpose Timers

- Wake-Up Capability: [0] [1]
- GP Timers Functional Description: [2]
- Accessing GP Timer Registers: [3]
- Writing to Timer Registers: [4]

General-Purpose Timers Register Manual

- GP Timer Register Mapping Summary: [5] [6] [7]

Table 16-28. TCLR

Address Offset	0x024		
Physical Address	0x4831 8024	**Instance**	GPT1
	0x4903 2024		GPT2
	0x4903 4024		GPT3
	0x4903 6024		GPT4
	0x4903 8024		GPT5
	0x4903 A024		GPT6
	0x4903 C024		GPT7
	0x4903 E024		GPT8
	0x4904 0024		GPT9
	0x4808 6024		GPT10
	0x4808 8024		GPT11
Description	This register controls optional features specific to the timer functionality.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																	GPO_CFG	CAPT_MODE	PT	TRG		TCM		SCPWM	CE	PRE	PTV			AR	ST

Bits	Field Name	Description	Type	Reset
31:15	Reserved	Reads return 0.	R	0x00000
14	GPO_CFG	PWM output/event detection input pin direction control: 0x0: Configures the pin as an output (needed when PWM mode is required) 0x1: Configures the pin as an input (needed when capture mode is required)	RW	0
13	CAPT_MODE	Capture mode select bit (first/second) 0x0: Capture the first enabled capture event in TCAR1. 0x1: Capture the second enabled capture event in TCAR2.	RW	0
12	PT	Pulse or toggle select bit 0x0: Pulse modulation 0x1: Toggle modulation	RW	0
11:10	TRG	Trigger output mode 0x0: No trigger 0x1: Overflow trigger 0x2: Overflow and match trigger 0x3: Reserved	RW	0x0
9:8	TCM	Transition capture mode 0x0: No capture 0x1: Capture on rising edges of EVENT_CAPTURE pin. 0x2: Capture on falling edges of EVENT_CAPTURE pin. 0x3: Capture on both edges of EVENT_CAPTURE pin.	RW	0x0
7	SCPWM	Pulse-width-modulation output pin default setting when counter is stopped or trigger output mode is set to no trigger. 0x0: Default value of PWM_out output: 0 0x1: Default value of PWM_out output: 1	RW	0
6	CE	Compare enable 0x0: Compare disabled	RW	0

Bits	Field Name	Description	Type	Reset
		0x1: Compare enabled		
5	PRE	Prescaler enable	RW	0
		0x0: Prescaler disabled		
		0x1: Prescaler enabled		
4:2	PTV	Trigger output mode		
		0x0: The timer counter is prescaled with the value: $2^{(PTV+1)}$. Example: PTV = 3, counter increases value (if started) after 16 functional clock periods.	RW	0x0
1	AR	Autoreload mode	RW	0
		0x0: One-shot mode overflow		
		0x1: Autoreload mode overflow		
0	ST	Start/stop timer control	RW	0
		0x0: Stop the timer		
		0x1: Start the timer		

Table 16-29. Register Call Summary for Register TCLR

General-Purpose Timers

- Timer Mode Functionality: [0] [1] [2] [3] [4] [5] [6]
- Capture Mode Functionality: [7] [8] [9] [10] [11] [12] [13] [14] [15]
- Compare Mode Functionality: [16] [17] [18]
- Prescaler Functionality: [19] [20]
- Pulse-Width Modulation: [21] [22] [23] [24] [25] [26] [27] [28] [29] [30]
- Timer Counting Rate: [31] [32] [33] [34] [35] [36] [37]
- Accessing GP Timer Registers: [38]
- Writing to Timer Registers: [39]
- Write Posting Synchronization Mode: [40]

General-Purpose Timers Register Manual

- GP Timer Register Mapping Summary: [41] [42] [43] [44]

Table 16-30. TCRR

Address Offset	0x028		
Physical Address	0x4831 8028	**Instance**	GPT1
	0x4903 2028		GPT2
	0x4903 4028		GPT3
	0x4903 6028		GPT4
	0x4903 8028		GPT5
	0x4903 A028		GPT6
	0x4903 C028		GPT7
	0x4903 E028		GPT8
	0x4904 0028		GPT9
	0x4808 6028		GPT10
	0x4808 8028		GPT11
Description	This register holds the value of the internal counter.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TIMER_COUNTER																															

Bits	Field Name	Description	Type	Reset
31:0	TIMER_COUNTER	The value of the timer counter register	RW	0x00000000

Table 16-31. Register Call Summary for Register TCRR

General-Purpose Timers

- Wake-Up Capability: [0] [1]
- Timer Mode Functionality: [2] [3] [4] [5] [6] [7] [8] [9] [10]
- 1-ms Tick Generation (Only GPTIMER1, GPTIMER2, and GPTIMER10): [11] [12] [13]
- Capture Mode Functionality: [14] [15]
- Compare Mode Functionality: [16] [17]
- Prescaler Functionality: [18] [19]
- Accessing GP Timer Registers: [20]
- Writing to Timer Registers: [21]
- Write Posting Synchronization Mode: [22] [23]
- Reading From Timer Counter Registers: [24]

General-Purpose Timers Register Manual

- GP Timer Register Mapping Summary: [25] [26] [27]
- GP Timer Register Descriptions: [28] [29] [30]

Table 16-32. TLDR

Address Offset	0x02C		
Physical Address	0x4831 802C	**Instance**	GPT1
	0x4903 202C		GPT2
	0x4903 402C		GPT3
	0x4903 602C		GPT4
	0x4903 802C		GPT5
	0x4903 A02C		GPT6
	0x4903 C02C		GPT7
	0x4903 E02C		GPT8
	0x4904 002C		GPT9
	0x4808 602C		GPT10
	0x4808 802C		GPT11
Description	This register holds the timer load values.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
LOAD_VALUE																															

Bits	Field Name	Description	Type	Reset
31:0	LOAD_VALUE	The value of the timer load register	RW	0x00000000

Table 16-33. Register Call Summary for Register TLDR

General-Purpose Timers

- Timer Mode Functionality: [0] [1] [2]
- 1-ms Tick Generation (Only GPTIMER1, GPTIMER2, and GPTIMER10): [3] [4] [5] [6] [7] [8] [9] [10] [11] [12]
- Prescaler Functionality: [13] [14]
- Pulse-Width Modulation: [15] [16] [17]
- Timer Counting Rate: [18] [19] [20] [21] [22]
- Accessing GP Timer Registers: [23]
- Writing to Timer Registers: [24]
- Write Posting Synchronization Mode: [25]

Table 16-33. Register Call Summary for Register TLDR (continued)

General-Purpose Timers Register Manual

- GP Timer Register Mapping Summary: [26] [27] [28]

Table 16-34. TTGR

Address Offset	0x030		
Physical Address	0x4831 8030	**Instance**	GPT1
	0x4903 2030		GPT2
	0x4903 4030		GPT3
	0x4903 6030		GPT4
	0x4903 8030		GPT5
	0x4903 A030		GPT6
	0x4903 C030		GPT7
	0x4903 E030		GPT8
	0x4904 0030		GPT9
	0x4808 6030		GPT10
	0x4808 8030		GPT11
Description	This register triggers a counter reload of timer by writing any value in it.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TTGR_ VALUE																															

Bits	Field Name	Description	Type	Reset
31:0	TTGR_ VALUE	The value of the trigger register. During reads, it always returns 0xFFFFFFFF.	RW	0xFFFFFFFF

Table 16-35. Register Call Summary for Register TTGR

General-Purpose Timers

- Timer Mode Functionality: [0] [1]
- Prescaler Functionality: [2]
- Accessing GP Timer Registers: [3]
- Writing to Timer Registers: [4]
- Write Posting Synchronization Mode: [5]

General-Purpose Timers Register Manual

- GP Timer Register Mapping Summary: [6] [7] [8]

Table 16-36. TWPS

Address Offset	0x034		
Physical Address	0x4831 8034	**Instance**	GPT1
	0x4903 2034		GPT2
	0x4903 4034		GPT3
	0x4903 6034		GPT4
	0x4903 8034		GPT5
	0x4903 A034		GPT6
	0x4903 C034		GPT7
	0x4903 E034		GPT8
	0x4904 0034		GPT9
	0x4808 6034		GPT10
	0x4808 8034		GPT11
Description	This register indicates if a Write-Posted is pending.		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																						W_PEND_TOWR	W_PEND_TOCR	W_PEND_TCVR	W_PEND_TNIR	W_PEND_TPIR	W_PEND_TMAR	W_PEND_TTGR	W_PEND_TLDR	W_PEND_TCRR	W_PEND_TCLR

Bits	Field Name	Description	Type	Reset
31:10	Reserved	Reads return 0.	R	0x0000000
9	W_PEND_TOWR	Write pending for register GPT_TOWR	R	0
		0x0: Overflow wrapping register write not pending		
		0x1: Overflow wrapping register write pending		
	Reserved for instances 3, 4, 5, 6, 7, 8, 9, 11, 12	Read returns reset value.	R	0
8	W_PEND_TOCR	Write pending for register GPT_TOCR	R	0
		0x0: Overflow counter register write not pending		
		0x1: Overflow counter register write pending		
	Reserved for instances 3, 4, 5, 6, 7, 8, 9, 11, 12	Read returns reset value.	R	0
7	W_PEND_TCVR	Write pending for register GPT_TCVR	R	0
		0x0: Counter value register write not pending		
		0x1: Counter value register write pending		
	Reserved for instances 3, 4, 5, 6, 7, 8, 9, 11, 12	Read returns reset value.	R	0
6	W_PEND_TNIR	Write pending for register GPT_TNIR	R	0
		0x0: Negative increment register write not pending		
		0x1: Negative increment register write pending		
	Reserved for instances 3, 4, 5, 6, 7, 8, 9, 11, 12	Read returns reset value.	R	0
5	W_PEND_TPIR	Write pending for register GPT_TPIR	R	0
		0x0: Positive increment register write not pending		
		0x1: Positive increment register write pending		
	Reserved for instances 3, 4, 5, 6, 7, 8, 9, 11, 12	Read returns reset value.	R	0
4	W_PEND_TMAR	Write pending for register GPT_TMAR	R	0
		0x0: Match register write not pending		

Bits	Field Name	Description	Type	Reset
		0x1: Match register write pending		
3	W_PEND_TTGR	Write pending for register GPT_TTGR	R	0
		0x0: Trigger register write not pending		
		0x1: Trigger register write pending		
2	W_PEND_TLDR	Write pending for register GPT_TLDR	R	0
		0x0: Load register write not pending		
		0x1: Load register write pending		
1	W_PEND_TCRR	Write pending for register GPT_TCRR	R	0
		0x0: Counter register write not pending		
		0x1: Counter register write pending		
0	W_PEND_TCLR	Write pending for register GPT_TCLR	R	0
		0x0: Control register write not pending		
		0x1: Control register write pending		

Table 16-37. Register Call Summary for Register TWPS

General-Purpose Timers • Write Posting Synchronization Mode: [0]
General-Purpose Timers Register Manual • GP Timer Register Mapping Summary: [1] [2] [3]

Table 16-38. TMAR

Address Offset	0x038		
Physical Address	0x4831 8038	**Instance**	GPT1
	0x4903 2038		GPT2
	0x4903 4038		GPT3
	0x4903 6038		GPT4
	0x4903 8038		GPT5
	0x4903 A038		GPT6
	0x4903 C038		GPT7
	0x4903 E038		GPT8
	0x4904 0038		GPT9
	0x4808 6038		GPT10
	0x4808 8038		GPT11
Description	This register holds the value to be compared with the counter value.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
COMPARE_ VALUE																															

Bits	Field Name	Description	Type	Reset
31:0	COMPARE_ VALUE	The value of the match register	RW	0x00000000

Table 16-39. Register Call Summary for Register TMAR

General-Purpose Timers • Wake-Up Capability: [0] • Compare Mode Functionality: [1] [2] [3] • Pulse-Width Modulation: [4] [5] [6] • Accessing GP Timer Registers: [7] • Writing to Timer Registers: [8] • Write Posting Synchronization Mode: [9]
General-Purpose Timers Register Manual • GP Timer Register Mapping Summary: [10] [11] [12]

Table 16-40. TCAR1

Address Offset	0x03C		
Physical Address	0x4831 803C	**Instance**	GPT1
	0x4903 203C		GPT2
	0x4903 403C		GPT3
	0x4903 603C		GPT4
	0x4903 803C		GPT5
	0x4903 A03C		GPT6
	0x4903 C03C		GPT7
	0x4903 E03C		GPT8
	0x4904 003C		GPT9
	0x4808 603C		GPT10
	0x4808 803C		GPT11
Description	This register holds the first captured value of the counter register.		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CAPTURE_ VALUE1																															

Bits	Field Name	Description	Type	Reset
31:0	CAPTURE_ VALUE1	The value of first captured counter register	R	0x00000000

Table 16-41. Register Call Summary for Register TCAR1

General-Purpose Timers • Capture Mode Functionality: [0] [1] [2] [3] [4] [5] [6] [7] [8] • Reading From Timer Counter Registers: [9]
General-Purpose Timers Register Manual • GP Timer Register Mapping Summary: [10] [11] [12] • GP Timer Register Descriptions: [13]

Table 16-42. TSICR

Address Offset	0x040		
Physical Address	0x4831 8040	**Instance**	GPT1
	0x4903 2040		GPT2
	0x4903 4040		GPT3
	0x4903 6040		GPT4
	0x4903 8040		GPT5
	0x4903 A040		GPT6
	0x4903 C040		GPT7
	0x4903 E040		GPT8
	0x4904 0040		GPT9
	0x4808 6040		GPT10
	0x4808 8040		GPT11
Description	This register contains the bits that control the interface between the L4 interface and functional clock domains-posted mode and functional SW reset.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																													POSTED	SFT	Reserved

Bits	Field Name	Description	Type	Reset
31:3	Reserved	Reads return 0.	R	0x00000000
2	POSTED	Posted mode selection	RW	1
		0x0: Non-posted mode selected		
		0x1: Posted mode selected		
1	SFT	Reset software functional registers. This bit is automatically reset by the hardware. During reads, it always returns 0.	RW	0
		0x0: Normal functional mode		
		0x1: The functional registers are reset.		
0	Reserved	Reads return 0.	R	0

Table 16-43. Register Call Summary for Register TSICR

General-Purpose Timers

- Software Reset: [0] [1]
- Accessing GP Timer Registers: [2]
- Writing to Timer Registers: [3]
- Write Posting Synchronization Mode: [4]
- Write Nonposting Synchronization Mode: [5]

General-Purpose Timers Register Manual

- GP Timer Register Mapping Summary: [6] [7] [8]

Table 16-44. TCAR2

Address Offset	0x044		
Physical Address	0x4831 8044	**Instance**	GPT1
	0x4903 2044		GPT2
	0x4903 4044		GPT3
	0x4903 6044		GPT4
	0x4903 8044		GPT5
	0x4903 A044		GPT6
	0x4903 C044		GPT7
	0x4903 E044		GPT8
	0x4904 0044		GPT9
	0x4808 6044		GPT10
	0x4808 8044		GPT11
Description	This register holds the second captured value of the counter register.		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CAPTURE_ VALUE2																															

Bits	Field Name	Description	Type	Reset
31:0	CAPTURE_ VALUE2	The value of second captured counter register	R	0x00000000

Table 16-45. Register Call Summary for Register TCAR2

General-Purpose Timers

- Capture Mode Functionality: [0] [1] [2] [3] [4] [5]
- Reading From Timer Counter Registers: [6]

General-Purpose Timers Register Manual

- GP Timer Register Mapping Summary: [7] [8] [9]
- GP Timer Register Descriptions: [10]

Table 16-46. TPIR

Address Offset	0x048		
Physical Address	0x4831 8048	**Instance**	GPT1
	0x4903 2048		GPT2
	0x4808 6048		GPT10
Description	This register is used for 1 ms tick generation. The TPIR register holds the value of the positive increment. The value of this register is added with the value of the TCVR to define whether next value loaded in TCRR will be the sub-period value or the over-period value.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
POSITIVE_INC_VALUE																															

Bits	Field Name	Description	Type	Reset
31:0	POSITIVE_INC_VALUE	The value of positive increment.	RW	0x00000000

Table 16-47. Register Call Summary for Register TPIR

General-Purpose Timers

- 1-ms Tick Generation (Only GPTIMER1, GPTIMER2, and GPTIMER10): [0] [1] [2] [3] [4] [5]
- Accessing GP Timer Registers: [6]
- Writing to Timer Registers: [7]
- Write Posting Synchronization Mode: [8]

General-Purpose Timers Register Manual

- GP Timer Register Mapping Summary: [9] [10]
- GP Timer Register Descriptions: [11]

Table 16-48. TNIR

Address Offset	0x04C		
Physical Address	0x4831 804C	**Instance**	GPT1
	0x4903 204C		GPT2
	0x4808 604C		GPT10
Description	This register is used for 1 ms tick generation. The TNIR register holds the value of the negative increment. The value of this register is added with the value of the TCVR to define whether next value loaded in TCRR will be the sub-period value or the over-period value.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
NEGATIVE_INC_VALUE																															

Bits	Field Name	Description	Type	Reset
31:0	NEGATIVE_INC_VALUE	The value of negative increment.	RW	0x00000000

Table 16-49. Register Call Summary for Register TNIR

General-Purpose Timers

- 1-ms Tick Generation (Only GPTIMER1, GPTIMER2, and GPTIMER10): [0] [1] [2] [3] [4] [5]
- Accessing GP Timer Registers: [6]
- Writing to Timer Registers: [7]
- Write Posting Synchronization Mode: [8]

General-Purpose Timers Register Manual

- GP Timer Register Mapping Summary: [9] [10]
- GP Timer Register Descriptions: [11]

Table 16-50. TCVR

Address Offset	0x050		
Physical Address	0x4831 8050	**Instance**	GPT1
	0x4903 2050		GPT2
	0x4808 6050		GPT10
Description	This register is used for 1 ms tick generation. The TCVR register defines whether next value loaded in TCRR will be the sub-period value or the over-period value.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
COUNTER_VALUE																															

Bits	Field Name	Description	Type	Reset
31:0	COUNTER_VALUE	The value of CVR counter.	RW	0x00000000

Table 16-51. Register Call Summary for Register TCVR

General-Purpose Timers

- 1-ms Tick Generation (Only GPTIMER1, GPTIMER2, and GPTIMER10): [0] [1]
- Accessing GP Timer Registers: [2]
- Writing to Timer Registers: [3]
- Write Posting Synchronization Mode: [4]

General-Purpose Timers Register Manual

- GP Timer Register Mapping Summary: [5] [6]
- GP Timer Register Descriptions: [7] [8] [9]

Table 16-52. TOCR

Address Offset	0x054		
Physical Address	0x4831 8054	**Instance**	GPT1
	0x4903 2054		GPT2
	0x4808 6054		GPT10
Description	This register is used to mask the tick interrupt for a selected number of ticks.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								OVF_COUNTER_VALUE																							

Bits	Field Name	Description	Type	Reset
31:24	Reserved	Reads return 0.	RW	0x00
23:0	OVF_COUNTER_VALUE	The number of overflow events.	RW	0x00000000

Table 16-53. Register Call Summary for Register TOCR

General-Purpose Timers

- 1-ms Tick Generation (Only GPTIMER1, GPTIMER2, and GPTIMER10): [0] [1] [2] [3] [4]
- Accessing GP Timer Registers: [5]
- Writing to Timer Registers: [6]
- Write Posting Synchronization Mode: [7]

General-Purpose Timers Register Manual

- GP Timer Register Mapping Summary: [8] [9]

Table 16-54. TOWR

Address Offset	0x058		
Physical Address	0x4831 8058	**Instance**	GPT1
	0x4903 2058		GPT2
	0x4808 6058		GPT10
Description	This register holds the number of masked overflow interrupts.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								OVF_WRAPPING_VALUE																							

Bits	Field Name	Description	Type	Reset
31:24	Reserved	Reads return 0.	RW	0x00
23:0	OVF_WRAPPING_VALUE	The number of masked interrupts.	RW	0x00000000

Table 16-55. Register Call Summary for Register TOWR

General-Purpose Timers • 1-ms Tick Generation (Only GPTIMER1, GPTIMER2, and GPTIMER10): [0] [1] [2] • Accessing GP Timer Registers: [3] • Writing to Timer Registers: [4] • Write Posting Synchronization Mode: [5]
General-Purpose Timers Register Manual • GP Timer Register Mapping Summary: [6] [7]

16.4 Watchdog Timers

16.4.1 WDTs Overview

The device includes two instances of the 32-bit WDT: WDT2 and WDT3. Figure 16-14 shows how each timer is connected in the device.

Note: WDTi (where i is the watchdog timer instance: i = 2 or 3) stands for the following:

- WDT2: Watchdog timer 2, also called MPU watchdog timer
- WDT3: Watchdog timer 3, also called IVA2 watchdog timer

Each WDT is an upward counter capable of generating both a pulse on the reset pin and an interrupt to the device system modules following an overflow condition. The MPU WDT serves resets to the PRCM module (its interrupt outputs are unused), and the IVA2 WDT serves watchdog interrupts to the MPU (its reset outputs are unused).

The WDTs can be accessed, loaded, and cleared by registers through the L4 interface. The MPU and IVA2 WDTs have the 32-kHz clock for their timer clock input.

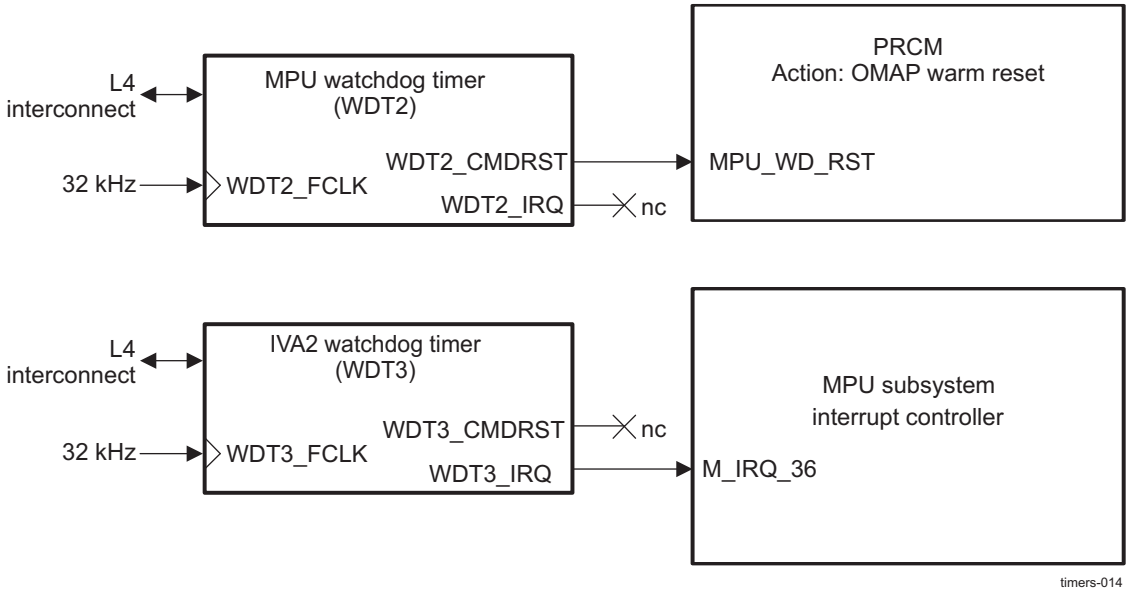
The MPU WDT directly generates a warm reset condition on overflow. The IVA2 WDT generates an MPU interrupt condition on overflow, which can be used by the application software via the PRCM to indirectly trigger a reset condition (that is, to the IVA2 subsystem).

The MPU WDT connects to a single target agent port on the L4 interconnect.

Table 16-56. WD Timers Default State for GP and EMU devices

Timer	Device			
	EMU		GP	
MPU WDTIMER2	Enabled	Running	Enabled	Running
IVA2 WDTIMER3	Enabled	NOT Running	Enabled	NOT Running

Figure 16-14. WDTs Block Diagram



16.4.1.1 WDT Features

The following are the main features of the WDT controllers:

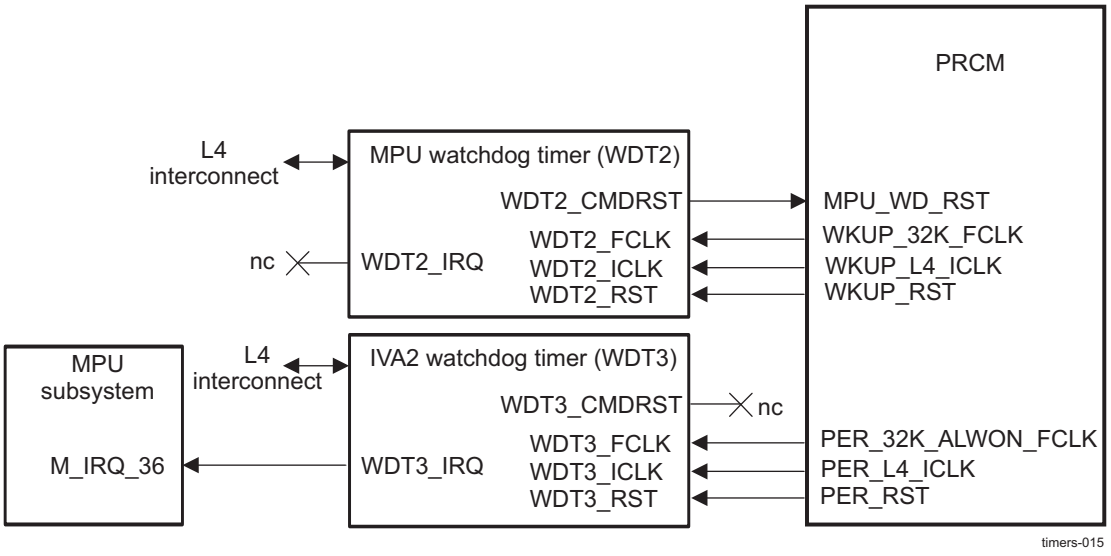
- L4 slave interface support:

  - 32-bit data bus width
  - 32-/16-bit access supported
  - 8-bit access not supported
  - 11-bit address bus width
  - Burst mode not supported
  - Write nonposted transaction mode only
- Free-running 32-bit upward counter
- Programmable divider clock source (2^n with n=[0:7])
- On-the-fly read/write register (while counting)
- Subset programming model of the GP timer
- WDTs are reset either on power-on or after a warm reset.
- Reset or interrupt actions when a timer overflow condition occurs
- WDT generates either a reset or an interrupt in its hardware integration (WDT2, or WDT3).

16.4.2 WDT Integration

Figure 16-15 shows the integration of the WDT in the device.

Figure 16-15. WDT Integration



16.4.2.1 Clocking, Reset, and Power-Management Scheme

16.4.2.1.1 Clock Management

There are two clock domains in the WDTs:

- Functional clock domain: WDTi_FCLK is the WDT functional clock. It is used to clock the WDT internal logic.
- Interface clock domain: WDTi_ICLK is the WDT interface clock. It is used to synchronize the WDT L4 port to the L4 interconnect. All accesses from the interconnect are synchronous to WDTi_ICLK.

Table 16-57 lists the the source clocks for each WDT in the device. For more information on clock control and domains, see Chapter 4, *Power, Reset, and Clock Management*.

Table 16-57. Clock, Power, and Reset Domains for WDTs

Timer	Interface Clock	Functional Clock	Power Domain
MPU WDT	WKUP_L4_ICLK	WKUP_32K_FCLK	WKUP
IVA2 WDT	PER_L4_ICLK	PER_32K_ALWON_FCLK	PER

From a global system power-management perspective, when one or both of the WDT clocks is no longer required, the WDTs can be deactivated at the PRCM level in the corresponding registers. Table 16-58 lists the WDT PRCM clock control bits.

Table 16-58. WDT PRCM Clock Control Bits

Name	Associated PRCM Clock Output	Enable Bit	Autoidle Bit
WDT2_FCLK	WKUP_32K_FCLK	PRCM.CM_FCLKEN_WKUP[5] EN_WDT2 bit	N/A
WDT3_FCLK	PER_32K_ALWON_FCLK	PRCM.CM_FCLKEN_PER[12] EN_WDT3 bit	N/A
WDT2_ICLK	WKUP_L4_ICLK	PRCM.CM_ICLKEN_WKUP[5] EN_WDT2 bit	PRCM.CM_AUTOIDLE_WKUP[5] AUTO_WDT2 bit
WDT3_ICLK	PER_L4_ICLK	PRCM.CM_ICLKEN_PER[12] EN_WDT3 bit	PRCM.CM_AUTOIDLE_PER[12] AUTO_WDT3 bit

Notes:

- The PRCM function clock outputs are gated at the PRCM level, assuming the modules that share it have been disabled in the corresponding register. Disabling the WDTs is a necessary but not sufficient action.The clock is effectively out when all PRCM conditions are fulfilled. For the other actions to be taken, see Chapter 4, *Power, Reset, and Clock Management*.
- The PRCM interface clock outputs are gated at the PRCM level, assuming the modules that share it have been disabled in the corresponding register. Disabling the WDTs is a necessary but not sufficient condition. The clock is effectively out when all PRCM conditions are fulfilled.For the other actions to be taken, see Chapter 4, *Power, Reset, and Clock Management*.
- The PRCM AUTOIDLE bits are used to link/unlink the WDTs from the clock domain transitions related to the GPTi_ICLK clocks.
- For further details about source clocks gating and domain transitions, see Chapter 4, *Power, Reset, and Clock Management.*

At the PRCM level, when the conditions to shut off the PRCM functional or interface output clocks are met (see Chapter 4, *Power, Reset, and Clock Management,* for details), the PRCM automatically launches a hardware handshake protocol to ensure the WDT is ready to have its clocks switched off. Namely, the PRCM module asserts an IDLE request to the WDT.

Although this handshake is a hardware function and out of software control, the way the WDT acknowledges the PRCM IDLE request is configurable through the WDTi.WD_SYSCONFIG[4:3] IDLEMODE bit. Table 16-59 lists the IDLEMODE settings and the related acknowledgement modes.

Table 16-59. IDLEMODE Settings

IDLEMODE Value	Selected Mode	Description
00	Force-idle	The WDT acknowledges unconditionally the IDLE request from the PRCM module, regardless of its internal operations. This mode must be used carefully, because it does not prevent loss of data when the clock is switched off.
01	No-idle	The WDT never acknowledges an IDLE request from the PRCM module. This mode is safe from a module point of view, because it ensures the clocks remain active; it is not efficient from a power-saving perspective, however, because it does not allow the PRCM output clock to be shut off and thus the power domain to be set to a lower power state.

Table 16-59. IDLEMODE Settings (continued)

IDLEMODE Value	Selected Mode	Description
10	Smart-idle	The WDT acknowledges the IDLE request, basing its decision on its internal activity. The acknowledge signal is asserted only when all pending transactions and IRQs requests are treated. This is the best approach for efficient system power management.
11	Reserved	

When configured in smart-idle mode, the WDT also offers an additional granularity on WDTi_FCLK and WDTi_ICLK gating. The WDTi.WD_SYSCONFIG[9:8] CLOCKACTIVITY bit field is used to determine which clock will be shut down (WDTi_FCLK, WDTi_ICLK, neither of them, or both of them).

The CLOCKACTIVITY setting is used internally to the WDT to determine the part of the module on which the conditions to acknowledge the PRCM IDLE request will be tested. For example, if WDTi_FCLK is not to be shut down on a PRCM IDLE request, the WDT considers only WDTi_ICLK and the associated pending activities before acknowledging the request.

Some WDT features are associated with WDTi_ICLK, and others are associated with WTDi_FCLK. Using the CLOCKACTIVITY setting along with the smart-idle mode ensures that the features associated with the clock that will remain active are always enabled, even if the WDT acknowledges an IDLE request.

Table 16-60 lists the CLOCKACTIVITY settings.

Table 16-60. CLOCKACTIVITY Settings

CLOCKACTIVITY Value	WDTi_ICLK Effects	WDTi_FCLK Effects	Description
00	OFF	OFF	Both WDTi_ICLK and WDTi_FCLK are considered for generating the acknowledge. This setting also means both WDTi_FCLK and WDTi_ICLK are likely to be shut down on a PRCM IDLE request.
01	OFF	ON	WDTi_FCLK is not shut down on a PRCM IDLE request; only WDTi_ICLK is concerned.
10	ON	OFF	WDTi_ICLK is not shut down on a PRCM IDLE request; only WDTi_FCLK is concerned.
11	ON	ON	None of the clocks are shut down. Therefore, the WDT can potentially acknowledge the IDLE request without checking the internal functionalities linked to its clocks.

CAUTION

The PRCM module does *not* have any hardware means to read the CLOCKACTIVITY settings. The software must ensure consistent programming between the WDT CLOCKACTIVITY and the PRCM functional clock and interface clock control bits. If the WDT is disabled in both the CM_FCLKEN and CM_ICLKEN PRCM registers while CLOCKACTIVITY is set to 11, nothing prevents the PRCM module from asserting its IDLE request, which is acknowledged regardless of the features associated with the WDT clocks. This can lead to unpredictable behavior.

16.4.2.1.2 Reset and Power Management

The MPU WDT (WDT2) belong to the WKUP power domain. As part of that domain, this WDT is sensitive to a WKUP_RST signal issued by the PRCM module. For further details about the WKUP power domain implementation and WKUP_RST signal, see Chapter 4, *Power, Reset, and Clock Management.*

The IVA WDT (WDT3) belongs to the PER power domain. As part of that domain, WDT3 is sensitive to the PER_RST signal issued by the PRCM module. For further details about the PER power domain implementation and PER_RST signal, see Chapter 4, *Power, Reset, and Clock Management.*

Note: WDT2 is reset on power-on or after a warm reset, and then it starts counting.

WDT3 is reset either on power-on or after a warm reset, and then it doesn't start counting.

16.4.2.2 Interrupts

Table 16-61 shows interrupt mapping from the three WDTs to the MPU.

Table 16-61. WDT Interrupt Names and Processor IRQ Mapping

Timer	Interrupt Name	Mapping	Comments
MPU WDT	WDT2_IRQ	Not connected	
IVA2 WDT	WDT3_IRQ	M_IRQ_36	IVA2 WDT overflow

16.4.3 WDTs Functional Description

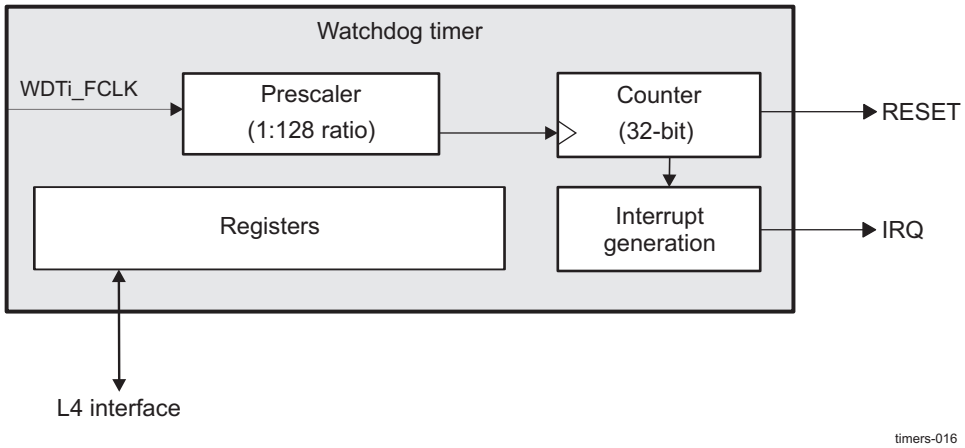
16.4.3.1 General WDT Operation

The WDTs are based on an upward 32-bit counter coupled with a prescaler. The counter overflow is signaled through two independent signals: a simple reset signal and an interrupt signal, both active low. The use of these signals depends on whether they are connected or not. For this information, see Figure 16-14. The interrupt generation mechanism is controlled through the WDTi.WIER and WDTi.WISR registers.

The prescaler ratio can be set between 1 and 128 by accessing the WDTi.WCLR[4:2] PTV and WDTi.WCLR[5] PRE fields of the watchdog control register (WDTi.WCLR).

The current timer value can be accessed on-the-fly by reading the WDT counter register (WDTi.WCRR), modified by accessing the WDT load register (WDTi.WLDR) (no on-the-fly update), or reloaded by following a specific reload sequence on the WDT trigger register (WDTi.WTGR). A start/stop sequence applied to the WDT start/stop register (WDTi.WSPR) can start and stop the WDT.

Figure 16-16. 32-Bit WDT Functional Block Diagram



16.4.3.2 Reset Context

After reset, the WDTs are enabled. Table 16-62 lists the default reset values of the three WDT load registers (WDTi.WLDR) and prescaler ratios (WDTi.WCLR[4:2] PTV field). To get these values, software must read the corresponding WDTi.WCLR[4:2] PTV fields and the 32-bit register to retrieve the static configuration of the module.

Table 16-62. Count and Prescaler Default Reset Values

Timer	WLDR Reset Value	PTV Reset Value
OMAP WDT (WDT2)	0xFFFB 0000	0
IVA2 WDT (WDT3)	0xFFFB 0000	0

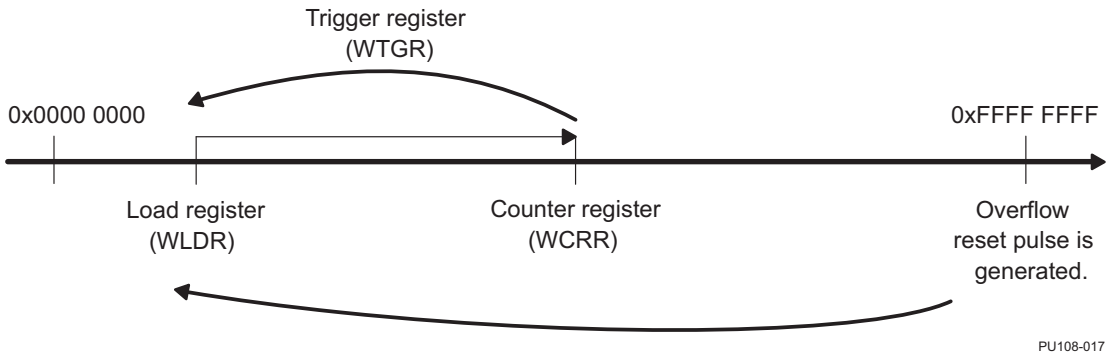
16.4.3.3 Overflow/Reset Generation

When the WDT counter register (WDTi.WCRR) overflows, an active-low reset pulse is generated to the PRCM module. This pulse is one prescaled timer clock cycle wide and occurs at the same time as the timer counter overflow.

After reset generation, the counter is automatically reloaded with the value stored in the watchdog load register (WDTi.WLDR) and the prescaler is reset (the prescaler ratio remains unchanged). Then, after the reset pulse output has been generated, the timer counter begins incrementing again.

Figure 16-17 shows a general functional view of the WDT.

Figure 16-17. WDT General Functional View



16.4.3.4 Prescaler Value/Timer Reset Frequency

Each WDT is composed of a prescaler stage and a timer counter.

The timer rate is defined by the following values:

- Value of the prescaler fields (the WDTi.WCLR[5] PRE bit and the WDTi.WCLR[4:2] PTV field)
- Value loaded into the timer load register (WDTi.WLDR)

The prescaler stage is clocked with the timer clock and acts as a clock divider for the timer counter stage. The ratio is managed by accessing the ratio definition field (the WDTi.WCLR[4:2] PTV field) and is enabled with the WDTi.WCLR[5] PRE bit.

Table 16-63 lists the prescaler clock ratio values.

Table 16-63. Prescaler Clock Ratios

PRE Bit (in WDTi.WCLR Register)	PTV Bits (in WDTi.WCLR Register)	Clock Divider (PS)
0	X	1
1	0	1
1	1	2
1	2	4
1	3	8
1	4	16
1	5	32
1	6	64
1	7	128

The watchdog timer overflow rate is expressed by: OVF_Rate = (0xFFFF FFFF - WDTn.WLDR + 1) * (wd-functional clock period) * PS

With wd-functional clock period = 1 / (wd-functional clock frequency) and PS = $2^{(PTV)}$.

CAUTION

Internal resynchronization causes any software write to GPTn.WSPR to have some latency before WSPR is updated with the programmed value:

1.5 * functional clock cycles ≤ write_WSPR_latency ≤ 2.5 * functional clock cycles

Remember to take this latency into account whenever the watchdog must be started or stopped.

For example, for a timer clock input of 32 kHz with a prescaler ratio value of 0x1 (clock divided by 2) and WDTi.WCLR[5] PRE bit = 1 (clock divider enabled), the reset period is as listed in Table 16-64.

Table 16-64. Reset Period Examples

WDTi.WLDR Value	Reset Period
0x0000 0000	74 h 56 min
0xFFFF 0000	4 s
0xFFFF FFF0	1 ms
0xFFFF FFFF	62.5 us

CAUTION

- Ensure that the reloaded value allows the correct operation of the application. When a WDT is enabled, the software must periodically trigger a reload before the counter overflows. Hence, the WDTi.WLDR[31:0] value must be chosen according to the ongoing activity preceding the watchdog reload.
- Due to design reasons, WDTi.WLDR[31:0] = 0xFFFF FFFF is a special case, although such a WDTi.WLDR value is meaningless. When WDTi.WLDR is programmed with the overflow value, a triggering event generates a reset/interrupt one functional clock cycle later, even if the WDT is stopped.

Table 16-65 lists the default reset periods for the WDTs.

Table 16-65. Default WDT Time Periods

WDTs	Clock Source	Default Reset Period
OMAP/IVA2 WDTs	32 kHz	10 s

16.4.3.5 Triggering a Timer Reload

To reload the timer counter and reset the prescaler before reaching overflow, a reload command is executed by accessing the WDT trigger register (WDTi.WTGR) using a specific reload sequence.

The specific reload sequence is performed whenever the written value on the WDT trigger register (WDTi.WTGR) differs from its previous value. In this case, reload is executed in the same way as an overflow autoreload, but without a reset pulse generation.

The timer counter is loaded with the WDT load register value (the WDTi.WLDR[31:0] TIMER_LOAD field), and the prescaler is reset.

16.4.3.6 Start/Stop Sequence for WDTs (Using WDTi.WSPR Register)

To start/stop a WDT, access must be made through the start/stop register (WDTi.WSPR) using a specific sequence.

To disable the timer, follow this sequence:

1. Write 0xXXXX AAAA in WDTi.WSPR.
2. Write 0xXXXX 5555 in WDTi.WSPR.

To enable the timer, follow this sequence:

1. Write 0xXXXX BBBB in WDTi.WSPR.
2. Write 0xXXXX 4444 in WDTi.WSPR.

All other write sequences on WDTi.WSPR have no effect on the start/stop feature of the module.

16.4.3.7 Modifying Timer Count/Load Values and Prescaler Setting

To modify the timer counter value (WDTi.WCRR), prescaler ratio (the WDTi.WCLR[4:2] PTV field), or load value (the WDTi.WLDR[31:0] TIMER_LOAD field), the WDT must be disabled by using the start/stop sequence (the WDTi.WSPR register).

After a write access, the load register value and prescaler ratio registers are updated immediately, but new values are considered only after the next consecutive counter overflow or after a new trigger command (WDTi.WTGR).

16.4.3.8 Watchdog Counter Register Access Restriction (WDTi.WCRR Register)

Because the WDTi.WCRR register is directly related to the timer counter value and is updated on the timer clock (WDTi_FCLK), a 32-bit shadow register is implemented to read a coherent value of the WDTi.WCRR register. The shadow register is updated by a 16-bit LSB read command.

Note: Although the L4 clock (WDTi_ICLK) is completely asynchronous with the timer clock (WDTi_FCLK), some synchronization is done to ensure that the WDTi.WCRR value is not read while it is being incremented.

When 32-bit read access is performed, the shadow register is not updated. Read access is made directly from the accessed register.

To ensure that a coherent value is read inside WDTi.WCRR, the first read access is to the lower 16 bits (offset = 0x08), followed by read access to the upper 16 bits (offset = 0x0A).

16.4.3.9 WDT Interrupt Generation

The WDT issues an overflow interrupt if this interrupt is enabled in the WDT interrupt enable register (WDTi.WIER[0] OVF_IT_ENA = 1). When the overflow occurs, the interrupt status bit (the WDTi.WISR[0] OVF_IT_FLAG bit) is set to 1. The output interrupt line (WDTi_IRQ) is asserted (active low) when both status (OVF_IT_FLAG) and enable (OVF_IT_ENA) flags are set to 1; the order is not relevant. Writing 1 in the enable bit (the status is already set at 1) also triggers the interrupt in the normal order (enable first, status after). The pending interrupt event is cleared when the set status bit is overwritten by a value of 1 by a write command in the WTDi.WISR register. Reading the WTDi.WISR register and writing the value back allows a fast acknowledge interrupt process.

Note: Writing 0 in the WDTi.WISR[0] OVF_IT_FLAG bit has no effect on it.

Because the interrupt event is generated on the functional clock domain (WDTi_FCLK), during the interrupt status register (WDTi.WISR) update, the two clock domains are resynchronized.

16.4.3.10 WDT Under Emulation

During emulation mode, the WDT can/cannot continue running, according on the value of the WDTi.WD_SYSCONFIG[5] EMUFREE bit of the system configuration register (WDTi.WD_SYSCONFIG).

- When EMUFREE is 1, WDT execution is not stopped and a reset pulse is still generated when overflow is reached.
- When EMUFREE is 0, the counters (prescaler/timer) are frozen and incrementation restarts after exiting from emulation mode.

16.5 Watchdog Timer Register Manual

All registers are 32 bits wide and are accessible through the L4 interface with 16-bit or 32-bit access (read/write).

16.5.1 Instance Summary

Table 16-66 lists the base address and address space for the WDT module instances. All timers are memory mapped to the L4 peripheral bus memory space.

Table 16-66. WDT Instance Summary

Module Name	Base Address	Size
WDTIMER2	0x4831 4000	4K bytes
WDTIMER3	0x4903 0000	4K bytes

16.5.2 WDT Register Mapping Summary

CAUTION

The WDT registers are limited to 32-bit and 16-bit data accesses; 8-bit access is not allowed and can corrupt register content.

Table 16-67 lists the WDT2 registers and Table 16-68 lists the WDT3 registers.

Table 16-67. WDTIMER2 Register Summary

Register Name	Type	Register Width (Bits)	Address Offset	WDTIMER2 Physical Address
WIDR	R	32	0x000	0x4831 4000
WD_SYSCONFIG	RW	32	0x010	0x4831 4010
WD_SYSSTATUS	R	32	0x014	0x4831 4014
WISR	RW	32	0x018	0x4831 4018
WIER	RW	32	0x01C	0x4831 401C
WCLR	RW	32	0x024	0x4831 4024
WCRR	RW	32	0x028	0x4831 4028
WLDR	RW	32	0x02C	0x4831 402C
WTGR	RW	32	0x030	0x4831 4030
WWPS	R	32	0x034	0x4831 4034
WSPR	RW	32	0x048	0x4831 4048

Table 16-68. WDTIMER3 Register Summary

Register Name	Type	Register Width (Bits)	Address Offset	WDTIMER3 Physical Address
WIDR	R	32	0x000	0x4903 0000
WD_SYSCONFIG	RW	32	0x010	0x4903 0010
WD_SYSSTATUS	R	32	0x014	0x4903 0014
WISR	RW	32	0x018	0x4903 0018
WIER	RW	32	0x01C	0x4903 001C
WCLR	RW	32	0x024	0x4903 0024
WCRR	RW	32	0x028	0x4903 0028
WLDR	RW	32	0x02C	0x4903 002C

Table 16-68. WDTIMER3 Register Summary (continued)

Register Name	Type	Register Width (Bits)	Address Offset	WDTIMER3 Physical Address
WTGR	RW	32	0x030	0x4903 0030
WWPS	R	32	0x034	0x4903 0034
WSPR	RW	32	0x048	0x4903 0048

16.5.3 WDT Register Descriptions

Table 16-69 through Table 16-89 describe the WDT register bits.

Table 16-69. WIDR

Address Offset	0x000		
Physical Address	0x4831 4000	**Instance**	WDTIMER2
	0x4903 0000		WDTIMER3
Description	This register contains the IP revision code.		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																								WD_REV							

Bits	Field Name	Description	Type	Reset
31:8	Reserved	Reads return 0.	R	0x000000
7:0	WD_REV	IP revision [7:4] Major revision [3:0] Minor revision Examples: 0x10 for 1.0, 0x21 for 2.1	R	See (1)

(1) TI internal data

Table 16-70. Register Call Summary for Register WIDR

Watchdog Timer Register Manual

- WDT Register Mapping Summary: [0] [1]

Table 16-71. WD_SYSCONFIG

Address Offset	0x010		
Physical Address	0x4831 4010	**Instance**	WDTIMER2
	0x4903 0010		WDTIMER3
Description	This register controls the various parameters of the L4 interface.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																						CLOCKACTIVITY		Reserved		EMUFREE		IDLEMODE		ENAWAKEUP	SOFTRESET	AUTOIDLE

Bits	Field Name	Description	Type	Reset
31:10	Reserved	Write 0s for future compatibility. Reads return 0.	R	0x0000000
9:8	CLOCKACTIVITY	Clock Activity selection bits. 0x0: Both clocks can be cut off 0x1: Only L4 clock must be kept active; timer clock can be cut off 0x2: Only timer clock must be kept active; L4 clock can be cut off 0x3: both clocks must be kept active	RW	0x0
7:6	Reserved	Write 0s for future compatibility. Reads return 0.	R	0x0000000
5	EMUFREE	Emulation mode 0x0: Timer counter frozen in emulation 0x1: Timer counter free-running in emulation	RW	0
4:3	IDLEMODE	Idle mode selection bits 0x0: Force Idle mode 0x1: No Idle mode 0x2: Smart Idle mode 0x3: Reserved	RW	0x0
2	ENAWAKEUP	Enable wakeup control bit 0x0: Wakeup mechanism is disabled 0x1: Wakeup mechanism is enabled	RW	0
1	SOFTRESET	Software reset. This bit is automatically reset by the hardware. During reads, it always return 0. 0x0: Normal mode 0x1: The module is reset.	RW	0
0	AUTOIDLE	L4 interconnect clock gating strategy 0x0: L4 interface clock is free-running. 0x1: Automatic L4 interface clock gating strategy is applied, based on the L4 interface activity.	RW	0

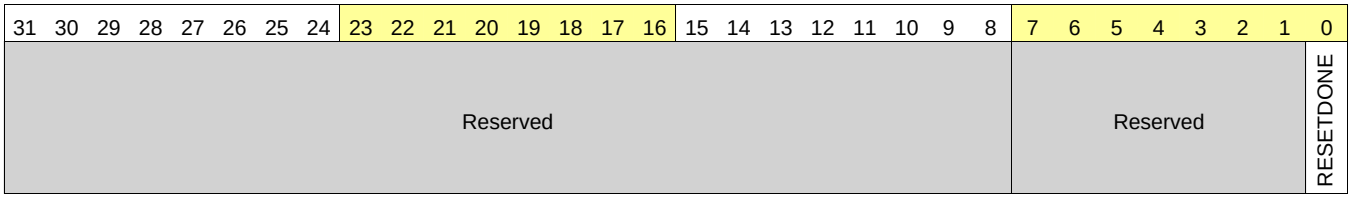
Table 16-72. Register Call Summary for Register WD_SYSCONFIG

Watchdog Timers
- Clock Management: [0] [1]
- WDT Under Emulation: [2] [3]

Watchdog Timer Register Manual
- WDT Register Mapping Summary: [4] [5]

Table 16-73. WD_SYSSTATUS

Address Offset	0x014		
Physical Address	0x4831 4014	**Instance**	WDTIMER2
	0x4903 0014		WDTIMER3
Description	This register provides status information about the module, excluding the interrupt status information.		
Type	R		



Bits	Field Name	Description	Type	Reset
31:8	Reserved	Reads return 0.	R	0x0000000
7:1	Reserved	Reads return 0.	R	0x00
0	RESETDONE	Internal reset monitoring	R	-
		0x0: Internal module reset in ongoing.		
		0x1: Reset completed.		

Table 16-74. Register Call Summary for Register WD_SYSSTATUS

Watchdog Timer Register Manual
- WDT Register Mapping Summary: [0] [1]

Table 16-75. WISR

Address Offset	0x018		
Physical Address	0x4831 4018	**Instance**	WDTIMER2
	0x4903 0014		WDTIMER3
Description	This register shows which interrupt events are pending inside the module.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																															OVF_IT_FLAG

Bits	Field Name	Description	Type	Reset
31:1	Reserved	Reads return 0.	R	0x00000000
0	OVF_IT_FLAG	Pending overflow interrupt status	RW	0
		Read 0x0: No overflow interrupt pending		
		Write 0x0: Status unchanged		
		Read 0x1: Overflow interrupt pending		
		Write 0x1: Status bit cleared		

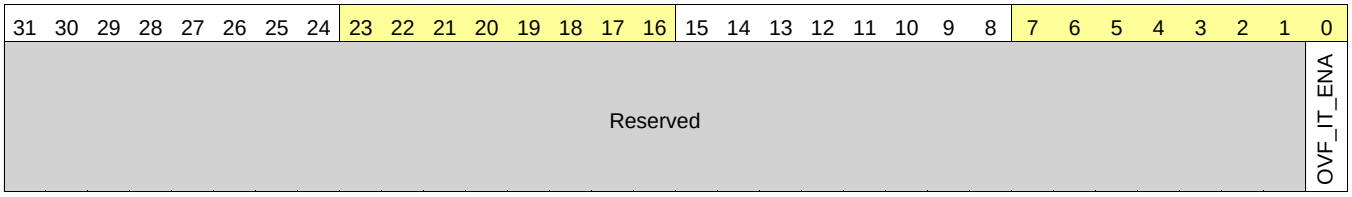
Table 16-76. Register Call Summary for Register WISR

Watchdog Timers
- General WDT Operation: [0]
- WDT Interrupt Generation: [1] [2] [3] [4] [5]

Watchdog Timer Register Manual
- WDT Register Mapping Summary: [6] [7]

Table 16-77. WIER

Address Offset	0x01C		
Physical Address	0x4831 4018	**Instance**	WDTIMER2
	0x4903 0014		WDTIMER3
Description	This register shows controls (enable/disable) the interrupt events.		
Type	RW		



Bits	Field Name	Description	Type	Reset
31:1	Reserved	Reads return 0.	R	0x00000000
0	OVF_IT_ENA	Enable overflow interrupt	RW	0
		0x0: Disable overflow interrupt.		
		0x1: Enable overflow interrupt.		

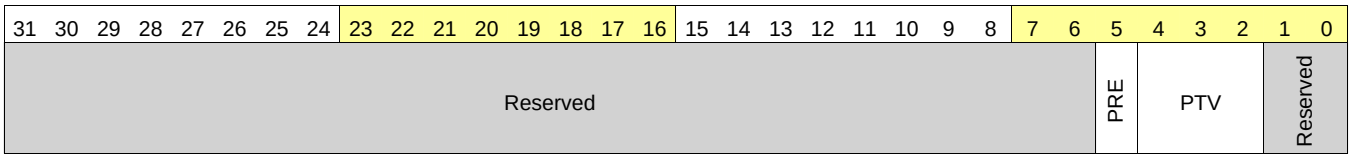
Table 16-78. Register Call Summary for Register WIER

Watchdog Timers
- General WDT Operation: [0]
- WDT Interrupt Generation: [1]

Watchdog Timer Register Manual
- WDT Register Mapping Summary: [2] [3]

Table 16-79. WCLR

Address Offset	0x024		
Physical Address	0x4831 4024	**Instance**	WDTIMER2
	0x4903 0024		WDTIMER3
Description	This register controls the prescaler stage of the counter.		
Type	RW		



Bits	Field Name	Description	Type	Reset
31:6	Reserved	Reads return 0.	R	0x0000000
5	PRE	Prescaler enable	RW	1
		0x0: Prescaler disabled		
		0x1: Prescaler enabled		
4:2	PTV	Prescaler value: The timer counter is prescaled with the value: pow(2,PTV) Example: PTV = 2: counter increases value is started after 4 functional clock periods.	RW	0x0
1:0	Reserved	Reads return 0.	R	0x0

Table 16-80. Register Call Summary for Register WCLR

Watchdog Timers

- General WDT Operation: [0] [1] [2]
- Reset Context: [3] [4]
- Prescaler Value/Timer Reset Frequency: [5] [6] [7] [8] [9] [10] [11]
- Modifying Timer Count/Load Values and Prescaler Setting: [12]

Watchdog Timer Register Manual

- WDT Register Mapping Summary: [13] [14]
- WDT Register Descriptions: [15]

Table 16-81. WCRR

Address Offset	0x028		
Physical Address	0x4831 4028	**Instance**	WDTIMER2
	0x4903 0028		WDTIMER3
Description	This register holds the value of the internal counter.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TIMER_COUNTER																															

Bits	Field Name	Description	Type	Reset
31:0	TIMER_COUNTER	The value of the timer counter register	RW	0x00000000

Table 16-82. Register Call Summary for Register WCRR

Watchdog Timers

- General WDT Operation: [0]
- Overflow/Reset Generation: [1]
- Modifying Timer Count/Load Values and Prescaler Setting: [2]
- Watchdog Counter Register Access Restriction (WDTi.WCRR Register): [3] [4] [5] [6]

Watchdog Timer Register Manual

- WDT Register Mapping Summary: [7] [8]
- WDT Register Descriptions: [9]

Table 16-83. WLDR

Address Offset	0x02C		
Physical Address	0x4831 402C	**Instance**	WDTIMER2
	0x4903 002C		WDTIMER3
Description	This register holds the timer load value.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TIMER_LOAD																															

Bits	Field Name	Description	Type	Reset
31:0	TIMER_LOAD	The value of the timer load register	RW	0xFFA6 0000 (WDT1) 0xFFFB 000 (WDT2 and 3)

Table 16-84. Register Call Summary for Register WLDR

Watchdog Timers
- General WDT Operation: [0]
- Reset Context: [1] [2]
- Overflow/Reset Generation: [3]
- Prescaler Value/Timer Reset Frequency: [4] [5] [6] [7] [8] [9] [10]
- Triggering a Timer Reload: [11]
- Modifying Timer Count/Load Values and Prescaler Setting: [12]

Watchdog Timer Register Manual
- WDT Register Mapping Summary: [13] [14]
- WDT Register Descriptions: [15]

Table 16-85. WTGR

Address Offset	0x030		
Physical Address	0x4831 4030	**Instance**	WDTIMER2
	0x4903 0030		WDTIMER3
Description	Writing a different value than the one already written in this register causes a watchdog counter reload.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TTGR_VALUE																															

Bits	Field Name	Description	Type	Reset
31:0	TTGR_VALUE	The value of the trigger register	RW	0x00000000

Table 16-86. Register Call Summary for Register WTGR

Watchdog Timers
- General WDT Operation: [0]
- Triggering a Timer Reload: [1] [2]
- Modifying Timer Count/Load Values and Prescaler Setting: [3]

Watchdog Timer Register Manual
- WDT Register Mapping Summary: [4] [5]
- WDT Register Descriptions: [6]

Table 16-87. WWPS

Address Offset	0x034		
Physical Address	0x4831 4034	**Instance**	WDTIMER2
	0x4903 0034		WDTIMER3
Description	This register contains the write posting bits for all write-able functional registers.		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																											W_PEND_WSPR	W_PEND_WTGR	W_PEND_WLDR	W_PEND_WCRR	W_PEND_WCLR

Bits	Field Name	Description	Type	Reset
31:5	Reserved	Reads return 0	R	0x0
4	W_PEND_WSPR	Write pending for register WSPR 0x0: No Start-Stop Register write pending 0x1: Start-Stop Register write pending	R	0
3	W_PEND_WTGR	Write pending for register WTGR 0x0: No Trigger Register write pending 0x1: Trigger Register write pending	R	0
2	W_PEND_WLDR	Write pending for register WLDR 0x0: No Load Register write pending 0x1: Load Register write pending	R	0
1	W_PEND_WCRR	Write pending for register WCRR 0x0: No Counter Register write pending 0x1: Counter Register write pending	R	0
0	W_PEND_WCLR	Write pending for register WCLR 0x0: No Control Register write pending 0x1: Control Register write pending	R	0

Table 16-88. Register Call Summary for Register WWPS

Watchdog Timer Register Manual
- WDT Register Mapping Summary: [0] [1]

Table 16-89. WSPR

Address Offset	0x048		
Physical Address	0x4831 4048	**Instance**	WDTIMER2
	0x4903 0048		WDTIMER3
Description	This register holds the start-stop value that controls the internal start-stop FSM.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
WSPR_VALUE																															

Bits	Field Name	Description	Type	Reset
31:0	WSPR_VALUE	The value of the start/stop register	RW	0x00000000

Table 16-90. Register Call Summary for Register WSPR

Watchdog Timers
- General WDT Operation: [0]
- Prescaler Value/Timer Reset Frequency: [1] [2]
- Start/Stop Sequence for WDTs (Using WDTi.WSPR Register): [3] [4] [5] [6] [7] [8]
- Modifying Timer Count/Load Values and Prescaler Setting: [9]

Watchdog Timer Register Manual
- WDT Register Mapping Summary: [10] [11]
- WDT Register Descriptions: [12]

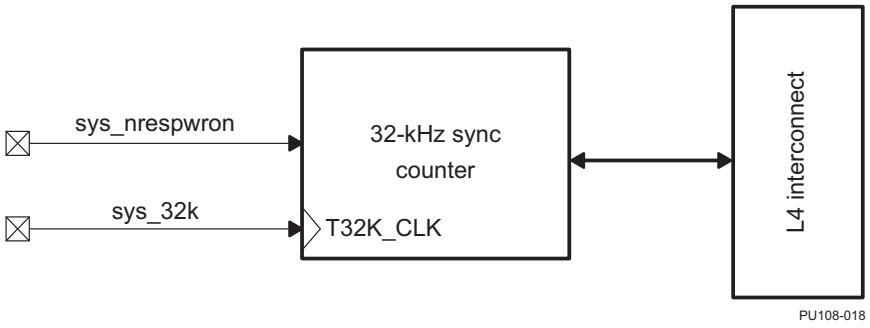
16.6 32-kHz Synchronized Timer

16.6.1 32-kHz Sync Timer Functional Description

The 32-kHz sync timer is a 32-bit counter, clocked by the falling edge of the 32-kHz system clock. It is reset while the external asynchronous power-up reset (sys_nrespwron) primary I/O is active (main device reset). When sys_nrespwron is released (on the rising edge of sys_nrespwron), after three 32-kHz clock periods, the counter starts counting up from the reset value of the counter register on the falling edge of the 32-kHz system clock. After reaching its highest value, the counter wraps back to 0 and starts counting again. Figure 16-18 shows the block diagram of the 32-kHz sync timer.

Note: sys_nrespwron is an active low I/O.

Figure 16-18. 32-kHz Sync Timer Block Diagram



16.6.1.1 Reading the 32-kHz Sync Timer

The sync counter register is 32 bits wide and for correct count capture must be accessed as 16-bit LSB access first and two 16-bit MSB access last. Internal synchronization logic allows reading of the counter value while the counter is running. The time latency to read the counter is one L4 interconnect clock period.

16.6.1.2 32-kHz Sync Timer Features

The following are the main features of the 32-kHz sync timer controller:

- L4 slave interface support:
  - 32-bit data bus width
  - 32-/16-bit access supported
  - 8-bit access not supported
  - 16-bit address bus width
  - Burst mode not supported
  - Write nonposted transaction mode supported
- Only read operations are supported on the module registers; no write operation is supported (no error/no action on write).
- Free-running 32-bit upward counter
- Start and keep counting after power-on reset
- Automatic roll over to 0, highest value reached (0xFFFF FFFF)
- On-the-fly read (while counting)

16.6.2 32-kHz Sync Timer Environment

The sync timer is accessible only through the L4 interface.

16.6.3 32-kHz Sync Timer Integration

16.6.3.1 Clocking, Reset, and Power-Management Scheme

Table 16-91 lists the power and reset domain and the source clock for the 32-kHz sync timer in the device. For more information on power, reset, and clock control and domains, see Chapter 4, *Power, Reset, and Clock Management*.

Table 16-91. Clock, Power, and Reset Domains for 32-kHz Sync Timer

Timer	Source Clock	Intertface Clock	Clock and Power Domain	Reset Domain
32-kHz sync timer	sys_32k	32KSYNC_ICLK	WKUP	SYNCT_RST(1)

16.6.3.2 Interrupts

The 32-kHz sync timer has no interrupt outputs.

16.6.3.3 Sync Timer 32k and MSuspend Signal

By default, the Sync Timer 32k is not stopped when the MPU or DSP is in *halt*. To activate its sensitivity to the MSuspend signal, the CONTROL.CONTROL_MSUSPEND_2[29:27] SyncTMMsctrl bit has been added to the system control module.

16.7 32-kHz Sync Timer Register Manual

16.7.1 32-kHz Sync Timer Instance Summary

Table 16-92 lists the base address and block size for the 32-kHz sync timer. It is memory mapped to the L4 peripheral bus memory space.

Table 16-92. 32-kHz Sync Timer Instance Summary

Module Name	Base Address	Size
32-kHz Sync Timer	0x4832 0000	4K bytes

16.7.2 32-kHz Sync Timer Register Mapping Summary

CAUTION

The 32-kHz sync timer registers are limited to 32-bit and 16-bit data accesses; 8-bit access is not allowed and can corrupt the register content.

Table 16-93 lists the 32-kHz sync timer registers. Table 16-94 through Table 16-98 describe the register bits.

Table 16-93. 32-kHz Sync Timer Register Summary

Register Name	Type	Register Width (Bits)	Offset Address	32-kHz-Sync Timer Physical Address
REG_32KSYNCNT_REV	R	32	0x0000	0x4832 0000
REG_32KSYNCNT_SYSCONFIG	R/W	32	0x0004	0x4832 0004
REG_32KSYNCNT_CR	R	32	0x0010	0x4832 0010

16.7.3 32-kHz Sync Timer Register Descriptions

Table 16-94. REG_32KSYNCNT_REV

Address Offset	0x0000
Physical Address	0x4832 0000
Description	This register contains the sync counter IP revision code.
Type	R

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																								CID_REV							

Bits	Field Name	Description	Type	Reset
31:8	Reserved	Reads return 0.	R	0x000000
7:0	CID_REV	Counter revision number [7:4] = Major revision [3:0] = Minor revision (Examples: 0x10 for 1.0, 0x21 for 2.1)	R	See (1)

(1) TI internal data

Table 16-95. Register Call Summary for Register REG_32KSYNCNT_REV

32-kHz Sync Timer Register Manual

- 32-kHz Sync Timer Register Mapping Summary: [0]

Table 16-96. REG_32KSYNCNT_SYSCONFIG

Address Offset	0x0004
Physical Address	0x4832 0004
Description	This register is used for IDLE modes only.
Type	RW

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																											IDLEMODE		Reserved		

Bits	Field Name	Description	Type	Reset
31:5	Reserved	Reads return 0.	R	0x0
4:3	IDLEMODE	Power management REQ/ACK control	RW	0x0
		0x0: Force idle. An idle request is acknowledged unconditionally.		
		0x1: No-idle. An idle request is never acknowledged.		
		0x2: Reserved		
		0x3: Reserved		
2:0	Reserved	Reads return 0.	R	0x0

Table 16-97. Register Call Summary for Register REG_32KSYNCNT_SYSCONFIG

32-kHz Sync Timer Register Manual

- 32-kHz Sync Timer Register Mapping Summary: [0]

Table 16-98. REG_32KSYNCNT_CR

Address Offset	0x0010
Physical Address	0x4832 0010
Description	This register contains the 32-kHz sync counter value.
Type	R

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
COUNTER_VALUE																															

Bits	Field Name	Description	Type	Reset
31:0	COUNTER_VALUE	Counter register value	R	0x00000003

Table 16-99. Register Call Summary for Register REG_32KSYNCNT_CR

32-kHz Sync Timer Register Manual

- 32-kHz Sync Timer Register Mapping Summary: [0]

UART/IrDA/CIR

This chapter describes the function, operation, and configuration of the universal asynchronous receiver/transmitter (UART)/infrared data association (IrDA)/consumer infrared (CIR) module on the OMAP34xx multimedia device.

Topic **Page**

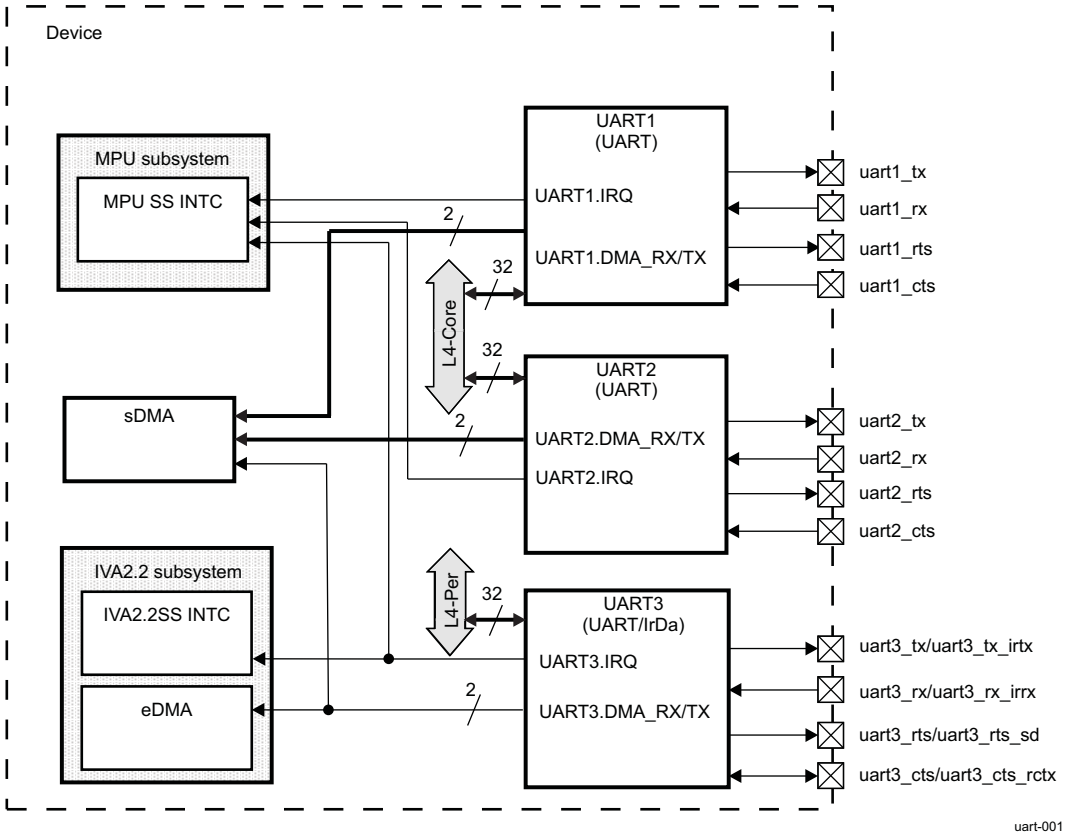
17.1 UART/IrDA/CIR Overview 2624
17.2 UART/IrDA/CIR Environment 2627
17.3 UART/IrDA/CIR Integration 2640
17.4 UART/IrDA/CIR Functional Description 2644
17.5 UART/IrDA/CIR Basic Programming Model 2674
17.6 UART/IrDA/CIR Register Manual 2682

17.1 UART/IrDA/CIR Overview

The OMAP34xx multimedia device contains three universal asynchronous receiver/transmitter (UART) devices controlled by the microprocessor unit (MPU) (see Figure 17-1):

- Two UART-only modules, UART1 and UART2, are pinned out for use as UART devices only. UART1 and UART2 must be programmed by setting the UARTi.MDR1_REG[2:0] MODE_SELECT field to one of the three UART operating modes.
- UART3, which adds infrared communication support, is pinned out for use as a UART, infrared data association (IrDA), or consumer infrared (CIR) device, and can be programmed to any available operating mode.

Figure 17-1. UART Module



17.1.1 UART Features

The UARTs (UART1, UART2, and UART3 when in UART mode) include the following key features:

- 16C750 compatibility
- 64-byte FIFO for receiver and 64-byte FIFO for transmitter
- Programmable interrupt trigger levels for FIFOs
- Baud generation based on programmable divisors N (N = 1…16,384) operating from a fixed functional clock of 48 MHz

Oversampling is programmed by software as 16 or 13; thus, the baud rate computation is either:

- Baud rate = (functional clock/16)/N
- Baud rate = (functional clock/13)/N

This software programming mode enables higher baud rates with the same error amount without changing the clock source:

- Break character detection and generation
- Configurable data format
  - Data bit: 5, 6, 7, or 8 bits
  - Parity bit: Even, odd, none
  - Stop-bit: 1, 1.5, 2 bit(s)
- Flow control: Hardware (RTS/CTS) or software (XON/XOFF)

The UART clocks are connected to produce a baud rate of up to 3.6M bits/s. Table 17-1 lists the supported baud rates, the requested divisor, and the corresponding error versus the standard baud rate.

Table 17-1. UART Mode Baud Rates, Divisor Values, and Error Rates

Baud Rate	Oversampling	Divisor	Error (%)
300	16	10000	0
600	16	5000	0
1200	16	2500	0
2400	16	1250	0
4800	16	625	0
9600	16	312	0.16
14,400	16	208	0.16
19,200	16	156	0.16
28,800	16	704	0.16
38,400	16	78	0.16
57,600	16	52	0.16
115,200	16	26	0.16
230,400	16	13	0.16
460,800	13	8	0.16
921,600	13	4	0.16
1,843,200	13	2	0.16
3,000,000	16	1	0
3,686,400	13	1	0.16

Note: The maximum baud rate required of the module does not depend on the change of OPP.

17.1.2 IrDA Features

The IrDA (UART3 only) includes the following key features:

- Support of IrDA 1.4 slow infrared (SIR), medium infrared (MIR), and fast infrared (FIR) communications
  - Frame formatting: Addition of variable beginning-of-frame (xBOF) characters and end-of-frame (EOF) characters
  - Uplink/downlink cyclic redundancy check (CRC) generation/detection
  - Asynchronous transparency (automatic insertion of break character)
  - Eight-entry status FIFO (with selectable trigger levels) to monitor frame length and frame errors
  - Framing error, CRC error, illegal symbol (FIR), and abort pattern (SIR, MIR) detection

Table 17-2 lists the supported baud rates, the requested divisor, and the corresponding error versus the standard baud rate.

Table 17-2. UART IrDA Mode Baud Rates, Divisor Values, and Error Rates

Baud Rate	IR Mode	Encoding	Divisor	Error (%)
2400	SIR	3/16	1250	0
9600	SIR	3/16	312	0.16
19,200	SIR	3/16	156	0.16
38,400	SIR	3/16	78	0.16
57,600	SIR	3/16	52	0.16
115,200	SIR	3/16	26	0.16
576,000	MIR	1/4	2	0
1,152,000	MIR	1/4	1	0
4,000,000	FIR	4 PPM(1)	1	0

(1) PPM = pulse-position-modulation

17.1.3 CIR Features

The CIR mode uses a variable pulse-width modulation (PWM) technique (based on multiples of a programmable *t* period) to encompass the various formats of infrared encoding for remote-control applications. The CIR logic transmits data packets based on a user-definable frame structure and packet content.

The CIR (UART3 only) includes the following key features to provide CIR support for remote control applications:

- Transmit mode only (receive mode is not supported)
- Free data format (supports any remote-control private standards)
- Selectable bit rate
- Configurable carrier frequency
- 1/2, 5/12, 1/3, or 1/4 carrier duty cycle

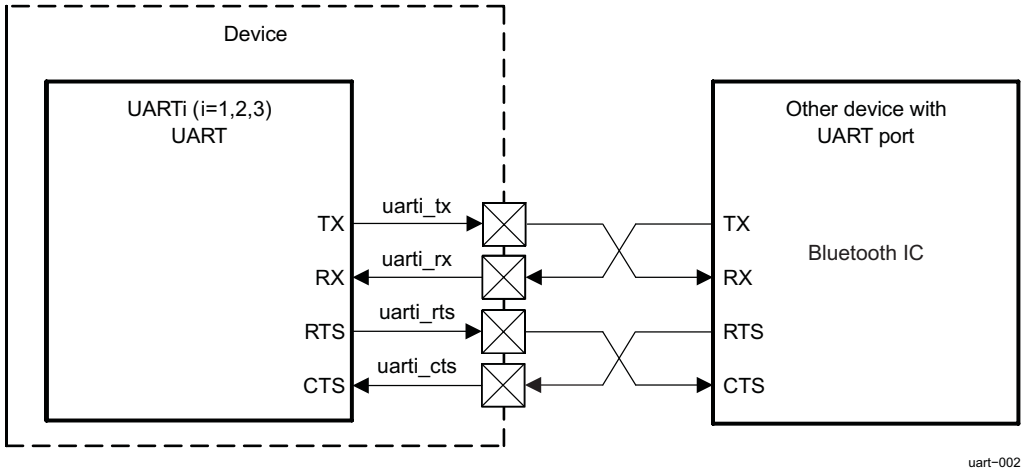
17.2 UART/IrDA/CIR Environment

This section describes the UART/IrDA/CIR connection with an external device.

17.2.1 System Using UART Communication with Hardware Handshake

UART1, UART2, or UART3 can be connected easily to the UART port of an external IC (see Figure 17-2).

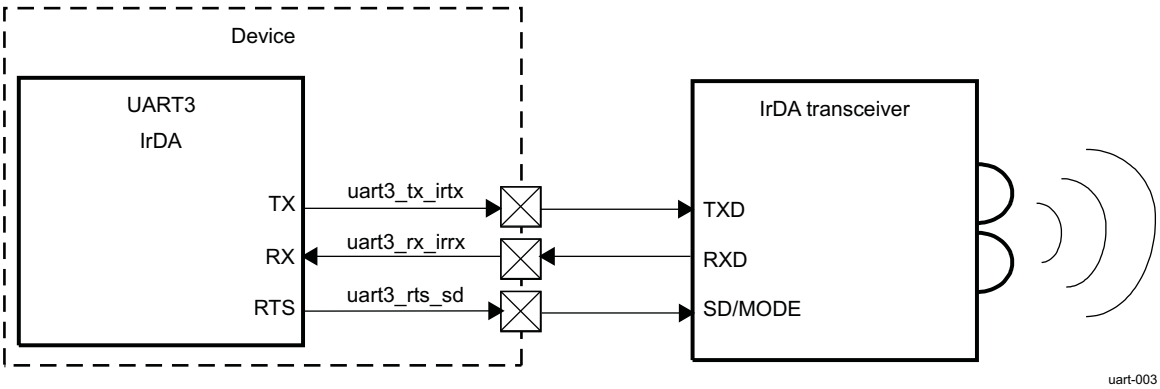
Figure 17-2. UART Mode Bus System Overview



17.2.2 System Using IrDA Communication Protocol

As Figure 17-3 shows, UART3 can be connected to an external infrared transceiver in the IrDA modes (FIR, SIR, and MIR).

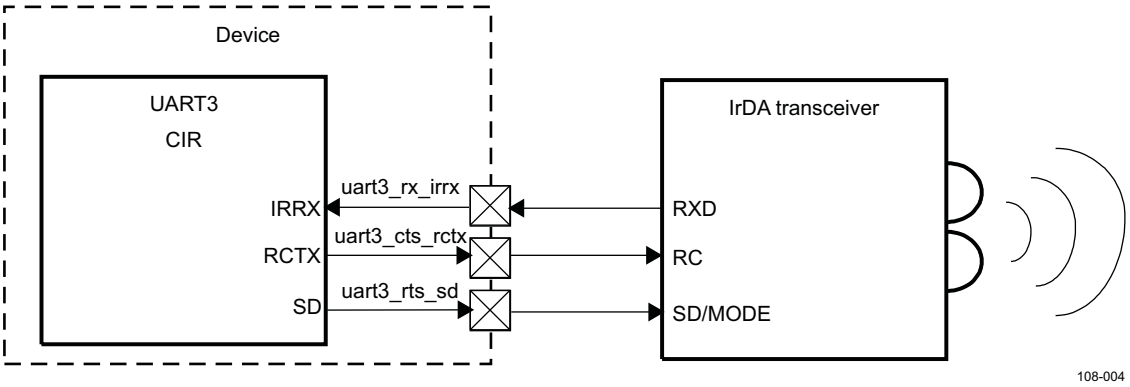
Figure 17-3. IrDA System Overview



17.2.3 System Using CIR Communication Protocol with Remote Control

UART3 can be connected to an external Infrared transceiver in CIR mode (see Figure 17-4).

Figure 17-4. CIR System Overview



17.2.4 UART Interface Description

17.2.4.1 UART Interface Description

Table 17-3 describes the UART interface.

Table 17-3. UART I/O Pin Description

Signal	I/O[1]	Description	Reset
UART Modem Signals			
uarti_rx	I	Serial data input	Unknown
uarti_tx	O	Serial data output Because this pin is active high in the IrDA mode and the output is muxed, this pin is set to low on reset (when UARTi.MDR1_REG[2:0] is set to 0x7) and takes the defined inactive level of that signal corresponding to when and how UARTi.MDR1_REG is programmed; that is, the output is 1 (inactive for UART modem modes) and 0 (inactive for IrDA modes).	0
uarti_cts	I	Clear to send Active-low modem status signal. Reading the UARTi.MSR_REG[4] NCTS_STS bit checks the condition of uarti_cts. Reading the UARTi.MSR_REG[0] CTS_STS bit checks a change of state of uarti_cts since the last read of the modem status register. The auto-nCTS mode uses uarti_cts to control the transmitter.	0
uarti_rts	O	Request to send When active (low), the module is ready to receive data. Setting the UARTi.MCR_REG[1] RTS bit activates uarti_rts, which becomes inactive as the result of a module reset, loopback mode, or clearing the UARTi.MCR_REG[1] RTS bit. In auto-RTS mode, uarti_rts becomes inactive as a result of the receiver threshold logic.	0

(1) I = Input, O = Output

17.2.4.2 UART Protocol and Data Format

The UART device operates in three modes:

- UART 16x mode (≤230.4K bits/s)
- UART 16x mode with autobauding (≥1200 bits/s and ≤115.2K bits/s)

- UART 13x mode (≥460.8K bits/s)

CAUTION

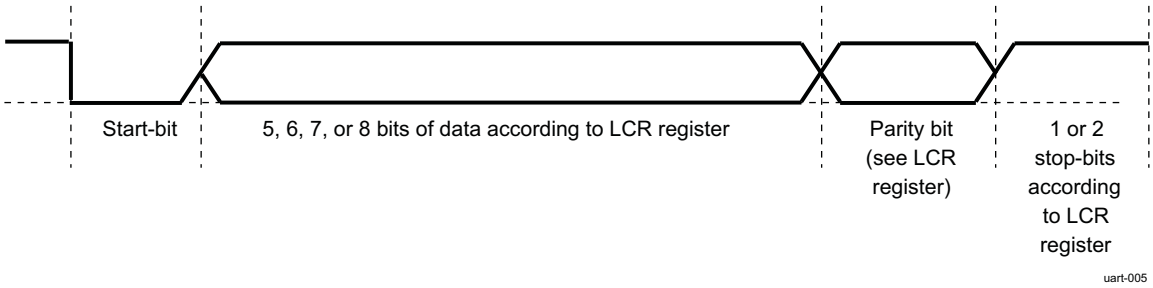
To be used as a UART, the operating mode must be programmed appropriately in the UARTi.MDR1_REG[2:0] MODE_SELECT field to select the UART, IrDA, or CIR mode.

The UART uses a wired interface for serial communication with a remote device.

The UART module is functionally compatible with the TL16C750 UART and earlier designs such as the TL16C550.

Figure 17-5 shows the UART frame data format.

Figure 17-5. UART Frame Data Format



17.2.5 IrDA Functional Interfaces

17.2.5.1 UART3 Interface Description

Table 17-4 describes the UART3 interface.

Table 17-4. UART3 I/O Description

Signal	I/O[1]	Description	Reset
IrDA Signals			
uart3_rx_irrx	I	Serial data input	Unknown
uart3_tx_irtx	O	Serial data output in IrDA modes (SIR, MIR, and FIR). In other modes, this pin is set to the reset value (inactive state).	0
uart3_rts_sd	O	SD mode is used to configure the transceivers. The SD pinout is an inverted value of the UART3.ACREG_REG[6] SD_MOD bit.	1

(1) I = Input, O = Output

17.2.5.2 IrDA Protocol and Data Format

17.2.5.2.1 SIR Mode

In SIR mode, data is transferred between the MPU and peripheral devices at speeds of up to 115,200 baud. A SIR transmit frame begins with start flags (a single 0xC0, a multiple 0xC0, or a single 0xC0 preceded by a number of 0xFF flags), is followed by frame data and a CRC-16, and ends with a stop flag (0xC1).

The bit format for a single word uses 1 start bit, 8 data bits, and 1 stop bit, and is unaffected by the use and settings of the UART3.LCR_REG register.

The UART3.BLR_REG[6] XBOF_TYPE bit selects whether the 0xC0 or 0xFF start patterns are used when multiple start flags are required.

The SIR transmit state-machine attaches start flags, CRC-16, and stop flags, and checks the outgoing data to establish whether data transparency is required.

SIR transparency is carried out if the outgoing data between the start and stop flags contains 0xC0, 0xC1, or 0x7D. If one of these start flags is about to be transmitted, the SIR state-machine first sends an escape character (0x7D), then inverts the fifth bit of the real data to be sent, and sends this data immediately after the 0x7D character.

The SIR receive state-machine recovers the receive clock, removes the start flags and any transparency from the incoming data, and determines the frame boundary with reception of the stop flag. The SIR state-machine also checks for errors, such as a frame abort (0x7D character followed immediately by a 0xC1 stop flag without transparency), a CRC error, and a frame-length error. At the end of a frame reception, the MPU reads the line status register (UART3.LSR_REG) to find possible errors of the received frame.

Note: Data can be transferred both ways by the module, but when the device is transmitting, the IR RX circuitry is automatically disabled by hardware. See the UART3.ACREG_REG[5] DIS_IR_RX bit description. This applies to all three modes: SIR, MIR, and FIR.

Infrared output in SIR mode can be either 1.6-us or 3/16th encoding, selected by the UART3.ACREG_REG[7] PULSE_TYPE bit. In 1.6-us encoding, the infrared pulse width is 1.6 µs; and in 3/16th encoding, the infrared pulse width is 3/16th of a bit duration (1/baud rate).

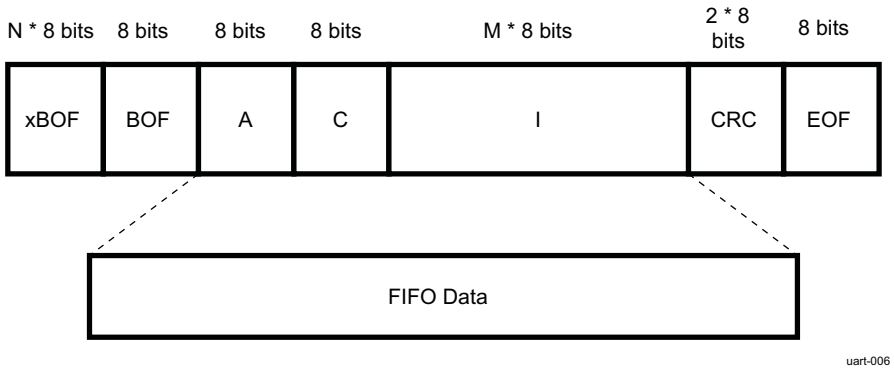
The transmitting device must send at least two start flags at the start of each frame for back-to-back frames.

Note: Reception supports variable-length stop-bits.

17.2.5.2.1.1 Frame Format

Figure 17-6 shows the IrDA SIR frame format.

Figure 17-6. IrDA SIR Frame Format



The CRC is applied on the address (A), control (C), and information (I) bytes.

Note: The two words of CRC are written to the FIFO in reception.

17.2.5.2.1.2 Asynchronous Transparency

Before transmitting a byte, the UART IrDA controller examines each byte of the payload and the CRC field (between BOF and EOF). For each byte equal to 0xC0 (BOF), 0xC1 (EOF), or 0x7D (control escape), the controller performs certain tasks:

- In transmission:
  - Inserts a control escape (CE) byte preceding the byte
  - Complements bit 5 of the byte (that is, exclusive ORs the byte with 0x20)

  The byte sent for the CRC computation is the initial byte written in the TX FIFO (before the XOR with 0x20).
- In reception:

  For the A, C, I, CRC field:
  - Compares the byte with the CE byte; if they are not equal, sends the byte to the CRC detector and stores it in the RX FIFO
  - If the byte is equal to the CE byte, discards the CE byte
  - Complements bit 5 of the byte following the CE
  - Sends the complemented byte to the CRC detector and stores it in the RX FIFO

17.2.5.2.1.3 Abort Sequence

The transmitter can decide to prematurely close a frame. The transmitter aborts by sending the following sequence: 0x7DC1. The abort pattern closes the frame without a CRC field or an ending flag.

The receiver treats a frame as an aborted frame when a 0x7D character followed immediately by a 0xC1 character is received without transparency.

17.2.5.2.1.4 Pulse Shaping

The SIR mode supports both the 3/16th and the 1.6-µs pulse duration methods. The UART3.ACREG_REG[7] PULSE_TYPE bit selects the pulse width method in transmit mode.

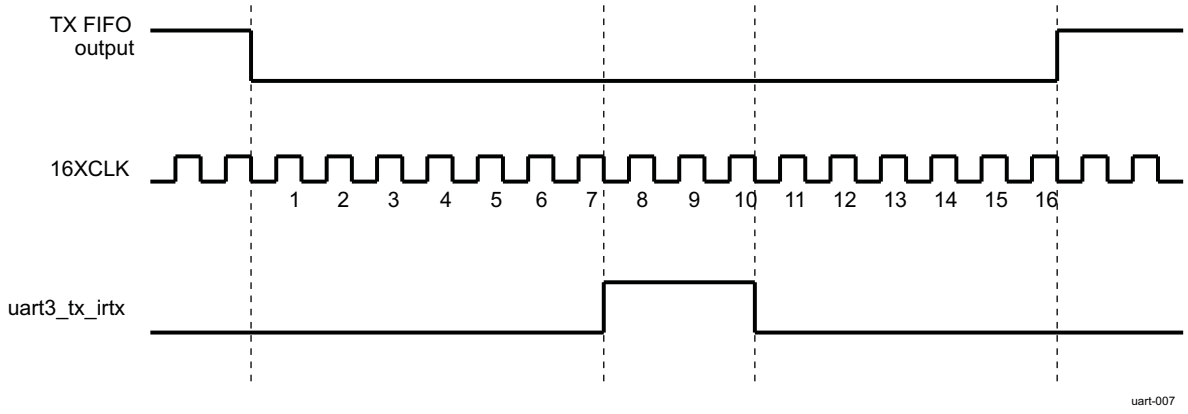
17.2.5.2.1.5 Encoder

Serial data from the transmit state-machine is encoded to transmit data to the optoelectronics. While the TX FIFO output is high, the uart3_tx_irtx line is always low, and the counter used to form a pulse on uart3_tx_irtx is cleared continuously.

After the TX FIFO output resets to 0, uart3_tx_irtx rises on the falling edge of the 7th 16XCLK. On the falling edge of the 10th 16XCLK pulse, uart3_tx_irtx falls, creating a three-clock-wide pulse. While the TX FIFO output stays low, a pulse is transmitted during the 7th to the 10th clock of each 16-clock bit cycle.

Figure 17-7 shows the IrDA SIR encoding mechanism.

Figure 17-7. IrDA SIR Encoding Mechanism

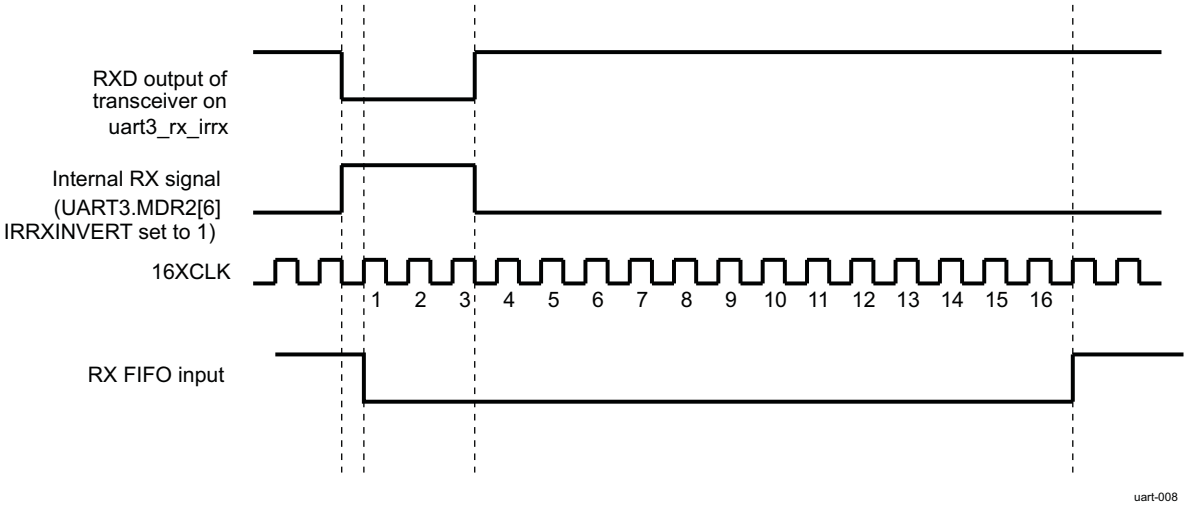


17.2.5.2.1.6 Decoder

After reset, the RX FIFO input is high and the 4-bit counter is cleared. When a rising edge is detected on RX, the RX FIFO input falls on the next rising edge of 16XCLK with sufficient setup time. The RX FIFO input stays low for 16 cycles (16XCLK) and then returns to high as required by the IrDA specification. As long as no pulses (rising edges) are detected on the RX, the RX FIFO input remains high.

Figure 17-8 shows the IrDA SIR decoding mechanism.

Figure 17-8. IrDA SIR Decoding Mechanism



Data can be transferred both ways by the module, but when the device is transmitting, the IR RX circuitry is automatically disabled by hardware. The operation of the uart3_rx_irrx input can be disabled using the UART3.ACREG_REG[5] DIS_IR_RX bit. Furthermore, the UART3.MDR2_REG[6] IRRXINVERT bit can invert the signal from the transceiver (RXD) pin to the IRRX logic inside the UART. This inversion is performed by default.

17.2.5.2.1.7 IR Address Checking

In all IR modes, when address checking is enabled by setting EFR_REG[1:0] (see Table 17-5), only frames intended for the device are written to the RX FIFO. This is to avoid receiving frames not meant for this device in a multipoint infrared environment. To program two frame addresses that the UART3 receives in IrDA mode, use the UART3.XON1_ADDR1_REG[7:0] field and the UART3.XON2_ADDR2_REG[7:0] field.

Table 17-5. EFR_REG[0-1] IR Address Checking Options

EFR_REG[1]	EFR_REG[0]	IR Address Checking
0	0	All address-checking operations disabled
0	1	Only address 1 checking enabled
1	0	Only address 2 checking enabled
1	1	All address-checking operations enabled

17.2.5.2.2 SIR Free Format Mode

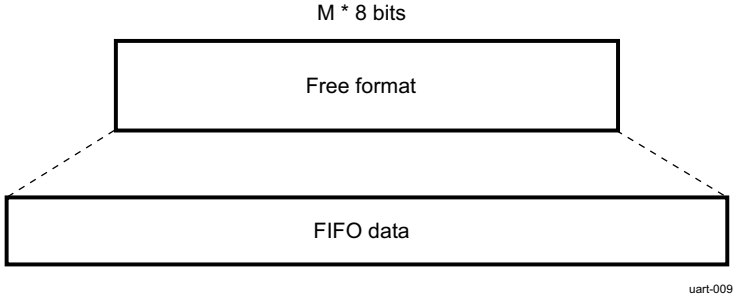
To allow complete software flexibility when transmitting and receiving infrared data packets, the SIR free format (FF) mode is a subfunction of the existing SIR mode; all frames going to and from the FIFO buffers are untouched with respect to appending and removing control characters and CRC values.

This mode corresponds to a UART mode with a pulse modulation of 3/16 of baud rate pulse width.

For example, a normal SIR packet has BOF control and CRC error-checking data appended (transmitting) or removed (receiving) from the data going to and from the FIFOs.

Figure 17-9 shows the SIR free format mode.

Figure 17-9. SIR Free Format Mode

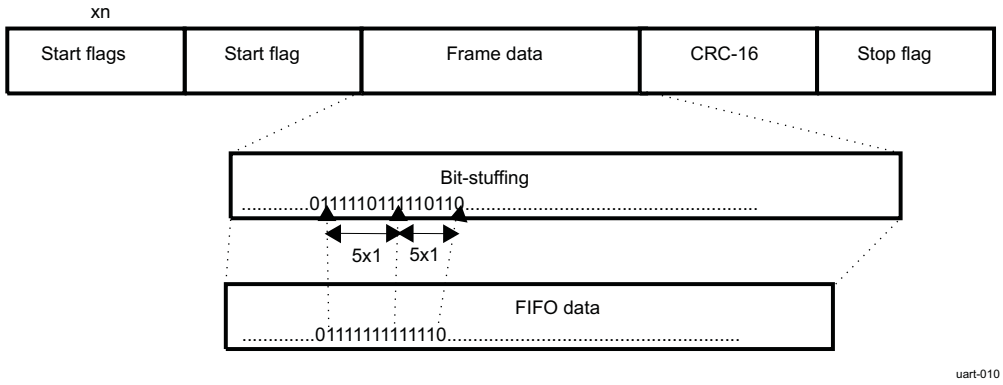


In the SIR free format mode, the MPU software must construct (that is, encode and decode) the entire FIFO data packet.

17.2.5.2.3 MIR Mode

In MIR mode, data is transferred between the MPU and the peripheral devices at 0.576 or 1.152M bits/s speed. A MIR transmit frame starts with start flags (at least two), followed by a frame data, CRC-16, and ends with a stop flag (see Figure 17-10).

Figure 17-10. MIR Transmit Frame Format



On transmit, the MIR state-machine attaches start flags, CRC-16, and stop flags, as in SIR mode. All fields are transmitted least-significant bit (LSB) of each byte first.

Contrary to SIR mode:

- The state-machine looks for consecutive 1s in the frame data and automatically inserts 0 after five consecutive 1s (this is called bit-stuffing).
- 0x7E is used for both start and stop flags (unambiguously, not data, because of bit-stuffing).
- Abort sequence requires a minimum of seven consecutive 1s (unambiguously, not data, because of bit stuffing).
- Back-to-back frames are allowed with three or more stop flags in between. If two consecutive frames are not back to back, the gap between the last stop flag of the first frame and the start flag of the second frame must be separated by at least seven bit durations.

On receive, the MIR receive state-machine recovers the receive clock, removes the start flags, destuffs the incoming data, and determines the frame boundary with reception of the stop flag. The state-machine also checks for errors, such as frame abort, CRC error, and frame-length error. At the end of a frame reception, the MPU reads the line status register (UART3.LSR_REG) to detect possible errors of the received frame.

Data can be transferred both ways by the module, but when the device is transmitting, the IR RX circuitry is automatically disabled by hardware.

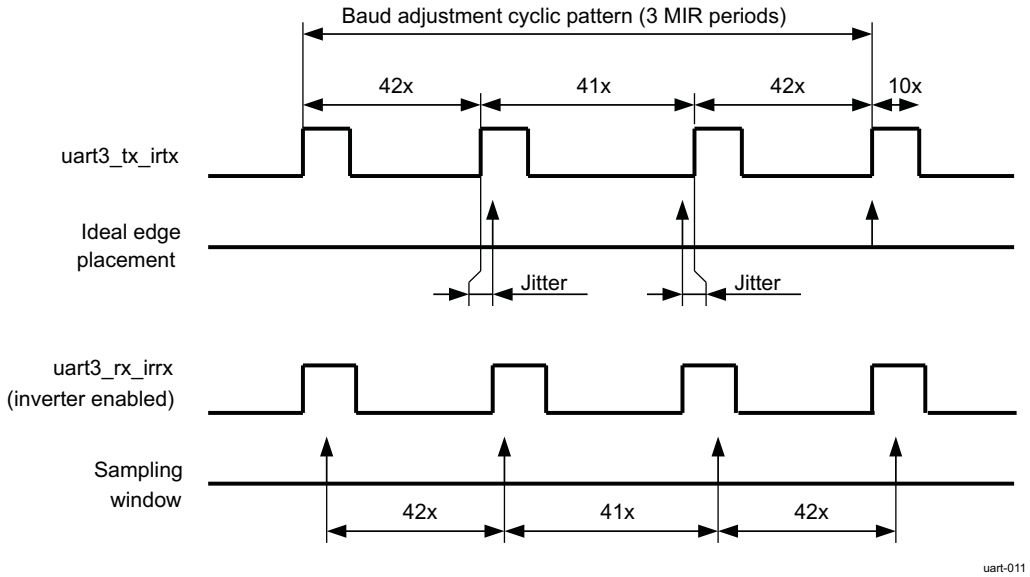
17.2.5.2.3.1 MIR Encoder/Decoder

To meet the MIR baud rate tolerance of +/–0.1 percent with a 48-MHz clock input, a 42-41-42 encoding/decoding adjustment is performed. The reference start point is the first start flag, and the 42-41-42 cyclic pattern is repeated until the stop flag is sent or detected.

The jitter created this way is within MIR tolerances. The pulse width is not exactly 1/4, but it is within the tolerances defined by the IrDA specifications.

Figure 17-11 shows the MIR baud rate adjustment mechanism.

Figure 17-11. MIR Baud Rate Adjustment Mechanism

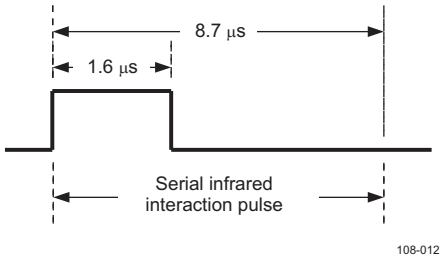


17.2.5.2.3.2 SIP Generation

In the MIR and FIR operation modes, the transmitter must send a serial infrared interaction pulse (SIP) at least once every 500 ms. The SIP informs slow devices (operating in SIR mode) that the medium is currently occupied.

Figure 17-12 shows the SIP.

Figure 17-12. SIP



17.2.5.2.4 FIR Mode

In FIR mode, data is transferred between the MPU and the peripheral devices at 4M bits/s. A FIR transmit frame starts with a preamble that is followed by a start flag, frame data, CRC-32, and ends with a stop flag.

Table 17-6 shows the FIR transmit frame format.

Table 17-6. FIR Transmit Frame Format

Preamble (16x)	Start flag	Frame data	CRC-32	Stop flag

On transmit, the FIR transmit state-machine attaches the preamble, start flag, CRC-32, and stop flag. An abort sequence requires at least two transmissions of 0000. Back-to-back frames are allowed, but each frame must be complete.

The state-machine also encodes the transmit data into 4-PPM format (see Table 17-7) and generates the SIP (see Section 17.2.5.2.3.2, *SIP Generation*).

Table 17-7. 4-PPM format

Data Bit Pair (Bin)	4-PPM Data Symbol (Bin)
00	1000
01	0100
10	0010
11	0001

The four symbols described in Table 17-7 are the legal encoded data symbols. All other combinations are illegal for encoding data. Some of these illegal symbols are used in the definition of the preamble, start flag, and stop flag because they are unambiguously not data (see Table 17-8).

Table 17-8. FIR Preamble, Start Flag, and Stop Flag

Frame Part	Transmitted Frame (Bin)
Preamble	1000 0000 1010 1000 (16 repeated transmissions)
Start Flag	0000 1100 0000 1100 0110 0000 0110 0000
Stop Flag	0000 1100 0000 1100 0000 0110 0000 0110

All fields are transmitted the LSBs of each byte first (see Table 17-9).

Table 17-9. FIR Data Byte Transmission Order Example

Data Byte (Hex)	Data Byte Pair (Bin)	4-PPM Data Symbol (Bin)	Transmission Order
0x0B	00	1000	4
	00	1000	3
	10	0010	2
	11	0001	1

On receive, the FIR receive state-machine recovers the receive clock, removes the preamble and the start flag, decodes the 4-PPM incoming data, and determines the frame boundary with reception of the stop flag. The state-machine also checks for errors, such as illegal symbol, CRC error, and frame-length error. At the end of a frame reception, the MPU reads the line status register (UART3.LSR_REG) to detect possible errors of the received frame.

Data can be transferred both ways by the module, but when the device is transmitting, the IR RX circuitry is automatically disabled by hardware.

17.2.6 CIR Functional Interfaces

17.2.6.1 CIR Interface Description

Table 17-10 describes the CIR interface.

Table 17-10. CIR I/O Description

Signal	I/O[1]	Description	Reset
CIR Signals			
uart3_rx_irrx	I	Serial data input	Unknown
uart3_cts_rctx	O	Serial data output in CIR mode. In other modes, this pin is set to the reset value (inactive state).	0
uart3_rts_sd	O	SD mode is used to configure the transceivers. The SD pinout is an inverted value of the UART3.ACREG_REG[6] SD_MOD bit.	1

(1) I = Input, O = Output

17.2.6.2 CIR Protocol and Data Format

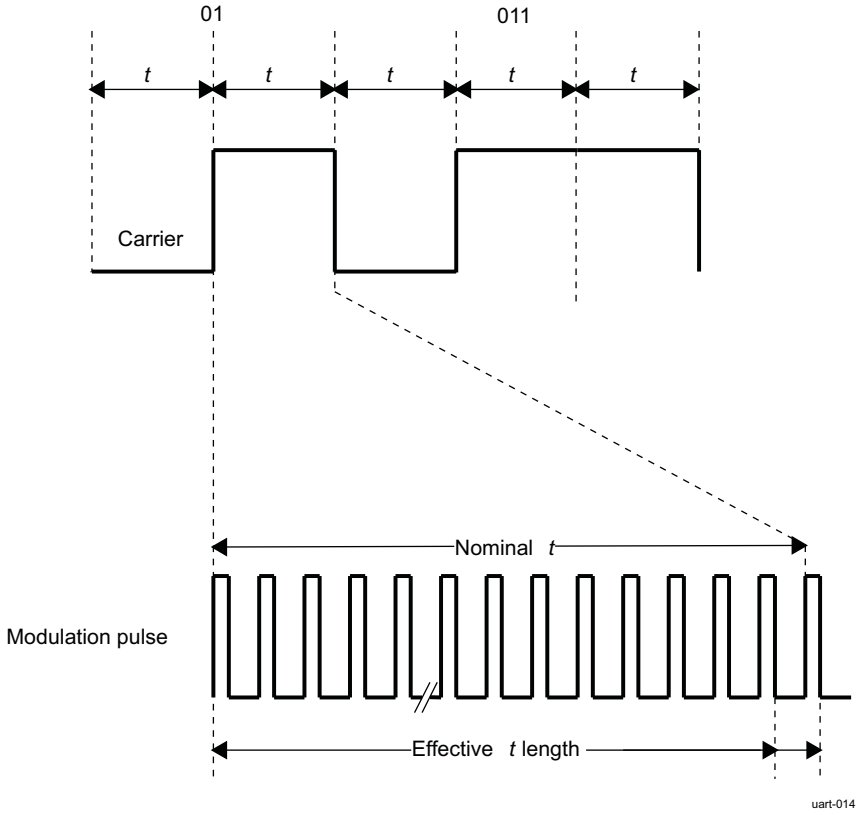
In the CIR mode, the infrared operation functions as a programmable (universal) remote control.

The CIR mode uses a variable PWM technique (based on multiples of a programmable *t* period) to encompass the various formats of infrared encoding for remote control applications. The CIR logic transmits data packets based on the user-defined frame structure and packet content.

17.2.6.2.1 Carrier Modulation

Each modulated pulse that constitutes a digit is a train of on/off pulses (see Figure 17-13).

Figure 17-13. CIR Pulse Modulation

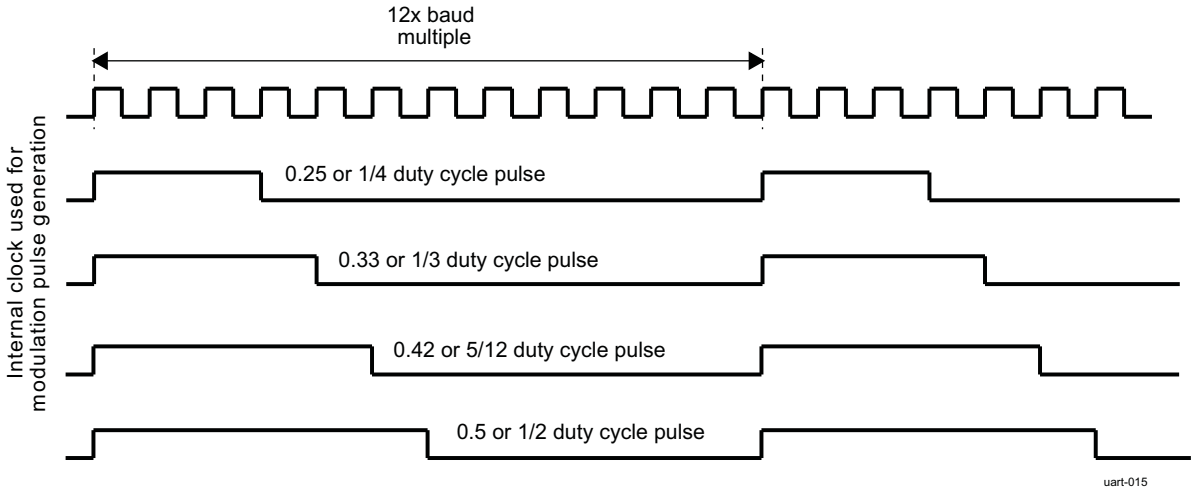


17.2.6.2.2 Pulse Duty Cycle

The programmer can choose between four possible duty cycles for modulation pulses by setting the appropriate value in the UART3.MDR2_REG[5:4] CIR_PULSE_MODE field (1/4, 1/3, 5/12, or 1/2).

Figure 17-14 shows the CIR modulation duty cycles.

Figure 17-14. CIR Modulation Duty Cycle



The transmission logic ensures that all pulses are transmitted completely (that is, no cutoff during transmission). Furthermore, while transmitting continuous bytes back-to-back, no delay is inserted between 2 transmitted bytes. Thus, software must handle the delay between consecutively transmitted bytes if the receiving end needs it.

17.2.6.2.3 Consumer IR Encoding/Decoding

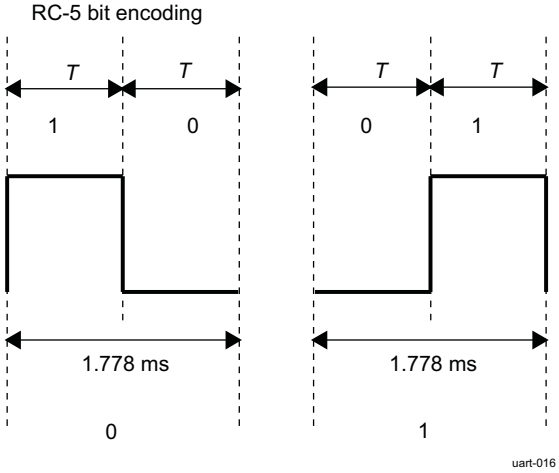
There are two methods of encoding for remote control applications. The first method uses time-extended bit forms (a variable pulse distance, or duration, in which the difference between a logic 1 and logic 0 is the length of the pulse width).

The second encoding method uses a biphase in which the encoding of the logic 0 and logic 1 is in the change of signal level from 1 to 0 or 0 to 1, respectively. Japanese manufacturers favor pulse duration encoding; European manufacturers favor biphase encoding.

The CIR mode uses a completely flexible free-format encoding in which the TX FIFO is transmitted as a modulated pulse with duration T.

Similarly, 0 is transmitted as a blank duration T. The MPU constructs and deciphers the protocol of the data. For example, the RC-5 protocol using Manchester encoding can be emulated as using a 01 pair for 1 and a 10 pair for 0 (see Figure 17-15.).

Figure 17-15. RC-5 Bit Encoding

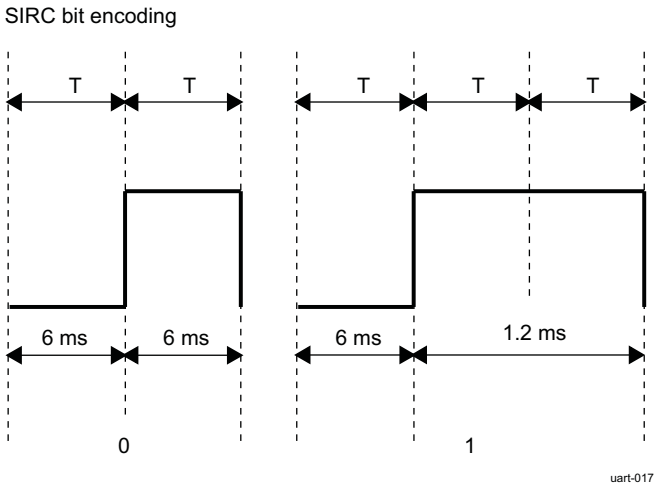


Because the CIR mode logic does not impose a fixed format for infrared packets of data, the MPU software can define the format using simple data structures that are then modulated into an industry standard, such as RC-5 or SIRC. To send a sequence of 0101 in RC-5, the MPU software must write an 8-bit binary character of 10011001 to the data FIFO of the UART.

For SIRC, the modulation length (that is, multiples of T) is the method used to distinguish between 1 and 0. The following SIRC digits show the difference in encoding between this and, for example, RC-5. The pulse width is extended for one digit.

Figure 17-16 shows SIRC bit encoding.

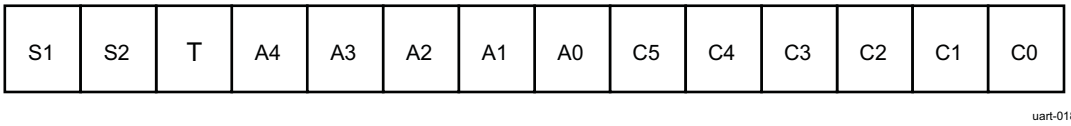
Figure 17-16. SIRC Bit Encoding



To construct comprehensive packets constituting remote-control commands, the MPU software must combine a number of 8-bit data characters in a sequence that follows one of the universally accepted formats.

Figure 17-17 shows a standard RC-5 frame as detected by the UART3 in CIR mode (the SIRC format follows this). Each field in RC-5 can be considered as two T pulses (digital bits) from the TX and RX FIFOs.

Figure 17-17. RC-5 Standard Packet Format



Where:

S1, S2:	Start bits (always 1)
T:	Toggle bit
A4..A0:	Address (or system) bits
C5..C0:	Command bits

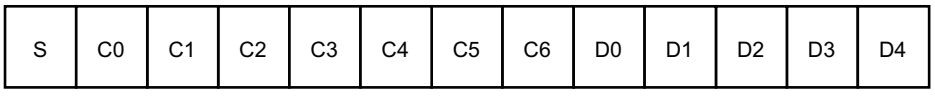
The toggle bit T changes when a new command is transmitted to detect when the same key is pressed twice (effectively receiving the same data from the host consecutively). A brief delay in the transmission of the same command is detected by the use of the toggle bit because a code is sent while the MPU transmits characters to the UART for transmission. The address bits define the machine or device for which the infrared transmission is intended, and the command defines the operation.

To accommodate an extended RC-5 format, the S2 bit is replaced by an additional command bit (C6) that allows the command range to increase to 7 bits. This format is known as the extended RC-5 format.

SIRC encoding uses the duration of modulation for mark and space; therefore, the duration of data bits inside the standard frame length varies.

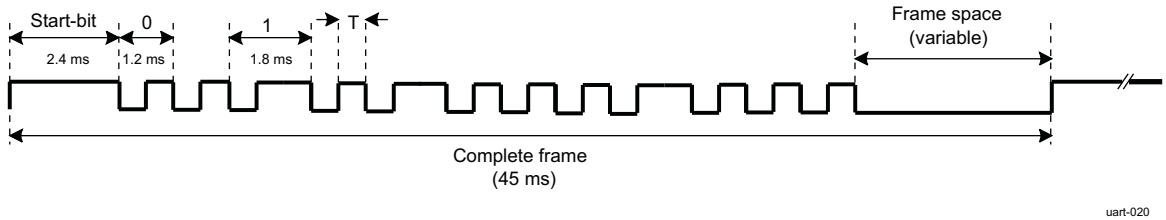
Figure 17-18 shows the packet format and bit encoding. As Figure 17-19 shows, 1 start bit of 2 ms and control codes are followed by data that constitute the entire frame.

Figure 17-18. SIRC Packet Format



Note: The encoding must take a standard duration, but the contents of the data can vary. This implies that the control software for emitting and receiving data packets must exercise a scheme of interpacket delay, where the emission of successive packets can be done only after a real-time delay has expired.

Figure 17-19. SIRC Bit Transmission Example

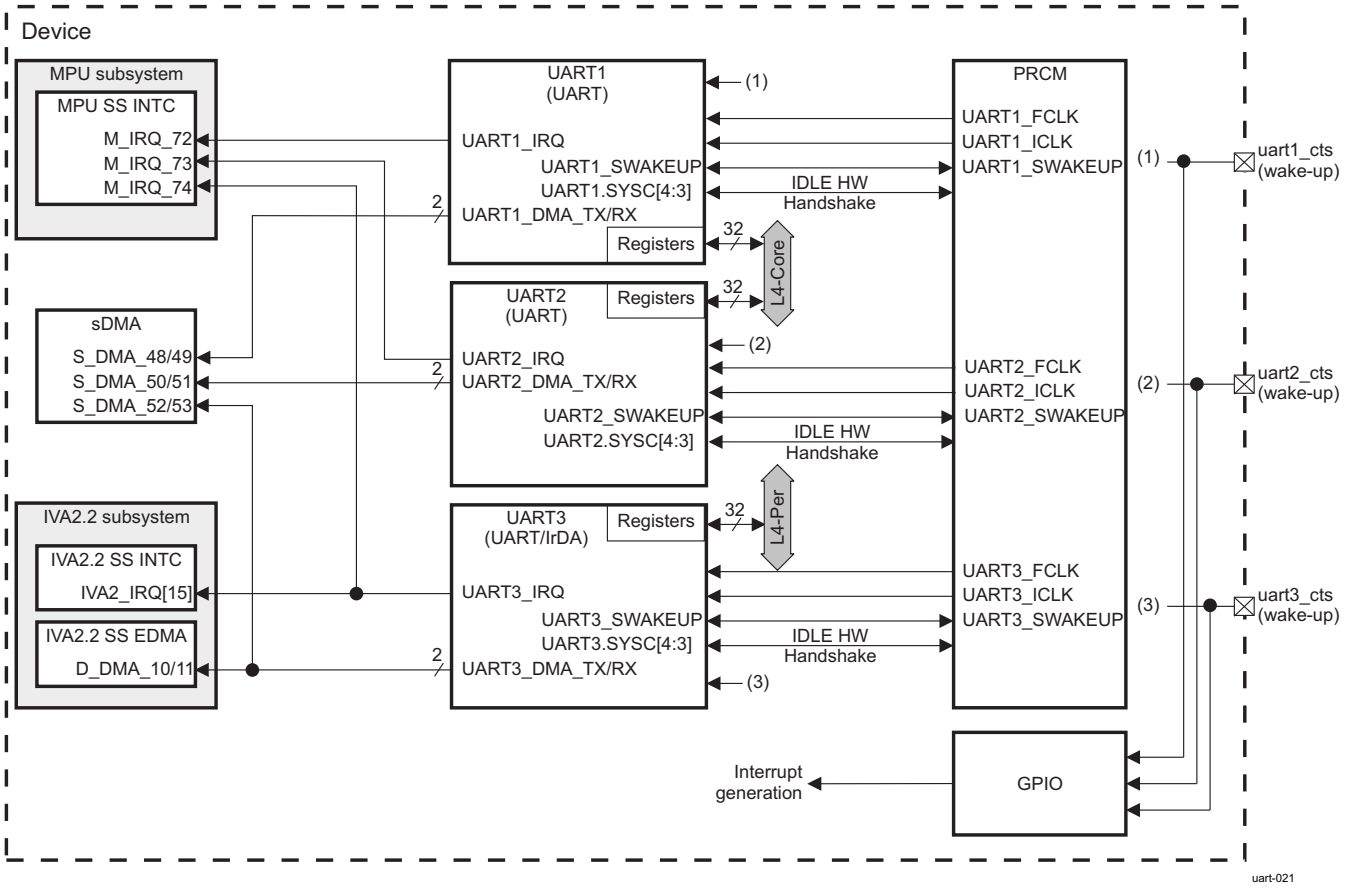


This document does not describe all encoding methods and techniques; the preceding information shows the considerations required to employ different encoding methods for different industry-standard protocols. See industry-standard documentation for specific methods of encoding and protocol usage.

17.3 UART/IrDA/CIR Integration

Figure 17-20 shows the OMAP34xx internal connections with related modules for UART functions.

Figure 17-20. UART Functional Integration



17.3.1 Clocking, Reset, and Power-Management Scheme

17.3.1.1 Clocking

Each UART uses a 48-MHz functional clock for its logic and the generation of external interface signals. Each UART uses an interface clock for register accesses. The power, reset, and clock management (PRCM) module generates and controls all these clocks (for more information, see Chapter 4, *Power, Reset, and Clock Management*).

Table 17-11 describes the UART clocks.

Table 17-11. UART Clocks

Clock Type	Frequency	Name	Comments
UART1			
Functional clock	CORE_48M_FCLK	UART1_FCLK	Source and gating is the PRCM module. Enable/disable this clock with the PRCM.CM_FCLKEN1_CORE[13] EN_UART1 bit.
Interface clock	CORE_L4_ICLK	UART1_ICLK	Source and gating is the PRCM module. Enable/disable this clock with the PRCM.CM_ICLKEN1_CORE[13] EN_UART1 bit. Enable/disable auto-idle for this clock with the PRCM.CM_AUTOIDLE1_CORE[13] AUTO_UART1 bit.

Table 17-11. UART Clocks (continued)

Clock Type	Frequency	Name	Comments
UART2			
Functional clock	CORE_48M_FCLK	UART2_FCLK	Source and gating is the PRCM module. Enable/disable this clock with the PRCM.CM_FCLKEN1_CORE[14] EN_UART2 bit.
Interface clock	CORE_L4_ICLK	UART2_ICLK	Source and gating is the PRCM module. Enable/disable this clock with the PRCM. CM_ICLKEN1_CORE[14] EN_UART2 bit. Enable/disable auto-idle for this clock with the PRCM.CM_AUTOIDLE1_ CORE[14] AUTO_UART2 bit.
UART3			
Functional clock	PER_48M_FCLK	UART3_FCLK	Source and gating is the PRCM module. Enable/disable this clock with the PRCM.CM_FCLKEN_PER[11] EN_UART3 bit.
Interface clock	PER_L4_ICLK	UART3_ICLK	Source and gating is the PRCM module. Enable/disable this clock with the PRCM. CM_ICLKEN_PER[11] EN_UART3 bit. Enable/disable auto-idle for this clock with the PRCM.CM_AUTOIDLE_ PER[11] AUTO_UART3 bit.

The idle and wake-up processes use a handshake protocol between the PRCM module and the UARTs (for a description of the protocol, see Chapter 4, *Power, Reset, and Clock Management*). The UARTi.SYSC_REG[4:3] IDLEMODE field controls the UART idle mode.

17.3.1.2 Hardware Reset

Table 17-12 lists the UART reset domain.

Table 17-12. Reset Domain

Peripherals	Reset Domain
UART1	CORE_RST
UART2	CORE_RST
UART3	PER_RST

17.3.1.3 Software Reset

The UARTi.SYSC_REG[1] SOFTRESET bit controls the software reset; writing 1 to this bit triggers a software-reset functionally equivalent to hardware reset.

17.3.1.4 Power Domain

Each UART in the CORE or PER power domain can dynamically switch between available voltage levels (see Table 17-13).

Table 17-13. Power Domain

Peripherals	Power Domain
UART1	CORE
UART2	CORE
UART3	PER

17.3.2 Hardware Requests

17.3.2.1 Interrupts

Table 17-14 describes the UART interrupt mappings.

Table 17-14. Interrupt Mapping to MPU Subsystem

IRQ	Source	Description
M_IRQ_72	UART1_IRQ	UART module 1 interrupt to MPU
M_IRQ_73	UART2_IRQ	UART module 2 interrupt to MPU
M_IRQ_74	UART3_IRQ	UART module 3 interrupt to MPU

Table 17-15. Interrupt Mapping to IVA2.2 Subsystem

IRQ	SOURCE	DESCRIPTION
IVA2_IRQ[15]	UART3_IRQ	UART module 3 interrupt to IVA2.2 subsystem

17.3.2.2 DMA Requests

Table 17-16 describes the UART DMA requests.

Table 17-16. UART DMA Requests to System DMA

DMA[(1)]	Source	Description
S_DMA_48	UART1_DMA_TX	UART module 1 transmit request
S_DMA_49	UART1_DMA_RX	UART module 1 receive request
S_DMA_50	UART2_DMA_TX	UART module 2 transmit request
S_DMA_51	UART2_DMA_RX	UART module 2 receive request
S_DMA_52	UART3_DMA_TX	UART module 3 transmit request
S_DMA_53	UART3_DMA_RX	UART module 3 receive request

(1) This table assumes that DMA mode 1 is used.

Table 17-17. UART DMA Requests to IVA2.2 Subsystem DMA

DMA	SOURCE	DESCRIPTION
D_DMA_10	UART3_DMA_TX	UART module 3 transmit request
D_DMA_11	UART3_DMA_RX	UART module 3 receive request

17.3.2.3 Wake-up Request

The UART can wake up the system on different events (see Section 17.6, *UART/IrDA/CIR Register Manual*).

One important wake-up generation is event detection on the uarti_cts pin when the system is idle. The uarti_cts pin uses an asynchronous path to transmit the wake-up request to the system (PRCM), which enables wake-up capabilities to be active even if all module clocks are off (see Table 17-18).

Table 17-18. Wake-Up Requests from PRCM

Wake-Up Pin	PRCM Input	Description
uart1_cts	UART1_SWAKEUP	Wake UART1 (system wake-up capabilities)

Table 17-18. Wake-Up Requests from PRCM (continued)

Wake-Up Pin	PRCM Input	Description
uart2_cts	UART2_SWAKEUP	Wake UART2 (system wake-up capabilities)
uart3_cts	UART3_SWAKEUP	Wake UART3 (system wake-up capabilities)

CAUTION

UARTs are not in the WAKEUP power domain, which implies limitations on wake-up capability. If the CORE power domain is off, UART1 and UART2 cannot wake up the system, because power is not supplied. If the PER power domain is off, UART3 cannot wake up the system, because power is not supplied. In these cases, a GPIO multiplexed on the uarti_cts pin can be used to capture the event and wake up the system. Software must enable this strategy, if required. See Chapter 25, *General-Purpose Interface*, for a full description of this mechanism.

17.4 UART/IrDA/CIR Functional Description

17.4.1 UART/IrDA/CIR Block Description

The UART/IrDA/CIR module can be divided into three main blocks:

- FIFO management
- Mode selection
- Protocol formatting

FIFO management is common to all functions and enables the transmission and reception of data from the host processor point of view.

There are two mode selections:

- Function mode selection: Route the data to the chosen functionality (UART, IrDA, or CIR) and enable the mechanism corresponding to the chosen functionality.
- Register mode selection, which enables conditional access to registers

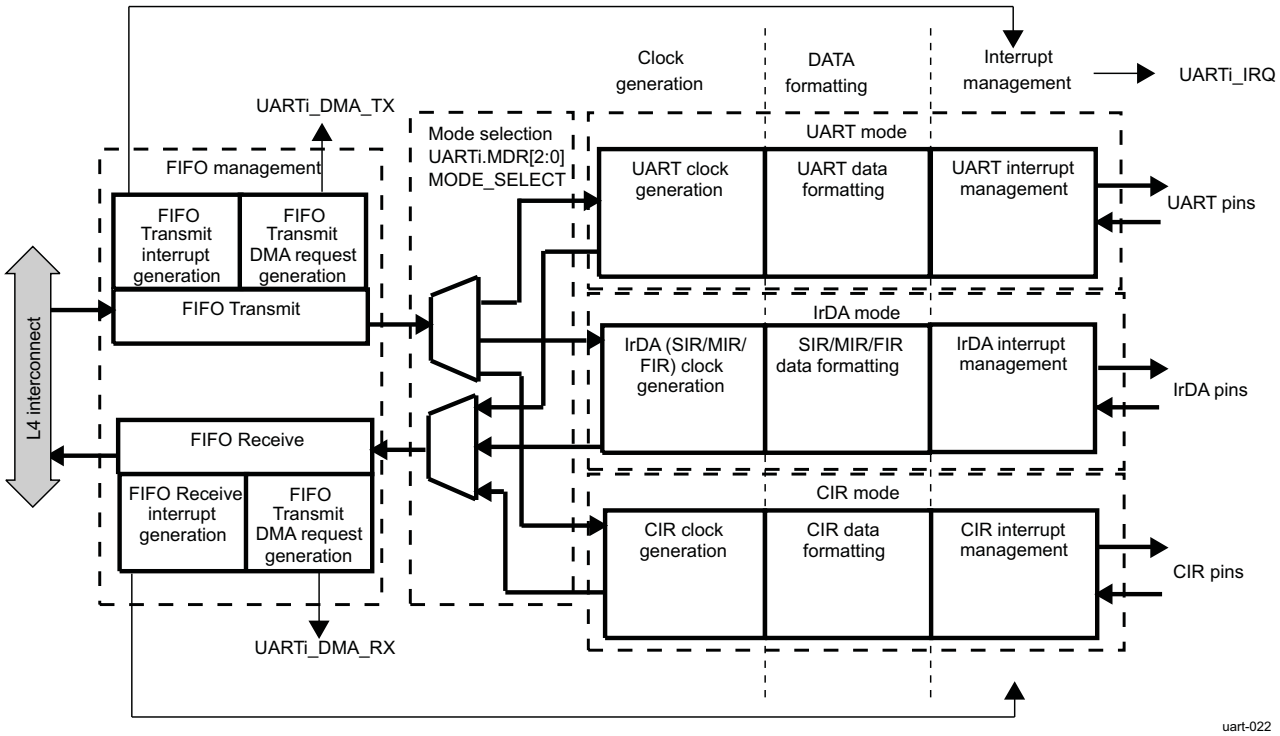
Protocol formatting has three subcategories:

- Clock generation: The 48-MHz input clock generates all necessary clocks.
- Data formatting: Each function uses its own state-machine, which assumes the transition between FIFO data and frame data.
- Interrupt management: Different interrupt types are generated depending on the chosen function.

In parallel with these functional blocks, a power-saving strategy exists for each function.

Figure 17-21 is the UART/IrDA/CIR block diagram.

Figure 17-21. UART/IrDA/CIR Block Diagram



17.4.2 FIFO Management

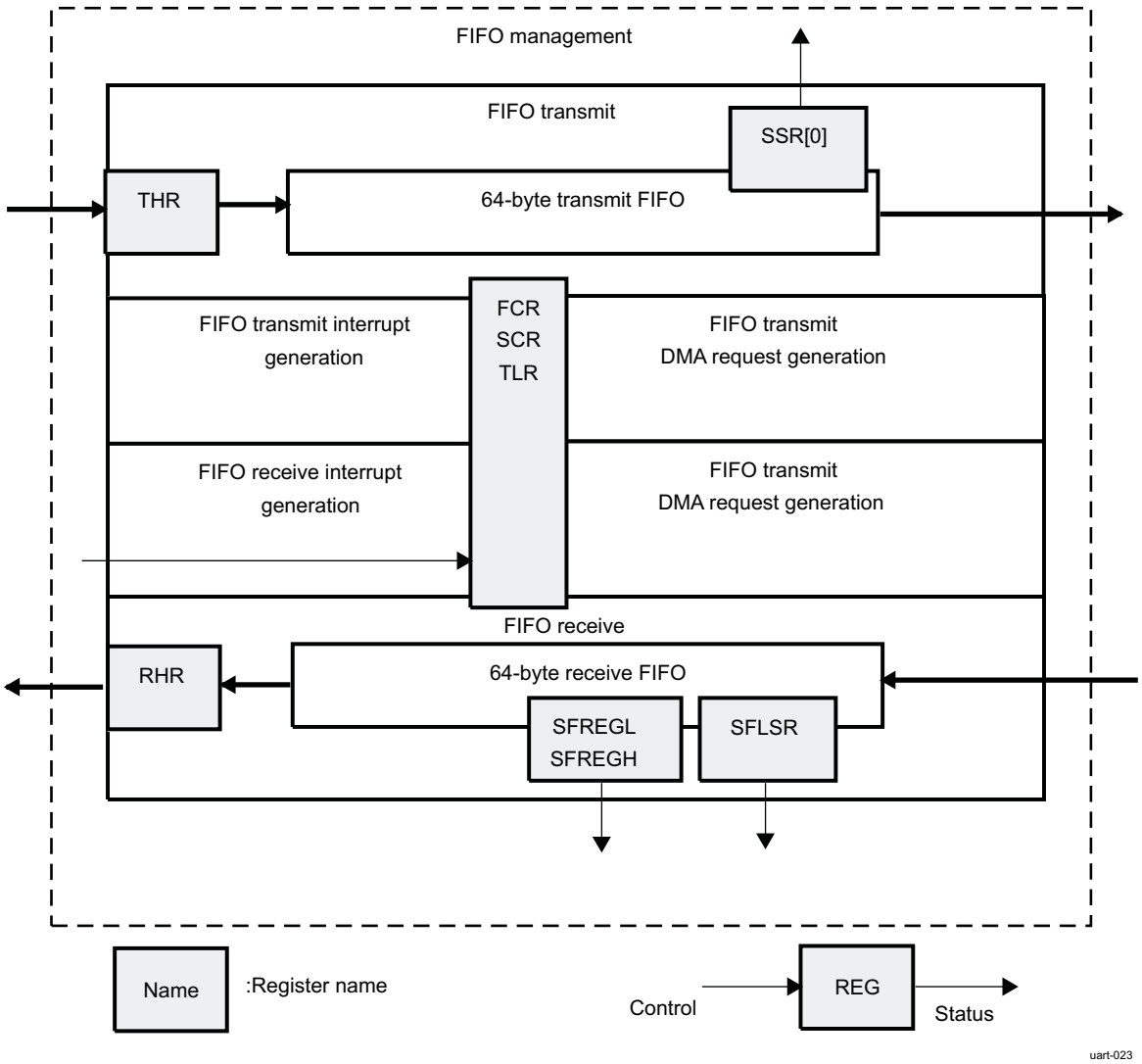
The FIFO is accessed by reading/writing the UARTi.RHR_REG and UARTi.THR_REG registers. Parameters are controlled using the FIFO control register (UARTi.FCR_REG) and supplementary control register (UARTi.SCR_REG). Reading the UARTi.SSR_REG[0] TX_FIFO_FULL bit at 1 means the FIFO is full.

The UARTi.TLR_REG register controls the FIFO trigger level, which enables the DMA and interrupt generation. After reset, both transmitter and receiver FIFOs are disabled; so, in effect, the trigger level is the default value of 1 byte. Figure 17-22 shows the FIFO management registers.

Note: Data in the UARTi.RHR_REG register is not overwritten when an overflow occurs.

Note: The UARTi.SFLSR_REG, UARTi.SFREGL_REG, and UARTi.SFREGH_REG status registers are used in IrDA mode only. For use, see Section 17.4.4.2.3, *IrDA Data Formatting*.

Figure 17-22. FIFO Management Registers



17.4.2.1 FIFO Trigger

17.4.2.1.1 Transmit FIFO Trigger

Table 17-19 summarizes the transmit FIFO trigger level settings.

Table 17-19. TX FIFO Trigger Level Setting Summary

SCR_REG[6]	TLR_REG[3:0]	TX FIFO Trigger Level
0	= 0x0	Defined by the UARTi.FCR_REG[5:4] TX_FIFO_TRIG field (8,16, 32, or 56 spaces)
0	≠ 0x0	Defined by the UARTi.TLR_REG[3:0] TX_FIFO_TRIG_DMA field (from 4 to 60 spaces with a granularity of 4 spaces)
1	Value	Defined by the concatenated value of TX_FIFO_TRIG_DMA and TX_FIFO_TRIG (from 1 to 63 spaces with a granularity of 1 space) **Note:** The combination of TX_FIFO_TRIG_DMA = 0x0 and TX_FIFO_TRIG = 0x0 (all zeros) is not supported (minimum of one space required). All zeros result in unpredictable behavior.

17.4.2.1.2 Receive FIFO Trigger

Table 17-20 summarizes the receive FIFO trigger level settings.

Table 17-20. RX FIFO Trigger Level Setting Summary

SCR_REG[7]	TLR_REG[7:4]	RX FIFO Trigger Level
0	= 0x0	Defined by the UARTi.FCR_REG[7:6] RX_FIFO_TRIG field (8,16, 56, or 60 characters)
0	≠ 0x0	Defined by UARTi.TLR_REG[7:4] RX_FIFO_TRIG_DMA field (from 4 to 60 characters with a granularity of 4 characters)
1	Value	Defined by the concatenated value of RX_FIFO_TRIG_DMA and RX_FIFO_TRIG (from 1 to 63 characters with a granularity of 1 character). **Note:** The combination of RX_FIFO_TRIG_DMA = 0x0 and RX_FIFO_TRIG = 0x0 (all zeros) is not supported (minimum 1 character required). All zeros result in unpredictable behavior.

The receive threshold is programmed using the UARTi.TCR_REG[7:4] RX_FIFO_TRIG_START and UARTi.TCR_REG[3:0] RX_FIFO_TRIG_HALT fields.

- Trigger levels from 0 to 60 bytes are available with a granularity of four. (Trigger level = 4 x [4-bit register value])
- To ensure correct device operation, ensure that RX_FIFO_TRIG_HALT > RX_FIFO_TRIG when auto-RTS is enabled.

  Delay = [4 + 16 × (1 + CHAR_LENGTH + Parity + Stop – 0.5)] x Baud_rate + 4 × FCLK

  Note: The RTS signal is deasserted after the UART module receives the data over RX_FIFO_TRIG_HALT. Delay means how long the UART module takes to deassert the RTS signal after reaching RX_FIFO_TRIG_HALT.

- In the FIFO interrupt mode with flow control, ensure that the trigger level to HALT transmission is greater than or equal to the receive FIFO trigger level (either the UARTi.TCR_REG[7:4] RX_FIFO_TRIG_START field or the UARTi.FCR_REG[7:6] RX_FIFO_TRIG field); otherwise, FIFO operation stalls. In the FIFO DMA mode with flow control, this concept does not exist, because a DMA request is sent when a byte is received.

17.4.2.2 FIFO Interrupt Mode

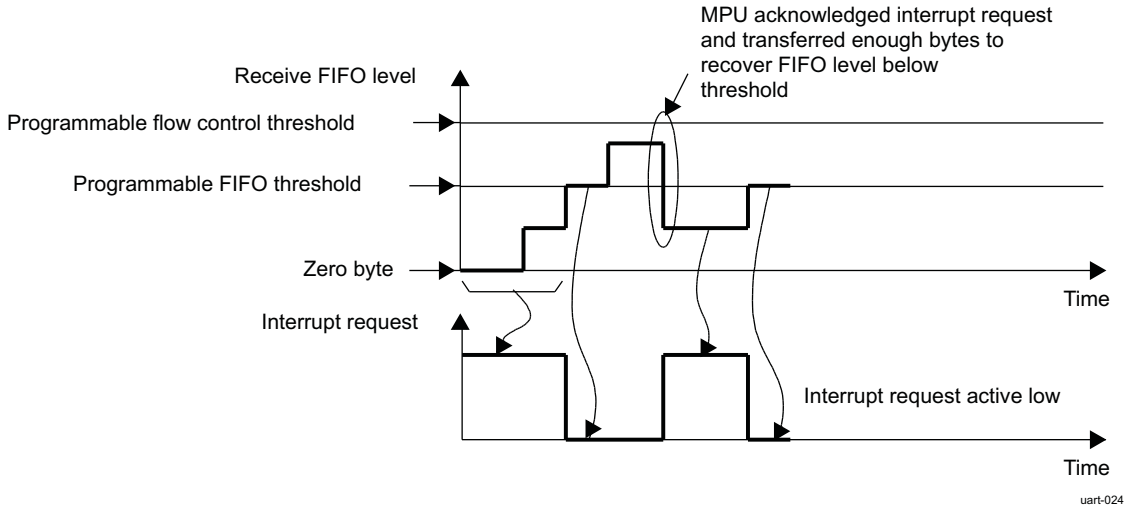
In the FIFO interrupt mode (the FIFO control register UARTi.FCR_REG[0] FIFO_EN bit is set to 1 and relevant interrupts are enabled by the UARTi.IER_REG register), an interrupt signal informs the processor of the status of the receiver and transmitter. These interrupts are raised when the receive/transmit FIFO threshold (the UARTi.TLR_REG[7:4] RX_FIFO_TRIG_DMA and UARTi.TLR_REG[3:0] TX_FIFO_TRIG_DMA fields or the UARTi.FCR_REG[7:6] RX_FIFO_TRIG and UARTi.FCR_REG[5:4] TX_FIFO_TRIG fields, respectively) is reached.

The interrupt signals instruct the MPU to transfer data to the destination (from the UART module in receive mode and/or from any source to the UART FIFO in transmit mode).

When the UART flow control is enabled with the interrupt capabilities, the UART flow control FIFO threshold (UARTi.TCR_REG[3:0] RX_FIFO_TRIG_HALT field) must be greater than or equal to the receive FIFO threshold.

Figure 17-23 shows the generation of the receive FIFO interrupt request.

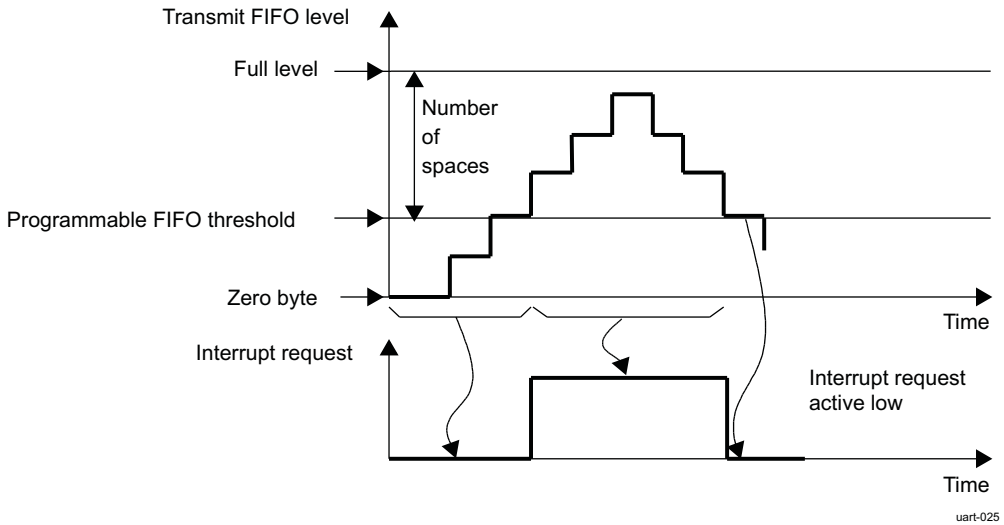
Figure 17-23. Receive FIFO Interrupt Request Generation



In receive mode, no interrupt is generated until the receive FIFO reaches its threshold. Once low, the interrupt can be deasserted only when the MPU has handled enough bytes to make the FIFO level below threshold. The flow control threshold is set at a higher value than the FIFO threshold.

Figure 17-24 shows the generation of the transmit FIFO interrupt request.

Figure 17-24. Transmit FIFO Interrupt Request Generation



In transmit mode, an interrupt request is automatically asserted when the TX FIFO is empty. This request is deasserted when the TX FIFO crosses the threshold level. The interrupt line is deasserted until a sufficient number of elements is transmitted to go below the TX FIFO threshold.

17.4.2.3 FIFO Polled Mode Operation

In the FIFO polled mode (the UARTi.FCR_REG[0] FIFO_EN bit is set to 0 and the relevant interrupts are disabled by the UARTi.IER_REG register), the status of the receiver and transmitter can be checked by polling the line status register (UARTi.LSR_REG).

This mode is an alternative to the FIFO interrupt mode of operation in which the status of the receiver and transmitter is automatically determined by sending interrupts to the MPU.

17.4.2.4 FIFO DMA Mode Operation

Although DMA operation includes four modes (DMA modes 0/1/2/3), the information in Table 17-16 assumes that mode 1 is used. (Mode 2 and mode 3 are legacy modes that use only one DMA request for each module.)

In mode 2, the remaining DMA request is used for RX. In mode 3, the remaining DMA request is used for TX.

The DMA requests in mode 2 and mode 3 use S_DMA_48, S_DMA_50, and S_DMA_52/D_DMA_10. S_DMA_49, S_DMA_51, and S_DMA_53/D_DMA_11 are not used by the module in mode 2 and mode 3 and can be selected as follows:

- When the UARTi.SCR_REG[0] DMA_MODE_CTL bit is set to 0, setting the UARTi.FCR_REG[3] DMA_MODE bit to 0 enables DMA mode 0. Setting the DMA_MODE bit to 1 enables DMA mode 1.
- When the DMA_MODE_CTL bit is set to 1, the UARTi.FCR_REG[2:1] DMA_MODE_2 field determines DMA mode 0 to 3 based on the supplementary control register (SCR_REG) description.

For example:

- If no DMA operation is desired, set the DMA_MODE_CTL bit to 1 and the DMA_MODE_2 field to 0x0. (The DMA_MODE bit is discarded.)
- If DMA mode 1 is desired, set either the DMA_MODE_CTL bit to 0 and the DMA_MODE bit to 1, or set the DMA_MODE_CTL bit to 1 and the DMA_MODE_2 field to 01. (The DMA_MODE bit is discarded.)

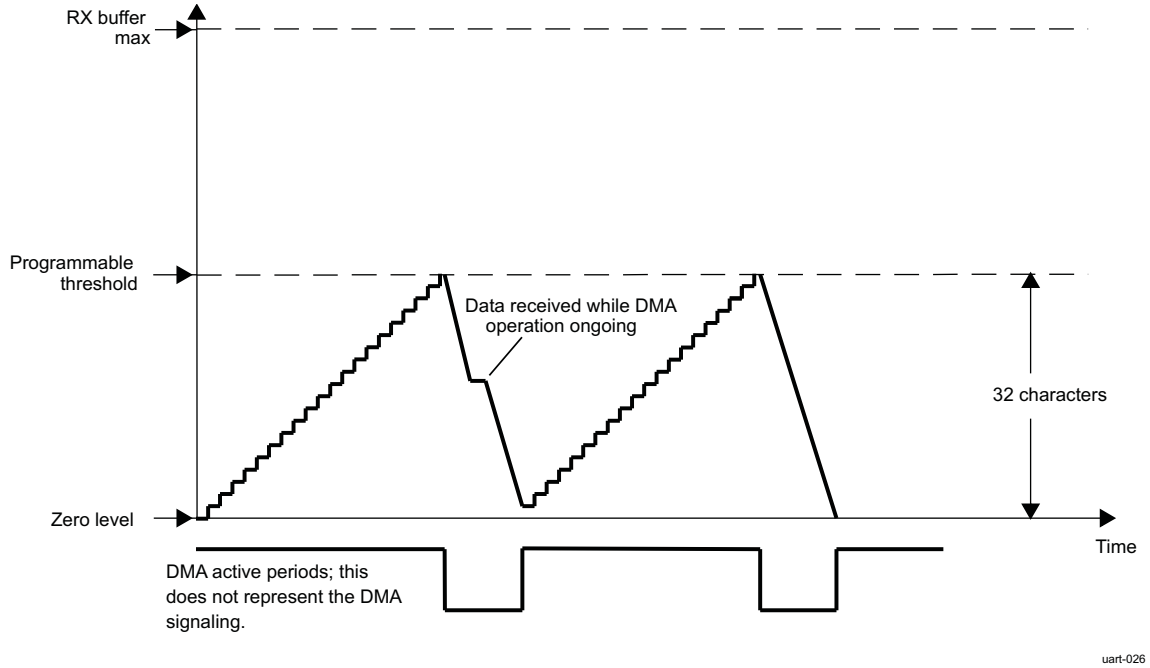
If the FIFOs are disabled (the UARTi.FCR_REG[0] FIFO_EN bit = 0), the DMA occurs in single-character transfers.

When DMA mode 0 is programmed, the signals associated with DMA operation are not active.

17.4.2.4.1 DMA Transfers (DMA Mode 1, 2, or 3)

Figure 17-25 through Figure 17-28 show the supported DMA operations.

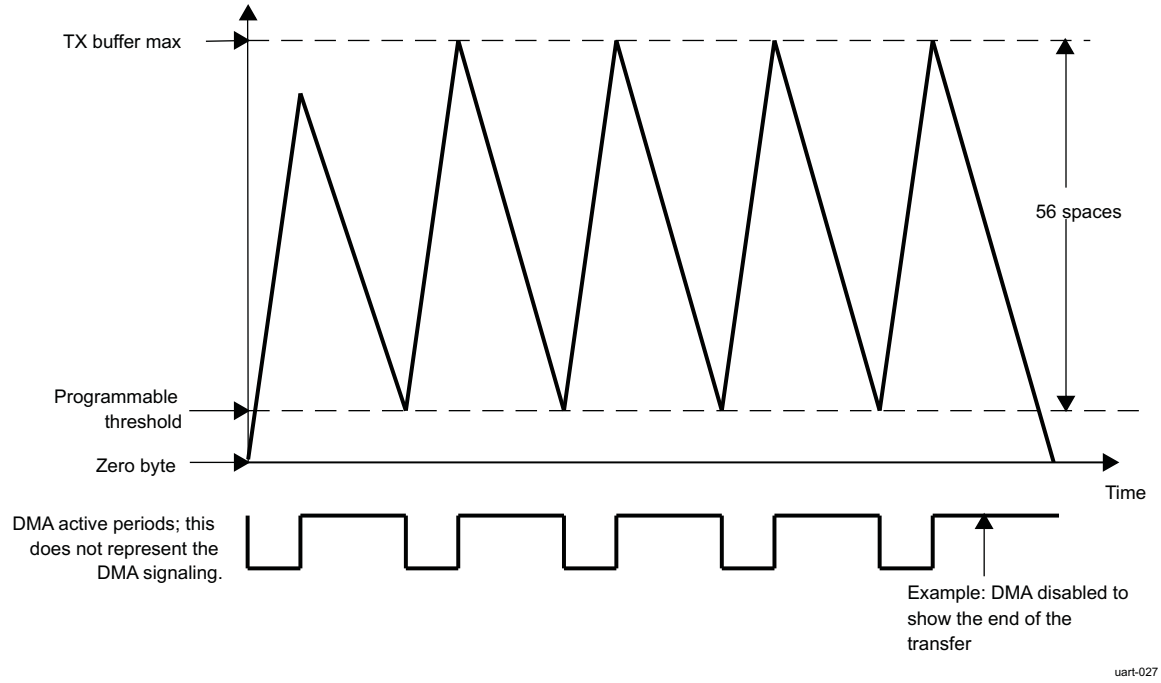
Figure 17-25. Receive FIFO DMA Request Generation (32 Characters)



In receive mode, a DMA request is generated when the receive FIFO reaches its threshold level defined in the trigger level register (UARTi.TLR_REG). This request is deasserted when the number of bytes defined by the threshold level is read by the system DMA (sDMA).

In transmit mode, a DMA request is automatically asserted when the transmit FIFO is empty. This request is deasserted when the number of bytes defined by the number of spaces in the trigger level register (UARTi.TLR_REG) is written by the sDMA. If an insufficient number of characters is written, the DMA request stays active.

Figure 17-26. Transmit FIFO DMA Request Generation (56 Spaces)



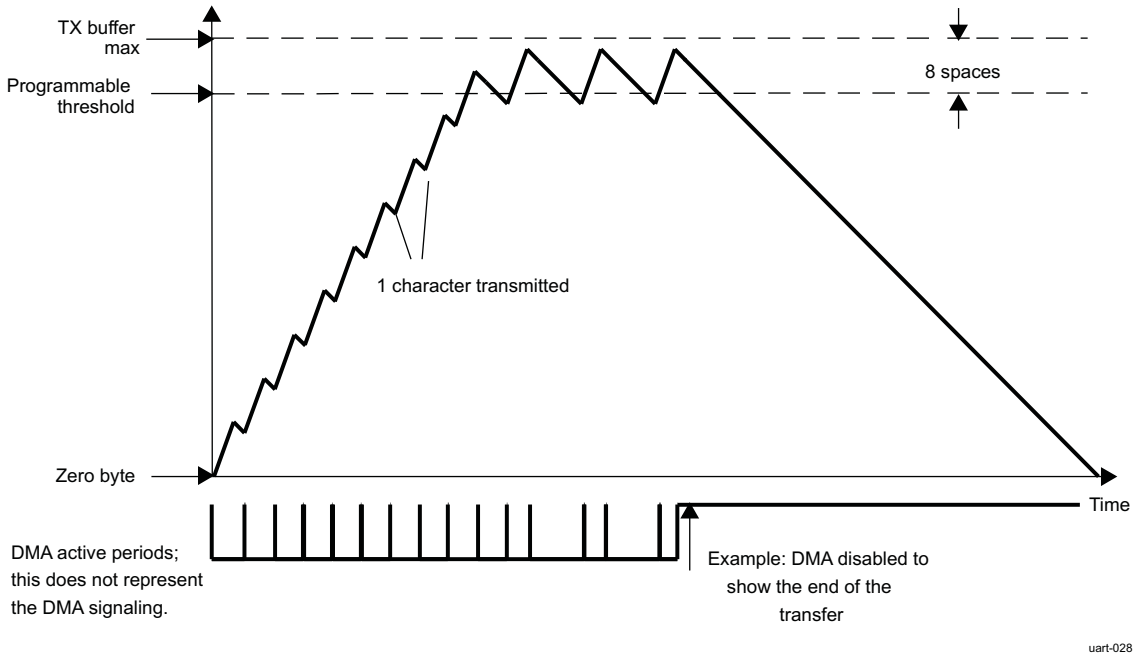
The DMA request is again asserted if the FIFO can receive the number of bytes defined by the UARTi.TLR_REG register.

The threshold can be programmed in a number of ways. Figure 17-26 shows a DMA transfer operating with a space setting of 56 that can arise from using the auto settings in the UARTi.FCR_REG[5:4] TX_FIFO_TRIG field or the UARTi.TLR_REG[3:0] TX_FIFO_TRIG_DMA field concatenated with the TX_FIFO_TRIG field.

The setting of 56 spaces in the UART/IrDA/CIR module must correlate with the settings of the sDMA so that the buffer does not overflow (program the DMA request size of the local host controller to be equal to the number of spaces value in the UART/IrDA/CIR module).

Figure 17-27 shows an example with eight spaces to show the buffer level crossing the space threshold. Again, the local host DMA controller settings must correspond to that of the UART/IrDA/CIR module.

Figure 17-27. Transmit FIFO DMA Request Generation (8 Spaces)

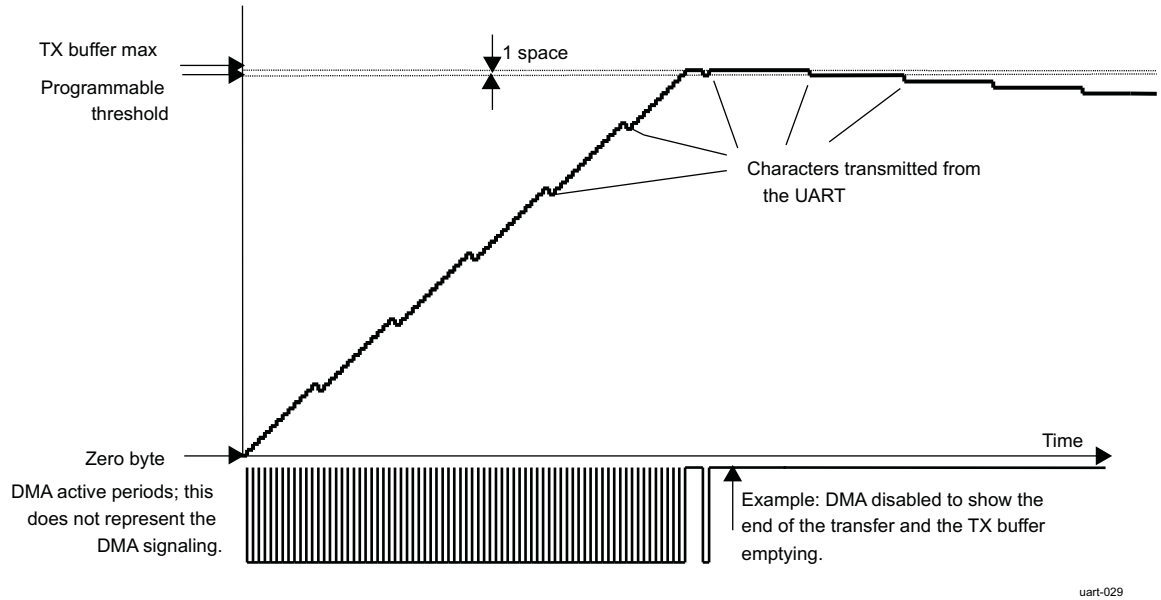


The final example shows the setting of one space that uses the DMA for each transfer of one character to the transmit buffer (see Figure 17-28). The buffer is filled at a faster rate than the baud rate at which data is transmitted to the TX pin. Eventually, the buffer is completely full and the DMA operations stop transferring data to the transmit buffer.

On two occasions the buffer holds the maximum amount of data words; shortly after this, the DMA is disabled to show the slower transmission of the data words to the TX pin. Eventually, the buffer is emptied at the rate specified by the baud rate settings of the UARTi.DLL_REG and the UARTi.DLH_REG registers.

Again, the DMA settings must correspond to the system local host DMA controller settings to ensure correct operation of this logic.

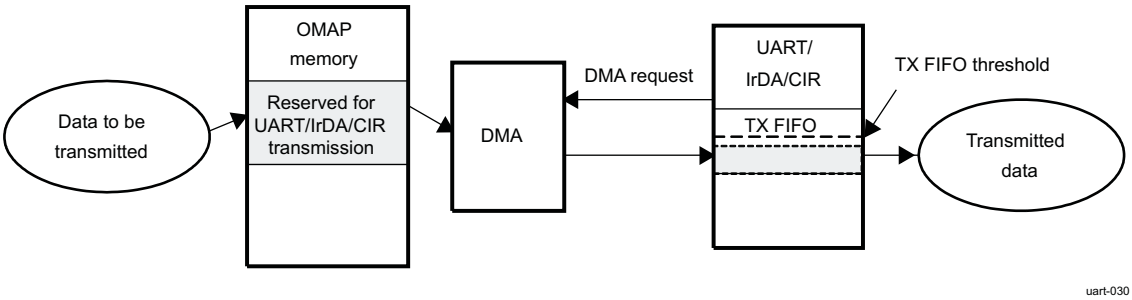
Figure 17-28. Transmit FIFO DMA Request Generation (1 Space)



17.4.2.4.2 DMA Transmission

Figure 17-29 shows the DMA transmission process.

Figure 17-29. Transmission Process



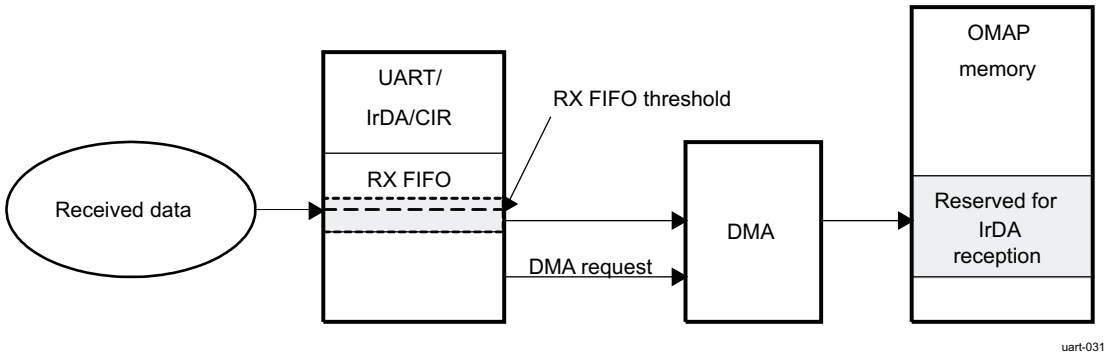
- Data to be transmitted are put in the OMAP34xx memory reserved for UART/IrDA/CIR transmission by the DMA:
  - Until the TX FIFO trigger level is not reached, a DMA request is generated.
  - An element (1 byte) is transferred from the SDRAM to the TX FIFO at each DMA request (DMA element synchronization).
- Data in the TX FIFO are automatically transmitted.
- The end of the transmission is signaled by the UARTi.THR_REG empty (TX FIFO empty).

Note: In IrDA mode, the transmission is not ended immediately after the TX FIFO empties, at which point there are still the last data byte, the CRC field, and the stop flag to be transmitted; thus, the end of transmission is a few milliseconds after the UARTi.THR_REG register empties.

17.4.2.4.3 DMA Reception

Figure 17-30 shows the DMA reception process.

Figure 17-30. Reception Process



1. Enable the reception.
2. Received data are put in the RX FIFO.
3. Data are transferred from the RX FIFO to the device memory by the DMA.
   - At each received byte, the RX FIFO trigger level (one character) is reached and a DMA request is generated.
   - An element (1 byte) is transferred from the RX FIFO to the SDRAM at each DMA request (DMA element synchronization).
4. The end of the reception is signaled by the EOF interrupt.

17.4.3 Mode Selection

17.4.3.1 Register Access Modes

17.4.3.1.1 Operational Mode and Configuration Modes

Register access depends on the register access mode, although register access modes are not correlated to functional mode selection. Three different modes are available:

- Operational mode
- Configuration mode A
- Configuration mode B

Operational mode is the selected mode when the function is active; serial data transfer can be performed in this mode.

Both configuration mode A and configuration mode B are used during module initialization steps. These modes enable access to configuration registers, which are hidden in the operational mode. The modes are used when the module is inactive (no serial data transfer processed) and only in the initialization step or reconfiguration of the module.

The value of the UARTi.LCR_REG register determines the register access mode (see Table 17-21).

Table 17-21. UART/IrDA/CIR Register Access Mode Programming (Using LCR_REG)

Mode	Condition
Configuration_mode_A	LCR_REG[7] = 0x1 and LCR_REG[7:0]! = 0xBF
Configuration_mode_B	LCR_REG[7] = 0x1 and LCR_REG[7:0] = 0xBF
Operational_mode	LCR_REG[7] = 0x0

17.4.3.1.2 Register Access Submode

In each access register mode (operational mode or configuration mode A/B), some register accesses are conditional to the programming of a submode (MSR_SPR, TCR_TLR, and XOFF). These registers are identified in Section 17.6, *UART/IrDA/CIR Register Manual*.

Table 17-22 through Table 17-24 summarize the register access submodes.

Table 17-22. Sub-Configuration_Mode_A Mode Summary

Mode	Condition
MSR_SPR	(EFR_REG[4] = 0x0 or MCR_REG[6] = 0x0)
TCR_TLR	EFR_REG[4] = 0x1 and MCR_REG[6] = 0x1

Table 17-23. Sub-Configuration_Mode_B Mode Summary

Mode	Condition
TCR_TLR	EFR_REG[4] = 0x1 and MCR_REG[6] = 0x1
XOFF	(EFR_REG[4] = 0x0 or MCR_REG[6] = 0x0)

Table 17-24. Sub-Operational_Mode Mode Summary

Mode	Condition
MSR_SPR	EFR_REG[4] = 0x0 or MCR_REG[6] = 0x0)
TCR_TLR	EFR_REG[4] = 0x1 and MCR_REG[6] = 0x1

17.4.3.1.3 Registers Available for the Register Access Modes

Table 17-25 lists the names of the register hits in each access register mode. Gray-shaded registers do not depend on the register access mode (available in all modes).

Table 17-25. UART/IrDA/CIR Register Access Mode Overview

Address Offset	Registers					
	Configuration_Mode_A		Configuration_Mode_B		Operational_Mode	
	Read	Write	Read	Write	Read	Write
0x000	DLL_REG	DLL_REG	DLL_REG	DLL_REG	RHR_REG	THR_REG
0x004	DLH_REG	DLH_REG	DLH_REG	DLH_REG	IER_REG	IER_REG
0x008	IIR_REG	FCR_REG	EFR_REG	EFR_REG	IIR_REG	FCR_REG
0x00C	LCR_REG	LCR_REG	LCR_REG	LCR_REG	LCR_REG	LCR_REG
0x010	MCR_REG	MCR_REG	XON1_ADDR1_REG	XON1_ADDR1_REG	MCR_REG	MCR_REG
0x014	LSR_REG	-	XON2_ADDR2_REG	XON2_ADDR2_REG	LSR_REG	-
0x018	MSR_REG(1)/TCR_REG(2)	TCR_REG (2)	TCR_REG(2)/XOFF1_REG(3)	TCR_REG(2)/XOFF1_REG(3)	MSR_REG(1)/TCR_REG(2)	TCR_REG (2)
0x01C	SPR_REG(1)/TLR_REG(2)	SPR_REG(1)/TLR_REG(2)	TLR_REG(2)/XOFF2_REG(3)	TLR_REG(2)/XOFF2_REG(3)	SPR_REG(1)/TLR_REG(2)	SPR_REG(1)/TLR_REG(2)
0x020	MDR1_REG	MDR1_REG	MDR1_REG	MDR1_REG	MDR1_REG	MDR1_REG
0x024	MDR2_REG	MDR2_REG	MDR2_REG	MDR2_REG	MDR2_REG	MDR2_REG
0x028	SFLSR_REG	TXFLL_REG	SFLSR_REG	TXFLL_REG	SFLSR_REG	TXFLL_REG
0x02C	RESUME_REG	TXFLH_REG	RESUME_REG	TXFLH_REG	RESUME_REG	TXFLH_REG
0x030	SFREGL_REG	RXFLL_REG	SFREGL_REG	RXFLL_REG	SFREGL_REG	RXFLL_REG
0x034	SFREGH_REG	RXFLH_REG	SFREGH_REG	RXFLH_REG	SFREGH_REG	RXFLH_REG
0x038	UASR_REG	-	UASR_REG	-	BLR_REG	BLR_REG
0x03C	-	-	-	-	ACREG_REG	ACREG_REG
0x040	SCR_REG	SCR_REG	SCR_REG	SCR_REG	SCR_REG	SCR_REG
0x044	SSR_REG	-	SSR_REG	-	SSR_REG	-
0x048	-	-	-	-	EBLR_REG	EBLR_REG
0x050	MVR_REG	-	MVR_REG	-	MVR_REG	-
0x054	SYSC_REG	SYSC_REG	SYSC_REG	SYSC_REG	SYSC_REG	SYSC_REG
0x058	SYSS_REG	-	SYSS_REG	-	SYSS_REG	-
0x05C	WER_REG	WER_REG	WER_REG	WER_REG	WER_REG	WER_REG
0x060	CFPS_REG	CFPS_REG	CFPS_REG	CFPS_REG	CFPS_REG	CFPS_REG

(1) if MSR_SPR mode is active (see Section 17.4.3.1.2, *Register Access Submode*)
(2) if TCR_TLR mode is active (see Section 17.4.3.1.2, *Register Access Submode*)
(3) if XOFF mode is active (see Section 17.4.3.1.2, *Register Access Submode*)

17.4.3.2 UART/IrDA (SIR, MIR, FIR)/CIR Mode Selection

To select a mode, set the UARTi.MDR1_REG[2:0] MODE_SELECT field (see Table 17-26).

Table 17-26. UART Mode Selection

Value	Mode
0x0:	UART 16x mode
0x1:	SIR mode (UART3 only)
0x2:	UART 16x auto-baud
0x3:	UART 13x mode
0x4:	MIR mode (UART3 only)
0x5:	FIR mode (UART3 only)
0x6:	CIR mode (UART3 only)

MODE_SELECT is effective when the module is in operational mode (see Section 17.4.3.1, *Register Access Modes*).

17.4.3.2.1 Registers Available for the UART Function

Only the registers listed in Table 17-27 are used for the UART function.

Table 17-27. UART Mode Register Overview[(1)][(2)]

Address Offset	Registers					
	Configuration_Mode_A		Configuration_Mode_B		Operational_Mode	
	Read	Write	Read	Write	Read	Write
0x000	DLL_REG	DLL_REG	DLL_REG	DLL_REG	RHR_REG	THR_REG
0x004	DLH_REG	DLH_REG	DLH_REG	DLH_REG	IER_REG(UART)	IER_REG(UART)
0x008	IIR_REG	FCR_REG	EFR_REG[4]	EFR_REG[4]	IIR_REG(UART)	FCR_REG(UART)
0x00C	LCR_REG	LCR_REG	LCR_REG	LCR_REG	LCR_REG	LCR_REG
0x010	MCR_REG	MCR_REG	XON1_ADDR1_REG	XON1_ADDR1_REG	MCR_REG	MCR_REG
0x014	LSR_REG(UART)	-	XON2_ADDR2_REG	XON2_ADDR2_REG	LSR_REG(UART)	-
0x018	MSR_REG/TCR_REG	TCR_REG	XOFF1_REG/TCR_REG	XOFF1_REG/TCR_REG	MSR_REG/TCR_REG	TCR_REG
0x01C	TLR_REG/SPR_REG	TLR_REG/SPR_REG	TLR_REG/XOFF2_REG	TLR_REG/XOFF2_REG	TLR_REG/SPR_REG	TLR_REG/SPR_REG
0x020	MDR1_REG	MDR1_REG[2:0]	MDR1_REG[2:0]	MDR1_REG[2:0]	MDR1_REG[2:0]	MDR1_REG[2:0]
0x024	MDR2_REG	MDR2_REG	MDR2_REG	MDR2_REG	MDR2_REG	MDR2_REG
0x028	-	-	-	-	-	-
0x02C	-	-	-	-	-	-
0x030	-	-	-	-	-	-
0x034	-	-	-	-	-	-
0x038	UASR_REG	-	UASR_REG	-	-	-
0x03C	-	-	-	-	-	-
0x040	SCR_REG	SCR_REG	SCR_REG	SCR_REG	SCR_REG	SCR_REG
0x044	SSR_REG	-	SSR_REG	-	SSR_REG	-

(1) REGISTER_NAME(UART) notation indicates that the register exists for other functions (IrDA or CIR), but fields have different meanings for other functions (described separately in Section 17.6, *UART/IrDA/CIR Register Manual*).

(2) REGISTER_NAME[m:n] notation indicates that only register bits numbered m to n apply to the UART function.

Table 17-27. UART Mode Register Overview (continued)

Address Offset	Registers					
	Configuration_Mode_A		Configuration_Mode_B		Operational_Mode	
	Read	Write	Read	Write	Read	Write
0x048	-	-	-	-	-	-
0x050	MVR_REG	-	MVR_REG	-	MVR_REG	-
0x054	SYSC_REG	SYSC_REG	SYSC_REG	SYSC_REG	SYSC_REG	SYSC_REG
0x058	SYSS_REG	-	SYSS_REG	-	SYSS_REG	-
0x05C	WER_REG	WER_REG	WER_REG	WER_REG	WER_REG	WER_REG
0x060	-	-	-	-	-	-

17.4.3.2.2 Registers Available for the IrDA Function (UART3 Only)

Only the registers listed in Table 17-28 are used for the IrDA function.

Table 17-28. IrDA Mode Register Overview[(1)][(2)]

Address Offset	Registers					
	Configuration_Mode_A		Configuration_Mode_B		Operational_Mode	
	Read	Write	Read	Write	Read	Write
0x000	DLL_REG	DLL_REG	DLL_REG	DLL_REG	RHR_REG	THR_REG
0x004	DLH_REG	DLH_REG	DLH_REG	DLH_REG	IER_REG(IrDA)	IER_REG(IrDA)
0x008	IIR_REG	FCR_REG	EFR_REG[4]	EFR_REG[4]	IIR_REG(IrDA)	FCR_REG(IrDA)
0x00C	LCR_REG	LCR_REG	LCR_REG	LCR_REG	LCR_REG	LCR_REG
0x010	-	-	XON1_ADDR1_REG	XON1_ADDR1_REG	MCR_REG	MCR_REG
0x014	LSR_REG(IrDA)	-	XON2_ADDR2_REG	XON2_ADDR2_REG	LSR_REG(IrDA)	-
0x018	MSR_REG/TCR_REG	TCR_REG	TCR_REG	TCR_REG	MSR_REG/TCR_REG	TCR_REG
0x01C	TLR_REG/SPR_REG	TLR_REG/SPR_REG	TLR_REG	TLR_REG	TLR_REG/SPR_REG	TLR_REG/SPR_REG
0x020	MDR1_REG	MDR1_REG	MDR1_REG	MDR1_REG	MDR1_REG	MDR1_REG
0x024	MDR2_REG	MDR2_REG	MDR2_REG	MDR2_REG	MDR2_REG	MDR2_REG
0x028	SFLSR_REG	TXFLL_REG	SFLSR_REG	TXFLL_REG	SFLSR_REG	TXFLL_REG
0x02C	RESUME_REG	TXFLH_REG	RESUME_REG	TXFLH_REG	RESUME_REG	TXFLH_REG
0x030	SFREGL_REG	RXFLL_REG	SFREGL_REG	RXFLL_REG	SFREGL_REG	RXFLL_REG
0x034	SFREGH_REG	RXFLH_REG	SFREGH_REG	RXFLH_REG	SFREGH_REG	RXFLH_REG
0x038	-	-	-	-	BLR_REG	BLR_REG
0x03C	-	-	-	-	ACREG_REG	ACREG_REG
0x040	SCR_REG	SCR_REG	SCR_REG	SCR_REG	SCR_REG	SCR_REG
0x044	SSR_REG	-	SSR_REG	-	SSR_REG	-
0x048	-	-	-	-	EBLR_REG	EBLR_REG
0x050	MVR_REG	-	MVR_REG	-	MVR_REG	-
0x054	SYSC_REG	SYSC_REG	SYSC_REG	SYSC_REG	SYSC_REG	SYSC_REG
0x058	SYSS_REG	-	SYSS_REG	-	SYSS_REG	-
0x05C	WER_REG[6:4]	WER_REG[6:4]	WER_REG[6:4]	WER_REG[6:4]	WER_REG[6:4]	WER_REG[6:4]

(1) REGISTER_NAME(UART) notation indicates that the register exists for other functions (IrDA or CIR), but fields have different meanings for other functions (described separately in Section 17.6, *UART/IrDA/CIR Register Manual*).

(2) REGISTER_NAME[m:n] notation indicates that only register bits numbered m to n apply to the UART function.

Table 17-28. IrDA Mode Register Overview (continued)

Address Offset	Registers					
	Configuration_Mode_A		Configuration_Mode_B		Operational_Mode	
	Read	Write	Read	Write	Read	Write
0x060	-	-	-	-	-	-

17.4.3.2.3 Registers Available for the CIR Function (UART3 Only)

Only the registers listed in Table 17-29 are used for CIR function.

Table 17-29. CIR Mode Register Overview[1][2]

Address Offset	Registers					
	Configuration_Mode_A		Configuration_Mode_B		Operational_Mode	
	Read	Write	Read	Write	Read	Write
0x000	DLL_REG	DLL_REG	DLL_REG	DLL_REG		THR_REG
0x004	DLH_REG	DLH_REG	DLH_REG	DLH_REG	IER_REG(CIR)	IER_REG(CIR)
0x008	IIR_REG	FCR_REG	EFR_REG	EFR_REG	IIR_REG(CIR)	FCR_REG(CIR)
0x00C	LCR_REG	LCR_REG[7]	LCR_REG[7]	LCR_REG[7]	LCR_REG[7]	LCR_REG[7]
0x010	-	-	-	-	-	-
0x014	LSR_REG(IrDA)	-	-	-	LSR_REG(IrDA)	-
0x018	MSR_REG/TCR_REG	TCR_REG	TCR_REG	TCR_REG	MSR_REG/TCR_REG	TCR_REG
0x01C	TLR_REG/SPR_REG	TLR_REG/SPR_REG	TLR_REG	TLR_REG	TLR_REG/SPR_REG	TLR_REG/SPR_REG
0x020	MDR1_REG[3:0]	MDR1_REG[3:0]	MDR1_REG[3:0]	MDR1_REG[3:0]	MDR1_REG[3:0]	MDR1_REG[3:0]
0x024	MDR2_REG	MDR2_REG	MDR2_REG	MDR2_REG	MDR2_REG	MDR2_REG
0x028	-	-	-	-	-	-
0x02C	RESUME_REG	-	RESUME_REG	-	RESUME_REG	-
0x030	-	-	-	-	-	-
0x034	-	-	-	-	-	-
0x038	-	-	-	-	-	-
0x03C	-	-	-	-	ACREG_REG	ACREG_REG
0x040	SCR_REG	SCR_REG	SCR_REG	SCR_REG	SCR_REG	SCR_REG
0x044	SSR_REG	-	SSR_REG	-	SSR_REG	-
0x048	-	-	-	-	EBLR_REG	EBLR_REG
0x050	MVR_REG	-	MVR_REG	-	MVR_REG	-
0x054	SYSC_REG	SYSC_REG	SYSC_REG	SYSC_REG	SYSC_REG	SYSC_REG
0x058	SYSS_REG	-	SYSS_REG	-	SYSS_REG	-
0x05C	WER_REG[6:4]	WER_REG[6:4]	WER_REG[6:4]	WER_REG[6:4]	WER_REG[6:4]	WER_REG[6:4]
0x060	CFPS_REG	CFPS_REG	CFPS_REG	CFPS_REG	CFPS_REG	CFPS_REG

(1) REGISTER_NAME(UART) notation indicates that the register exists for other functions (IrDA or CIR), but fields have different meanings for other functions (described separately in Section 17.6, *UART/IrDA/CIR Register Manual*).

(2) REGISTER_NAME[m:n] notation indicates that only register bits numbered m to n apply to the UART function.

17.4.4 Protocol Formatting

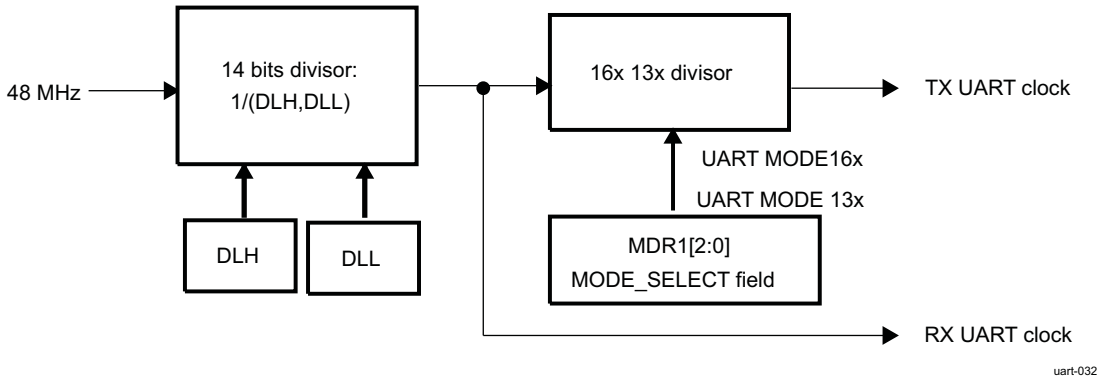
17.4.4.1 UART Mode

17.4.4.1.1 UART Clock Generation: Baud Rate Generation

The UART function contains a programmable baud generator and a set of fixed dividers that divide the 48-MHz clock input down to the expected baud rate.

Figure 17-31 shows the baud rate generator and associated controls.

Figure 17-31. Baud Rate Generation



CAUTION

It is mandatory that MODE_SELECT = DISABLE (UARTi.MDR1_REG[2:0] = 0x7) before initializing or modifying clock parameter controls (UARTi.DLH_REG, UARTi.DLL_REG). Failure to observe this rule can result in unpredictable module behavior.

17.4.4.1.2 Choosing the Appropriate Divisor Value

Two divisor values are:

- UART 16x mode: Divisor value = Operating frequency/(16x baud rate)
- UART 13x mode: Divisor value = Operating frequency/(13x baud rate)

Table 17-30 describes the UART baud rate settings.

Table 17-30. UART Baud Rate Settings (48-MHz Clock)

Baud Rate	Baud Multiple	DLH,DLL (Decimal)	DLH,DLL (Hex)	Actual Baud Rate	Error (%)
0.3 Kbps	16x	10000	0x27, 0x10	0.3 Kbps	0
0.6 Kbps	16x	5000	0x13, 0x88	0.6 Kbps	0
1.2 Kbps	16x	2500	0x09, 0xC4	1.2 Kbps	0
2.4 Kbps	16x	1250	0x04, 0xE2	2.4 Kbps	0
4.8 Kbps	16x	625	0x02, 0x71	4.8 Kbps	0
9.6 Kbps	16x	312	0x01, 0x38	9.6153 Kbps	+0.16
14.4 Kbps	16x	208	0x00, 0xD0	14.423 Kbps	+0.16
19.2 Kbps	16x	156	0x00, 0x9C	19.231 Kbps	+0.16
28.8 Kbps	16x	104	0x00, 0x68	28.846 Kbps	+0.16

Table 17-30. UART Baud Rate Settings (48-MHz Clock) (continued)

Baud Rate	Baud Multiple	DLH,DLL (Decimal)	DLH,DLL (Hex)	Actual Baud Rate	Error (%)
38.4 Kbps	16x	78	0x00, 0x4E	38.462 Kbps	+0.16
57.6 Kbps	16x	52	0x00, 0x34	57.692 Kbps	+0.16
115.2 Kbps	16x	26	0x00, 0x1A	115.38 Kbps	+0.16
230.4 Kbps	16x	13	0x00, 0x0D	230.77 Kbps	+0.16
460.8 Kbps	13x	8	0x00, 0x08	461.54 Kbps	+0.16
921.6 Kbps	13x	4	0x00, 0x04	923.08 Kbps	+0.16
1.843 Mbps	13x	2	0x00, 0x02	1.846 Mbps	+0.16
3.6884 Mbps	13x	1	0x00, 0x01	3.6923 Mbps	+0.16

17.4.4.1.3 UART Data Formatting

The UART module can use hardware flow control to manage transmission/reception. Hardware flow control significantly reduces software overhead and increases system efficiency by automatically controlling serial data flow using the RTS output and CTS input signals.

The UART module is enhanced with the autobauding function. In control mode, autobauding allows the speed, the number of bits per character, and the parity selected to be set automatically.

17.4.4.1.3.1 Frame Formatting

When autobauding is not used, frame format attributes must be defined in the UARTi.LCR_REG register.

Character length is specified using the UARTi.LCR_REG[1:0] CHAR_LENGTH field.

The number of stop bits is specified using the UARTi.LCR_REG[2] NB_STOP register bit.

The parity bit is programmed using the UARTi.LCR_REG[5:3] PARITY_EN, PARITY_TYPE_1, and PARITY_TYPE_2 register bits (see Table 17-31).

Table 17-31. UART Parity Bit Encoding

PARITY_EN	PARITY_TYPE1	PARITY_TYPE2	Parity
0	N/A	N/A	No parity
1	0	0	Odd parity
1	1	0	Even parity
1	0	1	Forced 1
1	1	1	Forced 0

17.4.4.1.3.2 Hardware Flow Control

Hardware flow control is composed of auto-CTS and auto-RTS. Auto-CTS and auto-RTS can be enabled/disabled independently by programming the UARTi.EFR_REG[7:6] AUTO_CTS_EN and AUTO_RTS_EN bits, respectively.

With auto-CTS, uarti_cts must be active before the module can transmit data.

Auto-RTS activates the uarti_rts output only when there is enough room in the RX FIFO to receive data. It deactivates the uarti_rts output when the RX FIFO is sufficiently full. The HALT and RESTORE trigger levels in the UARTi.TCR_REG register determine the levels at which uarti_rts is activated/deactivated.

If both auto-CTS and auto-RTS are enabled, data transmission does not occur unless the receiver FIFO has empty space. Thus, overrun errors are eliminated during hardware flow control. If auto-CTS and auto-RTS are not enabled, overrun errors occur if the transmit data rate exceeds the receive FIFO latency.

- **Auto-RTS**

Auto-RTS data flow control originates in the receiver block. The receiver FIFO trigger levels used in auto-RTS are stored in the UARTi.TCR_REG register. uarti_rts is active if the RX FIFO level is below the HALT trigger level in the UARTi.TCR_REG[3:0] RX_FIFO_TRIG_HALT field. When the receiver FIFO HALT trigger level is reached, uarti_rts is deasserted. The sending device (for example, another UART) might send an additional byte after the trigger level is reached because it might not recognize the deassertion of RTS until it begins sending the additional byte.

uarti_rts is automatically reasserted when the receiver FIFO reaches the RESUME trigger level programmed by the UARTi.TCR_REG[7:4] RX_FIFO_TRIG_START field. This reassertion requests the sending device to resume transmission.

In this case, uarti_rts is an active-low signal.

- **Auto-CTS**

  The transmitter circuitry checks uarti_cts before sending the next data byte. When uarti_cts is active, the transmitter sends the next byte. To stop the transmitter from sending the next byte, uarti_cts must be deasserted before the middle of the last stop bit currently sent.

  The auto-CTS function reduces interrupts to the host system. When auto-CTS flow control is enabled, the uarti_cts state changes do not have to trigger host interrupts because the device automatically controls its own transmitter. Without auto-CTS, the transmitter sends any data present in the transmit FIFO, and a receiver overrun error can result.

  In this case, uarti_cts is an active-low signal.

17.4.4.1.3.3 Software Flow Control

Software flow control is enabled through the enhanced feature register (UARTi.EFR_REG) and the modem control register (UARTi.MCR_REG). Different combinations of software flow control can be enabled by setting different combinations of UARTi.EFR_REG[3:0] (see Table 17-32).

Two other enhanced features relate to software flow control:

- XON any function (UARTi.MCR_REG[5]): Operation resumes after receiving any character after recognition of the XOFF character.

Note: The XON-any character is written into the RX FIFO even if it is a software flow character.

- Special character (UARTi.EFR_REG[5]): Incoming data is compared to XOFF2. When the special character is detected, the XOFF interrupt (UARTi.IIR_REG[4]) is set, but it does not halt transmission. The XOFF interrupt is cleared by a read of UARTi.IIR_REG. The special character is transferred to the RX FIFO.

Table 17-32. EFR_REG[0-3] Software Flow Control Options

Bit 3	Bit 2	Bit 1	Bit 0	Tx, Rx Software Flow Controls
0	0	X	X	No transmit flow control
1	0	X	X	Transmit XON1, XOFF1
0	1	X	X	Transmit XON2, XOFF2
1	1	X	X	Transmit XON1, XON2: XOFF1, XOFF2(1)
X	X	0	0	No receive flow control
X	X	1	0	Receiver compares XON1, XOFF1
X	X	0	1	Receiver compares XON2, XOFF2
X	X	1	1	Receiver compares XON1, XON2: XOFF1, XOFF2(1)

(1) In these cases, the XON1 and XON2 characters or the XOFF1 and XOFF2 characters must be transmitted/received sequentially with XON1/XOFF1 followed by XON2/XOFF2.
XON1 is defined in the UARTi.XON1_ADDR1_REG[7:0] XON_WORD1 field, XON2 is defined in UARTi.XON2_ADDR2_REG[7:0] XON_WORD2.
XOFF1 is defined in the UARTi.XOFF1_REG[7:0] XOFF_WORD1 field, XOFF2 is defined in UARTi.XOFF2_REG[7:0] XOFF_WORD2.

17.4.4.1.3.3.1 Receive (RX)

When software flow control operation is enabled, the UART compares incoming data with XOFF1/2 programmed characters (in certain cases, XOFF1 and XOFF2 must be received sequentially). When the correct XOFF characters are received, transmission stops after transmission of the current character is complete. XOFF detection also sets UARTi.IIR_REG[4] (if enabled by UARTi.IER_REG[5]) and causes the interrupt line to go low.

To resume transmission, an XON1/2 character must be received (in certain cases, XON1 and XON2 must be received sequentially). When the correct XON characters are received, UARTi.IIR_REG[4] is cleared and the XOFF interrupt disappears.

Note: When a parity, framing, or break error occurs while receiving a software flow control character, this character is treated as normal data and is written to the RX FIFO.

When XON-any and special character detect are disabled and software flow control is enabled, no valid XON or XOFF characters are written to the RX FIFO. For example, when UARTi.EFR_REG[1:0] = 0x2, if XON1 and XOFF1 characters are received, they do not get written to the RX FIFO.

When pairs of software flow characters are programmed to be received sequentially (UARTi.EFR_REG[1:0] = 0x3), the software flow characters are not written to the RX FIFO if they are received sequentially. However, received XON1/XOFF1 characters must be written to the RX FIFO if the subsequent character is not XON2/XOFF2.

17.4.4.1.3.3.2 Transmit (TX)

XOFF1: Two characters are transmitted when the RX FIFO passes the trigger level programmed by UARTi.TCR_REG[3:0].

XON1: Two characters are transmitted when the RX FIFO reaches the trigger level programmed by UARTi.TCR_REG[7:4].

Note: If software flow control is disabled after an XOFF character is sent, the module transmits XON characters automatically to enable normal transmission to proceed.

The transmission of XOFF(s)/XON(s) follows the same protocol as transmission of an ordinary byte from the TX FIFO. This means that even if the word length is set to 5, 6, or 7 characters, the 5, 6, or 7 LSBs of XOFF1/2 and XON1/2 are transmitted. The 5, 6, or 7 bits of a character are seldom transmitted, but this function is included to maintain compatibility with earlier designs.

It is assumed that software flow control and hardware flow control are never enabled simultaneously.

17.4.4.1.3.4 Autobauding Modes

In autobauding mode, the UART can extract transfer characteristics (speed, length, and parity) from an AT command (ASCII code). These characteristics are used to receive data after an “at” (AT) and to send data.

The valid AT commands follow:

AT	DATA	<CR>
at	DATA	<CR>
A/		
a/		

A line break during the acquisition of the sequence AT is not recognized, and echo functionality is not implemented in hardware.

A/ and a/ are not used to extract characteristics, but they must be recognized because of their special meaning. Either A/ or a/ is used to instruct the software to repeat the last received AT command; therefore, an a/ always follows an AT, and transfer characteristics are not expected to change between an AT and an a/.

When a valid AT is received, AT and all subsequent data, including the final <CR> (0x0D), are saved to RX FIFO. The autobaud state-machine waits for the next valid AT command. If an a/ (A/) is received, the a/ (A/) is saved into RX FIFO and the state-machine waits for the next valid AT command.

On the first successful detection of the baud rate, the UART activates an interrupt to signify that the AT (upper or lower case) sequence is detected. The UARTi.UASR_REG register reflects the correct settings for the baud rate detected. The interrupt activity can continue in this fashion when a subsequent character is received. Therefore, it is recommended that the software enable the RHR interrupt when using the autobaud mode.

The following settings are detected in autobaud mode with a module clock of 48 MHz:

- Speed: 115.2k baud, 57.6k baud, 38.4k baud, 28.8k baud, 19.2k baud, 14.4k baud, 9.6k baud, 4.8k baud, 2.4k baud, or 1.2k baud
- Length: 7 or 8 bits
- Parity: Odd, even, or space

Note: The combination of 7-bit character + space parity is not supported.

Autobauding mode is selected when the UARTi.MDR1_REG[2:0] MODE_SELECT field is set to 0x2. In the UART autobauding mode, UARTi.DLL_REG, UARTi.DLH_REG, UARTi.LCR_REG[5:0] settings are not used; instead, UASR_REG is updated with the configuration detected by the autobauding logic.

UASR_REG Autobauding Status Register Use

This register is used to set up transmission according to the characteristics of the previous reception instead of the UARTi.LCR_REG, UARTi.DLL_REG, and UARTi.DLH_REG registers when the UART is in autobauding mode.

To reset the autobauding hardware (to start a new AT detection) or to set the UART in standard mode (no autobaud), the UARTi.MDR1_REG[2:0] MODE_ SELECT field must be set to reset state (0x7) and then to the UART in autobauding mode (0x2) or to the UART in standard mode (0x0).

Use limitation:

- Only 7- and 8-bit characters (5- and 6-bit not supported)
- 7-bit character with space parity not supported
- Baud rate between 1,200 and 115,200 bps (10 possibilities)

17.4.4.1.3.5 Error Detection

When the UARTi.LSR_REG register is read, UARTi.LSR_REG[4:2] reflects the error bits (BI: break condition, FE: framing error, PE: parity error) of the character at the top of the RX FIFO (next character to be read). Therefore, reading the UARTi.LSR_REG register and then reading the UARTi.RHR_REG register identifies errors in a character.

Reading the UARTi.RHR_REG register updates the BI, FE, and PE bits (see Table 17-33 for the UART mode interrupts).

The UARTi.LSR_REG [7] RX_FIFO_STS bit is set when there is an error in the RX FIFO and is cleared only when no errors remain in the RX FIFO.

Note: Reading UARTi.LSR_REG does not cause an increment of the RX FIFO read pointer. The RX FIFO read pointer is incremented by reading UARTi.RHR_REG.

Reading UARTi.LSR_REG clears the OE bit if it is set (see Table 17-33 for the UART mode interrupts).

17.4.4.1.3.6 Overrun During Receive

Overrun during receive occurs if the RX state-machine tries to write data into the RX FIFO when it is already full. When overrun occurs, the device interrupts the MPU with UARTi.IIR_REG[5:1] IT_TYPE = 0x3 (receiver line status error) and discards the remaining portion of the frame.

Overrun also causes an internal flag to be set, which disables further reception. Before the next frame can be received, the system (MPU) must:

- Reset the RX FIFO
- Read the UARTi.RESUME_REG register (which clears the internal flag)

17.4.4.1.3.7 Time-out and Break Conditions

17.4.4.1.3.7.1 Time-out Counter

An RX idle condition is detected when the receiver line (uarti_rx) is high for a time equivalent to 4x programmed word length + 12 bits. uarti_rx is sampled midway through each bit.

For sleep mode, the counter is reset when there is activity on uarti_rx.

For the time-out interrupt, the counter counts only when there is data in the RX FIFO, and the count is reset when there is activity on uarti_rx or when the UARTi.RHR_REG is read.

17.4.4.1.3.7.2 Break Condition

When a break condition occurs, uarti_tx is pulled low. A break condition is activated by setting the UARTi.LCR_REG[6] BREAK_EN bit. The break condition is not aligned on word stream (that is, a break condition can occur in the middle of a character). The only way to send a break condition on a full character is as follows:

1. Reset the transmit FIFO (if enabled).
2. Wait for the transmit shift register to empty (UARTi.LSR_REG[6] TX_SR_E = 1).
3. Take a guard time according to stop-bit definition.
4. Set the BREAK_EN bit to 1.

The break condition is asserted while the BREAK_EN bit is set to 1.

The time-out counter and break condition apply only to UART modem operation and not to IrDA/CIR mode operation.

17.4.4.1.4 UART Mode Interrupt Management

The UART mode includes seven possible interrupts prioritized to six levels.

When an interrupt is generated, the interrupt identification register (UARTi.IIR_REG) sets the UARTi.IIR_REG[0] IT_PENDING bit to 0 to indicate that an interrupt is pending, and provides the type of interrupt through UARTi.IIR_REG[5:1]. Table 17-33 summarizes the interrupt control functions.

Table 17-33. UART Mode Interrupts

IIR_REG[5:0]	Priority Level	Interrupt Type	Interrrupt Source	Interrupt Reset Method
000001	None	None	None	None
000110	1	Receiver line status	OE, FE, PE, or BI errors occur in characters in the RX FIFO.	FE, PE, BI: Read RHR_REG. OE: Read LSR_REG.
001100	2	RX time-out	Stale data in RX FIFO	Read RHR_REG.
000100	2	RHR interrupt	DRDY (data ready) (FIFO disable)	Read RHR_REG until interrupt condition disappears.
			RX FIFO above trigger level (FIFO enable)	
000010	3	THR interrupt	TFE (THR_REG empty) (FIFO disable)	Write to THR_REG until interrupt condition disappears.

Table 17-33. UART Mode Interrupts (continued)

IIR_REG[5:0]	Priority Level	Interrupt Type	Interrrupt Source	Interrupt Reset Method
			TX FIFO below trigger level (FIFO enable)	
000000	4	Modem status	See the MSR_REG register.	Read MSR_REG.
010000	5	XOFF interrupt/special character interrupt	Receive XOFF characters/special character	Receive XON character(s), if XOFF interrupt/read of IIR_REG, if special character interrupt.
100000	6	CTS, RTS	RTS pin or CTS pin change state from active (low) to inactive (high).	Read IIR_REG.

For the receiver-line status interrupt, the RX_FIFO_STS bit (UARTi.LSR_REG[7]) generates the interrupt.

For the XOFF interrupt, if an XOFF flow character detection caused the interrupt, the interrupt is cleared by an XON flow character detection. If special character detection caused the interrupt, the interrupt is cleared by a read of the UARTi.IIR_REG register.

17.4.4.1.4.1 Wake-up Interrupt

Wake-up interrupt is a special interrupt that is not designed the same way as other interrupts. This interrupt is enabled when the UARTi.SCR_REG[4] RX_CTS_WU_EN bit is set to 1. The UARTi.IIR_REG register is not modified when it occurs; the UART3.SSR_REG[1] RX_CTS_WU_STS bit must be checked to detect a wake-up event.

When a wake-up interrupt occurs, the only way to clear it is to reset the UARTi.SCR_REG[4] RX_CTS_WU_EN bit. This bit must be re-enabled (set to 1) after the current wake-up interrupt event is processed to detect the next incoming wake-up event.

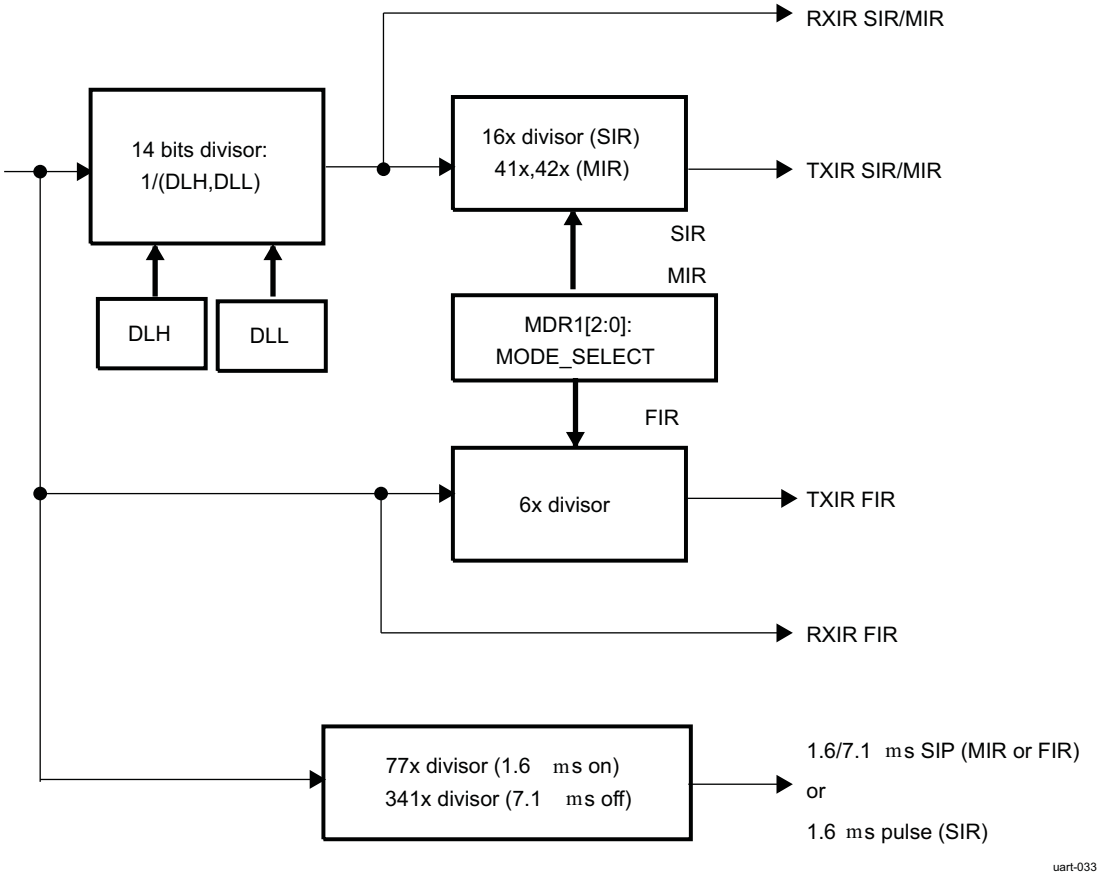
17.4.4.2 IrDA Mode (UART3 Only)

17.4.4.2.1 IrDA Clock Generation: Baud Generator

The IrDA function contains a programmable baud generator and a set of fixed dividers that divide the 48-MHz clock input down to the expected baud rate.

Figure 17-32 shows the baud rate generator and associated controls.

Figure 17-32. Baud Rate Generator



CAUTION

Before trying to initialize or modify clock parameter controls (UARTi.DLH_REG, UARTi.DLL_REG), it is mandatory to set MODE_SELECT=DISABLE (UARTi.MDR1_REG[2:0]=0x7).

Failure to observe this rule can result in unpredictable module behavior.

17.4.4.2.2 Choosing the Appropriate Divisor Value

- SIR mode: Divisor value = Operating frequency/(16x baud rate)
- MIR mode: Divisor value = Operating frequency/(41x/42x baud rate)
- FIR mode: Divisor value = None

Table 17-34 lists the IrDA baud rate settings.

Table 17-34. IrDA Baud Rates Settings

Baud Rate	IR Mode	Baud Multiple	Encoding	DLH, DLL (Decimal)	Actual Baud Rate	Error (%)$	Source Jitter (%)	Pulse duration
2.4 Kbps	SIR	16x	3/16	1250	2.4 Kbps	0	0	78.1 µs
9.6 Kbps	SIR	16x	3/16	312	9.6153 Kbps	+0.16	0	19.5 µs
19.2 Kbps	SIR	16x	3/16	156	19.231 Kbps	+0.16	0	9.75 µs
38.4 Kbps	SIR	16x	3/16	78	38.462 Kbps	+0.16	0	4.87 µs

Table 17-34. IrDA Baud Rates Settings (continued)

Baud Rate	IR Mode	Baud Multiple	Encoding	DLH, DLL (Decimal)	Actual Baud Rate	Error (%)$	Source Jitter (%)	Pulse duration
57.6 Kbps	SIR	16x	3/16	52	57.692 Kbps	+0.16	0	3.25 µs
115.2 Kbps	SIR	16x	3/16	26	115.38 Kbps	+0.16	0	1.62 µs
0.576 Mbps	MIR	41x/42x	1/4	2	0.5756 Mbps(1)	0	+1.63/- 0.80	416 ns
1.152 Mbps	MIR	41x/42x	1/4	1	1.1511 Mbps(1)	0	+1.63/- 0.80	208 ns
4 Mbps	FIR	6x	4 PPM	-	4 Mbps	0	0	125 ns

(1) Average value

Note: Baud rate error and source jitter table values do not include 48-MHz reference clock error and jitter.

17.4.4.2.3 IrDA Data Formatting

The methods described in this section apply to all IrDA modes (SIR, MIR, and FIR).

17.4.4.2.3.1 IRRX Polarity Control

The UART3.MDR2_REG[6] IRRXINVERT bit provides the flexibility to invert the uart3_rx_irrx pin in the UART module to ensure that the protocol at the output of the transceiver module has the same polarity at module level. By default, the uart3_rx_irrx pin is inverted because most transceivers invert the IR receive pin.

17.4.4.2.3.2 IrDA Reception Control

Data can be transferred both ways by the module, but when the device is transmitting, the IR RX circuitry is automatically disabled by hardware.

Operation of the uart3_rx_irrx input can be disabled by the UART3.ACREG_REG[5] DIS_IR_RX bit.

17.4.4.2.3.3 IR Address Checking

In all IR modes, when address checking is enabled, only frames intended for the device are written to the RX FIFO. This restriction avoids receiving frames not meant for this device in a multipoint infrared environment. It is possible to program two frame addresses that the UART IrDA receives with the UART3.XON1_ADDR1_REG[7:0] XON_WORD1 and UART3.XON2_ADDR2_REG[7:0] XON_WORD2 fields.

Setting the EFR_REG[0] bit to 1 selects address1 checking. Setting the EFR_REG[1] bit to 1 selects address2 checking. Setting the EFR_REG[1:0] bit to 0 disables all address checking operations. If both bits are set, the incoming frame is checked for both private and public addresses.

If address checking is disabled, all received frames write to the reception FIFO.

17.4.4.2.3.4 Frame Closing

A transmission frame can be correctly terminated in two ways:

- Frame-length method: The frame-length method is selected by setting the UART3.MDR1_REG[7] FRAME_END_MODE bit to 0. The MPU writes the frame-length value to the UART3.TXFLH_REG and UART3.TXFLL_REG registers. The device automatically attaches end flags to the frame when the number of bytes transmitted equals the frame-length value.
- Set-EOT bit method: The set-EOT bit method is selected by setting the FRAME_END_MODE bit to 1. The MPU writes 1 to the UART3.ACREG_REG[0] EOT bit just before it writes the last byte to the TX FIFO. When the MPU writes the last byte to the TX FIFO, the device internally sets the tag bit for that character in the TX FIFO. As the TX state-machine reads data from the TX FIFO, it uses this tag-bit information to attach end flags and correctly terminate the frame.

17.4.4.2.3.5 Store and Controlled Transmission

In store and controlled transmission (SCT) mode, the MPU starts writing data to the TX FIFO. Then, after writing a part of a frame (for a bigger frame) or a whole frame (a small frame; that is, a supervisory frame), the MPU writes 1 to the UART3.ACREG_REG[2] SCTX_EN bit (deferred TX start) to start transmission.

SCT mode is enabled by setting the UART3.MDR1_REG[5] SCT bit to 1. This transmission method is different from the normal mode, in which data transmission starts immediately after data is written to the TX FIFO. SCT mode is useful for sending short frames without TX underrun.

17.4.4.2.3.6 Error Detection

When the UART3.LSR_REG register is read, the UART3.LSR_REG[4:2] field reflects the error bits [FL, CRC, ABORT] of the frame at the top of the STATUS FIFO (the next frame status to be read).

The error is triggered by an interrupt (see Table 17-35 for IrDA mode interrupts). STATUS FIFO must be read until empty (a maximum of eight reads is required).

17.4.4.2.3.7 Underrun During Transmission

Underrun during transmission occurs when the TX FIFO is empty before the end of the frame is transmitted. When underrun occurs, the device closes the frame with end flags but attaches an incorrect CRC value. The receiving device detects a CRC error and discards the frame; it can then ask for a retransmission.

Underrun also causes an internal flag to be set, which disables additional transmissions. Before the next frame can be transmitted, the system (MPU) must:

- Reset the TX FIFO.
- Read the UART3.RESUME_REG register (which clears the internal flag).

This functionality can be disabled by the UART3.ACREG_REG[4] DIS_TX_UNDERRUN bit, compensated by the extension of the stop bit in transmission if the TX FIFO is empty.

17.4.4.2.3.8 Overrun During Receive

Overrun during receive for the IrDA mode has the same functionality as that for the UART mode (see Section 17.4.4.1.3.6, *Overrun During Receive*).

17.4.4.2.3.9 Status FIFO

In IrDA modes, a status FIFO records the received frame status. When a complete frame is received, the length of the frame and the error bits associated with the frame are written to the status FIFO.

Reading UART3.SFREGH_REG[3:0] (MSB) and UART3.SFREGL_REG[3:0] (LSB) obtains the frame length. The frame error status is read in the UART3.SFLSR_REG register. Reading the UART3.SFLSR_REG register increments the status FIFO read pointer. The status FIFO is eight entries deep and, therefore, can hold the status of eight frames.

The MPU uses the frame-length information to locate the frame boundary in the received frame data. The MPU can screen bad frames using the error status information and can later request the sender to resend only the bad frames.

This status FIFO can be used effectively in DMA mode because the MPU must be interrupted only when the programmed status FIFO trigger level is reached, not each time a frame is received.

17.4.4.2.4 SIR Mode DATA Formatting

This section provides specific instructions for SIR mode programming.

17.4.4.2.4.1 Abort Sequence

When the transmitter prematurely closes a frame, it sends the following sequence to abort: 0x7DC1. The abort pattern closes the frame without a CRC field or an ending flag.

A transmission frame can be aborted by setting the UART3.ACREG_REG[1] ABORT_EN bit to 1.

When this bit is set to 1, 0x7D and 0xC1 are transmitted and the frame is not terminated with CRC or stop flags.

The receiver treats a frame as an aborted frame when a 0x7D character followed immediately by a 0xC1 character is received without transparency.

CAUTION

When the transmit FIFO is not empty and the UART3.MDR1_REG[5] SCT bit is set to 1, the UART IrDA starts a new transfer with data of a previous frame when the abort frame is sent. Therefore, TX FIFO must be reset before sending an abort frame.

17.4.4.2.4.2 Pulse Shaping

The SIR mode supports both the 3/16th or the 1.6-µs pulse duration methods. The UART3.ACREG_REG[7] PULSE_TYPE bit selects the pulse width method in the transmit mode.

17.4.4.2.4.3 SIR Free Format Programming

The SIR FF mode is selected by setting the module in the UART mode (UART3.MDR1_REG[2:0] MODE_SELECT = 0x0) and the UART3.MDR2_REG[3] UART_PULSE bit to 1 to allow pulse shaping.

Because the bit format remains the same, some UART mode configuration registers must be set at specific values:

- UART3.LCR_REG[1:0] CHAR_LENGTH field = 0x3 (8 data bits)
- UART3.LCR_REG[2] NB_STOP bit = 0x0 (1 stop-bit)
- UART3.LCR_REG[3] PARITY_EN bit = 0x0 (no parity)

The UART mode interrupts are used for the SIR FF mode, but many of them are not relevant (XOFF, RTS, CTS, modem status register, etc.).

17.4.4.2.5 MIR and FIR Mode Data Formatting

This section describes common instructions for FIR and MIR mode programming.

At the end of a frame reception, the MPU reads the line status register (UART3.LSR_REG) to detect possible errors in the received frame.

When the UART3.MDR1_REG[6] SIP_MODE bit is set to 1, the TX state-machine always sends one SIP at the end of a transmission frame. However, when the SIP_MODE bit is set to 0, SIP transmission depends on the UART3.ACREG_REG[3] SEND_SIP bit.

The system (MPU) can set the SEND_SIP bit at least once every 500 ms. The advantage of this approach over the default approach is that the TX state-machine does not have to send the SIP at the end of each frame, which can reduce the overhead required.

17.4.4.2.6 IrDA Mode Interrupt Management

17.4.4.2.6.1 IrDA Interrupts

The IrDA function generates interrupts. All interrupts can be enabled/disabled by writing to the appropriate bit in the interrupt enable register (UART3.IER_REG). The interrupt status of the device can be checked at any time by reading the interrupt identification register (UART3.IIR_REG).

The UART, IrDA, and CIR modes have different interrupts in the UART/IrDA/CIR module and, therefore, different UART3.IER_REG and UART3.IIR_REG mappings, depending on the selected mode.

The IrDA modes have eight possible interrupts (see Table 17-35). The interrupt line is activated when any of the eight interrupts is generated (there is no priority).

Table 17-35. IrDA Mode Interrupts

IIR_REG Bit	Interrupt Type	Interrupt Source	Interrupt Reset Method
0	RHR interrupt	DRDY (data ready) (FIFO disable) RX FIFO above trigger level (FIFO enable)	Read RHR_REG until interrupt condition disappears.
1	THR interrupt	TFE (THR_REG empty) (FIFO disable) TX FIFO below trigger level (FIFO enable)	Write to THR_REG until interrupt condition disappears.
2	Last byte in RX FIFO	Last byte of frame in RX FIFO is available to be read at the RHR port.	Read RHR_REG.
3	RX overrun	Write to RHR_REG when RX FIFO full.	Read RESUME_REG register.
4	Status FIFO interrupt	Status FIFO triggers level reached.	Read STATUS FIFO.
5	TX status	1. THR_REG empty before EOF sent. Last bit of transmission of the IrDA frame occurred, but with an underrun error. OR 2. Transmission of the last bit of the IrDA frame completed successfully.	1. Read RESUME_REG register. OR 2. Read IIR_REG.
6	Receiver line status interrupt	CRC, ABORT, or frame-length error is written into STATUS FIFO.	Read STATUS FIFO (read until empty - maximum of eight reads required).
7	Received EOF	Received end-of-frame.	Read IIR_REG.

17.4.4.2.6.2 Wake-up Interrupts

The wake-up interrupt for the IrDA mode has the same functionality as that for the UART mode (see Section 17.4.4.1.4.1, *Wake-Up Interrupt*).

CAUTION

Wake-up interface implementation in this mode is based on the UARTi_SIDLEACK low-to-high transition instead of the UARTi_SIDLEACK state.

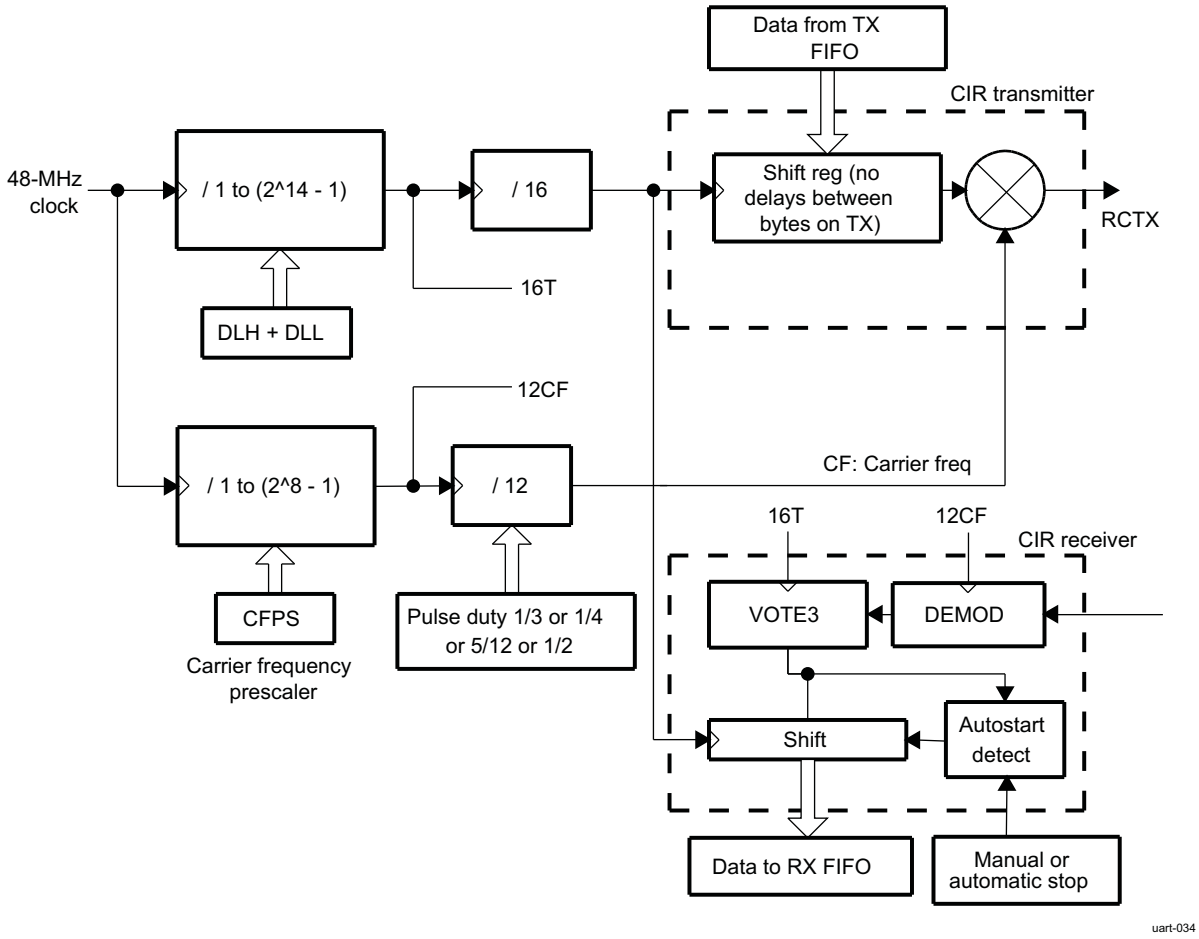
This does not ensure wake-up event generation as expected when configured in smart-idle mode and the system wakes up for a short period.

17.4.4.3 CIR Mode (UART3 Only)

17.4.4.3.1 CIR Mode Clock Generation

Depending on the encoding method (variable pulse distance/biphase), the MPU must develop a data structure that combines 1 and 0 with a t period to encode the complete frame to transmit. This can then be transmitted to the infrared output with a modulation method, as shown in Figure 17-33.

Figure 17-33. CIR Mode Block Components



Based on the requested modulation frequency, the UART3.CFPS_REG register must be set with the correct dividing value to provide the more accurate pulse frequency:

Dividing value = (FCLK/12)/MODfreq

Where:

FCLK = System clock frequency (48 MHz)

12 = Real value of baud multiple

MODfreq = Effective frequency of the modulation (MHz)

Example: For a targeted modulation frequency of 36 kHz, the CFPS_REG value must be set to 111 (decimal), which provides a modulation frequency of 36.04 kHz.

Note: The UART3.CFPS_REG register starts with a reset value of 105 (decimal), which translates to a frequency of 38.1 kHz.

The duty cycle of these pulses is user-defined by the pulse duty register bits in the UART3.MDR2_REG configuration register. Table 17-36 shows the duty cycle.

Table 17-36. Duty Cycle

MDR2_REG[5:4]	Duty Cycle (High Level)
00	1/4
01	1/3
10	5/12
11	1/2

17.4.4.3.2 CIR Data Formatting

The methods described in this section apply to all CIR modes.

17.4.4.3.2.1 IRRX Polarity Control

The IRRX polarity control for the CIR mode has the same functionality as that for the IrDA mode (see Section 17.4.4.2.3.1, *IRRX Polarity Control*).

17.4.4.3.2.2 CIR Transmission

In transmission, the MPU software must exercise an element of real-time control to transmit data packets, each of which must be emitted at a constant delay from the start bits of each individual packet. Thus, when sending a series of packets, the packet-to-packet delay must respect a specific delay. Two methods can be used to control this delay:

- Filling the TX FIFO with a number of zero bits that are transmitted with a *t* period
- Using an external system timer to control the delay either between each start-of-frame or between the end of a frame and the start of the next one. This can be performed by:
  - Controlling the start of the frame using the UART3.MDR1_REG[5] SCT bit and UART3.ACREG_REG[2] SCTX_EN bit, depending on the timer status
  - Using the UART3.IIR_REG[5] TX_STATUS_IT interrupt to preload the next frame in the TX FIFO and to control the start of the timer (in case of control delay between the end of a frame and the start of the next frame).

17.4.4.3.2.3 CIR Reception

There are two methods of stopping reception:

- The MPU can disable the reception by setting the UART3.ACREG_REG[5] DIS_IR_RX bit to 1 when it detects that the reception is complete because of the large number of 0s received. To receive a new frame, the DIS_IR_RX bit must be set to 0.
- The reception stops automatically, depending on the value set in the BOF-length register (the UART3.EBLR_REG[7:0] EBLR field). If the value set in the UART3.EBLR_REG[7:0] EBLR field is different from 0, this feature is enabled and counts the number of bits received at 0.

When the counter achieves the value defined in the EBLR_REG register, the reception automatically stops and the UART3.IIR_REG[2] RX_STOP_IT bit is set. When 1 is detected on the uart3.rx_irrx pin, the reception is automatically enabled.

CAUTION

If the UART3.IER_REG[2] RX_STOP_IT interrupt occurs before a byte boundary, the remaining bits of the last byte are filled with zeros and then passed into the RX FIFO.

17.4.4.3.3 CIR Mode Interrupt Management

17.4.4.3.3.1 CIR Interrupts

The CIR function generates interrupts that can be enabled/disabled by writing to the appropriate bit in the interrupt enable register (UART3.IER_REG). The interrupt status of the device can be checked at any time by reading the interrupt identification register (UART3.IIR_REG).

The UART, IrDA, and CIR modes have different interrupts in the UART/IrDA/CIR module and, therefore, different UART3.IER_REG and UART3.IIR_REG mappings, depending on the selected mode.

Table 17-37 lists the interrupt modes to be maintained. In CIR mode, the sole purpose of the UART3.IIR_REG[5] bit is to indicate that the last bit of infrared data was passed to the uart3_cts_rctx pin.

Table 17-37. CIR Mode Interrupts

IIR_REG Bit Number	Interrupt Type	Interrupt Source	Interrupt Reset Method
0	N/A for CIR mode	N/A for CIR mode	N/A for CIR mode
1	THR interrupt	TFE (THR_REG empty) (FIFO disable) TX FIFO below trigger level (FIFO enable)	Write to THR_REG until interrupt condition disappears.
2	N/A for CIR mode	N/A for CIR mode	N/A for CIR mode
3	N/A for CIR mode	N/A for CIR mode	N/A for CIR mode
4	N/A for CIR mode	N/A for CIR mode	N/A for CIR mode
5	TX status	Transmission of the last bit of the frame is completed successfully.	Read IIR_REG.
6	N/A for CIR mode	N/A for CIR mode	N/A for CIR mode
7	N/A for CIR mode	N/A for CIR mode	N/A for CIR mode

17.4.4.3.3.2 Wake-up Interrupts

The wake-up interrupt for the IrDA mode has the same functionality as that for the UART mode (see Section 17.4.4.1.4.1, *Wake-Up Interrupt*).

17.4.5 Power Management

17.4.5.1 UART Mode Power Management

17.4.5.1.1 Module Power Saving

In UART modes, sleep mode is enabled by setting the UARTi.IER_REG[4] SLEEP_MODE bit to 1 (when the UARTi.EFR_REG[4] ENHANCED_EN bit is set to 1).

Sleep mode is entered when all the following conditions exist:

- The serial data input line, uarti_rx, is idle.
- The TX FIFO and TX shift register are empty.
- The RX FIFO is empty.
- No interrupts are pending except THR interrupts.

Sleep mode is a good way to lower power consumption of the UART, but this state can be achieved only when the UART is set in modem mode. Therefore, even if the UART has no functional key role, it must be initialized in a functional mode to take advantage of sleep mode.

In sleep mode, the module clock and baud rate clock are stopped internally. Because most registers are clocked using these clocks, this greatly reduces power consumption. The module wakes up when a change is detected on the uarti_rx line, when data is written to the TX FIFO, and when there is a change in the state of the modem input pins.

An interrupt can be generated on a wake-up event by setting the UARTi.SCR_REG[4] RX_CTS_WU_EN bit to 1. See Section 17.4.4.1.4.1, *Wake-Up Interrupt*, to understand how to manage the interrupt.

Note: There must be no writing to the divisor latches, UARTi.DLL_REG and UARTi.DLH_REG, to set the baud clock, BCLK, while in sleep mode. It is advisable to disable sleep mode using the UARTi.IER_REG[4] SLEEP_MODE bit before writing to the UARTi.DLL_REG register or the UARTi.DLH_REG register.

17.4.5.1.2 System Power Saving

Sleep and auto-idle modes are embedded power-saving features. At the system level, power-reduction techniques can be applied by shutting down certain internal clock and power domains of the device.

The UART supports an idle req/idle ack handshaking protocol. This protocol is used at the system level to shut down clocks of the UART in a clean and controlled manner and to switch the UART from interrupt-generation mode to wake-up generation mode for unmasked events (see the UARTi.SYSC_REG[2] ENAWAKEUP bit and the UARTi.WER_REG register).

For more information, see Chapter 4, *Power, Reset, and Clock Management.*

17.4.5.2 IrDA Mode Power Management (UART3 Only)

17.4.5.2.1 Module Power Saving

In IrDA modes, sleep mode is enabled by setting the UART3.MDR[3] IR_SLEEP bit to 1.

Sleep mode is entered when all the following conditions exist:

- The serial data input line, uart3.rx_irrx, is idle.
- The TX FIFO and TX shift register are empty.
- The RX FIFO is empty.
- No interrupts are pending except THR interrupts.

The module wakes up when a change is detected on the uart3_rx_irrx line or when data is written to the TX FIFO.

17.4.5.2.2 System Power Saving

System power saving for the IrDA mode has the same functionality as that for the UART mode (see Section 17.4.5.1.2, *System Power Saving*).

17.4.5.3 CIR Mode Power Management (UART3 Only)

17.4.5.3.1 Module Power Saving

Module power saving for the CIR mode has the same functionality as that for the IrDA mode (see Section 17.4.5.2.1, *Module Power Saving*).

17.4.5.3.2 System Power Saving

System power saving for the CIR mode has the same functionality as that for the UART mode (see Section 17.4.5.2.2, *System Power Saving*).

17.5 UART/IrDA/CIR Basic Programming Model

17.5.1 UART Programming Model

17.5.1.1 Quick start

This section outlines the procedure for operating the UART module with FIFO and DMA or interrupts. This 3-part procedure ensures the quick start of the UART module. It does not cover every UART module feature.

The first programming model covers software reset of the module. The second programming model deals with FIFO and DMA configuration. The last programming model deals with protocol, baud rate and interrupt configuration.

Note: Each programming model can be used independently of the other two; for instance, reconfiguring the FIFOs and DMA settings only.

Each programming model can be executed starting from any UART register access mode (register modes, submodes, and other register dependencies). However, if the UART register access mode is known before executing the programming model, some steps that enable or restore register access are optional. For more information see Section 17.4.3.1, *Register Access Modes*.

17.5.1.1.1 Software Reset

To clear the UART registers, perform the following steps:

1. Initiate a software reset:
   Set the UARTi.SYSC_REG[1] SOFTRESET bit to 1.
2. Wait for the end of the reset operation:
   Poll the UARTi.SYSS_REG[0] RESETDONE bit until it equals 1.

17.5.1.1.2 FIFOs and DMA Settings

To enable and configure the receive and transmit FIFOs and program the DMA mode, perform the following steps:

1. Switch to register configuration mode B to access the UARTi.EFR_REG register:
   a. Save the current UARTi.LCR_REG value.
   b. Set UARTi.LCR_REG to 0x00BF.
2. Enable register submode TCR_TLR to access UARTi.TLR_REG (part 1 of 2):
   a. Save the UARTi.EFR_REG[4] ENHANCED_EN value.
   b. Set the UARTi.EFR_REG[4] ENHANCED_EN bit to 1.
3. Switch to register configuration mode A to access the UARTi.MCR_REG register:
   Set UARTi.LCR_REG to 0x0080.
4. Enable register submode TCR_TLR to access UARTi.TLR_REG (part 2 of 2):
   a. Save the UARTi.MCR_REG[6] TCR_TLR value.
   b. Set the UARTi.MCR_REG[6] TCR_TLR bit to 1.
5. Enable FIFO, load the new FIFO triggers (part 1 of 3) and the new DMA mode (part 1 of 2):
   Set the following bits to the desired values:
   - UARTi.FCR_REG[7:6] RX_FIFO_TRIG
   - UARTi.FCR_REG[5:4] TX_FIFO_TRIG
   - UARTi.FCR_REG[3] DMA_MODE
   - UARTi.FCR_REG[0] FIFO_ENABLE (0: Disable the FIFO/1: Enable the FIFO)

Note: The UARTi.FCR_REG register is not readable.

6. Switch to register configuration mode B to access the UARTi.EFR_REG register:
   Set UARTi.LCR_REG to 0x00BF.
7. Load the new FIFO triggers (part 2 of 3):
   Set the following bits to the desired values:
   - UARTi.TLR_REG[7:4] RX_FIFO_TRIG_DMA
   - UARTi.TLR_REG[3:0] TX_FIFO_TRIG_DMA
8. Load the new FIFO triggers (part 3 of 3) and the new DMA mode (part 2 of 2):
   Set the following bits to the desired values:
   - UARTi.SCR_REG[7] RX_TRIG_GRANU1
   - UARTi.SCR_REG[6] TX_TRIG_GRANU1
   - UARTi.SCR_REG[2:1] DMA_MODE_2
   - UARTi.SCR_REG[0] DMA_MODE_CTL
9. Restore the UARTi.EFR_REG[4] ENHANCED_EN value saved in Step 2a.
10. Switch to register configuration mode A to access the UARTi.MCR_REG register:
    Set UARTi.LCR_REG to 0x0080.
11. Restore the UARTi.MCR_REG[6] TCR_TLR value saved in Step 4a.
12. Restore the UARTi.LCR_REG value saved in Step 1a.

Triggers are used to generate interrupt and DMA requests. See Section 17.4.2.1.1, *Transmit FIFO Trigger,* to choose the following values:

- UARTi.FCR_REG[5:4] TX_FIFO_TRIG
- UARTi.TLR_REG[3:0] TX_FIFO_TRIG_DMA
- UARTi.SCR_REG[6] TX_TRIG_GRANU1

Triggers are used to generate interrupt and DMA requests. See Section 17.4.2.1.2, *Receive FIFO Trigger*, to choose the following values:

- UARTi.FCR_REG[7:6] RX_FIFO_TRIG
- UARTi.TLR_REG[7:4] RX_FIFO_TRIG_DMA
- UARTi.SCR_REG[7] RX_TRIG_GRANU1

DMA mode enables the different DMA requests. See Section 17.4.2.4, *FIFO DMA Mode Operation*, to choose the following values:

- UARTi.FCR_REG[3] DMA_MODE
- UARTi.SCR_REG[2:1] DMA_MODE_2
- UARTi.SCR_REG[0] DMA_MODE_CTL

17.5.1.1.3 Protocol, Baud Rate, and Interrupt Settings

To program the protocol, baud rate and interrupt settings, perform the following steps:

1. Disable UART to access UARTi.DLL_REG and UARTi.DLH_REG:
   Set UARTi.MDR1_REG[2:0] MODE_SELECT to 0x7.
2. Switch to register configuration mode B to access the UARTi.EFR_REG register:
   Set UARTi.LCR_REG to 0x00BF.
3. Enable access to UARTi.IER_REG[7:4]:
   a. Save the UARTi.EFR_REG[4] ENHANCED_EN value.
   b. Set the UARTi.EFR_REG[4] ENHANCED_EN bit to 1.
4. Switch to register operational mode to access the UARTi.IER_REG register:
   Set UARTi.LCR_REG to 0x0000.
5. Clear the UARTi.IER_REG (UARTi.IER_REG[4] SLEEP_MODE bit to 0 to change UARTi.DLL_REG and UARTi.DLH_REG).Set UARTi.IER_REG to 0x0000.

6. Switch to register configuration mode B to access the UARTi.DLL_REG and UARTi.DLH_REG registers:
   Set UARTi.LCR_REG to 0x00BF.
7. Load the new divisor value:
   Set the UARTi.DLL_REG[7:0] CLOCK_LSB and UARTi.DLH_REG[5:0] CLOCK_MSB fields to the desired value.
8. Switch to register operational mode to access the UARTi.IER_REG register:
   Set UARTi.LCR_REG to 0x0000.
9. Load the new interrupt configuration.(0: Disable the interrupt/1: Enable the interrupt):
   Set the following bits to the desired values:
   - UARTi.IER_REG[7] CTS_IT
   - UARTi.IER_REG[6] RTS_IT
   - UARTi.IER_REG[5] XOFF_IT
   - UARTi.IER_REG[4] SLEEP_MODE
   - UARTi.IER_REG[3] MODEM_STS_IT
   - UARTi.IER_REG[2] LINE_STS_IT
   - UARTi.IER_REG[1] THR_IT
   - UARTi.IER_REG[0] RHR_IT
10. Switch to register configuration mode B to access the UARTi.EFR_REG register:
    Set UARTi.LCR_REG to 0x00BF.
11. Restore the UARTi.EFR_REG[4] ENHANCED_EN value saved in Step 3a.
12. Load the new protocol formatting (parity, stop bit, char length) and switch to register operational mode:
    Set UARTi.LCR_REG[7] DIV_EN to 0.
    Set UARTi.LCR_REG[6] BREAK_EN to 0.
    Set the following bits to the desired values:
    - UARTi.LCR_REG[5] PARITY_TYPE_2
    - UARTi.LCR_REG[4] PARITY_TYPE_1
    - UARTi.LCR_REG[3] PARITY_EN
    - UARTi.LCR_REG[2] NB_STOP
    - UARTi.LCR_REG[1:0] CHAR_LENGTH
13. Load the new UART mode:
    Set UARTi.MDR1_REG[2:0] MODE_SELECT to the desired value.

See Section 17.4.4.1.2, *Choosing the Appropriate Divisor Value*, to choose the following values:

- UARTi.DLL_REG[7:0] CLOCK_LSB
- UARTi.DLH_REG[5:0] CLOCK_MSB
- UARTi.MDR1_REG[2:0] MODE_SELECT

See Section 17.4.4.1.3.1, *Frame Formatting*, to choose the following values:

- UARTi.LCR_REG[5] PARITY_TYPE_2
- UARTi.LCR_REG[4] PARITY_TYPE_1
- UARTi.LCR_REG[3] PARITY_EN
- UARTi.LCR_REG[2] NB_STOP
- UARTi.LCR_REG[1:0] CHAR_LENGTH

17.5.1.2 Hardware and Software Flow Control Configuration

This section outlines the programming steps to enable and configure hardware and software flow control. Hardware and software flow control cannot be used at the same time.

Note: Each programming model can be executed starting from any UART register access mode (register modes, submodes, and other register dependencies). However, if the UART register access mode is known before executing the programming model, some steps that enable or restore register access are optional. For more information, see Section 17.4.3.1, *Register Access Modes*.

17.5.1.2.1 Hardware Flow Control Configuration

To enable and configure hardware flow control, perform the following procedure:

1. Switch to register configuration mode A to access the UARTi.MCR_REG register:
   a. Save the current UARTi.LCR_REG.
   b. Set UARTi.LCR_REG to 0x0080.
2. Enable register submode TCR_TLR to access UARTi.TCR_REG (part 1 of 2):
   a. Save the UARTi.MCR_REG[6] TCR_TLR value.
   b. Set UARTi.MCR_REG[6] TCR_TLR = 1.
3. Switch to register configuration mode B to access the UARTi.EFR_REG register:
   Set UARTi.LCR_REG to 0x00BF.
4. Enable register submode TCR_TLR to access the UARTi.TCR_REG register (part 2 of 2):
   a. Save the UARTi.EFR_REG[4] ENHANCED_EN value.
   b. Set the UARTi.EFR_REG[4] ENHANCED_EN bit to 1.
5. Load the new start and halt trigger values for hardware flow control:
   Set the following bits to the desired values:
   - UARTi.TCR_REG[7:4] AUTO_RTS_START
   - UARTi.TCR_REG[3:0] AUTO_RTS_HALT
6. Enable or disable receive and transmit hardware flow control mode and restore the UARTi.EFR_REG[4] ENHANCED_EN value saved in Step 4a.
   Set the following bits to the desired values:
   - UARTi.EFR_REG[7] AUTO_CTS_EN (0: Disable/1: Enable)
   - UARTi.EFR_REG[6] AUTO_RTS_EN (0: Disable/1: Enable)

   Restore UARTi.EFR_REG[4] ENHANCED_EN to the saved value.
7. Switch to register configuration mode A to access UARTi.MCR_REG:
   Set UARTi.LCR_REG to 0x0080.
8. Restore the UARTi.MCR_REG[6] TCR_TLR value saved in Step 2a.
9. Restore the UARTi.LCR_REG value saved in Step 1a.

See Section 17.4.4.1.3.2, *Hardware Flow Control*, to choose the following values:

- UARTi.EFR_REG[7] AUTO_CTS_EN
- UARTi.EFR_REG[6] AUTO_RTS_EN
- UARTi.TCR_REG[7:4] AUTO_RTS_START
- UARTi.TCR_REG[3:0] AUTO_RTS_HALT

17.5.1.2.2 Software Flow Control Configuration

To enable and configure software flow control, perform the following procedure:

1. Switch to register configuration mode B to access the UARTi.EFR_REG register.
   a. Save the current UARTi.LCR_REG.
   b. Set UARTi.LCR_REG to 0x00BF.
2. Enable register submode XOFF to access the UARTi.XOFF1_REG and UARTi.XOFF2_REG registers:
   a. Save the UARTi.EFR_REG[4] ENHANCED_EN value.
   b. Set the UARTi.EFR_REG[4] ENHANCED_EN bit to 0.

3. Load the new software flow control characters:
   Set the following bits to the desired values:
   - UARTi.XON1_ADDR1_REG[7:0] XON_WORD1
   - UARTi.XON2_ADDR2_REG[7:0] XON_WORD2
   - UARTi.XOFF1_REG[7:0] XOFF_WORD1
   - UARTi.XOFF2_REG[7:0] XOFF_WORD2
4. Enable access to UARTi.MCR_REG[7:5] and enable register submode TCR_TLR to access UARTi.TCR_REG (part 1 of 2):
   Set the UARTi.EFR_REG[4] ENHANCED_EN bit to 1.
5. Switch to register configuration mode A to access the UARTi.MCR_REG register:
   Set UARTi.LCR_REG to 0x0080.
6. Enable register submode TCR_TLR to access UARTi.TCR_REG (part 2 of 2) and enable or disable XON any function:
   a. Save the UARTi.MCR_REG[6] TCR_TLR value.
   b. Set the UARTi.MCR_REG[6] TCR_TLR to 1.
      Set UARTi.MCR_REG[5] XON_EN to the desired value (0: Disable/1: Enable).
7. Switch to register configuration mode B to access the UARTi.EFR_REG register:
   Set UARTi.LCR_REG to 0x00BF.
8. Load the new start and halt trigger values for software flow control:
   Set the following bits to the desired values:
   - UARTi.TCR_REG[7:4] AUTO_RTS_START
   - UARTi.TCR_REG[3:0] AUTO_RTS_HALT
9. Enable or disable special char function and load the new software flow control mode and restore the UARTi.EFR_REG[4] ENHANCED_EN value saved in Step 2a:
   Set the following bits to the desired values:
   - UARTi.EFR_REG[5] SPEC_CHAR (0: Disable/1: Enable)
   - UARTi.EFR_REG[3:0] SW_FLOW_CONTROL

   Restore UARTi.EFR_REG[4] ENHANCED_EN to the saved value.
10. Switch to register configuration mode A to access the UARTi.MCR_REG register:
    Set UARTi.LCR_REG to 0x0080.
11. Restore the UARTi.MCR_REG[6] TCR_TLR value saved in Step 6a.
12. Restore the UARTi.LCR_REG value saved in Step 1a.

See Section 17.4.4.1.3.3, *Software Flow Control*, to choose the following values:

- UARTi.EFR_REG[5] SPEC_CHAR
- UARTi.EFR_REG[3:0] SW_FLOW_CONTROL
- UARTi.TCR_REG[7:4] AUTO_RTS_START
- UARTi.TCR_REG[3:0] AUTO_RTS_HALT
- UARTi.XON1_ADDR1_REG[7:0] XON_WORD1
- UARTi.XON2_ADDR2_REG[7:0] XON_WORD2
- UARTi.XOFF1_REG[7:0] XOFF_WORD1
- UARTi.XOFF2_REG[7:0] XOFF_WORD2

17.5.2 IrDA Programming Model (UART3 Only)

17.5.2.1 SIR Mode

17.5.2.1.1 Receive

The following programming model explains how to program the module to receive IrDA frame with parity forced to 1, baud rate = 112.5 Kbs, FIFOs disable, 2 stop bits, 8-bit word length.

1. **Disable UART before accessing UARTi.DLL_REG and UARTi.DLH_REG:**
   Set UARTi.MDR1_REG[2:0] MODE_SELECT to 0x7.
2. **Grant access to DLL_REG and DLH_REG (LCR_REG[7] DIV_EN = 0x1**
   UART3.LCR_REG = 0x80 (Note: Data format is unaffected by the use and settings of the UART3.LCR_REG register in IrDA mode.)
3. **Load the new baud rate (115.2Kbs)**
   UART3.DLL_REG = 0x1A
   UART3.DLH_REG = 0x00
4. **Set SIR Mode**
   UART3.MDR1_REG[2:0] MODE_SELECT = 0x1
5. **Disable access to DLL_REG and DLH_REG and switch to register operational mode**
   UART3.LCR_REG = 0x00.
6. **Optional: Enable RHR interrupt**
   UART3.IER_REG[0] RHR_IT = 0x1

17.5.2.1.2 Transmit

The following programming model explains how to program the module to transmit IrDA 6 bytes frame with no parity, baud rate = 112.5Kbs, FIFOs disable, 3/16 encoding, 2 stop bits, 7 bits word length

1. **Disable UART before accessing UARTi.DLL_REG and UARTi.DLH_REG:**
   Set MDR1_REG[2:0] MODE_SELECT = 0x7).
2. **Grant access to EFR_REG**
   UART3.LCR_REG = 0xBF
3. **Enable the enhanced features (EFR_REG[4] ENAHNCED_EN = 0x1)**
   UART3.EFR_REG = 0x10.
4. **Grant access to DLL_REG and DLH_REG (LCR_REG[7] DIV_EN = 0x1)**
   UART3.LCR_REG = 0x80 (Note: Data format is unaffected by the use and settings of the UART3.LCR_REG register in IrDA mode).
5. **Load the new baud rate (115.2Kbs)**
   UART3.DLL_REG = 0x1A
   UART3.DLH_REG = 0x00
6. **Set SIR Mode** (MDR1_REG[2:0] MODE_SELECT = 0x1)
   UART3.MDR1_REG = 0x01.
7. **Disable access to DLL_REG and DLH_REG and switch to register operational mode UART3.LCR_REG**
   UART3.LCR_REG = 0x00.
8. **Force DTR output to active**
   UART3.MCR_REG[0] DTR = 0x1
9. **Optional: Enable THR interrupt**
   UART3.IER_REG[1] THR_IT = 0x1
10. **Set transmit frame length to 6 bytes**
    UART3.TXFLL_REG = 0x06
11. **Set 7 starts of frame transmission**
    UART3.EBLR_REG = 0x08
12. **Optional: Set SIR pulse width to be 1.6 µs**
    UART3.ACREG_REG[7] PULSE_TYPE = 0x1
13. **Load THR_REG with the desired data to be transmitted**

17.5.2.2 MIR Mode

17.5.2.2.1 Receive

The following programming model explains how to program the module to receive IrDA frame with no parity, baud rate = 1.152Mbs, FIFOs disable

1. **Disable UART before accessing UARTi.DLL_REG and UARTi.DLH_REG**
   Set UARTi.MDR1_REG[2:0] MODE_SELECT to 0x7.
2. **Grant access to DLL_REG and DLH_REG (LCR_REG[7] DIV_EN = 0x1)**
   UART3.LCR_REG = 0x80 (Note: Data format is unaffected by the use and settings of the UART3.LCR_REG register in IrDA mode).
3. **Load the new baud rate (1.152 Mbs)**
   UART3.DLL_REG = 0x01
   UART3.DLH_REG = 0x00.
4. **Set MIR Mode**
   UART3.MDR1_REG[2:0] MODE_SELECT = 0x4
5. **Disable access to DLL_REG and DLH_REG and switch to register operational mode**
   UART3.LCR_REG = 0x00.
6. **Force DTR output to active (MCR_REG[0] DTR = 0x1)**
   Force RTS output to active (MCR_REG[1] RTS = 0x1)
   UART3.MCR_REG = 0x3
7. **Optional: Enable RHR interrupt**
   UART3.IER_REG[0] RHR_IT = 0x1

17.5.2.2.2 Transmit

The following programming model explains how to program the module to transmit IrDA 60 bytes frame with no parity, baud rate = 1.152Mbs, FIFOs disable

1. **Disable UART before accessing UARTi.DLL_REG and UARTi.DLH_REG:**
   Set UARTi.MDR1_REG[2:0] MODE_SELECT to 0x7.
2. **Grant access to DLL_REG and DLH_REG (LCR_REG[7] DIV_EN = 0x1)**
   UART3.LCR_REG = 0x80 (Note: Data format is unaffected by the use and settings of the UART3.LCR_REG register in IrDA mode.)
3. **Load the new baud rate (1.152 Mbs)**
   UART3.DLL_REG = 0x01
   UART3.DLH_REG = 0x00
4. **Set MIR Mode**
   UART3.MDR1_REG[2:0] MODE_SELECT = 0x4.
5. **Disable access to DLL_REG and DLH_REG and switch to register operational mode:**
   UART3.LCR_REG = 0x00.
6. **Force DTR output to active**
   UART3.MCR_REG[0] DTR = 0x1
7. **Optional: Enable THR interrupt**
   UART3.IER_REG[1] THR_IT = 0x1
8. **Set frame length to 60 bytes**
   UART3.TXFLL_REG = 0x3C
9. **Optional: Transmit 8 additional starts of frame (MIR mode requires 2 starts anyway)**
   UART3.EBLR_REG = 0x08
10. **SIP will be send at the end of transmission**
    UART3.ACREG_REG[3] = 0x1
11. **Load THR_REG with the desired data to be transmitted**

17.5.2.2.3 FIR Mode

17.5.2.2.3.1 Receive

The following programming model explains how to program the module to receive IrDA frame with no parity, baud rate = 4M bits/s, FIFOs enable, 8-bit word length.

1. Disable UART before accessing UARTi.DLL_REG and UARTi.DLH_REG: Set UARTi.MDR1_REG[2:0] MODE_SELECT to 0x7.
2. Grant access to DLL_REG and DLH_REG (LCR_REG[7] DIV_EN = 0x1 UART3.LCR_REG = 0x80 (Note: Data format is unaffected by the use and settings of the UART3.LCR_REG register in IrDA mode.)
3. FIFO clear and enable FCR_REG = 0x7 (Tx/Rx FIFO trigger : FCR_REG[7:6] & FCR_REG[5:4]) LCR_REG[7]=0.
4. Set FIR Mode UART3.MDR1_REG[2:0] MODE_SELECT = 0x5.
5. Set frame length RXFLL_REG = 0xA (Data + CRC + STO).
6. Disable access to DLL_REG and DLH_REG and switch to register operational mode UART3.LCR_REG[7] DIV_EN = 0x0.
7. Optional: Enable RHR interrupt UART3.IER_REG[0] RHR_IT = 0x1.

17.5.2.2.3.2 Transmit

The following programming model explains how to program the module to transmit IrDA 4 bytes frame with no parity, baud rate = 4M bits/s, FIFOs enable, 8-bit word length.

1. Disable UART before accessing UARTi.DLL_REG and UARTi.DLH_REG: Set MDR1_REG[2:0] MODE_SELECT = 0x7).
2. Grant access to EFR_REG UART3.LCR_REG = 0xBF.
3. Enable the enhanced features (EFR_REG[4] ENAHNCED_EN = 0x1) UART3.EFR_REG = 0x10.
4. FIFO clear and enable FCR_REG = 0x7 (Tx/Rx FIFO trigger: FCR_REG[7:6] and FCR_REG[5:4]) LCR_REG[7] = 0
5. Set FIR mode and enable auto-SIP mode MDR1_REG = 0x45.
6. Set Frame Length TXFLL_REG = 0x4 TXFLH_REG = 0x0 RXFLL_REG = 0xA (Data + CRC + STO) RXFLH_REG = 0x0.
7. Force DTR output to active UART3.MCR_REG[0] DTR = 0x1.
8. Optional: Enable THR interrupt UART3.IER_REG[1] THR_IT = 0x1.
9. Load THR_REG with the desired data to be transmitted.

17.6 UART/IrDA/CIR Register Manual

17.6.1 UART/IrDA/CIR Instance Summary

Table 17-38 shows the base address and address space for the UART/IrDA/CIR module instances.

Table 17-38. UART/IrDA/CIR Instance Summary

Module Name	Base Address	Size
UART1[(1)]	0x4806 A000	1K byte
UART2[(1)]	0x4806 C000	1K byte
UART3[(2)]	0x4902 0000	1K byte

(1) UART only
(2) UART, IrDA and CIR

17.6.2 UART/IrDA/CIR Register Summary

Note: UART/IrDA/CIR registers access depends on access mode and submode. For more information, see Section 17.4.3.1, *Register Access Modes*.

Table 17-39. UART/IrDA/CIR Register Summary

Register Name	Type	Register Width (Bits)	Address Offset	UART1 Physical Address	UART2 Physical Address	UART3 Physical Address
DLL_REG	RW	8	0x000	0x4806 A000	0x4806 C000	0x4902 0000
RHR_REG	R	8	0x000	0x4806 A000	0x4806 C000	0x4902 0000
THR_REG	W	8	0x000	0x4806 A000	0x4806 C000	0x4902 0000
DLH_REG	RW	8	0x004	0x4806 A004	0x4806 C004	0x4902 0004
IER_REG	RW	8	0x004	0x4806 A004	0x4806 C004	0x4902 0004
IIR_REG	R	8	0x008	0x4806 A008	0x4806 C008	0x4902 0008
FCR_REG	W	8	0x008	0x4806 A008	0x4806 C008	0x4902 0008
EFR_REG	RW	8	0x008	0x4806 A008	0x4806 C008	0x4902 0008
LCR_REG	RW	8	0x00C	0x4806 A00C	0x4806 C00C	0x4902 000C
MCR_REG	RW	8	0x010	0x4806 A010	0x4806 C010	0x4902 0010
XON1_ADDR1_REG	RW	8	0x010	0x4806 A010	0x4806 C010	0x4902 0010
LSR_REG	R	8	0x014	0x4806 A014	0x4806 C014	0x4902 0014
XON2_ADDR2_REG	RW	8	0x014	0x4806 A014	0x4806 C014	0x4902 0014
MSR_REG	R	8	0x018	0x4806 A018	0x4806 C018	0x4902 0018
TCR_REG	RW	8	0x018	0x4806 A018	0x4806 C018	0x4902 0018
XOFF1_REG	RW	8	0x018	0x4806 A018	0x4806 C018	0x4902 0018
SPR_REG	RW	8	0x01C	0x4806 A01C	0x4806 C01C	0x4902 001C
TLR_REG	RW	8	0x01C	0x4806 A01C	0x4806 C01C	0x4902 001C
XOFF2_REG	RW	8	0x01C	0x4806 A01C	0x4806 C01C	0x4902 001C
MDR1_REG	RW	8	0x020	0x4806 A020	0x4806 C020	0x4902 0020
MDR2_REG	RW	8	0x024	0x4806 A024	0x4806 C024	0x4902 0024
SFLSR_REG	R	8	0x028	N/A	N/A	0x4902 0028
TXFLL_REG	W	8	0x028	N/A	N/A	0x4902 0028
RESUME_REG	R	8	0x02C	N/A	N/A	0x4902 002C
TXFLH_REG	W	8	0x02C	N/A	N/A	0x4902 002C

Table 17-39. UART/IrDA/CIR Register Summary (continued)

Register Name	Type	Register Width (Bits)	Address Offset	UART1 Physical Address	UART2 Physical Address	UART3 Physical Address
SFREGL_REG	R	8	0x030	N/A	N/A	0x4902 0030
RXFLL_REG	W	8	0x030	N/A	N/A	0x4902 0030
SFREGH_REG	R	8	0x034	N/A	N/A	0x4902 0034
RXFLH_REG	W	8	0x034	N/A	N/A	0x4902 0034
UASR_REG	R	8	0x038	0x4806 A038	0x4806 C038	0x4902 0038
BLR_REG	RW	8	0x038	N/A	N/A	0x4902 0038
ACREG_REG	RW	8	0x03C	N/A	N/A	0x4902 003C
SCR_REG	RW	8	0x040	0x4806 A040	0x4806 C040	0x4902 0040
SSR_REG	R	8	0x044	0x4806 A044	0x4806 C044	0x4902 0044
EBLR_REG	RW	8	0x048	N/A	N/A	0x4902 0048
MVR_REG	R	8	0x050	0x4806 A050	0x4806 C050	0x4902 0050
SYSC_REG	RW	8	0x054	0x4806 A054	0x4806 C054	0x4902 0054
SYSS_REG	R	8	0x058	0x4806 A058	0x4806 C058	0x4902 0058
WER_REG	RW	8	0x05C	0x4806 A05C	0x4806 C05C	0x4902 005C
CFPS_REG	RW	8	0x060	N/A	N/A	0x4902 0060

17.6.3 UART/IrDA/CIR Register Description

Table 17-40. DLL_REG

Address Offset	0x000
Physical Address	Please refer to Table 17-39
Description	Divisor latches low This register, with DLH_REG, stores the 14-bit divisor for generation of the baud clock in the baud rate generator. DLH_REG stores the most-significant part of the divisor. DLL_REG stores the least-significant part of the divisor. **Note:** DLL_REG and DLH_REG can be written to only before sleep mode is enabled (before IER_REG[4] is set).
Type	RW

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								CLOCK_LSB							

Bits	Field Name	Description	Type	Reset
15:8	Reserved	Read returns 0x00.	R	0x00
7:0	CLOCK_LSB	Stores the 8-bit LSB divisor value	RW	0x00

Table 17-41. Register Call Summary for Register DLL_REG

UART/IrDA/CIR Functional Description
- FIFO DMA Mode Operation: [0]
- Register Access Modes: [1] [2] [3] [4]
- UART/IrDA (SIR, MIR, FIR)/CIR Mode Selection: [5] [6] [7] [8] [9] [10] [11] [12] [13] [14] [15] [16]
- UART Mode: [17] [18] [19]
- IrDA Mode (UART3 Only): [20]
- UART Mode Power Management: [21] [22]

UART/IrDA/CIR Basic Programming Model
- Quick start: [23] [24] [25] [26] [27]
- SIR Mode: [28] [29] [30] [31] [32] [33] [34] [35]
- MIR Mode: [36] [37] [38] [39] [40] [41] [42] [43] [44] [45] [46] [47]

UART/IrDA/CIR Register Manual
- UART/IrDA/CIR Register Summary: [48]
- UART/IrDA/CIR Register Description: [49] [50] [51] [52] [53] [54] [55] [56] [57]

Table 17-42. RHR_REG

Address Offset	0x000
Physical Address	Please refer to Table 17-39
Description	Receive holding register The receiver section consists of the receiver holding register (RHR_REG) and the receiver shift register. The RHR_REG is actually a 64-byte FIFO. The receiver shift register receives serial data from RX input. The data is converted to parallel data and moved to the RHR_REG. If the FIFO is disabled, location zero of the FIFO is used to store the single data character. Note: If an overflow occurs the data in the RHR_REG is not overwritten.
Type	R

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								RHR							

Bits	Field Name	Description	Type	Reset
15:8	Reserved	Read returns 0x00.	R	0x00
7:0	RHR	Receive holding register	R	0x–

Table 17-43. Register Call Summary for Register RHR_REG

UART/IrDA/CIR Functional Description

- FIFO Management: [0] [1]
- Register Access Modes: [2]
- UART/IrDA (SIR, MIR, FIR)/CIR Mode Selection: [3] [4]
- UART Mode: [5] [6] [7] [8] [9] [10] [11]
- IrDA Mode (UART3 Only): [12] [13] [14]

UART/IrDA/CIR Register Manual

- UART/IrDA/CIR Register Summary: [15]
- UART/IrDA/CIR Register Description: [16] [17] [18] [19] [20] [21]

Table 17-44. THR_REG

Address Offset	0x000
Physical Address	Please refer to Table 17-39
Description	Transmit holding register The transmitter section consists of the transmit holding register (THR_REG) and the transmit shift register. The transmit holding register is a 64-byte FIFO. The MPU writes data to the THR_REG. The data is placed in the transmit shift register where it is shifted out serially on the TX output. If the FIFO is disabled, location zero of the FIFO is used to store the data.
Type	W

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								THR							

Bits	Field Name	Description	Type	Reset
15:8	Reserved	Write has no functional effect.	W	0x00
7:0	THR	Transmit holding register	W	0x–

Table 17-45. Register Call Summary for Register THR_REG

UART/IrDA/CIR Functional Description

- FIFO Management: [0]
- FIFO DMA Mode Operation: [1] [2]
- Register Access Modes: [3]
- UART/IrDA (SIR, MIR, FIR)/CIR Mode Selection: [4] [5] [6]
- UART Mode: [7] [8]
- IrDA Mode (UART3 Only): [9] [10] [11]
- CIR Mode (UART3 Only): [12] [13]

UART/IrDA/CIR Basic Programming Model

- SIR Mode: [14]
- MIR Mode: [15] [16]

UART/IrDA/CIR Register Manual

- UART/IrDA/CIR Register Summary: [17]
- UART/IrDA/CIR Register Description: [18] [19] [20] [21]

Table 17-46. IER_REG

Address Offset	0x004
Physical Address	Please refer to Table 17-39
Description	Interrupt enable register
Type	RW

UART Bitfield Details

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								CTS_IT	RTS_IT	XOFF_IT	SLEEP_MODE	MODEM_STS_IT	LINE_STS_IT	THR_IT	RHR_IT

Bits	Field Name	Description	Type	Reset
15:8	Reserved	Read returns 0x00. Write has no functional effect.	RW	0x00
7	CTS_IT	Can be written only when EFR_REG[4] = 1	RW	0
		0x0: Disables the nCTS interrupt		
		0x1: Enables the nCTS interrupt		
6	RTS_IT	Can be written only when EFR_REG[4] = 1	RW	0
		0x0: Disables the interrupt		
		0x1: Enables the nRTS interrupt		
5	XOFF_IT	Can be written only when EFR_REG[4] = 1	RW	0
		0x0: Disables the XOFF interrupt		
		0x1: Enables the XOFF interrupt		
4	SLEEP_MODE	Can be only written when EFR_REG[4] = 1	RW	0
		0x0: Disables sleep mode		
		0x1: Enables sleep mode (stop baud rate clock when the module is inactive)		
3	MODEM_STS_IT		RW	0
		0x0: Disables the modem status register interrupt		
		0x1: Enables the modem status register interrupt		
2	LINE_STS_IT		RW	0
		0x0: Disables the receiver line status interrupt		
		0x1: Enables the receiver line status interrupt		
1	THR_IT		RW	0
		0x0: Disables the THR interrupt		
		0x1: Enables the THR interrupt		
0	RHR_IT		RW	0
		0x0: Disables the RHR interrupt and time out interrupt.		
		0x1: Enables the RHR interrupt and time out interrupt.		

Table 17-47. Register Call Summary for Register IER_REG

UART/IrDA/CIR Functional Description

- FIFO Interrupt Mode: [0]
- FIFO Polled Mode Operation: [1]
- Register Access Modes: [2] [3]
- UART/IrDA (SIR, MIR, FIR)/CIR Mode Selection: [4] [5] [6] [7] [8] [9]
- UART Mode: [10]
- IrDA Mode (UART3 Only): [11] [12]
- CIR Mode (UART3 Only): [13] [14] [15]
- UART Mode Power Management: [16] [17]

UART/IrDA/CIR Basic Programming Model

- Quick start: [18] [19] [20] [21] [22] [23] [24] [25] [26] [27] [28] [29] [30] [31]
- SIR Mode: [32] [33]
- MIR Mode: [34] [35] [36] [37]

UART/IrDA/CIR Register Manual

- UART/IrDA/CIR Register Summary: [38]
- UART/IrDA/CIR Register Description: [39] [40] [41] [42] [43]

CIR Bitfield Details

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								Reserved		TX_STATUS_IT	Reserved	Reserved	Reserved	THR_IT	Reserved

Bits	Field Name	Description	Type	Reset
15:8	Reserved	Read returns 0x00. Write has no functional effect.	RW	0x00
7:6	Reserved	Read returns 0x0. Write has no functional effect.	RW	0x0
5	TX_STATUS_IT	In IR-CIR mode, contrary to the IR-IrDA mode, the TX_STATUS_IT has only one meaning corresponding to the case MDR2_REG[0] = 0.	RW	0
4	Reserved	Read returns 0. Write has no functional effect.	RW	0
3	Reserved	Reserved	R	0
2	Reserved	Reserved	R	0
1	THR_IT	0x0: Disables the THR interrupt 0x1: Enables the THR interrupt	RW	0
0	Reserved	Reserved	R	0

IrDA Bitfield Details

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								EOF_IT	LINE_STS_IT_I	TX_STATUS_IT	STS_FIFO_ TRIG_IT	RX_OVERRUN_ IT	LAST_RX_ BYTE_IT	THR_IT	RHR_IT

Bits	Field Name	Description	Type	Reset
15:8	Reserved	Read returns 0. Write has no functional effect.	RW	0x00
7	EOF_IT			
		0x0: Disables the received EOF interrupt		
		0x1: Enables the received EOF interrupt		
6	LINE_STS_IT_I		RW	0
		0x0: Disables the receiver line status interrupt		
		0x1: Enables the receiver line status interrupt		
5	TX_STATUS_IT	TX_STATUS_IT interrupt reflects two possible conditions. The MDR2_REG[0] must be read to determine the status in the event of this interrupt.	RW	0
		0x0: Disables the TX status interrupt		
		0x1: Enables the TX status interrupt		
4	STS_FIFO_ TRIG_IT		RW	0
		0x0: Disables status FIFO trigger level interrupt		
		0x1: Enables status FIFO trigger level interrupt.		
3	RX_OVERRUN_ IT		RW	0
		0x0: Disables the RX overrun interrupt		
		0x1: Enables the RX overrun interrupt		
2	LAST_RX_ BYTE_IT		RW	0
		0x0: Disables the last byte of frame in RX FIFO interrupt		
		0x1: Enables the last byte of frame in RX FIFO interrupt		
1	THR_IT		RW	0
		0x0: Disables the THR interrupt		
		0x1: Enables the THR interrupt		
0	RHR_IT		RW	0
		0x0: Disables the RHR interrupt		
		0x1: Enables the RHR interrupt		

Table 17-48. DLH_REG

Address Offset	0x004
Physical Address	Please refer to Table 17-39
Description	Divisor latches high
	This register, with DLL_REG, stores the 14-bit divisor for generation of the baud clock in the baud rate generator. DLH_REG stores the most-significant part of the divisor. DLL_REG stores the least-significant part of the divisor.
	Note: DLL_REG and DLH_REG can be written to only before sleep mode is enabled (before IER_REG[4] is set).
Type	RW

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								Reserved		CLOCK_MSB					

Bits	Field Name	Description	Type	Reset
15:8	Reserved	Read returns 0x00.	R	0x00
7:6	Reserved	Read returns 0x0.	R	0x0
5:0	CLOCK_MSB	Stores the 6-bit most-significant bit (MSB) divisor value	RW	0x00

Table 17-49. Register Call Summary for Register DLH_REG

UART/IrDA/CIR Functional Description

- FIFO DMA Mode Operation: [0]
- Register Access Modes: [1] [2] [3] [4]
- UART/IrDA (SIR, MIR, FIR)/CIR Mode Selection: [5] [6] [7] [8] [9] [10] [11] [12] [13] [14] [15] [16]
- UART Mode: [17] [18] [19]
- IrDA Mode (UART3 Only): [20]
- UART Mode Power Management: [21] [22]

UART/IrDA/CIR Basic Programming Model

- Quick start: [23] [24] [25] [26] [27]
- SIR Mode: [28] [29] [30] [31] [32] [33] [34] [35]
- MIR Mode: [36] [37] [38] [39] [40] [41] [42] [43] [44] [45] [46] [47]

UART/IrDA/CIR Register Manual

- UART/IrDA/CIR Register Summary: [48]
- UART/IrDA/CIR Register Description: [49] [50] [51] [52] [53] [54] [55] [56] [57]

Table 17-50. FCR_REG

Address Offset	0x008
Physical Address	Please refer to Table 17-39
Description	FIFO control register
Type	W

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								RX_FIFO_TRIG		TX_FIFO_TRIG		DMA_MODE	TX_FIFO_CLEAR	RX_FIFO_CLEAR	FIFO_EN

Bits	Field Name	Description	Type	Reset
15:8	Reserved	Write has no functional effect.	W	0x00
7:6	RX_FIFO_TRIG	Sets the trigger level for the RX FIFO: If SCR_REG[7] = 0 and TLR_REG[7:4] ≠ 0000, RX_FIFO_TRIG is not considered. If SCR_REG[7] = 1, RX_FIFO_TRIG is 2 LSB of the trigger level (1 to 63 on 6 bits) with a granularity of 1. If SCR_REG[7] = 0 and TLR_REG[7:4] = 0000: 0x0: 8 characters 0x1: 16 characters 0x2: 56 characters 0x3: 60 characters	W	0x0
5:4	TX_FIFO_TRIG	Can be written only if EFR_REG[4] = 1. Sets the trigger level for the TX FIFO. If SCR_REG[6] = 0 and TLR_REG[3:0] ≠ 0000, TX_FIFO_TRIG is not considered. If SCR_REG[6] = 1, RX_FIFO_TRIG is 2 LSB of the trigger level (1 to 63 on 6 bits) with a granularity of 1. If SCR_REG[6] = 0 and TLR_REG[3:0] = 0000: 0x0: 8 characters 0x1: 16 characters 0x2: 32 characters 0x3: 56 characters	W	0x0
3	DMA_MODE	Can be changed only when the baud clock is not running (DLL_REG and DLH_REG set to 0).	W	0

Bits	Field Name	Description	Type	Reset
		This register is considered if SCR_REG[0] = 0. 0x0: DMA_MODE 0 (No DMA) 0x1: DMA_MODE 1 (UART_NDMA_REQ[0] in TX, UART_NDMA_REQ[1] in RX)		
2	TX_FIFO_CLEAR	0x0: No change 0x1: Clears the transmit FIFO and resets its counter logic to 0. Returns to 0 after clearing FIFO.	W	0
1	RX_FIFO_CLEAR	0x0: No change 0x1: Clears the receive FIFO and resets its counter logic to 0. Returns to 0 after clearing FIFO.	W	0
0	FIFO_EN	Can be changed only when the baud clock is not running (DLL_REG and DLH_REG set to 0). 0x0: Disables the transmit and receive FIFOs. The transmit and receive holding registers are 1-byte FIFOs. 0x1: Enables the transmit and receive FIFOs. The transmit and receive holding registers are 64-byte FIFOs.	W	0

Table 17-51. Register Call Summary for Register FCR_REG

UART/IrDA/CIR Functional Description

- FIFO Management: [0]
- FIFO Trigger: [1] [2] [3]
- FIFO Interrupt Mode: [4] [5] [6]
- FIFO Polled Mode Operation: [7]
- FIFO DMA Mode Operation: [8] [9] [10] [11]
- Register Access Modes: [12] [13]
- UART/IrDA (SIR, MIR, FIR)/CIR Mode Selection: [14] [15] [16] [17] [18] [19]

UART/IrDA/CIR Basic Programming Model

- Quick start: [20] [21] [22] [23] [24] [25] [26] [27]
- MIR Mode: [28] [29] [30] [31] [32] [33]

UART/IrDA/CIR Register Manual

- UART/IrDA/CIR Register Summary: [34]
- UART/IrDA/CIR Register Description: [35] [36] [37] [38] [39]

Table 17-52. IIR_REG

Address Offset	0x008
Physical Address	Please refer to Table 17-39
Description	Interrupt Identification register
	The IIR_REG is a read-only register that provides the source of the interrupt in a prioritized manner.
	Note: An interrupt source can be flagged only if enabled in the IER_REG register.
Type	R

UART Bitfield Details

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								FCR_MIRROR		IT_TYPE					IT_PENDING

Bits	Field Name	Description	Type	Reset
15:8	Reserved	Read returns 0x00.	R	0x00
7:6	FCR_MIRROR	Mirror the contents of FCR_REG[0] on both bits.	R	0x0
5:1	IT_TYPE	Seven possible interrupts in UART mode; other combinations never occur: 0x0: Modem interrupt. Priority = 4 0x1: THR interrupt. Priority = 3 0x2: RHR interrupt. Priority = 2 0x3: Receiver line status error. Priority = 1 0x6: Rx timeout. Priority = 2 0x8: Xoff/special character. Priority = 5 0x10: CTS, RTS change state from active (low) to inactive (high). Priority = 6	R	0x00
0	IT_PENDING	0x0: An interrupt is pending. 0x1: No interrupt is pending.	R	1

Table 17-53. Register Call Summary for Register IIR_REG

UART/IrDA/CIR Functional Description

- Register Access Modes: [0] [1]
- UART/IrDA (SIR, MIR, FIR)/CIR Mode Selection: [2] [3] [4] [5] [6] [7]
- UART Mode: [8] [9] [10] [11] [12] [13] [14] [15] [16] [17] [18] [19] [20]
- IrDA Mode (UART3 Only): [21] [22] [23] [24] [25]
- CIR Mode (UART3 Only): [26] [27] [28] [29] [30] [31] [32]

UART/IrDA/CIR Register Manual

- UART/IrDA/CIR Register Summary: [33]
- UART/IrDA/CIR Register Description: [34] [35] [36] [37] [38]

CIR Bitfield Details

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								Reserved		TX_STATUS_IT	Reserved	Reserved	Reserved	THR_IT	Reserved

Bits	Field Name	Description	Type	Reset
15:8	Reserved	Read returns 0x00.	R	0x00
7:6	Reserved	Read returns 0x0.	R	0x0
5	TX_STATUS_IT	0x0: TX status interrupt inactive 0x1: TX status interrupt active	R	0
4	Reserved	Read returns 0	R	0
3	Reserved	Reserved	R	0
2	Reserved	Reserved	R	0
1	THR_IT	0x0: THR interrupt inactive 0x1: THR interrupt active	R	0
0	Reserved	Reserved	R	0

IrDA Bitfield Details

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								EOF_IT	LINE_STS_IT	TX_STATUS_IT	STS_FIFO_IT	RX_OE_IT	RX_FIFO_LB_IT	THR_IT	RHR_IT

Bits	Field Name	Description		Type	Reset
15:8	Reserved	Read returns 0x00.		R	0x00
7	EOF_IT				
		0x0:	Received EOF interrupt inactive		
		0x1:	Received EOF interrupt active		
6	LINE_STS_IT			R	0
		0x0:	Receiver line status interrupt inactive		
		0x1:	Receiver line status interrupt active		
5	TX_STATUS_IT			R	0
		0x0:	TX status interrupt inactive		
		0x1:	TX status interrupt active		
4	STS_FIFO_IT			R	0
		0x0:	Status FIFO trigger level interrupt inactive		
		0x1:	Status FIFO trigger level interrupt active		
3	RX_OE_IT			R	0
		0x0:	RX overrun interrupt inactive		
		0x1:	RX overrun interrupt active		
2	RX_FIFO_LB_IT		Receive FIFO last byte interrupt	R	0
		0x0:	Last byte of frame in RX FIFO interrupt inactive		
		0x1:	Last byte of frame in RX FIFO interrupt active		
1	THR_IT			R	0
		0x0:	THR interrupt inactive		
		0x1:	THR interrupt active		
0	RHR_IT			R	0
		0x0:	RHR interrupt inactive		
		0x1:	RHR interrupt active		

Table 17-54. EFR_REG

Address Offset	0x008
Physical Address	Please refer to Table 17-39
Description	Enhanced feature register
	This register enables or disables enhanced features. Most enhanced functions apply only to UART modes, but EFR_REG[4] enables write accesses to FCR_REG[5:4], the TX trigger level, which is also used in IrDA modes.
Type	RW

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								AUTO_CTS_EN	AUTO_RTS_EN	SPEC_CHAR	ENHANCED_EN	SW_FLOW_CONTROL			

Bits	Field Name	Description	Type	Reset
15:8	Reserved	Read returns 0x00.	R	0x00
7	AUTO_CTS_EN	Auto-CTS enable bit (UART mode only) 0x0: Normal operation 0x1: Auto-CTS flow control is enabled; transmission is halted when the nCTS pin is high (inactive).	RW	0
6	AUTO_RTS_EN	Auto-RTS enable bit (UART mode only) 0x0: Normal operation 0x1: Auto-RTS flow control is enabled; nRTS pin goes high (inactive) when the receiver FIFO HALT trigger level, TCR_REG[3:0], is reached and goes low (active) when the receiver FIFO RESTORE transmission trigger level is reached.	RW	0
5	SPEC_CHAR	(UART mode only) Special character detect 0x0: Normal operation 0x1: Special character detect enable. Received data is compared with XOFF2 data. If a match occurs, the received data is transferred to RX FIFO and IIR_REG bit 4 is set to 1 to indicate that a special character was detected.	RW	0
4	ENHANCED_EN	Enhanced functions write enable bit 0x0: Disables writing to IER_REG bits [7:4], FCR_REG bits [5:4], and MCR_REG bits [7:5] 0x1: Enables writing to IER_REG bits [7:4], FCR_REG bits [5:4], and MCR_REG bits [7:5]	RW	0
3:0	SW_FLOW_ CONTROL	Combinations of software flow control can be selected by programming bit [3:0]. See Table 17-32. In IrDA mode, bits [1:0] select IR address to check. See Section 17.2.5.2.1.7, *IR Address Checking*.	RW	0x0

Table 17-55. Register Call Summary for Register EFR_REG

UART/IrDA/CIR Environment

- IrDA Protocol and Data Format: [0] [1] [2]

UART/IrDA/CIR Functional Description

- Register Access Modes: [3] [4] [5] [6] [7] [8] [9] [10]
- UART/IrDA (SIR, MIR, FIR)/CIR Mode Selection: [11] [12] [13] [14] [15] [16]
- UART Mode: [17] [18] [19] [20] [21] [22]
- IrDA Mode (UART3 Only): [23] [24] [25]
- UART Mode Power Management: [26]

UART/IrDA/CIR Basic Programming Model

- Quick start: [27] [28] [29] [30] [31] [32] [33] [34] [35] [36]
- Hardware and Software Flow Control Configuration: [37] [38] [39] [40] [41] [42] [43] [44] [45] [46] [47] [48] [49] [50] [51] [52] [53] [54] [55] [56]
- SIR Mode: [57] [58] [59]
- MIR Mode: [60] [61] [62]

UART/IrDA/CIR Register Manual

- UART/IrDA/CIR Register Summary: [63]
- UART/IrDA/CIR Register Description: [64] [65] [66] [67] [68] [69] [70] [71]

Table 17-56. LCR_REG

Address Offset	0x00C
Physical Address	Please refer to Table 17-39
Description	Line control register
	LCR_REG[6:0] define parameters of the transmission and reception for UART mode. LCR_REG[7] is used to put the module in operational mode or Configuration_Mode_A/B.
Type	RW

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								DIV_EN	BREAK_EN	PARITY_TYPE2	PARITY_TYPE1	PARITY_EN	NB_STOP	CHAR_LENGTH	

Bits	Field Name	Description	Type	Reset
15:8	Reserved	Read returns 0x00	R	0x00
7	DIV_EN		RW	0
		0x0: Operational mode		
		0x1: Divisor latch enable; put the module in Configuration_Mode_A/B. Allows access to DLL_REG, DLH_REG, and other registers (see Section 17.6.2, *UART/IrDA/CIR Register Mapping Summary*). Configuration_Mode_B: LCR_REG[7:0] = 0xBF Else, Configuration_Mode_A		
6	BREAK_EN	Break control bit. UART mode only.	RW	0
		Note: When LCR_REG[6] is set to 1, the TX line is forced to 0 and remains in this state as long as LCR_REG[6] = 1.		
		0x0: Normal operating condition		
		0x1: Forces the transmitter output to go low to alert the communication terminal. TX line is forced to 0 and remains in this state while BREAK_EN = 1.		
5	PARITY_TYPE2	Selects the forced parity format (if LCR_REG[3] = 1). UART mode only. If LCR_REG[5] = 1and LCR_REG[4] = 0, the parity bit is forced to 1 in the transmitted and received data. If LCR_REG[5] = 1 and LCR_REG[4] = 1, the parity bit is forced to 0 in the transmitted and received data.	RW	0
4	PARITY_TYPE1	UART mode only	RW	0
		0x0: Odd parity is generated (if LCR_REG[3] = 1).		
		0x1: Even parity is generated (if LCR_REG[3] = 1).		
3	PARITY_EN	UART mode only	RW	0
		0x0: No parity		
		0x1: A parity bit is generated during transmission, and the receiver checks for received parity.		
2	NB_STOP	Specifies the number of stop bits. UART mode only.	RW	0
		0x0: 1 stop bit (word length = 5, 6, 7, 8)		
		0x1: 1.5 stop bits (word length = 5)1 - 2 stop bits (word length = 6, 7, 8)		
1:0	CHAR_LENGTH	Specifies the word length to be transmitted or received. UART mode only.	RW	0x0
		0x0: 5 bits		
		0x1: 6 bits		
		0x2: 7 bits		
		0x3: 8 bits		

Table 17-57. Register Call Summary for Register LCR_REG

UART/IrDA/CIR Environment

- IrDA Protocol and Data Format: [0]

UART/IrDA/CIR Functional Description

- Register Access Modes: [1] [2] [3] [4] [5] [6] [7] [8] [9] [10] [11] [12]
- UART/IrDA (SIR, MIR, FIR)/CIR Mode Selection: [13] [14] [15] [16] [17] [18] [19] [20] [21] [22] [23] [24] [25] [26] [27] [28] [29] [30]
- UART Mode: [31] [32] [33] [34] [35] [36] [37]
- IrDA Mode (UART3 Only): [38] [39] [40]

UART/IrDA/CIR Basic Programming Model

- Quick start: [41] [42] [43] [44] [45] [46] [47] [48] [49] [50] [51] [52] [53] [54] [55] [56] [57] [58] [59] [60] [61] [62] [63]
- Hardware and Software Flow Control Configuration: [64] [65] [66] [67] [68] [69] [70] [71] [72] [73] [74]
- SIR Mode: [75] [76] [77] [78] [79] [80] [81] [82] [83] [84]
- MIR Mode: [85] [86] [87] [88] [89] [90] [91] [92] [93] [94] [95] [96] [97] [98] [99]

UART/IrDA/CIR Register Manual

- UART/IrDA/CIR Register Summary: [100]
- UART/IrDA/CIR Register Description: [101] [102] [103] [104] [105] [106] [107] [108] [109] [110] [111] [112] [113]

Table 17-58. MCR_REG

Address Offset	0x010
Physical Address	Please refer to Table 17-39
Description	Modem control register MCR_REG[3:0] controls the interface with the modem, data set, or peripheral device that is emulating the modem.
Type	RW

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								Reserved	TCR_TLR	XON_EN	LOOPBACK_EN	CD_STS_CH	RI_STS_CH	RTS	DTR

Bits	Field Name	Description	Type	Reset
15:8	Reserved	Read returns 0x00.	R	0x00
7	Reserved	Read returns 0. Write has no functional effect.	RW	0
6	TCR_TLR	Can be written only when EFR_REG[4] ENHANCED_EN = 1 0x0: No action 0x1: Enables access to the TCR_REG and TLR_REG registers	RW	0
5	XON_EN	Can be written only when EFR_REG[4] ENHANCED_EN = 1 0x0: Disable XON any function 0x1: Enable XON any function	RW	0
4	LOOPBACK_EN	0x0: Normal operating mode 0x1: Enable local loopback mode (internal). In this mode, the MCR_REG[3:0] signals are looped back into MSR_REG[7:4]. The transmit output is looped back to the receive input internally.	RW	0
3	CD_STS_CH	0x0: In loopback, forces nDCD input high and IRQ outputs to INACTIVE state.	RW	0

Bits	Field Name	Description	Type	Reset
		0x1: In loopback, forces nDCD input low and IRQ outputs to INACTIVE state.		
2	RI_STS_CH		RW	0
		0x0: In loopback, forces nRI input inactive (high).		
		0x1: In loopback, forces nRI input active (low).		
1	RTS	In loop back, controls MSR_REG[4].If auto-RTS is enabled, the nRTS output is controlled by hardware flow control.	RW	0
		0x0: Force nRTS output to inactive (high).		
		0x1: Force nRTS output to active (low).		
0	DTR	0: Force DTR output (used in loop back mode) to inactive (high)	RW	0
		1: Force DTR output (used in loop back mode) to active (low)		

Table 17-59. Register Call Summary for Register MCR_REG

UART/IrDA/CIR Environment
- UART Interface Description: [0] [1]

UART/IrDA/CIR Functional Description
- Register Access Modes: [2] [3] [4] [5] [6] [7] [8] [9] [10] [11]
- UART/IrDA (SIR, MIR, FIR)/CIR Mode Selection: [12] [13] [14] [15] [16] [17]
- UART Mode: [18] [19]

UART/IrDA/CIR Basic Programming Model
- Quick start: [20] [21] [22] [23] [24]
- Hardware and Software Flow Control Configuration: [25] [26] [27] [28] [29] [30] [31] [32] [33] [34] [35] [36]
- SIR Mode: [37]
- MIR Mode: [38] [39] [40] [41] [42]

UART/IrDA/CIR Register Manual
- UART/IrDA/CIR Register Summary: [43]
- UART/IrDA/CIR Register Description: [44] [45] [46] [47] [48] [49] [50] [51] [52] [53] [54] [55]

Table 17-60. XON1_ADDR1_REG

Address Offset	0x010
Physical Address	Please refer to Table 17-39
Description	UART mode: XON1 character, IrDA mode: ADDR1 address
Type	RW

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								XON_WORD1							

Bits	Field Name	Description	Type	Reset
15:8	Reserved	Read returns 0x00	R	0x00
7:0	XON_WORD1	Used to store the 8-bit XON1 character in UART modes and ADDR1 address 1 for IrDA modes	RW	0x00

Table 17-61. Register Call Summary for Register XON1_ADDR1_REG

UART/IrDA/CIR Environment
- IrDA Protocol and Data Format: [0]

UART/IrDA/CIR Functional Description
- Register Access Modes: [1] [2]
- UART/IrDA (SIR, MIR, FIR)/CIR Mode Selection: [3] [4] [5] [6]
- IrDA Mode (UART3 Only): [7]

UART/IrDA/CIR Basic Programming Model
- Hardware and Software Flow Control Configuration: [8] [9]

Table 17-61. Register Call Summary for Register XON1_ADDR1_REG (continued)

UART/IrDA/CIR Register Manual

- UART/IrDA/CIR Register Summary: [10]

Table 17-62. LSR_REG

Address Offset	0x014
Physical Address	Please refer to Table 17-39
Description	Line status register
Type	R

UART Bitfield Details

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								RX_FIFO_STS	TX_SR_E	TX_FIFO_E	RX_BI	RX_FE	RX_PE	RX_OE	RX_FIFO_E

Bits	Field Name	Description		Type	Reset
15:8	Reserved	Read returns 0x00.		R	0x00
7	RX_FIFO_STS			R	0
		0x0:	Normal operation		
		0x1:	At least one parity error, framing error, or break indication in the RX FIFO. Bit 7 is cleared when no errors are present in the RX FIFO.		
6	TX_SR_E			R	1
		0x0:	Transmitter hold (TX FIFO) and shift registers are not empty.		
		0x1:	Transmitter hold (TX FIFO) and shift registers are empty		
5	TX_FIFO_E			R	1
		0x0:	Transmit hold register (TX FIFO) is not empty.		
		0x1:	Transmit hold register (TX FIFO) is empty. The transmission is not necessarily complete.		
4	RX_BI			R	0
		0x0:	No break condition		
		0x1:	A break was detected while the data being read from the RX FIFO was being received (RX input was low for one character + 1 bit time frame).		
3	RX_FE			R	0
		0x0:	No framing error in data being read from RX FIFO		
		0x1:	Framing error occurred in data being read from RX FIFO (received data did not have a valid stop bit).		
2	RX_PE			R	0
		0x0:	No parity error in data being read from RX FIFO		
		0x1:	Parity error in data being read from RX FIFO		
1	RX_OE			R	0
		0x0:	No overrun error		
		0x1:	Overrun error occurred. Set when the character held in the receive shift register is not transferred to the RX FIFO. This case occurs only when receive FIFO is full.		
0	RX_FIFO_E			R	0

Bits	Field Name	Description		Type	Reset
		0x0:	No data in the receive FIFO		
		0x1:	At least one data character in the RX FIFO		

Table 17-63. Register Call Summary for Register LSR_REG

UART/IrDA/CIR Environment

- IrDA Protocol and Data Format: [0] [1] [2]

UART/IrDA/CIR Functional Description

- FIFO Polled Mode Operation: [3]
- Register Access Modes: [4] [5]
- UART/IrDA (SIR, MIR, FIR)/CIR Mode Selection: [6] [7] [8] [9] [10] [11]
- UART Mode: [12] [13] [14] [15] [16] [17] [18] [19] [20]
- IrDA Mode (UART3 Only): [21] [22] [23]

UART/IrDA/CIR Register Manual

- UART/IrDA/CIR Register Summary: [24]
- UART/IrDA/CIR Register Description: [25]

CIR Bitfield Details

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								THR_EMPTY	Reserved	Reserved	Reserved				Reserved

Bits	Field Name	Description		Type	Reset
15:8	Reserved	Read returns 0x00.		R	0x00
7	THR_EMPTY			R	0
		0x0:	Transmit holding register (TX FIFO) is not empty.		
		0x1:	Transmit hold register (TX FIFO) is empty. The transmission is not necessarily complete.		
6	Reserved				
5	Reserved	Reserved		R	1
4:1	Reserved				
0	Reserved	Reserved		R	1

IrDA Bitfield Details

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								THR_EMPTY	STS_FIFO_FUL	RX_LAST_BYTE	FRAME_TOO_LONG	ABORT	CRC	STS_FIFO_E	RX_FIFO_E

Bits	Field Name	Description		Type	Reset
15:8	Reserved	Read returns 0x00.		R	0x00
7	THR_EMPTY			R	0
		0x0:	Transmit holding register (TX FIFO) is not empty.		
		0x1:	Transmit hold register (TX FIFO) is empty. The transmission is not necessarily complete.		
6	STS_FIFO_FUL			R	1

Bits	Field Name	Description	Type	Reset
		0x0: Status FIFO not full		
		0x1: Status FIFO full		
5	RX_LAST_BYTE	Receive last byte	R	1
		0x0: The RX FIFO (RHR_REG) does not contain the last byte of the frame to be read.		
		0x1: The RX FIFO (RHR_REG) contains the last byte of the frame to be read. This bit is set only when the last byte of a frame is available to be read. It is used to determine the frame boundary. It is cleared on a single read of the LSR_REG register.		
4	FRAME_TOO_LONG	Frame too long	R	0
		0x0: No frame-too-long error in frame		
		0x1: Frame-too-long error in the frame at the top of the STATUS FIFO (next character to be read). This bit is set to 1 when a frame exceeding the maximum length (set by RXFLH_REG and RXFLL_REG registers) is received. When this error is detected, current frame reception is terminated. Reception is stopped until the next START flag is detected.		
3	ABORT		R	0
		0x0: No abort pattern error in frame		
		0x1: Abort pattern received. SIR and MIR: abort pattern. FIR: Illegal symbol.		
2	CRC		R	0
		0x0: No CRC error in frame		
		0x1: CRC error in the frame at the top of the STATUS FIFO (next character to be read)		
1	STS_FIFO_E		R	0
		0x0: Status FIFO not empty		
		0x1: Status FIFO empty		
0	RX_FIFO_E		R	1
		0x0: At least one data character in the RX FIFO		
		0x1: No data in the receive FIFO		

Table 17-64. XON2_ADDR2_REG

Address Offset	0x014
Physical Address	Please refer to Table 17-39
Description	
Type	RW

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								XON_WORD2							

Bits	Field Name	Description	Type	Reset
15:8	Reserved	Read returns 0x00.	R	0x00
7:0	XON_WORD2	Stores the 8-bit XON2 character in UART modes and ADDR2 address 2 for IrDA modes	RW	0x00

Table 17-65. Register Call Summary for Register XON2_ADDR2_REG

UART/IrDA/CIR Environment

- IrDA Protocol and Data Format: [0]

Table 17-65. Register Call Summary for Register XON2_ADDR2_REG (continued)

UART/IrDA/CIR Functional Description • Register Access Modes: [1] [2] • UART/IrDA (SIR, MIR, FIR)/CIR Mode Selection: [3] [4] [5] [6] • IrDA Mode (UART3 Only): [7]
UART/IrDA/CIR Basic Programming Model • Hardware and Software Flow Control Configuration: [8] [9]
UART/IrDA/CIR Register Manual • UART/IrDA/CIR Register Summary: [10]

Table 17-66. XOFF1_REG

Address Offset	0x018
Physical Address	Please refer to Table 17-39
Description	UART mode XOFF1 character
Type	RW

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								XOFF_WORD1							

Bits	Field Name	Description	Type	Reset
15:8	Reserved	Read returns 0x00	R	0x00
7:0	XOFF_WORD1	Stores the 8-bit XOFF1 character used in UART modes	RW	0x00

Table 17-67. Register Call Summary for Register XOFF1_REG

UART/IrDA/CIR Functional Description • Register Access Modes: [0] [1] • UART/IrDA (SIR, MIR, FIR)/CIR Mode Selection: [2] [3]
UART/IrDA/CIR Basic Programming Model • Hardware and Software Flow Control Configuration: [4] [5] [6]
UART/IrDA/CIR Register Manual • UART/IrDA/CIR Register Summary: [7]

Table 17-68. TCR_REG

Address Offset	0x018
Physical Address	Please refer to Table 17-39
Description	Transmission control register. This register stores the receive FIFO threshold levels to start/stop transmission during hardware flow control.
Type	RW

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								RX_FIFO_TRIG_ START				RX_FIFO_TRIG_ HALT			

Bits	Field Name	Description	Type	Reset
15:8	Reserved	Read returns 0x00.	R	0x00
7:4	RX_FIFO_TRIG_ START	RX FIFO trigger level to RESTORE transmission (0 to 60)	RW	0x0
3:0	RX_FIFO_TRIG_ HALT	RX FIFO trigger level to HALT transmission (0 to 60)	RW	0xF

Table 17-69. Register Call Summary for Register TCR_REG

UART/IrDA/CIR Functional Description

- FIFO Trigger: [0] [1] [2]
- FIFO Interrupt Mode: [3]
- Register Access Modes: [4] [5] [6] [7] [8] [9]
- UART/IrDA (SIR, MIR, FIR)/CIR Mode Selection: [10] [11] [12] [13] [14] [15] [16] [17] [18] [19] [20] [21] [22] [23] [24] [25] [26] [27]
- UART Mode: [28] [29] [30] [31] [32] [33]

UART/IrDA/CIR Basic Programming Model

- Hardware and Software Flow Control Configuration: [34] [35] [36] [37] [38] [39] [40] [41] [42] [43] [44] [45]

UART/IrDA/CIR Register Manual

- UART/IrDA/CIR Register Summary: [46]
- UART/IrDA/CIR Register Description: [47] [48]

Table 17-70. MSR_REG

Address Offset	0x018
Physical Address	Please refer to Table 17-39
Description	Modem status register. UART mode only. This register provides information about the current state of the control lines from the modem, data set, or peripheral device to the MPU. It also indicates when a control input from the modem changes state.
Type	R

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								NCD_STS	NRI_STS	NDSR_STS	NCTS_STS	DCD_STS	RI_STS	DSR_STS	CTS_STS

Bits	Field Name	Description	Type	Reset
15:8	Reserved	Read returns 0x00.	R	0x00
7	NCD_STS	In loopback mode, it is equivalent to MCR_REG[3].	R	-
6	NRI_STS	This bit is the complement of the nRI input. In loopback mode, it is equivalent to MCR_REG[2].	R	-
5	NDSR_STS	In loopback mode, it is equivalent to MCR_REG[0].	R	-
4	NCTS_STS	This bit is the complement of the nCTS input. In loopback mode, it is equivalent to MCR_REG[1].	R	-
3	DCD_STS	Indicates that MCR_REG[3] in loopback changed. Cleared on a read.	R	0
2	RI_STS	Indicates that nRI input (or MCR_REG[2] in loopback) changed state from low to high. Cleared on a read.	R	0
1	DSR_STS	0x1: Indicates that MCR_REG[0] in loopback changed state. Cleared on a read.	R	0
0	CTS_STS	0x1: Indicates that nCTS input (or MCR_REG[1] in loopback) changed state. Cleared on a read.	R	0

Table 17-71. Register Call Summary for Register MSR_REG

UART/IrDA/CIR Environment • UART Interface Description: [0] [1]
UART/IrDA/CIR Functional Description • Register Access Modes: [2] [3] • UART/IrDA (SIR, MIR, FIR)/CIR Mode Selection: [4] [5] [6] [7] [8] [9] • UART Mode: [10] [11]
UART/IrDA/CIR Register Manual • UART/IrDA/CIR Register Summary: [12] • UART/IrDA/CIR Register Description: [13] [14]

Table 17-72. SPR_REG

Address Offset	0x01C
Physical Address	Please refer to Table 17-39
Description	Scratchpad register This read/write register does not control the module. It is a scratchpad register used by the programmer to hold temporary data.
Type	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								SPR_WORD							

Bits	Field Name	Description	Type	Reset
15:8	Reserved	Read returns 0x00.	R	0x00
7:0	SPR_WORD	Scratchpad register	RW	0x00

Table 17-73. Register Call Summary for Register SPR_REG

UART/IrDA/CIR Functional Description • Register Access Modes: [0] [1] [2] [3] • UART/IrDA (SIR, MIR, FIR)/CIR Mode Selection: [4] [5] [6] [7] [8] [9] [10] [11] [12] [13] [14] [15]
UART/IrDA/CIR Register Manual • UART/IrDA/CIR Register Summary: [16]

Table 17-74. XOFF2_REG

Address Offset	0x01C
Physical Address	Please refer to Table 17-39
Description	UART mode XOFF2 character.
Type	RW

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								XOFF_WORD2							

Bits	Field Name	Description	Type	Reset
15:8	Reserved	Read returns 0x00.	R	0x00
7:0	XOFF_WORD2	Stores the 8-bit XOFF2 character used in UART modes	RW	0x00

Table 17-75. Register Call Summary for Register XOFF2_REG

UART/IrDA/CIR Functional Description • Register Access Modes: [0] [1] • UART/IrDA (SIR, MIR, FIR)/CIR Mode Selection: [2] [3]
UART/IrDA/CIR Basic Programming Model • Hardware and Software Flow Control Configuration: [4] [5] [6]

Table 17-75. Register Call Summary for Register XOFF2_REG (continued)

UART/IrDA/CIR Register Manual • UART/IrDA/CIR Register Summary: [7]

Table 17-76. TLR_REG

Address Offset	0x01C
Physical Address	Please refer to Table 17-39
Description	Trigger level register. Stores the programmable transmit and receive FIFO trigger levels used for DMA and IRQ generation
Type	RW

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								RX_FIFO_TRIG_ DMA				TX_FIFO_TRIG_ DMA			

Bits	Field Name	Description	Type	Reset
15:8	Reserved	Read returns 0x00.	R	0x00
7:4	RX_FIFO_TRIG_ DMA	Receive FIFO trigger level	RW	0x0
3:0	TX_FIFO_TRIG_ DMA	Transmit FIFO trigger level	RW	0x0

Table 17-77. Register Call Summary for Register TLR_REG

UART/IrDA/CIR Functional Description • FIFO Management: [0] • FIFO Trigger: [1] [2] [3] [4] • FIFO Interrupt Mode: [5] [6] • FIFO DMA Mode Operation: [7] [8] [9] [10] • Register Access Modes: [11] [12] [13] [14] [15] [16] • UART/IrDA (SIR, MIR, FIR)/CIR Mode Selection: [17] [18] [19] [20] [21] [22] [23] [24] [25] [26] [27] [28] [29] [30] [31] [32] [33] [34]
UART/IrDA/CIR Basic Programming Model • Quick start: [35] [36] [37] [38] [39] [40]
UART/IrDA/CIR Register Manual • UART/IrDA/CIR Register Summary: [41] • UART/IrDA/CIR Register Description: [42] [43] [44] [45] [46]

Table 17-78. MDR1_REG

Address Offset	0x020
Physical Address	Please refer to Table 17-39
Description	Mode definition register 1. The mode of operation can be programmed by writing to MDR1_REG[2:0]; therefore, the MDR1_REG must be programmed on startup after configuration of the configuration registers (DLL_REG, DLH_REG, LCR_REG). The value of MDR1_REG[2:0] must not be changed again during normal operation.
Type	RW

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								FRAME_END_M ODE	SIP_MODE	SCT	SET_TXIR	IR_SLEEP	MODE_SELECT		

Bits	Field Name	Description	Type	Reset
15:8	Reserved	Read returns 0x00.	R	0x00
7	FRAME_END_ MODE	IrDA mode only	RW	0
		0x0: Frame-length method		
		0x1: Set EOT bit method		
6	SIP_MODE	MIR/FIR modes only. IrDA only.	RW	0
		0x0: Manual SIP mode: SIP is generated with the control of ACREG_REG[3].		
		0x1: Automatic SIP mode: SIP is generated after each transmission.		
5	SCT	Store and control the transmission.	RW	0
		0x0: Starts the infrared transmission when a value is written to THR_REG		
		0x1: Starts the infrared transmission with the control of ACREG_REG[2]		
		Note: Before starting any transmission, there must be no reception ongoing.		
4	SET_TXIR	Used to configure the infrared transceiver. IrDA only.	RW	0
		0x0: No action		
		0x1: IRTX pin output is forced high.		
3	IR_SLEEP		RW	0
		0x0: IrDA/CIR sleep mode disabled		
		0x1: IrDA/CIR sleep mode enabled		
2:0	MODE_SELECT	UART-IrDA-CIR mode selection	RW	0x7
		0x0: UART 16x mode		
		0x1: SIR mode		
		0x2: UART 16x auto-baud		
		0x3: UART 13x mode		
		0x4: MIR mode		
		0x5: FIR mode		
		0x6: CIR mode		
		0x7: Disable (default state)		

Table 17-79. Register Call Summary for Register MDR1_REG

UART/IrDA/CIR Overview • UART/IrDA/CIR Overview: [0]
UART/IrDA/CIR Environment • UART Interface Description: [1] [2] • UART Protocol and Data Format: [3]

Table 17-79. Register Call Summary for Register MDR1_REG (continued)

UART/IrDA/CIR Functional Description

- Register Access Modes: [4] [5] [6] [7] [8] [9]
- UART/IrDA (SIR, MIR, FIR)/CIR Mode Selection: [10] [11] [12] [13] [14] [15] [16] [17] [18] [19] [20] [21] [22] [23] [24] [25] [26] [27] [28]
- UART Mode: [29] [30] [31]
- IrDA Mode (UART3 Only): [32] [33] [34] [35] [36] [37]
- CIR Mode (UART3 Only): [38]

UART/IrDA/CIR Basic Programming Model

- Quick start: [39] [40] [41]
- SIR Mode: [42] [43] [44] [45] [46]
- MIR Mode: [47] [48] [49] [50] [51] [52] [53] [54]

UART/IrDA/CIR Register Manual

- UART/IrDA/CIR Register Summary: [55]
- UART/IrDA/CIR Register Description: [56] [57] [58] [59] [60]

Table 17-80. MDR2_REG

Address Offset	0x024
Physical Address	Please refer to Table 17-39
Description	Mode definition register 2
	IR-IrDA and IR-CIR modes only
	MDR2_REG[0] describes the status of the interrupt in IIR_REG[5]. The IRTX_UNDERRUN bit must be read after an IIR_REG[5] TX_STATUS_IT interrupt occurs. The bits [2:1] of this register set the trigger level for the frame status FIFO (8 entries) and must be programmed before the mode is programmed in MDR1_REG[2:0].
Type	RW

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								Reserved	IRRXINVERT	CIR_PULSE_ MODE		UART_PULSE	STS_FIFO_TRIG		IRTX_UNDERRUN

Bits	Field Name	Description	Type	Reset
15:8	Reserved	Read returns 0x00.	R	0x00
7	Reserved		R	0
6	IRRXINVERT	Only for IR mode (IrDA and CIR). Invert RX pin in the module before the voting or sampling system logic of the infrared block. This does not affect the RX path in UART modem modes. 0x0: Inversion is performed. 0x1: No inversion is performed.	RW	0
5:4	CIR_PULSE_ MODE	CIR pulse modulation definition. Defines high level of the pulse width associated with a digit: 0x0: Pulse width of 3 from 12 cycles 0x1: Pulse width of 4 from 12 cycles 0x2: Pulse width of 5 from 12 cycles 0x3: Pulse width of 6 from 12 cycles	RW	0x00
3	UART_PULSE	UART mode only. Used to allow pulse shaping in UART mode. 0x0: Normal UART mode 0x1: UART mode with pulse shaping	RW	0
2:1	STS_FIFO_TRIG	Only for IR-IrDA mode	RW	0x00

Bits	Field Name	Description	Type	Reset
		Frame status FIFO threshold select:		
		0x0: 1 entry		
		0x1: 4 entries		
		0x2: 7 entries		
		0x3: 8 entries		
0	IRTX_UNDERRUN	IrDA transmission status interrupt. When the IIR_REG[5] interrupt occurs, the meaning of the interrupt is: 0x0: The last bit of the frame was transmitted successfully without error. 0x1: An underrun occurred. The last bit of the frame was transmitted but with an underrun error. The bit is reset to 0 when the RESUME_REG register is read.	R	0

Table 17-81. Register Call Summary for Register MDR2_REG

UART/IrDA/CIR Environment
- IrDA Protocol and Data Format: [0]
- CIR Protocol and Data Format: [1]

UART/IrDA/CIR Functional Description
- Register Access Modes: [2] [3] [4] [5] [6] [7]
- UART/IrDA (SIR, MIR, FIR)/CIR Mode Selection: [8] [9] [10] [11] [12] [13] [14] [15] [16] [17] [18] [19] [20] [21] [22] [23] [24] [25]
- IrDA Mode (UART3 Only): [26] [27]
- CIR Mode (UART3 Only): [28] [29]

UART/IrDA/CIR Register Manual
- UART/IrDA/CIR Register Summary: [30]
- UART/IrDA/CIR Register Description: [31] [32] [33]

Table 17-82. TXFLL_REG

Address Offset	0x028
Physical Address	Please refer to Table 17-39
Description	Transmit frame length register low IrDA modes only The registers TXFLL_REG and TXFLH_REG hold the 13-bit transmit frame length (expressed in bytes). TXFLL_REG holds the LSBs and TXFLH_REG holds the MSBs. The frame length value is used if the frame length method of frame closing is used.
Type	W

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								TXFLL							

Bits	Field Name	Description	Type	Reset
15:8	Reserved	Write has no functional effect.	W	0x00
7:0	TXFLL	LSB register used to specify the frame length	W	0x00

Table 17-83. Register Call Summary for Register TXFLL_REG

UART/IrDA/CIR Functional Description
- Register Access Modes: [0] [1] [2]
- UART/IrDA (SIR, MIR, FIR)/CIR Mode Selection: [3] [4] [5]
- IrDA Mode (UART3 Only): [6]

UART/IrDA/CIR Basic Programming Model
- SIR Mode: [7]
- MIR Mode: [8] [9]

Table 17-83. Register Call Summary for Register TXFLL_REG (continued)

UART/IrDA/CIR Register Manual

- UART/IrDA/CIR Register Summary: [10]
- UART/IrDA/CIR Register Description: [11] [12] [13] [14]

Table 17-84. SFLSR_REG

Address Offset	0x028
Physical Address	Please refer to Table 17-39
Description	Status FIFO line status register
	IrDA modes only
	Reading this register effectively reads frame status information from the status FIFO (this register does not physically exist). Reading this register increments the status FIFO read pointer (SFREGL_REG and SFREGH_REG must be read first).
Type	R

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								Reserved			OE_ERROR	FTL_ERROR	ABORT_DETECT	CRC_ERROR	Reserved

Bits	Field Name	Description	Type	Reset
15:8	Reserved	Read returns 0x00.	R	0x00
7:5	Reserved		R	0x0
4	OE_ERROR		R	-
		0x1: Overrun error in RX FIFO when frame at top of RX FIFO was received. Top of RX FIFO = next frame to be read from RX FIFO.		
3	FTL_ERROR	Frame-too-long error	R	-
		0x1: Frame-length too long error in frame at top of RX FIFO		
2	ABORT_DETECT		R	-
		0x1: Abort pattern detected in frame at top of RX FIFO		
		0x1: CRC error in frame at top of RX FIFO		
1	CRC_ERROR		R	-
0	Reserved	Read returns 0.	R	0

Table 17-85. Register Call Summary for Register SFLSR_REG

UART/IrDA/CIR Functional Description

- FIFO Management: [0]
- Register Access Modes: [1] [2] [3]
- UART/IrDA (SIR, MIR, FIR)/CIR Mode Selection: [4] [5] [6]
- IrDA Mode (UART3 Only): [7] [8]

UART/IrDA/CIR Register Manual

- UART/IrDA/CIR Register Summary: [9]
- UART/IrDA/CIR Register Description: [10] [11]

Table 17-86. RESUME_REG

Address Offset	0x02C
Physical Address	Please refer to Table 17-39
Description	IR-IrDA and IR-CIR modes only This register is used to clear internal flags, which halt transmission/reception when an underrun/overrun error occurs. Reading this register resumes the halted operation. This register does not physically exist and always reads as 0x00.
Type	R

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								RESUME							

Bits	Field Name	Description	Type	Reset
15:8	Reserved	Read returns 0x00	R	0x00
7:0	RESUME	Dummy read to restart the TX or RX	R	0x00

Table 17-87. Register Call Summary for Register RESUME_REG

UART/IrDA/CIR Functional Description

- Register Access Modes: [0] [1] [2]
- UART/IrDA (SIR, MIR, FIR)/CIR Mode Selection: [3] [4] [5] [6] [7] [8]
- UART Mode: [9]
- IrDA Mode (UART3 Only): [10] [11] [12]

UART/IrDA/CIR Register Manual

- UART/IrDA/CIR Register Summary: [13]
- UART/IrDA/CIR Register Description: [14]

Table 17-88. TXFLH_REG

Address Offset	0x02C
Physical Address	Please refer to Table 17-39
Description	Transmit frame length register low IrDA modes only The registers TXFLL_REG and TXFLH_REG hold the 13-bit transmit frame length (expressed in bytes). TXFLL_REG holds the LSBs and TXFLH_REG holds the MSBs. The frame length value is used if the frame length method of frame closing is used.
Type	W

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								Reserved			TXFLH				

Bits	Field Name	Description	Type	Reset
15:8	Reserved	Read returns 0x00.	R	0x00
7:5	Reserved		R	0x0
4:0	TXFLH	MSB register used to specify the frame length	W	0x00

Table 17-89. Register Call Summary for Register TXFLH_REG

UART/IrDA/CIR Functional Description

- Register Access Modes: [0] [1] [2]
- UART/IrDA (SIR, MIR, FIR)/CIR Mode Selection: [3] [4] [5]
- IrDA Mode (UART3 Only): [6]

UART/IrDA/CIR Basic Programming Model

- MIR Mode: [7]

Table 17-89. Register Call Summary for Register TXFLH_REG (continued)

UART/IrDA/CIR Register Manual • UART/IrDA/CIR Register Summary: [8] • UART/IrDA/CIR Register Description: [9] [10] [11] [12]

Table 17-90. RXFLL_REG

Address Offset	0x030
Physical Address	Please refer to Table 17-39
Description	Received frame length register low IrDA modes only The registers RXFLL_REG and RXFLH_REG hold the 12-bit receive maximum frame length. RXFLL_REG holds the LSBs, and RXFLH_REG holds the MSBs. If the intended maximum receive frame length is n bytes, program RXFLL_REG and RXFLH_REG to be n + 3 in SIR or MIR modes and n + 6 in FIR mode (+3 and +6 are the result of frame format with CRC and stop flag; two bytes are associated with the FIR stop flag).
Type	W

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								RXFLL							

Bits	Field Name	Description	Type	Reset
15:8	Reserved	Write has no functional effect.	W	0x00
7:0	RXFLL	LSB register used to specify the frame length in reception	W	0x00

Table 17-91. Register Call Summary for Register RXFLL_REG

UART/IrDA/CIR Functional Description • Register Access Modes: [0] [1] [2] • UART/IrDA (SIR, MIR, FIR)/CIR Mode Selection: [3] [4] [5]
UART/IrDA/CIR Basic Programming Model • MIR Mode: [6] [7]
UART/IrDA/CIR Register Manual • UART/IrDA/CIR Register Summary: [8] • UART/IrDA/CIR Register Description: [9] [10] [11] [12] [13] [14] [15]

Table 17-92. SFREGL_REG

Address Offset	0x030
Physical Address	Please refer to Table 17-39
Description	Status FIFO register low IrDA modes only The frame lengths of received frames are written into the status FIFO. This information can be read by reading the SFREGL_REG and SFREGH_REG registers (these registers do not physically exist). The LSBs are read from SFREGL_REG, and the MSBs are read from SFREGH_REG. Reading these registers does not alter the status FIFO read pointer. These registers must be read before the pointer is incremented by reading the SFLSR_REG.
Type	R

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								SFREGL							

Bits	Field Name	Description	Type	Reset
15:8	Reserved	Read returns 0x00.	R	0x00
7:0	SFREGL	LSB part of the frame length	R	0x–

Table 17-93. Register Call Summary for Register SFREGL_REG

UART/IrDA/CIR Functional Description

- FIFO Management: [0]
- Register Access Modes: [1] [2] [3]
- UART/IrDA (SIR, MIR, FIR)/CIR Mode Selection: [4] [5] [6]
- IrDA Mode (UART3 Only): [7]

UART/IrDA/CIR Register Manual

- UART/IrDA/CIR Register Summary: [8]
- UART/IrDA/CIR Register Description: [9] [10] [11] [12] [13]

Table 17-94. SFREGH_REG

Address Offset	0x034
Physical Address	Please refer to Table 17-39
Description	Status FIFO register high IrDA modes only The frame lengths of received frames are written into the status FIFO. This information can be read by reading the SFREGL_REG and SFREGH_REG registers (these registers do not physically exist). The LSBs are read from SFREGL_REG, and the MSBs are read from SFREGH_REG. Reading these registers does not alter the status FIFO read pointer. These registers must be read before the pointer is incremented by reading the SFLSR_REG.
Type	R

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								Reserved				SFREGH			

Bits	Field Name	Description	Type	Reset
15:8	Reserved	Read returns 0x00.	R	0x00
7:4	Reserved	Read returns 0x0.	R	0x0
3:0	SFREGH	MSB part of the frame length	R	0x-

Table 17-95. Register Call Summary for Register SFREGH_REG

UART/IrDA/CIR Functional Description

- FIFO Management: [0]
- Register Access Modes: [1] [2] [3]
- UART/IrDA (SIR, MIR, FIR)/CIR Mode Selection: [4] [5] [6]
- IrDA Mode (UART3 Only): [7]

UART/IrDA/CIR Register Manual

- UART/IrDA/CIR Register Summary: [8]
- UART/IrDA/CIR Register Description: [9] [10] [11] [12] [13]

Table 17-96. RXFLH_REG

Address Offset	0x034
Physical Address	Please refer to Table 17-39
Description	Received frame length register high IrDA modes only The registers RXFLL_REG and RXFLH_REG hold the 12-bit receive maximum frame length. RXFLL_REG holds the LSBs, and RXFLH_REG holds the MSBs. If the intended maximum receive frame length is n bytes, program RXFLL_REG and RXFLH_REG to be n + 3 in SIR or MIR modes and n + 6 in FIR mode (+3 and +6 are the result of frame format with CRC and stop flag; there are two bytes associated with the FIR stop flag).
Type	W

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								Reserved				RXFLH			

Bits	Field Name	Description	Type	Reset
15:8	Reserved	Write has no functional effect.	W	0x00
7:4	Reserved	Write has no functional effect.	W	0x0
3:0	RXFLH	MSB register used to specify the frame length in reception	W	0x0

Table 17-97. Register Call Summary for Register RXFLH_REG

UART/IrDA/CIR Functional Description
- Register Access Modes: [0] [1] [2]
- UART/IrDA (SIR, MIR, FIR)/CIR Mode Selection: [3] [4] [5]

UART/IrDA/CIR Basic Programming Model
- MIR Mode: [6]

UART/IrDA/CIR Register Manual
- UART/IrDA/CIR Register Summary: [7]
- UART/IrDA/CIR Register Description: [8] [9] [10] [11] [12] [13] [14]

Table 17-98. BLR_REG

Address Offset	0x038
Physical Address	Please refer to Table 17-39
Description	BOF control register IrDA modes only BLR_REG[6] is used to select whether 0xC0 or 0xFF start patterns are to be used, when multiple start flags are required in SIR mode. If only one start flag is required, this is always 0xC0. If n start flags are required, either (–1) 0xC0 or (–1) 0xFF flags are sent, followed by a single 0xC0 flag (immediately preceding the first data byte).
Type	RW

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								STS_FIFO_ RESET	XBOF_TYPE	Reserved					

Bits	Field Name	Description	Type	Reset
15:8	Reserved	Read returns 0x00.	R	0x00
7	STS_FIFO_ RESET	Status FIFO reset. This bit is self-clearing.	RW	0
6	XBOF_TYPE	SIR xBOF select 0x0: 0xFF	RW	1

Bits	Field Name	Description	Type	Reset
		0x1: 0xC0		
5:0	Reserved	Read returns 0x00.	R	0x00

Table 17-99. Register Call Summary for Register BLR_REG

UART/IrDA/CIR Environment • IrDA Protocol and Data Format: [0]
UART/IrDA/CIR Functional Description • Register Access Modes: [1] [2] • UART/IrDA (SIR, MIR, FIR)/CIR Mode Selection: [3] [4]
UART/IrDA/CIR Register Manual • UART/IrDA/CIR Register Summary: [5] • UART/IrDA/CIR Register Description: [6]

Table 17-100. UASR_REG

Address Offset	0x038
Physical Address	Please refer to Table 17-39
Description	UART autobauding status register UART autobauding mode only This status register returns the speed, the number of bits by characters, and the type of parity in UART autobauding mode. In autobauding mode, the input frequency of the UART modem must be fixed to 48 MHz. Any other module clock frequency results in incorrect baud rate recognition.
Type	R

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								PARITY_TYPE		BIT_BY_CHAR	SPEED				

Bits	Field Name	Description	Type	Reset
15:8	Reserved	Read returns 0x00.	R	0x00
7:6	PARITY_TYPE		R	0x0
		0x0: No parity identified		
		0x1: Parity space		
		0x2: Even parity		
		0x3: Odd parity		
5	BIT_BY_CHAR		R	0
		0x0: 7-bit character identified		
		0x1: 8-bit character identified		
4:0	SPEED	Used to report the speed identified	R	0x00
		0x0: No speed identified		
		0x1: 115 200 baud		
		0x2: 57 600 baud		
		0x3: 38 400 baud		
		0x4: 28 800 baud		
		0x5: 19 200 baud		
		0x6: 14 400 baud		
		0x7: 9 600 baud		

Bits	Field Name	Description	Type	Reset
		0x8: 4 800 baud		
		0x9: 4 800 baud		
		0xA: 1 200 baud		

Table 17-101. Register Call Summary for Register UASR_REG

UART/IrDA/CIR Functional Description

- Register Access Modes: [0] [1]
- UART/IrDA (SIR, MIR, FIR)/CIR Mode Selection: [2] [3]
- UART Mode: [4] [5] [6]

UART/IrDA/CIR Register Manual

- UART/IrDA/CIR Register Summary: [7]

Table 17-102. ACREG_REG

Address Offset	0x03C
Physical Address	Please refer to Table 17-39
Description	Auxiliary control register
	IrDA-CIR mode only
Type	RW

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								PULSE_TYPE	SD_MOD	DIS_IR_RX	DIS_TX_UNDERRUN	SEND_SIP	SCTX_EN	ABORT_EN	EOT_EN

Bits	Field Name	Description	Type	Reset
15:8	Reserved	Read returns 0x00.	R	0x00
7	PULSE_TYPE	SIR pulse-width select:	RW	0
		0x0: 3/16 of baud-rate pulse width		
		0x1: 1.6 µs		
6	SD_MOD	Primary output used to configure transceivers. Connected to the SD/MODE input pin of IrDA transceivers.	RW	0
		0x0: SD pin is set to high.		
		0x1: SD pin is set to low.		
5	DIS_IR_RX		RW	0
		0x0: Normal operation (RX input automatically disabled during transmit, but enabled outside of transmit operation).		
		0x1: Disables RX input (permanent state; independent of transmit)		
4	DIS_TX_ UNDERRUN		RW	0
		0x0: Long stop bits cannot be transmitted. TX underrun is enabled.		
		0x1: Long stop bits can be transmitted. TX underrun is disabled.		
3	SEND_SIP	MIR/FIR modes only. Send serial infrared interaction pulse (SIP).	RW	0

Bits	Field Name	Description	Type	Reset
		If this bit is set during an MIR/FIR transmission, the SIP is sent at the end of it. This bit is automatically cleared at the end of the SIP transmission. 0x0: No action 0x1: Send SIP pulse.		
2	SCTX_EN	Store and control TX start. When MDR1_REG[5] = 1 and the MPU writes 1 to this bit, the TX state-machine starts frame transmission. This bit is self-clearing.	RW	0
1	ABORT_EN	Frame abort. The MPU can intentionally abort transmission of a frame by writing 1 to this bit. Neither the end flag nor the CRC bits are appended to the frame.	RW	0
0	EOT_EN	EOT (end-of-transmission) bit. The MPU writes 1 to this bit just before it writes the last byte to the TX FIFO in the set-EOT bit frame-closing method. This bit is automatically cleared when the MPU writes to the THR_REG (TX FIFO).	RW	0

Table 17-103. Register Call Summary for Register ACREG_REG

UART/IrDA/CIR Environment
- UART3 Interface Description: [0]
- IrDA Protocol and Data Format: [1] [2] [3] [4]
- CIR Interface Description: [5]

UART/IrDA/CIR Functional Description
- Register Access Modes: [6] [7]
- UART/IrDA (SIR, MIR, FIR)/CIR Mode Selection: [8] [9] [10] [11]
- IrDA Mode (UART3 Only): [12] [13] [14] [15] [16] [17] [18]
- CIR Mode (UART3 Only): [19] [20]

UART/IrDA/CIR Basic Programming Model
- SIR Mode: [21]
- MIR Mode: [22]

UART/IrDA/CIR Register Manual
- UART/IrDA/CIR Register Summary: [23]
- UART/IrDA/CIR Register Description: [24] [25]

Table 17-104. SCR_REG

Address Offset	0x040
Physical Address	Please refer to Table 17-39
Description	Supplementary control register
Type	RW

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								RX_TRIG_GRANU1	TX_TRIG_GRANU1	Reserved	RX_CTS_WU_EN	TX_EMPTY_CTL_IT	DMA_MODE_2		DMA_MODE_CTL

Bits	Field Name	Description	Type	Reset
15:8	Reserved	Read returns 0x00.	R	0x00
7	RX_TRIG_GRANU1		RW	0
		0x0: Disables the granularity of 1 for TRIGGER RX level		
		0x1: Enables the granularity of 1 for TRIGGER RX level		

Bits	Field Name	Description	Type	Reset
6	TX_TRIG_ GRANU1		RW	0
		0x0: Disables the granularity of 1 for TRIGGER TX level		
		0x1: Enables the granularity of 1 for trigger TX level		
5	Reserved	Read returns 0. Write has no functional effect.	RW	0
4	RX_CTS_ WU_EN	RX CTS wake-up enable	RW	0
		0x0: Disables the WAKE UP interrupt and clears SSR_REG[1]		
		0x1: Waits for a falling edge of pins RX, nCTS, or nDSR to generate an interrupt		
3	TX_EMPTY_ CTL_IT		RW	0
		0x0: Normal mode for THR interrupt (see Table 17-33 for details about UART mode interrupts)		
		0x1: The THR interrupt is generated when TX FIFO and TX shift register are empty.		
2:1	DMA_MODE_2	Specifies the DMA mode valid if SCR_REG[0] = 1	RW	0x0
		0x0: DMA mode 0 (no DMA)		
		0x1: DMA mode 1 (UARTi_DMA_TX, UARTi_DMA_RX)		
		0x2: DMA mode 2 (UARTi_DMA_RX)		
		0x3: DMA mode 3 (UARTi_DMA_TX)		
0	DMA_MODE_ CTL		RW	0
		0x0: The DMA_MODE is set with FCR_REG[3].		
		0x1: The DMA_MODE is set with SCR_REG[2:1].		

Table 17-105. Register Call Summary for Register SCR_REG

UART/IrDA/CIR Functional Description

- FIFO Management: [0]
- FIFO Trigger: [1] [2]
- FIFO DMA Mode Operation: [3] [4]
- Register Access Modes: [5] [6] [7] [8] [9] [10]
- UART/IrDA (SIR, MIR, FIR)/CIR Mode Selection: [11] [12] [13] [14] [15] [16] [17] [18] [19] [20] [21] [22] [23] [24] [25] [26] [27] [28]
- UART Mode: [29] [30]
- UART Mode Power Management: [31]

UART/IrDA/CIR Basic Programming Model

- Quick start: [32] [33] [34] [35] [36] [37] [38] [39]

UART/IrDA/CIR Register Manual

- UART/IrDA/CIR Register Summary: [40]
- UART/IrDA/CIR Register Description: [41] [42] [43] [44] [45] [46] [47] [48] [49] [50]

Table 17-106. SSR_REG

Address Offset	0x044
Physical Address	Please refer to Table 17-39
Description	Supplementary status register
Type	R

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								Reserved						RX_CTS_WU_STS	TX_FIFO_FULL

Bits	Field Name	Description	Type	Reset
15:8	Reserved	Read returns 0x00.	R	0x00
7:2	Reserved	Read returns 0x00.	R	0x00
1	RX_CTS_WU_STS	Pin falling edge detection: Reset only when SCR_REG[4] is reset to 0.	R	0
		0x0: No falling-edge event on RX, nCTS, and nDSR		
		0x1: A falling edge occurred on RX, nCTS, or nDSR.		
0	TX_FIFO_FULL		R	0
		0x0: TX FIFO is not full.		
		0x1: TX FIFO is full.		

Table 17-107. Register Call Summary for Register SSR_REG

UART/IrDA/CIR Functional Description

- FIFO Management: [0]
- Register Access Modes: [1] [2] [3]
- UART/IrDA (SIR, MIR, FIR)/CIR Mode Selection: [4] [5] [6] [7] [8] [9] [10] [11] [12]
- UART Mode: [13]

UART/IrDA/CIR Register Manual

- UART/IrDA/CIR Register Summary: [14]
- UART/IrDA/CIR Register Description: [15]

Table 17-108. EBLR_REG

Address Offset	0x048
Physical Address	Please refer to Table 17-39
Description	BOF length register. IR-IrDA mode only In IR-IrDA SIR operation, this register specifies the number of BOF + xBOFs to transmit. Value set into this register must consider the BOF character; therefore, to send only one BOF with no XBOF, this register must be set to 1. To send one BOF with N XBOF, this register must be set to N + 1. Furthermore, the value 0 sends 1 BOF plus 255 XBOF. In IR-IrDA MIR mode, this register specifies the number of additional start flags (MIR protocol mandates a minimum of 2 start flags).
Type	RW

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								EBLR							

Bits	Field Name	Description	Type	Reset
15:8	Reserved	Read returns 0x00.	R	0x00
7:0	EBLR	IR-IrDA mode: This register allows definition of up to 176 xBOFs, the maximum required by IrDA specification. IR-CIR mode: N/A 0x00: Feature disabled 0x01: Generate RX_STOP interrupt after receiving one zero bit. 0xFF: Generate RX_STOP interrupt after receiving 255 zero bits.	RW	0x00

Table 17-109. Register Call Summary for Register EBLR_REG

UART/IrDA/CIR Functional Description • Register Access Modes: [0] [1] • UART/IrDA (SIR, MIR, FIR)/CIR Mode Selection: [2] [3] [4] [5] • CIR Mode (UART3 Only): [6] [7] [8]
UART/IrDA/CIR Basic Programming Model • SIR Mode: [9] • MIR Mode: [10]
UART/IrDA/CIR Register Manual • UART/IrDA/CIR Register Summary: [11]

Table 17-110. MVR_REG

Address Offset	0x050
Physical Address	Please refer to Table 17-39
Description	Module version register The reset value is fixed by hardware and corresponds to the RTL revision of this module. A reset has no effect on the value returned. UART/IrDA SIR only module is revision 1.x (WMU_012_1 specification). UART/IrDA with SIR, MIR, and FIR support is revision 2.x (WMU_012_2 specification). UART/IrDA with SIR, MIR, and FIR/CIR support is revision 3.x (this specification). For example: MVR_REG = 0x30 => version 3.0 MVR_REG = 0x38 => version 3.8.
Type	R

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								MAJOR_REV				MINOR_REV			

Bits	Field Name	Description	Type	Reset
15:8	Reserved	Read returns 0x00.	R	0x00
7:4	MAJOR_REV	Major revision of the module	R	See (1)
3:0	MINOR_REV	Minor revision of the module	R	See (1)

(1) TI internal data

Table 17-111. Register Call Summary for Register MVR_REG

UART/IrDA/CIR Functional Description • Register Access Modes: [0] [1] [2] • UART/IrDA (SIR, MIR, FIR)/CIR Mode Selection: [3] [4] [5] [6] [7] [8] [9] [10] [11]
UART/IrDA/CIR Register Manual • UART/IrDA/CIR Register Summary: [12] • UART/IrDA/CIR Register Description: [13] [14]

Table 17-112. SYSC_REG

Address Offset	0x054
Physical Address	Please refer to Table 17-39
Description	System configuration register. The auto idle bit controls a power-saving technique to reduce the logic power consumption of the module interface; that is, when the feature is enabled, the interface clock is gated off until the module interface is accessed. When the software reset bit is set high, it causes a full device reset.
Type	RW

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								Reserved			IDLEMODE		ENAWAKEUP	SOFTRESET	AUTOIDLE

Bits	Field Name	Description	Type	Reset
15:8	Reserved	Read returns 0x00.	R	0x00
7:5	Reserved	Read returns 0x0.	R	0x0
4:3	IDLEMODE	Power management req/ack control 0x0: Force idle: Idle request is acknowledged unconditionally. 0x1: No-idle: Idle request is never acknowledged. 0x2: Smart idle: Idle request is acknowledged based on the internal module activity. 0x3: Reserved	RW	0x0
2	ENAWAKEUP	Wakeup control 0x0: Wakeup is disabled. 0x1: Wakeup capability is enabled.	RW	0
1	SOFTRESET	Software reset. Set this bit to 1 to trigger a module reset. This bit is automatically reset by the hardware. Read returns 0. 0x0: Normal mode 0x1: The module is reset.	RW	0
0	AUTOIDLE	Internal interface clock-gating strategy 0x0: Clock is running. 0x1: Automatic interface clock-gating strategy is applied based on interface activity.	RW	0

Table 17-113. Register Call Summary for Register SYSC_REG

UART/IrDA/CIR Integration
- Clocking: [0]
- Software Reset: [1]

UART/IrDA/CIR Functional Description
- Register Access Modes: [2] [3] [4] [5] [6] [7]
- UART/IrDA (SIR, MIR, FIR)/CIR Mode Selection: [8] [9] [10] [11] [12] [13] [14] [15] [16] [17] [18] [19] [20] [21] [22] [23] [24] [25]
- UART Mode Power Management: [26]

UART/IrDA/CIR Basic Programming Model
- Quick start: [27]

UART/IrDA/CIR Register Manual
- UART/IrDA/CIR Register Summary: [28]

Table 17-114. SYSS_REG

Address Offset	0x058
Physical Address	Please refer to Table 17-39
Description	System status register
Type	R

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								Reserved							RESETDONE

Bits	Field Name	Description	Type	Reset
15:8	Reserved	Read returns 0x00.	R	0x00
7:1	Reserved	Read returns 0x00.	R	0x00
0	RESETDONE	Internal reset monitoring	R	0
		0x0: Internal module reset is ongoing.		
		0x1: Reset complete		

Table 17-115. Register Call Summary for Register SYSS_REG

UART/IrDA/CIR Functional Description
- Register Access Modes: [0] [1] [2]
- UART/IrDA (SIR, MIR, FIR)/CIR Mode Selection: [3] [4] [5] [6] [7] [8] [9] [10] [11]

UART/IrDA/CIR Basic Programming Model
- Quick start: [12]

UART/IrDA/CIR Register Manual
- UART/IrDA/CIR Register Summary: [13]

Table 17-116. WER_REG

Address Offset	0x05C
Physical Address	Please refer to Table 17-39
Description	Wake-up enable register The UART wake-up enable register is used to mask and unmask a UART event that subsequently notifies the system. An event is any activity in the logic that can cause an interrupt and/ or an activity that requires the system to wake up. Even if wakeup is disabled for certain events, if these events are also an interrupt to the UART, the UART still registers the interrupt as such.
Type	RW

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								Reserved	EVENT_6_RLS_INTERRUPT	EVENT_5_RHR_INTERRUPT	EVENT_4_RX_ACTIVITY	Reserved	EVENT_2_RI_ACTIVITY	Reserved	EVENT_0_CTS_ACTIVITY

Bits	Field Name	Description	Type	Reset
15:8	Reserved	Read returns 0x00.	R	0x00
7	Reserved	Read returns 0.	R	0
6	EVENT_6_RLS_INTERRUPT	Receiver line status interrupt 0x0: Event is not allowed to wake up the system. 0x1: Event can wake up the system.	RW	1
5	EVENT_5_RHR_INTERRUPT	0x0: Event is not allowed to wake up the system. 0x1: Event can wake up the system.	RW	-
4	EVENT_4_RX_ACTIVITY	0x0: Event is not allowed to wake up the system. 0x1: Event can wake up the system.	RW	1
3	Reserved	Read returns 1. Must be set to 1 for correct behavior.	RW	1
2	EVENT_2_RI_ACTIVITY	0x0: Event is not allowed to wake up the system. 0x1: Event can wake up the system.	RW	1
1	Reserved	Read returns 1. Must be set to 1 for correct behavior.	RW	1
0	EVENT_0_CTS_ACTIVITY	UART mode only 0x0: Event is not allowed to wake up the system. 0x1: Event can wake up the system.	RW	1

Table 17-117. Register Call Summary for Register WER_REG

UART/IrDA/CIR Functional Description

- Register Access Modes: [0] [1] [2] [3] [4] [5]
- UART/IrDA (SIR, MIR, FIR)/CIR Mode Selection: [6] [7] [8] [9] [10] [11] [12] [13] [14] [15] [16] [17] [18] [19] [20] [21] [22] [23]
- UART Mode Power Management: [24]

UART/IrDA/CIR Register Manual

- UART/IrDA/CIR Register Summary: [25]

Table 17-118. CFPS_REG

Address Offset	0x060
Physical Address	Please refer to Table 17-39
Description	Carrier frequency prescaler
Type	RW

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								CFPS							

Bits	Field Name	Description	Type	Reset
15:8	Reserved	Read returns 0x00.	R	0x00
7:0	CFPS	Because the consumer IR works at modulation rates of 30-56.8 kHz, the 48-MHz clock must be prescaled before the clock can drive the IR logic. This register sets the divisor rate to give a range to accommodate remote-control requirements in BAUD multiples of 12x. The value of the CFPS at reset is 0105 (decimal), which equates to a 38.1-kHz output from starting conditions. The 48-MHz carrier is prescaled by the CFPS, which is then divided by the 12x BAUD multiple. Example:	RW	0x69

Bits	Field Name	Description			Type	Reset
		Target Freq (kHz)	**CFPS (decimal)**	**Actual Freq(kHz)**		
		30	133	30.08		
		32.75	122	32.79		
		36	111	36.04		
		36.7	109	36.69		
		38	105	38.1		
		40	100	40		
		56.8	70	57.14		
		CFPS = 0 is not supported.				

Table 17-119. Register Call Summary for Register CFPS_REG

UART/IrDA/CIR Functional Description • Register Access Modes: [0] [1] [2] [3] [4] [5] • UART/IrDA (SIR, MIR, FIR)/CIR Mode Selection: [6] [7] [8] [9] [10] [11] • CIR Mode (UART3 Only): [12] [13] [14]
UART/IrDA/CIR Register Manual • UART/IrDA/CIR Register Summary: [15]

I2C

This chapter describes the four high-speed I^{2}C controller modules in the OMAP34xx multimedia device.

Topic **Page**

18.1 High-Speed I^{2}C Controller Overview 2724
18.2 High-Speed I^{2}C Controller Environment 2726
18.3 High-Speed I^{2}C Controller Integration 2738
18.4 High-Speed I^{2}C Controller Functional Description 2746
18.5 High-Speed I^{2}C Controller Basic Programming Model 2755
18.6 High-Speed Multimaster I^{2}C Use Cases and Tips 2774
18.7 High-Speed I^{2}C Controllers Register Manual 2785

18.1 High-Speed I2C Controller Overview

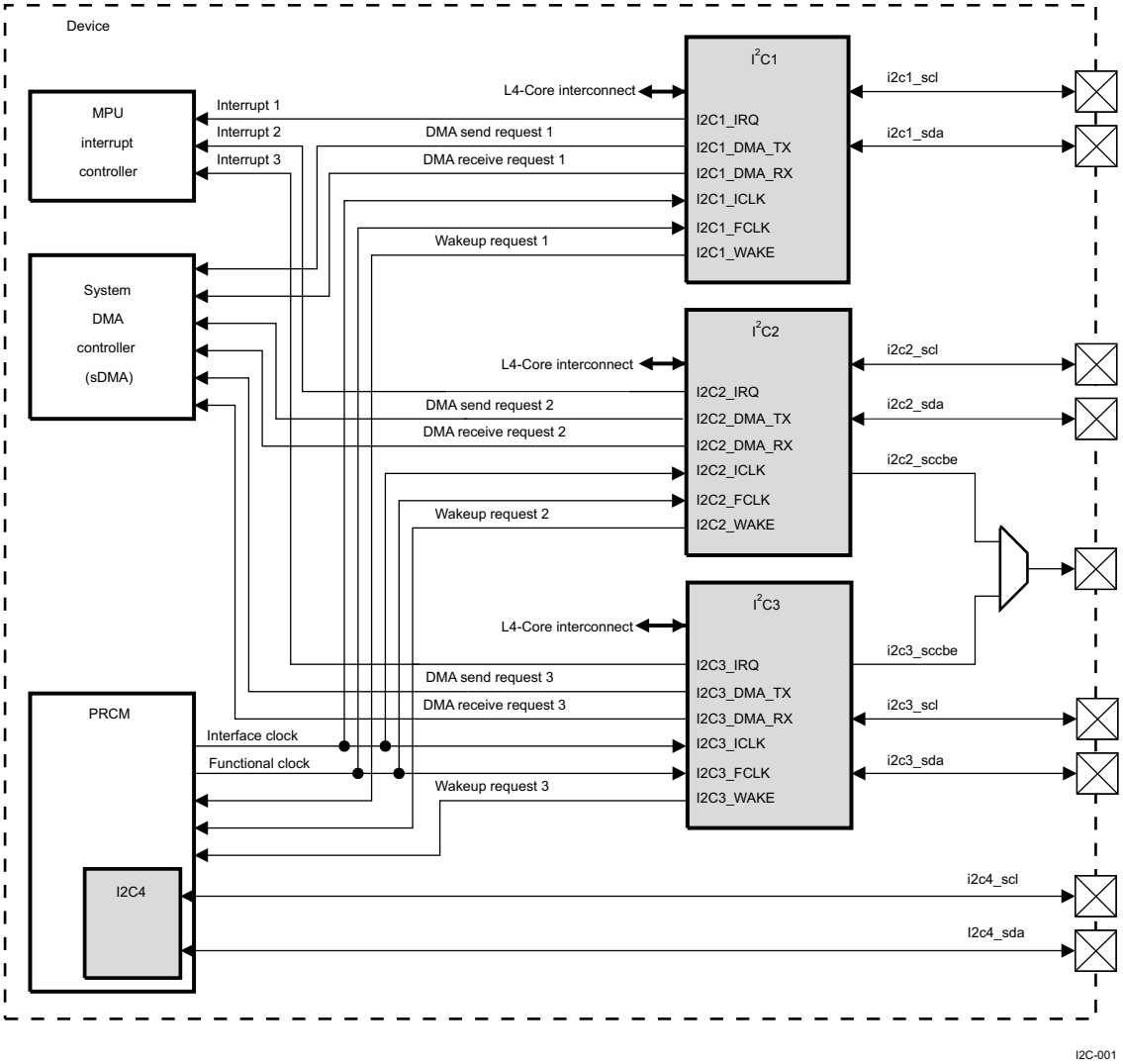
The device contains three multimaster high-speed (HS) inter-integrated circuit (I2C) controllers (I2Ci modules, where i = 1, 2, 3), each of which provides an interface between a local host (LH), such as the MPU subsystem, and any I2C-bus-compatible device that connects through the I2C serial bus. External components attached to the I2C bus can serially transmit/receive up to 8 bits of data to/from the LH device through the 2-wire I2C interface.

Each multimaster HS I2C controller can be configured to act like a slave or master I2C-compatible device. Moreover, each multimaster HS I2C controller can be configured in serial camera control bus (SCCB) mode (the SCCB is a serial bus developed by Omnivision Technologies, Inc.) to act as a master on a 2-wire SCCB bus. Only multimaster HS I2C controllers I2C2 and I2C3 can be configured in SCCB mode to act as a master device on a 3-wire SCCB bus.

The device contains an additional master transmitter HS I2C interface (I2C4) in the power, reset, and clock management (PRCM) module to perform dynamic voltage control and power sequencing. Texas Instruments provides a global solution with the device connected to power chips (TWL4030 device).

Figure 18-1 shows the HS I2C controllers in the device.

Figure 18-1. HS I2C Controllers



The three multimaster HS I2C controllers have the following features:

- Compliance with Philips I2C specification version 2.1
- Support for standard mode (up to 100K bits/s) and fast mode (up to 400K bits/s)
- Support for HS mode for transfer up to 3.4M bits/s
- Support for 3-wire/2-wire SCCB master mode for I2C2 and I2C3 modules, 2-wire SCCB master mode for I2C1 module, up to 100K bits/s
- 7-bit and 10-bit device addressing modes
- General call
- Start/restart/stop
- Multimaster transmitter/slave receiver mode
- Multimaster receiver/slave transmitter mode
- Combined master transmit/receive and receive/transmit mode
- Built-in FIFO for buffered read or write
  - 8 bytes for I2C1 and I2C2
  - 64 bytes for I2C3
- Module enable/disable capability
- Programmable clock generation
- 8-bit-wide data access
- Low-power consumption design
- Two DMA channels
- Wide interrupt capability

The master transmitter HS I2C controller I2C4 has the following features:

- Support of HS and fast modes
- 7-bit addressing mode only
- Master transmitter mode only
- Start/restart/stop

Note: The I2C4 clock frequency in HS mode is equal to the SYS_CLK clock frequency divided by 15.

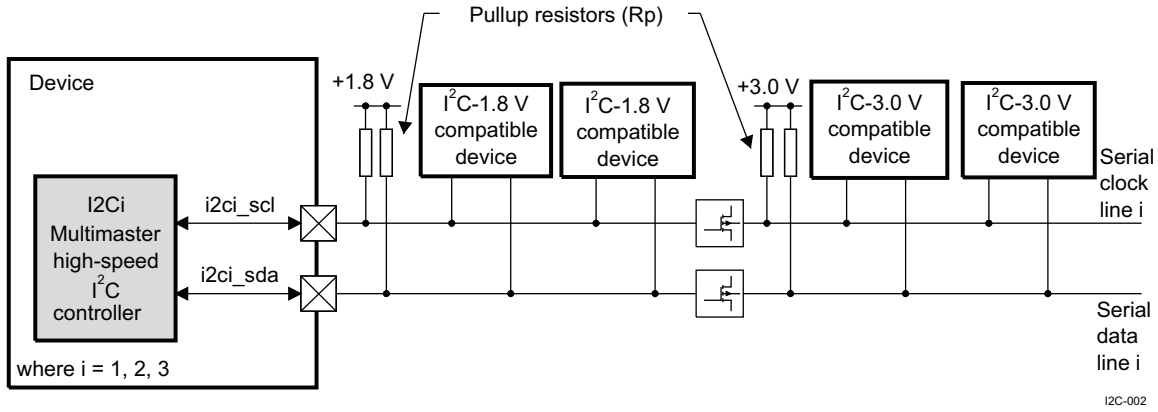
Note: Before using HS I2C mode, determine that the target device supports this mode.

18.2 High-Speed I²C Controller Environment

18.2.1 Multimaster HS I²C Controllers in I²C Mode

Figure 18-2 shows the multimaster HS I²C controllers and their related connections with I²C-compliant devices in I²C mode.

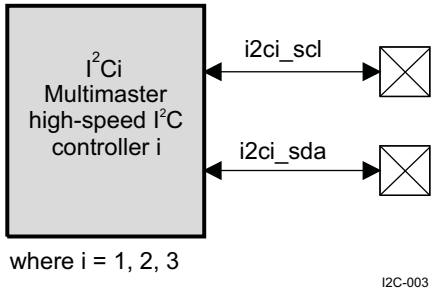
Figure 18-2. Multimaster HS I²C Controllers and Typical Connections to I²C Devices



18.2.1.1 Multimaster HS I²C Controller Pins for Typical Connections in I²C Mode

Figure 18-3 shows the multimaster HS I²C controller pins used for typical connections with I²C devices.

Figure 18-3. Multimaster HS I²C Controller Interface Signals in I²C Mode



18.2.1.2 I²C Interface Typical Connections

Table 18-1 lists the pins associated with the I²C interface.

Table 18-1. Input/Output

Signal	I/O[1]	Description	Reset Value
i2ci_scl	I/O(OD)	I²C serial clock line[2]. Open-drain output buffer. Requires external pullup resistor (Rp).	Hi-Z
i2ci_sda	I/O(OD)	I²C serial data line. Open-drain output buffer. Requires external Rp.	Hi-Z

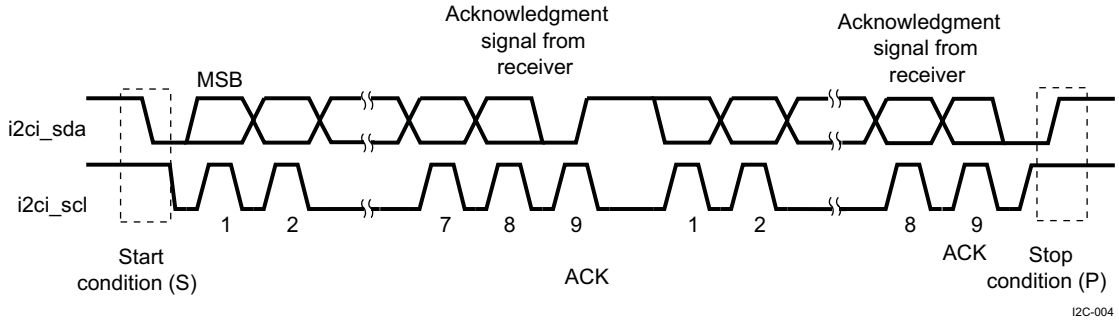
(1) I = Input; O = Output; OD = Open Drain; Hi-Z = High Impedance
(2) This signal is also used as re-timing input.

18.2.1.3 I²C Typical Connection Protocol and Data Format

18.2.1.3.1 Serial Data Format

The I2C controller operates in 8-bit word data format (byte write access supported for the last access). Each byte transmitted or received on the serial data line (i2ci_sda) is 8 bits long. The number of bytes that can be transmitted or received is not restricted. The data is transferred with the most-significant bit (MSB) first. In receiver mode, each byte is followed by an acknowledge bit from the I2C. Figure 18-4 shows a typical I2C communication format.

Figure 18-4. I2C Data Transfer

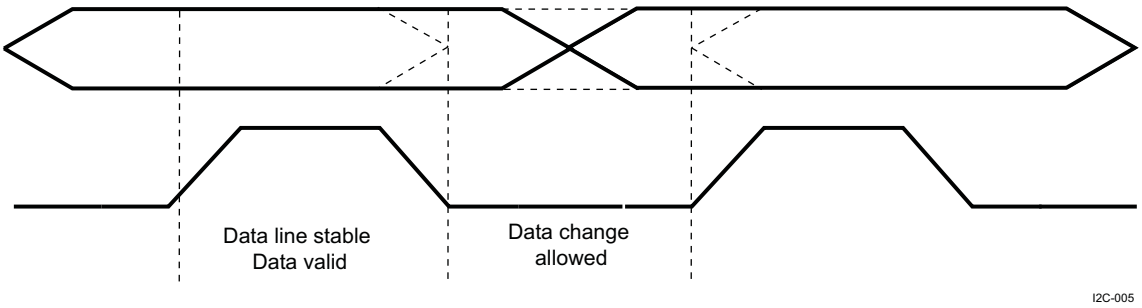


18.2.1.3.2 Data Validity

The data on the serial data line must be stable during the high period of the clock i2ci_scl. The high and low states of the data line can change only when the clock signal on the serial clock line is low.

Figure 18-5 is an example of data validity requirements.

Figure 18-5. Bit Transfer on the I2C Bus



18.2.1.3.3 Start and Stop Conditions

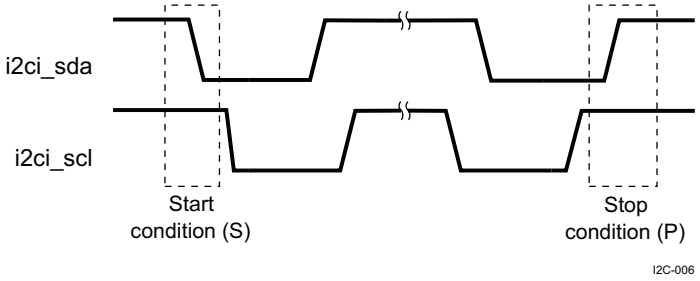
The I2C module generates start (S) and stop (P) conditions when it is configured as a master.

- An S condition is a high-to-low transition on the i2ci_sda line while i2ci_scl is high.
- A P condition is a low-to-high transition on the i2ci_sda line while i2ci_scl is high.

The bus is considered busy after the S condition (the I2Ci.I2C_STAT[12] BB bit is 1 to indicate that the bus is busy) and free after the P condition (the I2Ci.I2C_STAT[12] BB bit is 0 to indicate that the bus is free).

Figure 18-6 shows the waveforms that occur during an S and a P condition.

Figure 18-6. S and P Condition Events



18.2.1.3.4 Addressing

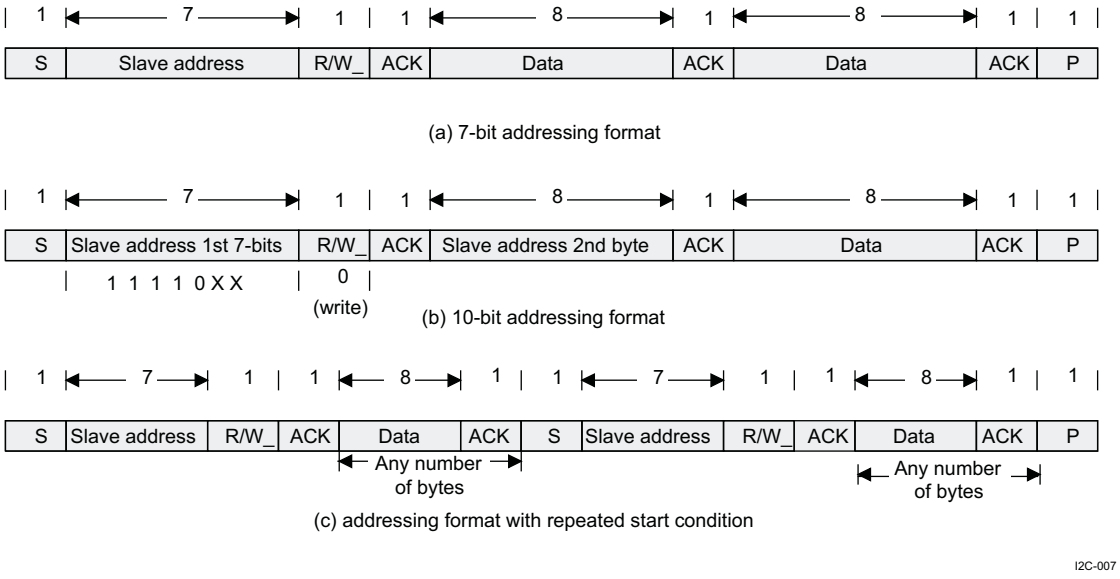
The I2C module supports two data formats in fast/standard (F/S) and HS modes:

- 7-bit/10-bit addressing format
- 7-bit/10-bit addressing format with repeated start (Sr) condition

18.2.1.3.4.1 Data Transfer Format in F/S Mode

Figure 18-7 shows the I2C data transfer format in F/S mode.

Figure 18-7. I2C Data Transfer Formats in F/S Mode



The first word after a S condition consists of 8 bits. In acknowledge mode, an extra dedicated acknowledgment bit is inserted after each byte.

In addressing formats with 7-bit addresses, the first byte is composed of 7 MSB slave address bits and 1 least-significant bit (LSB) R/W_ bit.

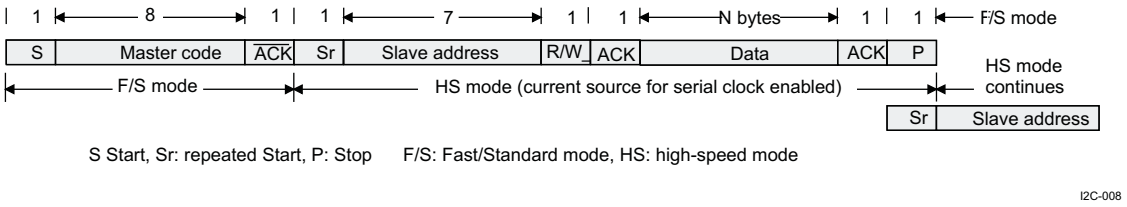
The LSB R/W_ bit of the address byte indicates the transmission direction of the data bytes that follow it. If R/W_ is 0, the master writes data to the selected slave; if it is 1, the master reads data from the slave.

In addressing formats with 10-bit addresses, the structure of the first byte is 11110XXY, where XX is the two MSBs of the 10-bit addresses, and Y is the R/W_ bit. If the R/W_ bit is 0, the next byte contains the last 8 bits of the slave address. If the R/W_ bit is 1, the next byte contains data transmitted from the slave to the master.

18.2.1.3.4.2 Data Transfer Format in HS Mode

Figure 18-8 shows the I²C data transfer format in HS mode.

Figure 18-8. I²C Data Transfers in HS Mode



Each multimaster HS I²C controller module can also operate in HS mode. In this case, after the S condition, the module, which is in F/S mode, writes the master code address (000001XXX, where XXX is the variable portion of the master code) on the bus. No device connected on the same bus acknowledges this address. The module switches the clock to the HS clock and after an Sr condition, and sends the slave address and the data, as shown in Figure 18-8.

Note: For more information, see *I²C Bus Specification v2.1*, January 2000.

18.2.1.3.5 Master Transmitter

In master transmitter mode, data assembled in one of the previously described data formats is shifted out on serial data line i2ci_sda in synchronization with the self-generated clock pulses on serial clock line i2ci_scl. When the intervention of the processor is required (I2Ci.I2C_STAT[10] XUDF bit) after a byte is transmitted, the clock pulses are inhibited and the serial clock line is held low.

18.2.1.3.6 Master Receiver

Master receiver mode can be entered only from master transmitter mode. With any of the address formats (a), (b), or (c) (see Figure 18-7), if R/W_ is high, the module enters master receiver mode after the slave address byte and bit R/W_ are transmitted. Serial data bits received on bus line i2ci_sda are shifted in synchronization with the self-generated clock pulses on i2ci_scl. When the intervention of the processor is required (I2Ci.I2C_STAT[11] ROVR bit) after a byte is transmitted, the clock pulses are inhibited and the serial clock line is held low. At the end of a transfer, a P condition is generated.

18.2.1.3.7 Slave Transmitter

Slave transmitter mode can be entered only from slave receiver mode. With any of the address formats (a), (b), or (c) (see Figure 18-7), slave transmitter mode is entered if the slave address byte is the same as its own address, and when the R/W_ bit transmitted by the master transmitter is high. The slave transmitter shifts the serial data out on data line i2ci_sda in synchronization with the clock pulses generated by the master device. It does not generate the clock, but it can hold clock line i2ci_scl low when intervention of the LH is required (the I2Ci.I2C_STAT[10] XUDF bit).

18.2.1.3.8 Slave Receiver

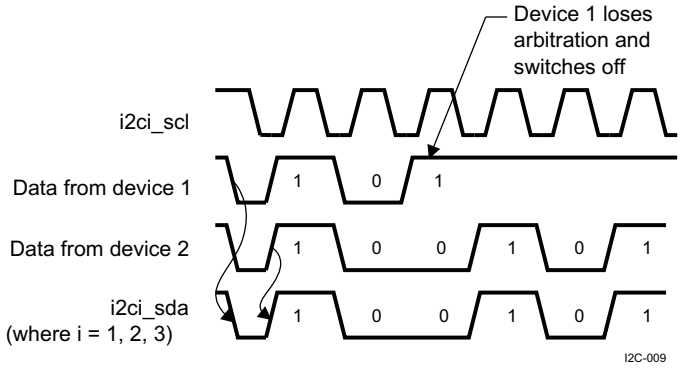
In slave receiver mode, serial data bits received on bus line i2ci_sda are shifted in in synchronization with the clock pulses on i2ci_scl generated by the master device. The slave receiver does not generate the clock, but it can hold the serial clock line low when intervention of the LH is required (the I2Ci.I2C_STAT[11] ROVR bit) after a byte is received.

18.2.1.3.9 Bus Arbitration

If two or more master transmitters start a transmission on the same bus simultaneously, an arbitration procedure is invoked. This arbitration procedure uses the data presented on the serial bus by the competing transmitters. When a transmitter senses that a high signal it has presented on the bus has

been overruled by a low signal, it switches to slave receiver mode, sets the arbitration lost flag (I2Ci.I2C_STAT[0] AL bit), and generates the arbitration lost interrupt. Figure 18-9 shows arbitration between two devices. The arbitration procedure gives priority to the device that transmits the serial data stream with the lowest binary value. If two or more devices send identical first bytes, arbitration continues on the subsequent bytes.

Figure 18-9. Arbitration Between Master Transmitters

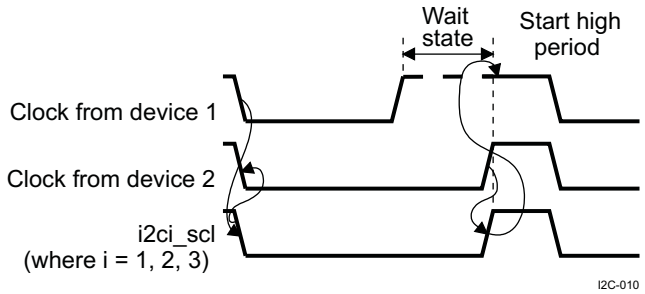


18.2.1.3.10 I2C Clock Generation and Synchronization

Under normal conditions, only one master device generates the clock signal i2ci_scl. During arbitration, however, there are two or more master devices, and the clock must be synchronized so that the data output can be compared. The wired-AND property of the clock line means that the device that first generates a low period of the clock line overrules the other devices. At this high-low transition, the clock generators of the other devices are forced to start generating their own low periods. The clock line is then held low by the device with the longest low period, while the other devices that finish their low periods must wait for the clock line to be released before starting their high periods. A synchronized signal on the clock line is thus obtained, where the slowest device determines the length of the low period and the fastest device determines the length of the high period.

If a device pulls down the clock line for a longer time, all clock generators must enter the wait state. In this way, a slave can slow down a fast master, and the slow device can create enough time to store a received byte or prepare a byte to be transmitted. Figure 18-10 shows clock synchronization.

Figure 18-10. Synchronization of I2C Clock Generators



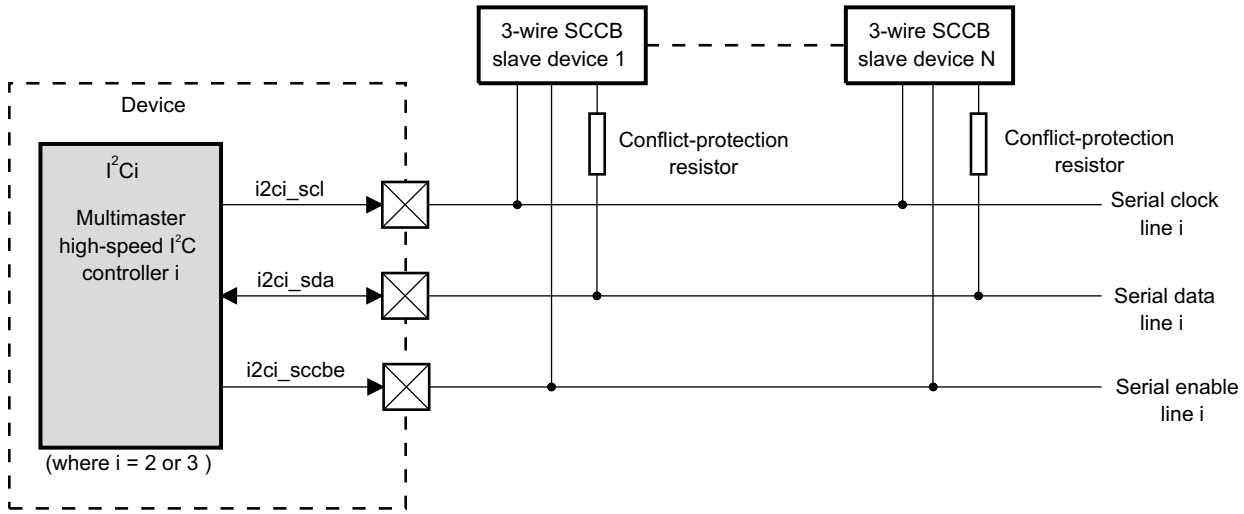
18.2.2 Multimaster High-Speed I2C Controllers in SCCB Mode

The SCCB is a 3-wire serial bus developed by Omnivision Technologies, Inc. The SCCB can also operate in a modified 2-wire serial mode. For details, see the SCCB specifications version 2.1 document at *http://www.ovt.com/pdfs/ds_note.pdf*.

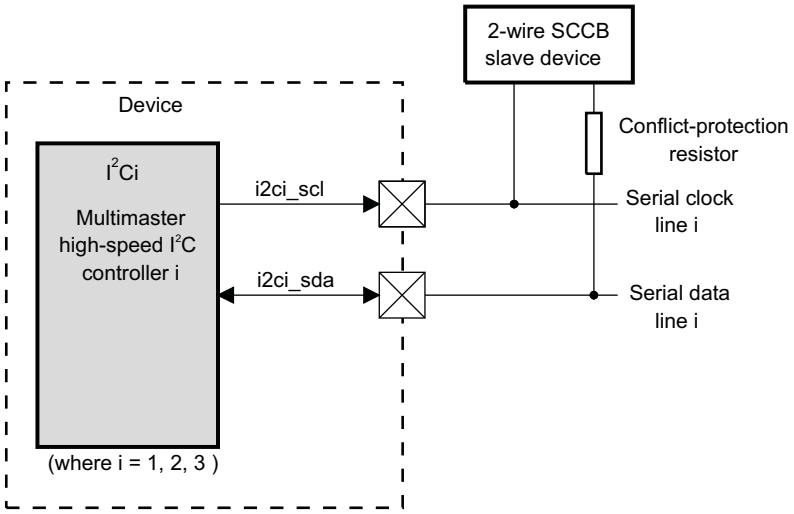
The multimaster HS I2C controllers support the 2-wire SCCB protocol in master mode. Only the I2C2 and I2C3 modules support the 3-wire SCCB protocol in master mode.

Figure 18-11 shows the multimaster HS I2C controllers and their related connections with 3-wire or 2-wire SCCB-compliant devices.

Figure 18-11. Multimaster HS I²C Controllers and Typical Connections to SCCB Devices



(a) Typical connection with 3-wire SCCB devices



(b) Typical connection with 2-wire SCCB devices

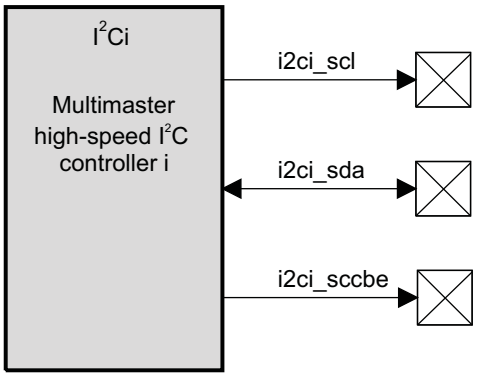
I2C-011

Note: Only one 2-wire SCCB slave device can be connected to the 2-wire SCCB bus.

18.2.2.1 Multimaster HS I²C Controller Pins for Typical Connections in SCCB Mode

Figure 18-12 shows the multimaster HS I²C controller pins used for typical connections with 3-wire or 2-wire SCCB devices.

Figure 18-12. Multimaster HS I²C Controller Interface Signals in SCCB Mode



(where i = 1, 2, 3 and i2ci_sccbe is not available for I2C1)

I2C-012

18.2.2.2 SCCB Interface Typical Connections

Table 18-2 lists the pins associated with the SCCB interface.

Table 18-2. Input/Output

Signal	I/O(1)	Description	Reset Value	I2Ci.I2C_CON[15] I2C_EN bit = 0
i2ci_scl	O	SCCB serial clock line(2). Standard CMOS output buffer.	Hi-Z	High
i2ci_sda	I/O(OD)	SCCB serial data line. Standard CMOS 3-state output buffer. Requires external conflict-protection resistor for each slave device connected to the bus.	Hi-Z	Hi-Z
i2ci_sccbe (3)	O	SCCB enable line. Standard CMOS output buffer.	High	High

(1) I = Input; O = Output; OD = Open Drain; Hi-Z = High Impedance
(2) This output signal is also used as re-timing input.
(3) This signal is used for the 3-wire SCCB protocol only.

Note: Because they share the same ball, the i2c2_sccbe and i2c3_sccbe signals are not available at the same time. For detailed information about pin configuration, see Chapter 7, *System Control Module*.

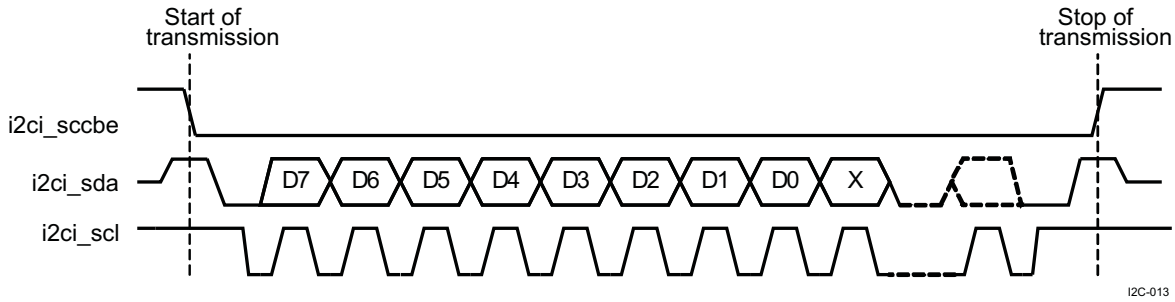
The I2C1 module does not provide any i2c1_sccbe signal at the chip boundary of the device; thus, this module does not support the 3-wire SCCB protocol

18.2.2.3 SCCB Typical Connection Protocol and Data Format

18.2.2.3.1 Serial Transmission Timing Diagram

Figure 18-13 is the timing diagram of the 3-wire SCCB data transmission.

Figure 18-13. 3-wire SCCB Transmission Timing Diagram

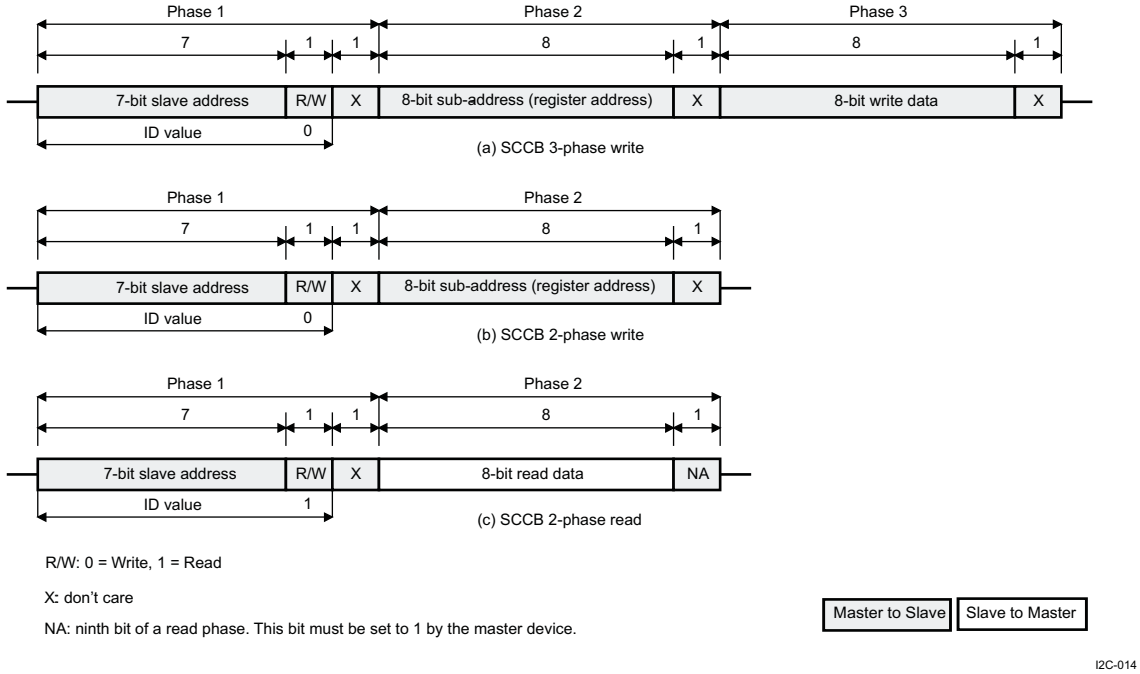


Note: When operating in 2-wire SCCB mode, the i2ci_sccbe signal is not used by the 2-wire SCCB-compliant slave device attached to the 2-wire SCCB bus.

18.2.2.3.2 SCCB Transmission Data Formats

Figure 18-14 describes the data format of the three kinds of transmission.

Figure 18-14. SCCB Transmission Data Formats



The basic element of a data transmission is a phase. A phase contains 9 bits, which consist of an 8-bit sequential data transmission followed by a ninth bit. The ninth bit is a don't-care bit (X) or an NA bit, depending on whether the data transmission is a write or a read. The maximum number of phases that can be included in a transmission is three. The MSB is always asserted first for each phase.

A data transmission is one of three types:

- **3-phase write transmission**: The 3-phase write transmission cycle (see (a) of Figure 18-14) is a full write operation in which the master can write 1 byte of data to a specific slave(s). The 7-bit slave address in the ID value identifies the specific slave that the master intends to access. The subaddress identifies the register location of the specified slave. The write data contains 8-bit data that the master intends to write over the content of this specific address. The ninth bit of each of the three phases is a don't-care bit (X bit).
- **2-phase write transmission**: The 2-phase write transmission cycle is followed by a 2-phase read transmission cycle (see below). The purpose of issuing a 2-phase write transmission cycle (see (b) of Figure 18-14) is to identify the subaddress of some specific slave from which the master intends to read data for the following 2-phase read transmission cycle. The ninth bit of each phase is a don't-care bit (X bit).
- **2-phase read transmission**: Either a 3-phase or a 2-phase write transmission cycle must be asserted ahead of a 2-phase read transmission cycle. The 2-phase read transmission cycle (see (c) of Figure 18-14) has no ability to identify the subaddress. The 2-phase write transmission cycle contains read data of 8 bits and a ninth don't-care bit or NA bit. The master must drive the NA bit at logical 1.

  In each transmission type, phase 1 (7-bit slave address of the ID value) is asserted by the master to identify the selected slave to which the data is read or written. Each slave has a unique 7-bit slave address. The 7-bit slave address of the ID value comprises 7 bits, from bit 7 to bit 1, that can identify up to 128 slaves. The eighth bit, bit 0, is the read/write selector bit that specifies the transmission direction of the current cycle. A logical 0 represents a write cycle and a logical 1 represents a read cycle. The ninth bit of phase 1 is a don't-care bit (X bit).

  Phase 2 (subaddress/read data) is asserted by the master (subaddress) or the slave(s) (read data). A phase 2 transmission asserted by the master identifies the subaddress of the slave(s) the master intends to access. A phase 2 transmission asserted by the slave(s) indicates the read data that the master will receive. The slave(s) recognize the subaddress of this read data according to the previous 3-phase or 2-phase write transmission cycle. The ninth bit is defined as a don't-care bit (X bit) when the master asserts phase 2. The ninth bit is defined as an NA bit when the slave(s) asserts the phase 2 transmission. The master is responsible for a logical 1 during the period of the NA bit.

  Phase 3 (write data) is asserted only by the master. This phase contains the data the master intends to write to the slave(s). Because the master asserts the transmission, the ninth bit of the phase 3 transmission is defined as a don't-care bit (X bit).

Note: A multimaster HS I2C controller configured in SCCB mode can perform two operations:

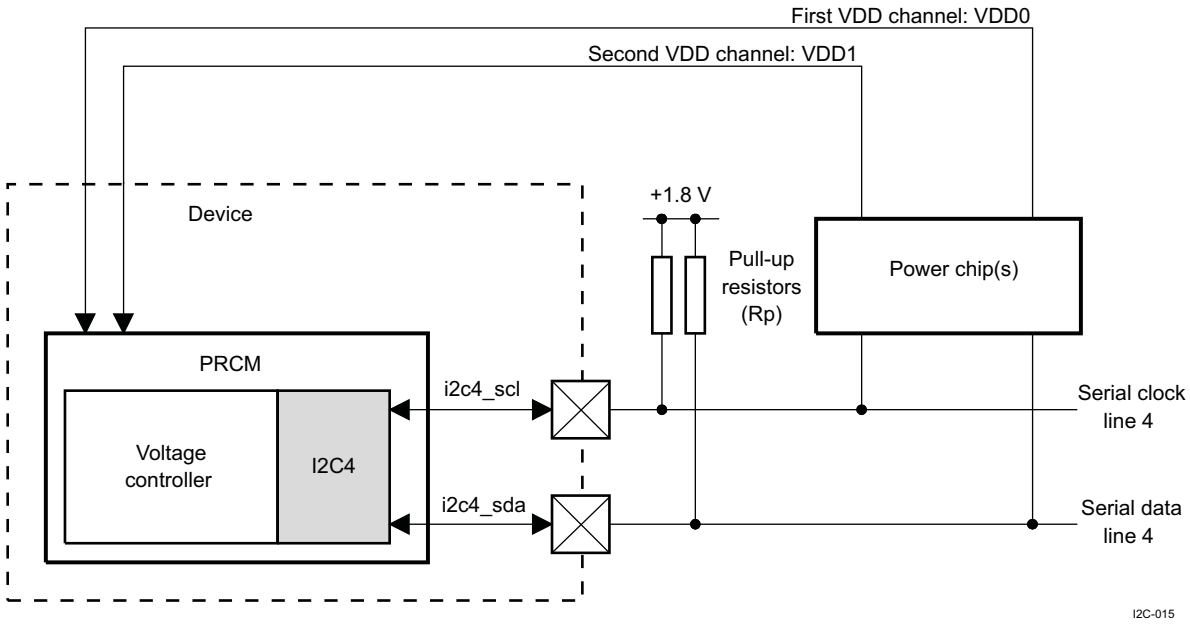
- Writing a single byte to an SCCB slave device by using the 3-phase write transmission cycle
- Reading a single byte from an SCCB slave device by using the 2-phase write transmission cycle followed by the 2-phase read transmission cycle

18.2.3 High-Speed I2C Controller for Communication With Power Chip(s)

The master transmitter HS I2C controller I2C4 interfaces between the and external power chip(s) for voltage control. This module is always configured as an I2C master transmitter; it does not support the SCCB protocol. For a definition of master transmitter mode, see Section 18.2.1.3.5, *Master Transmitter*.

Figure 18-15 shows a typical connection between the master transmitter HS I2C controller I2C4 of the device and external power chip(s).

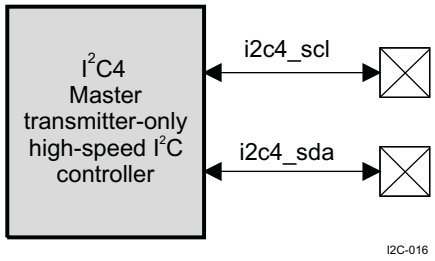
Figure 18-15. Typical Connection Between the HS I2C Controller and Power Chip(s)



18.2.3.1 HS I2C Controller I2C4 Pins for Typical Connections

Figure 18-16 shows the HS I2C controller I2C4 pins used for typical connections with power chips.

Figure 18-16. HS I2C Controller I2C4 Interface Signals



18.2.3.2 HS I2C Controller I2C4 Interface Typical Connections

Table 18-3 lists the pins associated with the I2C interface of the HS I2C controller I2C4 of the device.

Table 18-3. Input/Output Description

Signal	I/O(1)	Description	Reset Value
i2c4_scl	I/O(OD)	I2C serial clock line(2). Open-drain output buffer. Requires external Rp.	Hi-Z
i2c4_sda	I/O(OD)	I2C serial data line. Open-drain output buffer. Requires external Rp.	Hi-Z

(1) I = Input; O = Output; OD = Open Drain; Hi-Z = High Impedance
(2) This signal is also used as re-timing input.

Note: The I2C4 clock frequency in HS mode is equal to the SYS_CLK clock frequency divided by 15.

18.2.3.3 I2C Typical Connections Protocol and Data Format for I2C4

18.2.3.3.1 Serial Data Format

The serial data format is the same as described in Section 18.2.1.3.1, *Serial Data Format*.

18.2.3.3.2 Data Validity

The data validity is the same as described in Section 18.2.1.3.2, *Data Validity*.

18.2.3.3.3 S and P Conditions

The S and stop P conditions are the same as described in Section 18.2.1.3.3, *Start and Stop Conditions*.

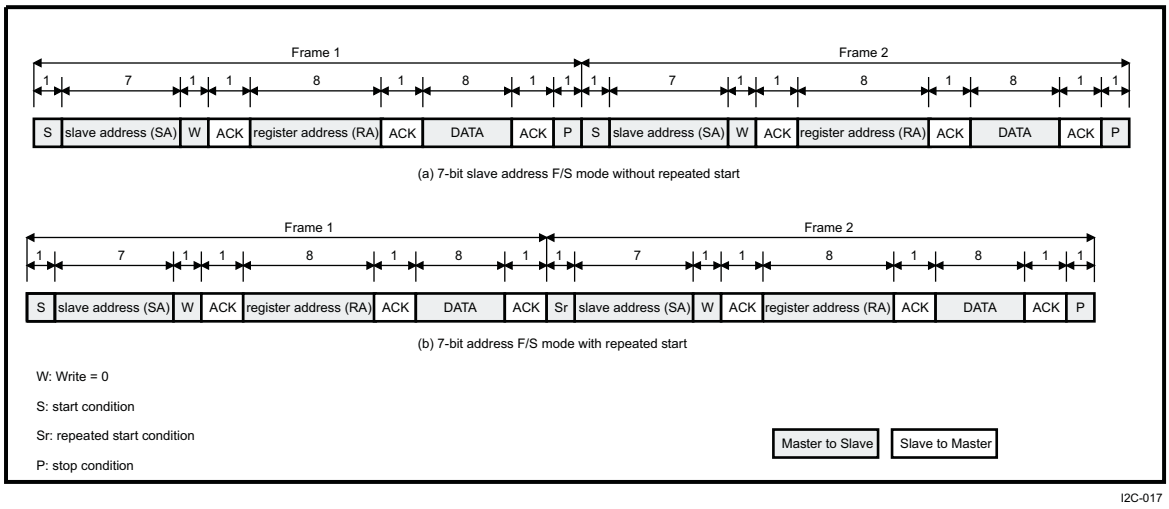
18.2.3.3.4 Addressing

The master transmitter HS I2C controller I2C4 supports only the 7-bit addressing mode. For each frame, the master writes the 8-bit value (DATA) in the register specified by the 8-bit register address (RA) of the slave addressed by the slave address (SA).

18.2.3.3.4.1 Data Transfer Format in F/S Mode

Figure 18-17 shows the I2C data transfer format in F/S mode for the I2C4 module.

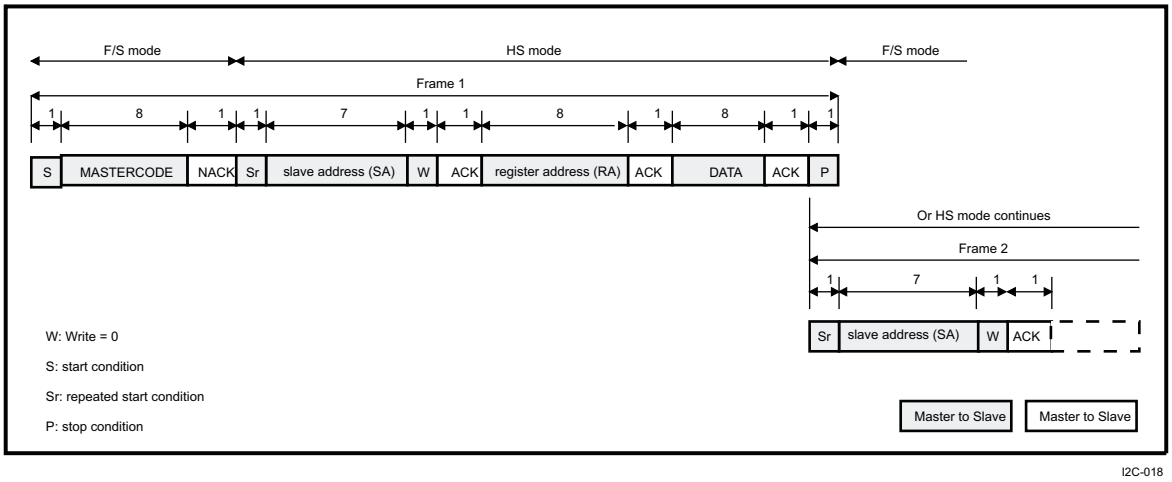
Figure 18-17. I2C Data Transfer Format in F/S Mode for the I2C4 Module



18.2.3.3.4.2 Data Transfer Format in HS Mode

Figure 18-18 shows the I²C data transfer format in HS mode for the I2C4 module.

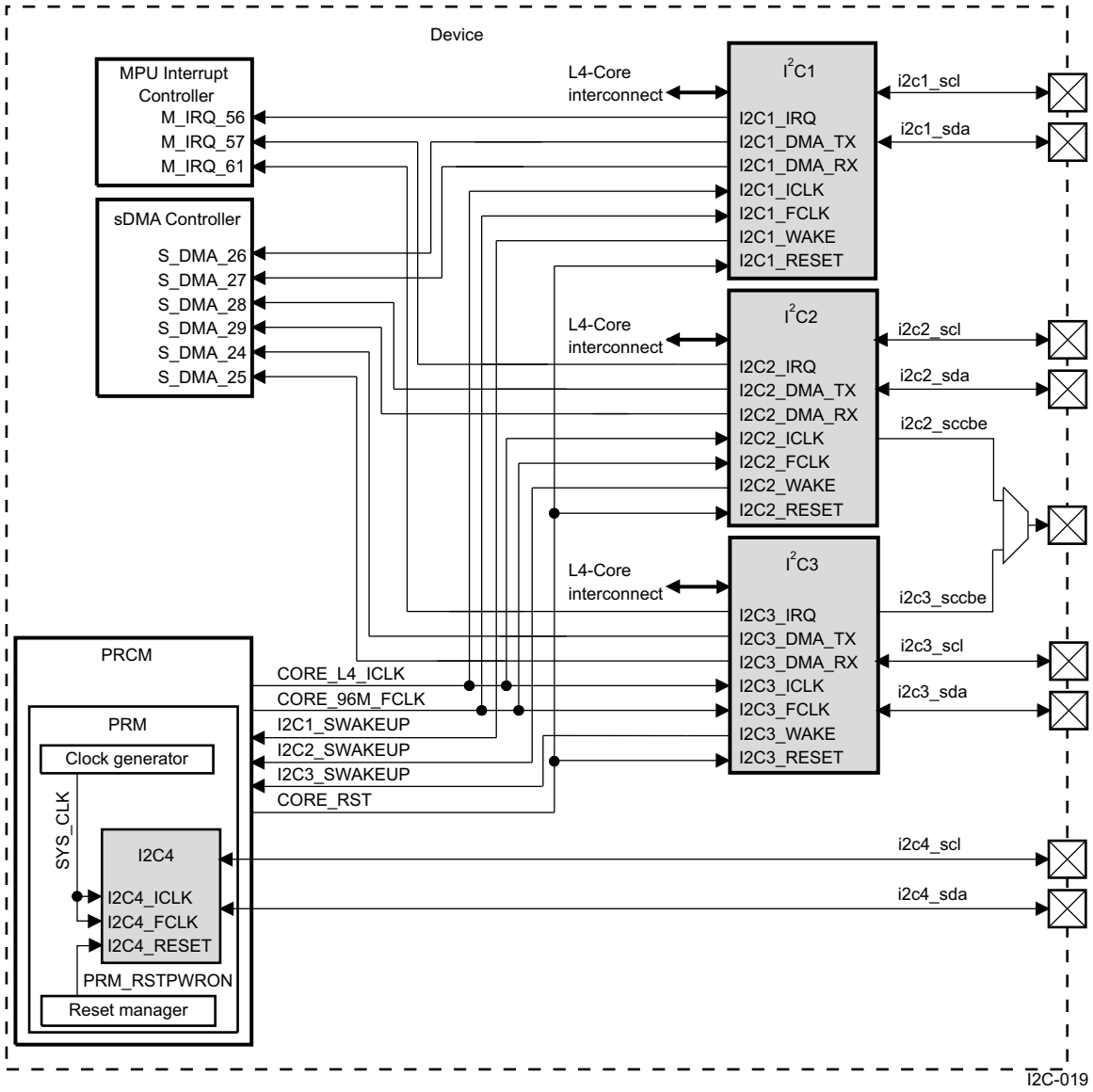
Figure 18-18. I²C Data Transfer Format in HS Mode for the I2C4 Module



18.3 High-Speed I²C Controller Integration

Figure 18-19 shows the integration of the four HS I²C controllers in the device.

Figure 18-19. HS I²C Controller Integration



18.3.1 Clocking, Reset, and Power-Management Scheme

18.3.1.1 Clocks

18.3.1.1.1 Module Clocks

Each multimaster HS I²C controller is clocked with an independent functional clock of 96 MHz (I2Ci_FCLK) and an interface clock (I2Ci_ICLK) for interfacing with the L4-Core interconnect. These clocks are provided by the PRCM module.

The SYS_CLK clock provided by the clock generator of the PRCM module is connected to the functional and interface clocks of the HS I²C controller I2C4. For detailed information about the module clocking, see Chapter 4, *Power, Reset, and Clock Management*.

The interface clock can be enabled or disabled for each multimaster HS I2C controller by setting the following bits in the PRCM module:

- PRCM.CM_ICLKEN1_CORE[15] EN_I2C1 bit for the I2C1 module
- PRCM.CM_ICLKEN1_CORE[16] EN_I2C2 bit for the I2C2 module
- PRCM.CM_ICLKEN1_CORE[17] EN_I2C3 bit for the I2C3 module

The functional clock can be enabled or disabled for each multimaster HS I2C controller by setting the following bits in the PRCM module:

- PRCM.CM_FCLKEN1_CORE[15] EN_I2C1 bit for the I2C1 module
- PRCM.CM_FCLKEN1_CORE[16] EN_I2C2 bit for the I2C2 module
- PRCM.CM_FCLKEN1_CORE[17] EN_I2C3 bit for the I2C3 module

The functional clock is processed by a prescaler block to produce the internal sampling clock. This clock is generated by the I2C prescaler block. The prescaler block consists of the I2Ci.I2C_PSC[7:0] PSC bit field (where i = 1, 2, 3) that is used to divide down the functional clock to obtain an internal sampling clock with a frequency value of I2Ci_FCLK/(I2Ci.I2C_PSC[7:0] PSC bit field value + 1, where i = 1, 2, 3).

Note: The I2C4.I2C_PSC[7:0] PSC bit field of the I2C4 module is not accessible by software. For details about the bit rates available for the I2C4 module, see Section 18.4.7.

18.3.1.2 Power Management

18.3.1.2.1 Module Power Saving

This section describes power-saving techniques for the multimaster HS I2C controllers. To conserve power, when no activity is detected on the L4-Core interconnect interface of the module, each of these modules supports an automatic idle mode that is enabled or disabled by setting the I2Ci.I2C_SYSC[0] AUTOIDLE bit.

When this bit is asserted (set to 1), if no activity is detected on the L4-Core interface, the automatic idle mode is enabled and the interface clock I2Ci_ICLK is disabled internally to the module, thus reducing power consumption.

When new activity is detected on the L4-Core interconnect interface of the module, the clock restarts with no latency penalty. After reset, the automatic idle mode is disabled; thus, this mode must be enabled by software for reduced power consumption.

18.3.1.2.2 System Power Management

As part of the system-wide power-management scheme, each multimaster HS I2C controller supports a communication protocol with the PRCM module to request the module to enter a low-power mode. When a module acknowledges a low-power mode request from the PRCM module, the interface and/or the functional clocks are gated off at the PRCM clock generator. Because the clocks are disabled at the source, the low-power mode offers lower power consumption than the internal clock autogating method used by local power management.

The PRCM.CM_ICLKEN1_CORE[15] EN_I2C1, PRCM.CM_ICLKEN1_CORE[16] EN_I2C2, and PRCM.CM_ICLKEN1_CORE[17] EN_I2C3 bits in the PRCM module control the interface clocks of the I2C1, I2C2, and I2C3 modules, respectively.

The PRCM.CM_FCLKEN1_CORE[15] EN_I2C1, PRCM.CM_FCLKEN1_CORE[16] EN_I2C2, and PRCM.CM_FCLKEN1_CORE[17] EN_I2C3 bits in the PRCM module control the functional clocks of the I2C1, I2C2, and I2C3 modules, respectively.

For details about clock enabling/disabling in the PRCM module, see Chapter 4, *Power, Reset, and Clock Management*.

Each multimaster HS I2C controller can be configured through the I2Ci.I2C_SYSC[4:3] IDLEMODE bit field as one of the following acknowledgment modes:

- Force-idle mode (I2Ci.I2C_SYSC[4:3] IDLEMODE bit field = b00): The module immediately enters idle mode on receiving a low-power-mode request from the PRCM module. In this mode, the software must ensure that there are no asserted output interrupts before requesting this mode to go to idle state.
- No-idle mode (I2Ci.I2C_SYSC[4:3] IDLEMODE bit field = b01): The module never enters idle mode.
- Smart-idle mode (I2Ci.I2C_SYSC[4:3] IDLEMODE bit field = b10): After receiving a low-power-mode request from the PRCM module, the module enters idle mode only after all asserted output interrupts are acknowledged and there is no pending internal event.

Note: The value I2Ci.I2C_SYSC[4:3] IDLEMODE bit field = b11 must not be used.

Table 18-4 describes the multimaster HS I2C controller power management modes.

Table 18-4. Multimaster HS I2C Controller Power Management Modes

Power management mode requested by the PRCM module	I2Ci.I2C_SYSC[4:3] IDLEMODE bit field value (where i = 1, 2, 3)
Force-idle	b00
No-idle	b01
Smart-idle	b10
Reserved (not used)	b11

The PRCM module gates the interface and/or the functional clocks after receiving the acknowledgment of the I2Ci module (where i = 1, 2, 3). The I2Ci.I2C_SYSC[9:8] CLOCKACTIVITY bit field indicates the state of the interface and functional clocks of the module when in idle mode. Table 18-5 lists the value of the I2Ci.I2C_SYSC[9:8] CLOCKACTIVITY bit field and indicates the state of the interface and functional clocks at the PRCM clock generator output in idle mode.

Table 18-5. State of the Interface and Functional Clocks When the Module is in Idle Mode

I2Ci.I2C_SYSC[9:8] CLOCKACTIVITY bit field value (where i= 1, 2, 3)	Functional Clock	Interface Clock
b00	OFF	OFF
b01	OFF	ON
b10	ON	OFF
b11	ON	ON

Note: The PRCM.CM_AUTOIDLE1_CORE[15] AUTO_I2C1, PRCM.CM_AUTOIDLE1_CORE[16] AUTO_I2C2, and PRCM.CM_AUTOIDLE1[17] AUTO_I2C3 bits control, for each multimaster HS I2C controller, whether the L4-Core interconnect interface clock is enabled or disabled in synchronization with the CORE power domain state transition (see Chapter 4, *Power, Reset, and Clock Management*).

Note: The voltage controller, in which the HS I2C controller I2C4 is implemented, has no idle request/acknowledge mechanism. The idle modes for the voltage controller are directly managed by the PRM module. For details, see Chapter 4, *Power, Reset, and Clock Management*.

18.3.1.2.3 Wake-Up Capability

Each multimaster HS I²C controller I2Ci can wake up the system by generating a wake-up request through the I2Ci_WAKE signal connected to the PRCM module.

The wake-up request is composed of the merge of all wake-up events. Each wake-up event can be separately enabled or disabled by setting the corresponding bit in the I2Ci.I2C_WE register.

The global wake-up capability of the module can be enabled or disabled by setting the I2Ci.I2C_SYSC[2] ENAWAKEUP bit (1: enabled, 0: disabled).

Figure 18-20 shows the wake-up generation flow.

Figure 18-20. Wake-up Generation Flow

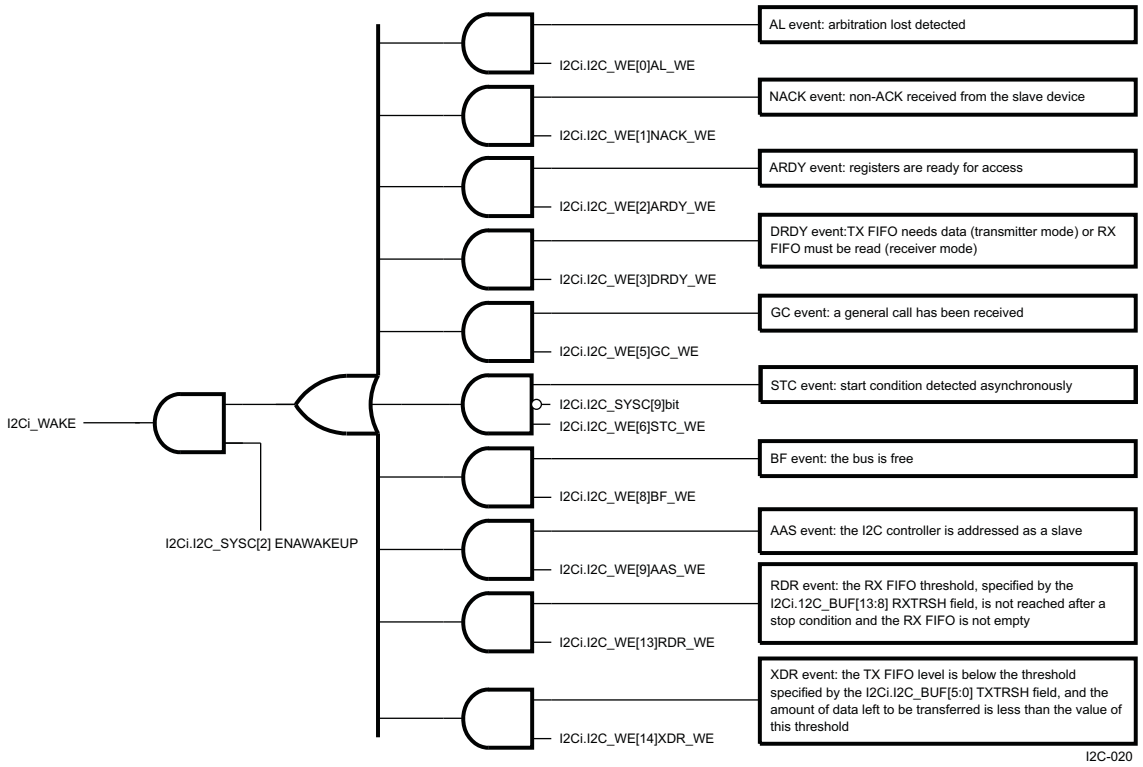


Table 18-6 lists all wake-up events with the corresponding enable/disable bit in the I2Ci.I2C_WE register.

Table 18-6. Wake-up Events

Wake-up Event Name	Supported Configuration Mode	Enable/Disable bit(1)	Event Generated When:
AL event	I²C mode only	I2Ci.I2C_WE[0] AL_WE	The I2Ci module in the device loses the arbitration of the I²C bus in master transmitter mode.
NACK event	I²C mode only	I2Ci.I2C_WE[1] NACK_WE	A nonacknowledgment has been generated on the I²C bus, indicating a transmission error.
ARDY event	I²C receive mode and SCCB read mode	I2Ci.I2C_WE[2] ARDY_WE	The current transaction is finished and the module registers can be accessed.
DRDY event	I²C and SCCB modes	I2Ci.I2C_WE[3] DRDY_WE	The TX FIFO needs some data to be transferred or the RX FIFO must be read.
GC event	I²C mode only	I2Ci.I2C_WE[5] GC_WE	A general call is received on the I²C bus.

(1) To enable the corresponding wake-up event generation, set the bit to 1. To disable the corresponding wake-up event generation, set the bit to 0.

Table 18-6. Wake-up Events (continued)

Wake-up Event Name	Supported Configuration Mode	Enable/Disable bit(1)	Event Generated When:
STC event (2)	I2C mode only	I2Ci.I2C_WE[6] STC_WE	A start (S) condition is detected on the I2C bus.(3)
BF event	I2C and SCCB modes	I2Ci.I2C_WE[8] BF_WE	The bus is free and the LH can initiate its own transfer.
AAS event	I2C mode only	I2Ci.I2C_WE[9] AAS_WE	An external master I2C device addresses the module of the device to inform the LH that it can check which of its own addresses was used by the external master I2C device to access the module of the device.
RDR event	I2C receive mode only	I2Ci.I2C_WE[13] RDR_WE	The module, configured as a receiver, has detected a stop (P) condition on the I2C bus and the RX FIFO threshold (I2Ci.I2C_BUF[13:8] RTRSH field value + 1) is not reached and the RX FIFO is not empty. This allows the module to inform the LH that it can check the amount of data to be transferred from the RX FIFO.
XDR event	I2C master transmit mode only	I2Ci.I2C_WE[14] XDR_WE	The TXFIFO level is below the threshold (I2Ci.I2C_BUF[5:0] XTRSH field value + 1) and the amount of data left to be transferred is less than this threshold. This allows the module to inform the LH that it can check the amount of data to be written to the TX FIFO.

(2) Wake-up event asynchronously detected.
(3) This event must not be enabled if the functional clock cannot be disabled.

Note: With the exception of the STC event, the functional clock must be active for wake-up event detection to occur. The HS I2C controller I2C4 has no wake-up capability.

18.3.1.3 Resets

18.3.1.3.1 Hardware Reset

The three multimaster HS I2C controllers receive their reset signal CORE_RST (the reset signal of the CORE power domain) from the PRCM module.

The master transmitter HS I2C controller I2C4 gets its reset signal PRM_RSTPWRON from the reset manager in the PRCM module.

18.3.1.3.2 Software Reset

Each multimaster HS I2C controller supports the software reset by accessing the I2Ci.I2C_SYSC[1] SRST bit (1: reset, 0: normal mode).

The software reset status can be checked by accessing the I2Ci.I2C_SYSS[0] RDONE bit (1: reset is done, 0: reset is ongoing).

To do a software reset, the following steps must be done:

1. Ensure that the module is disabled (clear the I2Ci.I2C_CON[15] I2C_EN bit to 0).
2. Set the I2Ci.I2C_SYSC[1] SRST bit to 1.
3. Enable the module by setting I2Ci.I2C_CON[15] I2C_EN bit to 1.
4. Check the I2Ci.I2C_SYSS[0] RDONE bit until it is set to 1 to indicate the software reset is complete.

Note: The I2Ci.I2C_CON[15] I2C_EN bit can hold the functional clock domain of the multimaster HS I2C controller in reset after the device reset has been released. When the system bus reset is removed, this bit remains cleared. The functional part of the I2C controller is held in reset state while this bit is 0, and all configuration registers can be accessed.

The I2Ci.I2C_CON[15] I2C_EN bit must be set to 1 to enable the functional part of the I2C controller.

The I2Ci.I2C_SYSS[0] RDONE bit is asserted only after the module is enabled by setting the I2Ci.I2C_CON[15] I2C_EN bit to 1.

18.3.1.4 Power Domain

The three multimaster HS I2C controllers are connected to the CORE power domain, whereas the HS I2C controller I2C4 belongs to the WKUP power domain.

18.3.2 Hardware Requests

18.3.2.1 DMA Requests

Each multimaster HS I2C controller can generate two DMA requests to the system DMA (sDMA) controller. Table 18-7 lists the DMA requests with mapping on the sDMA controller.

Table 18-7. Multimaster HS I2C Controller DMA Requests

Name	Source	Destination (sDMA controller)	Description
I2C1_DMA_TX	I2C1	S_DMA_26	I2C1 DMA write request to inform the sDMA to write new data in the I2C1.I2C_DATA[7:0] register
I2C1_DMA_RX	I2C1	S_DMA_27	I2C1 DMA read request to inform the sDMA to read the data in the I2C1.I2C_DATA[7:0] register
I2C2_DMA_TX	I2C2	S_DMA_28	I2C2 DMA write request to inform the sDMA to write new data in the I2C2.I2C_DATA[7:0] register
I2C2_DMA_RX	I2C2	S_DMA_29	I2C2 DMA read request to inform the sDMA to read the data in the I2C2.I2C_DATA[7:0] register
I2C3_DMA_TX	I2C3	S_DMA_24	I2C3 DMA write request to inform the sDMA to write new data in the I2C3.I2C_DATA[7:0] register
I2C3_DMA_RX	I2C3	S_DMA_25	I2C3 DMA read request to inform the sDMA to read the data in the I2C3.I2C_DATA[7:0] register

Note: The HS I2C controller I2C4 does not generate any DMA request.

18.3.2.2 Interrupt Requests

Each multimaster HS I2C controller can generate an interrupt I2Ci_IRQ to the MPU subsystem. Table 18-8 lists the interrupt requests with the mapping on the MPU interrupt controller.

Table 18-8. Multimaster HS I2C Controller Interrupt Requests

Name	Source	Destination (MPU interrupt controller)
I2C1_IRQ	I2C1	M_IRQ_56
I2C2_IRQ	I2C2	M_IRQ_57
I2C3_IRQ	I2C3	M_IRQ_61

An event can generate an interrupt request when the corresponding mask bit in the I2Ci.I2C_IE register is set to 1. Table 18-9 summarizes the events causing the generation of an interrupt request.

Table 18-9. Multimaster HS I²C Controller Interrupt Events

Event name	Supported configuration mode	Status bit	Mask bit	This event happens when:
AL event	I²C mode only	I2Ci.I2C_STAT [0] AL	I2Ci.I2C_IE[0] AL_IE	Two or more I²C master devices initiate a transfer and the I²C module I2Ci in the device loses arbitration of the I²C bus.
NACK event	I²C mode only	I2Ci.I2C_STAT [1] NACK	I2Ci.I2C_IE[1] NACK_IE	A nonacknowledgment has been received. In master mode, the transfer is automatically ended by generating a stop condition on the bus. The I2Ci.I2C_CON[1] STP, I2Ci.I2C_CON[10] MST and I2Ci.I2C_CON[9] TRX bits are automatically cleared to 0 (slave receiver mode). TX and RX FIFOs must be cleared (I2Ci.I2C_BUF[6] TXFIFO_CLR and I2Ci.I2C_BUF[14] RXFIFO_CLR bits set to 1).
ARDY event	I²C and SCCB modes	I2Ci.I2C_STAT [2] ARDY	I2Ci.I2C_IE[2] ARDY_IE	One of the following cases occurs: • In I²C master receiver mode, the I2Ci.I2C_CON[1] STP bit is set to 1, the I2Ci.I2C_CNT[15:0] DCOUNT field value is 0, and the RX FIFO is empty. • In I²C master transmitter mode, the I2Ci.I2C_CON[1] STP bit is set to 1 and the I2Ci.I2C_CNT[15:0] DCOUNT field value is 0. • In I²C master transmitter mode, the I2Ci.I2C_CON[1] STP bit is cleared to 0 and the I2Ci.I2C_CNT[15:0] DCOUNT field value passed 0. • In I²C master receiver mode, the I2Ci.I2C_CON[1] STP bit is set to 1, the I2Ci.I2C_CNT[15:0] DCOUNT field value passed 0, and the RX FIFO is empty. • In I²C slave transmitter mode, a stop or start (S) condition is received from the external I²C master device. • In I²C slave receiver mode, a stop or start (S) condition is received from the external I²C master device and the RX FIFO is empty. • In SCCB master transmitter or receiver mode, a stop (P) condition is detected.
RRDY event	I²C receive mode and SCCB read mode	I2Ci.I2C_STAT [3] RRDY	I2Ci.I2C_IE[3] RRDY_IE	The RX FIFO level is above the threshold (I2Ci.I2C_BUF[13:8] RTRSH field value + 1).
XRDY event	I²C transmit mode and SCCB write mode	I2Ci.I2C_STAT [4] XRDY	I2Ci.I2C_IE[4] XRDY_IE	The module requires new data to be served. A master transmitter module requests new data when the TX FIFO level is below the threshold (I2Ci.I2C_BUF[5:0] XTRSH field value + 1) and the required amount of data to be transmitted specified by the I2Ci.I2C_BUFSTAT[5:0] TXSTAT field is greater than the threshold. A slave transmitter requests new data when the TX FIFO is below the threshold (if the I2Ci.I2C_BUF[5:0] XTRSH field value is greater than 1), or when there is a read request from the external master device (for each acknowledge received from the master) when the I2Ci.I2C_BUF[5:0] XTRSH field value is 1.
GC event	I²C mode only	I2Ci.I2C_STAT [5] GC	I2Ci.I2C_IE[5] GC_IE	The module detects a general call on the I²C bus (all bits of the address cleared to 0).
STC event	I²C mode only	I2Ci.I2C_STAT [6] STC	I2Ci.I2C_IE[6] STC_IE	The module is in idle mode and a start (S) condition is asynchronously detected on the I²C bus and signaled with a wakeup. When the active mode is restored and the interrupt generated, this bit indicates the reason for the wakeup. See (1).

(1) The interrupt request generation on an STC event must be enabled only if the module is configured to allow the possibility of switching off the functional clock while in idle state (the I2Ci.I2C_SYSC[9:8] CLOCKACTIVITY field set to b00 or b01). The first transfer (corresponding to the detected start (S) condition) is lost, and is used only to restore the active mode of the module. On the I²C bus, the external master that generated the transfer detects this behavior as a nonacknowledge to the address phase and may possibly restart the transfer.

Table 18-9. Multimaster HS I2C Controller Interrupt Events (continued)

Event name	Supported configuration mode	Status bit	Mask bit	This event happens when:
AERR event	I2C and SCCB modes	I2Ci.I2C_STAT [7] AERR	I2Ci.I2C_IE[7] AERR_IE	A write access to the I2Ci.I2C_DATA register by the LH through the L4-Core interconnect is performed while the TX FIFO is full or a read access to the I2Ci.I2C_DATA register by the LH through the L4-Core interconnect is performed while the RX FIFO is empty. When the RX FIFO is empty, the read of the RX FIFO returns the previous read data value. When the TX FIFO is full, a write in the TX FIFO is ignored.
BF event	I2C and SCCB modes	I2Ci.I2C_STAT [8] BF	I2Ci.I2C_IE[8] BF_IE	The I2C bus becomes free (after a transfer is ended on the bus and a stop (P) condition is detected).
AAS event	I2C mode only	I2Ci.I2C_STAT [9] AAS	I2Ci.I2C_IE[9] AAS_IE	The module has recognized its own slave address or an alternate Own Address, or a general call (all address bits cleared to 0).
RDR event	I2C receive mode only	I2Ci.I2C_STAT [13] RDR	I2Ci.I2C_IE[13] RDR_IE	The module is configured as a receiver, a stop (P) condition was received on the I2C bus, and the RX FIFO level is below the threshold (I2Ci.I2C_BUF[13:8] RTRSH field value + 1).
XDR event	I2C master transmit mode only	I2Ci.I2C_STAT [14] XDR	I2Ci.I2C_IE[14] XDR_IE	The module is configured as a master transmitter, the TX FIFO level is below the threshold (I2Ci.I2C_BUF[5:0] XTRSH field value + 1), and the amount of data still to be transferred is less than this threshold.

When an interrupt request is generated, the software must read the I2Ci.I2C_STAT register to check which event caused the interrupt request generation, process accordingly, and acknowledge each processed event by writing 1 to the corresponding bit in the I2Ci.I2C_STAT register.

Note: The I2Ci.I2C_STAT[9] AAS status bit, corresponding to the AAS event, can be cleared in two ways:

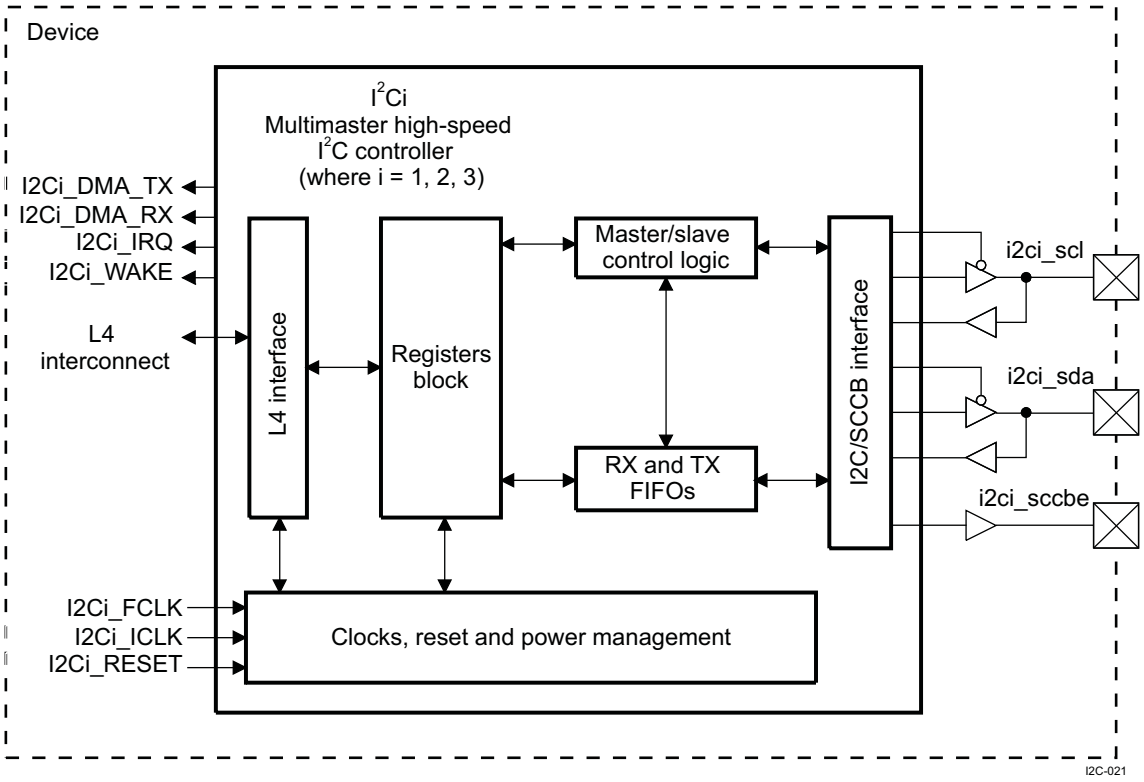
- If the I2Ci.I2C_IE[9] AAS_IE bit is set to 1 (interrupt generation enabled), the status bit is cleared by writing 1 to the I2Ci.I2C_STAT[9] AAS status bit
- If the I2Ci.I2C_IE[9] AAS_IE bit is cleared to 0 (interrupt generation disabled), the status bit is cleared when a new start or stop (P) condition is detected on the I2C bus.

18.4 High-Speed I2C Controller Functional Description

18.4.1 Block Diagram

Figure 18-21 is a functional block diagram of the multimaster I2C HS controller.

Figure 18-21. Multimaster HS I2C Controller Block Diagram



Note: The i2c1_sccbe signal is not available.

The three multimaster HS I2C controllers can be configured in F/S I2C mode, in HS I2C mode, or in SCCB mode. The operation mode is selected by configuring the I2Ci.I2C_CON[13:12] OPMODE field. Table 18-10 lists the available operation modes.

Table 18-10. Operation Mode Selection

Operation Mode	I2Ci.I2C_CON[13:12] OPMODE field value
Fast/standard (F/S) I2C	0x0
High-speed (HS) I2C	0x1
SCCB	0x2
Reserved (not used)	0x3

18.4.2 Transmit Mode in I2C Mode

This mode is available for master or slave. The master and slave modes are configurable with the I2Ci.I2C_CON[10] MST bit (0: slave mode, 1: master mode).

In master mode, the transmit mode is configured by setting the I2Ci.I2C_CON[9] TRX bit to 1. The MPU subsystem puts the data to transmit in the TX FIFO by writing to the I2Ci.I2C_DATA[7:0] DATA bit field.

The transmitter can write new data to this register when the I2Ci.I2C_STAT[4] XRDY bit is set to 1, or when the I2Ci.I2C_STAT[14] XDR bit is set to 1 according to the draining mechanism description.

A master transmitter requests new data if the I2Ci.I2C_CNT[15:0] DCOUNT bit field value is not 0. A slave transmitter requests new data if a read is performed by the external master.

Note: The HS I2C controller I2C4 is configured in master transmitter mode, and its configuration cannot be changed.

18.4.3 Receive Mode in I2C Mode

This mode is available for master or slave. In master mode, it is configured by clearing the I2Ci.I2C_CON[9] TRX bit to 0.

The MPU subsystem can read new data from the I2Ci.I2C_DATA[15:0] DCOUNT bit field when the I2Ci.I2C_STAT[3] RRDY bit is set to 1, or when the I2Ci.I2C_STAT[13] RDR bit is set according to the draining mechanism description.

Each time the I2Ci.I2C_STAT[3] RRDY bit is set to 1, if the interrupt is enabled (the I2Ci.I2C_IE[3] RRDY_IE bit must be set to 1), the I2C controller generates an interrupt to the MPU subsystem.

Note: In interrupt mode, the MPU subsystem must read this bit after each read in the I2Ci.I2C_DATA register to ensure that no other data is waiting for the FIFO to be read. For a new interrupt to be received, the I2Ci.I2C_STAT[3] RRDY bit must be cleared in the interrupt routine.

If the DMA receive mode is enabled (the I2Ci.I2C_BUF[15] RDMA_EN bit is set), this bit is forced to 0 and no interrupt is generated; instead, a DMA RX request to the sDMA controller is generated.

18.4.4 FIFO Management

Each multimaster HS I2C controller implements two internal 8-bit FIFOs, the RX and TX FIFOs.

Table 18-11 lists the depth of these FIFOs, depending on the instance of the I2C controller.

Table 18-11. RX and TX FIFO Depths

I2C Controller Instance	RX and TX FIFO Depth (in bytes)
I2C1	8
I2C2	8
I2C3	64

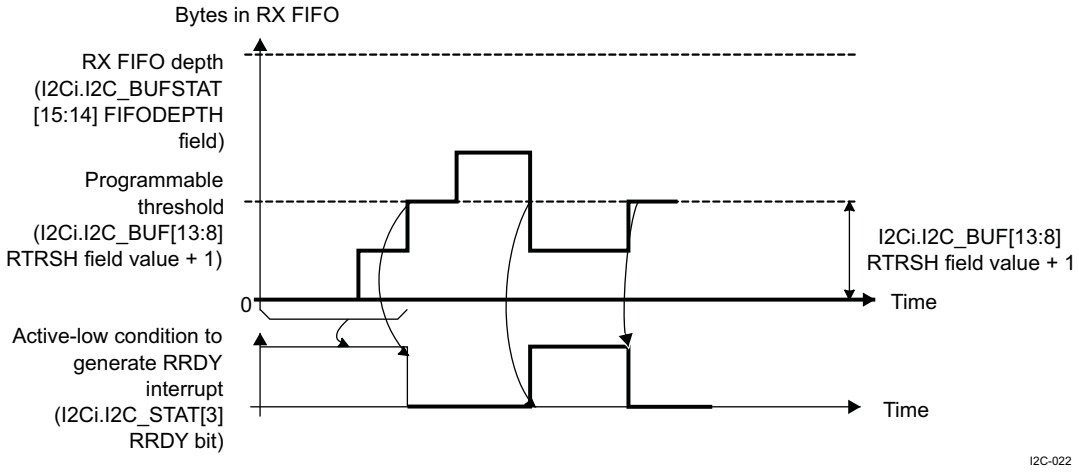
The depth of the RX and TX FIFOs can be checked by reading the I2Ci.I2C_BUFSTAT[15:14] FIFODEPTH bit field (0x0: 8 bytes, 0x1: 16 bytes, 0x2: 32 bytes, and 0x3: 64 bytes).

18.4.4.1 FIFO Interrupt Mode Operation

In FIFO interrupt mode (relevant interrupts enabled by the I2Ci.I2C_IE register), the processor is informed of the receiver and transmitter status by an interrupt signal. These interrupts are raised when the RX/TX FIFO thresholds (defined by the I2Ci.I2C_BUF[13:8] RTRSH field value + 1 for the RX FIFO or the I2Ci.I2C_BUF[5:0] XTRSH field value + 1 for the TX FIFO) are reached; the interrupt signals instruct the LH to transfer data to the destination (from the I2C controller module in receive mode and/or from any source to the I2C controller FIFO in transmit mode).

Figure 18-22 and Figure 18-23 show receive and transmit operations, respectively, from a FIFO management point of view.

Figure 18-22. Receive FIFO Interrupt Request Generation

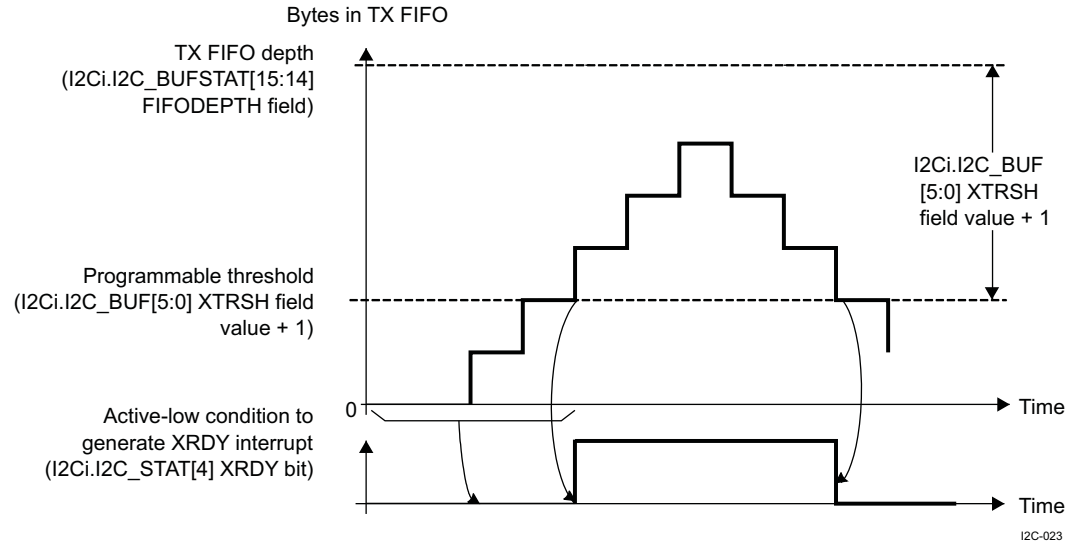


In Figure 18-22, the RRDY interrupt condition shows that the condition for generating an RRDY interrupt is achieved. The interrupt request is generated when this signal is active, and it can be cleared only by the LH by writing 1 in the I2Ci.I2C_STAT[3] RRDY bit. If the condition is still present after clearing the previous interrupt, another interrupt request is generated.

In receive mode, an RRDY interrupt is generated as soon as the FIFO reaches its receive threshold (I2Ci.I2C_BUF[13:8] RTRSH field value + 1). The interrupt can be deasserted only when the LH has handled enough bytes to make the number of bytes in the RX FIFO lower than the programmed threshold. For each interrupt, the LH can be configured to read a number of bytes equal to the value of the RX FIFO threshold.

Note: In SCCB mode, the RX and TX threshold values must be set to 1 by setting the I2Ci.I2C_BUF[13:8]RTRSH and I2Ci.I2C_BUF[5:0]XTRSH fields to 0x0.

Figure 18-23. Transmit FIFO Interrupt Request Generation



In Figure 18-23, the XRDY interrupt condition shows that the condition for generating an XRDY interrupt is achieved. The interrupt request is generated when TX FIFO is empty or when the TX FIFO threshold is not reached, and the LH can clear the XRDY status bit by writing 1 in the I2Ci.I2C_STAT[4] XRDY bit after transmitting the configured number of bytes. If the condition is still present after clearing the previous interrupt, another interrupt request is generated.

In interrupt mode, the module offers two options for the LH application to handle the interrupts:

- When detecting an interrupt request (XRDY/RRDY type), the LH can write/read 1 data byte to/from the TX/RX FIFO and then clear the interrupt. The module reasserts the interrupt until the interrupt condition is no longer met.
- When detecting an interrupt request (XRDY/RRDY type), the LH can be programmed to write/read the number of data bytes specified by the corresponding FIFO threshold (I2Ci.I2C_BUF[5:0] XTRSH field value + 1 for the TX threshold or I2Ci.I2C_BUF[13:8] RTRSH field value + 1 for the RX threshold). In this case, the interrupt condition is cleared and the next interrupt is asserted again when the XRDY/RRDY condition is met again.

If the second-interrupt-serving approach is used, an additional mechanism (draining feature) is implemented for cases where the transfer length is not a multiple of the FIFO threshold value (see Section 18.4.4.4).

Note: In slave transmit mode (the I2Ci.I2C_CON[10] MST bit is cleared and the I2Ci.I2C_CON[9] TRX bit is set to 1), the draining feature must not be used, because the transfer length is not known at configuration time, and the external master can end the transfer at any point by not acknowledging 1 data byte. If the draining feature is used in slave transmit mode, data can remain in the TX FIFO without being transmitted over the I²C bus. In this case, the TX FIFO must be cleared by setting the I2Ci.I2C_BUF[6] TXFIFO_CLR bit.

18.4.4.2 FIFO Polling Mode Operation

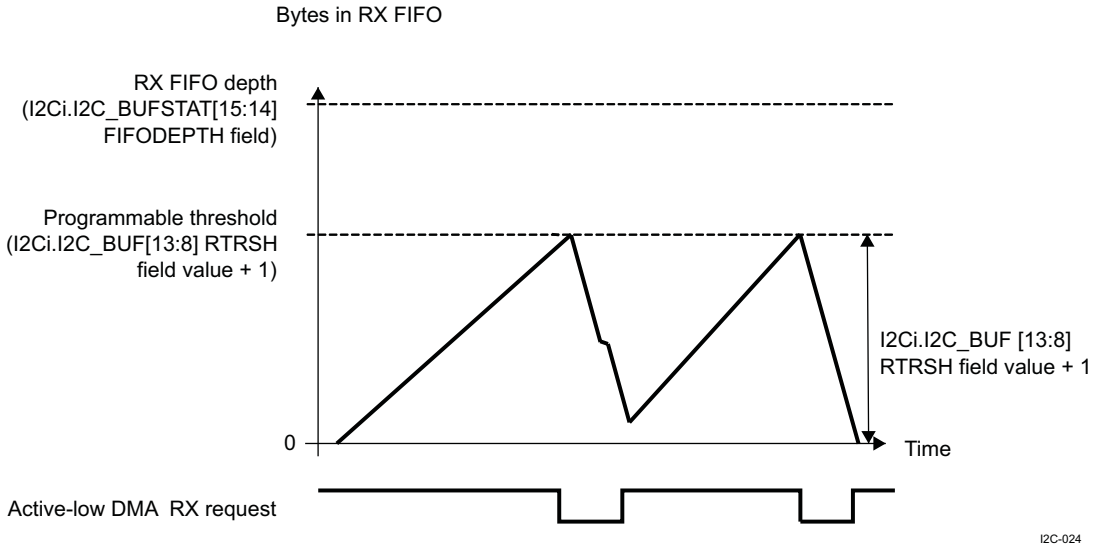
In FIFO polled mode (the I2Ci.I2C_IE[4] XRDY_IE and I2Ci.I2C_IE[3] RRDY_IE bits are disabled, and the I2Ci.I2C_BUF[15] RDMA_EN and I2Ci.I2C_BUF[7] XDMA_EN bits are disabled), the status of the module (receiver or transmitter) can be checked by polling the I2Ci.I2C_STAT[4] XRDY and the I2Ci.I2C_STAT[3] RRDY bits (the I2Ci.I2C_STAT[13] RDR and I2Ci.I2C_STAT[14] XDR bits can also be polled if the draining feature is enabled). The I2Ci.I2C_STAT[4] XRDY and I2Ci.I2C_STAT[3] RRDY bits accurately reflect the interrupt conditions described in the discussion of the FIFO interrupt mode operation.

18.4.4.3 FIFO DMA Mode Operation (I²C Mode Only)

In receive mode, a DMA request is generated by the I2Ci_DMA_RX signal as soon as the RX FIFO exceeds its threshold level (I2Ci.I2C_BUF[13:8] RTRSH field value + 1). This request is deasserted when the number of bytes defined by the threshold level is read by the sDMA controller.

Figure 18-24 shows the DMA request generation in receive mode.

Figure 18-24. Receive FIFO DMA Request Generation



In transmit mode, a DMA request is automatically asserted by the I2Ci_DMA_TX signal when the TX FIFO is empty. This request is deasserted when the number of bytes (I2Ci.I2C_BUF[5:0] XTRSH field value + 1) is written in the FIFO by the sDMA controller. If an insufficient number of bytes is written, the DMA request remains active. Figure 18-25 and Figure 18-26 show the DMA TX transfers with different values for the I2Ci.I2C_BUF[5:0] XTRSH field.

Figure 18-25. Transmit FIFO Request Generation (High Threshold)

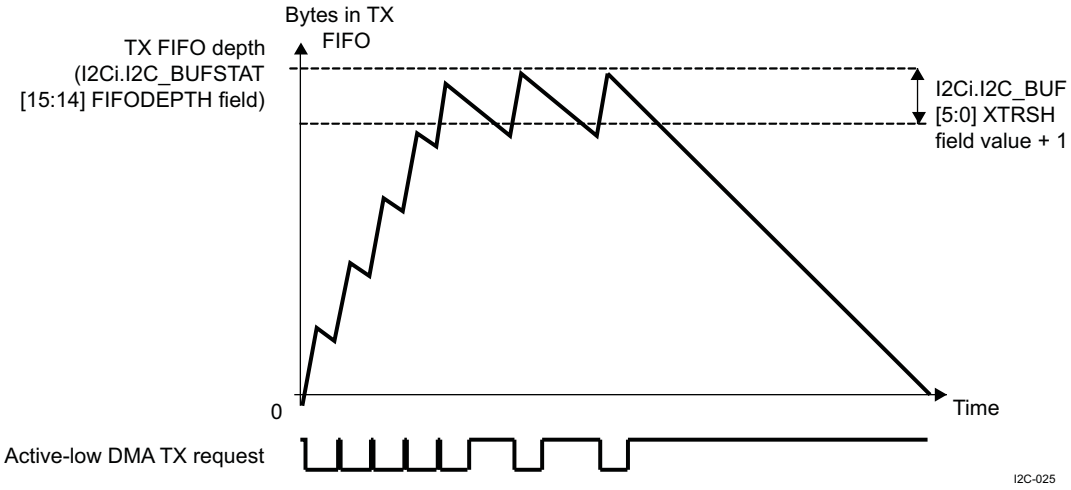
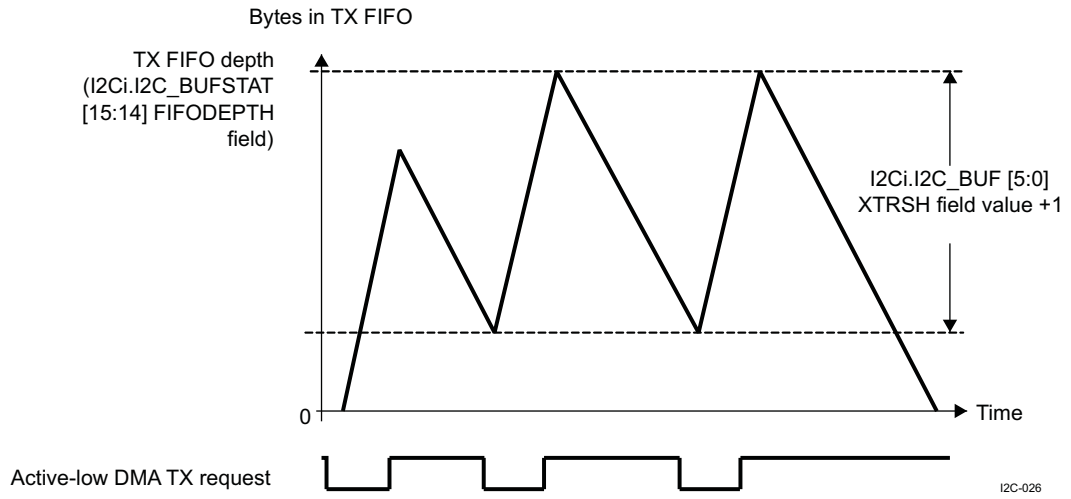


Figure 18-26. Transmit FIFO Request Generation (Low Threshold)



Note: In SCCB mode, the RX and TX threshold values must be set to 1 by setting the I2Ci.I2C_BUF[13:8]RTRSH and I2Ci.I2C_BUF[5:0]XTRSH fields to 0x0.

18.4.4.4 Draining Feature (I2C Mode Only)

The draining feature is implemented to handle the end of a transfer whose length is not a multiple of the FIFO threshold values (I2Ci.I2C_BUF[13:8] RTRSH field value + 1 for the RX threshold and I2Ci.I2C_BUF[5:0] XTRSH field value + 1 for the TX threshold). It also offers the possibility of transferring the remaining number of bytes (because the threshold is not reached).

This feature prevents the LH or the sDMA controller from trying more FIFO accesses than necessary (for example, to generate at the end of a transfer a DMA RX request having fewer bytes in the FIFO than the configured DMA transfer length). Otherwise, an AERR interrupt is generated by the I2Ci.I2C_STAT[7] AERR status bit.

The draining mechanism generates an interrupt using the I2Ci.I2C_STAT[13] RDR or the I2Ci.I2C_STAT[14] XDR status bit at the end of the transfer, informing the LH that it must check the amount of data left to be transferred (the I2Ci.I2C_BUFSTAT[13:8] RXSTAT or I2Ci.I2C_BUFSTAT[5:0] TXSTAT fields) and enable the draining feature of the DMA controller by reconfiguring the DMA transfer length according to this value (when the DMA mode is enabled) or perform only the required number of data accesses (when the DMA mode is disabled).

In receive mode (master or slave), if the RX FIFO threshold (I2Ci.I2C_BUF[13:8] RTRSH field value + 1) is not reached, but the transfer ends on the I2C bus and there is still data in the RX FIFO (less than the threshold), the receive draining interrupt (the I2Ci.I2C_STAT[13] RDR status bit) is asserted to inform the LH that it can read the amount of data in the FIFO (the I2Ci.I2C_BUFSTAT[13:8] RXSTAT field). The LH performs a number of data read accesses equal to the I2Ci.I2C_BUFSTAT[13:8] RXSTAT field value (interrupt or polling mode), or reconfigures the sDMA controller with the required value to drain the FIFO.

In master transmit mode, if the TX FIFO threshold (I2Ci.I2C_BUF[5:0] XTRSH field value + 1) is not reached, but the amount of data remaining to be written in the TX FIFO is less than the threshold, the transmit draining interrupt (the I2Ci.I2C_STAT[14] XDR status bit) is asserted to inform the LH that it can read the amount of data remaining to be written in the TX FIFO (the I2Ci.I2C_BUFSTAT[5:0] TXSTAT field). The LH must write the required number of data bytes specified by the I2Ci.I2C_BUFSTAT[5:0] TXSTAT field value or reconfigure the sDMA controller with the value required to transfer the last bytes to the FIFO.

In master mode, the LH can alternately skip the checking of the I2Ci.I2C_BUFSTAT[5:0] TXSTAT and I2Ci.I2C_BUFSTAT[13:8] RXSTAT field values, because it can obtain this information internally (by computing the the I2Ci.I2C_CNT[15:0] DATACOUNT field value modulo I2Ci.I2C_BUF[13:8] RTRSH or I2Ci.I2C_BUF[5:0] XTRSH).

By default, the draining feature is disabled; it can be enabled using the I2Ci.I2C_IE[14] XDR_IE or I2Ci.I2C_IE[13] RDR_IE bits (default disabled) only for transfers with lengths not equal to the threshold values (I2Ci.I2C_BUF[5:0] XTRSH field value + 1 for the TX threshold or I2Ci.I2C_BUF[13:8] RTRSH field value + 1 for the RX threshold).

18.4.5 Programmable Multislave Channel Feature (I2C Mode Only)

This feature allows each multimaster HS I2C controller to be addressed using four separate Own Addresses configured in the I2Ci.I2C_OAx registers (with x = 0, 1, 2, 3). An additional register (I2Ci.I2C_ACTOA) is used to indicate to the LH which address is used by the external master to communicate with the I2C controller.

Each Own Address can be independently configured in 7-bit or 10-bit mode by setting the corresponding bit (I2Ci.I2C_CON[7] XOA0, I2Ci.I2C_CON[6] XOA1, I2Ci.I2C_CON[5] XOA2, or I2Ci.I2C_CON[4] XOA3).

18.4.6 Automatic Blocking of the I2C Clock Feature (I2C Mode Only)

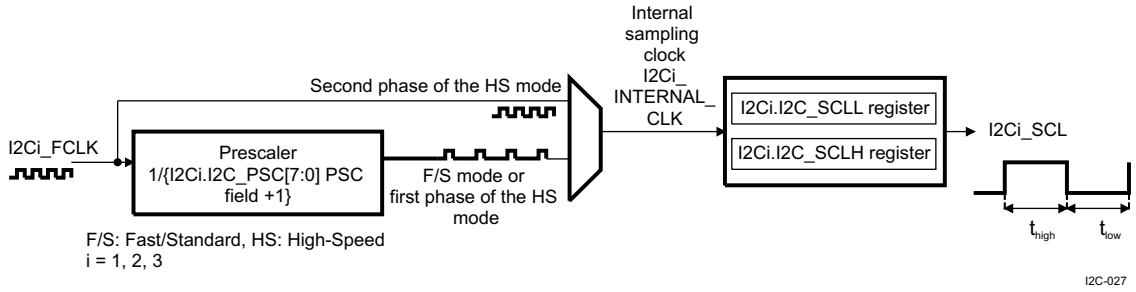
This feature offers the possibility for the LH to command the blocking of the I2C clock after the slave addressing phase, when the I2C controller is addressed by an external master device using a certain Own Address.

The release of the I2C clock (i2ci_scl signal, with i = 1, 2, 3) can be performed independently for each Own Address (I2Ci.I2C_OAi registers, with i = 0, 1, 2, 3) by deasserting the corresponding bit in the I2Ci.I2C_SBLOCK register.

18.4.7 Clocking

Figure 18-27 shows the I²C clock generation of the HS I²C controllers.

Figure 18-27. I²C Clock Generation



Each multimaster HS I²C controller uses the I2Ci_FCLK functional clock in the PRCM module. The internal sampling clock I2Ci_INTERNAL_CLK is generated by dividing the functional clock by (I2Ci.I2C_PSC[7:0] PSC field value + 1) in F/S mode, in SCCB mode, or in the first phase of HS mode; or by directly using the functional clock in the second phase of HS mode (prescaler is bypassed).

The low time of the I2Ci_SCL signal is determined by the I2Ci.I2C_SCLL[7:0] SCLL field in F/S mode, in SCCB mode, or in the first phase of HS mode; or by the I2Ci.I2C_SCLL[15:8] HSSCLL field in the second phase of HS mode.

The high time of the I2Ci_SCL signal is determined by the I2Ci.I2C_SCLH[7:0] SCLH field in F/S mode, in SCCB mode, or in the first phase of HS mode; or by the I2Ci.I2C_SCLH[15:8] HSSCLH field in the second phase of HS mode.

Table 18-12 lists the t_{LOW} and t_{high} values in master mode only (in slave mode, the I²C controller does not generate the I²C clock).

Table 18-12. t_{LOW} and t_{high} Values of the I²C Clock

Mode	I2Ci_INTERNAL_CLK	t_{LOW}	t_{high}
F/S, SCCB, or HS first phase	I2Ci_FCLK / (I2Ci.I2C_PSC[7:0] PSC field + 1)	(I2Ci.I2C_SCLL[7:0] SCLL field value + 7) x I2Ci_INTERNAL_CLK period	(I2Ci.I2C_SCLH[7:0] SCLH field value + 5) x I2Ci_INTERNAL_CLK period
HS second phase	I2Ci_FCLK	(I2Ci.I2C_SCLL[15:8] HSSCLL field value + 7) x I2Ci_INTERNAL_CLK period	(I2Ci.I2C_SCLH[15:8] HSSCLH field value + 5) x I2Ci_INTERNAL_CLK period

Note: The equations in Table 18-12 give the SCL timing values for SCLL/SCLH/HSSCLL/HSSCLH at HS I²C controller outputs. Actual tlow and thigh periods may vary, depending on the board (the load capacitance on the SCL signal). If necessary, any adjustments to the SCLL/SCLH/HSSCLL/HSSCLH values must be determined by measurements of actual SCL signal on the board

Note: For HS mode, both the I2Ci.I2C_SCLL[15:8] HSSCLL and I2Ci.I2C_SCLL[7:0] SCLL fields must be programmed (the first phase of an HS transaction is performed at F/S speed).

For HS mode, both the I2Ci.I2C_SCLH[15:8] HSSCLH and I2Ci.I2C_SCLH[7:0] SCLH fields must be programmed (the first phase of an HS transaction is performed at F/S speed).

CAUTION

During active mode (I2Ci.I2C_CON[15] I2C_EN bit is set to 1), make no changes to the I2Ci.I2C_SCLL and I2Ci.I2C_SCLH registers. Changes may result in unpredictable behavior.

Note: Each multimaster HS I2C controller can be used with an internal secondary pullup. This pullup is mandatory when the I2C controller is configured in HS mode for a bit rate of 3.4M bps, and the bus line capacitance exceeds 45 pF. Pullups can be programmed through the CONTROL.CONTROL_DEVCONF1[12] I2C1HSMASTER bit for I2C1, the CONTROL.CONTROL_DEVCONF1[13] I2C2HSMASTER bit for I2C2, and the CONTROL.CONTROL_DEVCONF1[14] I2C3HSMASTER bit for I2C3.

The maximum bit rate specified by the SCCB specifications is 100K b/s.

18.4.8 Noise Filter

The noise filter is used to suppress any noise that is 50 ns or less in case of F/S and SCCB operation modes, and any noise that is 10 ns or less in case of HS mode operation. The noise filter is always one period of the I2Ci_INTERNAL_CLK clock. This way, for HS mode operation (prescaler bypassed), the filter suppresses spikes of less than 10.4 ns.

For SCCB modes (for example, I2Ci.I2C_PSC[7:0] PSCbit field = 4), the maximum width of suppressed spikes is 52 ns.

To ensure correct filtering, the prescaler must be programmed accordingly by the 2Ci.I2C_PSC[7:0] PSC bit field.

18.4.9 System Test Mode

A system test mode is available for multimaster HS I2C controller module testing. This mode is enabled by setting the I2Ci.I2C_SYSTEST[15] ST_EN to 1. When this bit is cleared to 0, the I2C controller is configured in normal operation mode.

In system test mode, the I2Ci_SYSTEST[13:12] TMODE bit field selects the type of test. Table 18-13 lists the tests available for the multimaster HS I2C controllers.

Table 18-13. List of tests for the Multimaster HS I2C Controllers

I2Ci.I2C_SYSTEST[13:12] TMODE field value	Test	Description
b00	Functional mode	Normal operation mode
b01	Reserved (not used)	
b10	Test of i2ci_scl serial clock line	The i2ci_scl line is driven with a permanent clock as if mastered with the parameters set in the I2Ci.I2C_PSC, I2Ci.I2C_SCLL, and I2Ci.I2C_SCLH registers.
b11	Loop-back mode + i2ci_scl/ i2ci_sda/ i2ci_sccbe input/output	In the master transmit mode only, data transmitted out of the I2Ci.I2C_DATA register (write action) is received in the same I2Ci.I2C_DATA register through an internal path through the FIFO buffers. The DMA and interrupt requests are normally generated if they are enabled. Moreover, the i2ci_scl, i2ci_sda, and i2ci_sccbe lines are controlled with the I2Ci.I2C_SYSTEST[4:0] bits.

Note: When the I2Ci.I2C_SYSTEST[13:12] TMODE field = b11, the I2C controller must be configured in I2C F/S (I2Ci.I2C_CON[13:12] OPMODE = b00) or I2C HS mode (I2Ci.I2C_CON[13:12] OPMODE = b01). The loop-back mode is not available in SCCB mode (I2Ci.I2C_CON[13:12] OPMODE = b10).

Note: In normal operation mode (I2Ci.I2C_SYSTEST[15] ST_EN clear to 0), the I2Ci.I2C_SYSTEST[4:0] bits that control the i2ci_scl, i2ci_sda, and i2ci_sccbe lines in system test mode are read-only bits.

In system test mode (I2Ci.I2C_SYSTEST[15] ST_EN set to 1), the I2Ci.I2C_STAT[5:0] status bits can be set to 1 when the I2Ci.I2C_SYSTEST[11] SSB bit is set to 1. Clearing the I2Ci.I2C_SYSTEST[11] SSB bit to 0 does not clear the I2Ci.I2C_STAT[5:0] status bits to 0. The I2Ci.I2C_STAT[5:0] status bits can be cleared to 0 only by writing 1 in the corresponding bits.

18.4.10 Write and Read Operations in SCCB Mode

In SCCB mode, the multimaster HS I2C controller can write or read a single byte to or from the external SCCB device.

To write a single byte to the external SCCB device, the multimaster HS I2C controller must be configured in multimaster transmitter mode by setting the I2Ci.I2C_CON[10] MST and I2Ci.I2C_CON[9] TRX bits to 1. The external device slave address (7-bit address of the ID value) is set in the I2Ci.I2C_SA register; the register address (8-bit subaddress in the external SCCB device) is set in the I2Ci.I2C_OA0 register. The 8-bit data to be transmitted is written by the LH in the I2Ci.I2C_DATA register.

To read a single byte from the external SCCB device, the multimaster HS I2C controller must be configured in multimaster receiver mode by setting the I2Ci.I2C_CON[10] MST to 1 and by clearing the I2Ci.I2C_CON[9] TRX bit to 0. The external device slave address (7-bit address of the ID value) is set in the I2Ci.I2C_SA register; the register address (8-bit subaddress in the external SCCB device) is set in the I2Ci.I2C_OA0 register. The 8-bit data received from the external SCCB device is read by the LH from the I2Ci.I2C_DATA register.

Note: In SCCB mode, the RX and TX thresholds must be set to 1 by configuring the I2Ci.I2C_BUF[13:8] RTRSH and I2Ci.I2C_BUF[5:0]XTRSH fields to 0x0.

18.4.11 Power Chip Communication Operations

The master transmitter HS I2C controller I2C4 inside the voltage controller of the PRM module is used to send configuration commands from the LH to external power chip(s).

For voltage control operations, the LH must set the slave address of the first power chip in the PRCM.PRM_VC_SMPS_SA[6:0] SA0 field, and the slave address of the second power chip (if necessary) in the PRCM.PRM_VC_SMPS_SA[22:16] SA1 field.

The LH must also configure the specific registers in the voltage controller of the PRCM module, for the voltage control and power-sequencing functions.

A high-priority bypass mode is available to allow the LH to configure the external power chip(s) through the I2C bus. To write an 8-bit data to the configuration registers of an external power chip, the LH must set the 7-bit external power chip slave address in the PRCM.PRM_VC_BYPASS_VAL[6:0] SLAVEADDR field, the configuration register address in the PRCM.PRM_VC_BYPASS_VAL[15:8] REGADDR field, and the 8-bit data in the PRCM.PRM_BYPASS_VAL[23:16] register.

For details about voltage control, see Chapter 4, *Power, Reset, and Clock Management*.

18.5 High-Speed I2C Controller Basic Programming Model

18.5.1 Multimaster HS I2C Controller Basic Programming Model in I2C Mode

This section describes the programming model of the multimaster HS I2C controllers configured in I2C mode.

18.5.1.1 Main Program

18.5.1.1.1 Configure the Module Before Enabling the I2C Controller

Before enabling the I2C controller, perform the following steps:

1. Enable the functional and interface clocks (see Section 18.3.1.1.1).
2. Program the prescaler to obtain an approximately 12-MHz internal sampling clock (I2Ci_INTERNAL_CLK) by programming the corresponding value in the I2Ci.I2C_PSC[3:0] PSC field. This value depends on the frequency of the functional clock (I2Ci_FCLK). Because this frequency is 96MHz, the I2Ci.I2C_PSC[7:0] PSC field value is 0x7.
3. Program the I2Ci.I2C_SCLL[7:0] SCLL and I2Ci.I2C_SCLH[7:0] SCLH fields to obtain a bit rate of 100K bps or 400K bps. These values depend on the internal sampling clock frequency (see Table 18-12).
4. (Optional) Program the I2Ci.I2C_SCLL[15:8] HSSCLL and I2Ci.I2C_SCLH[15:8] HSSCLH fields to obtain a bit rate of 400K bps or 3.4M bps (for the second phase of HS mode). These values depend on the internal sampling clock frequency (see Table 18-12).
5. (Optional) If a bit rate of 3.4M bps is used and the bus line capacitance exceeds 45 pF, program the CONTROL.CONTROL_DEVCONF1[12] I2C1HSMASTER bit for I2C1, the CONTROL.CONTROL_DEVCONF1[13] I2C2HSMASTER bit for I2C2, or the CONTROL.CONTROL_DEVCONF1[14] I2C3HSMASTER bit for I2C3.
6. Configure the Own Address of the I2C controller by storing it in the I2Ci.I2C_OA0 register. Up to four Own Addresses can be programmed in the I2Ci.I2C_OAi registers (with i = 0, 1, 2, 3) for each I2C controller.

Note: For a 10-bit address, set the corresponding expand Own Address bit in the I2Ci.I2C_CON register.

7. Set the TX threshold (in transmitter mode) and the RX threshold (in receiver mode) by setting the I2Ci.I2C_BUF[5:0]XTRSH field to (TX threshold - 1) and the I2Ci.I2C_BUF[13:8]RTRSH field to (RX threshold - 1), where the TX and RX thresholds are greater than or equal to 1.
8. Take the I2C controller out of reset by setting the I2Ci.I2C_CON[15] I2C_EN bit to 1.

18.5.1.1.2 Initialize the I2C Controller

To initialize the I2C controller, perform the following steps:

1. Configure the I2Ci.I2C_CON register:
   - For master or slave mode, set the I2Ci.I2C_CON[10] MST bit (0: slave, 1: master).
   - For transmitter or receiver mode, set the I2Ci.I2C_CON[9] TRX bit (0: receiver, 1: transmitter).
2. If using an interrupt to transmit/receive data, set to 1 the corresponding bit in the I2Ci.I2C_IE register (the I2Ci.I2C_IE[4] XRDY_IE bit for the transmit interrupt, the I2Ci.I2C_IE[3] RRDY bit for the receive interrupt).
3. If using DMA to receive/transmit data, set to 1 the corresponding bit in the I2Ci.I2C_BUF register (the I2Ci.I2C_BUF[15] RDMA_EN bit for the receive DMA channel, the I2Ci.I2C_BUF[7] XDMA_EN bit for the transmit DMA channel).

18.5.1.1.3 Configure Slave Address and the Data Control Register

In master mode, configure the slave address register by programming the I2Ci.I2C_SA[9:0] SA field and the number of data bytes (I^2C data payload) associated with the transfer by programming the I2Ci.I2C_CNT[15:0] DCOUNT field.

Note: For a 10-bit address, set the I2Ci.I2C_CON[8] XSA bit to 1.

18.5.1.1.4 Initiate a Transfer

Poll the I2Ci.I2C_STAT[12] BB bit. If it is cleared to 0 (bus not busy), configure the I2Ci.I2C_CON[0] STT and I2Ci.I2C_CON[1] STP bits. To initiate a transfer, the I2Ci.I2C_CON[0] STT bit must be set to 1, and it is not mandatory to set the I2Ci.I2C_CON[1] STP bit to 1.

18.5.1.1.5 Receive Data

Poll the I2Ci.I2C_STAT[3] RRDY bit, or use the RRDY interrupt (the I2Ci.I2C_IE[3] RRDY_IE bit must be set to 1) or the DMA RX channel (the I2Ci.I2C_BUF[15] RDMA_EN bit must be set to 1) to read the receive data in the I2Ci.I2C_DATA register.

If the transfer length does not equal the RX FIFO threshold (I2Ci.I2C_BUF[13:8]RTRSH field + 1), use the draining feature (enable the RDR interrupt by setting the I2Ci.I2C_IE[13] RDR_IE bit to 1).

Note: In receive mode only, the I2Ci.I2C_STAT[11] ROVR (receive overrun) bit indicates whether the receiver has experienced overrun. An overrun condition occurs when the shift register and the RX FIFO are full. An overrun condition does not result in data loss; the I^2C controller simply holds the serial clock line i2ci_scl to low to prevent other bytes from being received.

The I2Ci.I2C_STAT[7] AERR bit is set to 1 when a read access is performed in the I2Ci.I2C_DATA register while the RX FIFO is empty. The corresponding interrupt can be enabled by setting the I2Ci.I2C_IE[7] AERR_IE bit to 1.

18.5.1.1.6 Transmit Data

Poll the I2Ci.I2C_STAT[4] XRDY bit, or use the XRDY interrupt (the I2Ci.I2C_IE[4] XRDY_IE bit must be set to 1) or the DMA TX channel (the I2Ci.I2C_BUF[7] XDMA_EN bit must be set to 1) to write data to the I2Ci.I2C_DATA register.

If the transfer length does not equal the TX FIFO threshold (I2Ci.I2C_BUF[5:0] XTRSH field + 1), use the draining feature (enable the XDR interrupt by setting the I2Ci.I2C_IE[14] XDR_IE bit to 1).

Note: In transmit mode only, the I2Ci.I2C_STAT[10] XUDF bit indicates whether the transmitter has experienced underflow.

In master transmit mode, underflow occurs when the shift register and the TX FIFO are empty and there are still some bytes to transmit (the I2Ci.I2C_CNT[15:0] DCOUNT field value is not 0).

In slave transmit mode, underflow occurs when the shift register and the TX FIFO are empty and the external I^2C master device still requests data bytes to be read.

The I2Ci.I2C_STAT[7] AERR bit is set to 1 when a write access is performed in the I2Ci.I2C_DATA register while the TX FIFO is full. The corresponding interrupt can be enabled by setting the I2Ci.I2C_IE[7] AERR_IE bit to 1.

18.5.1.2 Interrupt Subroutine Sequence

1. Test for arbitration lost (I2Ci.I2C_STAT[0] AL status bit) and resolve accordingly.
2. Test for no acknowledgment (I2Ci.I2C_STAT[1] NACK status bit) and resolve accordingly.
3. Test for register access ready (I2Ci.I2C_STAT[2] ARDY status bit) and resolve accordingly.
4. Test for receive data ready (I2Ci.I2C_STAT[3] RRDY status bit) and resolve accordingly.
5. Test for transmit data ready (I2Ci.I2C_STAT[4] XRDY status bit) and resolve accordingly.
6. Test for general call (I2Ci.I2C_STAT[5] GC status bit) and resolve accordingly.
7. Test for start (S) condition (I2Ci.I2C_STAT[6] STC status bit) and resolve accordingly. For this test, the functional clock must be inactive.
8. Test for access error (I2Ci.I2C_STAT[7] AERR status bit) and resolve accordingly.
9. Test for bus free (I2Ci.I2C_STAT[8] BF status bit) and resolve accordingly.

18.5.1.3 Programming Flow Diagrams

Figure 18-28 through Figure 18-36 are procedure flowcharts for programming the F/S and HS I2C modes.

Figure 18-28. I2C Setup Procedure

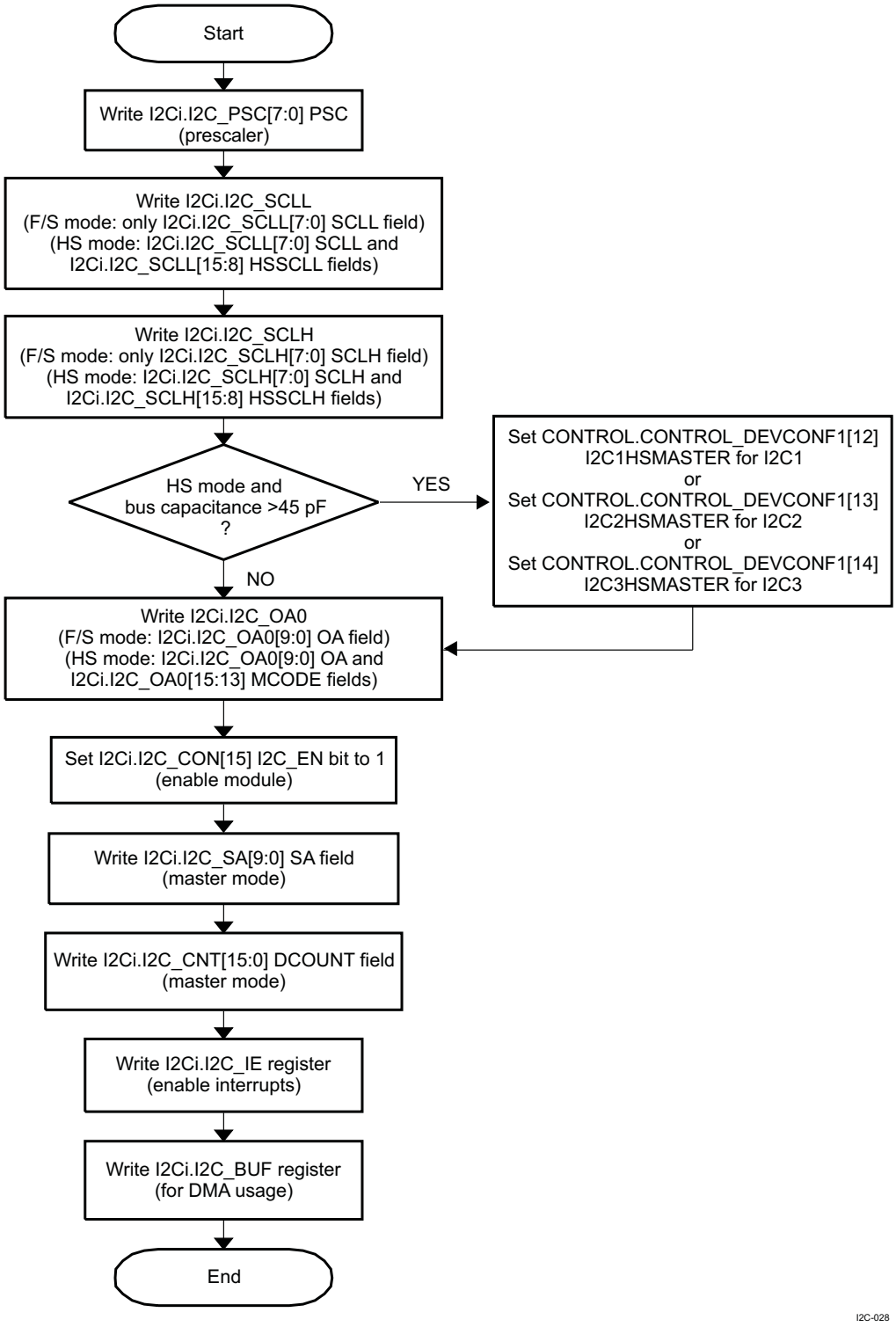
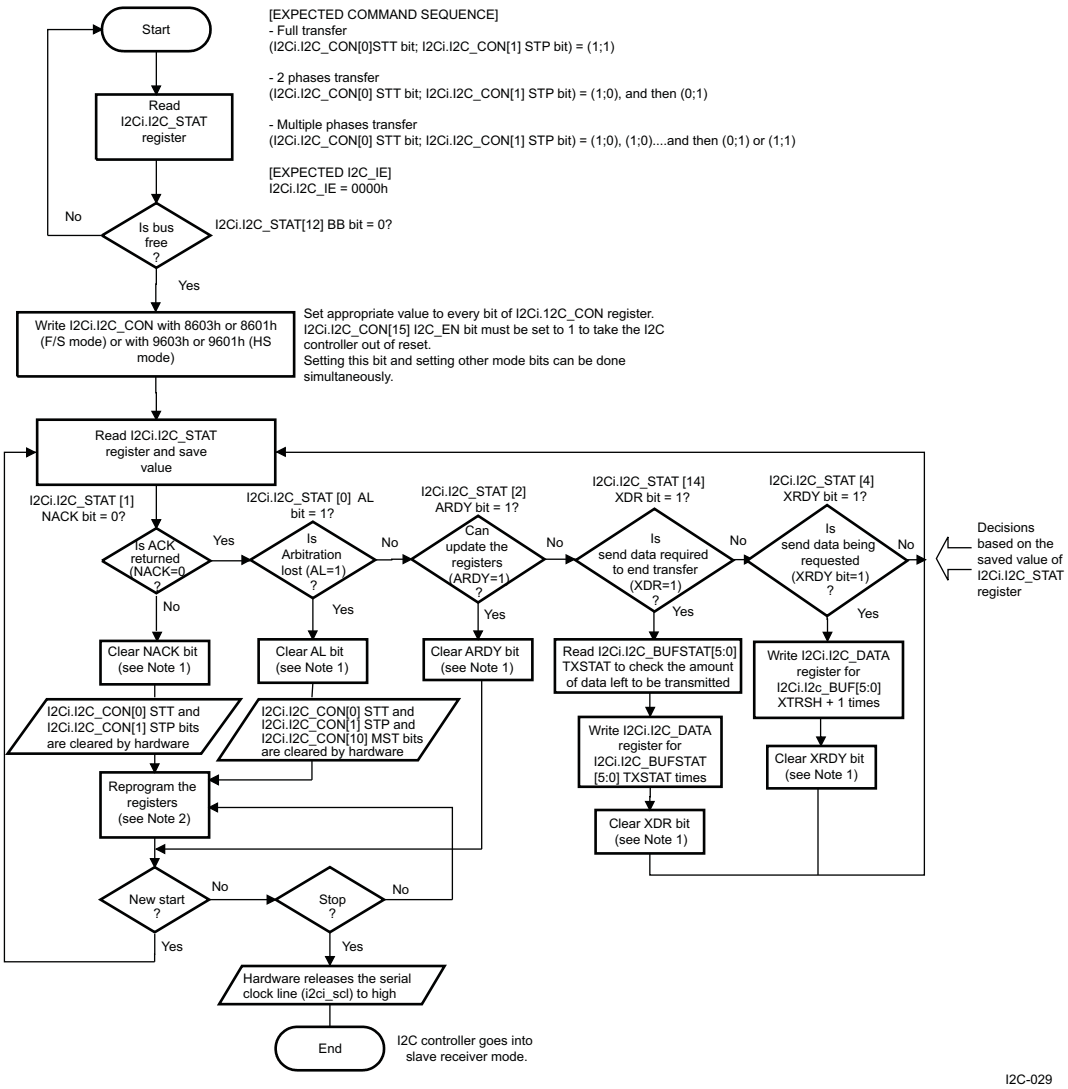


Figure 18-29. I2C Master Transmitter Mode, Polling Method, in F/S and HS Modes

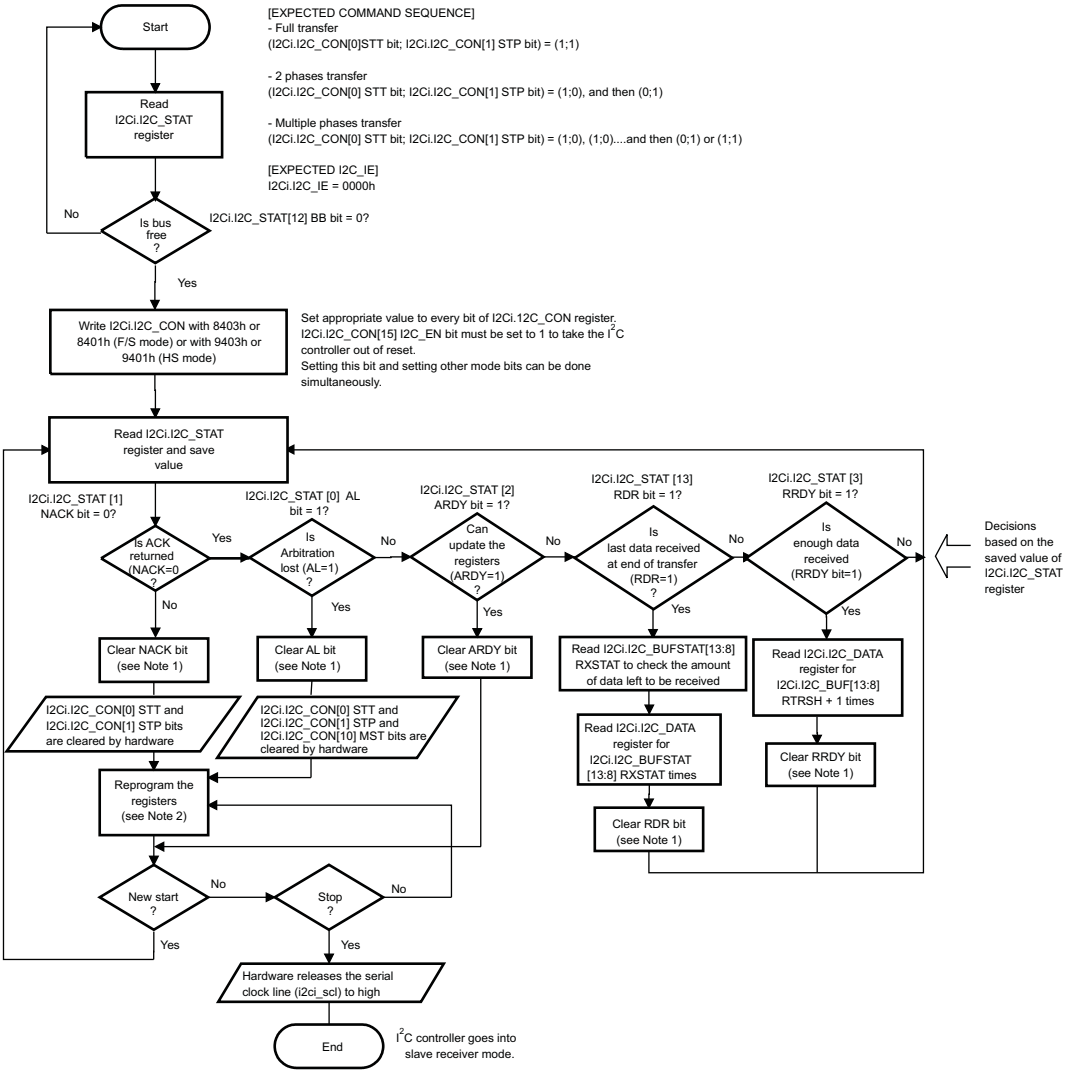


(1) The NAK, AL, ARDY, XDR, and XRDY bits are cleared by writing 1 to each corresponding bit in the I2Ci.I2C_STAT register.

(2) Reprogram the registers means: I2Ci.I2C_CON[11] STB and/or I2Ci.I2C_CON[10] MST bit and/or I2Ci.I2C_SA[9:0] SA register and/or I2Ci.I2C_CNT[15:0] DCOUNT register and/or I2Ci.I2C_CON[0] STT bit and/or I2Ci.I2C_CON[1] STP bit.

Note: In HS mode, the repeated start (Sr) condition and the clock frequency switching are automatically generated by the multimaster HS I2C controller.

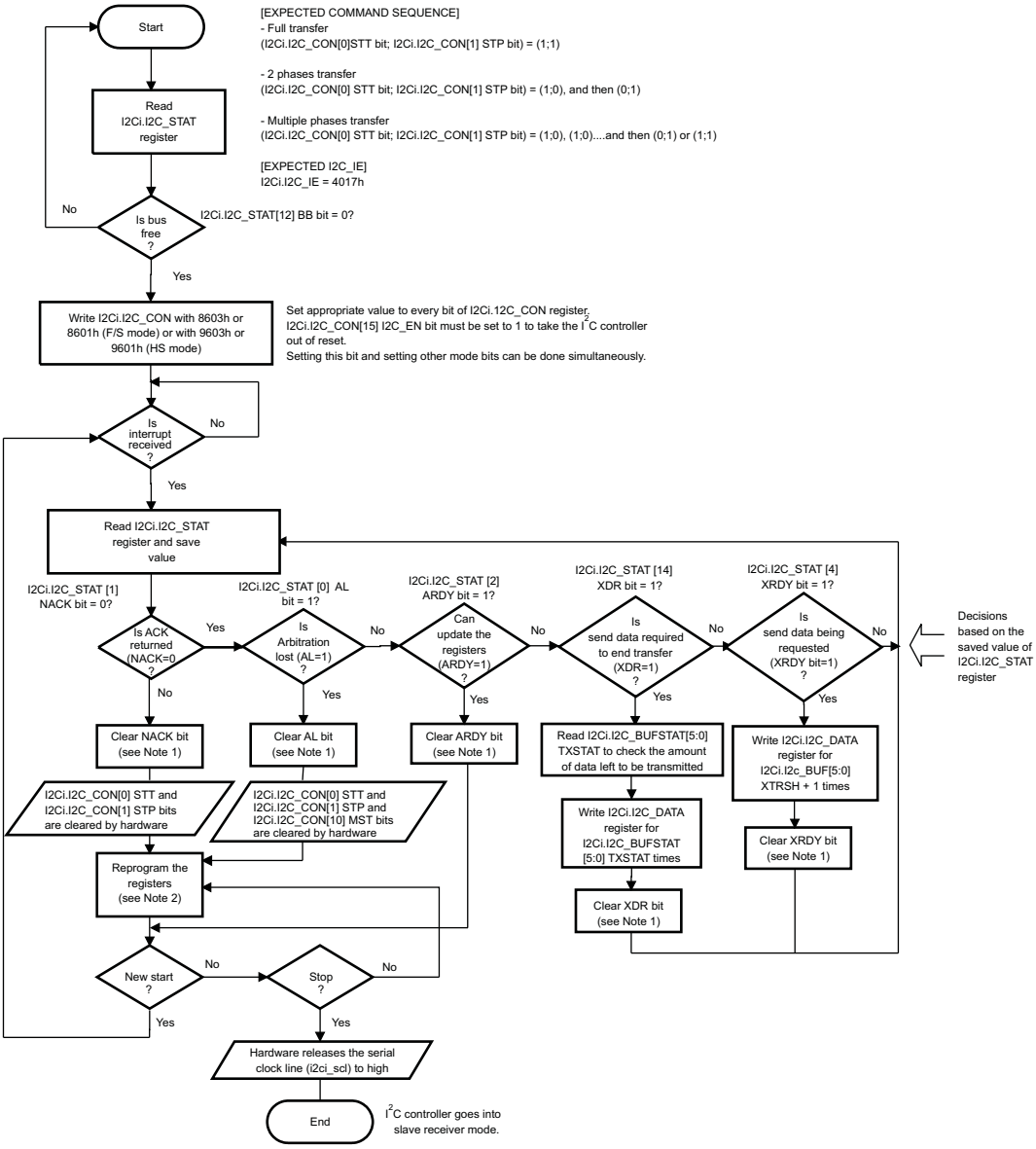
Figure 18-30. I²C Master Receiver Mode, Polling Method, in F/S and HS Modes



(1) The NAK, AL, ARDY, RDR, and RRDY bits are cleared by writing 1 to each corresponding bit in the I2Ci.I2C_STAT register.

(2) Reprogram registers means: I2Ci.I2C_CON[11] STB and/or I2Ci.I2C_CON[10] MST bit and/or I2Ci.I2C_SA[9:0] SA register and/or I2Ci.I2C_CNT[15:0] DCOUNT register and/or I2Ci.I2C_CON[0] STT bit and/or I2Ci.I2C_CON[1] STP bit.

Figure 18-31. I2C Master Transmitter Mode, Interrupt Method, in F/S and HS Modes

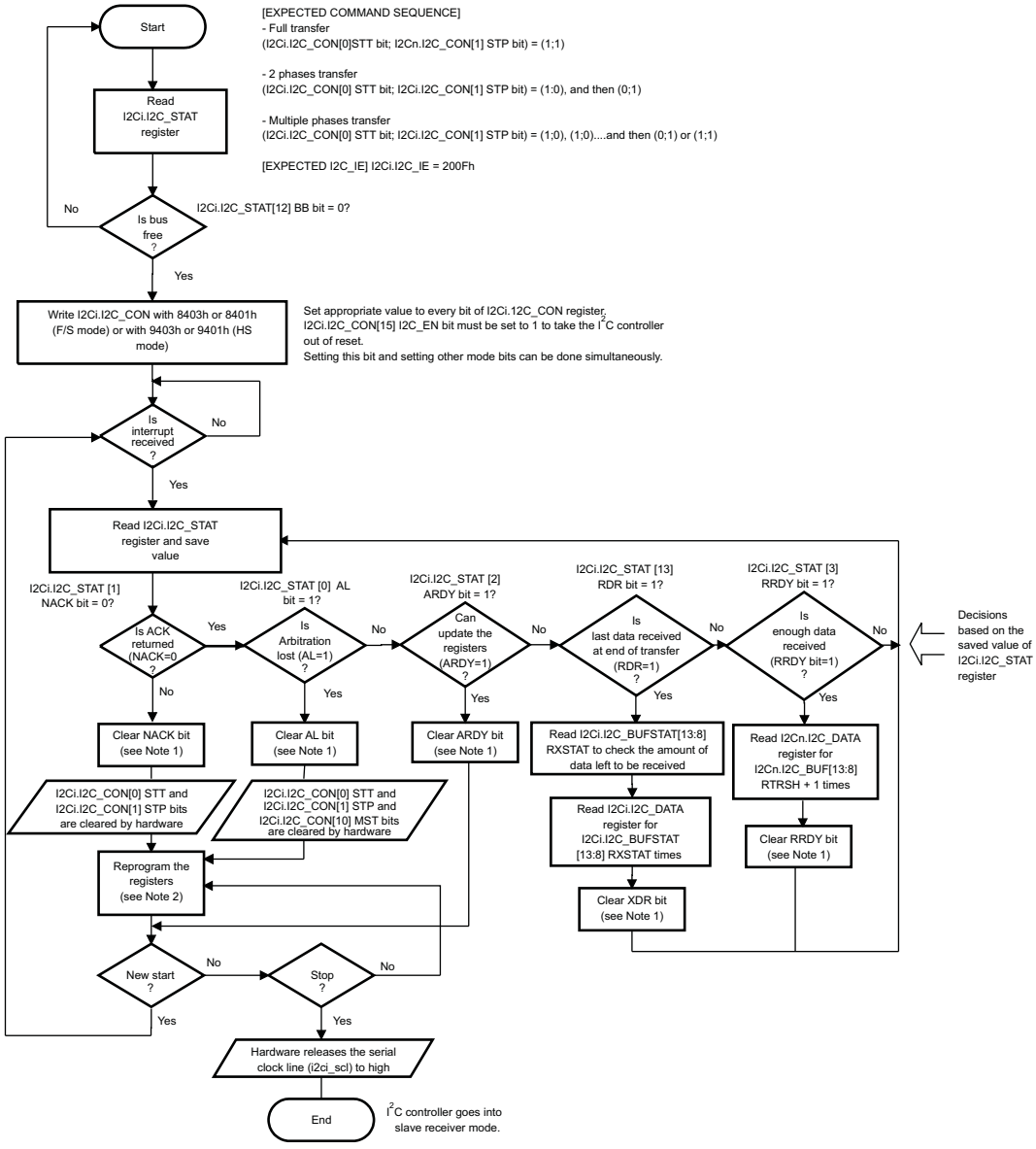


(1) The NAK, AL, ARDY, XDR, and XRDY bits are cleared by writing 1 to each corresponding bit in the I2Ci.I2C_STAT register.

(2) Reprogram registers means: I2Ci.I2C_CON[11] STB and/or I2Ci.I2C_CON[10] MST bit and/or I2Ci.I2C_SA[9:0] SA register and/or I2Ci.I2C_CNT[15:0] DCOUNT register and/or I2Ci.I2C_CON[0] STT bit and/or I2Ci.I2C_CON[1] STP bit.

Note: In HS mode, the repeated start (Sr) condition and the clock frequency switching are automatically generated by the multimaster HS I2C controller.

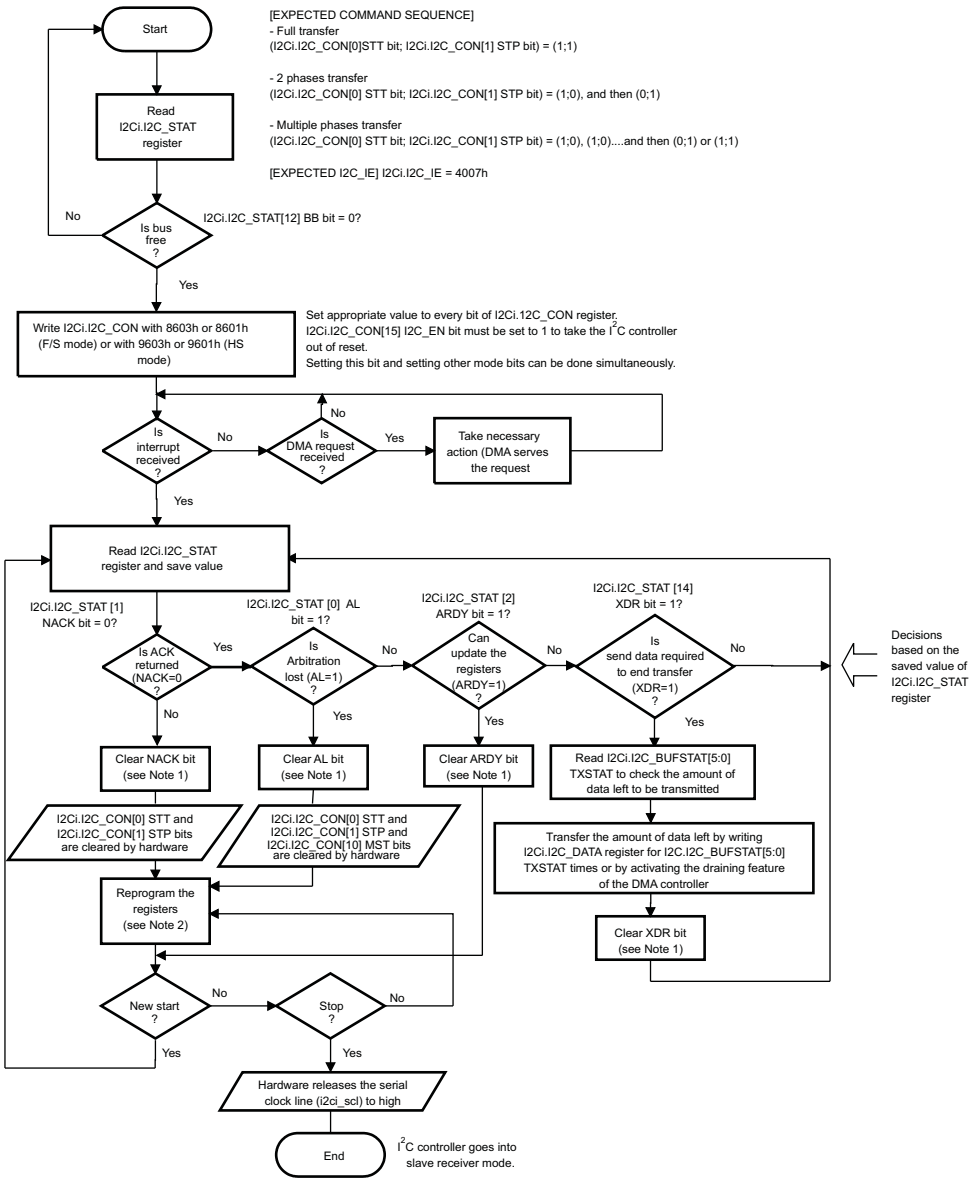
Figure 18-32. I²C Master Receiver Mode, Interrupt Method, in F/S and HS Modes



(1) The NAK, AL, ARDY, RDR, and RRDY bits are cleared by writing 1 to each corresponding bit in the I2Ci.I2C_STAT register.

(2) Reprogram registers means: I2Ci.I2C_CON[11] STB and/or I2Ci.I2C_CON[10] MST bit and/or I2Ci.I2C_SA[9:0] SA register and/or I2Ci.I2C_CNT[15:0] DCOUNT register and/or I2Ci.I2C_CON[0] STT bit and/or I2Ci.I2C_CON[1] STP bit.

Figure 18-33. I2C Master Transmitter Mode, DMA Method in F/S and HS Modes

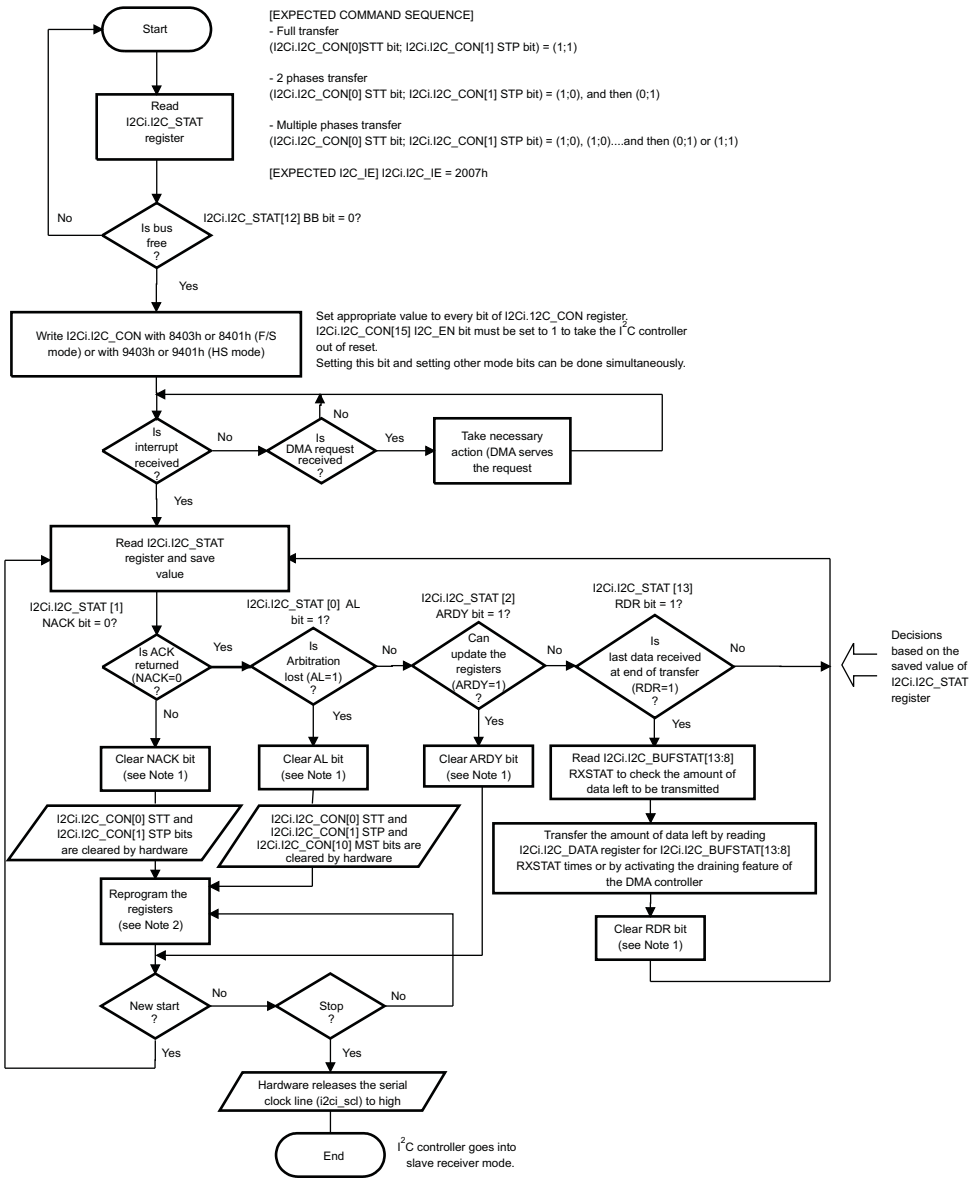


(1) The NAK, AL, ARDY, and XDR bits are cleared by writing 1 to each corresponding bit in the I2Ci.I2C_STAT register.

(2) Reprogram registers means: I2Ci.I2C_CON[11] STB and/or I2Ci.I2C_CON[10] MST bit and/or I2Ci.I2C_SA[9:0] SA register and/or I2Ci.I2C_CNT[15:0] DCOUNT register and/or I2Ci.I2C_CON[0] STT bit and/or I2Ci.I2C_CON[1] STP bit.

Note: In HS mode, the repeated start (Sr) condition and the clock frequency switching are automatically generated by the multimaster HS I2C controller.

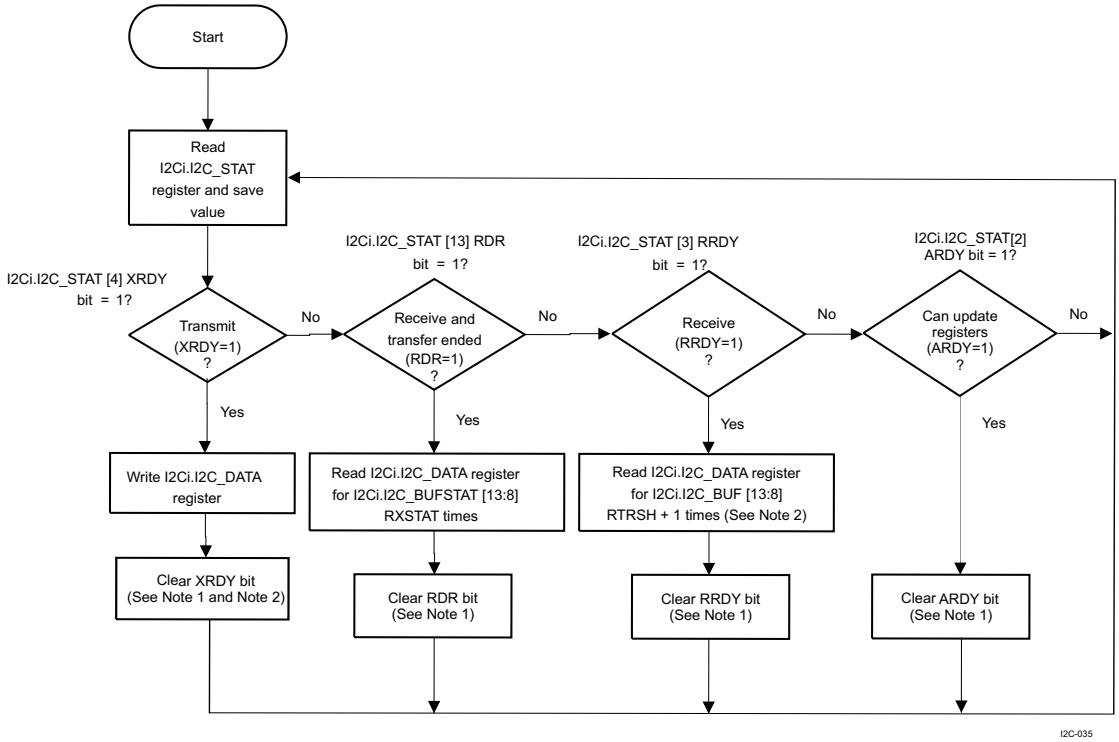
Figure 18-34. I²C Master Receiver Mode, DMA Method in F/S and HS Modes



(1) The NAK, AL, ARDY and RDR bits are cleared by writing 1 to each corresponding bit in the I2Ci.I2C_STAT register.

(2) Reprogram registers means: I2Ci.I2C_CON[11] STB and/or I2Ci.I2C_CON[10] MST bit and/or I2Ci.I2C_SA[9:0] SA register and/or I2Ci.I2C_CNT[15:0] DCOUNT register and/or I2Ci.I2C_CON[0] STT bit and/or I2Ci.I2C_CON[1] STP bit.

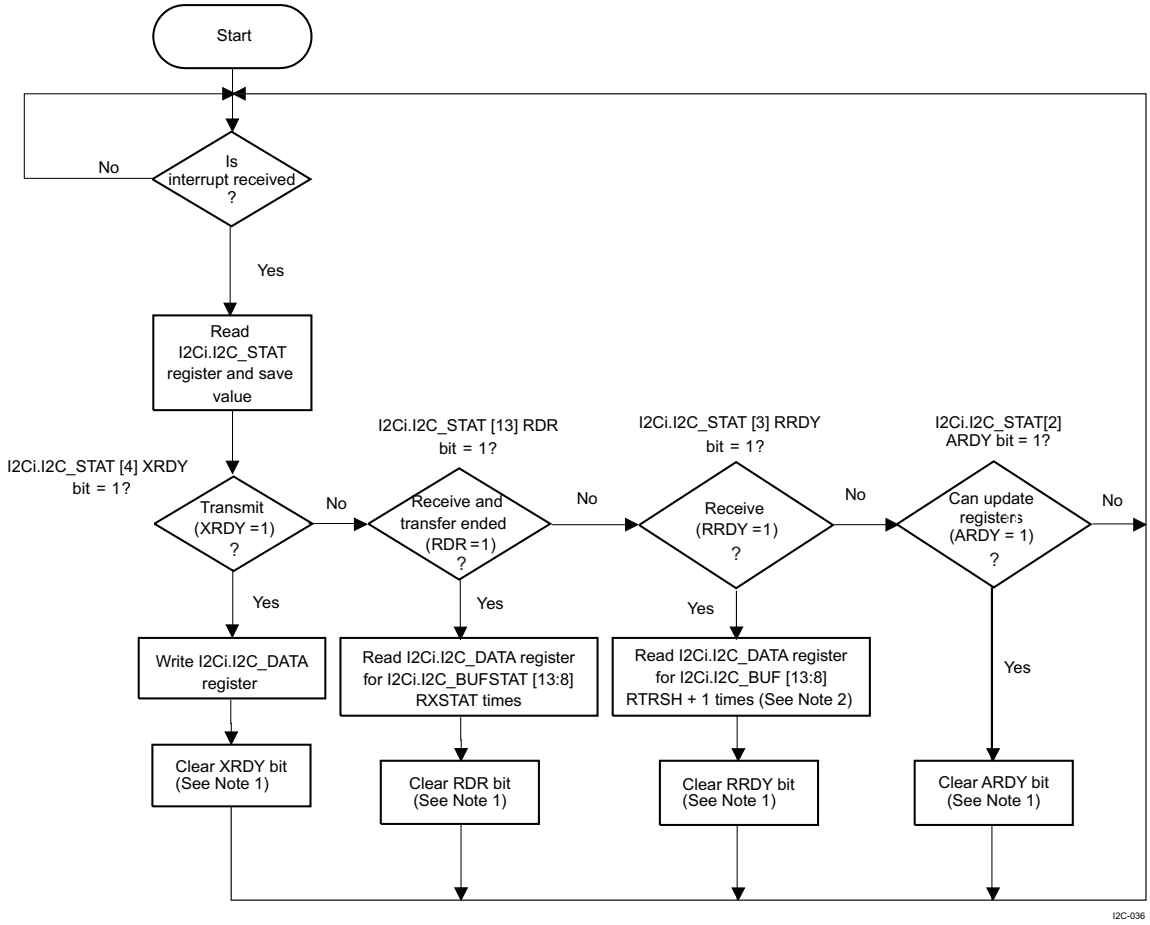
Figure 18-35. I2C Slave Transmitter/Receiver Mode, Polling



(1) The XRDY, RDR, RRDY and ARDY bits are cleared by writing 1 to each corresponding bit in the I2Ci.I2C_STAT register.

(2) In slave transmitter mode, the amount of data requested by the external master I2C device is unknown; thus, the I2Ci.I2C_BUF[5:0] XTRSH field must be configured to 0x0 (TX threshold = 1).

Figure 18-36. I2C Slave Transmitter/Receiver Mode, Interrupt



(1) The XRDY, RDR, RRDY and ARDY bits are cleared by writing 1 to each corresponding bit in the I2Ci.I2C_STAT register.

(2) In slave transmitter mode, the amount of data requested by the external master I2C device is unknown; thus, the I2Ci.I2C_BUF[5:0] XTRSH field must be configured to 0x0 (TX threshold = 1).

18.5.2 High-Speed I2C Controller Basic Programming Model in SCCB Mode

This section describes the programming model of the multimaster HS I2C controllers configured in SCCB mode.

18.5.2.1 Main Program

18.5.2.1.1 Configure the Module Before Enabling the I2C Controller

Before enabling the I2C controller, perform the following steps:

1. Enable the functional and interface clocks (see Section 18.3.1.1.1).
2. Program the prescaler to obtain an approximately 12-MHz internal sampling clock (I2Ci_INTERNAL_CLK) by programming the corresponding value in the I2Ci.I2C_PSC[7:0] PSC field. This value depends on the frequency of the functional clock (I2Ci_FCLK). Because this frequency is 96MHz, the I2Ci.I2C_PSC[7:0] PSC field value is 0x7.
3. Program the I2Ci.I2C_SCLL[7:0] SCLL and I2Ci.I2C_SCLH[7:0] SCLH fields to obtain a bit rate of 100K bps (maximum authorized bit rate in SCCB mode). This value depends on the internal sampling clock frequency (see Table 18-12).
4. Configure the 7-bit slave address (ID value) by storing it in the I2Ci.I2C_SA register.

5. Configure the 8-bit register address (subaddress) by storing it in the I2Ci.I2C_OA0 register.
6. Configure the I2Ci.I2C_BUF[13:8]RTRSH field to 0x0 (RX threshold to 1) and the I2Ci.I2C_BUF[5:0]XTRSH field to 0x0 (TX threshold to 1).
7. Take the I2C controller out of reset by setting the I2Ci.I2C_CON[15] I2C_EN bit to 1.

18.5.2.1.2 Initialize the I2C Controller

To initialize the I2C controller, perform the following steps:

1. Configure the I2Ci.I2C_CON register:
   - In SCCB mode, only the master mode is supported; set the I2Ci.I2C_CON[10] MST bit to 1.
   - For transmitter mode (write to the external SCCB device register) or receiver mode (read from the external SCCB device register), set the I2Ci.I2C_CON[9] TRX bit (0: receiver, 1: transmitter).
2. If using an interrupt to transmit/receive data, set to 1 the corresponding bit in the I2Ci.I2C_IE register (the I2Ci.I2C_IE[4] XRDY_IE bit for the transmit interrupt, the I2Ci.I2C_IE[3] RRDY bit for the receive interrupt).

18.5.2.1.3 Initiate a Transfer

Poll the I2Ci.I2C_STAT[12] BB bit. If it is cleared to 0 (bus not busy), set the I2Ci.I2C_CON[0] STT bit to 1. Because a transfer allows the LH to write or read only a single byte to or from the external SCCB device, the transmission automatically stops at the end of the transfer. When the transfer is complete, the I2Ci.I2C_STAT[2] ARDY bit is set to 1. In SCCB mode, the I2Ci.I2C_CON[1] STP bit is not used.

18.5.2.1.4 Receive Data

Poll the I2Ci.I2C_STAT[3] RRDY bit, or use the RRDY interrupt (the I2Ci.I2C_IE[3] RRDY_IE bit must be set to 1) to read the receive data in the I2Ci.I2C_DATA register.

Note: In SCCB mode, the I2Ci.I2C_BUF[13:8] RTRSH field (RX threshold) must be set to a value of 0x0 (RX threshold = 1).

18.5.2.1.5 Transmit Data

Poll the I2Ci.I2C_STAT[4] XRDY bit, or use the XRDY interrupt (the I2Ci.I2C_IE[4] XRDY_IE bit must be set to 1) to write the data to the I2Ci.I2C_DATA register.

Note: In SCCB mode, the I2Ci.I2C_BUF[5:0] XTRSH field (TX threshold) must be set to a value of 0x0 (TX threshold = 1).

18.5.2.2 Interrupt Subroutine Sequence

1. Test for register access ready (I2Ci.I2C_STAT[2] ARDY status bit) and resolve accordingly.
2. Test for receive data ready (I2Ci.I2C_STAT[3] RRDY status bit) and resolve accordingly.
3. Test for transmit data ready (I2Ci.I2C_STAT[4] XRDY status bit) and resolve accordingly.

18.5.2.3 Programming Flow Diagrams

Figure 18-37 through Figure 18-41 are procedure flowcharts for programming the SCCB mode.

Figure 18-37. SCCB Setup Procedure

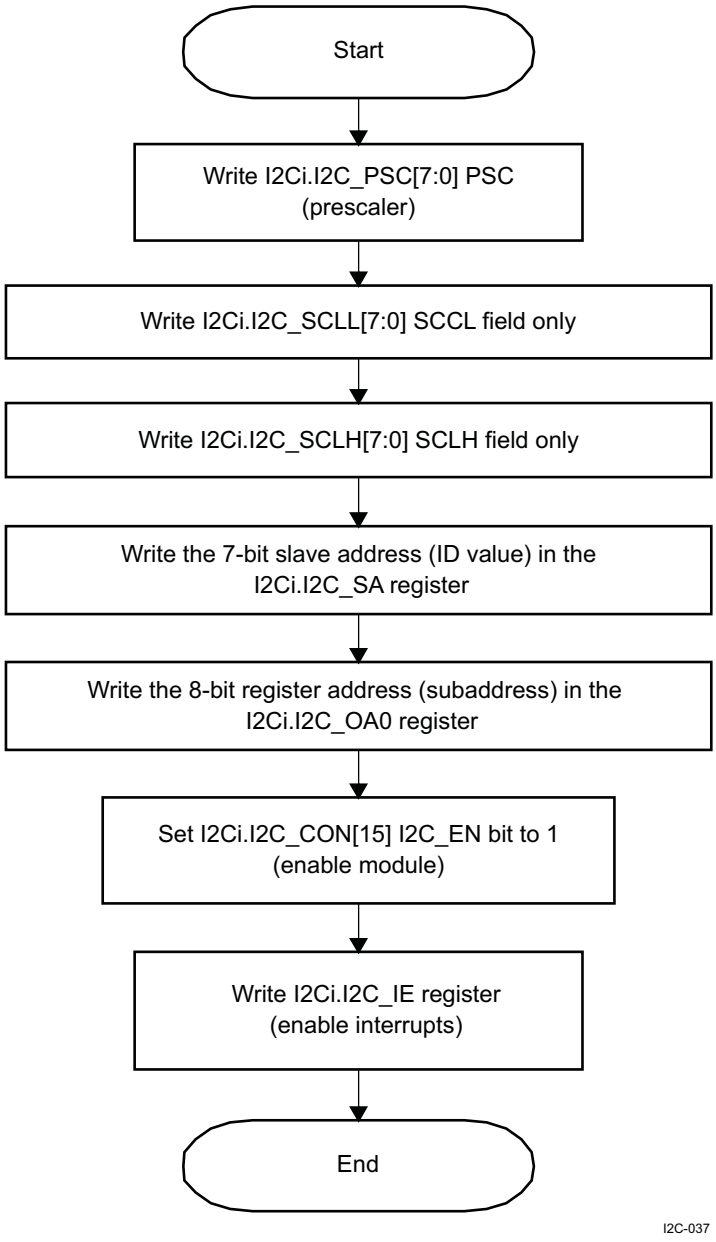
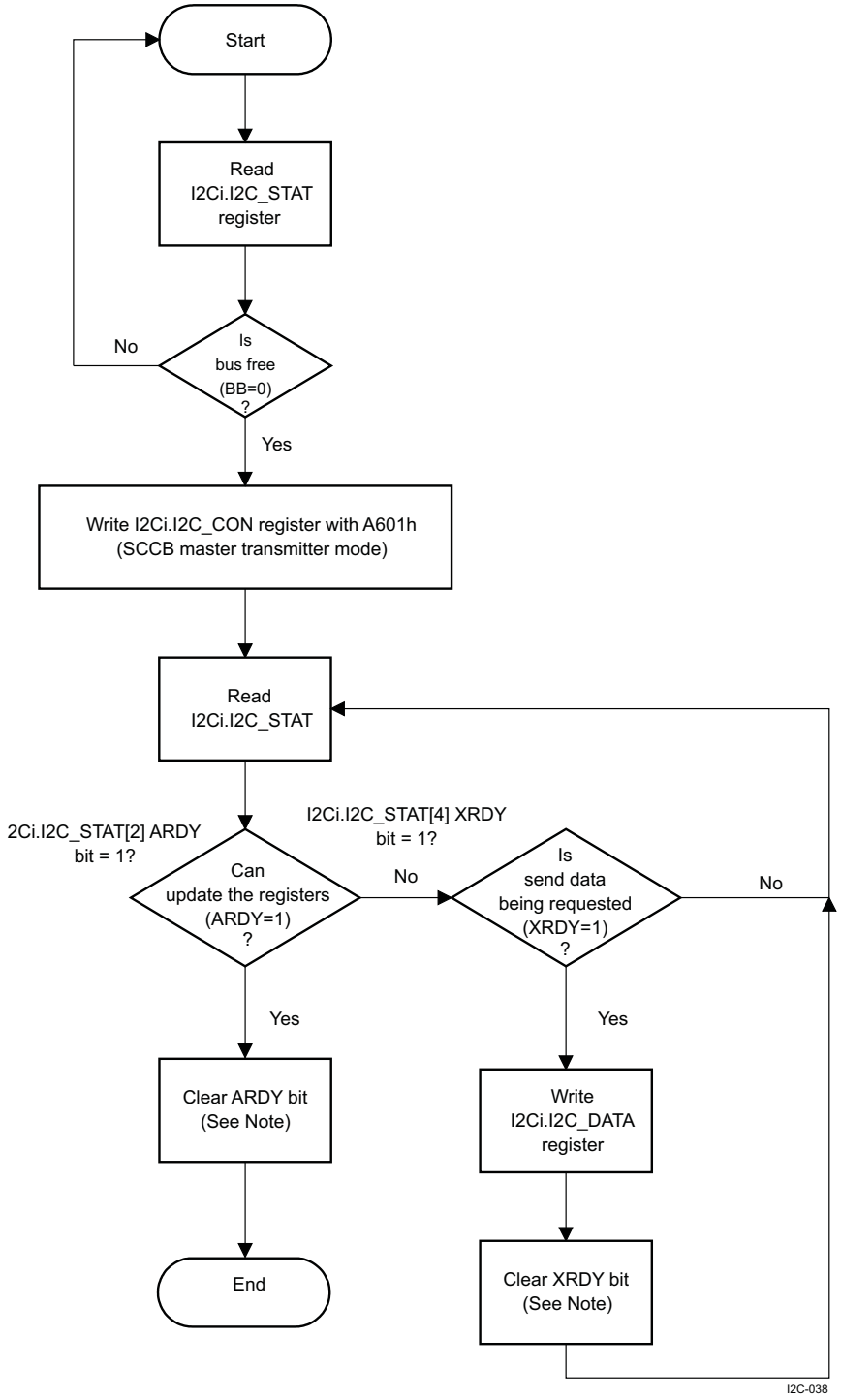
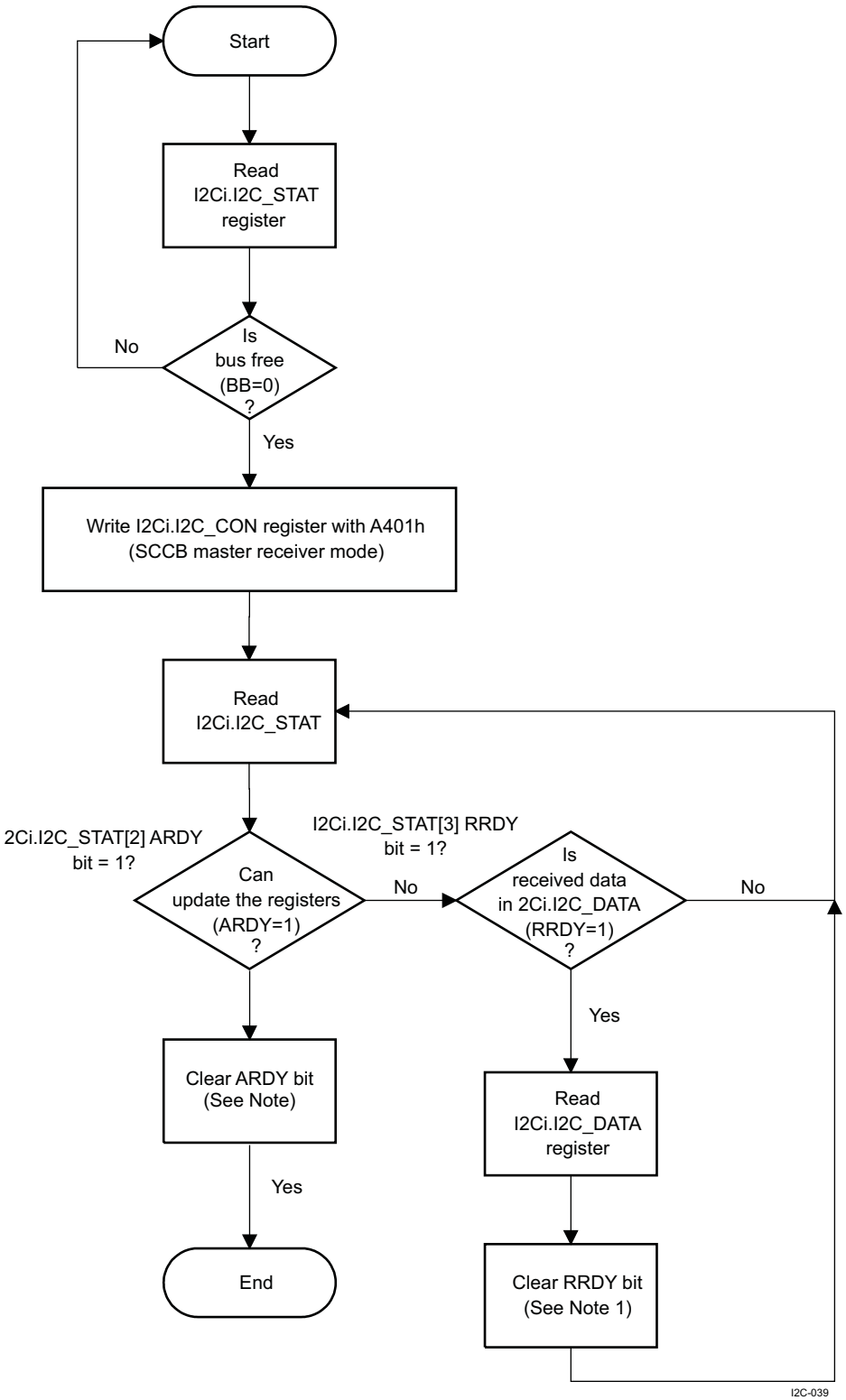


Figure 18-38. SCCB Master Transmitter Mode, Polling



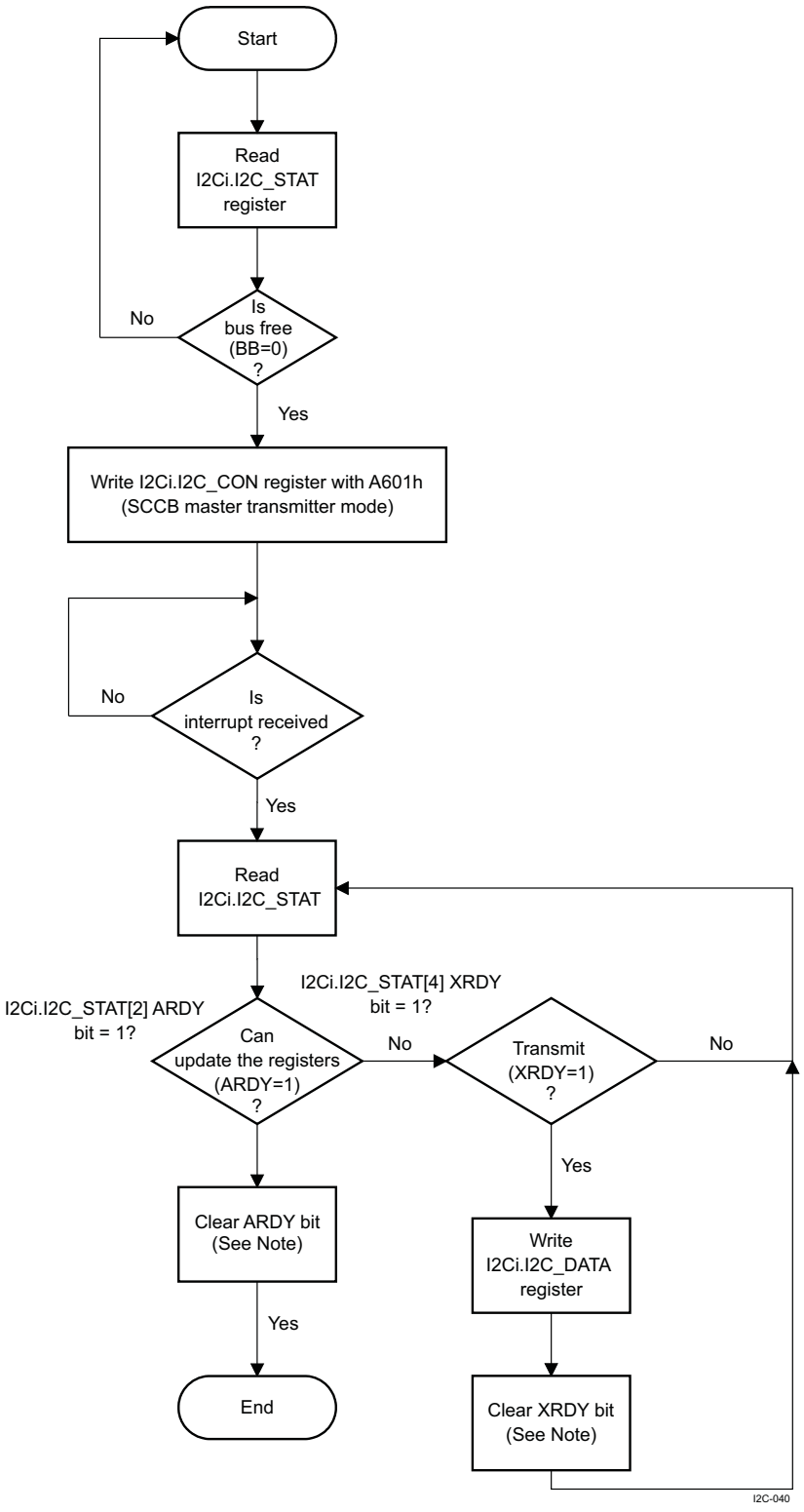
Note: The XRDY and ARDY bits are cleared by writing 1 to the corresponding bit in the I2Ci.I2C_STAT register.

Figure 18-39. SCCB Master Receiver Mode, Polling



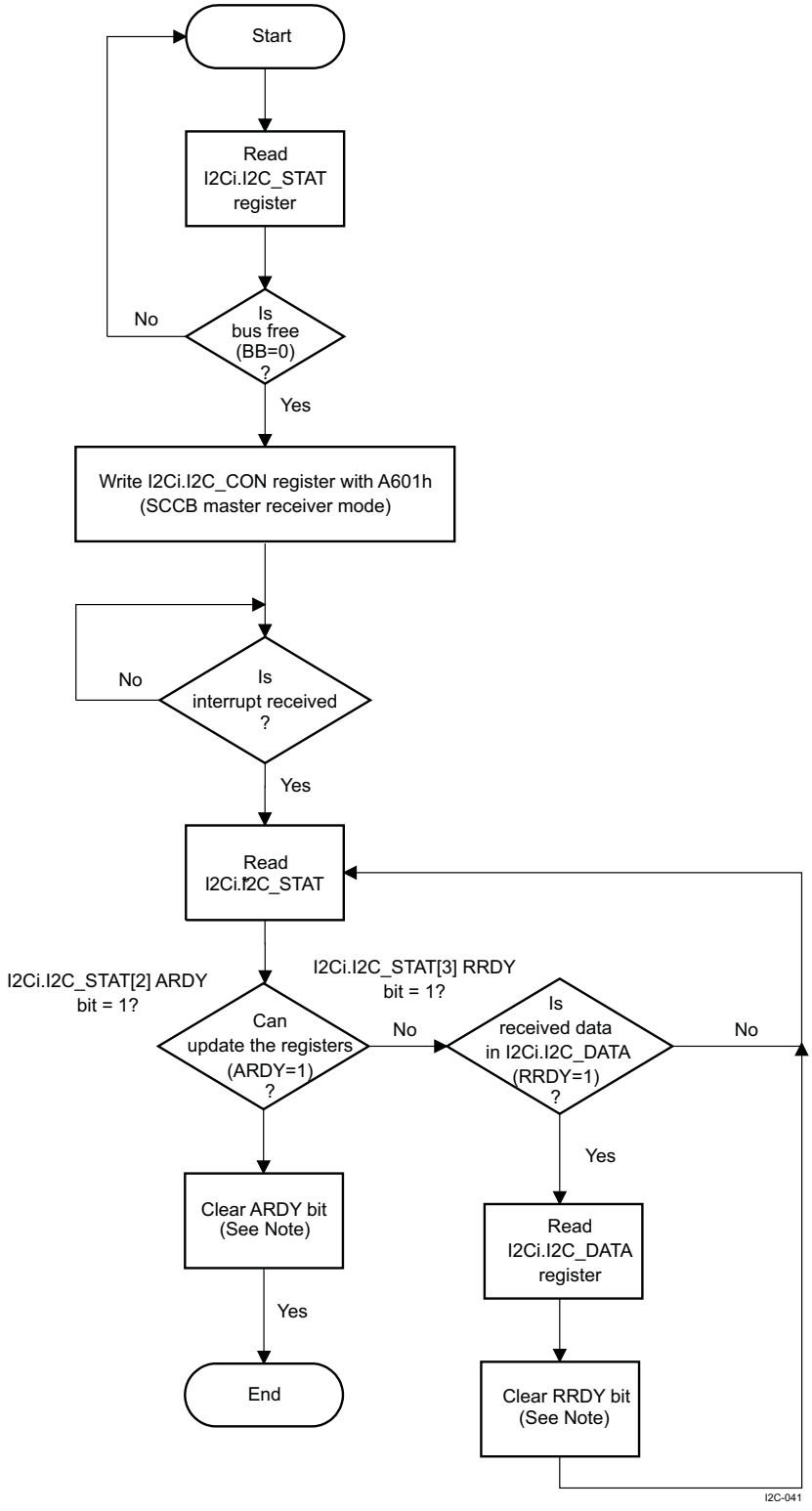
Note: The RRDY and ARDY bits are cleared by writing 1 in the corresponding bit in the I2Ci.I2C_STAT register.

Figure 18-40. SCCB Master Transmitter Mode, Interrupt



Note: The XRDY and ARDY bits are cleared by writing 1 in the corresponding bit in the I2Ci.I2C_STAT register.

Figure 18-41. SCCB Master Receiver Mode, Interrupt



Note: The RRDY and ARDY bits are cleared by writing 1 in the corresponding bit in the I2Ci.I2C_STAT register.

18.5.3 Master Transmitter HS I2C Controller I2C4 Basic Programming Model for Communication With Power Chips

This section describes the programming model of the master transmitter HS I2C controller I2C4 for communication with one or more power chips. The HS I2C controller I2C4 allowsthe two voltage Finite State Machines (FSMs) in the PRCM module of the device to be interfaced to external power chips through the I2C bus for dynamic voltage control of two power supplies and power sequencing. The primary programming tasks are to set up the configuration registers according to the external power supply chip(s) being used.

18.5.3.1 Configure the Voltage Controller Registers

To use the voltage control function, the LH must configure the following registers in the voltage controller of the PRCM module:

- The voltage configuration register address for each VDD channel in the PRCM.PRM_VC_SMPS_RA register
- The ON/ON-low-power-Retention/OFF command configuration register address for each VDD channel in the PRCM.PRM_VC_SMPS_CMD_RA register
- The set of ON/ON-low-power/Retention/OFF voltage/mode values in the PRCM.PRM_VC_CMD_VAL_0 register for the first VDD channel and the PRCM.PRM_VC_CMD_VAL_1 register for the second VDD channel
- The VDD channels configuration selection in the PRCM.PRM_VC_CH_CONF register

For details about voltage controller register configuration, see Chapter 4, *Power, Reset, and Clock Management.*

18.5.3.2 Configure the Master Transmitter HS I2C Controller I2C4

At power-on reset, the I2C4 is in HS mode (PRCM.PRM_VC_I2C_CFG[3] HSEN bit). In HS mode, the LH must configure the master code value for the preamble I2C high-speed transmission by configuring the PRCM.PRM_VC_I2C_CFG[2:0] MCODE field. If the external power chips do not support the I2C HS mode, the HS mode can be disabled and F/S mode enabled by clearing the PRCM.PRM_VC_I2C_CFG[3] HSEN bit to 0.

By default, the repeated start operation is enabled (PRCM.PRM_VC_I2C_CFG[4] SREN bit set to 1 at power-on reset). In F/S mode, the repeated start operation can be disabled by clearing the PRCM.PRM_VC_I2C_CFG[4] SREN bit to 0.

18.5.3.3 Configure the External Power Chip(s)

The LH can send configuration commands from one bypass input to the external power chip(s), using the same I2C interface. To send a configuration command, the LH must process as follows:

- Check whether the transmission of a configuration command is ongoing by polling the PRCM.PRM_VC_BYPASS_VAL[24]VALID bit (1: a transmission is ongoing, 0: a new transmission can start).
- Set the external power chip 7-bit slave address in the PRCM.PRM_VC_BYPASS_VAL[6:0] SLAVEADDR field, set the configuration register address of the external power chip in the PRCM.PRM_VC_BYPASS_VAL[15:8] REGADDR field, and set the data to be written to the external power chip configuration register in the PRCM.PRM_VC_BYPASS_VAL[23:16] DATA field.
- Set the PRCM.PRM_VC_BYPASS_VAL[24]VALID bit to 1 to send the configuration command to the external power chip.

18.6 High-Speed Multimaster I2C Use Cases and Tips

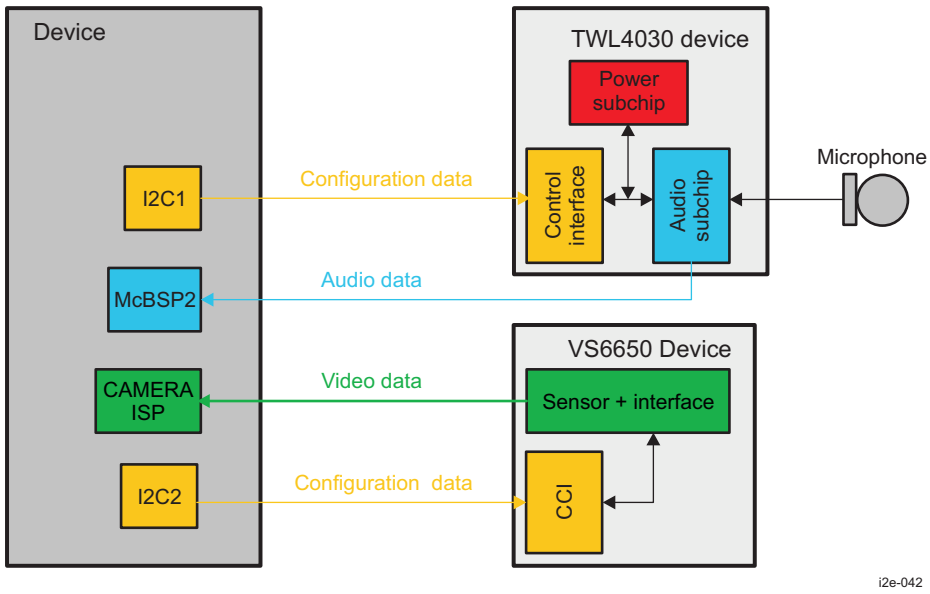
18.6.1 Camcorder Use Case: How to Configure the I2C Modules When Connecting TWL4030 Companion chip and VS6650 Digital Camera Device

18.6.1.1 Overview

In a whole Camcorder use case, the TWL4030 companion chip and the VS6650 digital camera device are configured by the I2C modules inside the OMAP34xx device.

The I2C1 module configures the TWL4030 companion chip, and the I2C3 module configures the VS6650 digital camera device.

Figure 18-42. Overview



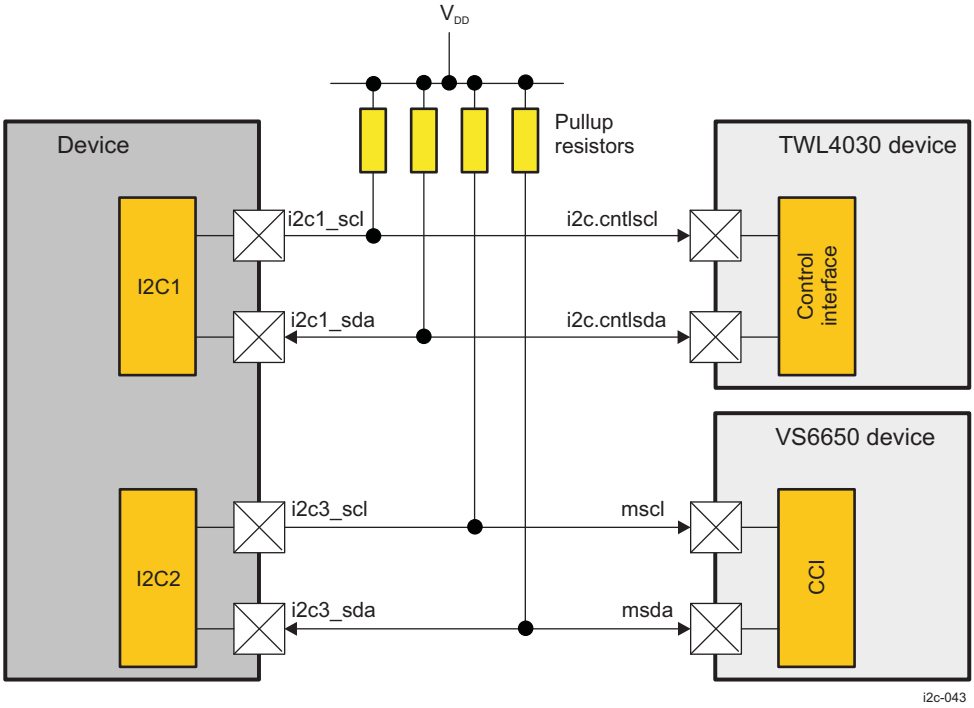
For the Camcorder use case, the configuration of the I2C modules inside the OMAP34xx device is the following:

- Master mode: the I2C clocks are generated by the I2C modules inside the OMAP34xx device.
- Transmit mode: the I2C modules write data into the TWL4030 companion chip and the VS6650 digital camera device.
- No start-byte generation.
- Fast/standard (F/S) mode at 100 kbps.
- 7-bit addressing mode to address the TWL4030 companion chip and the VS6650 digital camera device registers.
- Polling method to access the TWL4030 companion chip and the VS6650 digital camera device registers.

18.6.1.2 Environment

In this use case, the control interface of the TWL4030 companion chip is connected to the I2C1 module, while the communication control interface (CCI) of the VS6650 digital camera device is connected to the I2C3 module.

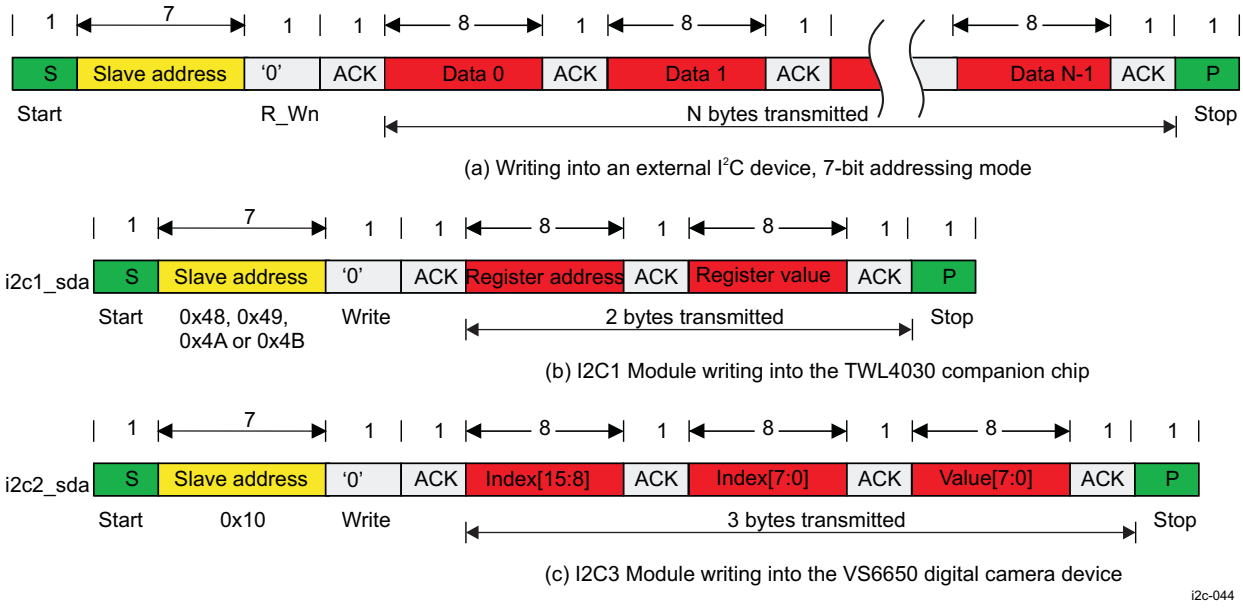
Figure 18-43. Environment



18.6.1.2.1 Data Transfer Formats

The data transfer format for configuring the TWL4030 companion chip and the VS6650 digital camera device is showed as follows:

Figure 18-44. Data Transfer Format



To configure a TWL4030 companion chip register, two data bytes must be transmitted by the I2C1 module: the register address (1 byte), and the register value (1 byte).

To configure a VS6650 digital camera device register, three data bytes must be transmitted by the I2C3 module: the index most-significant byte (1 byte), the index less-significant byte (1 byte), and the register value (1 byte).

18.6.1.3 Programming Flow

18.6.1.3.1 Initial Configuration

The initialization of the I2Ci module (where i = 1, 3) is done in four parts:

1. I2Ci module interface and functional clocks enabling.
2. I2Ci module clock generation configuration.
3. I2Ci module mode configuration.
4. I2Ci module slave address configuration.
5. I2Ci module enabling.

See Figure 18-45 for the corresponding programming flowchart.

18.6.1.3.1.1 I2Ci Module Interface and Functional Clocks Enabling

To enable the interface and functional clocks of the I2Ci module, the following steps must be done:

1. Enable the interface clock for the I2Ci module (set the PRCM.CM_ICLKEN1_CORE[15] EN_I2C1 bit to 1 for the I2C1 module and the PRCM.CM_ICLKEN1_CORE[17] EN_I2C3 bit to 1 for the I2C3 module).
2. Enable the functional clock for the I2Ci module (set the PRCM.CM_FCLKEN1_CORE[15] EN_I2C1 bit to 1 for the I2C1 module and the PRCM.CM_FCLKEN1_CORE[17] EN_I2C3 bit to 1 for the I2C3 module).

Table 18-14 shows the registers to be configured to initialize the I2C1 and I2C3 modules

Table 18-14. Registers Print for the I2Ci Module Initialization

Register	Address	Value	Value Description
PRCM.CM_ICLKEN1_CORE	0x4800 4A10	0x0002 8000	I2C1 and I2C3 interface clocks enabled
PRCM.CM_FCLKEN1_CORE	0x4800 4A00	0x0002 8000	I2C1 and I2C3 functional clocks enabled

18.6.1.3.1.2 I2Ci Module Clock Generation Configuration

To configure the clock generation of the I2Ci modules, perform the following steps:

1. Configure the I2Ci.I2C_PSC[3:0] PSC field to 0x17 (23).
2. Configure the I2Ci.I2C_SCLL[7:0] SCLL field to 0x0D (13).
3. Configure the I2Ci.I2C_SCLH[7:0] SCLH field to 0x0F (15).
   By programming these values, the I2C clock rate is configured to 100 kbps for the I2Ci modules.

Table 18-15 shows all the registers to be configured for the I2Ci module clock generation configuration.

Table 18-15. Registers Print for the I2Ci Module Clock Generation Configuration

Register	Address	Value	Value Description
I2C1.I2C_PSC	0x4807 0030	0x0017	PSC[3:0] field set to 0x17 for the I2C1 module
I2C1.I2C_SCLL	0x4807 0034	0x000D	SCLL[7:0] field set to 0x0D for the I2C1 module
I2C1.I2C_SCLH	0x4807 0038	0x000F	SCLH[7:0] field set to 0x0F for the I2C1 module

Table 18-15. Registers Print for the I2Ci Module Clock Generation Configuration (continued)

Register	Address	Value	Value Description
I2C3.I2C_PSC	0x4806 0030	0x0017	PSC[3:0] field set to 0x17 for the I2C3 module
I2C3.I2C_SCLL	0x4806 0034	0x000D	SCLL[7:0] field set to 0x0D for the I2C3 module
I2C3.I2C_SCLH	0x4806 0038	0x000F	SCLH[7:0] field set to 0x0F for the I2C3 module

18.6.1.3.1.3 I2Ci Module Mode Configuration

To configure the I2Ci modules in F/S (Fast/Standard) mode, set the I2Ci.I2C_CON[13:12] OPMODE field to 0x0.

Table 18-16. Registers Print for the I2Ci Slave Address Configuration

Register name	Address	Value	Value description
I2C1.I2C_CON	0x4807 0024	0x0000	OPMODE[13:12] configured to 0x0 for I2C1 (Fast/Standard mode)
I2C3.I2C_CON	0x4806 0024	0x0000	OPMODE[13:12] configured to 0x0 for I2C3 (Fast/Standard mode)

18.6.1.3.1.4 I2Ci Module Slave Address Configuration

To configure the slave address of the I2Ci modules, the following steps must be done:

1. Enable the 7-bit addressing mode by clearing the I2Ci.I2C_CON[8] XSA bit to 0 because the I2C addresses of the TWL4030 companion chip and the VS6650 digital camera device are 7-bit wide.
2. Configure the I2C1.I2C_SA[9:0] SA field to 0x48, 0x49, 0x4A or 0x4B, so that the I2C1 module can access the TWL4030 companion chip registers. The slave address configured into the I2C1.I2C_SA[9:0] SA field depends on the TWL4030 companion chip register.
3. Configure the I2C3.I2C_SA[9:0] SA field to 0x10, so that the I2C3 module can access the VS6650 digital camera device registers. For more information, see the ST VS6650 datasheet.

Table 18-17 shows all the registers to be configured for the I2Ci slave address configuration.

Table 18-17. Registers Print for the I2Ci Slave Address Configuration

Register name	Address	Value	Value description
I2C1.I2C_CON	0x4807 0024	0x0000	XSA bit is cleared to 0 for the I2C1 module
I2C3.I2C_CON	0x4806 0024	0x0000	XSA bit is cleared to 0 for the I2C3 module
I2C1.I2C_SA	0x4807 002C	0x0048, or 0x0049, or 0x004A, or 0x004B	Slave address of the TWL4030 companion sub-chip for the I2C1 module
I2C3.I2C_SA	0x4806 002C	0x0010	Slave address of the VS6650 digital camera device for the I2C3 module

18.6.1.3.1.5 I2Ci Module Enabling

To enable the I2Ci modules, set the I2Ci.I2C_CON[15] I2C_EN bit to 1.

Table 18-24 shows all the registers to be configured for the I2Ci slave address configuration.

Table 18-18. Registers Print for the I2Ci Module Enabling

Register name	Address	Value	Value description
I2C1.I2C_CON	0x4807 0024	0x8000	I2C1 module enabled
I2C3.I2C_CON	0x4806 0024	0x8000	I2C3 module enabled

18.6.1.3.2 Operational Mode

Writing a byte into a TWL4030 companion chip or VS6650 digital camera device register is done in five parts:

1. Transfer configuration.
2. I2Ci bus status checking.
3. Transfer initiation.
4. Data transfer.
5. Transfer completion.

See Figure 18-46 for the I2C1 programming flowchart and Figure 18-47 for the I2C3 programming flowchart.

18.6.1.3.2.1 Transfer Configuration

To configure the transfer for the I2Ci module, perform the following steps:

1. Configure the I2Ci.I2C_CNT[15:0] DCOUNT field to the number of data bytes to be transmitted: 0x02 to transmit 2 bytes to the TWL4030 companion chip by the I2C1 module, 0x03 to transmit 3 bytes to the VS6650 digital camera device by the I2C3 module.
2. Set the I2Ci.I2C_CON[10] MST bit to 1 (master mode).
3. Set the I2Ci.I2C_CON[9] TRX bit to 1 (transmit mode).

Table 18-19 shows all the registers to be configured for the transfer configuration.

Table 18-19. Registers Print for the Transfer Configuration

Register	Address	Value	Value Description
I2C1.I2C_CNT	0x4807 0018	0x0002	Set the number of transmitted bytes to 0x02 for the I2C1 module
I2C1.I2C_CON	0x4807 0024	0x8600	MST and TRX bits are set to 1 for the I2C1 module (master transmit mode)
I2C3.I2C_CNT	0x4806 0018	0x0003	Set the number of transmitted bytes to 0x03 for the I2C3 module
I2C3.I2C_CON	0x4806 0024	0x8600	MST and TRX bits are set to 1 for the I2C3 module (master transmit mode)

18.6.1.3.2.2 I2Ci Bus Status Checking

Before initiating a transfer , the status of the I2Ci bus must be checked by polling the I2Ci.I2C_STAT[12] BB bit. The transfer is initiated only when the I2Ci.I2C_STAT[12] BB bit is set to 0 by the I2Ci module.

Table 18-20 shows all the registers to be configured for the I2Ci bus status checking.

Table 18-20. Registers Print for the I2Ci Bus Status Checking

Register name	Address	Value	Value description
I2C1.I2C_STAT	0x4807 0008	0x1000	BB bit is set to 1 when the I2C1 bus is busy

Table 18-20. Registers Print for the I2Ci Bus Status Checking (continued)

Register name	Address	Value	Value description
I2C3.I2C_STAT	0x4806 0008	0x1000	BB bit is set to 1 when the I2C3 bus is busy

18.6.1.3.2.3 Transfer Initiation

Once the I2Ci bus is free, the transfer can be initiated by setting the I2Ci.I2C_CON[0] STT bit (start). The I2Ci.I2C_CON[1] STP bit (stop) can also be set to 1 as soon as the I2Ci.I2C_CON[0] STT bit is cleared to 0 by the I2Ci module in order to generate a stop condition at the end of the transfer.

Table 18-21 shows all the registers to be configured for the transfer initiation.

Table 18-21. Registers Print for the Transfer Initiation

Register name	Address	Value	Value description
I2C1.I2C_CON	0x4807 0024	0x8601	STT bit is set to 1 for the I2C1 module
I2C3.I2C_CON	0x4806 0024	0x8601	STT bit is set to 1 for the I2C3 module
I2C1.I2C_CON	0x4807 0024	0x8602	STP bit is set to 1for the I2C1 module when the STT bit is cleared by the I2C1 module
I2C3.I2C_CON	0x4806 0024	0x8602	STP bit is set to 1for the I2C3 module when the STT bit is cleared by the I2C3 module

Note: As soon as the I2Ci.I2C_CON[0] STT bit is set to 1, a start condition is generated on the I2Ci bus, and the I2Ci.I2C_CON[0] STT bit is automatically cleared to 0 by the I2Ci module.

The I2Ci.I2C_CON[1] STP bit is automatically cleared to 0 at the end of the transfer, when the stop condition is generated on the I2Ci bus.

18.6.1.3.2.4 Data Transfer

After the generation of the start condition on the I2Ci bus, the slave address programmed in the I2Ci.I2C_SA[9:0] SA field is automatically transmitted on the bus, followed by the R_Wn bit (set to 0) and an acknowledge bit is transmitted by the external I2C device to the I2Ci module.

Each time a data byte is requested to be transmitted, the I2Ci.I2C_STAT[4] XRDY bit is set to 1 by the I2Ci module. When it occurs, the data byte must be written into the I2Ci.I2C_DATA register and the I2Ci.I2C_STAT[4] XRDY bit must be cleared by setting it to 1. A software time-out counter can be implemented to detect when no data request is generated by the hardware.

The number of data bytes requested to be transmitted is configured in the I2Ci.I2C_CNT[15:0] DCOUNT bit field. This number is set to 0x02 for the I2C1 module and set to 0x03 for the I2C3 module.

Note: The transmit FIFO threshold must be configured to 1 by configuring the I2Ci.I2C_BUF[5:0] XTRSH field to 0x00 (done by default).

Table 18-22 shows all the registers to be configured for the data transfer.

Table 18-22. Registers Print for the Data Transfer

Register name	Address	Value	Value description
I2C1.I2C_STAT	0x4807 0008	0x0010	The XRDY is set to 1 by the I2C1 module
I2C3.I2C_STAT	0x4806 0008	0x0010	The XRDY is set to 1 by the I2C3 module

18.6.1.3.2.5 Transfer Completion

Once all data bytes have been transmitted, the stop condition is generated on the I2Ci bus and the I2Ci.I2C_STAT[2] ARDY status bit is set to 1 by the hardware.

Table 18-23 shows all the registers to be configured for the transfer completion.

Table 18-23. Registers Print for the Transfer Completion

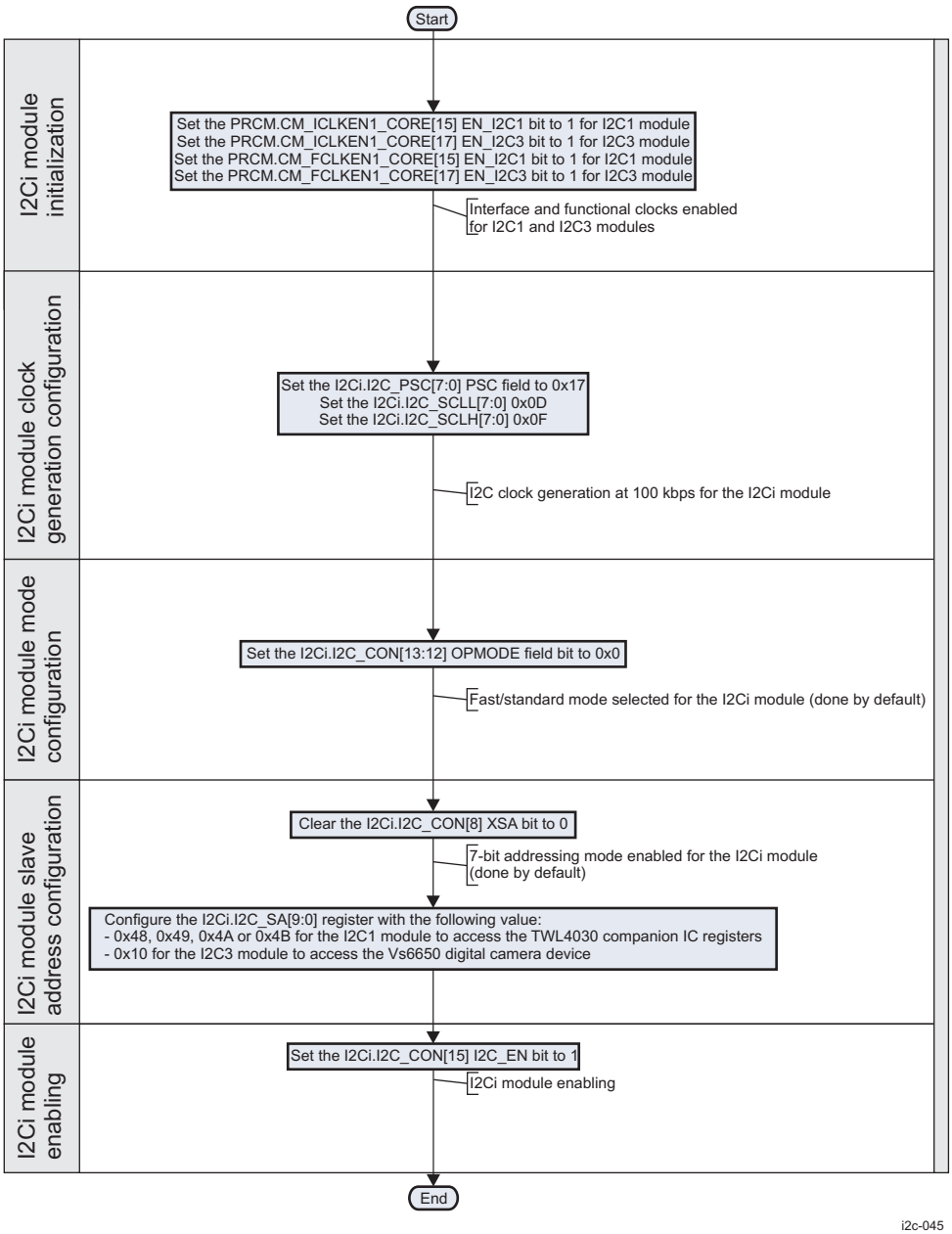
Register name	Address	Value	Value description
I2C1.I2C_STAT	0x4807 0024	0x8604	The ARDY is set to 1 by the I2C1 module (transfer completed)
I2C3.I2C_STAT	0x4806 0024	0x8604	The ARDY is set to 1 by the I2C3 module (transfer completed)

18.6.1.3.3 Flowcharts

18.6.1.3.3.1 Initial Configuration Flowchart

Figure 18-45 describes the programming flowchart corresponding to the initial configuration. See Section 18.6.1.3.1 for more information.

Figure 18-45. Initial Configuration Flowchart



18.6.1.3.3.2 Operational Mode Flowchart

Figure 18-46 describes the programming flowchart for writing a data byte into the TWL4030 companion chip by the I2C1 module. See Section 18.6.1.3.2 for more information. See Table 18-24 for more information about variables and constants used in the programming flowcharts in Figure 18-46 and Figure 18-47.

Figure 18-46. Data Writing into the TWL4030 Companion Chip Register by the I2C1 Module

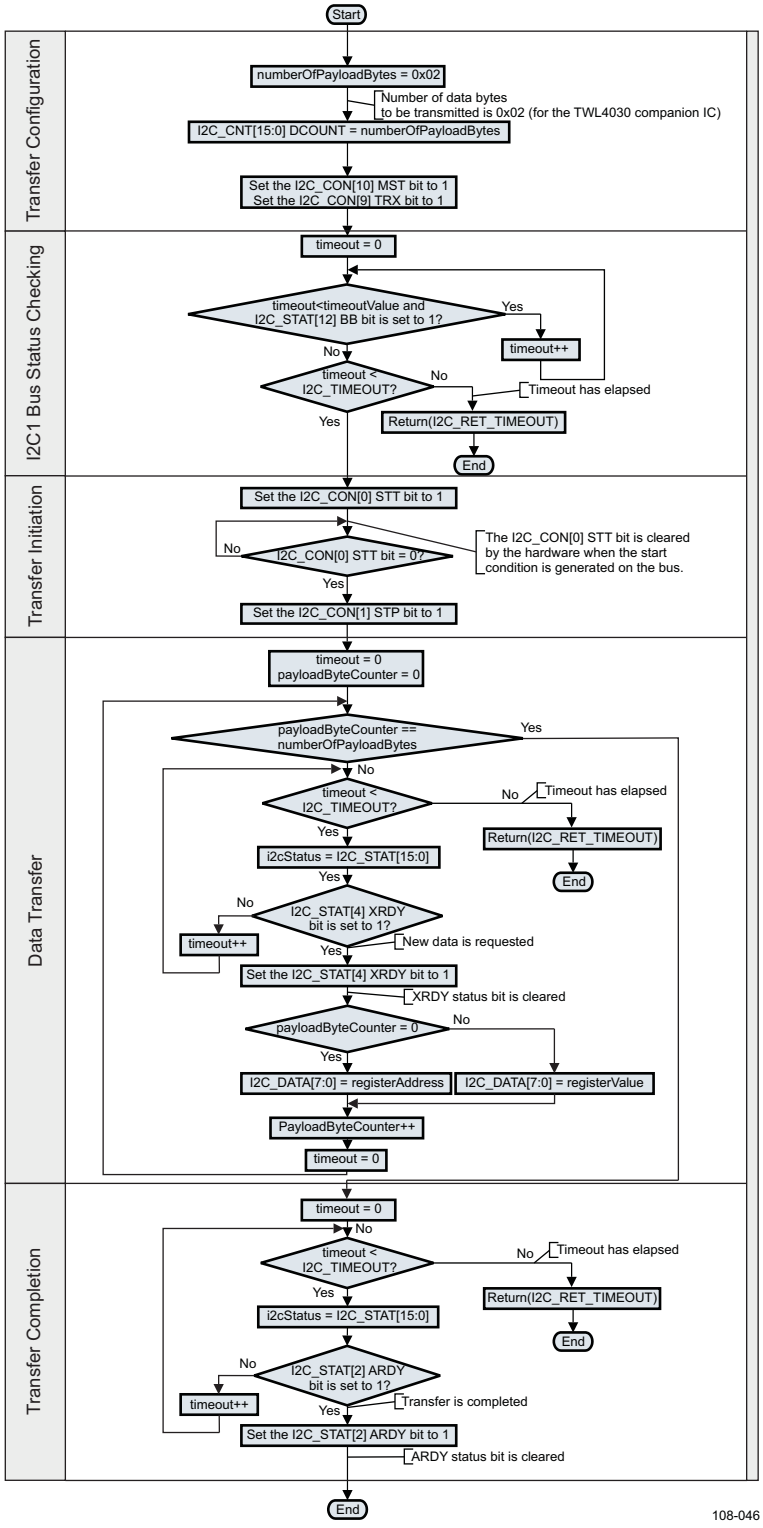
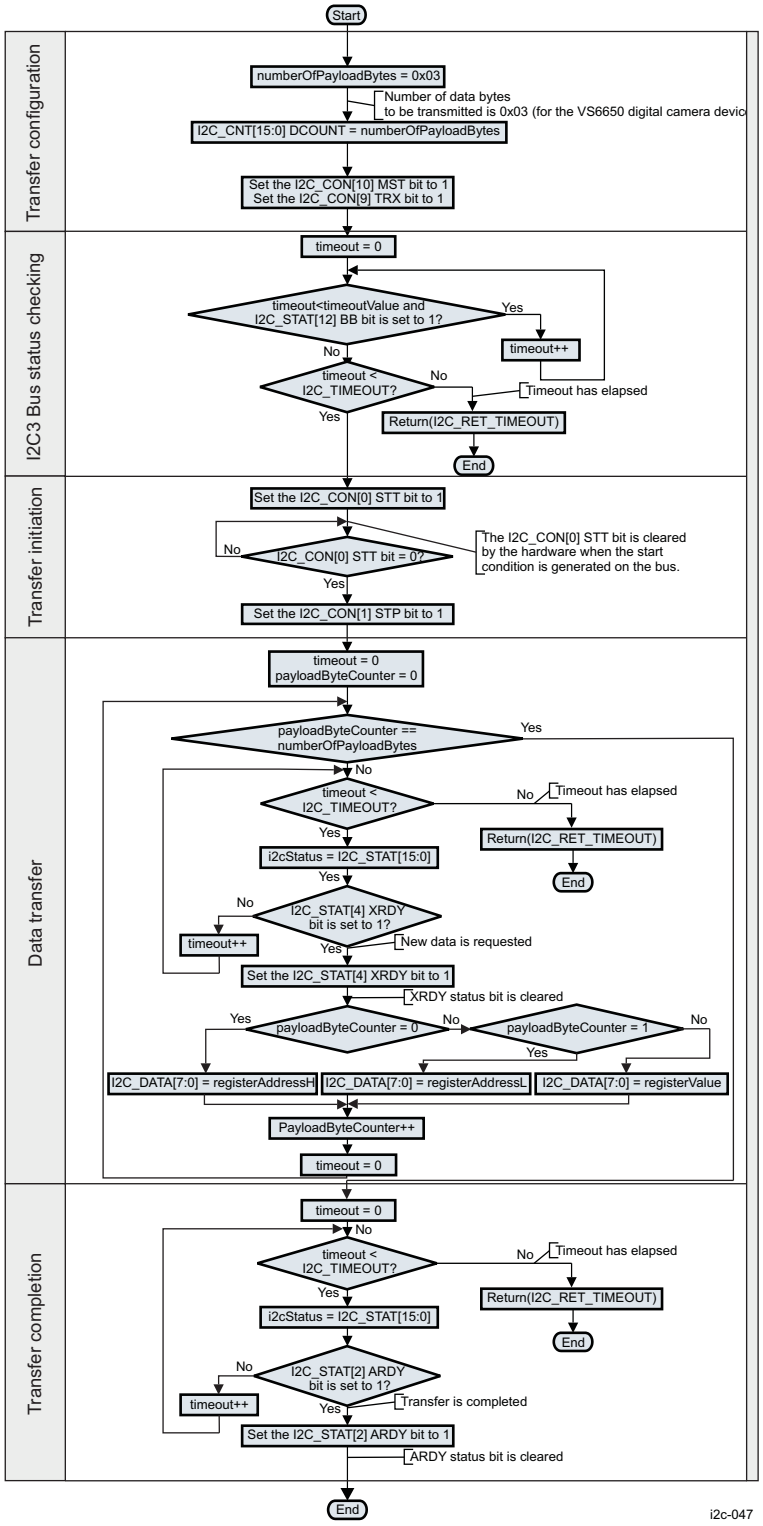


Figure 18-47 describes the programming flowchart for writing a data byte into the VS6650 digital camera device by the I2C3 module. See Section 18.6.1.3.2 for more information.

Figure 18-47. Data Writing into the VS6650 Digital Camera Device Register by the I2C3 Module



The variables and constants used in the programming flowcharts are described in the Table 18-24.

Table 18-24. Variables and Constants

Variable or Constant Name	Type	Tested Value	Description
numberOfPayloadBytes	16-bit unsigned	-	This variable contains the number of data bytes to be transmitted to the external device.
timeout	32-bit unsigned	-	This variable is the timeout counter. When it reaches the I2C_TIMEOUT value, it indicates that the timeout has elapsed.
I2C_TIMEOUT	32-bit unsigned	100 000	This constant contains the maximum value that can be reached by the timeout variable.
payloadByteCounter	16-bit unsigned	-	This variable is a counter that counts the data bytes transmitted to the external device.
registerAddress	8-bit unsigned	-	This variable contains the address of the TWL4030 companion chip register that must be written.
registerAddressM	8-bit unsigned	-	This variable contains the most-significant byte address of the VS6650 digital camera device register that must be written.
registerAddressL	8-bit unsigned	-	This variable contains the less-significant byte address of the VS6650 digital camera device register that must be written.
registerValue	8-bit unsigned	-	This variable contains the value to be written into the external device register

18.7 High-Speed I2C Controllers Register Manual

CAUTION

The I2Ci registers are limited to 16 bit and 8 bit data accesses, 32 bit data access is not allowed and can corrupt register content.

Note: For details about the master transmitter HS I2C controller I2C4, see Chapter 4, *Power, Reset, and Clock Management*.

Table 18-25 provides the instance summary.

Table 18-25. Instance Summary

Module Name	Base Address	Size
I2C1	0x4807 0000	128 bytes
I2C2	0x4807 2000	128 bytes
I2C3	0x4806 0000	128 bytes

18.7.1 Multimaster HS I2C Controller Register Mapping Summary

Table 18-26 lists the I2Ci registers (where i = 1, 2 and 3).

Table 18-26. Register Summary

Register Name	Type	Register Width (Bits)	Address Offset	Physical Address for I2C1	Physical Address for I2C2	Physical Address for I2C3
I2C_REV	R	32	0x00	0x4807 0000	0x4807 2000	0x4806 0000
I2C_IE	RW	32	0x04	0x4807 0004	0x4807 2004	0x4806 0004
I2C_STAT	RW	32	0x08	0x4807 0008	0x4807 2008	0x4806 0008
I2C_WE	RW	32	0x0C	0x4807 000C	0x4807 200C	0x4806 000C
I2C_SYSS	R	32	0x10	0x4807 0010	0x4807 2010	0x4806 0010
I2C_BUF	RW	32	0x14	0x4807 0014	0x4807 2014	0x4806 0014
I2C_CNT	RW	32	0x18	0x4807 0018	0x4807 2018	0x4806 0018
I2C_DATA	RW	32	0x1C	0x4807 001C	0x4807 201C	0x4806 001C
I2C_SYSC	RW	32	0x20	0x4807 0020	0x4807 2020	0x4806 0020
I2C_CON	RW	32	0x24	0x4807 0024	0x4807 2024	0x4806 0024
I2C_OA0	RW	32	0x28	0x4807 0028	0x4807 2028	0x4806 0028
I2C_SA	RW	32	0x2C	0x4807 002C	0x4807 202C	0x4806 002C
I2C_PSC	RW	32	0x30	0x4807 0030	0x4807 2030	0x4806 0030
I2C_SCLL	RW	32	0x34	0x4807 0034	0x4807 2034	0x4806 0034
I2C_SCLH	RW	32	0x38	0x4807 0038	0x4807 2038	0x4806 0038
I2C_SYSTEST	RW	32	0x3C	0x4807 003C	0x4807 203C	0x4806 003C
I2C_BUFSTAT	R	32	0x40	0x4807 0040	0x4807 2040	0x4806 0040
I2C_OA1	RW	32	0x44	0x4807 0044	0x4807 2044	0x4806 0044
I2C_OA2	RW	32	0x48	0x4807 0048	0x4807 2048	0x4806 0048
I2C_OA3	RW	32	0x4C	0x4807 004C	0x4807 204C	0x4806 004C
I2C_ACTOA	R	32	0x50	0x4807 0050	0x4807 2050	0x4806 0050
I2C_SBLOCK	RW	32	0x54	0x4807 0054	0x4807 2054	0x4806 0054

18.7.2 Register Description

Table 18-27. I2C_REV

Address Offset	0x00		
Physical Address	0x4806 0000	**Instance**	I2C3
	0x4807 0000		I2C1
	0x4807 2000		I2C2
Description	This register contains the IP revision code.		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																Reserved								REV							

Bits	Field Name	Description	Type	Reset
31:16	Reserved	Read returns 0.	R	0x0000
15:8	Reserved	Read returns 0.	R	0x00
7:0	REV	IP revision [7:4] Major revision [3:0] Minor revision Examples: 0x30 for 3.0, 0x31 for 3.1	R	See (1)

(1) TI internal data

Table 18-28. Register Call Summary for Register I2C_REV

High-Speed I2C Controllers Register Manual

- Multimaster HS I2C Controller Register Mapping Summary: [0]

Table 18-29. I2C_IE

Address Offset	0x04		
Physical Address	0x4806 0004	**Instance**	I2C3
	0x4807 0004		I2C1
	0x4807 2004		I2C2
Description	This register contains the interrupt enable bits.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																Reserved	XDR_IE	RDR_IE	Reserved			AAS_IE	BF_IE	AERR_IE	STC_IE	GC_IE	XRDY_IE	RRDY_IE	ARDY_IE	NACK_IE	AL_IE

Bits	Field Name	Description	Type	Reset
31:16	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0x0000
15	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0
14	XDR_IE	Transmit Draining interrupt enable. Mask or unmask the interrupt signaled by the bit in I2C_STAT[XDR]. 0x0: Transmit Draining interrupt disabled 0x1: Transmit Draining interrupt enabled	RW	0
13	RDR_IE	Receive Draining interrupt enable. Mask or unmask the interrupt signaled by the bit in I2C_STAT[RDR]. 0x0: Receive Draining interrupt disabled 0x1: Receive Draining interrupt enabled	RW	0

Bits	Field Name	Description	Type	Reset
12:10	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0x0
9	AAS_IE	Addressed as Slave interrupt enable. Mask or unmask the interrupt signaled by the bit in I2C_STAT[AAS]. 0x0: Addressed as Slave interrupt disabled 0x1: Addressed as Slave interrupt enabled	RW	0
8	BF_IE	Bus Free interrupt enable. Mask or unmask the interrupt signaled by the bit in I2C_STAT[BF]. 0x0: Bus Free interrupt disabled 0x1: Bus Free interrupt enabled	RW	0
7	AERR_IE	Access Error interrupt enable. Mask or unmask the interrupt signaled by the bit in I2C_STAT[AERR]. 0x0: Access Error interrupt disabled 0x1: Access Error interrupt enabled	RW	0
6	STC_IE	Start Condition interrupt enable. Mask or unmask the interrupt signaled by the bit in I2C_STAT[STC]. 0x0: Start Condition interrupt disabled 0x1: Start Condition interrupt enabled	RW	0
5	GC_IE	General Call interrupt enable. Mask or unmask the interrupt signaled by the bit in I2C_STAT[GC]. 0x0: General Call interrupt disabled 0x1: General Call interrupt enabled	RW	0
4	XRDY_IE	Transmit Data Ready interrupt enable. Mask or unmask the interrupt signaled by the bit in I2C_STAT[XRDY]. 0x0: Transmit Data Ready interrupt disabled 0x1: Transmit Data Ready interrupt enabled	RW	0
3	RRDY_IE	Receive Data Ready interrupt enable. Mask or unmask the interrupt signaled by the bit in I2C_STAT[RRDY]. 0x0: Receive Data Ready interrupt disabled 0x1: Receive Data Ready interrupt enabled	RW	0
2	ARDY_IE	Register Access Ready interrupt enable. Mask or unmask the interrupt signaled by the bit in I2C_STAT[ARDY]. 0x0: Register Access Ready interrupt disabled 0x1: Register Access Ready interrupt enabled	RW	0
1	NACK_IE	No Acknowledgment interrupt enable. Mask or unmask the interrupt signaled by the bit in I2C_STAT[NACK]. 0x0: No Acknowledge interrupt disabled 0x1: No Acknowledge Ready interrupt enabled	RW	0
0	AL_IE	Arbitration Lost interrupt enable. Mask or unmask the interrupt signaled by the bit in I2C_STAT[AL]. 0x0: Arbitration Lost interrupt disabled 0x1: Arbitration Lost interrupt enabled	RW	0

Table 18-30. Register Call Summary for Register I2C_IE

High-Speed I2C Controller Integration • Interrupt Requests: [0] [1] [2] [3] [4] [5] [6] [7] [8] [9] [10] [11] [12] [13] [14]
High-Speed I2C Controller Functional Description • Receive Mode in I2C Mode: [15] • FIFO Interrupt Mode Operation: [16] • FIFO Polling Mode Operation: [17] [18] • Draining Feature (I2C Mode Only): [19] [20]

Table 18-30. Register Call Summary for Register I2C_IE (continued)

High-Speed I2C Controller Basic Programming Model • Main Program: [21] [22] [23] [24] [25] [26] [27] [28] [29] • Main Program: [30] [31] [32] [33] [34]
High-Speed I2C Controllers Register Manual • Multimaster HS I2C Controller Register Mapping Summary: [35]

Table 18-31. I2C_STAT

Address Offset	0x08		
Physical Address	0x4806 0008	**Instance**	I2C3
	0x4807 0008		I2C1
	0x4807 2008		I2C2
Description	This register provides specific status information about the module, including interrupt status information.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																Reserved	XDR	RDR	BB	ROVR	XUDF	AAS	BF	AERR	STC	GC	XRDY	RRDY	ARDY	NACK	AL

Bits	Field Name	Description	Type	Reset
31:16	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0x0000
15	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0
14	XDR	Transmit Draining IRQ status Read 0x0: Transmit draining inactive Read 0x1: Transmit draining active Write 0x0: No effect Write 0x1: Clear this bit to 0	RW	0
13	RDR	Receive Draining IRQ status Read 0x0: Receive draining inactive Read 0x1: Receive draining active Write 0x0: No effect Write 0x1: Clear this bit to 0	RW	0
12	BB	Bus busy status. Read-only bit. Writing this bit does not affect the read value. Read 0x0: Bus is free. Read 0x1: Bus is occupied.	R	0
11	ROVR	Receive overrun status. Read-only bit. Writing this bit does not affect the read value. Read 0x0: Normal operation Read 0x1: Receiver overrun	R	0
10	XUDF	Transmit underflow status. Read-only bit. Writing this bit does not affect the read value. Read 0x0: Normal operation Read 0x1: Transmit underflow	R	0
9	AAS	Address recognized as slave IRQ status Read 0x0: No action Read 0x1: Address recognized Write 0x0: No effect Write 0x1: Clear this bit to 0.	RW	0
8	BF	Bus Free IRQ status	RW	0

Bits	Field Name	Description	Type	Reset
		Read 0x0: No action		
		Read 0x1: Bus free		
		Write 0x0: No effect		
		Write 0x1: Clear this bit to 0.		
7	AERR	Access Error IRQ status	RW	0
		Read 0x0: No action		
		Read 0x1: Access error		
		Write 0x0: No effect		
		Write 0x1: Clear this bit to 0.		
6	STC	Start Condition IRQ status	RW	0
		Read 0x0: No action		
		Read 0x1: Start condition detected		
		Write 0x0: No effect		
		Write 0x1: Clear this bit to 0.		
5	GC	General Call IRQ status. Set to 1 when general call address detected. Interrupt signaled to MPU subsystem	RW	0
		Read 0x0: No general call detected		
		Read 0x1: General call address detected		
		Write 0x0: No effect		
		Write 0x1: Clear this bit to 0.		
4	XRDY	Transmit Data Ready IRQ status. Set to 1 in transmit mode when new data is requested for transmission. When this bit is set to 1, an interrupt is signaled to MPU subsystem	RW	0
		Read 0x0: No transmit data is requested for transmission		
		Read 0x1: Transmit data is requested for transmission		
		Write 0x0: No effect		
		Write 0x1: Clear this bit to 0.		
3	RRDY	Receive Data Ready IRQ status. Set to 1 when in receive mode, causing new data to be able to be read. When this bit is set to 1, an interrupt is signaled to MPU subsystem.	RW	0
		Read 0x0: No data available		
		Read 0x1: Receive data available		
		Write 0x0: No effect		
		Write 0x1: Clear this bit to 0.		
2	ARDY	Register Access Ready IRQ status. Setting this bit to 1 indicates that previous access has been performed and registers are ready to be accessed again. An interrupt is signaled to MPU subsystem.	RW	0
		Read 0x0: Module busy		
		Read 0x1: Access ready		
		Write 0x0: No effect		
		Write 0x1: Clear this bit to 0.		
1	NACK	No Acknowledgment IRQ status. This bit is set when No Acknowledgment has been received. An interrupt is signaled to MPU subsystem. In master mode, the transfer is automatically ended by generating a stop condition on the bus. The I2Ci.I2C_CON[1] STP, I2Ci.I2C_CON[10] MST and I2Ci.I2C_CON[9] TRX bits are automatically cleared to 0 (slave receiver mode). TX and RX FIFOs must be cleared (I2Ci.I2C_BUF[6] TXFIFO_CLR and I2Ci.I2C_BUF[14] RXFIFO_CLR bits set to 1).	RW	0
		Read 0x0: Normal operation		
		Read 0x1: No Acknowledge detected		
		Write 0x0: No effect		
		Write 0x1: Clear this bit to 0.		
0	AL	Arbitration Lost IRQ status. This bit is set automatically by the hardware when the I2C controller inside the device loses arbitration in master transmit mode. An interrupt is signaled to MPU subsystem.	RW	0
		Read 0x0: Normal operation		

Bits	Field Name	Description	Type	Reset
		Read 0x1: Arbitration loss detected		
		Write 0x0: No effect		
		Write 0x1: Clear this bit to 0.		

Table 18-32. Register Call Summary for Register I2C_STAT

High-Speed I2C Controller Environment
- I2C Typical Connection Protocol and Data Format: [0] [1] [2] [3] [4] [5] [6]

High-Speed I2C Controller Integration
- Interrupt Requests: [7] [8] [9] [10] [11] [12] [13] [14] [15] [16] [17] [18] [19] [20] [21] [22]

High-Speed I2C Controller Functional Description
- Transmit Mode in I2C Mode: [23] [24]
- Receive Mode in I2C Mode: [25] [26] [27] [28]
- FIFO Interrupt Mode Operation: [29] [30]
- FIFO Polling Mode Operation: [31] [32] [33] [34] [35] [36]
- Draining Feature (I2C Mode Only): [37] [38] [39] [40] [41]
- System Test Mode: [42] [43] [44]

High-Speed I2C Controller Basic Programming Model
- Main Program: [45] [46] [47] [48] [49] [50] [51]
- Interrupt Subroutine Sequence: [52] [53] [54] [55] [56] [57] [58] [59] [60]
- Programming Flow Diagrams: [61] [62] [63] [64] [65] [66] [67] [68]
- Main Program: [69] [70] [71] [72]
- Interrupt Subroutine Sequence: [73] [74] [75]
- Programming Flow Diagrams: [76] [77] [78] [79]

High-Speed Multimaster I2C Use Cases and Tips
- Programming Flow: [80] [81] [82] [83] [84] [85] [86] [87] [88] [89] [90]

High-Speed I2C Controllers Register Manual
- Multimaster HS I2C Controller Register Mapping Summary: [91]
- Register Description: [92] [93] [94] [95] [96] [97] [98] [99] [100] [101] [102] [103] [104] [105] [106]

Table 18-33. I2C_WE

Address Offset	0x0C		
Physical Address	0x4806 000C	**Instance**	I2C3
	0x4807 000C		I2C1
	0x4807 200C		I2C2
Description	This register contains the wakeup enable bits.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																Reserved	XDR_WE	RDR_WE	Reserved			AAS_WE	BF_WE	Reserved	STC_WE	GC_WE	Reserved	DRDY_WE	ARDY_WE	NACK_WE	AL_WE

Bits	Field Name	Description	Type	Reset
31:16	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0x0000
15	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0
14	XDR_WE	Transmit draining wakeup enable	RW	0
		0x0: Transmit draining wakeup disabled		
		0x1: Transmit draining wakeup enabled		
13	RDR_WE	Receive draining wakeup enable	RW	0
		0x0: Receive draining wakeup disabled		

Bits	Field Name	Description	Type	Reset
		0x1: Receive draining wakeup enabled		
12:10	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0x0
9	AAS_WE	Address as slave wakeup enabled	RW	0
		0x0: Address as slave wakeup disabled		
		0x1: Address as slave wakeup enabled		
8	BF_WE	Bus Free wakeup enable	RW	0
		0x0: Bus Free wakeup disabled		
		0x1: Bus Free wakeup enabled		
7	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0
6	STC_WE	Start Condition wakeup enable	RW	0
		0x0: Start condition wakeup disabled		
		0x1: Start condition wakeup enabled		
5	GC_WE	General call wakeup enable	RW	0
		0x0: General call wakeup disabled		
		0x1: General call wakeup enabled		
4	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0
3	DRDY_WE	Transmit/receive data ready wakeup enable. Mask or unmask the interrupt signaled by bit in I2C_STAT[RRDY]	RW	0
		0x0: Transmit/receive data ready wakeup disabled		
		0x1: Transmit/receive data ready wakeup enabled		
2	ARDY_WE	Register access ready wakeup enable	RW	0
		0x0: Register access ready wakeup disabled		
		0x1: Register access ready wakeup enabled		
1	NACK_WE	No Acknowledgment wakeup enable	RW	0
		0x0: No Acknowledgment wakeup disabled		
		0x1: No Acknowledgment wakeup enabled		
0	AL_WE	Arbitration lost wakeup enable	RW	0
		0x0: Arbitration lost wakeup disabled		
		0x1: Arbitration lost wakeup enabled		

Table 18-34. Register Call Summary for Register I2C_WE

High-Speed I2C Controller Integration

- Power Management: [0] [1] [2] [3] [4] [5] [6] [7] [8] [9] [10] [11]

High-Speed I2C Controllers Register Manual

- Multimaster HS I2C Controller Register Mapping Summary: [12]

Table 18-35. I2C_SYSS

Address Offset	0x10		
Physical Address	0x4806 0010	**Instance**	I2C3
	0x4807 0010		I2C1
	0x4807 2010		I2C2
Description	This register provides status information about the module, excluding the interrupt status information.		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																Reserved								Reserved							RDONE

Bits	Field Name	Description	Type	Reset
31:16	Reserved	Read returns 0.	R	0x0000
15:8	Reserved	Read returns 0.	R	0x00
7:1	Reserved	Read returns 0.	R	0x00
0	RDONE	Internal reset monitoring	R	0
		Read 0x0: Internal module reset in ongoing		
		Read 0x1: Internal module reset complete		

Table 18-36. Register Call Summary for Register I2C_SYSS

High-Speed I2C Controller Integration
- Resets: [0] [1] [2]

High-Speed I2C Controllers Register Manual
- Multimaster HS I2C Controller Register Mapping Summary: [3]

Table 18-37. I2C_BUF

Address Offset	0x14		
Physical Address	0x4806 0014	**Instance**	I2C3
	0x4807 0014		I2C1
	0x4807 2014		I2C2
Description	Receive DMA channel disabled.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																RDMA_EN	RXFIFO_CLR	RTRSH						XDMA_EN	TXFIFO_CLR	XTRSH					

Bits	Field Name	Description	Type	Reset
31:16	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0x0000
15	RDMA_EN	Receive DMA channel enable	RW	0
		0x0: Receive DMA channel disabled		
		0x1: Receive DMA channel enabled		
14	RXFIFO_CLR	Receive FIFO clear	RW	0
		0x0: Normal mode		
		0x1: Rx FIFO is reset		
13:8	RTRSH	Threshold value for FIFO buffer in RX mode is equal to RTRSH + 1.	RW	0x00
7	XDMA_EN	Transmit DMA channel enable	RW	0
		0x0: Transmit DMA channel disabled		
		0x1: Transmit DMA channel enabled		
6	TXFIFO_CLR	Transmit FIFO clear	RW	0
		0x0: Normal mode		
		0x1: Tx FIFO is reset		
5:0	XTRSH	Threshold value for FIFO buffer in TX mode is equal to XTRSH + 1.	RW	0x00

Table 18-38. Register Call Summary for Register I2C_BUF

High-Speed I2C Controller Integration
- Power Management: [0] [1]
- Interrupt Requests: [2] [3] [4] [5] [6] [7] [8] [9]

Table 18-38. Register Call Summary for Register I2C_BUF (continued)

High-Speed I2C Controller Functional Description • Receive Mode in I2C Mode: [10] • FIFO Interrupt Mode Operation: [11] [12] [13] [14] [15] [16] [17] [18] • FIFO Polling Mode Operation: [19] [20] • FIFO DMA Mode Operation (I2C Mode Only): [21] [22] [23] [24] [25] • Draining Feature (I2C Mode Only): [26] [27] [28] [29] [30] [31] [32] [33] • Write and Read Operations in SCCB Mode: [34] [35]
High-Speed I2C Controller Basic Programming Model • Main Program: [36] [37] [38] [39] [40] [41] [42] [43] [44] • Programming Flow Diagrams: [45] [46] • Main Program: [47] [48] [49] [50]
High-Speed Multimaster I2C Use Cases and Tips • Programming Flow: [51]
High-Speed I2C Controllers Register Manual • Multimaster HS I2C Controller Register Mapping Summary: [52] • Register Description: [53] [54]

Table 18-39. I2C_CNT

Address Offset	0x18		
Physical Address	0x4806 0018	**Instance**	I2C3
	0x4807 0018		I2C1
	0x4807 2018		I2C2
Description	This read/write register is used to control the numbers of bytes in the I2C data payload.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																DCOUNT															

Bits	Field Name	Description	Type	Reset
31:16	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0x0000
15:0	DCOUNT	Data count **Note:** Because the transfer length for DCOUNT=0x0000 is 65536, the module does not allow the initiation of zero-data-byte transfers.	RW	0x0000

Table 18-40. Register Call Summary for Register I2C_CNT

High-Speed I2C Controller Integration • Interrupt Requests: [0] [1] [2] [3]
High-Speed I2C Controller Functional Description • Transmit Mode in I2C Mode: [4] • Draining Feature (I2C Mode Only): [5]
High-Speed I2C Controller Basic Programming Model • Main Program: [6] [7] • Programming Flow Diagrams: [8] [9] [10] [11] [12] [13]
High-Speed Multimaster I2C Use Cases and Tips • Programming Flow: [14] [15] [16] [17]
High-Speed I2C Controllers Register Manual • Multimaster HS I2C Controller Register Mapping Summary: [18]

Table 18-41. I2C_DATA

Address Offset	0x1C		
Physical Address	0x4806 001C	**Instance**	I2C3
	0x4807 001C		I2C1
	0x4807 201C		I2C2
Description	This register is the end point/entry point for the LH to read data from or write data to the FIFO buffer.		
	Read accesses from an empty FIFO (i.e. at reset) or write accesses to a full FIFO will return error.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																Reserved								DATA							

Bits	Field Name	Description	Type	Reset
31:16	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0x0000
15:8	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0x00
7:0	DATA	Transmit/Receive FIFO data	RW	0x–

Table 18-42. Register Call Summary for Register I2C_DATA

High-Speed I2C Controller Integration
- DMA Requests: [0] [1] [2] [3] [4] [5]
- Interrupt Requests: [6] [7]

High-Speed I2C Controller Functional Description
- Transmit Mode in I2C Mode: [8]
- Receive Mode in I2C Mode: [9] [10]
- System Test Mode: [11] [12]
- Write and Read Operations in SCCB Mode: [13] [14]

High-Speed I2C Controller Basic Programming Model
- Main Program: [15] [16] [17] [18]
- Main Program: [19] [20]

High-Speed Multimaster I2C Use Cases and Tips
- Programming Flow: [21]

High-Speed I2C Controllers Register Manual
- Multimaster HS I2C Controller Register Mapping Summary: [22]

Table 18-43. I2C_SYSC

Address Offset	0x20		
Physical Address	0x4806 0020	**Instance**	I2C3
	0x4807 0020		I2C1
	0x4807 2020		I2C2
Description	This register controls the various parameters of the L4-Core interconnect interface.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																Reserved						CLOCKACTIVITY		Reserved			IDLEMODE		ENAWAKEUP	SRST	AUTOIDLE

Bits	Field Name	Description	Type	Reset
31:16	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0x0000
15:10	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0x00
9:8	CLOCKACTIVITY	Clock Activity selection bits	RW	0x0
		0x0: Both clocks can be cut off		
		0x1: Only interface clock must be kept active; functional clock can be cut off		
		0x2: Only functional clock must be kept active; interface clock can be cut off		
		0x3: Both clocks must be kept active		
7:5	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0x0
4:3	IDLEMODE	Idle Mode selection bits	RW	0x0
		0x0: Force Idle mode		
		0x1: No Idle mode		
		0x2: Smart Idle mode		
		0x3: Reserved		
2	ENAWAKEUP	Enable wakeup control bit	RW	0
		0x0: Wakeup mechanism is disabled		
		0x1: Wakeup mechanism is enabled		
1	SRST	Software reset. This bit is automatically reset by the hardware. During reads, it always returns 0.	RWI	0
		0x0: Normal mode		
		0x1: The module is reset		
0	AUTOIDLE	Auto Idle enable control bit	RW	0
		0x0: Auto Idle mechanism is disabled		
		0x1: Auto Idle mechanism is enabled		

Table 18-44. Register Call Summary for Register I2C_SYSC

High-Speed I2C Controller Integration
- Power Management: [0] [1] [2] [3] [4] [5] [6] [7] [8] [9] [10]
- Resets: [11] [12]

High-Speed I2C Controllers Register Manual
- Multimaster HS I2C Controller Register Mapping Summary: [13]

Table 18-45. I2C_CON

Address Offset	0x24		
Physical Address	0x4806 0024	**Instance**	I2C3
	0x4807 0024		I2C1
	0x4807 2024		I2C2
Description	This register controls the functional features. Caution: during an active transfer phase (STT has been set to 1), no modification must be done in this register. Changing it may result in unpredictable behavior.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																I2C_EN	Reserved	OPMODE		STB	MST	TRX	XSA	XOA0	XOA1	XOA2	XOA3	Reserved		STP	STT

Bits	Field Name	Description	Type	Reset
31:16	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0x0000
15	I2C_EN	Module enable bit	RW	0
		0x0: Controller in reset. FIFO are cleared and status bits are set to their default value.		
		0x1: Module enabled		
14	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0
13:12	OPMODE	Operation mode selection	RW	0x0
		0x0: I2C Fast/Standard mode		
		0x1: I2C High Speed mode		
		0x2: SCCB mode		
		0x3: Reserved		
11	STB	Start byte mode (master mode only)	RW	0
		0x0: Normal mode		
		0x1: Start byte mode		
10	MST	Master/slave mode selection	RWI	0
		0x0: Slave mode		
		0x1: Master mode		
9	TRX	Transmitter/Receiver mode (master mode only)	RW	0
		0x0: Receiver mode		
		0x1: Transmitter mode		
8	XSA	Expand slave address enable bit	RW	0
		0x0: 7-bit address mode		
		0x1: 10-bit address mode		
7	XOA0	Expand Own Address 0 enable bit (default)	RW	0
		0x0: 7-bit address mode		
		0x1: 10-bit address mode		
6	XOA1	Expand Own Address 1 enable bit	RW	0
		0x0: 7-bit address mode		
		0x1: 10-bit address mode		
5	XOA2	Expand Own Address 2 enable bit	RW	0
		0x0: 7-bit address mode		
		0x1: 10-bit address mode		
4	XOA3	Expand Own Address 3 enable bit	RW	0
		0x0: 7-bit address mode		
		0x1: 10-bit address mode		
3:2	Reserved	Write 0s for future compatibility. Read returns 0	RW	0x0
1	STP	Stop condition (master mode only). The STP bit is cleared by the module itself once it has generated and detected the programmed stop condition on the bus.	RW	0
		0x0: No action or generated stop (P) condition detected on the bus (by the module)		
		0x1: Stop condition queried		
0	STT	Start condition (master mode only). The STT bit is cleared by the module itself once it has generated and detected the programmed start condition on the bus.	RW	0
		0x0: No action or generated start (S) condition detected (by the module)		
		0x1: Start condition queried		

Table 18-46. Register Call Summary for Register I2C_CON

High-Speed I2C Controller Environment

- SCCB Interface Typical Connections: [0]

Table 18-46. Register Call Summary for Register I2C_CON (continued)

High-Speed I2C Controller Integration • Resets: [1] [2] [3] [4] [5] • Interrupt Requests: [6] [7] [8] [9] [10] [11] [12]
High-Speed I2C Controller Functional Description • Block Diagram: [13] [14] • Transmit Mode in I2C Mode: [15] [16] • Receive Mode in I2C Mode: [17] • FIFO Interrupt Mode Operation: [18] [19] • Programmable Multislave Channel Feature (I2C Mode Only): [20] [21] [22] [23] • Clocking: [24] • System Test Mode: [25] [26] [27] • Write and Read Operations in SCCB Mode: [28] [29] [30] [31]
High-Speed I2C Controller Basic Programming Model • Main Program: [32] [33] [34] [35] [36] [37] [38] [39] [40] [41] • Programming Flow Diagrams: [42] [43] [44] [45] [46] [47] [48] [49] [50] [51] [52] [53] [54] [55] [56] [57] [58] [59] [60] [61] [62] [63] [64] [65] • Main Program: [66] [67] [68] [69] [70] [71]
High-Speed Multimaster I2C Use Cases and Tips • Programming Flow: [72] [73] [74] [75] [76] [77] [78] [79] [80] [81] [82] [83] [84] [85] [86] [87] [88] [89] [90] [91] [92] [93] [94]
High-Speed I2C Controllers Register Manual • Multimaster HS I2C Controller Register Mapping Summary: [95] • Register Description: [96] [97] [98]

Table 18-47. I2C_OA0

Address Offset	0x28		
Physical Address	0x4806 0028	**Instance**	I2C3
	0x4807 0028		I2C1
	0x4807 2028		I2C2
Description	This register is used to specify the module I2C 7-bit or 10-bit address for the I2C operations or the 8-bit subaddress of the SCCB module register for the SCCB operations.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																MCODE			Reserved						OA							

Bits	Field Name	Description	Type	Reset
31:16	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0x0000
15:13	MCODE	Master C ode value	RW	0x0
12:10	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0x0
9:0	OA	Own address 0 value	RW	0x000

Table 18-48. Register Call Summary for Register I2C_OA0

High-Speed I2C Controller Functional Description • Write and Read Operations in SCCB Mode: [0] [1]
High-Speed I2C Controller Basic Programming Model • Main Program: [2] • Main Program: [3]
High-Speed I2C Controllers Register Manual • Multimaster HS I2C Controller Register Mapping Summary: [4]

Table 18-49. I2C_SA

Address Offset	0x2C		
Physical Address	0x4806 002C	**Instance**	I2C3
	0x4807 002C		I2C1
	0x4807 202C		I2C2
Description	This register is used to specify the addressed I2C module 7-bit or 10-bit address for the I2C operations or the 7-bit address of the external SCCB module for the SCCB operations.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																Reserved								SA							

Bits	Field Name	Description	Type	Reset
31:16	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0x0000
15:10	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0x00
9:0	SA	Slave address value.	RW	0x3FF

Table 18-50. Register Call Summary for Register I2C_SA

High-Speed I2C Controller Functional Description
- Write and Read Operations in SCCB Mode: [0] [1]

High-Speed I2C Controller Basic Programming Model
- Main Program: [2]
- Programming Flow Diagrams: [3] [4] [5] [6] [7] [8]
- Main Program: [9]

High-Speed Multimaster I2C Use Cases and Tips
- Programming Flow: [10] [11] [12] [13] [14] [15]

High-Speed I2C Controllers Register Manual
- Multimaster HS I2C Controller Register Mapping Summary: [16]

Table 18-51. I2C_PSC

Address Offset	0x30		
Physical Address	0x4806 0030	**Instance**	I2C3
	0x4807 0030		I2C1
	0x4807 2030		I2C2
Description	This register is used to specify the internal clocking of the I2C peripheral core. The core logic is sampled at the clock rate of the functional clock for the module divided by (PSC+1).		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																Reserved								PSC							

Bits	Field Name	Description	Type	Reset
31:16	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0x0000
15:8	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0x00
7:0	PSC	Fast/Standard and SCCB modes prescale sampling clock divider value	RW	0x00

Table 18-52. Register Call Summary for Register I2C_PSC

High-Speed I2C Controller Integration
- Clocks: [0] [1] [2]

Table 18-52. Register Call Summary for Register I2C_PSC (continued)

High-Speed I2C Controller Functional Description • Clocking: [3] [4] • Noise Filter: [5] [6] • System Test Mode: [7]
High-Speed I2C Controller Basic Programming Model • Main Program: [8] [9] • Main Program: [10] [11]
High-Speed Multimaster I2C Use Cases and Tips • Programming Flow: [12] [13] [14]
High-Speed I2C Controllers Register Manual • Multimaster HS I2C Controller Register Mapping Summary: [15]

Table 18-53. I2C_SCLL

Address Offset	0x34		
Physical Address	0x4806 0034	**Instance**	I2C3
	0x4807 0034		I2C1
	0x4807 2034		I2C2
Description	This register is used to determine the SCL low time value when master.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																HSSCLL								SCLL							

Bits	Field Name	Description	Type	Reset
31:16	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0x0000
15:8	HSSCLL	I2C High Speed mode SCL low time value.	RW	0x00
7:0	SCLL	I2C Fast/Standard or SCCB modes SCL low time value	RW	0x00

Table 18-54. Register Call Summary for Register I2C_SCLL

High-Speed I2C Controller Functional Description • Clocking: [0] [1] [2] [3] [4] [5] [6] • System Test Mode: [7]
High-Speed I2C Controller Basic Programming Model • Main Program: [8] [9] • Main Program: [10]
High-Speed Multimaster I2C Use Cases and Tips • Programming Flow: [11] [12] [13]
High-Speed I2C Controllers Register Manual • Multimaster HS I2C Controller Register Mapping Summary: [14]

Table 18-55. I2C_SCLH

Address Offset	0x38		
Physical Address	0x4806 0038	**Instance**	I2C3
	0x4807 0038		I2C1
	0x4807 2038		I2C2
Description	This register is used to determine the SCL low time value when master.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																HSSCLH								SCLH							

Bits	Field Name	Description	Type	Reset
31:16	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0x0000
15:8	HSSCLH	I2C high-speed mode SCL high time value	RW	0x00
7:0	SCLH	I2C Fast/Standard or SCCB modes SCL high time value	RW	0x00

Table 18-56. Register Call Summary for Register I2C_SCLH

High-Speed I2C Controller Functional Description
- Clocking: [0] [1] [2] [3] [4] [5] [6]
- System Test Mode: [7]

High-Speed I2C Controller Basic Programming Model
- Main Program: [8] [9]
- Main Program: [10]

High-Speed Multimaster I2C Use Cases and Tips
- Programming Flow: [11] [12] [13]

High-Speed I2C Controllers Register Manual
- Multimaster HS I2C Controller Register Mapping Summary: [14]

Table 18-57. I2C_SYSTEST

Address Offset	0x3C		
Physical Address	0x4806 003C	**Instance**	I2C3
	0x4807 003C		I2C1
	0x4807 203C		I2C2
Description	This register is used to facilitate system-level tests by overriding some of the standard functional features of the peripheral.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																ST_EN	FREE	TMODE		SSB	Reserved			SCCBE_O	SCL_I	SCL_O	SDA_I	SDA_O			

Bits	Field Name	Description	Type	Reset
31:16	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0x0000
15	ST_EN	System test enable	RW	0
		0x0: Normal mode		
		0x1: System test enabled. Permit other system test registers bits to be set		
14	FREE	Free-running mode	RW	0
		0x0: Stop mode (on breakpoint condition). If Master mode, it stops after completion of the ongoing bit transfer. In slave mode, it stops during the phase transfer when 1 byte is completely transmitted/received.		
		0x1: Free-running mode		

Bits	Field Name	Description	Type	Reset
13:12	TMODE	Test mode select	R	0
		0x0: Functional mode (default)		
		0x1: Reserved		
		0x2: Test of SCL counters (SCLL, SCLH, PSC). SCL provides a permanent clock with master mode.		
		0x3: Loop back mode select + SDA/SCL IO mode select		
11	SSB	Set status bits	R	0
		0x0: No action		
		0x1: Set all interrupt status bits to 1		
10:5	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0x00
4	SCCBE_O	SCCBE line sense output value. Writing is possible only if ST_EN bit is set to 1	RW	0
		0x0: Write 0 to SCCBE line		
		0x1: Write 1 to SCCBE line		
3	SCL_I	SCL line sense input value	R	0
		0x0: Read 0 from SCL line		
		0x1: Read 1 from SCL line		
2	SCL_O	SCL line drive output value. Writing is possible only if ST_EN bit is set to 1	RW	0
		0x0: Write 0 to SCL line		
		0x1: Write 1 to SCL line		
1	SDA_I	SDA line sense input value	R	0
		0x0: Read 0 from SDA line		
		0x1: Read 1 from SDA line		
0	SDA_O	SDA line drive output value. Writing is possible only if ST_EN bit is set to 1	RW	0
		0x0: Write 0 to SDA line		
		0x1: Write 1 to SDA line		

Table 18-58. Register Call Summary for Register I2C_SYSTEST

High-Speed I2C Controller Functional Description
- System Test Mode: [0] [1] [2] [3] [4] [5] [6] [7] [8]

High-Speed I2C Controllers Register Manual
- Multimaster HS I2C Controller Register Mapping Summary: [9]

Table 18-59. I2C_BUFSTAT

Address Offset	0x40		
Physical Address	0x4806 0040	**Instance**	I2C3
	0x4807 0040		I2C1
	0x4807 2040		I2C2
Description	This register contains the FIFO status information.		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																FIFODEPTH		RXSTAT						Reserved				TXSTAT			

Bits	Field Name	Description	Type	Reset
31:16	Reserved	Write 0s for future compatibility. Read returns 0.	R	0x0000
15:14	FIFODEPTH (1)	FIFO depth indication. Read 0x0: 8-bytes FIFO Read 0x1: 16-bytes FIFO Read 0x2: 32-bytes FIFO Read 0x3: 64-bytes FIFO	R	0x-
13:8	RXSTAT	RX Buffer Status. It indicates the number of bytes to be read in the RX FIFO when the I2C_STAT[RDR] is asserted (set to 1). This indication is useful only in receiver mode when the draining feature is enabled.	R	0x00
7:6	Reserved	Write 0s for future compatibility. Read returns 0.	R	0x0
5:0	TXSTAT	TX Buffer Status. It indicates the number of bytes to be written in the TX FIFO when the I2C_STAT[XDR] is asserted (set to 1). This indication is useful only in transmitter mode when the draining feature is enabled.	R	0x00

(1) See Table 18-11

Table 18-60. Register Call Summary for Register I2C_BUFSTAT

High-Speed I2C Controller Integration
- Interrupt Requests: [0]

High-Speed I2C Controller Functional Description
- FIFO Management: [1]
- Draining Feature (I2C Mode Only): [2] [3] [4] [5] [6] [7] [8] [9]

High-Speed I2C Controllers Register Manual
- Multimaster HS I2C Controller Register Mapping Summary: [10]

Table 18-61. I2C_OA1

Address Offset	0x44		
Physical Address	0x4806 0044	**Instance**	I2C3
	0x4807 0044		I2C1
	0x4807 2044		I2C2
Description	This register is used to specify the module I2C 7-bit or 10-bit address.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																Reserved								OA1							

Bits	Field Name	Description	Type	Reset
31:16	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0x0000
15:10	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0x00
9:0	OA1	Own address 1 value	RW	0x000

Table 18-62. Register Call Summary for Register I2C_OA1

High-Speed I2C Controllers Register Manual
- Multimaster HS I2C Controller Register Mapping Summary: [0]

Table 18-63. I2C_OA2

Address Offset	0x48		
Physical Address	0x4806 0048	**Instance**	I2C3
	0x4807 0048		I2C1
	0x4807 2048		I2C2
Description	This register is used to specify the module I2C 7-bit or 10-bit address.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																Reserved								OA2							

Bits	Field Name	Description	Type	Reset
31:16	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0x0000
15:10	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0x00
9:0	OA2	Own address 2 value	RW	0x000

Table 18-64. Register Call Summary for Register I2C_OA2

High-Speed I2C Controllers Register Manual

- Multimaster HS I2C Controller Register Mapping Summary: [0]

Table 18-65. I2C_OA3

Address Offset	0x4C		
Physical Address	0x4806 004C	**Instance**	I2C3
	0x4807 004C		I2C1
	0x4807 204C		I2C2
Description	This register is used to specify the module I2C 7-bit or 10-bit address.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																Reserved								OA3							

Bits	Field Name	Description	Type	Reset
31:16	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0x0000
15:10	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0x00
9:0	OA3	Own address 3 value	RW	0x000

Table 18-66. Register Call Summary for Register I2C_OA3

High-Speed I2C Controllers Register Manual

- Multimaster HS I2C Controller Register Mapping Summary: [0]

Table 18-67. I2C_ACTOA

Address Offset	0x50		
Physical Address	0x4806 0050	**Instance**	I2C3
	0x4807 0050		I2C1
	0x4807 2050		I2C2
Description	This register contains the accessed slave Own Address indicators.		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																Reserved												OA3_ACT	OA2_ACT	OA1_ACT	OA0_ACT

Bits	Field Name	Description	Type	Reset
31:16	Reserved	Write 0s for future compatibility. Read returns 0.	R	0x0000
15:4	Reserved	Write 0s for future compatibility. Read returns 0.	R	0x000
3	OA3_ACT	Own Address 3 active	R	0
		Read 0x0: Own Address inactive		
		Read 0x1: Own Address active		
2	OA2_ACT	Own Address 2 active	R	0
		Read 0x0: Own Address inactive		
		Read 0x1: Own Address active		
1	OA1_ACT	Own Address 1 active	R	0
		Read 0x0: Own Address inactive		
		Read 0x1: Own Address active		
0	OA0_ACT	Own Address 0 active	R	0
		Read 0x0: Own Address inactive		
		Read 0x1: Own Address active		

Table 18-68. Register Call Summary for Register I2C_ACTOA

High-Speed I2C Controller Functional Description
- Programmable Multislave Channel Feature (I2C Mode Only): [0]

High-Speed I2C Controllers Register Manual
- Multimaster HS I2C Controller Register Mapping Summary: [1]

Table 18-69. I2C_SBLOCK

Address Offset	0x54		
Physical Address	0x4806 0054	**Instance**	I2C3
	0x4807 0054		I2C1
	0x4807 2054		I2C2
Description	This register controls the slave mode i2c bus lock features.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																Reserved												OA3_EN	OA2_EN	OA1_EN	OA0_EN

Bits	Field Name	Description	Type	Reset
31:16	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0x0000
15:4	Reserved	Write 0s for future compatibility. Read returns 0.	RW	0x000
3	OA3_EN	Enable I2C Clock Blocking for Own Address 3 0x0: I2C Clock Released 0x1: I2C Clock Blocked	RW	0
2	OA2_EN	Enable I2C Clock Blocking for Own Address 2 0x0: I2C Clock Released 0x1: I2C Clock Blocked	RW	0
1	OA1_EN	Enable I2C Clock Blocking for Own Address 1 0x0: I2C Clock Released 0x1: I2C Clock Blocked	RW	0
0	OA0_EN	Enable I2C Clock Blocking for Own Address 0 0x0: I2C Clock Released 0x1: I2C Clock Blocked	RW	0

Table 18-70. Register Call Summary for Register I2C_SBLOCK

High-Speed I2C Controller Functional Description • Automatic Blocking of the I2C Clock Feature (I2C Mode Only): [0]
High-Speed I2C Controllers Register Manual • Multimaster HS I2C Controller Register Mapping Summary: [1]

Multichannel SPI

This chapter describes the four multichannel serial port interface (McSPI) modules for the OMAP34xx multimedia device.

Topic **Page**

19.1 McSPI Overview 2808
19.2 McSPI Environment 2811
19.3 McSPI Functional Interface 2814
19.4 McSPI Integration 2819
19.5 McSPI Functional Description 2823
19.6 McSPI Basic Programming Model 2844
19.7 McSPI Use Cases and Tips 2865
19.8 McSPI Register Manual 2871

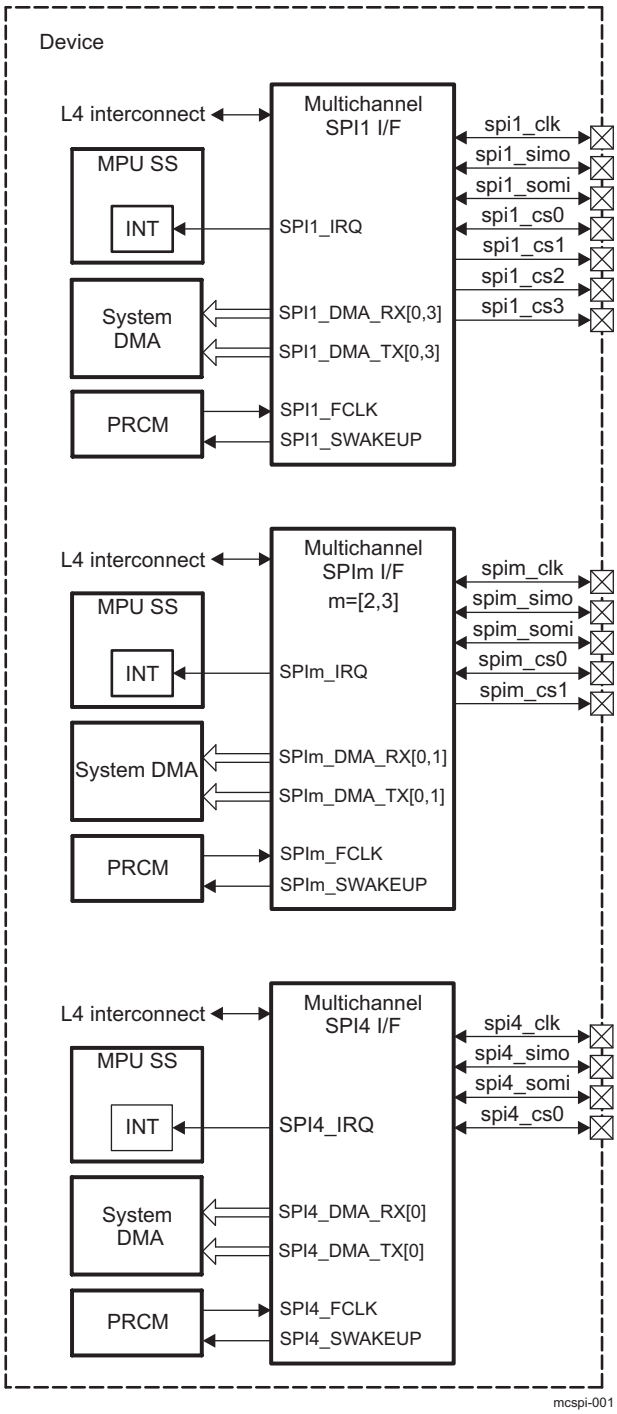
19.1 McSPI Overview

The multichannel serial port interface (McSPI) is a master/slave synchronous serial bus. There are four separate McSPI modules (SPI1, SPI2, SPI3, and SPI4) in the device (see Figure 19-1). The McSPI modules differ as follows: SPI1 supports up to four peripherals, SPI2 and SPI3 support up to two peripherals, and SPI4 supports only one peripheral.

Note: In this chapter, *m=[1,4]* represents the module instance and *x* represents the channel in signal and register naming. There are four McSPI instances. Each module instance has different channel numbers:

- SPI1: 4 channels (if *m=1*, *x=4*)
- SPI2: 2 channels (if *m=2*, *x=2*)
- SPI3: 2 channels (if *m=3*, *x=2*)
- SPI4: 1 channel (if *m=4*, *x=1*)

Figure 19-1. Multichannel Modules SPI1, SPI2, SPI3, and SPI4



The McSPI instances include the following main features:

- Serial clock with programmable frequency, polarity, and phase for each channel
- Wide selection of SPI word lengths ranging from 4 bits to 32 bits
- Up to four master channels or single channel in slave mode
- Master multichannel mode:
  - Full duplex/half duplex
  - Transmit-only/receive-only/transmit-and-receive modes

  - Flexible I/O port controls per channel
  - Two direct memory access (DMA) requests (read/write) per channel
- Single interrupt line for multiple interrupt source events
- Power management through wake-up capabilities
- Enable the addition of a programmable start-bit for SPI transfer per channel (start-bit mode)
- Support start-bit write command
- Support start-bit pause and break sequence
- 64 bytes built-in FIFO available for a single channel
- Force CS mode for continuous transfers

19.2 McSPI Environment

Figure 19-2 shows a simplified overview of a typical application system using the McSPI. This example is based on a TFS chipset (TFS6040-0098), including a 2.2-inch color-active matrix thin-film transistor (TFT) liquid crystal display (LCD) with a light-emitting diode (LED) front light, a four-wire resistive touch-screen panel, and LCD controllers. This chipset is associated with the TSC2005 or TSC2006 touch-screen controller, powered by a TWL4030 power-management unit, and driven by the device. The McSPI device interface is set to master mode; the touch-screen McSPI controller interface operates in slave mode.

Figure 19-2. Typical Application Using the McSPI

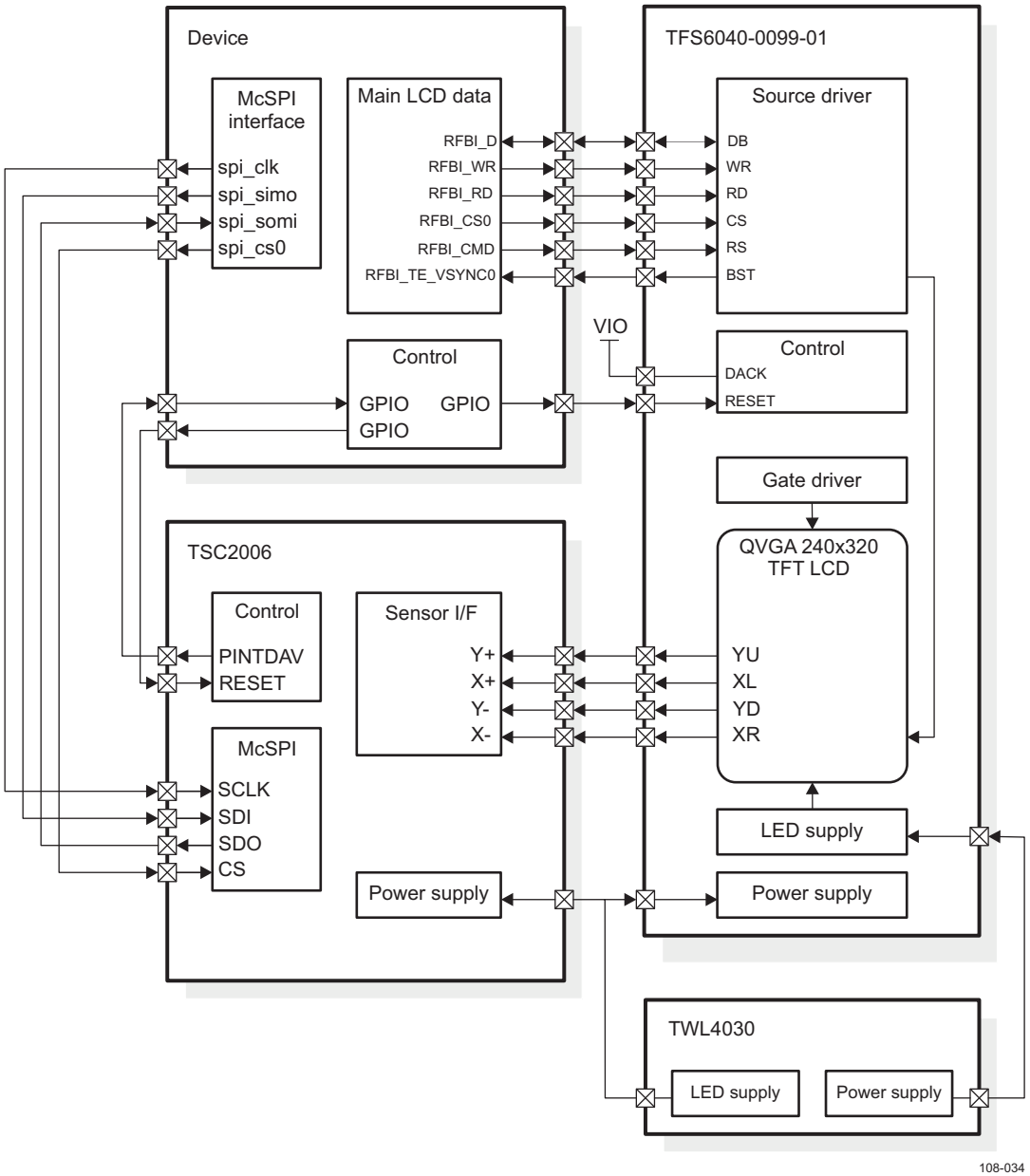
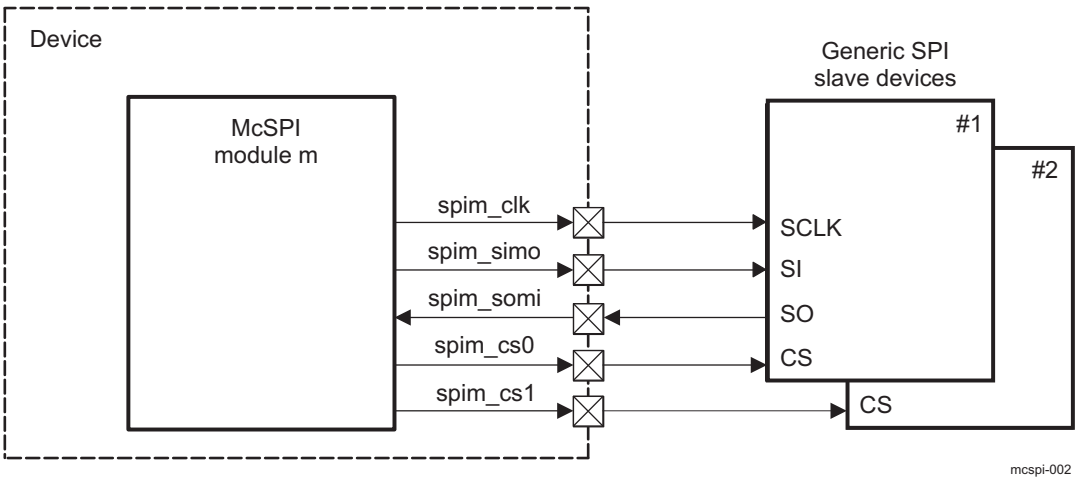


Figure 19-3 through Figure 19-8 show master mode and slave mode configurations. Each mode can be wired on a single-duplex or full-duplex configuration.

19.2.1 SPI Interface in Master Mode

Figure 19-3 shows a case in master mode (full-duplex) where the McSPI module is connected with two slave devices.

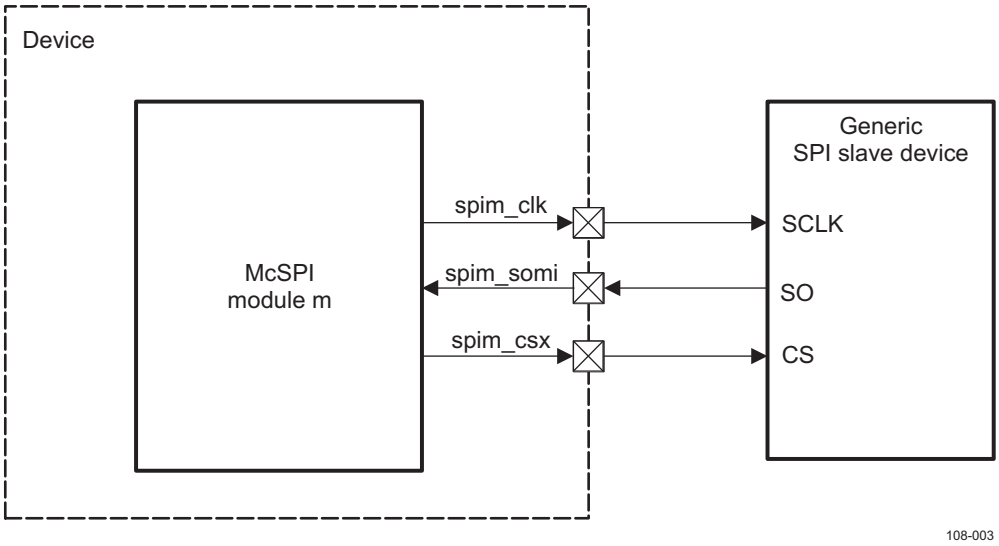
Figure 19-3. McSPI Master Mode (Full-Duplex)



Note: In this case m=[1,3].

Figure 19-4 shows the master single mode, which can also be configured in receive-only mode.

Figure 19-4. McSPI Master Single Mode (Receive-Only)



19.2.2 SPI Interface in Slave Mode

Figure 19-5 shows a case in slave mode (full-duplex).

Note: Only channel 0 can be configured as slave, and only spin_cs0 can be used as a chip-enable in slave mode.

Figure 19-5. McSPI Slave Mode (Full Duplex)

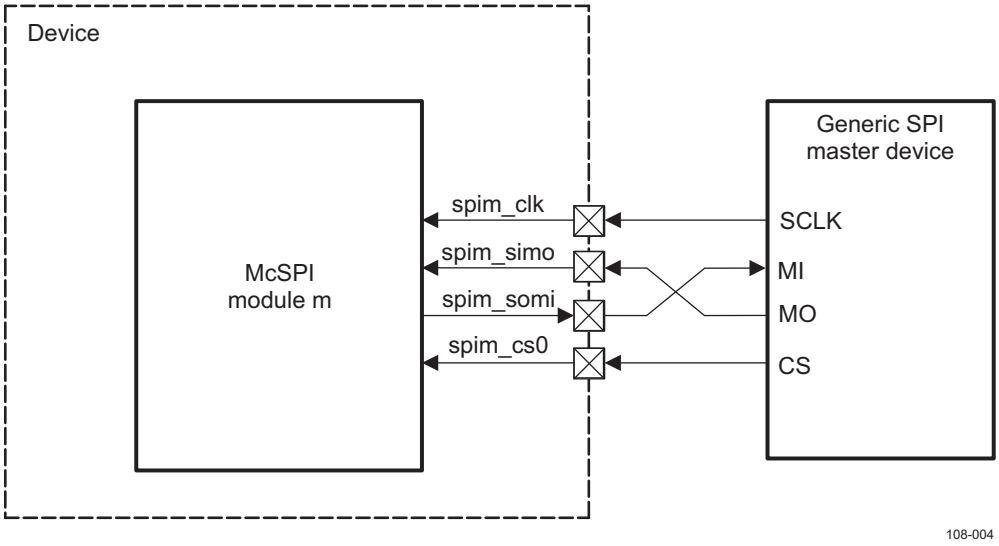
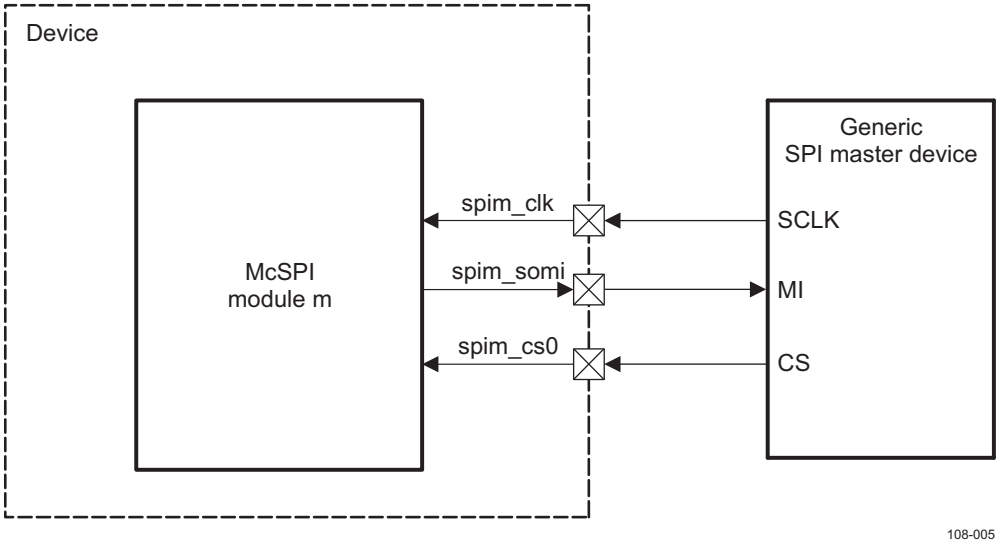


Figure 19-6 shows the slave single mode, which can also be configured in transmit-only mode.

Figure 19-6. McSPI Slave Single Mode (Transmit Only)



19.3 McSPI Functional Interface

19.3.1 Basic McSPI Pins for Master Mode

Figure 19-7 shows all of the McSPI interface signals in master mode.

Figure 19-7. McSPI Interface Signals in Master Mode

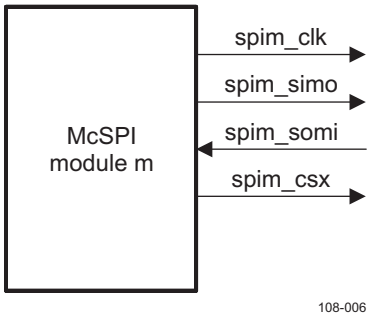


Table 19-1 describes the McSPI I/O in master mode.

Table 19-1. McSPI I/O Description (Master Mode)

Signal Name	I/O	Description	Reset (1)
spim_clk	O	SPIm module serial clock (2)	Unknown
spim_simo	O	SPIm module serial data master out (slave input, master output)	Unknown
spim_somi	I	SPIm module serial data master input (slave output, master input)	–
spim_csx	O	SPIm module chip-select x output	Low

(1) After reset, the SPI modules are in slave mode by default. This paragraph implies that the McSPI module is configured in slave mode. (See the MCSPI_MODULCTRL[2] MS bit in the module control register [MCSPI_MODULCTRL]).

(2) This output signal is also used as re-timing input.

19.3.2 Basic McSPI Pins for Slave Mode

Figure 19-8 shows all of the McSPI interface signals in slave mode.

Figure 19-8. McSPI Interface Signals in Slave Mode

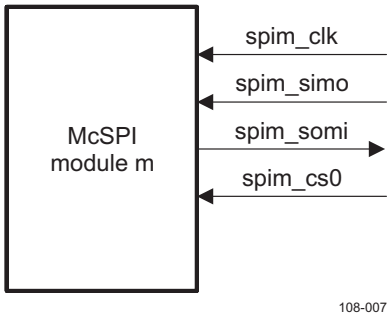


Table 19-2 describes the McSPI I/O in slave mode.

Table 19-2. McSPI I/O Description (Slave Mode)

Signal Name	I/O	Description	Reset [1]
spim_clk	I	SPIm module serial clock	Unknown
spim_simo	I	SPIm module serial data master out (slave input, master output)	Unknown
spim_somi	O	SPIm module serial data master input (slave output, master input)	–
spim_cs0	I/O	SPIm module chip-select 0 input	HiZ
spim_cs1/2/3	O	SPIm module chip-select 1/2/3 output	Low

(1) After reset, the SPI modules are in slave mode by default. This paragraph implies that the McSPI module is configured in slave mode. (See the MCSPI_MODULCTRL[2] MS bit in Module Control Register (MCSPI_MODULCTRL).)

19.3.3 Multichannel SPI Protocol and Data Format

The synchronous SPI protocol allows a master device to initiate serial data transfers to a slave device. A slave select line (spim_csx) allows selection of an individual slave SPI device. Slave devices that are not selected do not interfere with SPI bus activities.

McSPI offers the flexibility to modify the following parameters to adapt to the device features:

- Word length

  McSPI supports any SPI word ranging from 4 bits to 32 bits long (SPIm.MCSPI_CHxCONF[11:7] WL field).

  SPI word length can be changed between transmissions to allow the master device to communicate with peripheral slaves that have different requirements.
- SPI enable (spim_csx, for channel x of instance m)

  The polarity of the SPI enable signals is programmable (SPIm.MCSPI_CHxCONF[6] EPOL bit). spim_csx signals can be active high or low.

  The assertion of the spim_csx signals is programmable and can be manually or automatically asserted. The manual assertion mode is available in single master mode only. spim_csx can be kept active between words with the SPIm.MCSPI_CHxCONF[20] FORCE bit.

  Two consecutive words for two different slave devices can go along with active spim_csx signals with different polarity (see the example in Section 19.6, *McSPI Basic Programming Model*).
- Programmable start-bit

  In start-bit mode a start-bit is added before the SPI word length to indicate how the next SPI word must be handled. The start-bit is enabled by setting SPIm.MCSPI_CHxCONF[23] SBE bit to 1. The SPIm.MCSPI_CHxCONF[24] SBPOL bit defines the polarity of the start-bit.
- Programmable SPI clock
  - Bit rate

    In master mode, the baud rate of the SPI serial clock is programmable using the 48-MHz reference clock (from the power, reset, and clock management [PRCM] module). Table 19-3 gives the spim_clk bit rates obtained for data transfer when programming the clock divider (SPIm.MCSPI_CHxCONF[5:2] CLKD bit field).

Table 19-3. SPI Master Clock Rates

Divider	Clock Rate
1	48 MHz
2	24 MHz
4	12 MHz
8	6 MHz
16	3 MHz
32	1.5 MHz
64	750 kHz

Table 19-3. SPI Master Clock Rates (continued)

Divider	Clock Rate
128	375 kHz
256	~187 kHz
512	~93.7 kHz
1024	~46.8 kHz
2048	~23.4 kHz
4096	~11.7 kHz

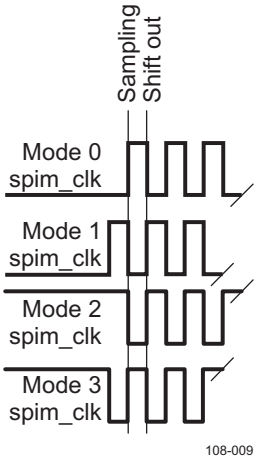
– Polarity and phase

The polarity (the SPIm.MCSPI_CHxCONF[1] POL bit) and the phase (the SPIm.MCSPI_CHxCONF[0] PHA bit) of the SPI serial clock (spim_clk) are configurable to offer four combinations. Software selects the right combination, depending on the device. See Table 19-4 and Figure 19-9.

Table 19-4. Phase and Polarity Combinations

Polarity (POL)	Phase (PHA)	SPI Mode	Comments
0	0	Mode 0	spim_clk is active high and sampling occurs on the rising edge.
0	1	Mode 1	spim_clk is active high and sampling occurs on the falling edge.
1	0	Mode 2	spim_clk is active low and sampling occurs on the falling edge.
1	1	Mode 3	spim_clk is active low and sampling occurs on the rising edge.

Figure 19-9. Phase and Polarity Combinations



19.3.3.1 Transfer Format

In both master and slave modes, McSPI drives the data lines when spim_csx is asserted.

Each word is transmitted starting with the most-significant bit (MSB).

This section explains the two cases of data transmission determined by the clock phase (PHA) and the type of data transmission using a start-bit (SBE) called the start-bit mode:

- Transmission in mode 0 and mode 2 (PHA = 0)

  When PHA = 0, the first bit of the SPI word to transmit (on the master or the slave data output pin) is valid one half cycle of spim_clk after the spim_csx assertion.

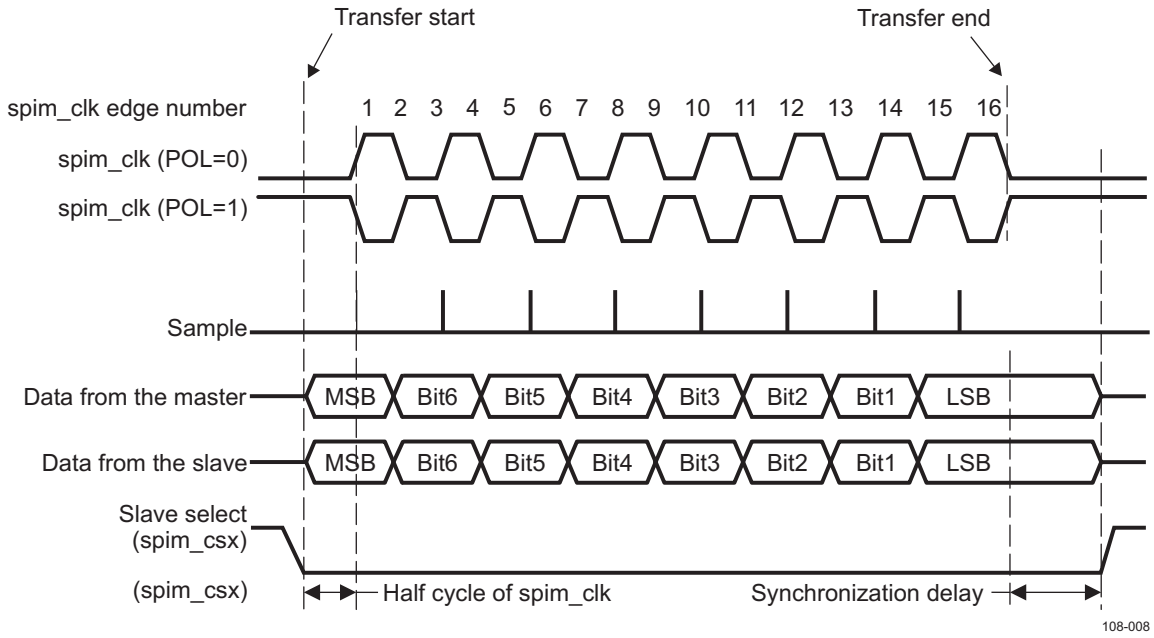
  Therefore, the first edge of the spim_clk line is used by the master to sample the first data bit sent by

the slave. On the same edge, the first data bit sent by the master is sampled by the slave.

On the next spim_clk edge, the received data bit is shifted into the receive shift register and a new data bit is transmitted on the serial data line.

This process continues for a number of pulses on the spim_clk line defined by the SPI word length programmed in the master device, with data being latched on odd-numbered edges and shifted on even-numbered edges. See Figure 19-10.

Figure 19-10. Full-Duplex Transfer Format With PHA = 0



- Transmission in mode 1 and mode 3 (PHA = 1)

  When PHA = 1, the first bit of the SPI word to transmit (on the master or the slave data output pin) is valid on the following spim_clk edge (one half cycle later). This is the sampling edge for the master and slave. A synchronization delay is added between the spim_csx activation and the first spim_clk edge.

  The received data bit is shifted into the shift register on the third spim_clk edge.

  This process continues for a number of pulses on the spim_clk line defined by the SPI word length programmed in the master device, with data being latched on even-numbered edges and shifted on odd-numbered edges.

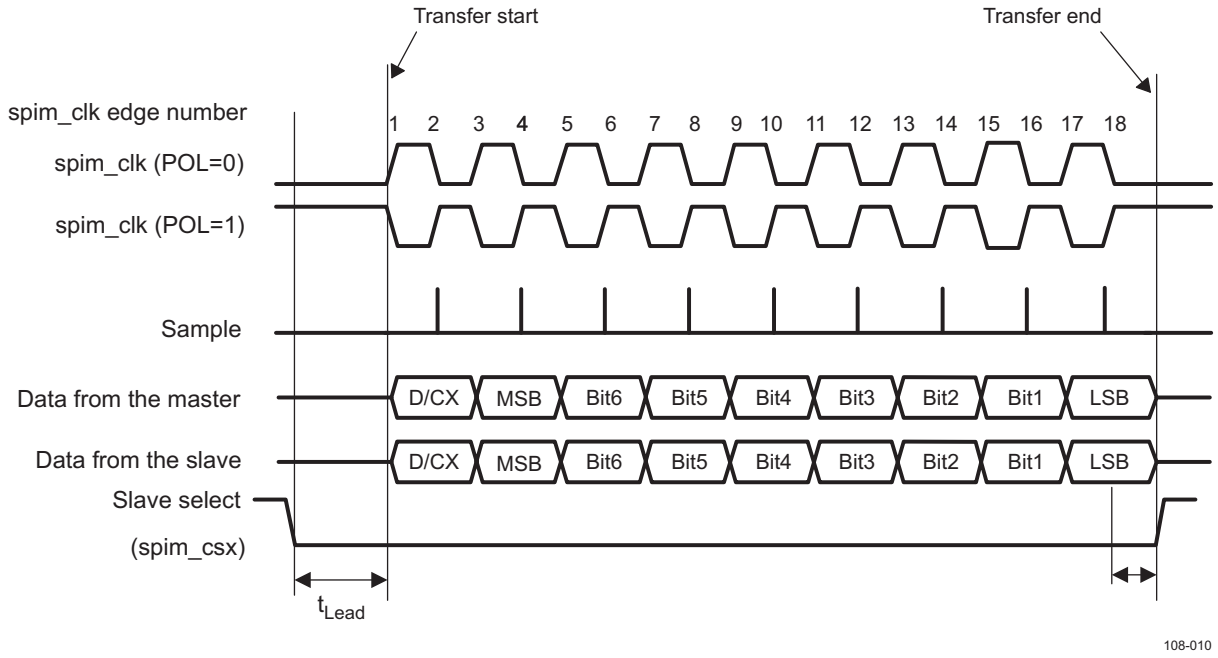
  Note: The minimum synchronization delay is one cycle of spim_clk, if the spim_clk frequency equals the SPIm_FCLK (McSPIm functional clock) frequency in master mode. The minimum synchronization delay is one-half cycle of spim_clk, if the spim_clk frequency is lower than the SPIm_FCLK frequency in both master and slave modes.

- Transmission with a start-bit (SBE = 1)

  When the SPIm.MCSPI_CHxCONF[23] SBE bit = 1, a start-bit is added before the MSB to indicate whether the next SPI word must be handled as a command or as data.

  Figure 19-11 shows an example of a data transfer with an extra start-bit.

Figure 19-11. Extended SPI Transfer With a Start-Bit (SBE = 1)

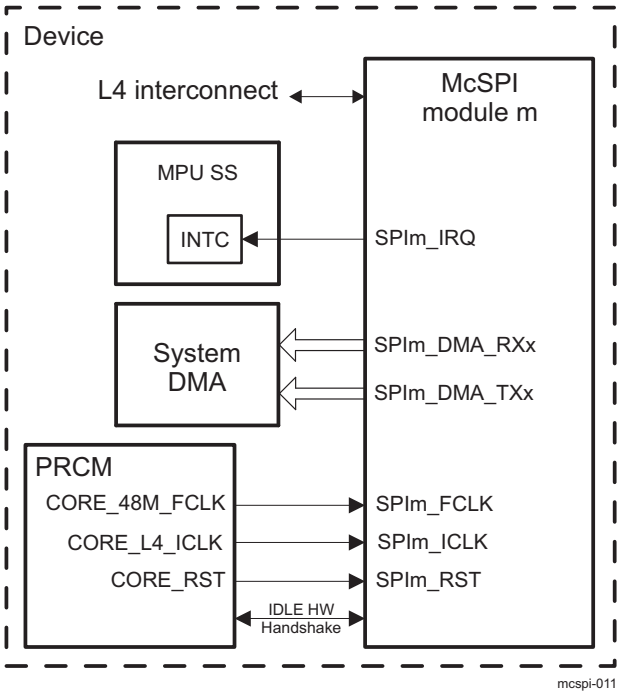


19.4 McSPI Integration

19.4.1 McSPI Description

Figure 19-12 highlights the McSPI module integration in the device.

Figure 19-12. McSPI Integration



19.4.2 Clocking, Reset, and Power-Management Scheme

19.4.2.1 Clocking

Each McSPI module is clocked with an independent functional clock of 48 MHz from the PRCM module (SPIm_FCLK with m = [1,4]) and with an interface clock, CORE_L4_ICLK (L4 interconnect):

- SPIm_FCLK is the McSPI module *m* functional clock and is used to clock the McSPI internal logic. The source of SPIm_FCLK is the PRCM CORE_48M_FCLK output clock.
- SPIm_ICLK is the McSPI module *m* interface clock and is used to synchronize the McSPI L4 port to the L4 interconnect. All accesses from the interconnect are synchronous with SPIm_ICLK. The source of SPIm_ICLK is the PRCM CORE_L4_ICLK output clock.

From a global system power management perspective, when one or both of the McSPI clocks is not required, the McSPI module can be deactivated at the PRCM level in the corresponding registers.

Table 19-5 describes the McSPI module PRCM clock control bits.

Table 19-5. McSPI Clocks

McSPI Clock	Associated PRCM Clock Output	Enable Bit	Autoidle Bit
SPIm_FCLK			
	CORE_48M_FCLK	PRCM.CM_FCLKEN1_CORE.EN_MCSPIm (with m=[1,4])	N/A

Table 19-5. McSPI Clocks (continued)

McSPI Clock	Associated PRCM Clock Output	Enable Bit	Autoidle Bit
SPIm_ICLK			
	CORE_L4_ICLK	PRCM.CM_ICLKEN1_CORE.EN_MCSPIm (with m=[1,4])	PRCM.CM_AUTOIDLE1_CORE.AUTO_MCSPIm (with m=[1,4])

Notes:

- The PRCM CORE_48M_FCLK output is gated at the PRCM level, assuming that all modules sharing it are disabled in the corresponding register. Disabling the McSPI module is a necessary but insufficient condition.
- The PRCM CORE_L4_ICLK output is gated at the PRCM level, assuming that all modules sharing it are disabled in the corresponding register. Disabling the McSPI module is a necessary but insufficient condition.
- The PRCM.CM_AUTOIDLE1_CORE bit is used to link/unlink the McSPI module from CORE_L4_ICLK-related clock domain transitions.

For more information about source clocks gating and domain transitions, see Chapter 4, *Power, Reset, and Clock Management*. For more information on power saving management, see Section 19.5.7.

19.4.2.2 Power Domain

The McSPI modules belong to the CORE power domain (see Table 19-6).

Table 19-6. Power Domain

Peripherals	Power Domain
McSPI modules	CORE power domain

19.4.2.3 Hardware Reset

As part of the CORE power domain, CORE_RST is issued by the PRCM module (for more information about the CORE power domain implementation and CORE_RST signal, see Chapter 4, *Power, Reset, and Clock Management*). The module is reset by hardware when an active-low reset signal is asserted. The hardware reset signal has a global reset effect on the module. All configuration registers and all state-machines are reset in all clock domains (see Table 19-7).

Table 19-7. McSPI Hardware Reset

Peripherals	Name	Comments
McSPI module	CORE_RST	Same as global reset

19.4.2.4 Software Reset

The SPIm.MCSPI_SYSCONFIG[1] SOFTRESET bit controls the software reset of the SPI interface. Writing 1 to this bit enables active software reset functionality, which is equivalent to a hardware reset. The bit is automatically reset by hardware.

Note: The SPIm.MCSPI_SYSCONFIG register is not sensitive to software reset.

19.4.3 Hardware Requests

19.4.3.1 DMA Requests

Each channel includes two DMA requests, one for reception and one for transmission (see Table 19-8).

Table 19-8. DMA Requests

Attributes	DMA Request	Name	Mapping	Comments
SPI1				
	8	SPI1_DMA_TX0	S_DMA_34	Destination is system DMA (sDMA).
		SPI1_DMA_RX0	S_DMA_35	
		SPI1_DMA_TX1	S_DMA_36	
		SPI1_DMA_RX1	S_DMA_37	
		SPI1_DMA_TX2	S_DMA_38	
		SPI1_DMA_RX2	S_DMA_39	
		SPI1_DMA_TX3	S_DMA_40	
		SPI1_DMA_RX3	S_DMA_41	
SPI2				
	4	SPI2_DMA_TX0	S_DMA_42	Destination is sDMA.
		SPI2_DMA_RX0	S_DMA_43	
		SPI2_DMA_TX1	S_DMA_44	
		SPI2_DMA_RX1	S_DMA_45	
SPI3				
	4	SPI3_DMA_TX0	S_DMA_14	Destination is sDMA.
		SPI3_DMA_RX0	S_DMA_15	
		SPI3_DMA_TX1	S_DMA_22	
		SPI3_DMA_RX1	S_DMA_23	
SPI4				
	2	SPI4_DMA_TX0	S_DMA_69	Destination is sDMA.
		SPI4_DMA_RX0	S_DMA_70	

19.4.3.2 Interrupt Requests

Each McSPI module handles one interrupt line (see Table 19-9).

Table 19-9. Interrupt Requests

Attributes	Interrupt Request	Name	Mapping	Comments
SPI1				
	1	SPI1_IRQ	M_IRQ_65	Destination is the microprocessor unit (MPU) interrupt controller.
SPI2				
	1	SPI2_IRQ	M_IRQ_66	Destination is the MPU interrupt controller.
SPI3				
	1	SPI3_IRQ	M_IRQ_91	Destination the MPU interrupt controller.
SPI4				
	1	SPI4_IRQ	M_IRQ_48	Destination is the MPU interrupt controller.

19.4.3.3 Wake-Up Requests

Table 19-10 lists the wake-up requests.

Table 19-10. Wake-Up Requests

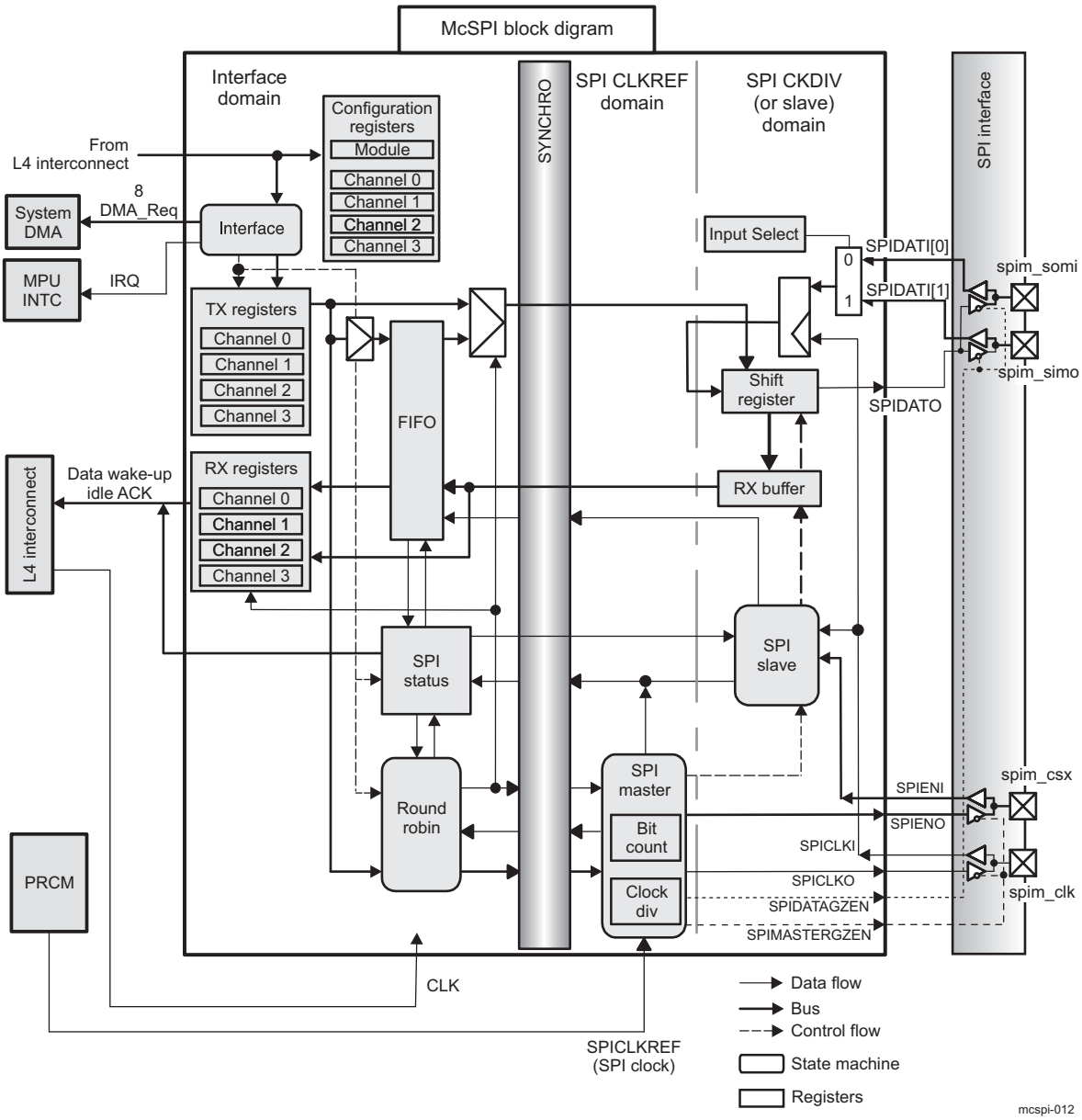
Attributes	Wake Request	Name	Mapping	Comments
SPI1				
	1	spi1_cs0	SPI1_ SWAKEUP	Destination is the PRCM module.
SPI2				
	1	spi2_cs0	SPI2_ SWAKEUP	Destination is the PRCM module.
SPI3				
	1	spi3_cs0	SPI3_ SWAKEUP	Destination is the PRCM module.
SPI4				
	1	spi4_cs0	SPI4_ SWAKEUP	Destination is the PRCM module.

19.5 McSPI Functional Description

19.5.1 McSPI Block Diagram

Figure 19-13 shows the McSPI module.

Figure 19-13. McSPI Block Diagram



19.5.2 Master Mode

19.5.2.1 Master Mode Features

The McSPI master mode supports multichannel communication with up to four independent SPI communication channel contexts. The McSPI initiates a data transfer on the data lines (spim_simo and spim_somi) and generates clock (spim_clk) and control signals (spim_csx).

Connected to multiple external devices, the McSPI exchanges data with one SPI device at a time through two main modes (available in slave mode):

- Two-data-pins interface mode (transmit-and-receive mode for full-duplex transmission)
- Single-data-pin interface mode (recommended for half-duplex transmission)

Two DMA request events (read and write) allow synchronized accesses of the DMA controller with the activity of McSPI.

Three interrupt events can be used for data transmission and reception in master mode (for more information on interrupts, see Section 19.5.5.1, *Interrupt Events in Master Mode*).

19.5.2.2 Master Transmit-and-Receive Mode (Full Duplex)

In full-duplex transmission, data is transmitted (shifted out serially on spim_simo) and received (shifted in serially on spim_somi) simultaneously on separate data lines.

The master transmit-and-receive mode is programmable per channel (the SPIm.MCSPI_CHxCONF[13:12] TRM field).

Channel access to the shift registers for transmission/reception is based on the MCSPI_TXx transmitter register state, the MCSPI_RXx receiver register state, and round robin arbitration.

Channels that meet the following rules are included in the round robin list of active channels scheduled for transmission and/or reception. The arbiter skips channels that do not meet the rules and searches in the rotation for the next enabled channel.

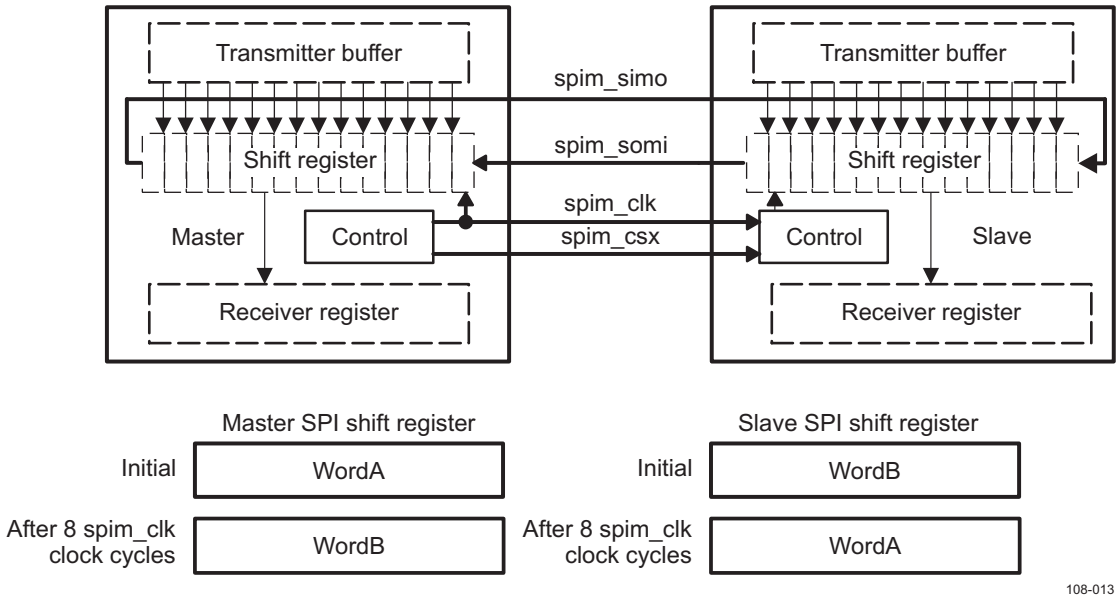
- Rule 1: Only enabled channels (the SPIm.MCSPI_CHxCTRL[0] EN bit) can be scheduled for transmission and/or reception.
- Rule 2: If its MCSPI_TXx transmitter register is not empty (the SPIm.MCSPI_CHxSTAT[1] TXS bit), an enabled channel can be scheduled when the shift register is assigned. If the MCSPI_TXx register is empty when the shift register is assigned, the TXx_UNDERFLOW event is activated, and the next enabled channel with new data to transmit is scheduled (see also the transmit-only mode).
- Rule 3: An enabled channel can be scheduled if its receive register is not full (the SPIm.MCSPI_CHxSTAT[0] RXS bit) when the shift register is assigned (see also the receive-only mode). Therefore, the MCSPI_RXx register cannot be overwritten. Note that the SPI1.MCSPI_IRQSTATUS[3] RX0_OVERFLOW bit is never set to this mode.

When SPI word transfer completes (the SPIm.MCSPI_CHxSTAT[2] EOT bit is set), the updated MCSPI_TXx register of the next scheduled channel is loaded into the shift register. The serialization (transmit-and-receive) starts depending on the channel communication configuration. When serialization completes, the received data transfers to the channel receive register.

The serial clock (spim_clk) synchronizes shifting and sampling of the information on the two serial data lines (spim_simo and spim_somi). Each time a bit transfers out from the master, 1 bit transfers in from the slave.

Figure 19-14 shows an example of a full-duplex system with a master device (McSPI module *m*) on the left and a slave device on the right. After eight cycles of the serial clock spim_clk, WordA transfers from the master to the slave. At the same time, WordB transfers from the slave to the master.

Figure 19-14. SPI Full-Duplex Transmission (Example)



19.5.2.3 Master Transmit-Only Mode (Half Duplex)

The master transmit-only mode prevents the MPU from reading the MCSPI_RXx register (minimizing data movement) when only transmission is meaningful.

The master transmit-only mode is programmable per channel (the SPIm.MCSPI_CHxCONF[13:12] TRM field). Transmission starts only after data is loaded into the MCSPI_TXx register.

Rule 1 and Rule 2, defined in Section 19.5.2.2, are applicable in this mode.

Rule 3, defined in Section 19.5.2.2, is not applicable.

In master transmit-only mode, the MCSPI_RXx register state FULL does not prevent transmission and the MCSPI_RXx register is always overwritten with the new SPI word. This event is not significant when only transmission is meaningful. Thus, the RX0_OVERFLOW bit in the SPIm.MCSPI_IRQSTATUS register is never set in this mode.

The hardware automatically disables the RX_FULL interrupt and the DMA read requests.

The transfer status is given by the SPIm.MCSPI_CHxSTAT[2] EOT bit.

19.5.2.4 Master Receive-Only Mode (Half Duplex)

The master receive mode prevents the MPU from refilling the MCSPI_TXx register (minimizing data movement) when only reception is meaningful.

The master receive mode is programmable per channel (the SPIm.MCSPI_CHxCONF[13:12] TRM field).

The master receive-only mode enables channel scheduling only on the empty state of the MCSPI_RXx register.

Rule 1 and Rule 3, defined in Section 19.5.2.2, are applicable in this mode.

Rule 2, defined in Section 19.5.2.2, is not applicable.

In the master receive-only mode, software must write dummy data to the MCSPI_TXx register. Only one dummy write is enough to receive any number of words from the slave. Software should ensure that the MCSPI_TXx register is always full (the TXx_EMPTY bits of SPIm.MCSPI_IRQSTATUS) when receiving. The content of the MCSPI_TXx register is always loaded into the shift register when the shift register is assigned. After writing the dummy data to the MCSPI_TXx register, the TXx_EMPTY and TXx_UNDERFLOW bits in the SPIm.MCSPI_IRQSTATUS register are never set in receive-only mode.

The SPIm.MCSPI_CHxSTAT[2] EOT bit gives the status of serialization. The RXx_FULL bits of the SPIm.MCSPI_IRQSTATUS register are set when received data is loaded from the shift register to the corresponding MCSPI_RXx register. The SPIm.MCSPI_IRQSTATUS[3] RX0_OVERFLOW bit is never set in this mode.

19.5.2.5 Single-Channel Master Mode

When the McSPI is configured as a master device with a single enabled channel, the assertion of the spim_csx signal can be controlled in two different ways:

- If the MCSPI_MODULCTRL[0] SINGLE bit is set to 0, spim_csx assertion/deassertion after each SPI word is automatically controlled by the McSPI module (see subsections of Section 19.5.2.1, *Master Mode Features*.
- If the MCSPI_MODULCTRL[0] SINGLE bit and the MCSPI_CHxCONF[20] FORCE bit are set to 1: spim_csx assertion/deassertion is controlled by software (see Section 19.5.2.5.1, *Programming Tips When Switching to Another Channel*.

19.5.2.5.1 Programming Tips When Switching to Another Channel

When a single channel is enabled and data transfer is ongoing:

- Wait for completion of the SPI word transfer (wait until the SPIm.MCSPI_CHxSTAT[2] EOT bit is set to 1) before disabling the current channel and enabling a different channel.
- Disable the current channel first, and then enable the other channel.

19.5.2.5.2 Force spim_csx Mode

Continuous transfers are manually allowed by keeping the spim_csx signal active for successive SPI words transfer. Several sequences (configuration/enable/disable of the channel) can be run without deactivating the spim_csx line. This mode is supported by all channels and any master sequence can be used (transmit-receive, transmit-only, receive-only).

Keeping the spim_csx active mode is supported when:

- A single channel is used (with the SPIm.MCSPI_MODULCTRL[0] SINGLE bit set to 1).
- Transfer parameters are loaded in the configuration register of the appropriate channel (SPIm.MCSPI_CHxCONF).

  The state of the spim_csx signal is programmable.

  – Writing 1 to the SPIm.MCSPI_CHxCONF[20] FORCE bit drives the spim_csx line high when the SPIm.MCSPI_CHxCONF[6] EPOL bit is set to 0. spim_csx is driven low when the SPIm.MCSPI_CHxCONF[6] EPOL bit is set to 1.

  – Writing 0 to the SPIm.MCSPI_CHxCONF[20] FORCE bit drives the spim_csx line low when the SPIm.MCSPI_CHxCONF[6] EPOL bit is set to 0. spim_csx is driven high when the SPIm.MCSPI_CHxCONF[6] EPOL bit is set to 1.
- A single channel is enabled (the SPIm.MCSPI_CHxCTRL[0] EN bit is set to 1). The first enabled channel activates the spim_csx line.

When the channel is enabled, the spim_csx signal activates with the programmed polarity. As in the multichannel master mode, the transfer start depends on the status of the MCSPI_TXx register (the SPIm.MCSPI_CHxSTAT[1] TXS bit), the status of the MCSPI_RXx register (the SPIm.MCSPI_CHxSTAT[1] RXS bit), and the defined mode (the SPIm.MCSPI_CHxCONF[13:12] TRM field) of the channel enabled.

The SPIm.MCSPI_CHxSTAT[2] EOT bit gives the transfer status of each SPI word. The RXx_FULL bit in the SPIm.MCSPI_IRQSTATUS register is set when received data is loaded from the shift register to the MCSPI_RXx register.

A change in the configuration parameters is propagated directly on the SPI interface. If the spim_csx signal is activated, ensure that the configuration is changed only between SPI words to avoid corrupting the current transfer.

Note: To avoid data corruption, spim_csx polarity and spim_clk phase and spim_clk polarity must not be modified when the spim_csx signal is activated.

A delay between SPI words that requires the connected SPI slave device to switch from one configuration (transmit-only for instance) to another (receive-only for instance) must be handled by software.

At the end of the last SPI word, the channel must be deactivated (the SPIm.MCSPI_CHxCTRL[0] EN bit set to 0) and spim_csx can be forced to its inactive state using the SPIm.MCSPI_CHxCONF[20] FORCE bit.

Figure 19-15 and Figure 19-16 show successive transfers with spim_csx maintained active low with a different configuration for each SPI word in single-data-pin and dual-data-pin interface modes, respectively.

Figure 19-15. Continuous Transfers With spim_csx Maintained Active (Single-Data-Pin Interface Mode)

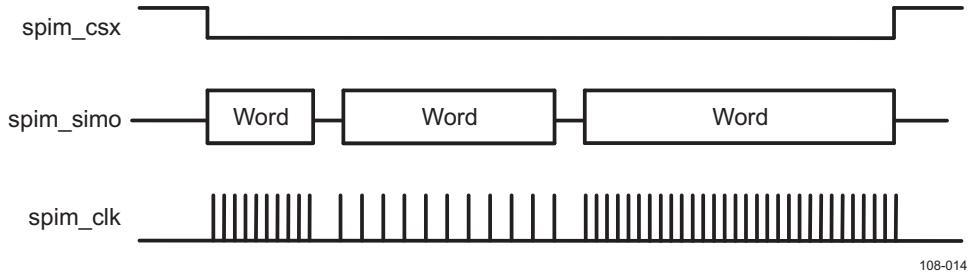
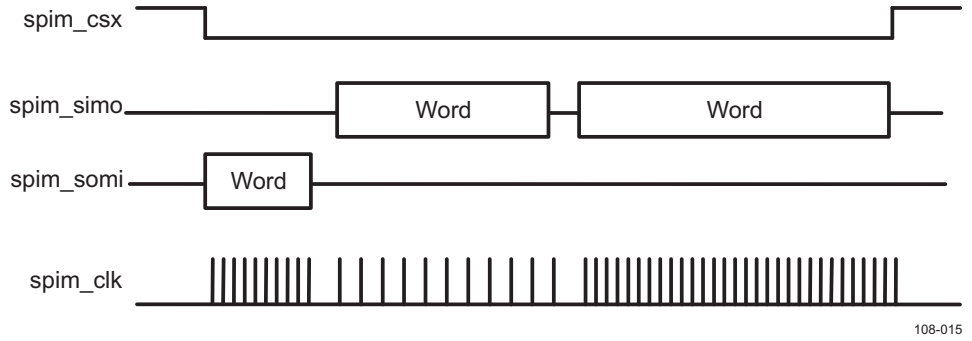


Figure 19-16. Continuous Transfers With spim_csx Maintained Active (Dual-Data-Pin Interface Mode)



Note: The turbo mode described in Section 19.5.2.5.3, *Turbo Mode*, maintains spim_csx in active mode when the following conditions are met:

- A single channel is explicitly used (the SPIm.MCSPI_MODULCTRL[0] SINGLE bit is set to 1).
- Turbo mode is enabled in the configuration of the channel. (The SPIm.MCSPI_CHxCONF[19] TURBO bit is set to 1.)

19.5.2.5.3 Turbo Mode

The turbo mode improves the throughput of the SPI interface when a single channel is enabled by allowing transfers until the shift register and the MCSPI_RXx register are full. The turbo mode is useful (time savings) when a transfer exceeds two words. This mode is programmable per channel (via the SPI1.MCSPI_CHxCONF[9] TURBO bit).

When several channels are enabled, the TURBO bit has no effect and the channel access to the shift registers remains as previously described.

In turbo mode, Rule 1 and Rule 2 applies, but Rule 3 does not (see Section 19.5.2.2). An enabled channel can be scheduled if its receive register is full (the SPIm.MCSPI_CHxSTAT[0]) RXS bit) at the time of the shift-register assignment until the shift register is full.

The MCSPI_RXx register cannot be overwritten in turbo mode. Consequently, the SPIm.MCSPI_IRQSTATUS[3] RX0_OVERFLOW bit is never set in this mode.

19.5.2.6 Start Bit Mode

In start bit mode, an extended bit is added before the SPI word in order to indicate whether the next SPI word must be handled as a command or as data. This feature is only available in master mode. The start bit mode cannot be used at the same time as turbo mode and/or force spim_csx mode. In this case, only one channel can be used; round-robin arbitration is not possible.

This mode is programmable per channel by setting the SPIm.MCSPI_CHxCONF[23] SBE bit to 1. The polarity of the extended bit is programmable per channel. When the SPIm.MCSPI_CHxCONF[24] SBPOL bit is set to 0, the SPI word must be handled as a command. When the SPIm.MCSPI_CHxCONF[24] SBPOL bit is set to 1, the SPI word must be handled as data. Moreover, start-bit polarity can be changed dynamically during start bit transfer without disabling the channel for reconfiguration; in this case, users must configure the SPIm.MCSPI_CHxCONF[24] SBPOL bit before writing the SPI word to be transmitted to the TX register.

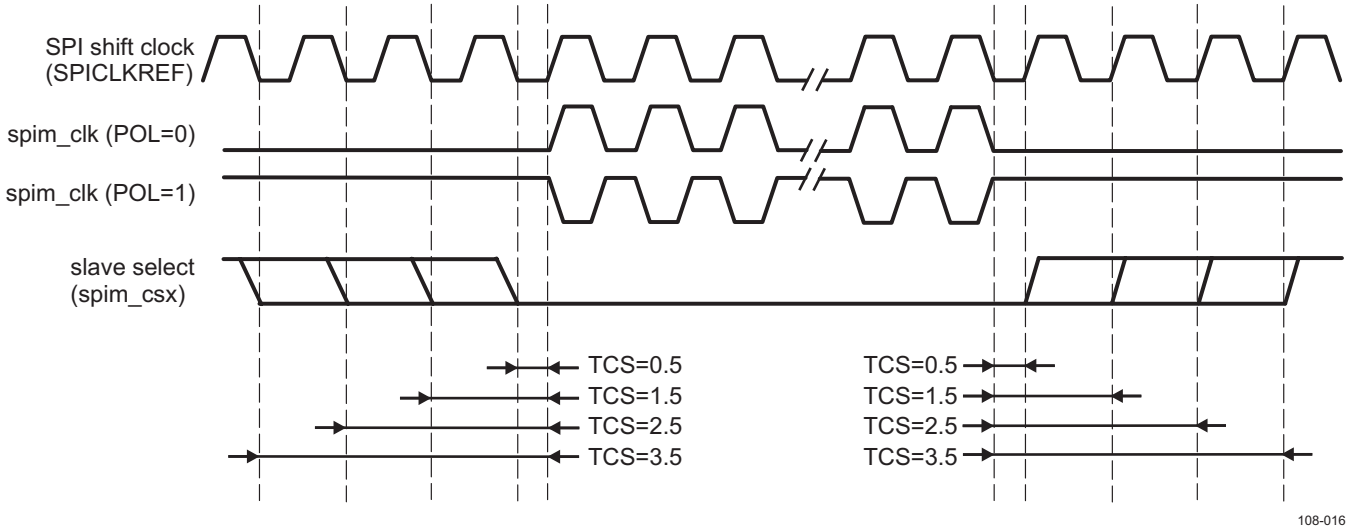
19.5.2.7 Chip-Select Timing Control

The chip select timing control is only available in master mode with automatic chip select generation (MCSPI_MODULCTRL[0] SINGLE bit field set to 0), to add a programmable delay between chip select assertion and first clock edge, or chip select removal and last clock edge.

This mode is programmable per channel (the TCS bit of the SPIm.MCSPI_CHxCONF register).

Figure 19-17 shows the chip-select SPIEN timing control.

Figure 19-17. Chip-Select SPIEN Timing Controls



Note: Because of the design implementation for transfers using a clock divider ratio set to 1 (clock bypassed), a half cycle must be added to the value between chip-select assertion and the first clock edge with PHA = 1 or between chip-select removal and the last clock edge with PHA = 0.

19.5.2.8 Programmable SPI Clock (spim_clk)

In master mode, the baud rate of the SPI serial clock is programmable.

An internal reference clock, SPIm_FCLK, is used as input of a programmable divider (the SPIm.MCSPI_CHxCONF[5:2] CLKD field) to generate the bit rate of the serial output clock spim_clk. Table 19-11 summarizes the supported divisor values.

Table 19-11. SPI Master Clock Rates

Divider	Clock Rate
1	48 MHz
2	24 MHz
4	12 MHz
8	6 MHz
16	3 MHz
32	1.5 MHz
64	750 kHz
128	375 kHz
256	~187 kHz
512	~93.7 kHz
1024	~46.8 kHz
2048	~23.4 kHz
4096	~11.7 kHz
8192 and higher: division not supported	-

19.5.2.8.1 Clock Ratio Granularity

By default the clock division ratio is defined by the SPIm.MCSPI_CHxCONF[5:2] CLKD bit field with power of two granularity leading to a clock division in range 1 to 4096, in this case the duty cycle is always 50%. With the SPIm.MCSPI_CHxCONF[29] CLKG bit, clock division granularity can be changed to one clock cycle, in that case the SPIm.MCSPI_CHxCTRL[15:8] EXTCLK bit field is concatenated with SPIm.MCSPI_CHxCONF[5:2] CLKD bit field to give a 12-bit width division ratio in range 1 to 4096.

When granularity is one clock cycle (CLKG set to 1), for odd value of clock ratio the clock high level lasts one clock cycle more than low level depending on SPIm.MCSPI_CHxCONF[1] POL bit and SPIm.MCSPI_CHxCONF[0] PHA bit refer to Table 19-12.

Table 19-12. CLKSPIO High/Low Time Computation

Clock ratio Fratio	CLKSPIO High Time	CLKSPIO Low Time
1	Thigh_ref	Tlow_ref
Even ≥ 2	T_ref * (Fratio/2)	T_ref * (Fratio/2)
Odd ≥ 3 (POL=PHA)	T_ref * (Fratio-1) /2	T_ref * (Fratio+1) /2
Odd ≥ 3 (POL≠PHA)	T_ref * (Fratio+1) /2	T_ref * (Fratio-1) /2

Note: Fratio = spi1_clk frequency (Fout) division ratio.
Thigh = spi1_clk High Time period.
Tlow = spi1_clk Low Time period.
T_ref = SPI1_FCLK period.
Thigh_ref = SPI1_FCLK high Time period.
Tlow_ref = SPI1_FCLK low Time period.

If CLKG = 1: Fratio = EXTCLK concatenated with CLKD + 1 as shown below:

Fratio - 1											
11	10	9	8	7	6	5	4	3	2	1	0
SPI1.[15:8] EXTCLK bit field								SPI1.[5:2] CLKD bit field			
15	14	13	12	11	10	9	8	5	4	3	2

For odd ratio values, the duty cycle is calculated as below:

Duty_cycle = (1-1/Fratio)/2

Table 19-13 shows clock granularity examples with a clock source frequency of 48 MHz.

Table 19-13. Clock Granularity Examples

EXTCLK	CLKD	CLKG	Fratio	PHA	POL	Thigh (ns)	Tlow (ns)	Tperiod (ns)	Duty Cycle	Fout (MHz)
X	0	0	1	X	X	10.4	10.4	20.8	50-50	48
X	1	0	2	X	X	20.8	20.8	41.6	50-50	24
X	2	0	4	X	X	41.6	41.6	83.2	50-50	12
X	3	0	8	X	X	83.2	83.2	166.4	50-50	6
0	0	1	1	X	X	10.4	10.4	20.8	50-50	48
0	1	1	2	X	X	20.8	20.8	41.6	50-50	24
0	2	1	3	1	0	41.6	20.8	62.4	66-33	16
0	2	1	3	1	1	20.8	41.6	62.4	33-66	16
0	3	1	4	X	X	41.6	41.6	83.2	50-50	12
5	0	1	81	1	0	852.8	832	1684.8	50.6-49.4	0.592
5	7	1	88	X	X	915.2	915.2	1830.4	50-50	0.545

19.5.3 Slave Mode

To select the McSPI slave mode, set the SPIm.MCSPI_MODULCTRL[2] MS bit.

A McSPI slave device can be connected to up to four external SPI master devices but handles transactions with one SPI master device at a time.

In slave mode, the McSPI initiates data transfer on the data lines (spim_simo and spim_somi) when it is selected by an active control signal (spim_csx) and receives an SPI clock (spim_clk) from the external SPI master device. Only channel 0 can be configured as a slave. In slave mode, the McSPI uses the edge of spim_csx to detect word length. For this reason, spim_csx must become inactive between each word.

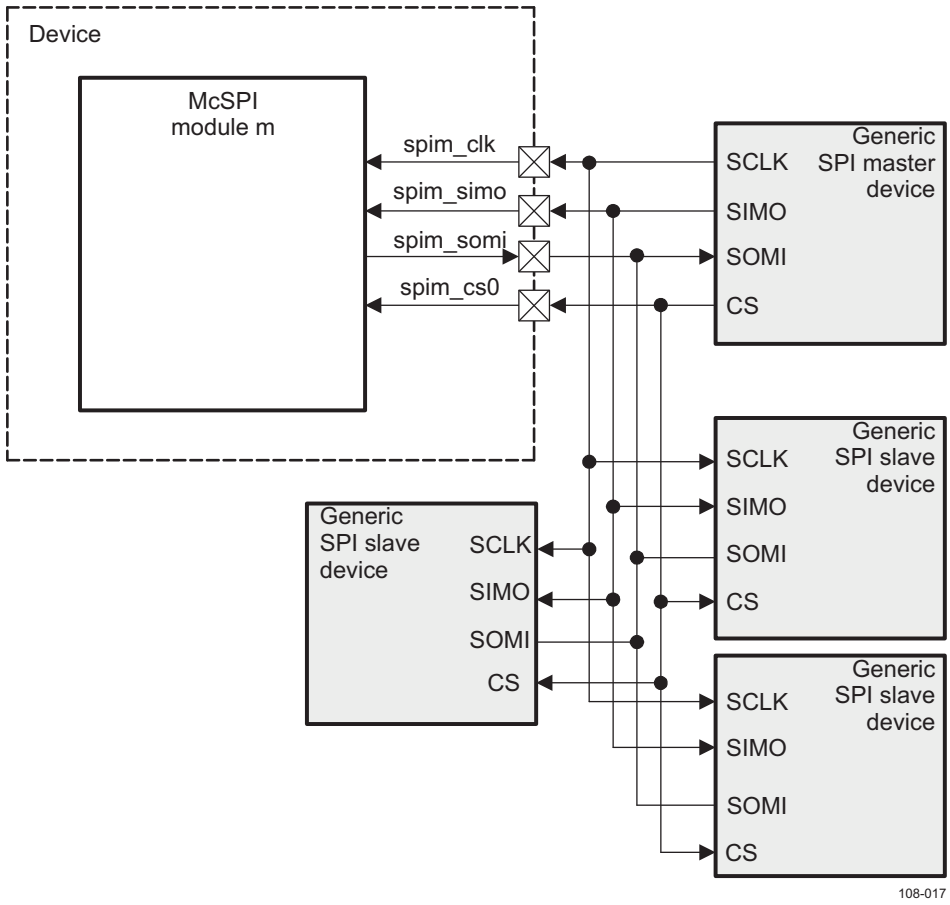
The McSPI does not support spim_csx active between SPI words. It uses the edge to detect word length.

19.5.3.1 Dedicated Resources

Only channel 0 can be enabled in slave mode. In this section registers name such as SPI1.MCSPI_CHxCTRL stand for SPI1.MCSPI_CH0CTRL where x=0 (channel 0 control register).

Figure 19-18 shows an example of four slaves wired on a single master device.

Figure 19-18. Example of McSPI Slave With One Master and Multiple Slave Devices on Channel 0



The channel 0 in slave mode has the following resources:

- Its own channel enable, programmable with the SPIm.MCSPI_CHxCTRL[0] EN bit (with x=0). This channel must be enabled before transmission and reception.
- For this mode, the slave-select signal can be detected only on spin_cs0.
- Its own transmitter register, SPIm.MCSPI_TXx (with x=0), on top of the common transmit shift register. If the MCSPI_TXx register is empty, the SPIm.MCSPI_CHxSTAT[1] TXS bit (with x=0) is set. If McSPI is selected by an external master (the active signal on the spim_csx port assigned to channel 0), the MCSPI_TXx register content of channel 0 is always loaded into the shift register, whether its content is updated or not. The MCSPI_TXx register must be loaded before McSPI is selected by a master.
- Its own receiver register, SPIm.MCSPI_RXx (with x=0), on top of the common receive shift register. If the MCSPI_RXx register is full, the SPIm.MCSPI_CHxSTAT[0] RXS bit (with x=0) is set.

Note: The MCSPI_TXx register and MCSPI_RXx registers of the other channels are not used. Reading from or writing to a channel register other than channel 0 has no effect.

- Its own communication configuration with the following parameters through the SPIm. MCSPI_CHxCONF register (with x=0):
  - Transmit and receive modes, programmable with the TRM field
  - Interface mode (two data pins or single data pin) and data pins assignment, both programmable with the IS and DPE bits. (The SPIm modules are in slave mode after reset and must be properly configured for the modules to act in master mode.)
  - SPI word length, programmable with the WL bit
  - spim_csx polarity, programmable with the EPOL bit
  - spim_clk polarity, programmable with the POL bit

– spim_clk phase, programmable with the PHA bit

The spim_clk frequency of a transfer is controlled by the external SPI master connected to the McSPI slave device. The SPIm.MCSPI_CHxCONF[5:2] CLKD field (with x=0) is not used in slave mode.

Note: The configuration of the channel can be loaded in the SPIm.MCSPI_CHxCONF register (with x=0) only when the channel is disabled.

- Two DMA request events, read and write, synchronize read/write accesses of the DMA controller with the activity of McSPI. DMA requests are asserted using the SPIm.MCSPI_CHxCONF[15] DMAR bit (with x=0) for reading and the SPIm.MCSPI_CHxCONF[14] DMAW bit (with x=0) for writing.
- Four interrupt events (see Section 19.5.5.2, *Interrupt Events in Slave Mode*)

19.5.3.2 Slave Transmit-and-Receive Mode

The slave receive mode is programmable (set the SPIm.MCSPI_CHxCONF[13:12] TRM field (with x=0) to 0x0).

In slave transmit-and-receive mode, the MCSPI_TXx register must be loaded before McSPI is selected by an external SPI master device.

After a channel is enabled, transmission and reception proceed with interrupt and DMA request events.

The MCSPI_TXx register content is always loaded in the shift register whether it is updated or not. The event TXx_UNDERFLOW is activated accordingly and does not prevent transmission.

When an SPI word transfer completes (the SPIm.MCSPI_CHxSTAT0[2] EOT bit (with x=0) set to 1), the received data is transferred to the channel receive register.

To use McSPI as a slave transmit-only device, the RXx_FULL and RX0_OVERFLOW interrupts and DMA read requests must be disabled due to the MCSPI_RXx register state (see Section 19.5.5.2, *Interrupt Events in Slave Mode*).

19.5.3.3 Slave Transmit-Only Mode

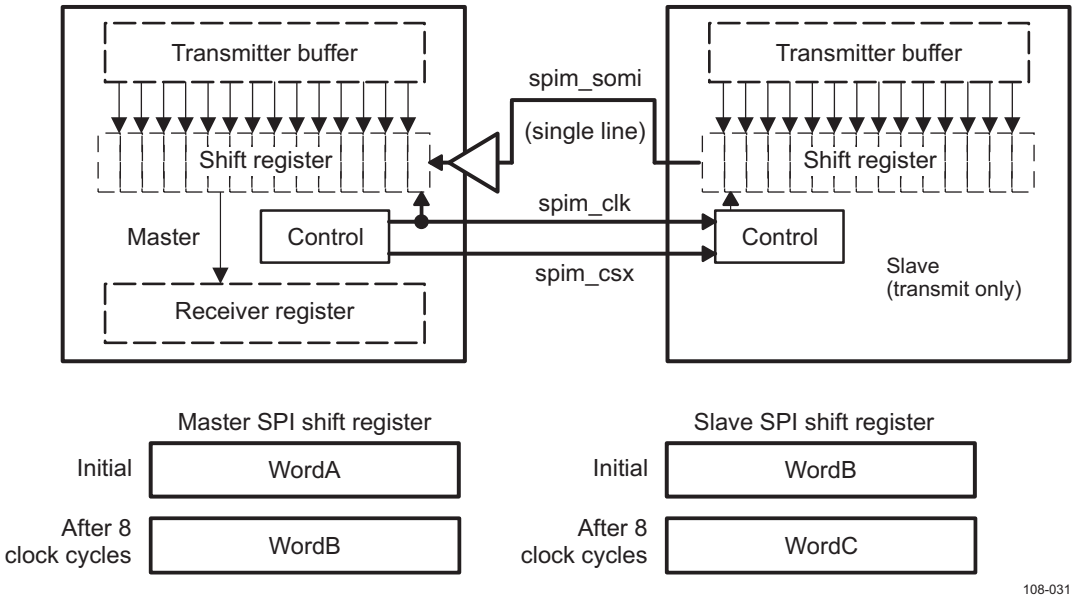
The slave transmit-only mode is programmable (set the SPIm.MCSPI_CHxCONF[13:12] TRM field (with x=0) to 0x2) and avoids the requirement for the MPU to read the MCSPI_RXx register (minimizing data movement) only when transmission is meaningful.

To use the McSPI as a slave transmit-only device, the RXx_FULL and RX0_OVERFLOW interrupts and DMA read requests must be disabled because of the MCSPI_RXx register state.

When the SPI word transfer completes, the SPIm.MCSPI_CHxSTAT[2] EOT bit is set (with x=0).

Figure 19-19 shows a half-duplex system with a master device on the left and a transmit-only slave device on the right. Each time a bit transfers out from the slave device, 1 bit transfers in the master. After eight cycles of the serial clock spim_clk, WordB transfers from the slave to the master.

Figure 19-19. SPI Half-Duplex Transmission (Transmit-Only Slave)



19.5.3.4 Slave Receive-Only Mode

The slave receive mode is programmable (set the SPIm.MCSPI_CHxCONF[13:12] TRM field (with x=0) to 0x1).

In receive-only mode, the MCSPI_TXx register must be loaded before the McSPI is selected by an external SPI master device. The MCSPI_TXx register content is always loaded into the shift register whether it is updated or not. The TXx_UNDERFLOW event is activated accordingly and does not prevent transmission.

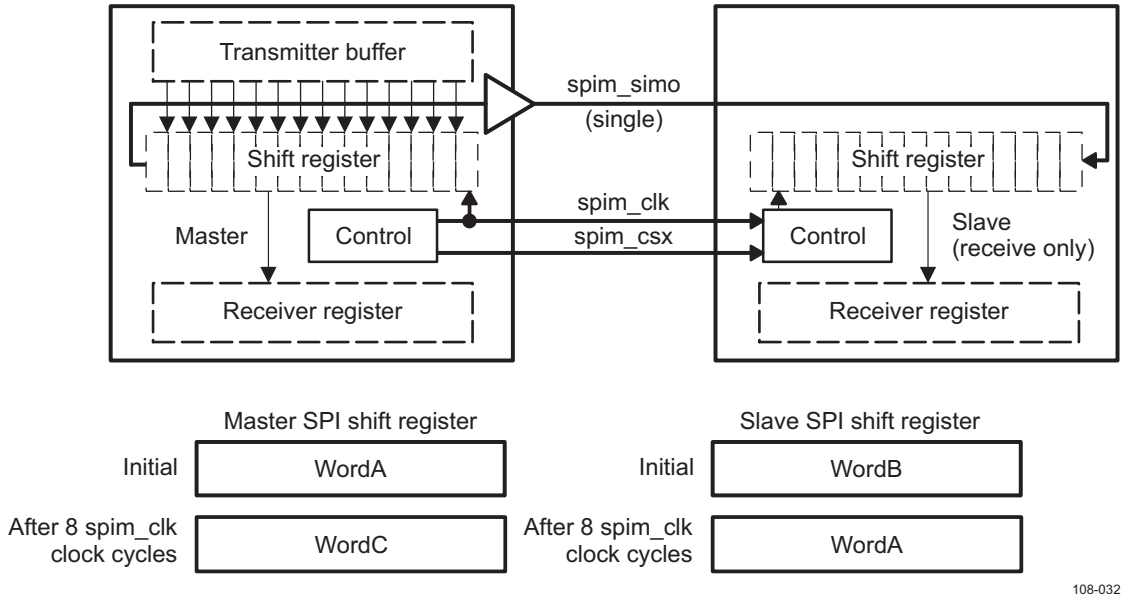
When an SPI word transfer completes (the SPIm.MCSPI_CHxSTAT0[2] EOT bit (with x=0) is set to 1), the received data is transferred to the channel receive register.

To use the McSPI as a slave receive-only device, the TXx_EMPTY and TXx_UNDERFLOW interrupts and the DMA write requests must be disabled due to the MCSPI_TXx register state.

For a full-duplex transmission, the serial clock (spim_clk) synchronizes shifting and sampling of the information on the single serial data line. For full duplex, two data lines are required. If spim_clk synchronizes on a single serial data line, the data line should be half-duplex.

Figure 19-20 shows an example of a half-duplex system with a master device on the left and a receive-only slave device on the right. Each time a bit transfers out from the master, 1 bit transfers in from the slave. After eight cycles of the serial clock spim_clk, Word A transfers from the master to the slave.

Figure 19-20. SPI Half-Duplex Transmission (Receive-Only Slave)



19.5.4 FIFO Buffer Management

The McSPI controller has a built-in 64 bytes buffer to unload DMA or interrupt handler and improve data throughput.

This buffer can be used by only one channel at once and is selected by setting SPIm.MCSPI_CHxCONF[28] FFER bit or SPIm.MCSPI_CHxCONF[27] FFEW bit to 1. If several channel are selected and several FIFO enable bit fields set to 1, the controller forces buffer not to be used, it is the responsibility of the driver to set only one FIFO enable bit field.

The buffer can be used in the modes defined below:

- Master or Slave mode.
- Transmit only, Receive only or Transmit/Receive mode.
- Single channel or turbo mode, or in normal round robin mode. In round robin mode the buffer is used by only one channel.
- Every word length SPIm.MCSPI_CHxCONF[11:7] WL are supported.

In transmit/receive mode, the buffer can be used in transmit (see Figure 19-21) or receive (see Figure 19-22) directions, or in both directions. If only one direction is chosen in transmit/receive mode, the full buffer is used for this direction. In both directions, the buffer is split into two 32-byte buffers, one for each direction. See Figure 19-23.

Figure 19-21. Buffer Use in Transmit Direction Only

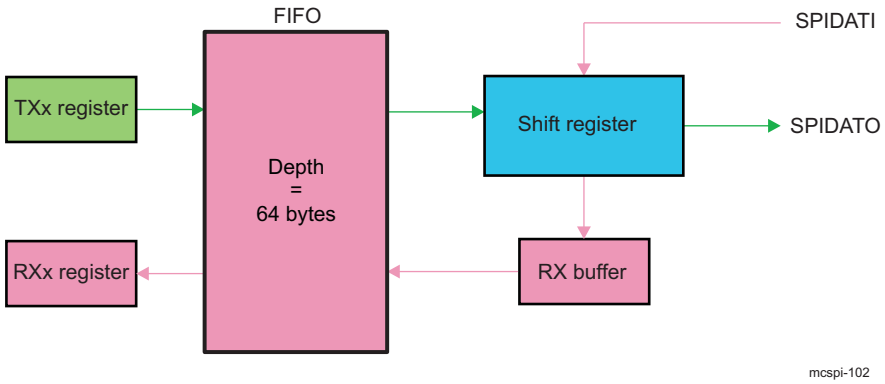


Figure 19-22. Buffer Use in Receive Direction Only

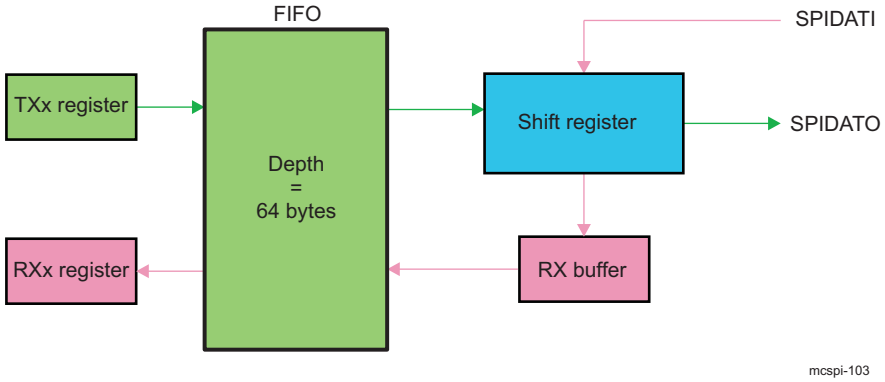
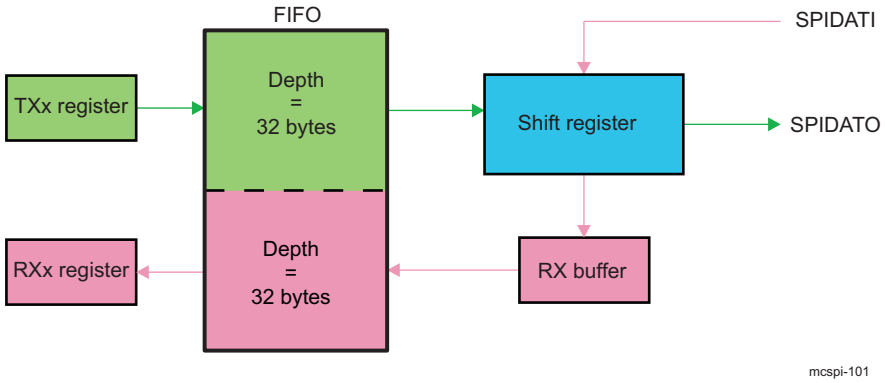


Figure 19-23. Buffer Used For Both Transmit/Receive Directions



Two levels SPIm.MCSPI_XFERLEVEL[5:0] AEL and SPIm.MCSPI_XFERLEVEL[13:8] AFL rule the buffer management. The granularity of these levels is one byte, then it is not aligned with SPI word length. It is the responsibility of the driver to set these values as a multiple of SPI word length defined in WL. Table 19-14 shows the number of byte written in the FIFO depending on the word length.

Table 19-14. FIFO Writes, Word Length Relationship

SPI Word Length WL	3 ≤ WL ≤ 7	8 ≤ WL ≤ 15	16 ≤ WL ≤ 31
Number of byte written in the FIFO	1 byte	2 bytes	4 bytes

The FIFO buffer pointers are reset when the corresponding channel is enabled or FIFO configuration changes.

19.5.4.1 Buffer Almost Full

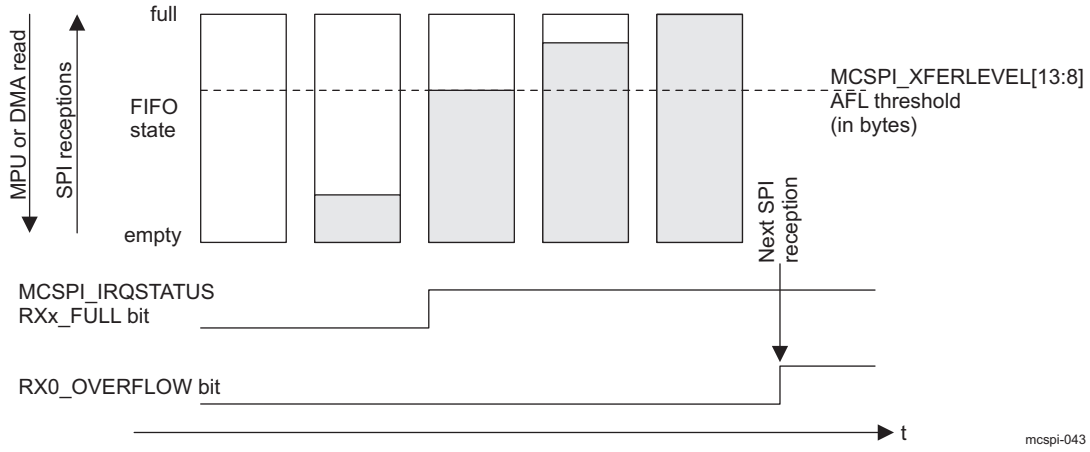
The MCSPI_XFERLEVEL[13:8] AFL bit field is needed when the buffer is used to receive SPI word from a slave (MCSPI_CHxCONF[28] FFER bit must be set to 1). It defines the Almost Full buffer status. See Figure 19-24.

When FIFO pointer reaches this level an interrupt or a DMA request is sent to the MPU to enable system to read AFL+1 bytes from Receive register. Be careful AFL+1 must correspond to a multiple value of MCSPI_CHxCONF[11:7] WL bit field.

When DMA is used, the request is de-asserted after the first Receive register read.

No new request will be asserted again as long as system has not performed the right number of read accesses.

Figure 19-24. Buffer Almost Full Level (AFL)



Note: MCSPI_IRQSTATUS register bits are not available in DMA mode. In DMA mode, the SPIm_DMA_RXx request is asserted on the same conditions than the MCSPI_IRQSTATUS RXx_FULL flag.

19.5.4.2 Buffer Almost Empty

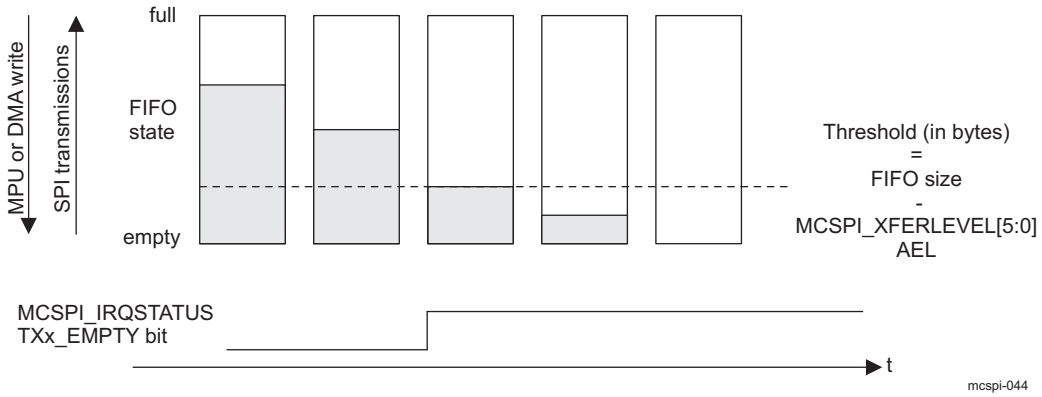
The MCSPI_XFERLEVEL[5:0] AEL bit field is needed when the buffer is used to transmit SPI word to a slave (MCSPI_CHxCONF[27] FFEW bit must be set to 1). It defines the Almost Empty buffer status. See Figure 19-25.

When FIFO pointer has not reached this level an interrupt or a DMA request is sent to the MPU to enable system to write AEL+1 bytes to Transmit register. Be careful AEL+1 must correspond to a multiple value of MCSPI_CHxCONF[11:7] WL bit field.

When DMA is used, the request is de-asserted after the first Transmit register write.

No new request will be asserted again as long as system has not performed the right number of write accesses.

Figure 19-25. Buffer Almost Empty Level (AEL)



Note: MCSPI_IRQSTATUS register bits are not available in DMA mode. In DMA mode, the SPIm_DMA_TXx request is asserted on the same conditions than the MCSPI_IRQSTATUS TXx_EMPTY flag.

19.5.4.3 End of Transfer Management

When the FIFO buffer is enabled for a channel, the user must previously configure in the MCSPI_XFERLEVEL register the AEL and AFL levels and especially the MCSPI_XFERLEVEL[31:16] WCNT bit field to define the number of SPI words to be transferred using the FIFO before enabling the channel.

This counter allows the controller to stop the transfer correctly after a defined number of SPI word transfers. If WNCT is set to 0x0000, the counter is not used and the user must stop the transfer manually by disabling the channel. In this case, the user does not know how many SPI transfers have been done. For received words, software must poll the CHxSTAT[5] RXFFE bit and read the MCSPI_RXx receive register to empty the FIFO buffer.

When the End Of Word count interrupt is generated (the MCSPI_IRQSTATUS[17] EOW bit is set), the user can disable the channel and poll the MCSPI_CHxSTAT[5] RXFFE bit to know the last SPI words in the FIFO buffer and read them.

No new request is asserted again as long as the system has not performed the correct number of write accesses.

Note: The RXFFE status bit shows only the FIFO status. The data received are stored not only in the FIFO, but also in the shift register and the MCSPI_RX register. Therefore, the FIFO can be empty even after receiving two words.

19.5.5 Interrupts

Each channel can issue interrupt events.

Each interrupt event has status bits in the SPIm.MCSPI_IRQSTATUS register (RXx_FULL, TXx_UNDERFLOW, TXx_EMPTY, ...) with x = [0,3] that indicate if service is required. Each status bit has an interrupt enable bit (a mask) in the SPIm.MCSPI_IRQENABLE register (RXx_FULL_ENABLE, TXx_UNDERFLOW_ENABLE, TXx_EMPTY_ENABLE, ...).

When an interrupt occurs and a mask is later applied on it, the interrupt line is not asserted again, even if the interrupt source is not serviced.

The McSPI supports interrupt-driven and polling operations.

19.5.5.1 Interrupt Events in Master Mode

In master mode, the interrupt events related to the MCSPI_TXx register state are TXx_EMPTY and TXx_UNDERFLOW. The interrupt event related to the MCSPI_RXx register state is RXx_FULL.

19.5.5.1.1 TXx_EMPTY

The TXx_EMPTY event is activated when a channel is enabled and its MCSPI_TXx register is empty (transient event). Enabling a channel automatically triggers this event, except in master receive-only mode (see Section 19.5.2.4, *Master Receive-Only Mode*). When the FIFO buffer is enabled (MCSPI_CHxCONF[27] FFEW bit set to 1), the MCSPI_IRQSTATUS TXx_EMPTY bit is set as soon as there is enough space in buffer to write a number of bytes defined by the MCSPI_XFERLEVEL[5:0] AEL bit field.

The MCSPI_TXx register must be loaded with data to remove the source of the interrupt; the SPIm.MCSPI_IRQSTATUS TXx_EMPTY interrupt status bit must be cleared for interrupt line deassertion (if the event is enabled as the interrupt source).

When FIFO is enabled, no new TXx_EMPTY event will be asserted as soon as the MPU has not performed the number of write into the MCSPI_TXx register defined by MCSPI_XFERLEVEL[5:0] AEL bit field. It is the responsibility of the MPU to perform the right number of writes.

19.5.5.1.2 TXx_UNDERFLOW

The event TXx_UNDERFLOW is activated when the channel is enabled and if the MCSPI_TXx register or if the FIFO is empty (not updated with new data) when an external master device starts a data transfer with the McSPI (transmit and receive).

The TXx_UNDERFLOW is a harmless warning in master mode.

To avoid having TXx_UNDERFLOW event at the beginning of a transmission, the event TXx_UNDERFLOW is not activated when no data has been loaded into the MCSPI_TXx register since channel has been enabled. To avoid having TXx_UNDERFLOW event, the MCSPI_TXx register must be loaded seldom.

The SPIm.MCSPI_IRQSTATUS TXx_UNDERFLOW interrupt status bit must be cleared for interrupt line de-assertion (if event enable as interrupt source).

19.5.5.1.3 RXx_ FULL

The RXx_FULL event is activated when channel is enabled and MCSPI_RXx register becomes filled (transient event). When FIFO buffer is enabled (MCSPI_CHxCONF[28] FFER bit set to 1), the RXx_ FULL is asserted as soon as the number of bytes holds in the FIFO to be read reaches the MCSPI_XFERLEVEL[13:8] AFL threshold.

The MCSPI_RXx register must be read to remove the source of the interrupt; the MCSPI_IRQSTATUS RXx_FULL interrupt status bit must be cleared for interrupt line deassertion (if the event is enabled as the interrupt source).

When FIFO is enabled, no new RXx_FULL event will be asserted as soon as the MPU has not performed AFL+1 reads into MCSPI_RXx. It is the responsibility of MPU to perform the right number of reads.

19.5.5.1.4 End Of Word Count

The MCSPI_IRQSTATUS[17] EOW event (End Of Word count) is activated when channel is enabled and configured to use the built-in FIFO. This interrupt is raised when the controller had performed the number of transfer defined in the MCSPI_XFERLEVEL[31:16] WCNT bit field. If WCNT is set to 0x0000, the counter is not enable and this interrupt is not generated.

The End of Word count interrupt also indicates that the SPI transfer is halt on channel using the FIFO buffer as soon as MCSPI_XFERLEVEL[31:16] WCNT is not reloaded and the channel is not re-enabled.

The MCSPI_IRQSTATUS[17] EOW interrupt status bit must be cleared for interrupt line de-assertion (if event enable as interrupt source).

19.5.5.2 Interrupt Events in Slave Mode

In slave mode, the interrupt events related to the MCSPI_TXx register state are TXx_EMPTY and TXx_UNDERFLOW. The interrupt events related to the MCSPI_RXx register state are RXx_FULL and RX0_OVERFLOW (channels 1, 2, and 3 do not have a receiver overflow status bit). See the MCSPI_IRQSTATUS register.

19.5.5.2.1 TXx_EMPTY

The TXx_EMPTY event is activated when a channel is enabled and its MCSPI_TXx register is empty. Enabling the channel automatically raises this event. If the FIFO buffer is enabled (MCSPI_CHxCONF[27] FFEW bit set to 1), the TXx_EMPTY event is asserted as soon as there is enough space in buffer to write a number of byte defined by the MCSPI_XFERLEVEL[5:0] AEL bit field.

The MCSPI_TXx register must be loaded with data to remove the source of the interrupt; the SPIm.MCSPI_IRQSTATUS TXx_EMPTY interrupt status bit must be cleared for interrupt line deassertion (if the event is enabled as the interrupt source).

When FIFO is enabled, no new TXx_EMPTY event will be asserted as soon as the MPU has not performed the number of write into the MCSPI_TXx register defined by MCSPI_XFERLEVEL[5:0] AEL bit field. It is the responsibility of the MPU to perform the right number of writes.

19.5.5.2.2 TXx_UNDERFLOW

The TXx_UNDERFLOW event is activated when a channel is enabled and if the MCSPI_TXx register is empty (not updated with new data) when an external master device starts a data transfer with the McSPI (transmit and receive).

When FIFO is enabled, the data emitted while underflow event is raised is not the last data written in the FIFO but the next data in the FIFO (an old transmitted value or a dummy data is the FIFO has been reset).

TXx_UNDERFLOW indicates an error (data loss) in slave mode.

To avoid having a TXx_UNDERFLOW event at the beginning of a transmission, the TXx_UNDERFLOW event is not activated when data is not loaded into the MCSPI_TXx register because the channel is enabled.

The SPIm.MCSPI_IRQSTATUS TXx_UNDERFLOW interrupt status bit must be cleared for interrupt line deassertion (if the event is enabled as the interrupt source).

19.5.5.2.3 RXx_FULL

The RXx_FULL event is activated when a channel is enabled and the MCSPI_RXx register is being filled (transient event). When FIFO buffer is enabled (MCSPI_CHxCONF[28] FFER bit set to 1), RXx_FULL is asserted as soon as there is a number of bytes holds in buffer to read defined by the MCSPI_XFERLEVEL[13:8] AFL bit field.

The MCSPI_RXx register must be read to remove the source of the interrupt; the SPIm.MCSPI_IRQSTATUS RXx_FULL interrupt status bit must be cleared for interrupt line deassertion (if the event is enabled as the interrupt source).

When FIFO is enabled, no new RXx_FULL event will be asserted as soon as the MPU has not performed AFL+1 reads into MCSPI_RXx. It is the responsibility of MPU to perform the right number of reads.

19.5.5.2.4 RX0_OVERFLOW

The RX0_OVERFLOW event is activated in slave mode in either transmit-and-receive or receive-only mode, when a channel is enabled and the MCSPI_RXx register or FIFO is full when a new SPI word is received. The MCSPI_RXx register is always overwritten with the new SPI word. If the FIFO is enabled data within the FIFO are overwritten, it must be considered as corrupted. The RX0_OVERFLOW event should not appear in slave mode using the FIFO.

The RX0_OVERFLOW indicates an error (data loss) in slave mode.

The MCSPI_IRQSTATUS[3] RX0_OVERFLOW interrupt status bit must be cleared for interrupt line deassertion (if the event is enabled as the interrupt source).

19.5.5.2.5 End Of Word Count

The MCSPI_IRQSTATUS[17] EOW event (End Of Word count) is activated when channel is enabled and configured to use the built-in FIFO. This interrupt is raised when the controller had performed the number of transfer defined in the MCSPI_XFERLEVEL[31:16] WCNT bit field. If WCNT is set to 0x0000, the counter is not enabled and this interrupt is not generated.

The End of Word count interrupt also indicates that the SPI transfer is halt on channel using the FIFO buffer as soon as WCNT is not reloaded and channel re-enabled.

The MCSPI_IRQSTATUS[17] EOW interrupt status bit must be cleared for interrupt line de-assertion (if event enable as interrupt source).

19.5.5.3 Interrupt-Driven Operation

An interrupt enable bit, in SPIm.MCSPI_IRQENABLE register, can be set to enable each event to generate interrupt requests when the corresponding event occurs. Status bits are automatically set by hardware logic conditions.

When an event occurs (the single interrupt line is asserted), the MPU must:

- Read the SPIm.MCSPI_IRQSTATUS register to identify which event occurred.
- Read the MCSPI_RXx register that corresponds to the event to remove the source of an RXx_FULL event or write into the MCSPI_TXx register that corresponds to the event to remove the source of a TXx_EMPTY event. No action is required to remove the source of the WKS (wake-up), TXx_ UNDERFLOW, and RX0_OVERFLOW events.
- Write 1 into the corresponding bit of the SPIm.MCSPI_IRQSTATUS register to clear an interrupt status and then release the interrupt line.

The interrupt status bit must always be reset after channel enabling and before events are enabled as interrupt sources.

19.5.5.4 Polling

When the interrupt capability of an event is disabled in the SPIm.MCSPI_IRQENABLE register, the interrupt line is not asserted, but the status bits in the SPIm.MCSPI_IRQSTATUS register can be polled by software to detect when the corresponding event occurs.

Once the expected event occurs:

- RXx_FULL: To remove the source of the event, the MPU must read the corresponding MCSPI_RXx register.
- TXx_EMPTY: To remove the source of the event, the MPU must write into the corresponding MCSPI_TXx register.
- WKS (wake-up), TXx_UNDERFLOW, and RX0_OVERFLOW: No action is required to remove the source of the event.

To clear an interrupt, set the corresponding status bit of the SPIm.MCSPI_IRQSTATUS register to 1. This does not affect the interrupt line state.

19.5.6 DMA Requests

The sDMA controller module manages DMA accesses. The sDMA controller advantage is to lower the MPU charge for data transfers.

Each McSPI channel can issue DMA requests if they are enabled. There are two DMA request lines per McSPI channel (one for read and one for write).

The DMA read request line is asserted when the McSPI channel is enabled and new data is available in the receive register of the McSPI channel. A DMA read request can be individually masked with the SPI1.MCSPI_CHxCONF[15] DMAR bit. The DMA read request line is deasserted on read completion of the MCSPI_RXx register of the McSPI channel.

The DMA write request line is asserted when the McSPI channel is enabled and the MCSPI_TXx register of the McSPI channel is empty. A DMA write request can be individually masked with the SPI1.MCSPI_CHxCONF[14] DMAW bit. The DMA write request line is deasserted on load completion of the MCSPI_TXx register of the channel.

19.5.7 Power Saving Management

Power consumption can be optimized by switching off internal clocks (interface and functional clock) when there is no activity. The McSPI is compliant with the idle and wake-up system handshake protocol.

19.5.7.1 Normal Mode

In normal mode, internal SPI module clocks are automatically switched off (autogating) when there is no activity in slave or master mode.

Autogating of the module interface clock and functional clock occurs when the following conditions are met:

- The SPIm.MCSPI_SYSCONFIG[0] AUTOIDLE bit is set.
- In master mode, there is no data to transmit or receive in all channels.

- In slave mode, the McSPI is not selected by the external master and there are no register accesses.

Autogating of the module interface clock and functional clock stops when the following conditions are met:

- In master mode, an internal access occurs.
- In slave mode, an internal access occurs or the McSPI is selected by the external master.

19.5.7.2 Idle Mode

At the PRCM module level, when all conditions are met to shut off the CORE_48M_FCLK or CORE_L4_ICLK output clocks (see Chapter 4, *Power, Reset, and Clock Management*, for details), the PRCM module automatically launches a hardware handshake protocol to ensure that the McSPI is ready to have its clocks switched off. Namely, the PRCM module asserts an idle request to the McSPI.

Although this handshake is completely hardware-oriented and out of software control, the method in which the McSPI module acknowledges the PRCM idle request is configurable through the McSPI.SYSCONFIG[4:3] SIDLEMODE bit.

The following list details the settings of the SIDLEMODE bit and the related acknowledgment modes:

- Force-idle mode (the SPIm.MCSPI_SYSCONFIG[4:3] SIDLEMODE bit set to 0x0): the McSPI module acknowledges unconditionally the idle request from the PRCM module, regardless of its internal operations. This mode must be used carefully in this case because it does not prevent the loss of data when the clock is switched off.
- No-idle mode the (SIDLEMODE bit set to 0x1): The McSPI module never acknowledges an idle request from the PRCM module and is safe from a module point of view because it ensures that the clocks remain active. However, it is not efficient to save power because it does not allow the PRCM output clock to be shut off and thus the power domain to be set to a lower power state.
- Smart-idle mode (the SIDLEMODE bit set to 0x2): The McSPI module acknowledges the idle request, basing its decision on its internal activity. Namely, the acknowledge signal is asserted only when all pending transactions, IRQs, or DMA requests are treated. This is the best approach for an efficient system power management.

When configured in smart-idle mode, the McSPI module also offers an additional granularity on the CORE_48M_FCLK and CORE_L4_ICLK gating. The SPIm.SYSCONFIG[9:8] CLOCKACTIVITY bit field determines which clock shuts down (the CORE_48M_FCLK, the CORE_L4_ICLK, neither clock, or both clocks).

The CLOCKACTIVITY setting is used internally to McSPI to determine on which part of the module the conditions to acknowledge the PRCM idle request are tested. For example, if CORE_48M_FCLK is not shut down on a PRCM idle request, McSPI considers only CORE_L4_ICLK and the associated pending activities before acknowledging the request.

Some McSPI features are associated with CORE_L4_ICLK and others with CORE_48M_FCLK. Using the CLOCKACTIVITY bit field along with the smart-idle mode ensures that the features associated with the clock that remains active are always enabled, even if McSPI acknowledges an idle request.

The following list details CLOCKACTIVITY settings and the associated features:

- CLOCKACTIVITY set to 00: ICLK OFF and FCLK OFF, both ICLK and FCLK are taken into account for generating the acknowledge. This setting also means that both FCLK and ICLK are likely to be shut down on a PRCM idle request.
- CLOCKACTIVITY set to 01: ICLK ON and FCLK OFF, ICLK is not shut down on a PRCM idle request; only FCLK is concerned.
- CLOCKACTIVITY set to 10: ICLK OFF and FCLK ON, FCLK is not shut down on a PRCM idle request; only ICLK is concerned.
- CLOCKACTIVITY set to 11: ICLK ON and FCLK ON, none of the clocks are shut down. This means McSPI can potentially acknowledge the idle request without checking the internal functionalities linked to its clocks.

CAUTION

The PRCM module does not have a hardware means of reading the CLOCKACTIVITY settings. Therefore, the software must ensure consistent programming between the CLOCKACTIVITY and the PRCM CORE_48M_FCLK and CORE_L4_ICLK control bits. If the McSPI is disabled in both the CM_FCLKEN and CM_ICLKEN PRCM registers while CLOCKACTIVITY is set to 11, nothing prevents the PRCM module from asserting its idle request, which is acknowledged regardless of the features associated with the McSPI clocks. This can lead to unpredictable behavior.

19.5.7.2.1 Wake-Up Event in Smart-Idle Mode

The module wake-up feature is enabled when both the SPIm.MCSPI_SYSCONFIG[2] ENAWAKEUP and SPIm.MCSPI_WAKEUPENABLE[0] WKEN bits are set. Wake-up capability is relevant only when the module is configured in slave mode.

The module generates an asynchronous wake-up request to the system power manager to switch the interface clock and the functional clock back. A wake-up is requested when channel 0 is enabled and an asynchronous selection occurs on the spim.csx port associated with channel 0 (see the definition for the SPIm.MCSPI_CHxCONF SPIENSLV field (with x=0) in the register description table).

After the McSPI wake-up request, the system power manager must reactivate the interface clock:

- *Before the beginning* of the second SPI word serialization when McSPI is in slave transmit-only mode or in slave transmit-and-receive mode
- *Before the end* of the second received SPI word in slave receive-only mode. To avoid data loss, the first received SPI word must be read from the SPIm.MCSPI_RXx register (with x=0) before the completion of the second SPI word serialization.

Table 19-15 lists the supported cases in wake-up mode.

Table 19-15. Smart-Idle Mode and Wake-Up Capabilities

Mode	Interface Clock	SPI Clock Ref	Functionality	Wake-Up Event
Master	Must be maintained	Must be maintained	Full functionality, but the module does not generate a new interrupt or DMA request until the system exits wake-up mode.	No wake-up event
Slave	Can be switched off	Can be switched off	An SPI word can be transmitted and/or received, but the module does not generate any new interrupts or DMA requests until the system exits wake-up mode.	The module asynchronously sends a wake-up request if an event on the spim_csx port associated to channel 0 is detected.

In wake-up mode, the interrupt and DMA request lines are no longer asserted.

Any access to the module in wake-up mode generates an error as long as the interface clock is alive.

19.5.7.2.2 Transitions From Smart-Idle Mode to Normal Mode

The McSPI detects the end of the wake period through the idle and wake-up hardware handshake protocol.

The interrupt status register (the SPIm.MCSPI_IRQSTATUS[16] WKS bit) is updated with the event causing the wake-up; the wake-up event at the origin of the transition to the normal mode is converted to its corresponding interrupt when enabled by the SPIm.MCSPI_IRQENABLE[16] WKE bit or the DMA request.

Interrupts and wake-up events have independent enable/disable controls, accessible through the SPIm.MCSPI_IRQENABLE and SPIm.MCSPI_WAKEUPENABLE registers. Software must ensure the overall consistency.

The interrupt status register SPIm.MCSPI_IRQSTATUS is updated with the event causing the wake-up; the wake-up event at the origin of the transition to normal mode is converted to its corresponding interrupt request or DMA request. The module is fully operational.

19.5.7.2.3 Force-Idle Mode

Force-idle mode is enabled and exited as follows:

- Force-idle mode is enabled when the SPIm.MCSPI_SYSCONFIG[4:3] SIDLEMODE field is set to 0x0.

  In force-idle mode, McSPI responds unconditionally to the idle request by deasserting unconditionally the interrupt and DMA request lines, if asserted. In addition, the wake-up capability is totally inhibited even if both the SPIm.MCSPI_SYSCONFIG[2] ENAWAKEUP and SPIm.MCSPI_WAKEUPENABLE[0] WKEN bits are set.

  The transition from normal mode to idle mode does not affect the interrupt event bits of the SPIm.MCSPI_IRQSTATUS register.

  In force-idle mode, the module must be disabled so the interrupt and DMA request lines are likely deasserted. The interface clock and SPI clock provided to the McSPI can be switched off.

  An idle request during an SPI data transfer can lead to an unexpected and unpredictable result. The software must avoid such a request.
- The module exits force-idle mode through the idle and wake-up hardware handshake protocol.

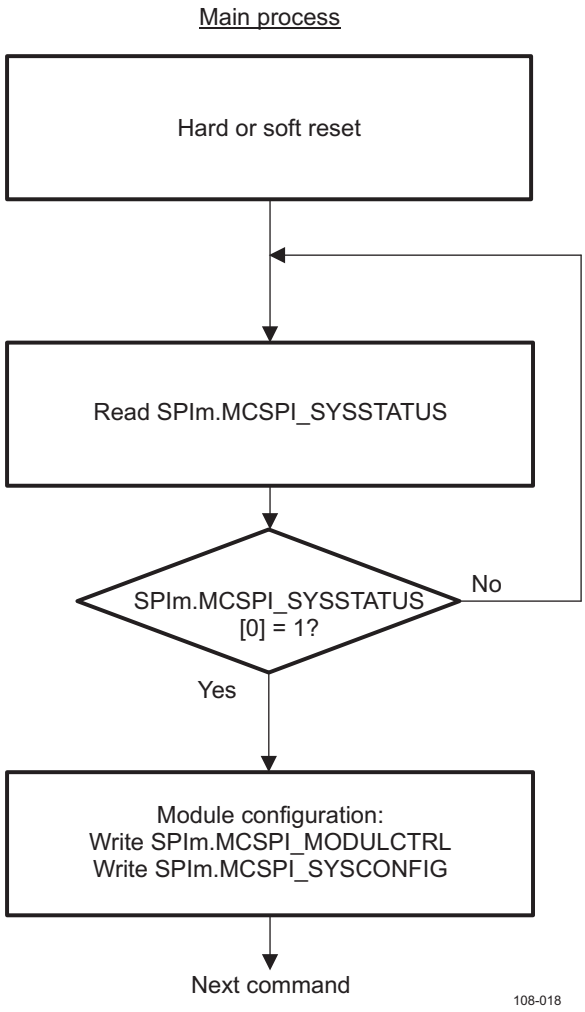
  The module is fully operational. The interrupt and DMA request lines are optionally asserted one clock cycle later.

19.6 McSPI Basic Programming Model

19.6.1 Initialization of Modules

Figure 19-26 shows the overview of the module initialization flow process. Section 19.6.2 and Section 19.6.3 show the steps required to configure McSPI modes.

Figure 19-26. Module Initialization Flow



Note: Before the SPIm.MCSPI_SYSSTATUS[0] RESETDONE bit is set, the CLK and CLKSPIREF clocks must be provided to the module.

To avoid unpredictable behavior, reset the module before changing from master mode to slave mode, or vice versa.

19.6.2 Transfer Procedures without FIFO

In the subsections below, the transfer procedures are described without FIFO using (MCSPI_CHxCONF[27:28] FFER and FFEW = 0).

The McSPI allows the transfer of one or more words based on the different modes:

- Master normal, master turbo, slave
- Transmit-and-receive, transmit-only, receive-only
- Write and read requests: Interrupts, DMA

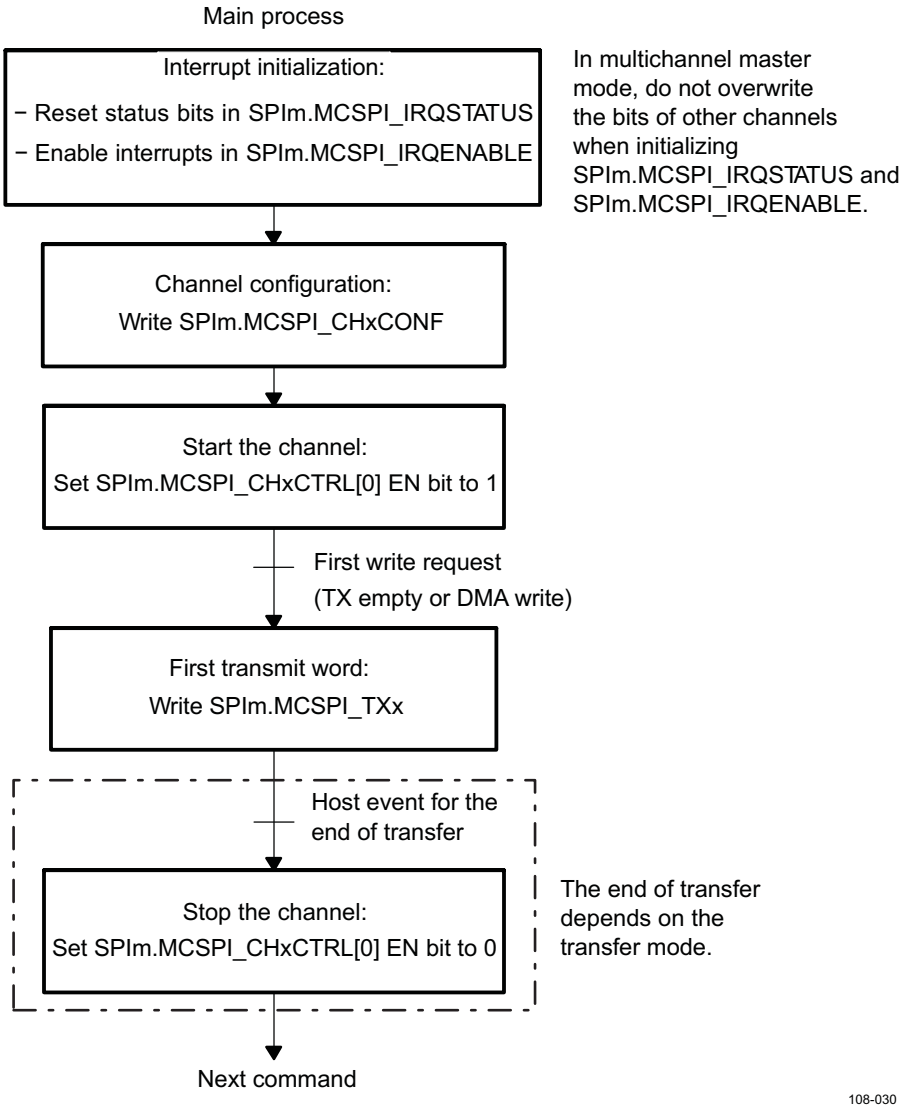
- spi1_csx line assertion/deassertion: Automatic, manual

For these flows, the host process contains the main process and the interrupt routines. The interrupt routines are called on the interrupt signals or by an internal call if the module is used in polling mode.

19.6.2.1 Common Transfer Procedure

Figure 19-27 shows the main sequence common to all transfers. In multichannel master mode, the flows of different channels can be run simultaneously.

Figure 19-27. Common Transfer Sequence: Main Process



19.6.2.2 End-of-Transfer Procedure

For transfers carried out with DMA or interrupt mode, the end of transfer must be achieved by following the steps shown in the flowcharts corresponding to Table 19-16, which summarizes the end-of-transfer types per transfer mode and provides cross-references for further information.

Table 19-16. End-of-Transfer Sequences

		Transmit Receive		Transmit Only		Receive Only	
		Interrupt	DMA	Interrupt	DMA	Interrupt	DMA
Master normal	End of transfer sequence	See Section 19.6.2.3		See Section 19.6.2.4	See Section 19.6.2.4	See Section 19.6.2.5.1	See Section 19.6.2.5.1
	Minimum number of words	1	1	1	1	1	1
	DMA transfer size[1]		w		w		w – 1
Master turbo	End of transfer sequence	See Section 19.6.2.3		See Section 19.6.2.4	See Section 19.6.2.4	See Section 19.6.2.5.2	See Section 19.6.2.5.2
	Minimum number of words	1	1	1	1	2	3
	DMA transfer size[1]		w		w		w – 2
Slave	End of transfer sequence	See Section 19.6.2.3		See Section 19.6.2.4	See Section 19.6.2.4	See Section 19.6.2.5.3	
	Minimum number of words	1	1	1	1	1	1
	DMA transfer size[1]		w		w	w	w

(1) w = number of words to transfer

The different sequences can be merged in one process to manage transfers of several types. The end-of-transfer sequences are described from the start of the channel.

In these sequences and in later sections of this chapter, some software variables are used:

- WRITE_COUNT (= 0 at initialization): Contains the number of words to transfer
- READ_COUNT (= 0 at initialization): Contains the number of words to receive
- CHANNEL_ENABLE (= false at initialization)
- LAST_TRANSFER (= false at initialization): Indicates that the last word is in transmission
- LAST_REQUEST (= false at initialization): Indicates that the last request is in progress

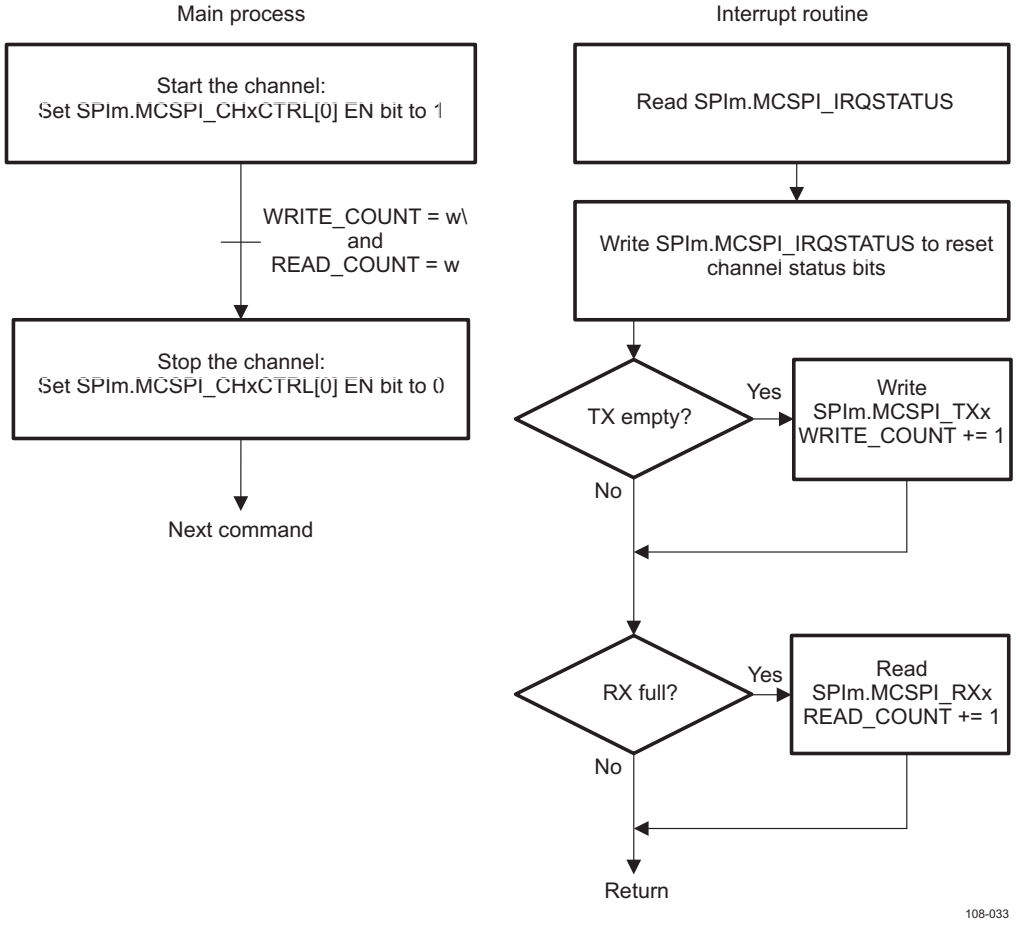
All variables are initialized before starting the channel.

19.6.2.3 Transmit and Receive Procedure

Figure 19-28 shows the handling procedure for words received and transmitted by interrupt in master and slave modes. The main process flow shows how the end of the transfer must be done after all words are received for this mode.

If the requests are configured in DMA, WRITE_COUNT and READ_COUNT are assigned with the value w when the DMA handler completes w interface accesses.

Figure 19-28. Transmit and Receive (Master and Slave)

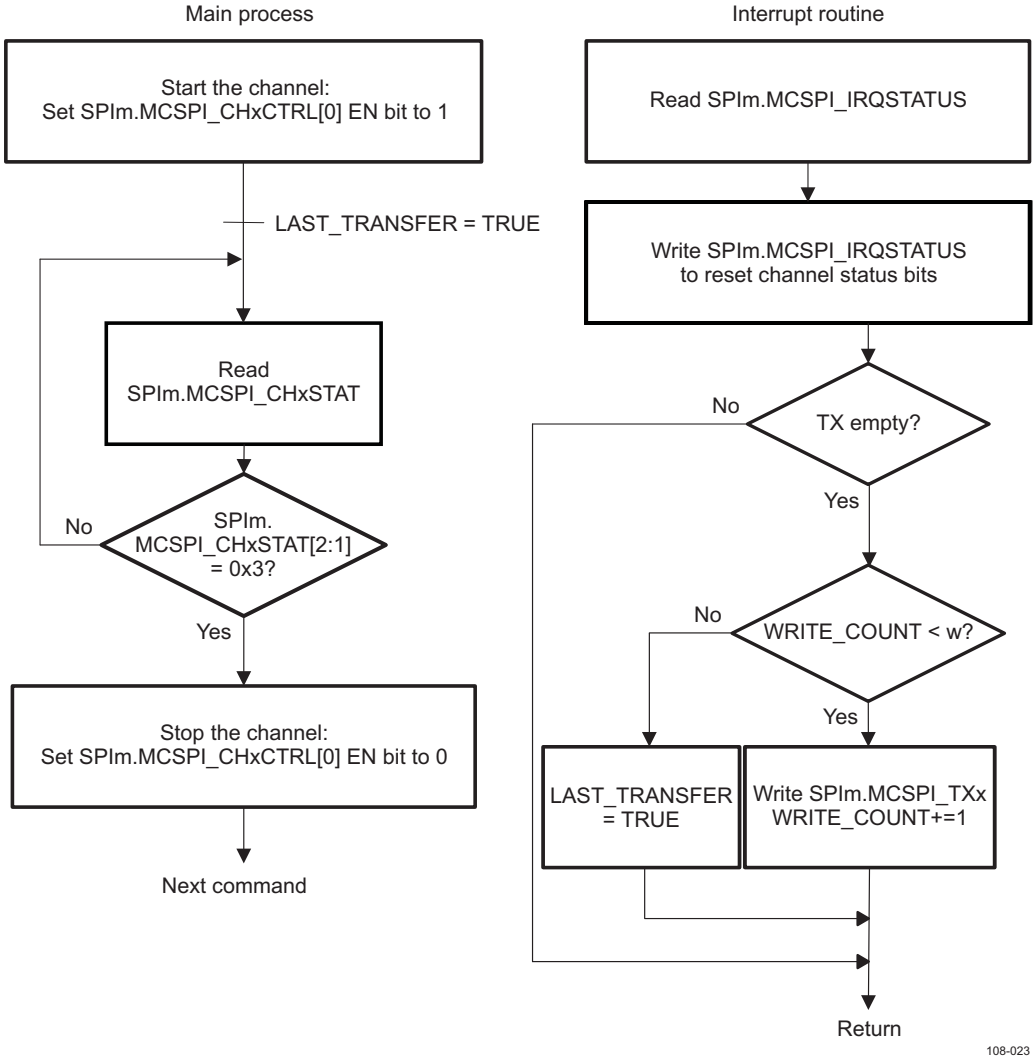


19.6.2.4 Transmit-Only Procedure

19.6.2.4.1 Based on Interrupt Requests

Figure 19-29 shows the handling procedure for words transmitted by interrupt in transmit-only mode. The main process flow shows how the end-of-transfer must be done after all words are received for this mode.

Figure 19-29. Transmit-Only With Interrupts (Master and Slave)

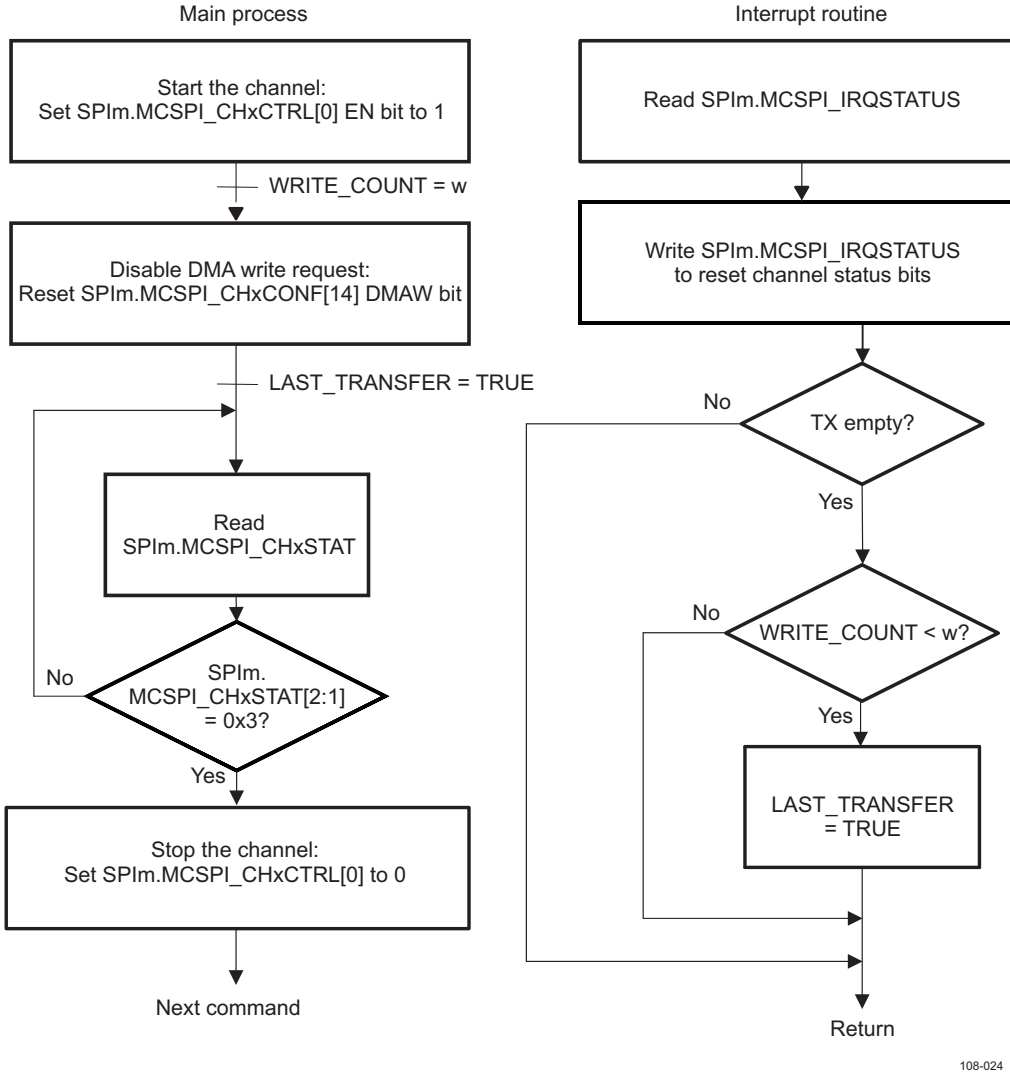


19.6.2.4.2 Transmit-Only Based on DMA Write Requests

In Figure 19-30, the main process shows completion of the transfer of words in transmit-only mode with DMA write requests.

When the DMA handler completes w interface accesses, WRITE_COUNT is assigned the value *w*.

Figure 19-30. Transmit-Only With DMA (Master and Slave)



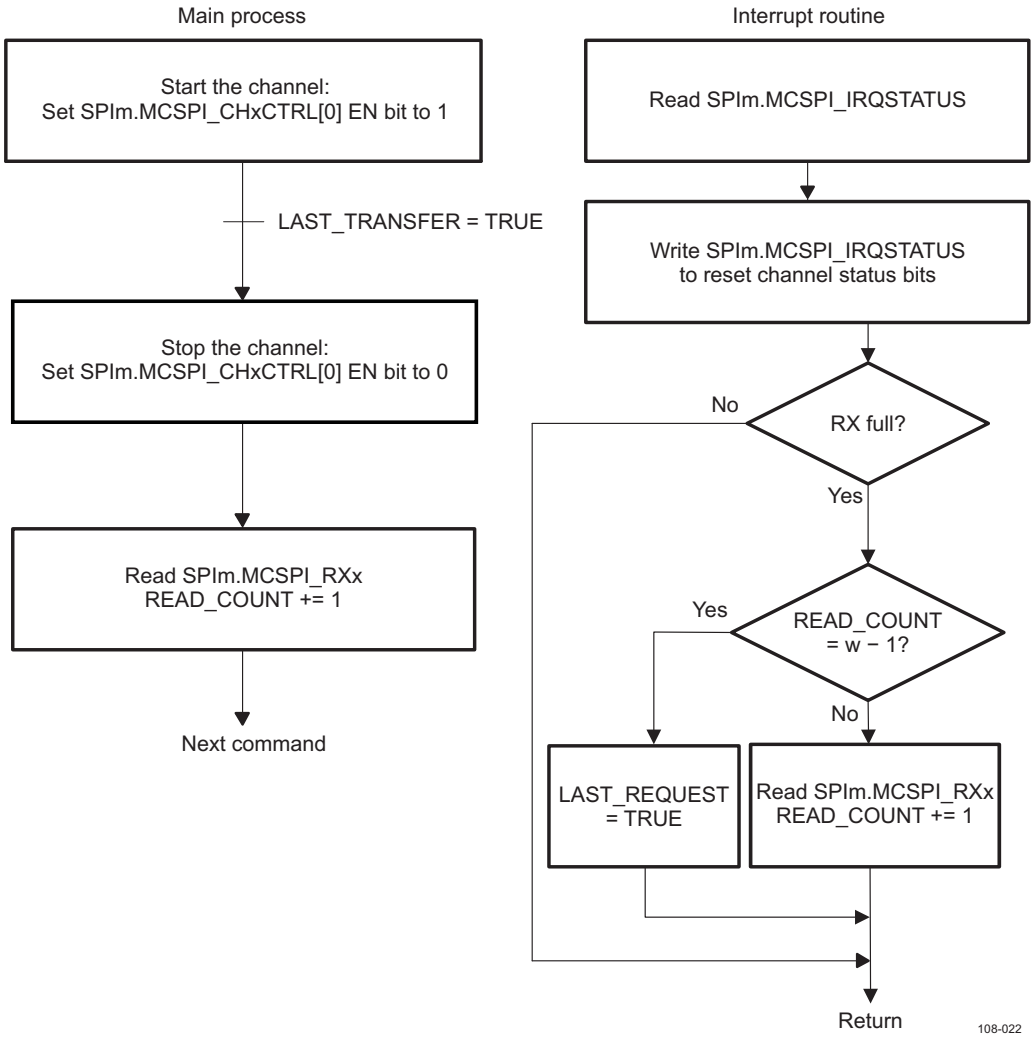
19.6.2.5 Receive-Only Procedure

19.6.2.5.1 Master Normal Receive-Only Procedure

19.6.2.5.1.1 Based on Interrupt Requests

Figure 19-31 shows the handling procedure for words received by interrupt in master normal receive-only mode. The main process flow shows how the end-of-transfer must be done after all words are received for this mode.

Figure 19-31. Receive Only With Interrupt (Master Normal)

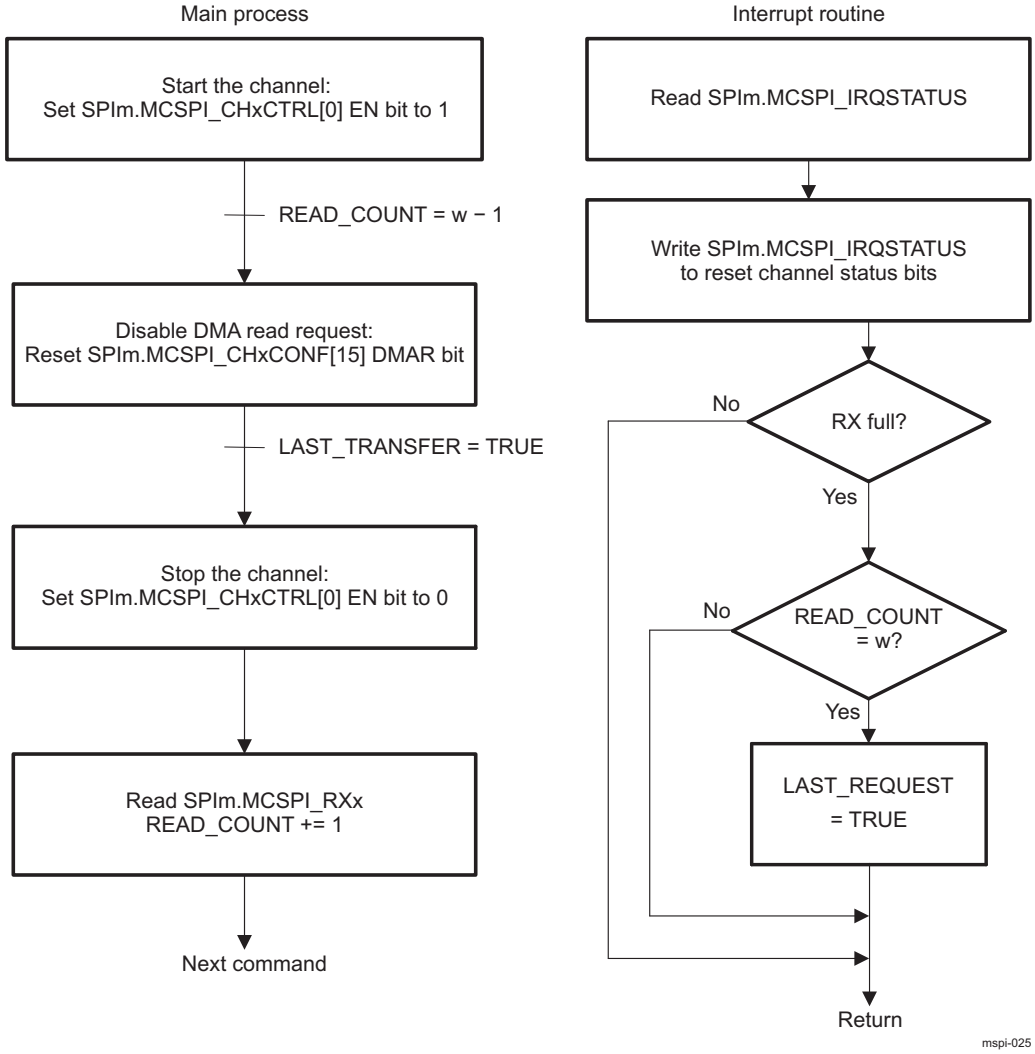


19.6.2.5.1.2 Receive-Only Based on DMA Read Requests

In Figure 19-32, the main process shows the completion of a word transfer in receive-only mode with DMA read requests.

When the DMA handler completes *w* - 1 interface accesses, READ_COUNT is assigned the value *w* - 1.

Figure 19-32. Receive-Only With DMA (Master Normal)

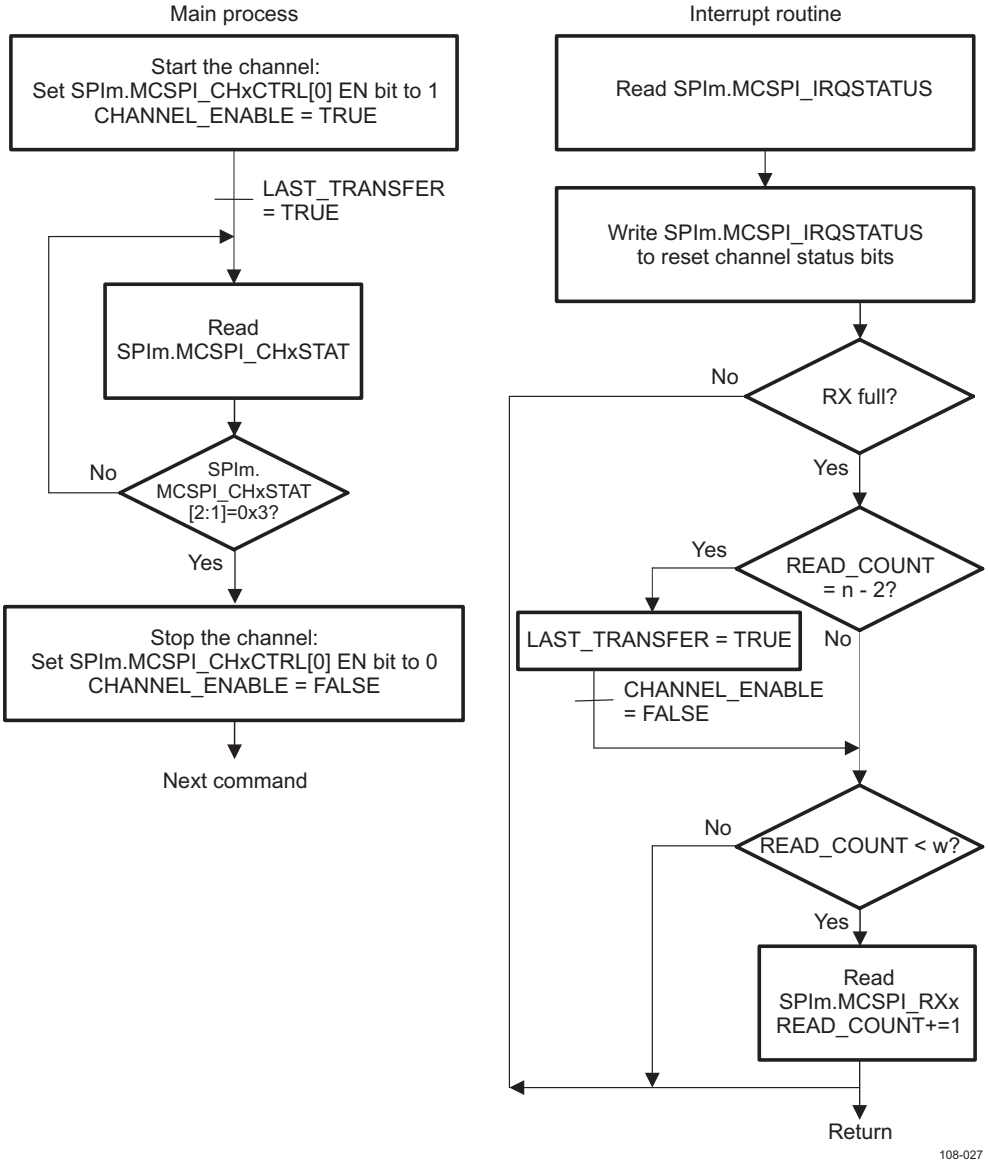


19.6.2.5.2 Master Turbo Receive-Only Procedure

19.6.2.5.2.1 Based on Interrupt Requests

Figure 19-33 shows the handling procedure for words received by interrupt in master turbo receive-only mode. The main process shows how the end-of-transfer must be done after all words are received for this mode.

Figure 19-33. Receive-Only With Interrupt (Master Turbo)

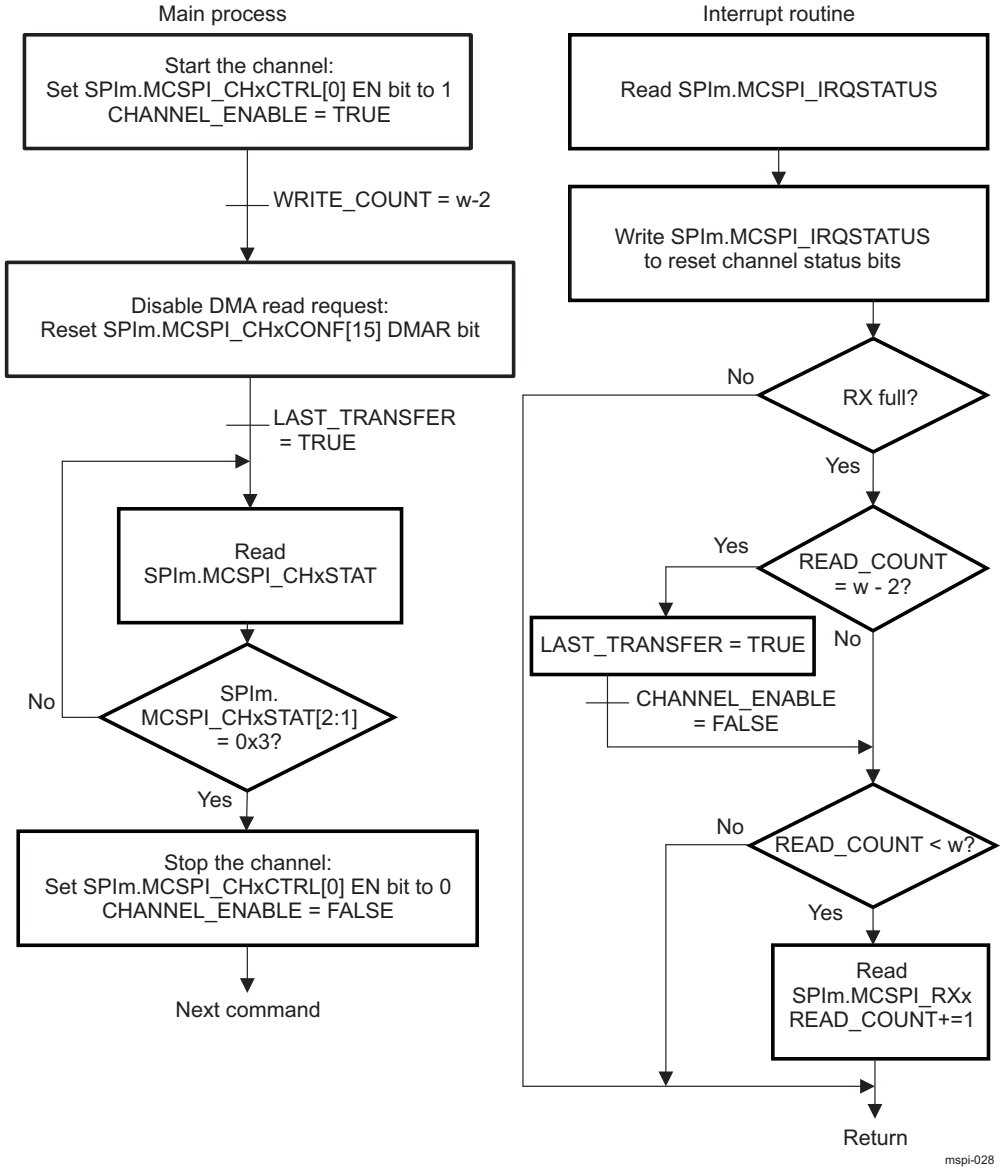


19.6.2.5.2.2 Based on DMA Read Requests

In Figure 19-34, the main process shows the completion of a word reception in master turbo receive-only mode with DMA write requests.

When the DMA handler completes *w* - 2 interface accesses, READ_COUNT is assigned the value *w* - 2.

Figure 19-34. Receive-Only With DMA (Master Turbo)

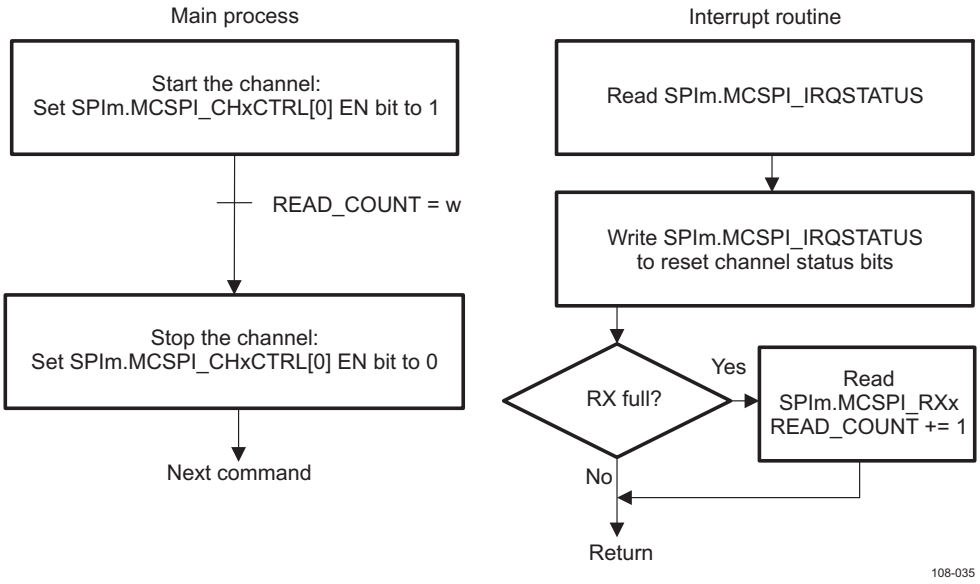


19.6.2.5.3 Slave Receive-Only Procedure

Figure 19-35 shows the handling procedure for words received by interrupt in slave receive-only mode. The main process shows how the end-of-transfer must be done after all words are received for this mode.

If the requests are configured in DMA, READ_COUNT is assigned the value *w* when the DMA handler completes *w* interface processes.

Figure 19-35. Receive Only (Slave)



19.6.2.6 McSPI Configuration and Operations Example

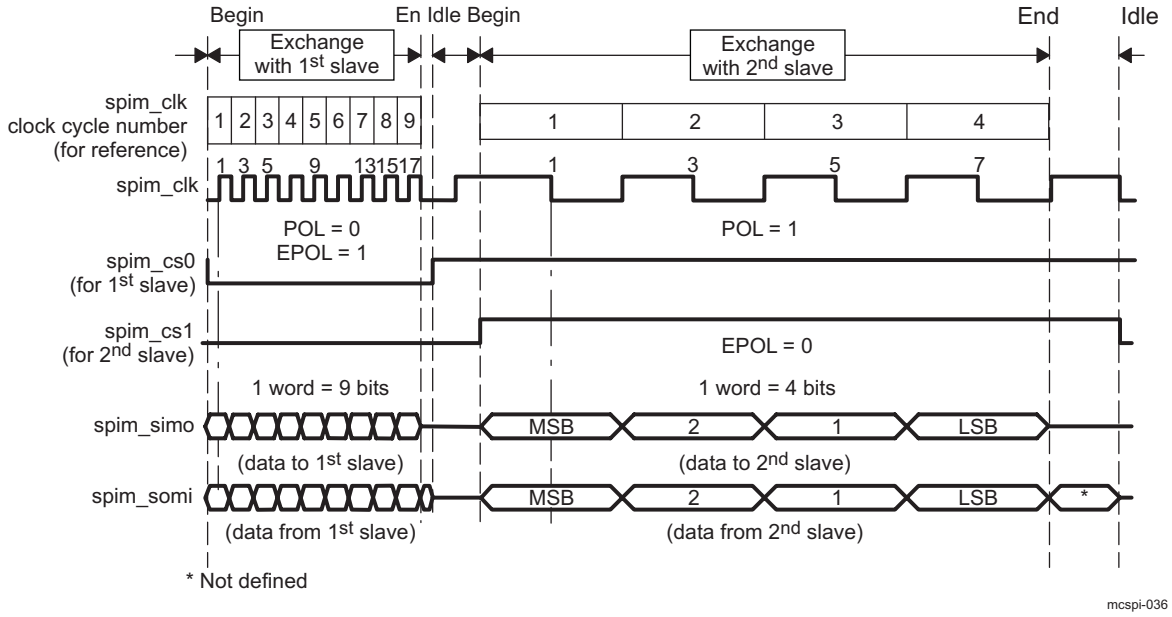
This section details an example of a typical configuration.

Figure 19-36 shows an implementation of successive transfers between two devices (slave) and the McSPI1 (master) in transmit-and-receive mode.

McSPI1 is the master, and spi1_simo shifts out the data to the external slaves. spi1_somi is connected to the data output port of two slave devices. The McSPI1 controls the first device with the spi1_cs0 signal (active low) for the exchange of 9-bit words synchronized with an active-high SPI clock.

The second device is controlled by the spi1_cs1 control signal (active high) for the exchange of 4-bit words synchronized with a SPI clock (active low) with a slower frequency than used with the first slave device.

Figure 19-36. Two SPI Transfers With PHA = 0 (Flexibility of McSPI)



This section details the steps required for this type of transmission.

19.6.2.6.1 McSPI Initialization Sequence

As shown in Figure 19-26, the McSPI module must first reset all registers and all state-machines.

For a software reset:

1. Set the SPI1.MCSPI_SYSCONFIG[1] SOFTRESET bit to 1.
2. Read the SPI1.MCSPI_SYSSTATUS[0] RESETDONE bit and check that it is set to 1.

19.6.2.6.2 Operations for the First Slave (On Channel 0)

19.6.2.6.2.1 Programming in Polling Mode

19.6.2.6.2.1.1 Mode Selection

The SPI1.MCSPI_CHxCONF register (with x = 0) allows configuration of the operating mode:

1. Set the SPI1.MCSPI_CHxCONF[18] IS bit to 0 for the spi1_somi pin in receive mode.
2. Set the SPI1.MCSPI_CHxCONF[17] DPE1 bit to 0 and the SPI1.MCSPI_CHxCONF[16] DPE0 bit to 1 for the spi1.simo pin in transmit mode.
3. Set the SPI1.MCSPI_CHxCONF[13:12] TRM field to 0x0 for transmit and receive mode.
4. Write 0x8 in the SPI1.MCSPI_CHxCONF[11:7] WL field for 9-bit word length.

5. Set the SPI1.MCSPI_CHxCONF[6] EPOL bit to 1 for spi1_cs0 activated low during active state.
6. Set the SPI1.MCSPI_CHxCONF[1] POL bit to 0 for spi1_clk held high during active state.
7. Set the SPI1.MCSPI_CHxCONF[0] PHA bit to 0 for data latched on odd-numbered edges of the SPI clock.

   Clock Initialization and spi1_cs0 Enable

   In master mode, the SPI must provide the clock and enable the channel:
8. Set the SPI1.MCSPI_MODULCTRL[2] MS bit to 0 to provide the clock.
9. Set the SPI1.MCSPI_CHxCTRL[0] EN bit to 1 (where x = 0) to enable channel 0.

19.6.2.6.2.1.2 Write Operation

1. Write 1 to the SPI1.MCSPI_IRQSTATUS[0] TX0_EMPTY bit to reset the status.
2. Write the command/address or data value in the SPI1.MCSPI_TX0 register to transmit the value.
3. If the SPI1.MCSPI_IRQSTATUS[0] TX0_EMPTY bit is set to 1, write 1 to it and return to Step 2 (polling method).

19.6.2.6.2.1.3 Read Operation

1. Read the SPI1.MCSPI_IRQSTATUS[2] RX0_FULL bit and if it is set to 1, go to Step 2.
2. If the SPI1.MCSPI_IRQSTATUS[2] RX0_FULL bit is set to 1, write 1 to it and return to Step 1 (polling method).

Note: Write and read operations can be performed simultaneously.

19.6.2.6.3 Programming in Interrupt Mode

This section follows the flow of Figure 19-26.

1. Initialize software variables: WRITE_COUNT = 0 and READ_COUNT = 0.
2. Initialize interrupts: Write 0x7 in the SPI1.MCSPI_IRQSTATUS[3:0] field and set the SPI1.MCSPI_IRQENABLE[3:0] field to 0x7.
3. Follow the steps described in Section 19.6.2.6.2.1.1, *Mode Selection*.
4. If WRITE_COUNT = w and READ_COUNT = w, write SPI1.MCSPI_CHxCTRL[0] = 0x0 (x = 0) to stop the channel.

This interrupt routine follows the flow of Table 19-16 and Figure 19-28.

1. Read the SPI1.MCSPI_IRQSTATUS[3:0] field.
2. If the SPI1.MCSPI_IRQSTATUS[0] TX0_EMPTY bit is set to 1:
   a. Write the command/address or data value in SPI1.MCSPI_TXx (where x = 0).
   b. WRITE_COUNT + = 1
   c. Write SPI1.MCSPI_IRQSTATUS[0] = 0x1.
3. If the SPI1.MCSPI_IRQSTATUS[2] RX0_FULL bit is set to 1:
   a. Read SPI1.MCSPI_RXx (where x = 0)
   b. READ_COUNT += 1
   c. Write SPI1.MCSPI_IRQSTATUS[2] = 0x1

19.6.2.6.4 Operations for the Second Slave (on Channel 1) in Polling Mode

19.6.2.6.4.1 Mode Selection

The SPI1.MCSPI_CHxCONF register (with x = 1) allows configuration of the operating mode:

1. Set the SPI1.MCSPI_CH1CONF[18] IS bit to 0 for the spi1_somi pin in receive mode.
2. Set the SPI1.MCSPI_CH1CONF[17] DPE1 bit to 0 and the SPI1.MCSPI_CH1CONF[16] DPE0 bit to 1 for the spi1_somi pin in transmit mode.
3. Set the SPI1.MCSPI_CH1CONF[13:12] TRM field to 0x0 for transmit-and-receive mode.

4. Write 0x3 in the SPI1.MCSPI_CH1CONF[11:7] WL field.
5. Set the SPI1.MCSPI_CH1CONF[6] EPOL bit to 0 for spi1_cs1 activated high during the active state.
6. Set the SPI1.MCSPI_CH1CONF[1] POL bit to 1 for spi1_clk held low during the active state.
7. Set the SPI1.MCSPI_CH1CONF[0] PHA bit to 1 for data latched on even numbered edges of spi1_clk.

   Clock Initialization and spi1_cs1 Enable

   The SPI1.MCSPI_MODULCTRL[2] MS bit was set to 0 (providing the clock).

   In master mode, the SPI must provide the clock and enable the channel:
8. Set the SPI1.MCSPI_CH0CTRL[0] EN bit to 0 to disable channel 0, and set the SPI1.MCSPI_CH1CTRL[0] EN bit to 1 to enable channel 1.

Note: Read and write operations for the second slave are identical to those for the first slave.

19.6.2.6.4.2 Write Operation

1. Write 1 to the SPI1.MCSPI_IRQSTATUS[4] TX1_EMPTY bit to reset the status.
2. Write the command/address or data value in the SPI1.MCSPI_TX1 register to transmit the value.
3. If the SPI1.MCSPI_IRQSTATUS[4] TX1_EMPTY bit is set to 1, write 1 to it and return to Step 2 (polling method).

19.6.2.6.4.3 Read Operation

1. Read the SPI1.MCSPI_IRQSTATUS[6] RX1_FULL bit and if it is set to 1, go to Step 2.
2. If the SPI1.MCSPI_IRQSTATUS[6] RX1_FULL bit is set to 1, write 1 to it and return to Step 1 (polling method).

Note: Write and read operations can be performed simultaneously.

19.6.3 Transfer Procedures with FIFO

In the subsections below, the transfer procedures are described with FIFO using (MCSPI_CHxCONF[27:28] FFER or/and FFEW = 1).

The MCSPI module allows the transfer of one or more words, according to different modes:

- Master normal, master turbo, slave
- Transmit-and-receive, transmit-only, receive-only
- Write and read requests: Interrupts, DMA

For these flows, the host process contains the main process and the interrupt routine, which is called on the IRQ signals or by an internal call if the module is used in polling mode.

19.6.3.1 Common Transfer Procedure

The common transfer sequence is the host sequence for a transfer of any type defined above.

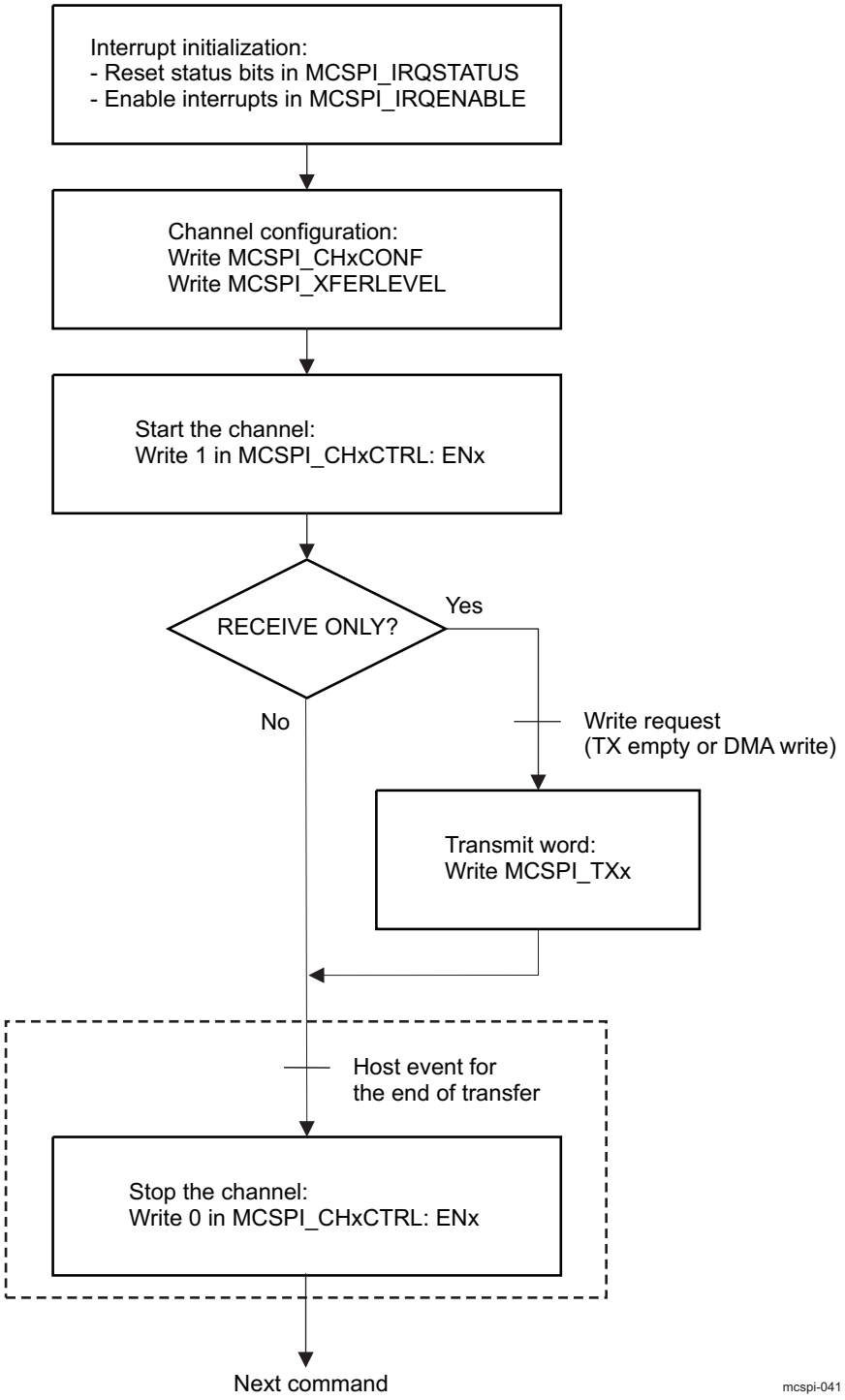
In multichannel, only one channel can use the FIFO. Before enabling the FIFO for a channel (FFExW and FFExR bits in the MCSPI_CHx_CONF register), the host must ensure that the FIFO is not enabled for another channel, even if these channels are not used.

In transmit/receive mode, the FIFO can be enabled for write or read request only, without FIFO for the other request.

In slave mode, only channel 0 only can be activated. The correct spim_csx line is chosen in the MCSPI_CH0CONF register with the SPIENSLV bits.

The MCSPI module can start the transfer only when the first write request is released by writing the MCSPI_TXx register, even in receive-only mode (only one write request occurs in this case). Figure 19-37 shows the main process of the common transfer sequence.

Figure 19-37. Common Transfer Sequence/Main Process



The MCSPI module can start the transfer only after the first write request is released by writing the MCSPI_TXx register, even in receive-only mode (only one write request occurs in this case).

This first write request can be managed by the IRQ routine or DMA handler, as are as other requests. It appears in Figure 19-37 to show this point.

The end of the transfer (dotted line in Figure 19-37) is more complex and depends on the transfer type. Table 19-17 describes the different types of end-of-transfer and the transfer types to which they are applicable.

Table 19-17. End-of-Transfer Types

Word Count	Transmit/Receive	Transmit Only	Receive Only
Yes	See Section 19.6.3.2, *Transmit-Receive With Word Count*.	See Section 19.6.3.4, *Transmit-Only*.	See Section 19.6.3.5 , *Receive-Only With Word Count*.
No	See Section 19.6.3.3, *Transmit-Receive Without Word Count*.	See Section 19.6.3.4, *Transmit-Only*.	See Section 19.6.3.6, *Receive-Only Without Word Count*.

The sequence differs depending on whether word count is used (MCSPI_XFERLEVEL: WCNT is set or not). The AEL and/or AFL values can be different, but they must be multiples of the word size in the FIFO: 1, 2, or 4 bytes, according to word length.

In these sequences, the transfer to execute has a size of N words.

When accessing the FIFO, only one word is written or read for each OCP access.

In these sequences, the number of words written or read for each write or read FIFO request is:

- write_request_size
- read_request_size

If they are not submultiples of N, the last request sizes are:

- last_write_request_size (< write_request_size)
- ast_read_request_size. (< read_request_size)

The different sequences can be merged in one process to manage transfers of several types.

The end-of-transfer sequences are described from the start of the channel.

In these sequences, some soft variables are used:

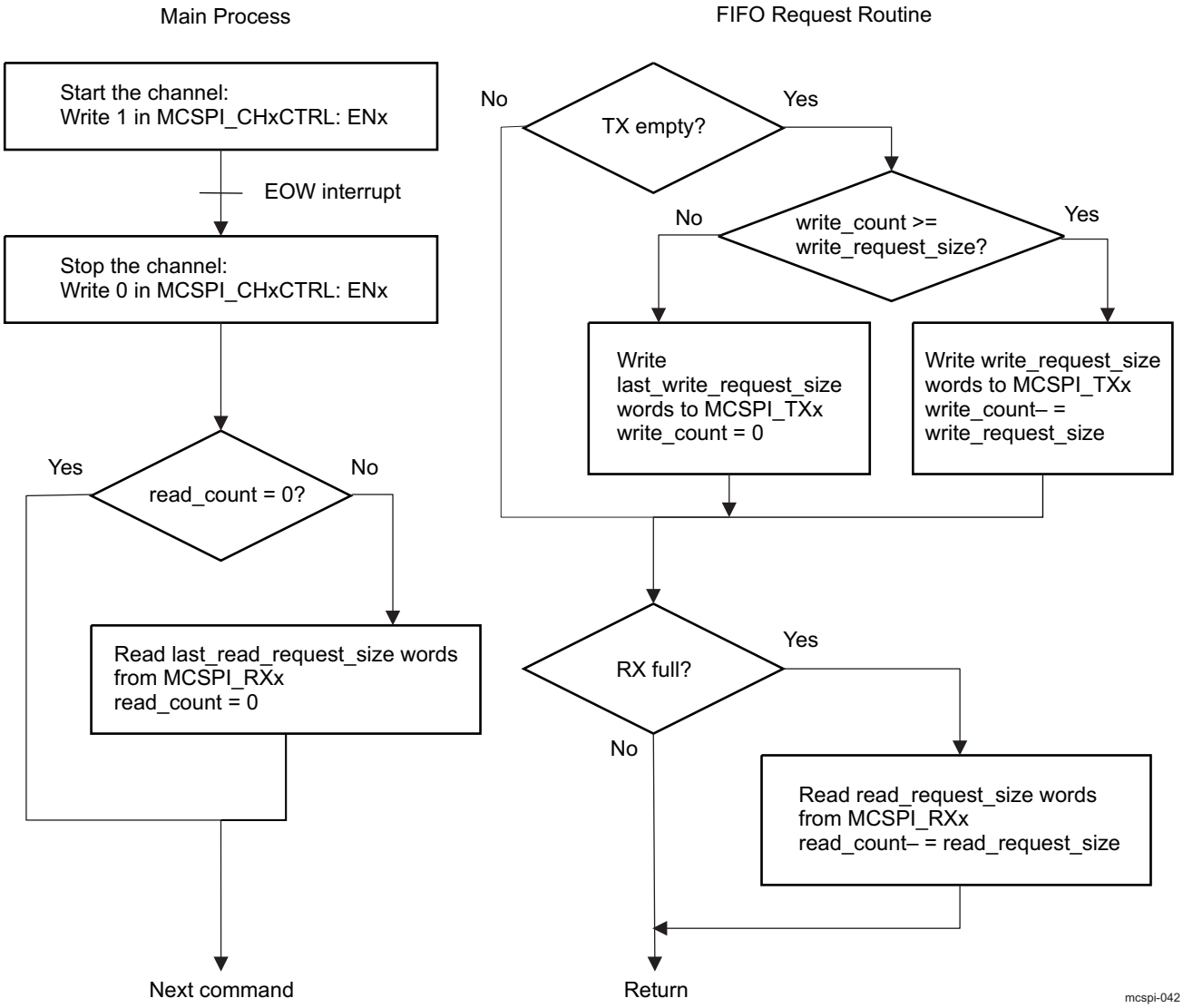
- write_count = N
- read_count = N
- last_request = FALSE

They are initialized before starting the channel.

19.6.3.2 Transmit-Receive Procedure With Word Count (WCNT≠0)

Figure 19-38 shows the flow of a transfer in transmit-receive mode, with word count.

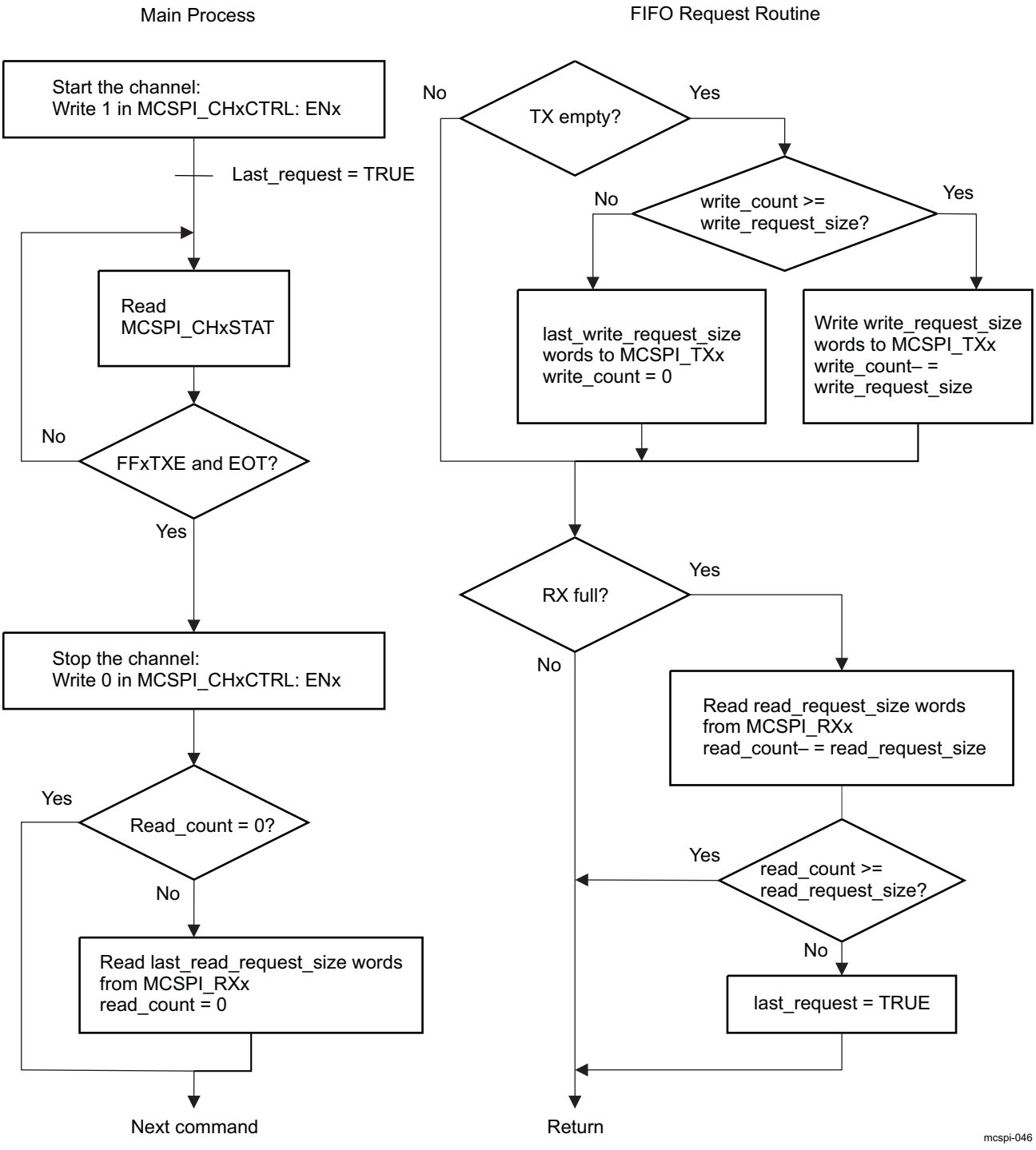
Figure 19-38. Transmit-Receive With Word Count



19.6.3.3 Transmit-Receive Procedure Without Word Count (WCNT=0)

Figure 19-39 shows the flow of a transfer in transmit-receive mode, without word count.

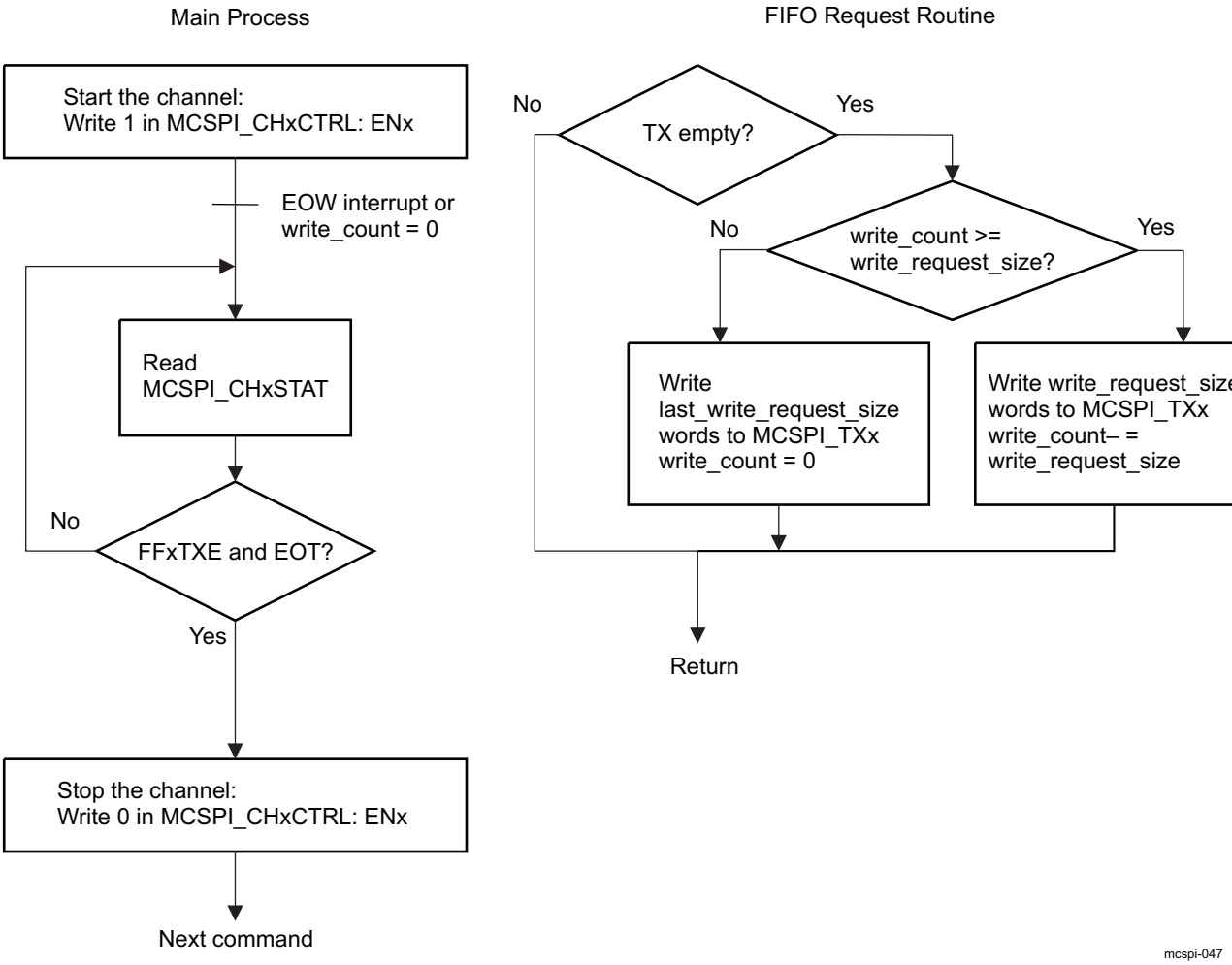
Figure 19-39. Transmit-Receive Without Word Count



19.6.3.4 Transmit-Only Procedure

Figure 19-40 shows the flow of a transfer in transmit only mode, with our without word count.

Figure 19-40. Transmit-Only



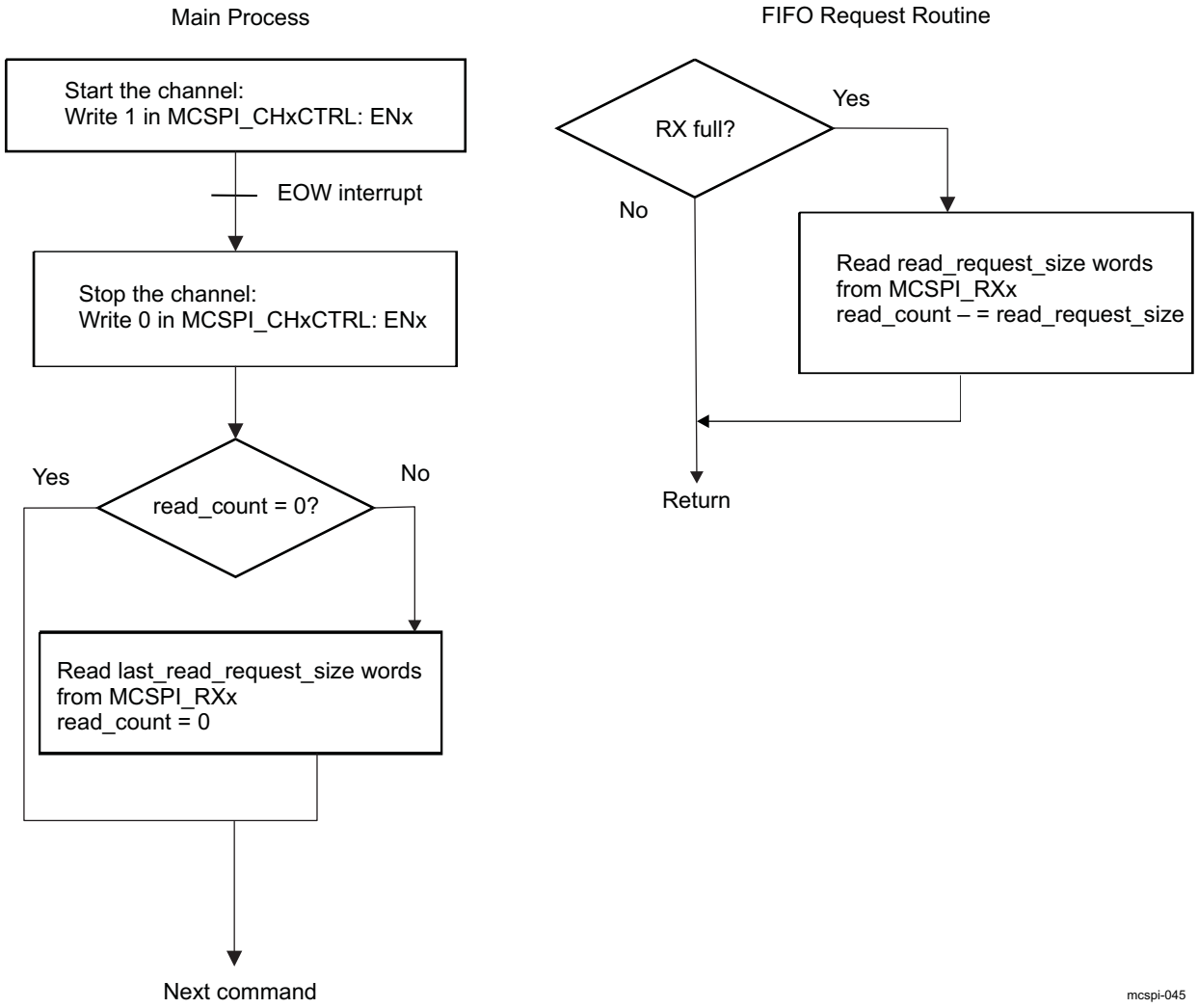
The difference between word count enabled or not is the condition after starting the channel:

- Word count enable: Wait for EOW interrupt.
- Word count disable: Wait for write_count = 0.

19.6.3.5 Receive-Only Procedure With Word Count (WCNT≠0)

Figure 19-41 shows the flow of a transfer in receive-only mode, with word count.

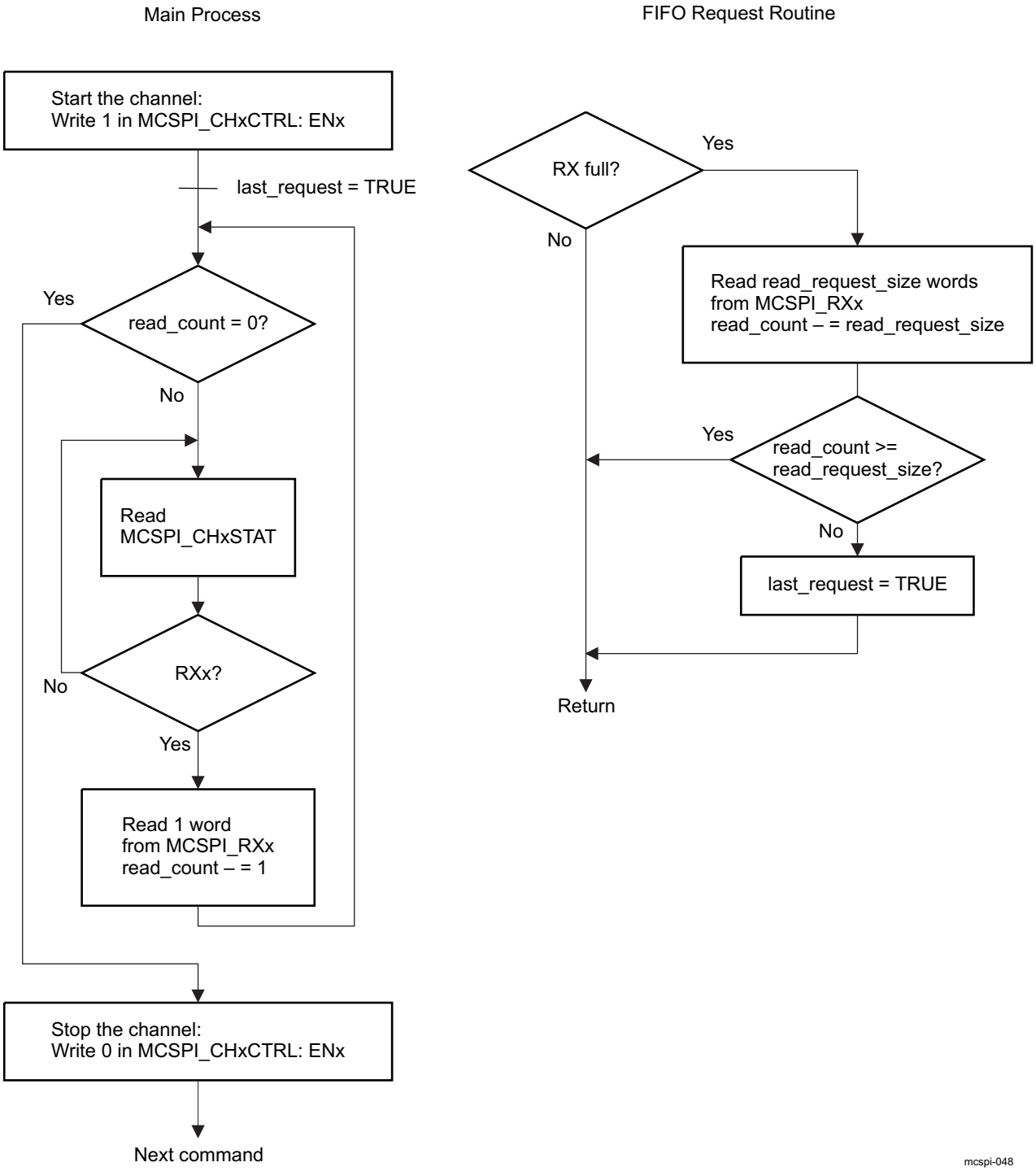
Figure 19-41. Receive-Only With Word Count



19.6.3.6 Receive-Only Procedure Without Word Count (WCNT=0)

Figure 19-42 shows the flow of a transfer in receive-only mode, without word count.

Figure 19-42. Receive-Only Without Word Count



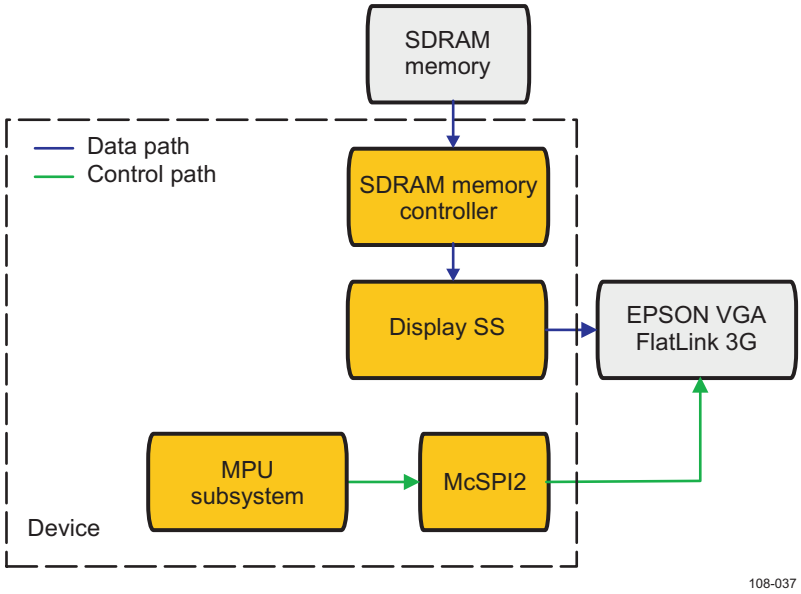
19.7 McSPI Use Cases and Tips

19.7.1 How to Configure the McSPI Interface When Connected with an EPSON VGA FlatLink™ 3G Device

19.7.1.1 Overview

This section deals with configuring an EPSON VGA FlatLink™3G device through the McSPI interface. Before transferring data from the external SDRAM memory to the external EPSON VGA display (Figure 19-43), the display subsystem must be set up via the McSPI interface. Several commands must be sent to the EPSON VGA for its configuration.

Figure 19-43. Overview



In this use case the configuration is the following:

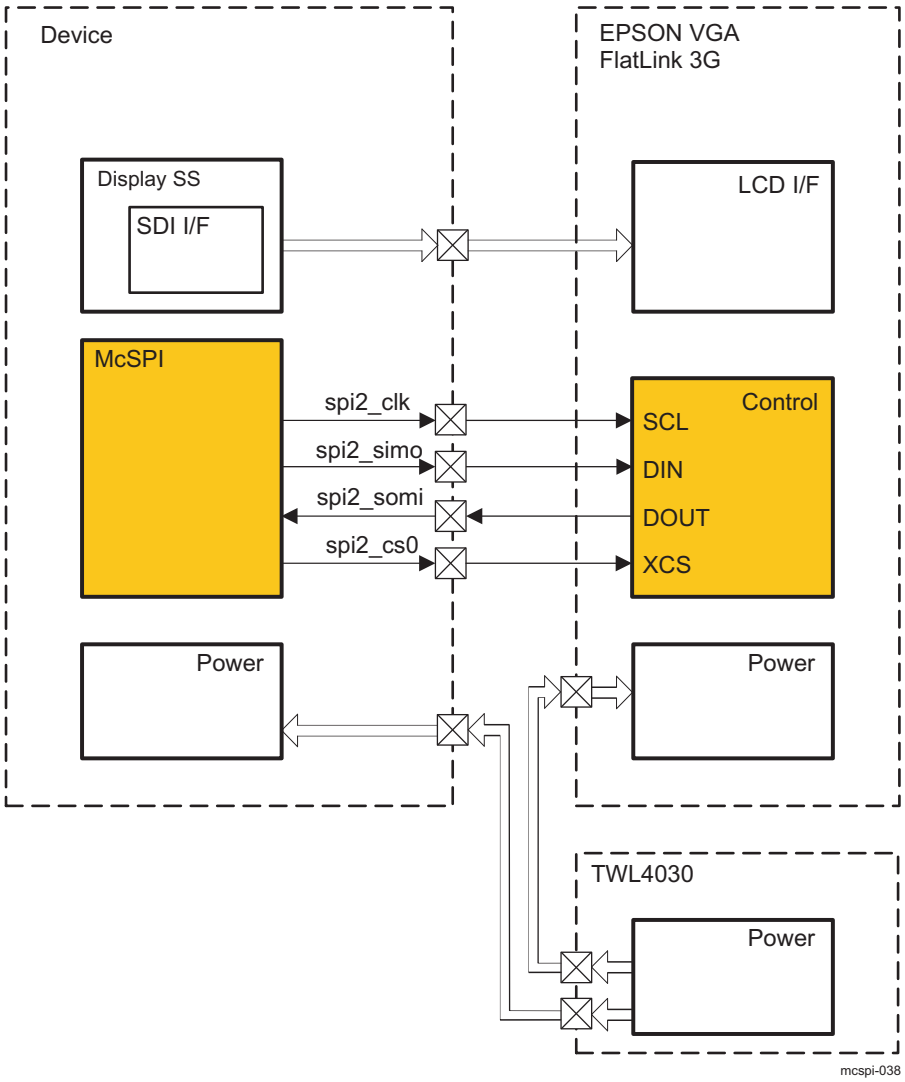
- McSPI2 channel 0 in master mode, EPSON VGA display in slave mode
- Transmit-only and receive-only modes
- No DMA requests and no interrupts used: transmit and receive flag polling

19.7.1.2 Environment

In this use case, the EPSON VGA is connected to the device through channel 0 of the McSPI instance 2 as shown in Figure 19-44 (through mcspi2_somi, mcspi2_simo, mcspi2_cs0).

The display clocks are provided to the EPSON VGA by the device display subsystem. The display configuration clock comes from the McSPI interface (mcspi2_clk).

Figure 19-44. Environment

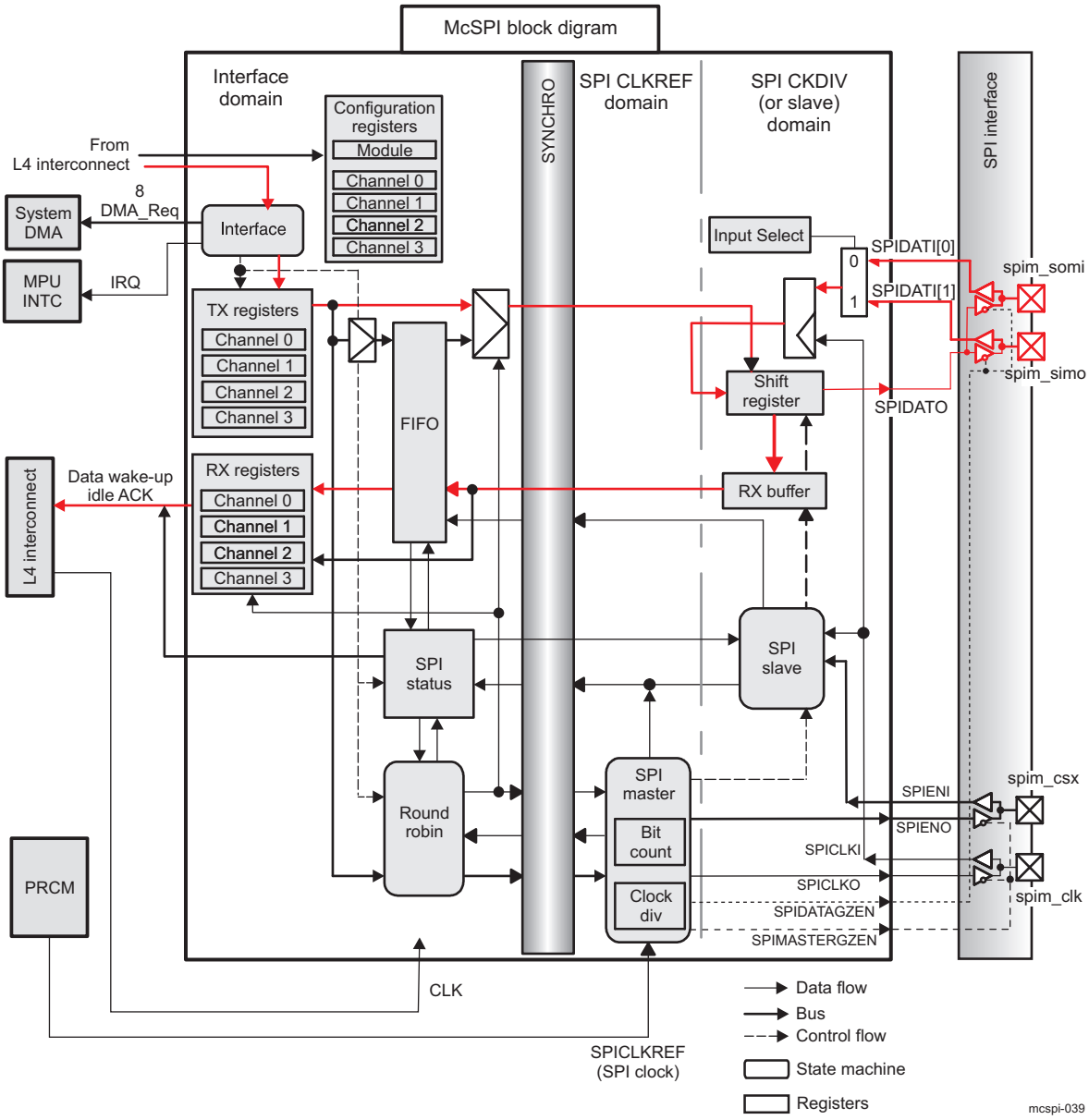


19.7.1.3 Data Path

In this use case, the McSPI2 manages the control path between the MPU subsystem and the EPSON VGA FlatLink™ 3G.

- Display configuration commands are transmitted via the channel 0 transmit register (TX Registers) and the shift register.
- A display status is read via the shift register, the receive buffer, and the channel 0 receive register (RX Registers).

Figure 19-45. McSPI Data Flow



19.7.1.4 Programming Flow

The initialization of the EPSON VGA is done in five steps:

1. McSPI module configuration
2. Send a 'SOFT RESET' command to the EPSON VGA
3. Send a 'SLEEP OUT' command
4. Send a 'DISPLAY ON' command
5. Send a 'READ DISPLAY STATUS' command and receive status data.

The four EPSON VGA commands needed in the EPSON VGA configuration are described in Table 19-18.

Table 19-18. EPSON VGA Configuration Commands

	D/CX	D7	D6	D5	D4	D3	D2	D1	D0	Dummy bit
SOFT RESET (001h)	0	0	0	0	0	0	0	0	1	n/a
SLEEP OUT (011h)	0	0	0	0	1	0	0	0	1	n/a
DISPLAY ON (029h)	0	0	0	1	0	1	0	0	1	n/a
READ DISPLAY STATUS (009h and dummy bit '0')	0	0	0	0	0	1	0	0	1	0

The following subsections describe each of these initialization steps.

19.7.1.4.1 McSPI Module Configuration

Before configuring the external display VGA, the multi-channel serial port interface must be initialized as detailed in the following steps:

1. Enable the McSPI2 functional and interface clocks through the PRCM.CM_FCLKEN1_CORE[18] EN_MCSPI2 and PRCM.CM_ICLKEN1_CORE[18] EN_MCSPI2 fields.
2. Proceed to a McSPI software reset through the MCSPI_SYSCONFIG[1] SOFTRESET field. The SOFTRESET bit is automatically reset by hardware.
3. Poll the MCSPI_SYSSTATUS[0] RESETDONE field to check the software reset status.
4. Disable channel 0 through the MCSPI_CHxCTRL[0] EN field (with x=0).
5. Disable all interrupts (MCSPI_IRQENABLE register) then clear interrupt status register (MCSPI_IRQSTATUS). Interrupts are disabled once at this stage. Even if some interrupts of the MCSPI_IRQSTATUS register are set, no interrupt can go out to the MPU subsystem.
6. Set the module in Master mode and multichannel mode through the MCSPI_MODULCTRL register.
7. Configure channel 0 in transmit-only mode, with a word length of 9-bit through the MCSPI_CHxCONF register (with x=0). A world length of 9 bits is required to transmit the required command (SOFT RESET, SLEEP OUT, or DISPLAY ON) to the EPSON display, as detailed in Table 19-18. Those 9 bits are serially transmitted on spi2_simo in the sequence D/CX, D7 to D0.

Table 19-19 shows all registers to be configured with the required value for the McSPI module configuration and the use case values.

Table 19-19. McSPI Configuration Registers Print

Register Name (with x=0)	Physical Address	Value	Value Description
PRCM.CM_FCLKEN1_CORE[18]	0x4800 4A00	0x1	Enable McSPI2 functional clock
PRCM.CM_ICLKEN1_CORE[18]	0x4800 4A10	0x1	Enable McSPI2 interface clock
MCSPI_SYSCONFIG[1]	0x4809 A010	0x1	Initiate a software reset
MCSPI_SYSSTATUS	0x4809 A014	0x0000 0001	The RESETDONE field is set at 1 when the reset is complete
MCSPI_CHxCTRL	0x4809 A034	0x0000 0000	Disable channel 0
MCSPI_IRQENABLE	0x4809 A01C	0x0000 0000	Disable all interrupts
MCSPI_IRQSTATUS	0x4809 A018	0x0001 777F	Clear all interrupts
MCSPI_MODULCTRL	0x4809 A028	0x0000 0000	Master mode, multichannel

Table 19-19. McSPI Configuration Registers Print (continued)

MCSPI_CHxCONF	0x4809 A02C	0x0001 2453	CLKD = 4 (CLKSPIREF divided internally by 16) WL = 9-bit Transmit-only Receive on spi2_somi Transmit on spi2_simo

19.7.1.4.2 'SOFT RESET', 'SLEEP OUT' and 'DISPLAY ON' Commands

Once the McSPI interface is configured, three commands must be send to the external display VGA: a software reset to reinitiate the external display, a 'SLEEP OUT' command to wake it up, and a 'DISPLAY ON' to turn on the display. Those three commands follow the same procedure, described here below, only the command opcode changes.

1. Enable channel 0 through the MCSPI_CHxCTRL[0] EN field (with x=0).
2. Write the command opcode to the MCSPI_TXx (with x=0) register so that it could be transmitted to the external display VGA. To transmit the SOFT RESET command, write 001h in the MCSPI_TXx (with x=0) register. To transmit a SLEEP OUT command write 011h and to transmit a DISPLAY ON command write 029h.
3. Poll the MCSPI_CHxSTAT[1] TXS field (with x=0) to check for transmit status. When the word has been transmitted, TXS is automatically set to 1.
4. Disable channel 0 through the MCSPI_CHxCTRL[0] EN field (with x=0).

Table 19-20 shows all registers to be configured with the required value for the those commands and the use case values.

Table 19-20. Display Configuration Registers Print

Register Name (with x=0)	Physical Address	Value	Value Description
MCSPI_CHxCTRL	0x4809 A034	0x0000 0001	Enable channel 0
MCSPI_TXx	0x4809 A038	0x001 0x011 0x029	Transmit a 'SOFT RESET' command to the EPSON VGA display Transmit a 'SLEEP OUT' command Transmit a 'DISPLAY ON' command
MCSPI_CHxSTAT[1]	0x4809 A030	0x1	The word in MCSPI_TXx has been transmitted when the TXS flag is set to 1
MCSPI_CHxCTRL	0x4809 A034	0x0000 0000	Disable channel 0

19.7.1.4.3 'READ DISPLAY STATUS' Command

Once the EPSON VGA display is initialized, a 'READ DISPLAY STATUS' command is sent to the external display VGA to read and check the status of the component, such as Partial mode, Sleep In mode, Display status, Horizontal and Vertical Synchronization On or Off states.

The 'READ DISPLAY STATUS' command is not 9- but 10-bit long because the EPSON VGA display requires a dummy cycle between the transmission of the last command bit (D0 in Table 19-18) and the reception of the first data bit (the display status data is 32-bit long). In other words, a dummy bit is added as less significant bit of the command. Hence, those 10 bits are serially transmitted on spi2_simo in the sequence D/CX, D7 to D0, dummy bit.

1. Set the module in Master mode and single channel mode through the MCSPI_MODULCTRL register.
2. Configure channel 0 in transmit-only mode, with a word length of 10-bit through the MCSPI_CHxCONF register (with x=0).
3. Enable channel 0 through the MCSPI_CHxCTRL[0] EN field.

4. Shift left of one bit the command opcode 009h so that the required dummy bit ('0') can be sent after the D/CX and D7 to D0 bits to the EPSON VGA display. Refer to Table 19-18 for more information. Then, write this command (012h) to the MCSPI_TXx (with x=0) register so that it could be transmitted to the EPSON VGA display.
5. Poll the MCSPI_CHxSTAT[1] TXS bit (with x=0) to check transmit status. When the word has been transmitted, TXS is automatically set to 1.
6. Read the MCSPI_CHxSTAT[1] RXS bit to check receive status. If data has already been received (RXS is set to 1) read the MCSPI_RXx (with x=0) receive register; this will automatically deactivate the RXS flag (RXS is automatically cleared).
7. Configure channel 0 in receive-only mode, with a word length of 32-bit through the MCSPI_CHxCONF register.
8. Poll the MCSPI_CHxSTAT[1] RXS bit to check receive status. When data has been received, RXS is automatically set to 1. The 32-bit display status information is available in the McSPI2 channel 0 receive register: MCSPI_RXx (with x=0).
9. Set channel 0 in transmit-only mode through the MCSPI_CHxCONF register.

Table 19-21 shows all registers to be configured with the required value for the McSPI module configuration and the use case values.

Table 19-21. Display Status Check Registers Print

Register Name (with x=0)	Physical Address	Value	Value Description
MCSPI_MODULCTRL	0x4809 A028	0x0000 0001	Master mode, single channel
MCSPI_CHxCONF	0x4809 A02C	0x0011 24D3	CLKD = 4 (CLKSPIREF divided internally by 16) WL = 10-bit Transmit-only Receive on spi2_somi Transmit on spi2_simo
MCSPI_CHxCTRL	0x4809 A034	0x0000 0001	Enable channel 0
MCSPI_TXx	0x4809 A038	0x0000 0012	Send a 'READ DISPLAY STATUS' command. See Table 19-18.
MCSPI_CHxSTAT[1] TXS	0x4809 A030	0x1	The word in MCSPI_TXx has been transmitted if the TXS flag is set to 1
MCSPI_CHxSTAT[0] RXS	0x4809 A030	-	Read RXS flag. Data has been received in MCSPI_RXx if the RXS flag is set to 1
MCSPI_RXx	0x4809 A03C	-	Read MCSPI_RXx only if the RXS bit is equal to 1 (this is a dummy read that will automatically clear the RXS flag)
MCSPI_CHxCONF	0x4809 A02C	0x0011 1FD0	CLKD = 4 (CLKSPIREF divided internally by 16) WL = 32-bit Receive-only Receive on spi2_somi Transmit on spi2_simo
MCSPI_CHxSTAT[0] RXS	0x4809 A030	0x0	Data has been received in MCSPI_RXx when the RXS flag is set to 1.
MCSPI_CHxCONF	0x4809 A02C	0x0011 2453	Back in transmit-only mode Receive on spi2_somi Transmit on spi2_simo

19.8 McSPI Register Manual

19.8.1 McSPI Instance Summary

Table 19-22 lists the McSPI instances.

Table 19-22. McSPI Instance Summary

Module Name	Base Address	Size
MCSPI1	0x4809 8000	4Kbytes
MCSPI2	0x4809 A000	4Kbytes
MCSPI3	0x480B 8000	4Kbytes
MCSPI4	0x480B A000	4Kbytes

19.8.2 McSPI Register Summary

Table 19-23 lists the McSPI registers. Each register has a 32-bit width. Table 19-24 through Table 19-50 describe the register bits.

Table 19-23. McSPI Register Summary

Register	Type	Offset Address	MCSPI1 Instance Physical Address	MCSPI2 Instance Physical Address	MCSPI3 Instance Physical Address	MCSPI4 Instance Physical Address
MCSPI_REVISION	R	0x00	0x4809 8000	0x4809 A000	0x480B 8000	0x480B A000
MCSPI_SYSCONFIG	RW	0x10	0x4809 8010	0x4809 A010	0x480B 8010	0x480B A010
MCSPI_SYSSTATUS	R	0x14	0x4809 8014	0x4809 A014	0x480B 8014	0x480B A014
MCSPI_IRQSTATUS	RW	0x18	0x4809 8018	0x4809 A018	0x480B 8018	0x480B A018
MCSPI_IRQENABLE	RW	0x1C	0x4809 801C	0x4809 A01C	0x480B 801C	0x480B A01C
MCSPI_WAKEUPENABLE	RW	0x20	0x4809 8020	0x4809 A020	0x480B 8020	0x480B A020
MCSPI_SYST	RW	0x24	0x4809 8024	0x4809 A024	0x480B 8024	0x480B A024
MCSPI_MODULCTRL	RW	0x28	0x4809 8028	0x4809 A028	0x480B 8028	0x480B A028
MCSPI_CHxCONF (1)	RW	0x2C + (0x14 * x)	0x4809 802C + (0x14 * x)	0x4809 A02C + (0x14 * x)	0x480B 802C + (0x14 * x)	0x480B A02C + (0x14 * x)
MCSPI_CHxSTAT (1)	R	0x30 + (0x14 * x)	0x4809 8030 + (0x14 * x)	0x4809 A030 + (0x14 * x)	0x480B 8030 + (0x14 * x)	0x480B A030 + (0x14 * x)
MCSPI_CHxCTRL (1)	RW	0x34 + (0x14 * x)	0x4809 8034 + (0x14 * x)	0x4809 A034 + (0x14 * x)	0x480B 8034 + (0x14 * x)	0x480B A034 + (0x14 * x)
MCSPI_TXx (1)	RW	0x38 + (0x14 * x)	0x4809 8038 + (0x14 * x)	0x4809 A038 + (0x14 * x)	0x480B 8038 + (0x14 * x)	0x480B A038 + (0x14 * x)
MCSPI_RXx (1)	R	0x3C + (0x14 * x)	0x4809 803C + (0x14 * x)	0x4809 A03C + (0x14 * x)	0x480B 803C + (0x14 * x)	0x480B A03C + (0x14 * x)
MCSPI_XFERLEVEL	RW	0x7C	0x4809 807C	0x4809 A07C	0x480B 807C	0x480B A07C

(1) x= 0 to 3 for MCSPI1.
x= 0 to 1 for MCSPI2 and MCSPI3.
x= 0 for MCSPI4.

19.8.3 McSPI Register Description

Table 19-24. MCSPI_REVISION

Address Offset	0x00		
Physical Address	0x4809 8000	**Instance**	MCSPI1
	0x4809 A000		MCSPI2
	0x480B 8000		MCSPI3
	0x480B A000		MCSPI4
Description	This register contains the hard coded RTL revision number.		
Type	R		
Write Latency	Not relevant		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																								REV							

Bits	Field Name	Description	Type	Reset
31:8	Reserved	Reads returns 0	R	0x000000
7:0	REV	IP revision [7:4] Major revision [3:0] Minor revision Example: 0x10 for 1.0, 0x21 for 2.1	R	See (1)

(1) TI internal data

Table 19-25. Register Call Summary for Register MCSPI_REVISION

McSPI Register Manual

- McSPI Register Summary: [0]

Table 19-26. MCSPI_SYSCONFIG

Address Offset	0x10		
Physical Address	0x4809 8010	**Instance**	MCSPI1
	0x4809 A010		MCSPI2
	0x480B 8010		MCSPI3
	0x480B A010		MCSPI4
Description	This register allows control of various parameters of the module interface. It is not sensitive to software reset.		
Type	RW		
Write Latency	Immediate		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																						CLOCKACTIVITY		Reserved			SIDLEMODE		ENAWAKEUP		SOFTRESET		AUTOIDLE

Bits	Field Name	Description	Type	Reset
31:10	Reserved	Reads returns 0	RW	0x000000
9:8	CLOCKACTIVITY	Clocks activity during wake up mode period 0x0: Interface and Functional clocks may be switched off.	RW	0x0

Bits	Field Name	Description	Type	Reset
		0x1: Interface clock is maintained. Functional clock may be switched off.		
		0x2: Functional clock is maintained. Interface clock may be switched off.		
		0x3: Interface and Functional clocks are maintained.		
7:5	Reserved	Reads returns 0	RW	0x0
4:3	SIDLEMODE	Power management	RW	0x0
		0x0: If an idle request is detected, the McSPI acknowledges it unconditionally and goes in Inactive mode. Interrupt, DMA requests and wake up lines are unconditionally de-asserted and the module wake-up capability is deactivated even if the bit MCSPI_SYSCONFIG[ENAWAKEUP] is set.		
		0x1: If an idle request is detected, the request is ignored and the module does not switch to wake up mode and keeps on behaving normally.		
		0x2: If an idle request is detected, the module will switch to wake up mode based on its internal activity and the wake up capability can be used if the bit MCSPI_SYSCONFIG[ENAWAKEUP] is set.		
		0x3: Reserved - do not use.		
2	ENAWAKEUP	Wake-up feature control	RW	0
		0x0: Wake-up capability disabled		
		0x1: Wake-up capability enabled		
1	SOFTRESET	Software reset. Read always returns 0.	RW	0
		0x0: Normal mode.		
		0x1: Trigger a module reset. This bit is automatically reset by hardware.		
0	AUTOIDLE	Internal interface Clock gating strategy	RW	0
		0x0: interface clock is free-running		
		0x1: Automatic interface clock gating strategy is applied, based on the module interface activity		

Table 19-27. Register Call Summary for Register MCSPI_SYSCONFIG

McSPI Integration • Software Reset: [0] [1]
McSPI Functional Description • Normal Mode: [2] • Idle Mode: [3] [4] [5] [6]
McSPI Basic Programming Model • McSPI Configuration and Operations Example: [7]
McSPI Use Cases and Tips • Programming Flow: [8] [9]
McSPI Register Manual • McSPI Register Summary: [10] • McSPI Register Description: [11] [12]

Table 19-28. MCSPI_SYSSTATUS

Address Offset	0x14		
Physical Address	0x4809 8014	**Instance**	MCSPI1
	0x4809 A014		MCSPI2
	0x480B 8014		MCSPI3
	0x480B A014		MCSPI4
Description	This register provides status information about the module excluding the interrupt status information		
Type	R		
Write Latency	Immediate		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																															RESETDONE

Bits	Field Name	Description		Type	Reset
31:1	Reserved	Reserved for module specific status information. Read returns 0		R	0x00000000
0	RESETDONE	Internal Reset Monitoring		R	0
		0x0:	Internal module reset is on-going		
		0x1:	Reset completed		

Table 19-29. Register Call Summary for Register MCSPI_SYSSTATUS

McSPI Basic Programming Model
- Initialization of Modules: [0]
- McSPI Configuration and Operations Example: [1]

McSPI Use Cases and Tips
- Programming Flow: [2] [3]

McSPI Register Manual
- McSPI Register Summary: [4]

Table 19-30. MCSPI_IRQSTATUS

Address Offset	0x18		
Physical Address	0x4809 8018	**Instance**	MCSPI1
	0x4809 A018		MCSPI2
	0x480B 8018		MCSPI3
	0x480B A018		MCSPI4
Description	The interrupt status regroups all the status of the module internal events that can generate an interrupt		
Type	RW		
Write Latency	Immediate		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														EOW	WKS	Reserved	RX3_FULL	TX3_UNDERFLOW	TX3_EMPTY	Reserved	RX2_FULL	TX2_UNDERFLOW	TX2_EMPTY	Reserved	RX1_FULL	TX1_UNDERFLOW	TX1_EMPTY	RX0_OVERFLOW	RX0_FULL	TX0_UNDERFLOW	TX0_EMPTY

Bits	Field Name	Description	Type	Reset
31:18	Reserved	Reads returns 0	RW	0x0000
17	EOW	End of word count event when a channel is enabled using the FIFO buffer and the channel had sent the number of SPI word defined by MCSPI_XFERLEVEL[31:16] WCNT bit field. Write 0x0: Event status bit unchanged. Read 0x0: Event false. Write 0x1: Event status bit is reset. Read 0x1: Event is pending.	RW	0x0
16	WKS	Wake-up event in slave mode when an active control signal is detected on the spim_csx line programmed in the MCSPI_CHxCONF[SPIENSLV] field (where x=0 only) Write 0x0: Event status bit unchanged Read 0x0: Event false Write 0x1: Event status bit is reset. Read 0x1: Event is pending.	RW	0x0
15	Reserved	Reads returns 0.	RW	0x0
14	RX3_FULL	MCSPI_RX3 register is full (only when channel 3 is enabled) Write 0x0: Event status bit unchanged Read 0x0: Event false Write 0x1: Event status bit is reset. Read 0x1: Event is pending.	RW	0x0
13	TX3_UNDERFLOW	MCSPI_TX3 register underflow (only when channel 3 is enabled) (1) Write 0x0: Event status bit unchanged Read 0x0: Event false Write 0x1: Event status bit is reset. Read 0x1: Event is pending.	RW	0x0
12	TX3_EMPTY	MCSPI_TX3 register is empty (only when channel 3 is enabled) (2) Write 0x0: Event status bit unchanged Read 0x0: Event false Write 0x1: Event status bit is reset. Read 0x1: Event is pending.	RW	0x0
11	Reserved	Read returns 0.	RW	0x0
10	RX2_FULL	MCSPI_RX2 register full (only when channel 2 is enabled) Write 0x0: Event status bit unchanged Read 0x0: Event false Write 0x1: Event status bit is reset. Read 0x1: Event is pending.	RW	0x0
9	TX2_UNDERFLOW	MCSPI_TX2 register underflow (only when channel 2 is enabled) (1) Write 0x0: Event status bit unchanged Read 0x0: Event false Write 0x1: Event status bit is reset. Read 0x1: Event is pending.	RW	0x0
8	TX2_EMPTY	MCSPI_TX2 register empty (only when channel 2 is enabled) (2) Write 0x0: Event status bit unchanged	RW	0x0

(1) The MCSPI_TXx register is empty (not updated by host or DMA with new data) before its time slot assignment.
Exception: No TX_UNDERFLOW event when no data has been loaded into the MCSPI_TXx register since channel has been enabled.

(2) Enabling the channel automatically rises this event.

Bits	Field Name	Description	Type	Reset
		Read 0x0: Event false		
		Write 0x1: Event status bit is reset.		
		Read 0x1: Event is pending.		
7	Reserved	Read returns 0.	RW	0x0
6	RX1_FULL	MCSPI_RX1 register full (only when channel 1 is enabled)	RW	0x0
		Write 0x0: Event status bit unchanged		
		Read 0x0: Event false		
		Write 0x1: Event status bit is reset.		
		Read 0x1: Event is pending.		
5	TX1_UNDERFLOW	MCSPI_TX1 register underflow (only when channel 1 is enabled)(1)	RW	0x0
		Write 0x0: Event status bit unchanged		
		Read 0x0: Event false		
		Write 0x1: Event status bit is reset.		
		Read 0x1: Event is pending.		
4	TX1_EMPTY	MCSPI_TX1 register empty (only when channel 1 is enabled)(2)	RW	0x0
		Write 0x0: Event status bit unchanged		
		Read 0x0: Event false		
		Write 0x1: Event status bit is reset.		
		Read 0x1: Event is pending.		
3	RX0_OVERFLOW	MCSPI_RX0 register overflow (only in slave mode)	RW	0x0
		Write 0x0: Event status bit unchanged		
		Read 0x0: Event false		
		Write 0x1: Event status bit is reset.		
		Read 0x1: Event is pending.		
2	RX0_FULL	MCSPI_RX0 register full (only when channel 0 is enabled)	RW	0x0
		Write 0x0: Event status bit unchanged		
		Read 0x0: Event false		
		Write 0x1: Event status bit is reset.		
		Read 0x1: Event is pending.		
1	TX0_UNDERFLOW	MCSPI_TX0 register underflow (only when channel 0 is enabled)(1)	RW	0x0
		Write 0x0: Event status bit unchanged		
		Read 0x0: Event false		
		Write 0x1: Event status bit is reset.		
		Read 0x1: Event is pending.		
0	TX0_EMPTY	MCSPI_TX0 register empty (only when channel 0 is enabled)(2)	RW	0x0
		Write 0x0: Event status bit unchanged		
		Read 0x0: Event false		
		Write 0x1: Event status bit is reset.		
		Read 0x1: Event is pending.		

Table 19-31. Register Call Summary for Register MCSPI_IRQSTATUS

McSPI Functional Description

- Master Transmit-and-Receive Mode (Full Duplex): [0]
- Master Transmit-Only Mode (Half Duplex): [1]
- Master Receive-Only Mode (Half Duplex): [2] [3] [4] [5]
- Single-Channel Master Mode: [6] [7]
- Buffer Almost Full: [8] [9]
- Buffer Almost Empty: [10] [11]
- End of Transfer Management: [12]
- Interrupts: [13]
- Interrupt Events in Master Mode: [14] [15] [16] [17] [18] [19]
- Interrupt Events in Slave Mode: [20] [21] [22] [23] [24] [25] [26]
- Interrupt-Driven Operation: [27] [28]
- Polling: [29] [30]
- Idle Mode: [31] [32] [33]

McSPI Basic Programming Model

- McSPI Configuration and Operations Example: [34] [35] [36] [37] [38] [39] [40] [41] [42] [43] [44] [45] [46] [47]

McSPI Use Cases and Tips

- Programming Flow: [48] [49] [50]

McSPI Register Manual

- McSPI Register Summary: [51]
- McSPI Register Description: [52]

Table 19-32. MCSPI_IRQENABLE

Address Offset	0x1C		
Physical Address	0x4809 801C	**Instance**	MCSPI1
	0x4809 A01C		MCSPI2
	0x480B 801C		MCSPI3
	0x480B A01C		MCSPI4
Description	This register allows to enable/disable the module internal sources of interrupt, on an event-by-event basis.		
Type	RW		
Write Latency	Immediate		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														EOWKE	WKE	Reserved	RX3_FULL_ ENABLE	TX3_UNDERFLOW_ ENABLE	TX3_EMPTY_ ENABLE	Reserved	RX2_FULL_ ENABLE	TX2_UNDERFLOW_ ENABLE	TX2_EMPTY_ ENABLE	Reserved	RX1_FULL_ ENABLE	TX1_UNDERFLOW_ ENABLE	TX1_EMPTY_ ENABLE	RX0_OVERFLOW_ ENABLE	RX0_FULL_ ENABLE	TX0_UNDERFLOW_ ENABLE	TX0_EMPTY_ ENABLE

Bits	Field Name	Description	Type	Reset
31:18	Reserved	Reads return 0	RW	0x0000
17	EOWKE	End of Word count Interrupt Enable. 0x0: Interrupt disabled 0x1: Interrupt enabled	RW	0x0
16	WKE	Wake-up event interrupt enable in slave mode when an active control signal is detected on the spim_csx line programmed in the MCSPI_CHxCONF[SPIENSLV] field (where x=0 only)	RW	0x0

Bits	Field Name	Description	Type	Reset
		0x0: Interrupt disabled		
		0x1: Interrupt enabled		
15	Reserved	Read returns 0.	RW	0x0
14	RX3_FULL_ ENABLE	MCSPI_RX3 register full interrupt enable (channel 3)	RW	0x0
		0x0: Interrupt disabled		
		0x1: Interrupt enabled		
13	TX3_UNDERFLOW_ ENABLE	MCSPI_TX3 register underflow interrupt enable (channel 3)	RW	0x0
		0x0: Interrupt disabled		
		0x1: Interrupt enabled		
12	TX3_EMPTY_ ENABLE	MCSPI_TX3 register empty interrupt enable (channel 3)	RW	0x0
		0x0: Interrupt disabled		
		0x1: Interrupt enabled		
11	Reserved	Read returns 0.	RW	0x0
10	RX2_FULL_ ENABLE	MCSPI_RX2 register full interrupt enable (channel 2)	RW	0x0
		0x0: Interrupt disabled		
		0x1: Interrupt enabled		
9	TX2_UNDERFLOW_ ENABLE	MCSPI_TX2 register underflow interrupt enable (channel 2)	RW	0x0
		0x0: Interrupt disabled		
		0x1: Interrupt enabled		
8	TX2_EMPTY_ ENABLE	MCSPI_TX2 register empty interrupt enable (channel 2)	RW	0x0
		0x0: Interrupt disabled		
		0x1: Interrupt enabled		
7	Reserved	Read returns 0.	RW	0x0
6	RX1_FULL_ ENABLE	MCSPI_RX1 register full interrupt enable (channel 1)	RW	0x0
		0x0: Interrupt disabled		
		0x1: Interrupt enabled		
5	TX1_UNDERFLOW_ ENABLE	MCSPI_TX1 register underflow interrupt enable (channel 1)	RW	0x0
		0x0: Interrupt disabled		
		0x1: Interrupt enabled		
4	TX1_EMPTY_ ENABLE	MCSPI_TX1 register empty interrupt enable (channel 1)	RW	0x0
		0x0: Interrupt disabled		
		0x1: Interrupt enabled		
3	RX0_OVERFLOW_ ENABLE	MCSPI_RX0 register overflow interrupt enable (channel 0)	RW	0x0
		0x0: Interrupt disabled		
		0x1: Interrupt enabled		
2	RX0_FULL_ ENABLE	MCSPI_RX0 register full interrupt enable (channel 0)	RW	0x0
		0x0: Interrupt disabled		
		0x1: Interrupt enabled		
1	TX0_UNDERFLOW_ ENABLE	MCSPI_TX0 register underflow interrupt enable (channel 0)	RW	0x0
		0x0: Interrupt disabled		
		0x1: Interrupt enabled		
0	TX0_EMPTY_ ENABLE	MCSPI_TX0 register empty interrupt enable (channel 0)	RW	0x0
		0x0: Interrupt disabled		
		0x1: Interrupt enabled		

Table 19-33. Register Call Summary for Register MCSPI_IRQENABLE

McSPI Functional Description • Interrupts: [0] • Interrupt-Driven Operation: [1] • Polling: [2] • Idle Mode: [3] [4]
McSPI Basic Programming Model • McSPI Configuration and Operations Example: [5]
McSPI Use Cases and Tips • Programming Flow: [6] [7]
McSPI Register Manual • McSPI Register Summary: [8]

Table 19-34. MCSPI_WAKEUPENABLE

Address Offset	0x20		
Physical Address	0x4809 8020	**Instance**	MCSPI1
	0x4809 A020		MCSPI2
	0x480B 8020		MCSPI3
	0x480B A020		MCSPI4
Description	The wake-up enable register allows to enable/disable the module internal sources of wake-up on event-by-event basis.		
Type	RW		
Write Latency	Immediate		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																															WKEN

Bits	Field Name	Description	Type	Reset
31:1	Reserved	Reads return 0	RW	0x00000000
0	WKEN	Wake-up functionality in slave mode when an active control signal is detected on the spim_cs line programmed in the field MCSPI_CHxCONF[SPIENSLV] 0x0: The event is not allowed to wake-up the system, even if the global control bit MCSPI_SYSCONF[ENAWAKEUP] is set. 0x1: The event is allowed to wake-up the system if the global control bit MCSPI_SYSCONF[ENAWAKEUP] is set.	RW	0

Table 19-35. Register Call Summary for Register MCSPI_WAKEUPENABLE

McSPI Functional Description • Idle Mode: [0] [1] [2]
McSPI Register Manual • McSPI Register Summary: [3]

Table 19-36. MCSPI_SYST

Address Offset	0x24		
Physical Address	0x4809 8024	**Instance**	MCSPI1
	0x4809 A024		MCSPI2
	0x480B 8024		MCSPI3
	0x480B A024		MCSPI4
Description	This register is used to check the correctness of the system interconnect either internally to peripheral bus, or externally to device IO pads, when the module is configured in system test (SYSTEST) mode.		
Type	RW		
Write Latency	Immediate		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																				SSB	SPIENDIR	SPIDATDIR1	SPIDATDIR0	WAKD	SPICLK	SPIDAT_1	SPIDAT_0	SPIEN_3	PIEN_2	SPIEN_1	SPIEN_0

Bits	Field Name	Description	Type	Reset
31:12	Reserved	Reads returns 0	RW	0x00000
11	SSB	Set status bit 0x0: No action. Writing 0 does not clear already set status bits; This bit must be cleared prior attempting to clear a status bit of the MCSPI_ IRQSTATUS register. 0x1: Force to 1 all status bits of MCSPI_ IRQSTATUS register. Writing 1 into this bit sets to 1 all status bits contained in the MCSPI_IRQSTATUS register.	RW	0
10	SPIENDIR	Set the direction of the spim_cs lines and spim_clk line 0x0: Output (as in master mode) 0x1: Input (as in slave mode)	RW	0
9	SPIDATDIR1	Set the direction of the SPIDAT[1] (spim_simo) 0x0: Output 0x1: Input	RW	0
8	SPIDATDIR0	Set the direction of the SPIDAT[0] (spim_somi) 0x0: Output 0x1: Input	RW	0
7	WAKD	SWAKEUP output (signal data value of internal signal to system). The signal is driven high or low according to the value written into this register bit. 0x0: The pin is driven low. 0x1: The pin is driven high.	RW	0
6	SPICLK	spim_clk line (signal data value) If MCSPI_SYST[SPIENDIR] = 1 (input mode direction), this bit returns the value on the spim_clk line (high or low) and a write into this bit has no effect. If MCSPI_SYST[SPIENDIR] = 0 (output mode direction), the spim_clk line is driven high or low according to the value written into this register.	RW	0
5	SPIDAT_1	spim_somi line (signal data value) If MCSPI_SYST[SPIDATDIR1] = 0 (output mode direction), the spim_somi line is driven high or low according to the value written into this register.	RW	0

Bits	Field Name	Description	Type	Reset
		If MCSPI_SYST[SPIDATDIR1] = 1 (input mode direction), this bit returns the value on the spim_somi line (high or low) and a write into this bit has no effect.		
4	SPIDAT_0	spim_simo line (signal data value) If MCSPI_SYST[SPIDATDIR0] = 0 (output mode direction), the spim_simo line is driven high or low according to the value written into this register. If MCSPI_SYST[SPIDATDIR0] = 1 (input mode direction), this bit returns the value on the spim_simo line (high or low) and a write into this bit has no effect.	RW	0
3	SPIEN_3	spim_cs3 line (signal data value) If MCSPI_SYST[SPIENDIR] = 0 (output mode direction), the spim_cs3 line is driven high or low according to the value written into this register. If MCSPI_SYST[SPIENDIR] = 1 (input mode direction), this bit returns the value on the spim_cs3 line (high or low) and a write into this bit has no effect.	RW	0
2	PIEN_2	spim_cs2 line (signal data value) If MCSPI_SYST[SPIENDIR] = 0 (output mode direction), the spim_cs2 line is driven high or low according to the value written into this register. If MCSPI_SYST[SPIENDIR] = 1 (input mode direction), this bit returns the value on the spim_cs2 line (high or low) and a write into this bit has no effect.	RW	0
1	SPIEN_1	spim_cs1 line (signal data value) If MCSPI_SYST[SPIENDIR] = 0 (output mode direction), the spim_cs1 line is driven high or low according to the value written into this register. If MCSPI_SYST[SPIENDIR] = 1 (input mode direction), this bit returns the value on the spim_cs1 line (high or low) and a write into this bit has no effect.	RW	0
0	SPIEN_0	spim_cs0 line (signal data value) If MCSPI_SYST[SPIENDIR] = 0 (output mode direction), the spim_cs0 line is driven high or low according to the value written into this register. If MCSPI_SYST[SPIENDIR] = 1 (input mode direction), this bit returns the value on the spim_cs0 line (high or low) and a write into this bit has no effect.	RW	0

Table 19-37. Register Call Summary for Register MCSPI_SYST

McSPI Register Manual

- McSPI Register Summary: [0]
- McSPI Register Description: [1] [2] [3] [4] [5] [6] [7] [8] [9] [10] [11] [12] [13] [14]

Table 19-38. MCSPI_MODULCTRL

Address Offset	0x28		
Physical Address	0x4809 8028	**Instance**	MCSPI1
	0x4809 A028		MCSPI2
	0x480B 8028		MCSPI3
	0x480B A028		MCSPI4
Description	This register is dedicated to the configuration of the serial port interface.		
Type	RW		
Write Latency	Immediate		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																												SYSTEM_TEST	MS	Reserved	SINGLE

Bits	Field Name	Description	Type	Reset
31:4	Reserved	Reads returns 0	RW	0x0000000
3	SYSTEM_TEST	Enables the system test mode	RW	0
		0x0: Functional mode		
		0x1: System test mode (SYSTEST)		
2	MS	Master/ Slave	RW	1
		0x0: Master - The module generates the spim_clk and spim_cs for channel x		
		0x1: Slave - The module receive		
1	Reserved	(returns 0 after writing 0) (returns 1 after writing 1)	RW	0
0	SINGLE	Single forced channel/multichannel (master mode only)	RW	0
		0x0: One or more channels will be used in master mode with automatic chip-select generation.		
		0x1: Only one channel will be used in master mode and the chip-select is driven by the MCSPI_CHxCONF[20] FORCE bit. This bit must be set in force spim_cs mode.		

Table 19-39. Register Call Summary for Register MCSPI_MODULCTRL

McSPI Functional Description
- Single-Channel Master Mode: [0] [1] [2] [3]
- Chip-Select Timing Control: [4]
- Slave Mode: [5]

McSPI Basic Programming Model
- McSPI Configuration and Operations Example: [6] [7]

McSPI Use Cases and Tips
- Programming Flow: [8] [9] [10] [11]

McSPI Register Manual
- McSPI Register Summary: [12]
- McSPI Register Description: [13]

Table 19-40. MCSPI_CHxCONF

Address Offset	0x2C + (0x14 * x)	**Index**	x= 0 to 3 for MCSPI1. x= 0 to 1 for MCSPI2 and MCSPI3. x= 0 for MCSPI4.
Physical Address	0x4809 802C + (0x14 * x)	**Instance**	MCSPI1
	0x4809 A02C + (0x14 * x)		MCSPI2
	0x480B 802C + (0x14 * x)		MCSPI3
	0x480B A02C + (0x14 * x)		MCSPI4
Description	This register is dedicated to the configuration of the channel x.		
Type	RW		
Write Latency	Immediate		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved		CLKG	FFER	FFEW	TCS		SBPOL	SBE	Reserved		FORCE	TURBO	IS	DPE1	DPE0	DMAR	DMAW	TRM		WL				EPOL	CLKD					POL	PHA

Bits	Field Name	Description	Type	Reset
31:30	Reserved	Read returns 0s.	RW	0x00
29	CLKG	Clock divider granularity. This register defines the granularity of channel clock divider: power of two or one clock cycle granularity. When this bit is set, the MCSPI_CHxCTRL[15:8] EXTCLK bit field must be configured to reach a maximum of 4096 clock divider ratio. Then the clock divider ratio is a concatenation of MCSPI_CHxCONF[5:2] CLKD and EXTCLK values. 0x0: Clock granularity of power of two 0x1: One clock cycle ganularity	RW	0x0
28	FFER	FIFO enabled for Receive. Only one channel can have this bit field set. 0x0: The buffer is not used to Receive data 0x1: The buffer is used to Receive data	RW	0x0
27	FFEW	FIFO enabled for Transmit. Only one channel can have this bit field set. 0x0: The buffer is not used to Transmit data 0x1: The buffer is used to Transmit data	RW	0x0
26:25	TCS	Chip select time control Defines the number of interface clock cycles between CS toggling and first (or last) edge of SPI clock. 0x0: 0.5 clock cycle 0x1: 1.5 clock cycles 0x2: 2.5 clock cycles 0x3: 3.5 clock cycles	RW	0x0
24	SBPOL	Start bit polarity 0x0: Start bit polarity is held to 0 during SPI transfer. 0x1: Start bit polarity is held to 1 during SPI transfer.	RW	0x0
23	SBE	Start bit enable for SPI transfer 0x0: Default SPI transfer length as specified by WL bit field 0x1: Start bit D/CX added before SPI transfer. Polarity is defined by MCSPI_CHxCONF[SBPOL].	RW	0x0
22:21	Reserved	Write the reset value. Read returns the reset value.	RW	0x0
20	FORCE	Manual spim_csx assertion to keep spim_csx for channel x active between SPI words (single channel master mode only). The MCSPI_MODULCTRL[0] SINGLE bit must bit set to 1.	RW	0x0

Bits	Field Name	Description	Type	Reset
		0x0: Writing 0 into this bit drives the spin_csx line low for channel x when MCSPI_CHxCONF [6] EPOL = 0, and drives it high when MCSPI_CHxCONF[6] EPOL = 1. 0x1: Writing 1 into this bit drives the spim_csx line high for channel x when MCSPI_CHxCONF[E6] EPOL = 0, and drives it low when MCSPI_CHxCONF[6] EPOL = 1.		
19	TURBO	Turbo mode 0x0: Turbo is deactivated (recommended for single SPI word transfer) 0x1: Turbo is activated to maximize the throughput for multi-SPI word transfers.	RW	0x0
18	IS	Input select 0x0: Data Line 0 (spim_somi) selected for reception 0x1: Data Line 1 (spim_simo) selected for reception	RW	0x1
17	DPE1	Transmission enable for data line 1 (spim_simo) 0x0: Data Line 1 (spim_simo) selected for transmission 0x1: No transmission on data Line 1 (spim_simo)	RW	0x1
16	DPE0	Transmission enable for data line 0 (spim_somi) 0x0: Data Line 0 (spim_somi) selected for transmission 0x1: No transmission on data Line 0 (spim_somi)	RW	0x0
15	DMAR	DMA Read request The DMA Read request line is asserted when the channel is enabled and a new data is available in the receive register of the channel. The DMA Read request line is deasserted on read completion of the receive register of the channel. 0x0: DMA read request disabled 0x1: DMA read request enabled	RW	0x0
14	DMAW	DMA Write request. The DMA write request line is asserted when the channel is enabled and the MCSPI_TXx register of the channel is empty. The DMA write request line is deasserted on load completion of the MCSPI_TXx register of the channel. 0x0: DMA write request disabled 0x1: DMA write request enabled	RW	0x0
13:12	TRM	Transmit/receive modes 0x0: Transmit and receive mode 0x1: Receive-only mode 0x2: Transmit-only mode 0x3: Reserved	RW	0x0
11:7	WL	SPI word length 0x0: Reserved 0x1: Reserved 0x2: Reserved 0x3: The SPI word is 4-bit long 0x4: The SPI word is 5-bit long 0x5: The SPI word is 6-bit long 0x6: The SPI word is 7-bit long 0x7: The SPI word is 8-bit long 0x8: The SPI word is 9-bit long 0x9: The SPI word is 10-bit long 0xA: The SPI word is 11-bit long	RW	0x00

Bits	Field Name	Description	Type	Reset
		0xB: The SPI word is 12-bit long		
		0xC: The SPI word is 13-bit long		
		0xD: The SPI word is 14-bit long		
		0xE: The SPI word is 15-bit long		
		0xF: The SPI word is 16-bit long		
		0x10: The SPI word is 17-bit long		
		0x11: The SPI word is 18-bit long		
		0x12: The SPI word is 19-bit long		
		0x13: The SPI word is 20-bit long		
		0x14: The SPI word is 21-bit long		
		0x15: The SPI word is 22-bit long		
		0x16: The SPI word is 23-bit long		
		0x17: The SPI word is 24-bit long		
		0x18: The SPI word is 25-bit long		
		0x19: The SPI word is 26-bit long		
		0x1A: The SPI word is 27-bit long		
		0x1B: The SPI word is 28-bit long		
		0x1C: The SPI word is 29-bit long		
		0x1D: The SPI word is 30-bit long		
		0x1E: The SPI word is 31-bit long		
		0x1F: The SPI word is 32-bit long		
6	EPOL	spim_csx polarity for channel x	RW	0x0
		0x0: spim_csx is held high during the active state.		
		0x1: spim_csx is held low during the active state.		
5:2	CLKD	Frequency divider for spim_clk (for master device only) A programmable clock divider divides the SPI reference clock (CLKSPIREF) by a 4-bit value and results in a new spim_clk clock available to shift data in and out.	RW	0x0
		0x0: 1		
		0x1: 2		
		0x2: 4		
		0x3: 8		
		0x4: 16		
		0x5: 32		
		0x6: 64		
		0x7: 128		
		0x8: 256		
		0x9: 512		
		0xA: 1024		
		0xB: 2048		
		0xC: 4096		
		0xD: Division not supported		
		0xE: Division not supported		
		0xF: Division not supported		
1	POL	spim_clk polarity	RW	0x0
		0x0: spim_clk is held high during the active state.		
		0x1: spim_clk is held low during the active state.		
0	PHA	spim_clk phase	RW	0x0
		0x0: Data are latched on odd-numbered edges of spim_clk.		
		0x1: Data are latched on even-numbered edges of spim_clk.		

Table 19-41. Register Call Summary for Register MCSPI_CHxCONF

McSPI Functional Interface
- Multichannel SPI Protocol and Data Format: [0] [1] [2] [3] [4] [5] [6] [7]
- Transfer Format: [8]

McSPI Functional Description
- Master Transmit-and-Receive Mode (Full Duplex): [9]
- Master Transmit-Only Mode (Half Duplex): [10]
- Master Receive-Only Mode (Half Duplex): [11]
- Single-Channel Master Mode: [12] [13] [14] [15] [16] [17] [18] [19] [20] [21] [22] [23]
- Start Bit Mode: [24] [25] [26] [27]
- Chip-Select Timing Control: [28]
- Programmable SPI Clock (spim_clk): [29] [30] [31] [32] [33] [34] [35]
- Dedicated Resources: [36] [37] [38] [39] [40]
- Slave Transmit-and-Receive Mode: [41]
- Slave Transmit-Only Mode: [42]
- Slave Receive-Only Mode: [43]
- FIFO Buffer Management: [44] [45] [46]
- Buffer Almost Full: [47] [48]
- Buffer Almost Empty: [49] [50]
- Interrupt Events in Master Mode: [51] [52]
- Interrupt Events in Slave Mode: [53] [54]
- DMA Requests: [55] [56]
- Idle Mode: [57]

McSPI Basic Programming Model
- Transfer Procedures without FIFO: [58]
- McSPI Configuration and Operations Example: [59] [60] [61] [62] [63] [64] [65] [66] [67] [68]
- Transfer Procedures with FIFO: [69]

McSPI Use Cases and Tips
- Programming Flow: [70] [71] [72] [73] [74] [75] [76] [77]

McSPI Register Manual
- McSPI Register Summary: [78]
- McSPI Register Description: [79] [80] [81] [82] [83] [84] [85] [86] [87] [88] [89] [90]

Table 19-42. MCSPI_CHxSTAT

Address Offset	0x30 + (0x14 * x)	**Index**	x= 0 to 3 for MCSPI1. x= 0 to 1 for MCSPI2 and MCSPI3. x= 0 for MCSPI4.
Physical Address	0x4809 8030 + (0x14 * x)	**Instance**	MCSPI1
	0x4809 A030 + (0x14 * x)		MCSPI2
	0x480B 8030 + (0x14 * x)		MCSPI3
	0x480B A030 + (0x14 * x)		MCSPI4
Description	This register provides status information about MCSPI_TXx and MCSPI_RXx registers of channel x.		
Type	R		
Write Latency	Immediate		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																								RXFFF	RXFFE	TXFFF	TXFFE	EOT	TXS	RXS	

Bits	Field Name	Description	Type	Reset
31:7	Reserved	Read returns 0s.	R	0x00000000
6	RXFFF	Channel x FIFO Receive Buffer Full Status	R	0x0
		Read 0x0: FIFO Receive Buffer is not full		

Bits	Field Name	Description	Type	Reset
		Read 0x1: FIFO Receive Buffer is full		
5	RXFFE	Channel x FIFO Receive Buffer Empty Status	R	0x0
		Read 0x0: FIFO Receive Buffer is not empty		
		Read 0x1: FIFO Receive Buffer is empty		
4	TXFFF	Channel x FIFO Transmit Buffer Full Status	R	0x0
		Read 0x0: FIFO Transmit Buffer is not full		
		Read 0x1: FIFO Transmit Buffer is full		
3	TXFFE	Channel x FIFO Transmit Buffer Empty Status	R	0x0
		Read 0x0: FIFO Transmit Buffer is not empty		
		Read 0x1: FIFO Transmit Buffer is empty		
2	EOT	Channel x end-of-transfer status. The definitions of beginning and end of transfer vary with master versus slave and the transfer format (transmit/receive mode, turbo mode). See dedicated chapters for details.	R	0x0
		Read 0x0: This flag is automatically cleared when the shift register is loaded with the data from the MCSPI_TXx register (beginning of transfer).		
		Read 0x1: This flag is automatically set to one at the end of an SPI transfer.		
1	TXS	Channel x MCSPI_TXx register status	R	0x0
		Read 0x0: Register is full.		
		Read 0x1: Register is empty.		
0	RXS	Channel x MCSPI_RXx register status	R	0x0
		Read 0x0: Register is empty.		
		Read 0x1: Register is full.		

Table 19-43. Register Call Summary for Register MCSPI_CHxSTAT

McSPI Functional Description

- Master Transmit-and-Receive Mode (Full Duplex): [0] [1] [2]
- Master Transmit-Only Mode (Half Duplex): [3]
- Master Receive-Only Mode (Half Duplex): [4]
- Single-Channel Master Mode: [5] [6] [7] [8] [9]
- Dedicated Resources: [10] [11]
- Slave Transmit-and-Receive Mode: [12]
- Slave Transmit-Only Mode: [13]
- Slave Receive-Only Mode: [14]
- End of Transfer Management: [15]

McSPI Use Cases and Tips

- Programming Flow: [16] [17] [18] [19] [20] [21] [22] [23]

McSPI Register Manual

- McSPI Register Summary: [24]

Table 19-44. MCSPI_CHxCTRL

Address Offset	0x34 + (0x14 * x)	**Index**	x= 0 to 3 for MCSPI1. x= 0 to 1 for MCSPI2 and MCSPI3. x= 0 for MCSPI4.
Physical Address	0x4809 8034 + (0x14 * x)	**Instance**	MCSPI1
	0x4809 A034 + (0x14 * x)		MCSPI2
	0x480B 8034 + (0x14 * x)		MCSPI3
	0x480B A034 + (0x14 * x)		MCSPI4
Description	This register is dedicated to enable the channel x		
Type	RW		
Write Latency	Immediate		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																EXTCLK								Reserved							EN

Bits	Field Name	Description		Type	Reset
31:16	Reserved	Read returns 0s.		RW	0x0000
15:8	EXTCLK	Clock ratio extension: This register is used to concatenate with MCSPI_CHxCONF[5:2] CLKD bit field for clock ratio only when granularity is one clock cycle (MCSPI_CHxCONF[28] CLKG bit set to 1). Then the max value reached is 4096 clock divider ratio.		RW	0x00
		0x0:	Clock ratio is CLKD + 1		
		0x1:	Clock ratio is CLKD + 1 + 16		
		0xFF:	Clock ratio is CLKD + 1 + 4080		
7:1	Reserved	Read returns 0s.		RW	0x00
0	EN	Channel enable		RW	0x0
		0x0:	Channel x is not active.		
		0x1:	Channel x is active.		

Table 19-45. Register Call Summary for Register MCSPI_CHxCTRL

McSPI Functional Description
- Master Transmit-and-Receive Mode (Full Duplex): [0]
- Single-Channel Master Mode: [1] [2]
- Programmable SPI Clock (spim_clk): [3] [4]
- Dedicated Resources: [5] [6]

McSPI Basic Programming Model
- McSPI Configuration and Operations Example: [7] [8]

McSPI Use Cases and Tips
- Programming Flow: [9] [10] [11] [12] [13] [14] [15] [16]

McSPI Register Manual
- McSPI Register Summary: [17]
- McSPI Register Description: [18]

Table 19-46. MCSPI_TXx

Address Offset	0x38 + (0x14 * x)	**Index**	x= 0 to 3 for MCSPI1. x= 0 to 1 for MCSPI2 and MCSPI3. x= 0 for MCSPI4.
Physical Address	0x4809 8038 + (0x14 * x)	**Instance**	MCSPI1
	0x4809 A038 + (0x14 * x)		MCSPI2
	0x480B 8038 + (0x14 * x)		MCSPI3
	0x480B A038 + (0x14 * x)		MCSPI4
Description	This register contains a single SPI word to transmit on the serial link, whatever SPI word length is.		
Type	RW		
Write Latency	Immediate		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TDATA																															

Bits	Field Name	Description	Type	Reset
31:0	TDATA	Channel 0 Data to transmit	RW	0x00000000

Table 19-47. Register Call Summary for Register MCSPI_TXx

McSPI Functional Description
- Master Transmit-and-Receive Mode (Full Duplex): [0] [1] [2] [3]
- Master Transmit-Only Mode (Half Duplex): [4]
- Master Receive-Only Mode (Half Duplex): [5] [6] [7] [8] [9]
- Single-Channel Master Mode: [10]
- Dedicated Resources: [11] [12] [13] [14] [15]
- Slave Transmit-and-Receive Mode: [16] [17]
- Slave Receive-Only Mode: [18] [19] [20]
- Interrupt Events in Master Mode: [21] [22] [23] [24] [25] [26] [27]
- Interrupt Events in Slave Mode: [28] [29] [30] [31] [32] [33]
- Interrupt-Driven Operation: [34]
- Polling: [35]
- DMA Requests: [36] [37]

McSPI Basic Programming Model
- McSPI Configuration and Operations Example: [38]
- Common Transfer Procedure: [39] [40]

McSPI Use Cases and Tips
- Programming Flow: [41] [42] [43] [44] [45] [46] [47]

McSPI Register Manual
- McSPI Register Summary: [48]
- McSPI Register Description: [49] [50] [51] [52] [53]

Table 19-48. MCSPI_RXx

Address Offset	0x3C + (0x14 * x)	**Index**	x= 0 to 3 for MCSPI1. x= 0 to 1 for MCSPI2 and MCSPI3. x= 0 for MCSPI4.
Physical Address	0x4809 803C + (0x14 * x)	**Instance**	MCSPI1
	0x4809 A03C + (0x14 * x)		MCSPI2
	0x480B 803C + (0x14 * x)		MCSPI3
	0x480B A03C + (0x14 * x)		MCSPI4
Description	This register contains a single SPI word received through the serial link, what ever SPI word length is.		
Type	R		
Write Latency	Immediate		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RDATA																															

Bits	Field Name	Description	Type	Reset
31:0	RDATA	Channel 0 Received Data	R	0x00000000

Table 19-49. Register Call Summary for Register MCSPI_RXx

McSPI Functional Description
- Master Transmit-and-Receive Mode (Full Duplex): [0] [1]
- Master Transmit-Only Mode (Half Duplex): [2] [3] [4]
- Master Receive-Only Mode (Half Duplex): [5] [6]
- Single-Channel Master Mode: [7] [8] [9] [10]
- Dedicated Resources: [11] [12] [13]
- Slave Transmit-and-Receive Mode: [14]
- Slave Transmit-Only Mode: [15] [16]
- End of Transfer Management: [17]
- Interrupt Events in Master Mode: [18] [19] [20] [21]
- Interrupt Events in Slave Mode: [22] [23] [24] [25] [26] [27]
- Interrupt-Driven Operation: [28]
- Polling: [29]
- DMA Requests: [30]
- Idle Mode: [31]

McSPI Basic Programming Model
- McSPI Configuration and Operations Example: [32]

McSPI Use Cases and Tips
- Programming Flow: [33] [34] [35] [36] [37] [38]

McSPI Register Manual
- McSPI Register Summary: [39]
- McSPI Register Description: [40] [41]

Table 19-50. MCSPI_XFERLEVEL

Address Offset	0x7C		
Physical Address	0x4809 807C	**Instance**	MCSPI1
	0x4809 A07C		MCSPI2
	0x480B 807C		MCSPI3
	0x480B A07C		MCSPI4
Description	This register provides transfer levels needed while using FIFO buffer during transfer.		
Type	RW		
Write Latency	Immediate		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
WCNT																Reserved		AFL						Reserved		AEL					

Bits	Field Name	Description	Type	Reset
31:16	WCNT	Spi word counter: This register holds the programmable value of number of SPI word to be transferred on channel which is using the FIFO buffer. When transfer had started, a read back in this register returns the current SPI word transfer index. 0x0: Counter not used 0x1: One spi word 0xFFFE : 65534 spi word 0xFFFF : 65535 spi word	RW	0x0000
15:14	Reserved	Read returns 0s.	RW	0x0
13:8	AFL	Buffer Almost Full: This register holds the programmable almost full level value used to determine almost full buffer condition. If the user wants an interrupt or a DMA read request to be issued during a receive operation when the data buffer holds at least I bytes, then this bit field must be set to I-1. 0x0: One byte 0x1: 2 bytes 0x3E: 63 bytes 0x3F: 64 bytes	RW	0x00
7:6	Reserved	Read returns 0s.	RW	0x0
5:0	AEL	Buffer Almost Empty: This register holds the programmable almost empty level value used to determine almost empty buffer condition. If the user wants an interrupt or a DMA write request to be issued during a transmit operation when the data buffer is able to receive I bytes, then this bit field must be set to I-1. 0x0: One byte 0x1: 2 bytes 0x3E: 63 bytes 0x3F: 64 bytes	RW	0x00

Table 19-51. Register Call Summary for Register MCSPI_XFERLEVEL

McSPI Functional Description • FIFO Buffer Management: [0] [1] • Buffer Almost Full: [2] • Buffer Almost Empty: [3] • End of Transfer Management: [4] [5] • Interrupt Events in Master Mode: [6] [7] [8] [9] [10] • Interrupt Events in Slave Mode: [11] [12] [13] [14]
McSPI Basic Programming Model • Common Transfer Procedure: [15]
McSPI Register Manual • McSPI Register Summary: [16] • McSPI Register Description: [17]

HDQ/1-Wire

This chapter describes the features and functions of the HDQ/1-Wire module for the OMAP34xx multimedia device.

Topic **Page**

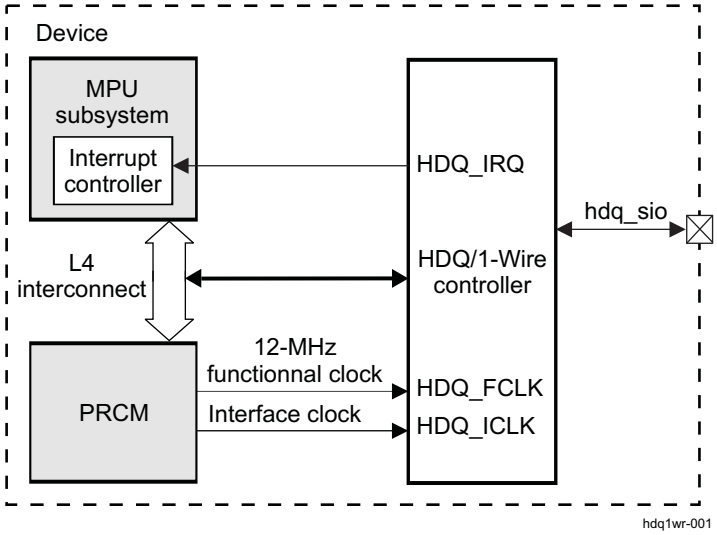
20.1 HDQ/1-Wire Overview 2894
20.2 HDQ/1-Wire Environment 2895
20.3 HDQ/1-Wire Integration 2898
20.4 HDQ/1-Wire Functional Description 2900
20.5 HDQ/1-Wire Basic Programming Model 2906
20.6 HDQ/1-Wire Use Cases and Tips 2909
20.7 HDQ/1-Wire Register Manual 2913

20.1 HDQ/1-Wire Overview

The HDQ/1-Wire module implements the hardware protocol of the master functions of the Benchmarq HDQ and the Dallas Semiconductor 1-Wire® protocols. These protocols use a single wire for communication between the master (HDQ/1-Wire controller) and the slave (HDQ/1-Wire external compliant device).

Figure 20-1 shows the HDQ/1-Wire controller module.

Figure 20-1. HDQ/1-Wire Highlight



The HDQ and 1-Wire module has a generic L4 interface and is intended to be used in an interrupt-driven fashion. The one-pin interface is implemented as an open-drain output at the device level.

The HDQ operates from a fixed 12-MHz functional clock provided by the PRCM module.

Only the MPU subsystem uses the HDQ/1-Wire module.

The main features of the HDQ/1-Wire module support the following:

- Benchmarq HDQ protocol
- Dallas Semiconductor 1-Wire protocol
- Power-down mode

The HDQ/1-Wire module provides a communication rate of 5K bits/s over an address space of 128 bytes.

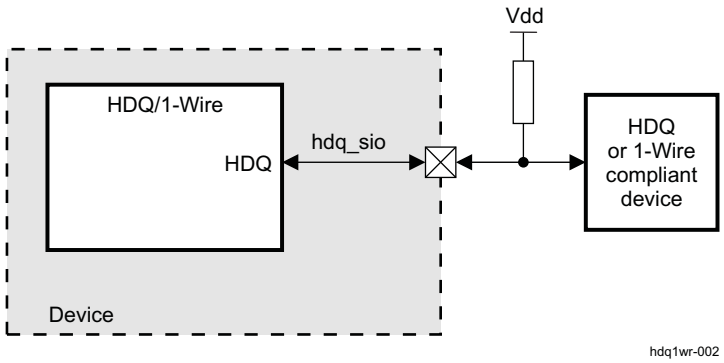
A typical application of the HDQ/1-Wire module is the communication with battery monitor (gas gauge) integrated circuits.

20.2 HDQ/1-Wire Environment

20.2.1 HDQ/1-Wire Functional Interface

Figure 20-2 shows a typical application using the HDQ/1-Wire connection.

Figure 20-2. HDQ/1-Wire Typical Application System Overview



An external pullup is required, because the two protocols use a return-to-1 mechanism (that is, after any command, the line is pulled to a logical high level).

The HDQ/1-Wire module operates according to a command structure that is programmed into transmit command registers (as described in Section 20.5).

The 1-Wire mode runs at slower speeds than the capabilities of the mode.

Table 20-1 describes the external pullup signal from an HDQ or 1-Wire compliant device.

Table 20-1. I/O Description

Signal Name	I/O	Description	Value at Reset
hdq_sio	Bidir	Serial data input/output	1

20.2.2 HDQ and 1-Wire (SDQ) Protocols

20.2.2.1 HDQ Protocol Initialization (Default)

In HDQ mode, the firmware does not require the host to create an initialization pulse to the slave. However, the slave can be reset by using an initialization pulse (also referred to as a break pulse). The initialization pulse is generated by setting the INITIALIZATION bit (HDQ.HDQ_CTRL_STATUS[2]). The slave does not respond with a presence pulse as it does in the 1-Wire protocol.

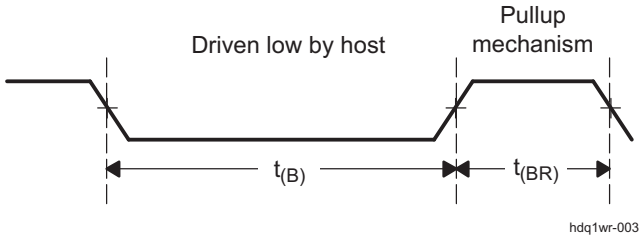
The HDQ is a command-based protocol in which the host sends a command byte to the slave. The command directs the slave either to store the next eight bits of data received to a register specified by the command byte (write operation) or to output the eight bits of data from a register specified by the command byte (read operation). The master implementation is a simple byte engine. The sending of the ID, command/address, and data is controlled by firmware. The master engine provides only a single HDQ.HDQ_TX_DATA register.

The command and data bytes consist of a stream of eight bits with a maximum transmission rate of 5K bits/s. The least significant bit (LSB) of a command or data byte is transmitted first. If a communication time-out occurs between the host and the slave (for example, if the host waits longer than the specified time for the slave to respond, or if this is the first access command), then the host must send an initialization pulse (BREAK pulse) before sending the command again.

The slave detects a break when the HDQ pin is driven to a logic-low state for a specified break time t(B) or greater. The HDQ pin then returns to its normal ready-high logic state for a specified break-recovery time t(BR). The slave is then ready for a command from the host processor. Figure 20-3 shows this behavior.

An interrupt condition indicates either a TX-complete, an RX-complete, or a time-out condition. Reading the interrupt status register clears all interrupt conditions. Only one interrupt signal is sent to the microcontroller, and only one overall mask bit can enable or disable the interrupt. The interrupt conditions cannot be individually masked.

Figure 20-3. HDQ Break-Pulse Timing Diagram

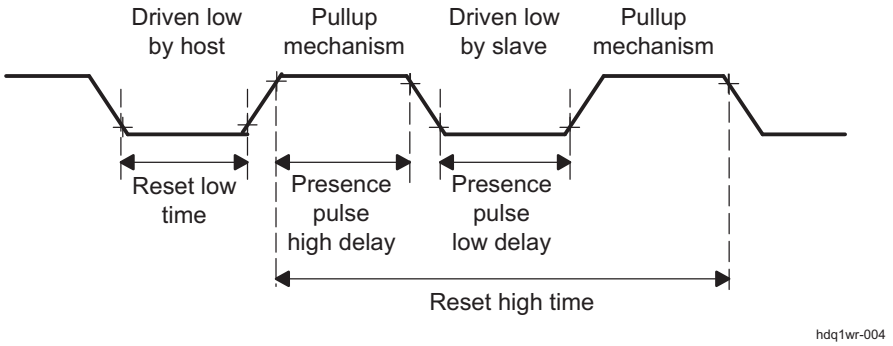


20.2.2.2 1-Wire (SDQ) Protocol Initialization

In 1-Wire (SDQ) protocol, the host first sends an initialization pulse (by pulling the line to a logic-low state) and then waits for the slave to respond with a presence pulse before enabling any communication sequence.

As for the initialization pulse, the presence pulse is a low-going edge on the line initiated by the slave. The timing diagram in Figure 20-4 shows the 1-Wire (SDQ) reset sequence.

Figure 20-4. 1-Wire (SDQ) Reset Timing Diagram



The host drives the line to a logic-low state for a minimum of reset low time. Once the slave detects this pulse, it must drive the line to a logic-low state within the presence pulse high delay for a minimum period of presence pulse low time.

If the slave does not respond within this interval of time, a time-out event occurs and no transaction can be initiated. The host must initiate the reset sequence again before sending any command to the slave.

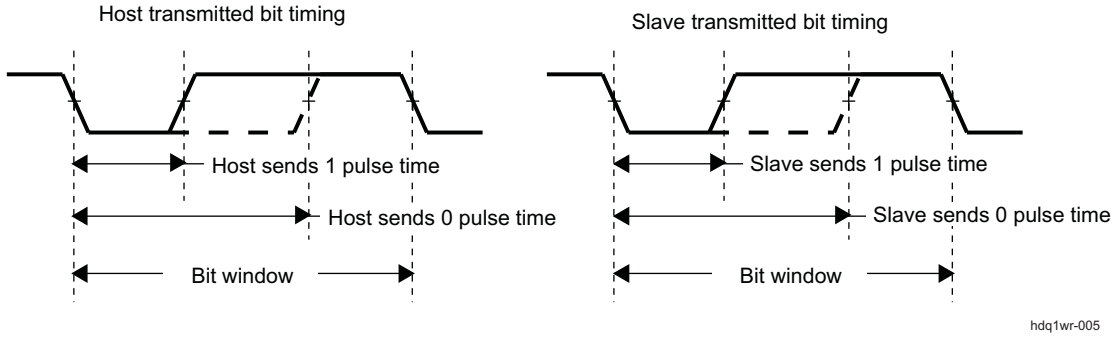
On the other hand, if the slave sends back its presence pulse within the specified interval of time, the communication can be enabled after the reset high time.

20.2.2.3 Communication Sequence (HDQ and 1-Wire Protocols)

This section describes the part common to both protocols.

After a successful break pulse (HDQ mode) or initialization sequence (1-Wire protocol), the host and slave are ready for bit transmission. Each bit to transmit (either from the host to the slave or from the slave to the host) is preceded by a low-going edge on the line, as shown in Figure 20-5.

Figure 20-5. HDQ/1-Wire Transmitted Bit Timing



The return-to-1 data-bit frame consists of three distinct sections. The first section starts the transmission when either the slave or the host takes the line to a logic-low state. The next section is the actual data transmission in which the data must be valid during a specified period of time after the negative edge that starts the communication. The final section stops the transmission by returning the HDQ/1-Wire line to a logic-high state. Communication with an HDQ/1-Wire slave always occurs with the LSB being transmitted first.

The command byte of the HDQ/1-Wire protocols consists of eight contiguous valid command bits. The command byte contains two fields: R/W command and address. The R/W bit of the command byte determines whether the command is a read or a write, and the address field containing bits AD6-AD0 indicates the address to be read or written. Table 20-2 lists the command byte values.

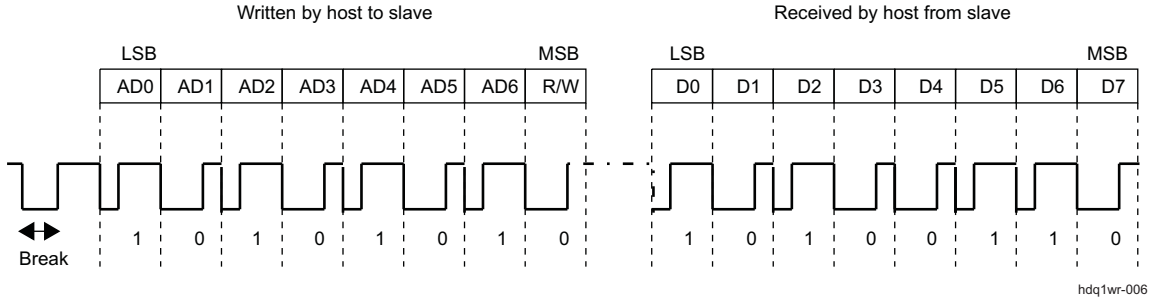
Table 20-2. HDQ/1-Wire Command Byte

7	6	5	4	3	2	1	0
R/W	AD6	AD5	AD4	AD3	AD2	AD1	AD0

R/W Indicates whether the command byte is a read or a write. A 1 indicates a write command; the following eight bits must be written to the register specified by the address field of the command byte. A 0 indicates that the command is a read. On a read command, the slave outputs the requested register contents.

AD6-AD0 Represent the seven bits labeled AD6-AD0 containing the address portion of the register to be accessed.
The communication sequence example in Figure 20-6 shows a read command at address 0x55; the received data is 0x65.

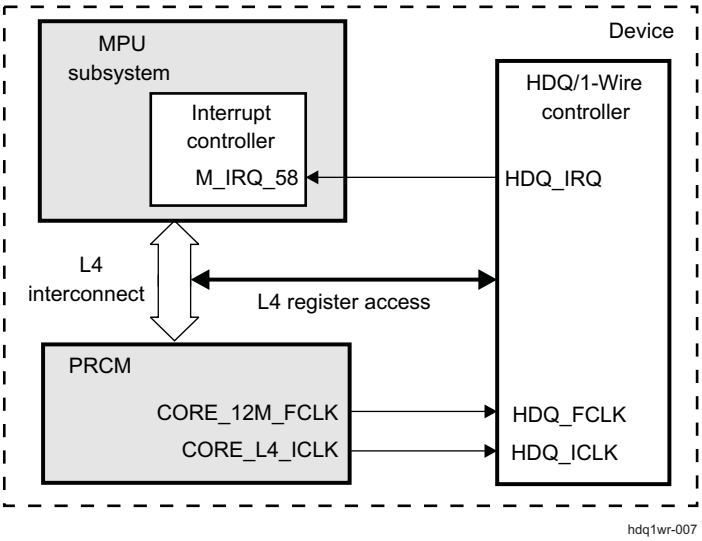
Figure 20-6. HDQ/1-Wire Communication Sequence



20.3 HDQ/1-Wire Integration

Figure 20-7 shows the HDQ/1-Wire module integration in the device.

Figure 20-7. HDQ/1-Wire Integration



20.3.1 Clocking, Reset, and Power Management Scheme

20.3.1.1 HDQ/1-Wire Clocks

There are two clock domains in the HDQ/1-Wire module: the functional clock domain and the interface clock domain.

- HDQ_FCLK belongs to the functional clock domain. It is a fixed 12-MHz clock provided by the PRCM. It is used to clock the internal module logic.

  The HDQ_FCLK source is the CORE_12M_FCLK PRCM output, which is derived from the CORE_48M_FCLK clock signal (a 1/4 ratio is applied). The CORE_12M_FCLK clock is issued from the peripherals DPLL4. For more information about the clock, see Chapter 4, *Power, Reset, and Clock Management*.

  When the HDQ/1-Wire module no longer requires the HDQ_FCLK, the software can disable it at the PRCM level by setting the EN_HDQ bit (PRCM.CM_FCLKEN1_CORE[22]) in the PRCM registers. The clock is effectively cut, provided the other modules that receive it do not require it either.

  For details about the PRCM register settings and DPLL4 configuration, see Chapter 4, *Power, Reset, and Clock Management*.

- HDQ_ICLK is the interface clock. It runs at L4 interconnect clock speed and is used to trigger access to the HDQ/1-Wire L4 interface. Its source is the PRCM CORE_L4_ICLK signal. It typically runs at L3/2 frequency.

  When the HDQ/1-Wire module no longer requires the HDQ_ICLK, the software can disable it at the PRCM level by setting the EN_HDQ bit (PRCM.CM_ICLKEN1_CORE[22]) in the PRCM registers. The clock is effectively cut, provided the other modules that receive it do not require it either.

  It is also possible to activate an autoidle mode for this clock (AUTO_HDQ bit PRCM.CM_AUTOIDLE1_CORE[22] set to 1). In this case, HDQ_ICLK follows the CORE clock domain behavior. For more information, see Chapter 4, *Power, Reset, and Clock Management*.

20.3.1.2 HDQ/1-Wire Reset Scheme

Global reset of the module is done either by activating the CORE_RST signal in the CORE_RST domain (for more information, see Chapter 4, *Power, Reset, and Clock Management*) or by setting to 1 the SOFTRESET bit HDQ.HDQ_SYSCONFIG[1]. Setting this bit enables an active software reset functionality equivalent to a hardware reset.

20.3.1.3 HDQ/1-Wire Power Domain

The HDQ/1-Wire belongs to the CORE power domain. For more information about the CORE power domain, see Chapter 4, *Power, Reset, and Clock Management*.

20.3.2 Hardware Requests

The HDQ/1-Wire can generate one interrupt:

- HDQ_IRQ: This is an interrupt to the MPU subsystem interrupt controller. It is mapped on M_IRQ_58.

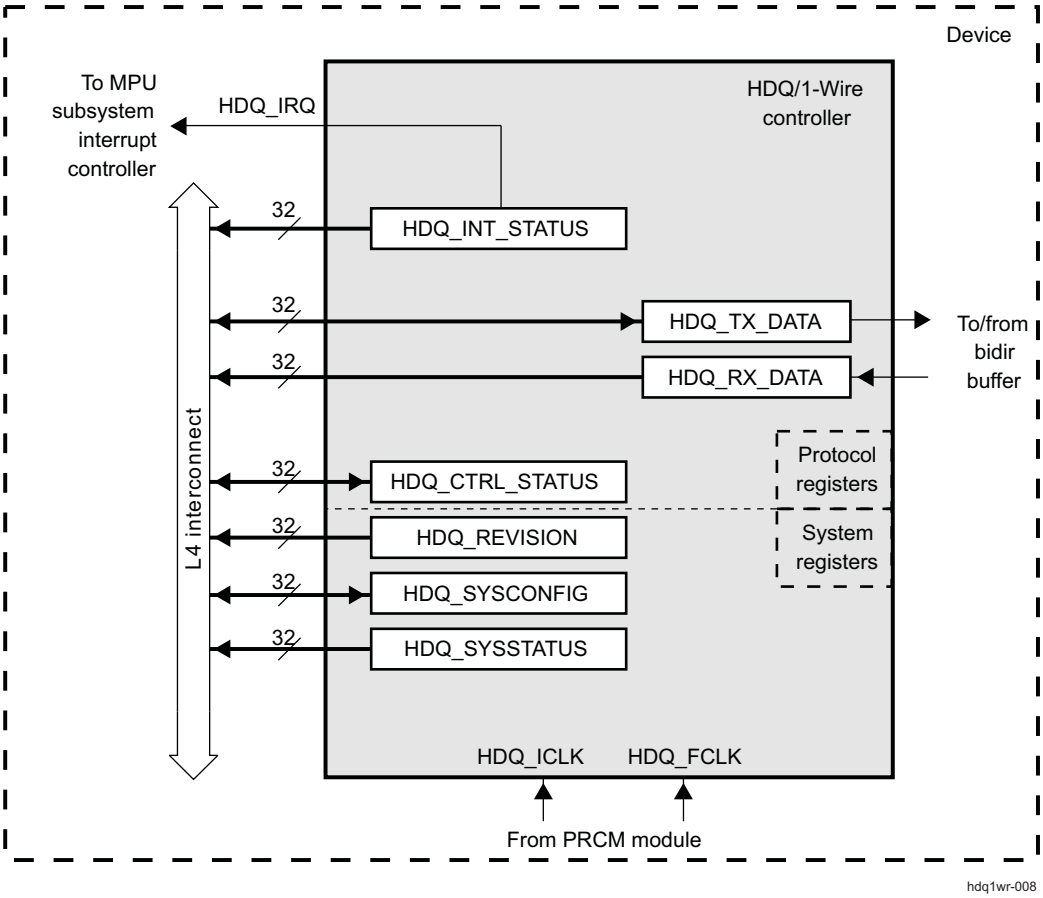
20.4 HDQ/1-Wire Functional Description

The HDQ/1-Wire module works with both the HDQ and 1-Wire protocols. The protocols use a single wire to establish communication between the master and the slave. Both protocols use a return-to-1 mechanism; that is, after any command is driven, the line is pulled to a high level. This mechanism requires an external pullup.

20.4.1 HDQ/1-Wire Block Diagram

Figure 20-8 shows the HDQ/1-Wire block diagram.

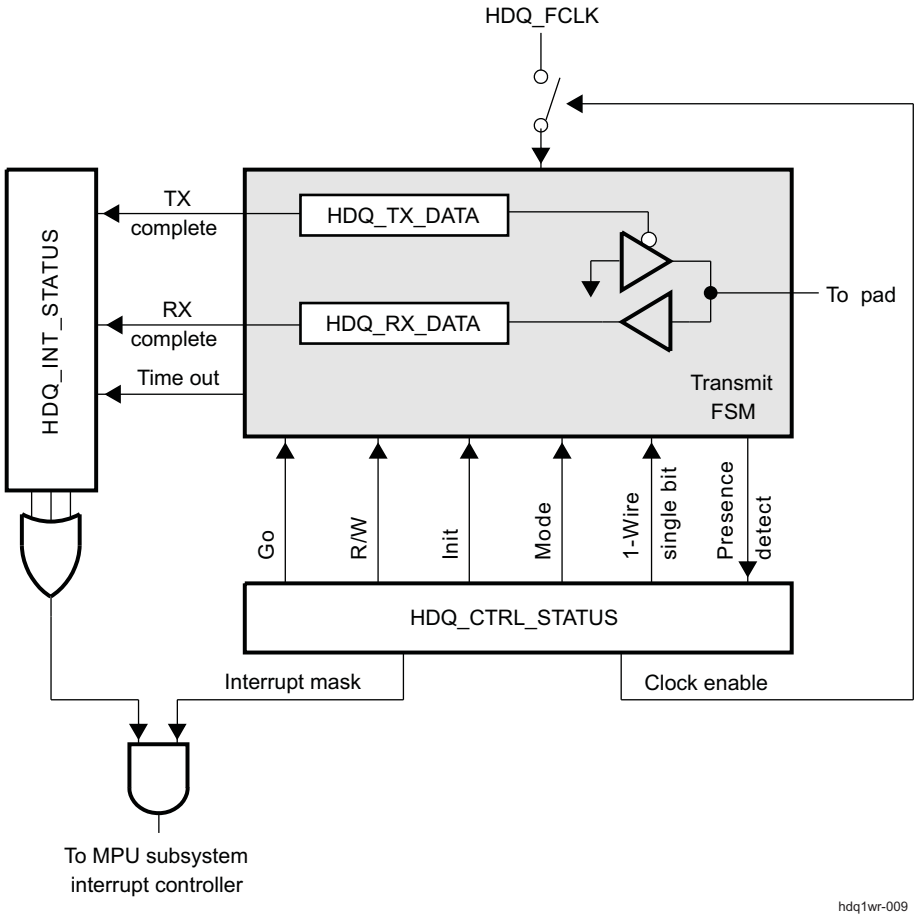
Figure 20-8. HDQ/1-Wire Block Diagram



The MODE bit HDQ.HDQ_CTRL_STATUS[0] allows selection between the HDQ and 1-Wire protocols. This bit is assumed static for design purposes. The configuration is in HDQ mode by default.

Figure 20-9 shows the protocol-dedicated register scheme.

Figure 20-9. Protocol Registers Description



The receive and transmit operations are performed with respect to the HDQ protocol timing. The module is implemented once in the OMAP device and is clocked with a single clock whether it runs HDQ or 1-Wire protocol. Although the data exchange is slower than the capabilities of the 1-Wire mode, this mode still meets the timing requirements and practical considerations. That is, the 1-Wire protocol runs at the HDQ protocol speed, which is slower (5K bits/s). The timing parameters and protocol are different in the two modes.

20.4.2 HDQ Mode (Default)

20.4.2.1 HDQ Mode Features

The HDQ mode supports the following:

- Benchmark HDQ protocol
- Power-down mode

20.4.2.2 Description

In the HDQ mode, there is no need for the host to create an initialization pulse to the slave. However, the host can reset the slave by using an initialization pulse (also known as a break pulse). Setting the INITIALIZATION bit HDQ.HDQ_CTRL_STATUS[2] creates this pulse by pulling the line down. When the slave receives the pulse, it is ready for communication but does not respond with a presence pulse.

In a typical write operation, two bytes are sent to the slave. The first byte corresponds to the command/address byte, and the second byte corresponds to the data to be written.

In a typical read operation, the host sends a command/address byte and the slave returns a byte of data.

The master is implemented to send and receive bytes. Sending the command/address and data is controlled by the firmware. The master provides only a single data TX register.

The HDQ protocol is a return-to-1 protocol. Consequently, after a byte is sent to the slave (either command/address + data for a write, or just command/address for a read), the host pulls the line up. The line is set to the high-impedance state in the device and an external pullup brings it to a logical high level.

In the case of a read operation, the slave also drives the line to a logic-low state before sending the requested data.

If the host initiates a read and does not receive data within a specified interval of time (that is, the slave does not drive the line low within this interval), the TIMEOUT bit HDQ.HDQ_INT_STATUS[0] is set, thereby indicating a read failure. The TIMEOUT bit remains set until the host reads the interrupt status register (HDQ.HDQ_INT_STATUS).

An interrupt condition indicates either a TX-complete, an RX-complete, or a time-out on a transaction. The corresponding bit is set in the interrupt status register (HDQ.HDQ_INT_STATUS). This register is cleared as soon as it is read.

Only one interrupt signal is sent to the MPU, and only an overall mask can enable or disable the interrupts. These interrupts cannot be individually masked.

20.4.2.3 Single-Bit Mode

In HDQ mode, the single-bit mode (1_WIRE_SINGLE_BIT bit HDQ.HDQ_CTRL_STATUS[7] set to 1) has no effect because the HDQ protocol supports only byte transfers.

20.4.2.4 Interrupt Conditions

The HDQ/1-Wire module provides the following interrupt status:

1. Transmission complete:

   A write operation of one byte was completed. Successful or failed completion is not indicated, because there is no acknowledgment from the slave in HDQ protocol. This interrupt condition is cleared by reading the interrupt status register (HDQ.HDQ_INT_STATUS).
2. Read complete:

   In HDQ mode, the interrupt status indicates that a byte has been successfully read. This interrupt condition is cleared by reading the interrupt status register (HDQ.HDQ_INT_STATUS).
3. Presence detect/time-out:

   In HDQ mode, the interrupt status indicates that after a read command initiated by the host, the slave did not pull the line low within the specified time. This interrupt condition is cleared by reading the interrupt status register (HDQ.HDQ_INT_STATUS).

   In HDQ mode, a time-out condition is also used to indicate the successful completion of a break pulse. That is, if the master has sent the break pulse, it is indicated with a time-out instead of a TX-complete.

Only one interrupt is generated to the MPU based on any of these interrupt conditions. A read operation on the interrupt status register clears all the interrupt status bits that were previously set.

20.4.3 1-Wire Mode

20.4.3.1 1-Wire Mode Features

The 1-Wire mode supports the following:

- Dallas Semiconductor 1-Wire protocol
- Power-down mode
- Single-bit mode

20.4.3.2 Description

The 1-Wire mode requires an initialization pulse to be sent to the slave(s) connected on the interface. If a slave is present, it responds with a presence pulse.

The initialization pulse is sent after INITIALIZATION bit HDQ.HDQ_CTRL_STATUS[2] is set. The firmware sends the initialization pulse depending on the value of this bit.

The slave presence on the line is detected by a presence bit in the control and status register. When the slave receives the initialization pulse, it sends back its presence pulse by pulling down the line. The module detects this low-going edge and sets the PRESENCEDETECT bit HDQ.HDQ_CTRL_STATUS[3].

In a similar way, if a presence pulse is not received from the slave after an initialization pulse is sent, the PRESENCEDETECT bit remains cleared.

Whether or not a presence pulse is detected after an initialization pulse is sent, the TIMEOUT bit HDQ.HDQ_INT_STATUS[0] is set and an interrupt condition is generated.

In 1-Wire mode, the generated interrupt condition means the maximum time allowed for receiving the response has elapsed and the software must check the PRESENCEDETECT bit to determine whether or not there was a presence pulse.

The INITIALIZATION bit is cleared at the end of the initialization pulse at the same time as the TIMEOUT bit is set. The TIMEOUT bit is cleared when the interrupt status register (HDQ.HDQ_INT_STATUS) is read.

For read operations, 1-Wire is a bit-by-bit protocol, which means the slave must be clocked by the host for each bit of the byte to read.

The line is pulled up at the end of the command/address byte. On the first read, the host creates a low-going edge to initiate a bit read. The line is then pulled up (pulled to the high-impedance state by the host and set to a high logical level by the external pullup) and the slave either drives the line low to transmit a 0 or does not drive the line to transmit a 1. This sequence is repeated for each bit to read.

The first bit the host receives is the LSB, and the last bit is the most significant bit (MSB) in the receive data register (HDQ.HDQ_RX_DATA).

An interrupt condition indicates either a TX-complete, an RX-complete, or a time-out condition (that is, the time allowed for the slave to indicate its presence has elapsed). A read operation on the interrupt status register clears the interrupt conditions previously set. As in the HDQ mode, only one interrupt signal is sent to the MPU. Only an overall mask bit can enable or disable the interrupt (the interrupt conditions cannot be masked individually).

20.4.3.3 1-Wire Single-Bit Mode Operation

A single-bit mode can be entered by setting the appropriate bit in the control and status register (1_WIRE_SINGLE_BIT bit HDQ.HDQ_CTRL_STATUS[7]). In this mode, only one bit of data at a time is transferred between the master and the slave. After the bit is transferred, an interrupt is generated (that is, there is an RX-complete for a read operation and a TX-complete for a write operation). Bit 0 of the RX register (HDQ.HDQ_RX_DATA) is updated each time a bit is received from the slave; bit 0 of the TX register (HDQ.HDQ_TX_DATA) contains the bit to be sent.

20.4.3.4 Interrupt Conditions

The HDQ/1-Wire module provides the following interrupt status:

1. Transmission complete:
   A write operation of one byte was completed. Successful or failed completion is not indicated, because there is no acknowledgment from the slave in 1-Wire protocol. This interrupt condition is cleared by reading the interrupt status register (HDQ.HDQ_INT_STATUS).
2. Read complete:
   In 1-Wire mode, the interrupt status indicates that a byte has been successfully read. This interrupt condition is cleared by reading the interrupt status register (HDQ.HDQ_INT_STATUS).
3. Presence detect/time-out:
   In 1-Wire mode, the interrupt status indicates that it is now valid to check the PRESENCEDETECT bit. This interrupt condition is cleared by reading the interrupt status register (HDQ.HDQ_INT_STATUS).

Only one interrupt is generated to the MPU based on any of these interrupt conditions. A read operation on the interrupt status register clears all interrupt status bits that were previously set.

20.4.3.5 Status Flags

The presence-condition-detected status flag is contained in the PRESENCEDETECT bit HDQ.HDQ_CTRL_STATUS[3]. This is valid only in 1-Wire mode. The flag is updated when TIMEOUT bit HDQ.HDQ_INT_STATUS[0] is set. Therefore, its correct value shows only after the interrupt is generated. The firmware must wait for the time-out condition; otherwise, the flag keeps its previous value and is undefined.

20.4.4 Module Power Saving

20.4.4.1 Autoidle Mode

The HDQ/1-Wire module provides an autoidle function in its interconnect clock domain.

The interconnect clock autoidle power-saving mode is enabled or disabled through the AUTOIDLE bit HDQ.HDQ_SYSCONFIG[0]. When this mode is enabled and there is no activity on the interconnect interface, the interconnect clock (HDQ_ICLK) is disabled inside the module, thereby reducing power consumption. When there is new activity on the interconnect interface, the interconnect clock is restarted with no latency penalty. This mode is disabled by default after a reset.

This mode is recommended to reduce power consumption.

20.4.4.2 Power-Down Mode

The HDQ/1-Wire module also provides a power-saving function in its functional clock domain.

Setting the CLOCKENABLE bit in the control and status register (CLOCKENABLE bit HDQ.HDQ_CTRL_STATUS[5]) to 0 shuts off the functional clock (HDQ_FCLK) to the state-machine. The state-machine is reset when the functional clock is disabled; if any transaction is ongoing, it is aborted into the reset state.

The register values are not affected by disabling the functional clock.

Do not access the module registers after the software has put the module in power-down mode except to write to the CLOCKENABLE bit to take the module out of power-down mode.

20.4.5 System Power Management and Wakeup

As part of the system-wide power-management scheme, the HDQ/1-Wire module can go into idle state at the request of the PRCM module (for more information, see Chapter 4, *Power, Reset, and Clock Management*). However, the HDQ/1-Wire module does not support handshake protocol with the PRCM. The HDQ/1-Wire module can go into idle mode only as part of the L4 interconnect clock domain (both CORE_L4_ICLK and FUNC_12M_CLK belong to the L4 interconnect clock domain).

When the AUTO_HDQ bit PRCM.CM_AUTOIDLE1_CORE[22] is set, the HDQ/1-Wire module behavior follows the L4 interconnect clock domain behavior. If the whole domain is put into idle, the HDQ/1-Wire is also put into idle.

Software must ensure correct clock management.

CAUTION

There is no hardware mechanism to prevent cutting off the HDQ/1-Wire clocks while the module is performing a transfer. The result would be a loss of data being transferred.

20.5 HDQ/1-Wire Basic Programming Model

The HDQ/1-Wire module can be considered a simple byte engine because it only implements the hardware interface layer for both HDQ and 1-Wire protocols. The correct sequencing is controlled by the firmware, which is described in this section.

20.5.1 Module Initialization Sequence

20.5.1.1 Mode Selection

MODE bit HDQ.HDQ_CTRL_STATUS[0] allows selection between the HDQ and 1-Wire protocols. When set to 0, the protocol is HDQ; when set to 1, the protocol is 1-Wire. The bit is assumed static for design purposes. The configuration is in HDQ mode by default.

Although this bit can be modified at any point, it is strongly recommended that it be modified only as part of the boot-up configuration.

20.5.1.2 Reset/Initialization

No slave presence test is required in HDQ mode; however, the slave can be reset by setting the INITIALIZATION bit HDQ.HDQ_CTRL_STATUS[2] and the GO bit HDQ.HDQ_CTRL_STATUS[4]. The line is then pulled down (break pulse) and the bit returns to 0 after the pulse is sent. Upon completion, a time-out interrupt is also generated. The slave does not respond to this pulse.

In 1-Wire mode, the slave returns a presence pulse when it receives the initialization pulse. The protocol for initialization is as follows:

1. Set the INITIALIZATION bit HDQ.HDQ_CTRL_STATUS[2] to 1 and the GO bit HDQ.HDQ_CTRL_STATUS[4] to 1 to send the pulse. When the pulse is sent, the bit is cleared in the register.
2. Wait for the presence detect flag (TIMEOUT bit HDQ.HDQ_INT_STATUS[0]) to generate an interrupt. This flag is set when the response time allowed to the slave has elapsed, whether it has sent a pulse or not.
3. Read the HDQ.HDQ_CTRL_STATUS register to check whether the presence pulse has been received before starting any transfer.

20.5.2 HDQ Protocol Basic Programming Model

20.5.2.1 Write Operation

The write operation sequence is as follows:

1. Write the command/address or data value to the TX write register (HDQ.HDQ_TX_DATA).

Note: Steps 2 and 3 can be performed simultaneously.

2. Set DIR bit HDQ.HDQ_CTRL_STATUS[1] to indicate a write.
3. Set GO bit HDQ.HDQ_CTRL_STATUS[4] to start the transmission.
   a. The hardware sends the byte from the TX write register.
   b. In a write operation, the TIMEOUT bit is always cleared, because there is no acknowledge mechanism from the slave.
   c. When the write operation is completed, the TX-complete flag is set in the interrupt status register (TXCOMPLETE bit HDQ.HDQ_INT_STATUS[2]). If interrupts are masked (that is, the corresponding bit has been previously set in the control and status register), no interrupt signal is generated.
   d. The GO bit HDQ.HDQ_CTRL_STATUS[4] is cleared at the end of a write operation.
4. The software must read the interrupt status register to clear the interrupt.
5. Repeat step 1 through step 4 for each successive byte to write.

20.5.2.2 Read Operation

The read operation sequence is as follows:

1. Write the command value to the TX write register (HDQ.HDQ_TX_DATA).
2. Set DIR bit HDQ.HDQ_CTRL_STATUS[1] to 0 and GO bit HDQ.HDQ_CTRL_STATUS[4] to 1 and wait for the TX-complete interrupt.

Note: Steps 3 and 4 can be performed simultaneously.

3. Set DIR bit HDQ.HDQ_CTRL_STATUS[1] to 1 to indicate a read.
4. Write 1 to GO bit HDQ.HDQ_CTRL_STATUS[4] to initiate the read.
   a. The hardware detects a low-going edge on the line (generated by the slave) and receives 8 bits of data in the RX receive buffer register (HDQ.HDQ_RX_DATA). The first bit received is the LSB and the last bit is the MSB. The master performs this step as soon as the slave sends the data, irrespective of the state of GO bit HDQ.HDQ_CTRL_STATUS[4]. However, an RX-complete interrupt is generated only when the software writes the GO bit.
   b. If a time-out occurs, the TIMEOUT bit HDQ.HDQ_CTRL_STATUS[0] is set.
   c. Completion of the operation is indicated by setting the RX-complete flag in the interrupt status register (RXCOMPLETE bit HDQ.HDQ_INT_STATUS[1]. If interrupts are masked (that is, the corresponding bit was previously set in the control and status register), no interrupt signal is generated.
   d. At the end of the read operation, the GO bit HDQ.HDQ_CTRL_STATUS[4] is cleared. It is also cleared if a time-out is detected
5. The software must read the interrupt status register (HDQ.HDQ_INT_STATUS) to determine whether an RX was successfully completed or a time-out occurred.
6. The software reads the RX receive buffer register (HDQ.HDQ_RX_DATA) to retrieve the read data from the slave.
7. Repeat step 1 through step 6 for each successive byte.

Note: In HDQ mode, the address/command is written only once to the slave. However, after the first byte is received, an RX-complete interrupt is set. Therefore, the software must initiate the read of the second byte by setting the GO bit in the control and status register. The first byte received is shadowed and provided to the software while the hardware is fetching the second byte of data.

20.5.3 1-Wire Mode (SDQ) Basic Programming Model

20.5.3.1 Write Operation

The write operation sequence is as follows:

1. Write the ID, command, or data value to the TX write register (HDQ.HDQ_TX_DATA).

Note: Steps 2 and 3 can be performed simultaneously.

2. Set DIR bit HDQ.HDQ_CTRL_STATUS[1] to 0 to indicate a write operation.
3. Set GO bit HDQ.HDQ_CTRL_STATUS[4] to 1 to start the transmission.
   a. The hardware sends the byte stored in the TX write register.
   b. In a write operation, the TIMEOUT bit is always cleared.
   c. When the operation is completed, the TX-complete flag is set in the interrupt status register (TXCOMPLETE bit HDQ_INT_STATUS [2]). No interrupt signal is generated if interrupts are masked (that is, the corresponding bit was previously set in the control and status register).
   d. The GO bit of the control and status register is cleared at the end of a write operation.
4. The software must read the interrupt status register (HDQ.HDQ_INT_STATUS) to clear the interrupt.

5. Repeat step 1 through step 4 for each successive byte to write.

20.5.3.2 Read Operation

The read operation sequence is as follows:

1. Write the address to the TX write register (HDQ.HDQ_TX_DATA).
2. Set DIR bit HDQ.HDQ_CTRL_STATUS[1] to 0 and the GO bit HDQ.HDQ_CTRL_STATUS[4] to 1 and wait for the TX-complete interrupt flag.
3. Write the command value to the TX write register (HDQ.HDQ_TX_DATA).
4. Set DIR bit HDQ.HDQ_CTRL_STATUS[1] to 0 and GO bit HDQ.HDQ_CTRL_STATUS[4] to 1 and wait for the TX-complete interrupt flag.

Note: Steps 5 and 6 can be performed simultaneously.

5. Set DIR bit HDQ.HDQ_CTRL_STATUS[1] to 1 to indicate a read.
6. Set GO bit HDQ.HDQ_CTRL_STATUS[4] to 1 to start the transmission.
   a. The hardware (master) generates a low-going edge and clocks 8 bits of data into the RX receive register (HDQ.HDQ_RX_DATA). The first bit received is the LSB and the last bit is the MSB.
   b. TIMEOUT bit HDQ.HDQ_INT_STATUS[0] is always cleared in a read operation.
   c. When the operation is complete, the RX-complete flag is set in the interrupt status register (RXCOMPLETE bit HDQ.HDQ_INT_STATUS[1]). No interrupt signal is generated if the interrupts are masked (that is, the corresponding bit was previously set in the control and status register).
   d. GO bit HDQ.HDQ_CTRL_STATUS[4] is cleared at the end of the read. It is also cleared if a time-out occurs.
7. The software must read the interrupt status register to determine whether an RX was successfully completed or a time-out occurred.
8. The software reads the RX receive buffer register (HDQ.HDQ_RX_DATA) to retrieve the read data from the slave.
9. Repeat step 1 through step 8 for each successive byte.

20.5.3.3 1-Wire Bit Mode Operation

Select the single-bit mode by setting the 1_WIRE_SINGLE_BIT bit HDQ.HDQ_CTRL_STATUS[7] to 1. In this mode, only one bit of data at a time is transferred between the master and the slave. After the bit is transferred, the corresponding interrupt flag is set (that is, there is an RX-complete (RXCOMPLETE bit HDQ.HDQ_INT_STATUS[1]) for a read operation and a TX-complete (TXCOMPLETE bit HDQ.HDQ_INT_STATUS[2]) for a write operation). Bit 0 of the RX register (HDQ.HDQ_RX_DATA) is updated each time a bit is received; bit 0 of the TX register (HDQ.HDQ_TX_DATA) contains the bit of data to be sent.

20.5.4 Power Management

The software has independent control of the two clock domains (interconnect clock: HDQ_ICLK and functional clock: HDQ_FCLK). Because there is no acknowledge mechanism from the HDQ/1-Wire module to an idle request, the software must ensure that a clock is not shut off while a transfer is being processed (the data would be lost).

If the autoidle function (AUTOIDLE bit HDQ.HDQ_SYSCONFIG[0] set to 1) provides a safe transfer (the module wakes up the HDQ_ICLK as soon as a transfer is initiated), the power-down mode and the PRCM idle requests (through the whole L4 clock domain idle request) must be handled carefully.

The following sections describe the steps to follow to use the power-down and idle modes.

20.5.4.1 Module Power-Down Mode

1. Before shutting off the HDQ_FCLK, wait for an RX-complete or a TX-complete interrupt.
   - In a read operation, the transfer is completed when the RX-complete flag (RXCOMPLETE bit HDQ.HDQ_INT_STATUS [1]) generates an interrupt.
   - In a write operation, the transfer is completed when the TX-complete flag (TXCOMPLETE bit HDQ.HDQ_INT_STATUS [2]) generates an interrupt. The software must check whether the interrupt was generated after the address/command byte was sent or after the data byte was sent. The clock must not be shut off after the command/ address byte is sent; otherwise, the data is not written to the slave.

   HDQ.HDQ_INT_STATUS must be read to clear the interrupt condition.
2. Set the CLOCKENABLE bit HDQ.HDQ_CTRL_STATUS[5] to 0 to disable the clock.

   Do not access the module registers after the software has put the module into power-down mode except to write to the clock-enable bit to take the module out of power-down mode.

20.5.4.2 System Idle Mode

This section describes the steps to follow at the module level before enabling the idle mode at the system level (for more information about the system power management scheme, see Chapter 4, *Power, Reset, and Clock Management*.

As part of the L4 interconnect clock domain, the HDQ/1-Wire clocks can be cut at the PRCM level. HDQ_FCLK can be cut if the EN_HDQ bit PRCM.CM_FCLKEN1_CORE [22] is set to 0 and no other modules require CORE_12M_FCLK. The software must verify that no transfer is in progress.

In a read operation:

1. Wait for an RX-complete interrupt.

   In a read operation, the transfer is completed when the RX-complete flag (RXCOMPLETE bit HDQ.HDQ_INT_STATUS [1]) generates an interrupt.
2. Read the HDQ.HDQ_INT_STATUS to clear the read-complete interrupt flag.
3. Read the HDQ.HDQ_RX_DATA to retrieve the read data.
4. The HDQ_ICLK can be shut off by entering the system idle mode.

In a write operation:

1. Wait for a TX-complete interrupt.

   In a write operation, the transfer is completed when the TX-complete flag (TXCOMPLETE bit HDQ.HDQ_INT_STATUS[2]) generates an interrupt. The software must check whether the interrupt was generated after the address/command byte was sent or after the data byte was sent. The clock must not be shut off after the command/address byte is sent; otherwise, the data is not written to the slave.
2. Read the HDQ.HDQ_INT_STATUS to clear the TX-complete interrupt flag.
3. The HDQ_ICLK can be shut off by entering the system idle mode.

Concerning HDQ_ICLK, two situations can occur:

- The clock is no longer required and EN_HDQ bit PRCM.CM_ ICLKEN1_CORE[22] is set by software. In this case, the clock is cut off, provided no other modules require it. Before setting the EN_HDQ bit, the software must follow the steps described in this section.
- AUTO_HDQ bit PRCM.CM_AUTOIDLE1_CORE[22] is set. In this case, the software must verify that all HDQ/1-Wire transfers are complete before enabling the L4 interconnect clock domain idle mode. Otherwise, the HDQ/1-Wire has no way to prevent the clock from being cut, because no hardware mechanism exists. The steps listed in this section must be verified before putting the L4 clock domain into idle state.

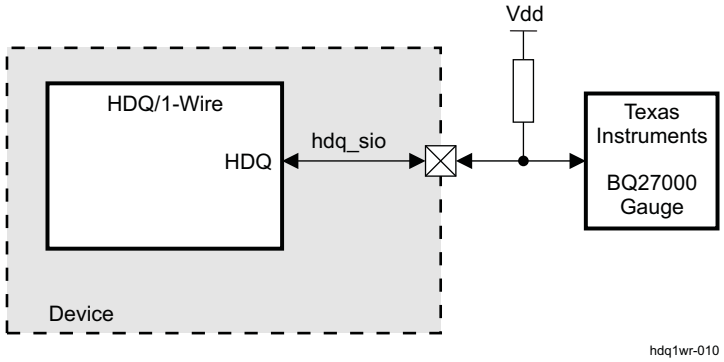
20.6 HDQ/1-Wire Use Cases and Tips

20.6.1 How to Configure the HDQ/1-Wire when Connected with a BQ27000 Gauge

20.6.1.1 Environment

Figure 20-10 details the OMAP device connections with the BQ27000 gauge.

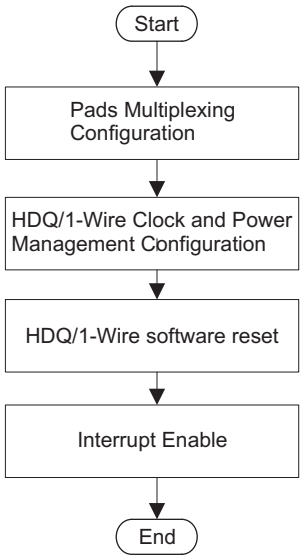
Figure 20-10. Environment



20.6.1.2 Programming Flow

This section details the programming flow of the HDQ/1-Wire. Figure 20-11 shows the main steps of this configuration. The BQ27000 gauge uses the HDQ mode.

Figure 20-11. HDQ/1-Wire Configuration in HDQ Mode



20.6.1.3 Pad Configuration and HDQ/1-Wire Clock and Power Management

Table 20-3 shows the pad multiplexing and the clock and power management configuration to select for the HDQ/1-Wire module.

Table 20-3. Registers Print for HDQ/1-Wire Configuration

Register Name	Address	Value	Value description
SCM.CONTROL_PADCONF_I2C3_SDA	0x 4800 21C4	0x0118 0100	Configure hdq_sio pad in mode 0

Table 20-3. Registers Print for HDQ/1-Wire Configuration (continued)

Register Name	Address	Value	Value description
PRCM.CM_FCLKEN 1_CORE	0x4800 4A00	0x0020 0000	Enable HDQ/1-Wire Functional clock
PRCM.CM_ICLKEN 1_CORE	0x4800 4A10	0x0020 0000	Enable HDQ/1-Wire Interface clock
HDQ_CTRL_STATUS	0x480B 200C	0x0000 0020	Enable clocks and select the HDQ mode
HDQ_SYSCONFIG	0x480B 2014	0x0000 0000	Module clock is free-running (Disable autoidle mode)

20.6.1.4 HDQ/1-Wire Software Reset

Perform a software reset as described in Figure 20-12.

Figure 20-12. Software Reset Flowchart

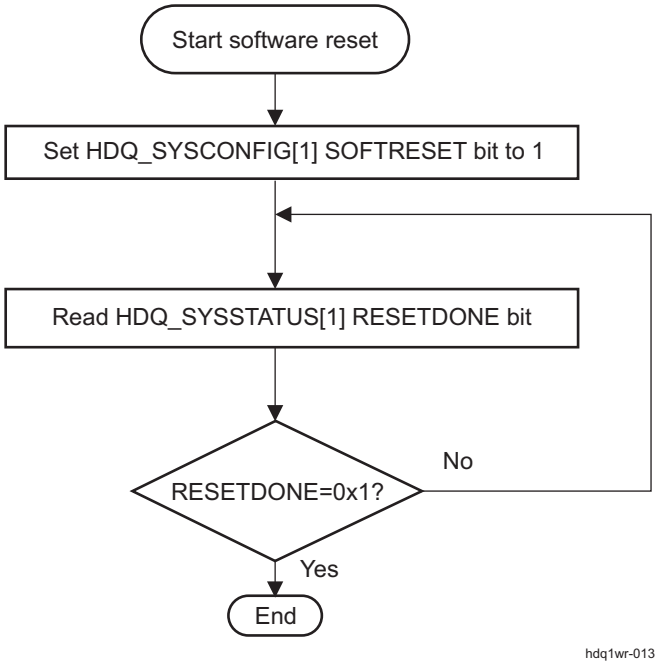


Table 20-4 describes the registers to be configured for the HDQ/1-Wire software reset step.

Table 20-4. Registers Print for HDQ/1-Wire Software Reset

Register Name	Address	Value	Value description
HDQ_SYSCONFIG	0x480B 2014	0x0000 0002	Initiate a software reset. The HDQ_SYSCONFIG[1] SOFTRESET is automatically reset by hardware.
HDQ_SYSSTATUS	0x480B 2018	0x0000 0001	The HDQ_SYSSTATUS[0] RESETDONE is set to 1 when the reset sequence is done.

20.6.1.5 Interrupts Enable

Table 20-5 describes the registers to be configured for the interrupts enable step and the use case values.

Table 20-5. Registers Print for HDQ/1-Wire Interrupts Enable

Register Name	Address	Value	Value description
HDQ_CTRL_STATUS	0x480B 200C	0x0000 0060	Enable Interrupts

20.6.1.6 Read and Write Operations

The Read and Write operations in HDQ mode are described in Section 20.5.2. Please, refer to this section to see how HDQ/1-Wire registers are configured for these operations.

Some write operations are needed to configure the BQ27000 gauge: for example, it is necessary to write a COMMAND KEY (0xA9 or 0x56) in the Device Control Register of the gauge. For more information, see the TI BQ27000 gauge specification.

20.7 HDQ/1-Wire Register Manual

20.7.1 HDQ/1-Wire Instance Summary

Table 20-6 lists the HDQ/1-Wire instances.

Table 20-6. Instance Summary

Module Name	Base Address	Size
HDQ/1-Wire	0x480B 2000	4K bytes

CAUTION

All reserved bits must be written with 0. There is no synchronization between the register clock domain and the state-machine domain. Therefore, the following rules must be observed when accessing the module registers:

- A read from the interrupt status register or the receive buffer register is not allowed unless the processor has been interrupted by the module.
- After the release of the GO bit in the control and status register, no access to the TX data register or the control and status register is allowed until the processor has been interrupted by the module.
- Polling of the interrupt status register by software to determine whether an interrupt was generated is not allowed.
- No access to the module registers should be done after the software puts the module in power-down mode (by setting bit 5 of the control and status register to 0) except to re-enable the clock.

CAUTION

The HDQ/1-Wire registers are limited to 32-bit data accesses. 16-bit and 8-bit are not allowed and can corrupt register content.

20.7.2 HDQ/1-Wire Register Mapping Summary

Table 20-7 lists the HDQ/1-Wire registers.

Table 20-7. HDQ/1-Wire Register Summary

Register Name	Type	Register Width (Bits)	Address Offset	HDQ/1-Wire Physical Address
HDQ_REVISION	R	32	0x000	0x480B2000
HDQ_TX_DATA	RW	32	0x004	0x480B2004
HDQ_RX_DATA	R	32	0x008	0x480B2008
HDQ_CTRL_STATUS	RW	32	0x00C	0x480B200C
HDQ_INT_STATUS	R	32	0x010	0x480B2010
HDQ_SYSCONFIG	RW	32	0x014	0x480B2014
HDQ_SYSSTATUS	R	32	0x018	0x480B2018

20.7.3 HDQ/1-Wire Register Description

Table 20-8 through Table 20-20 describe the individual bits of the HDQ/1-Wire registers.

Table 20-8. HDQ_REVISION

Address Offset	0x000		
Physical Address	0x480B 2000	**Instance**	HDQ/1-Wire
Description	This register contains the IP revision code.		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																								REVISION							

Bits	Field Name	Description	Type	Reset
31:8	Reserved	Reads return 0s.	R	0x000000
7:0	REVISION	IP revision The 4 LSBs indicate a minor revision. The 4 MSBs indicate a major revision. Ex: 0x21: Revision 2.1	R	See(1).

(1) TI internal data

Table 20-9. Register Call Summary for Register HDQ_REVISION

HDQ/1-Wire Register Manual
- HDQ/1-Wire Register Mapping Summary: [0]

Table 20-10. HDQ_TX_DATA

Address Offset	0x004		
Physical Address	0x480B 2004	**Instance**	HDQ/1-Wire
Description	This register contains the data to be transmitted.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																								TX_DATA							

Bits	Field Name	Description	Type	Reset
31:8	Reserved	Reads return 0s.	R	0x000000
7:0	TX_DATA	Transmit data (used in both HDQ and 1-Wire modes)	RW	0x00

Table 20-11. Register Call Summary for Register HDQ_TX_DATA

HDQ/1-Wire Environment
- HDQ Protocol Initialization (Default): [0]

HDQ/1-Wire Functional Description
- 1-Wire Single-Bit Mode Operation: [1]

HDQ/1-Wire Basic Programming Model
- Write Operation: [2]
- Read Operation: [3]
- Write Operation: [4]
- Read Operation: [5] [6]
- 1-Wire Bit Mode Operation: [7]

HDQ/1-Wire Register Manual
- HDQ/1-Wire Register Mapping Summary: [8]

Table 20-12. HDQ_RX_DATA

Address Offset	0x008		
Physical Address	0x480B 2008	**Instance**	HDQ/1-Wire
Description	This register contains the data to be received.		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																								RX_DATA							

Bits	Field Name	Description	Type	Reset
31:8	Reserved	Reads return 0s.	R	0x000000
7:0	RX_DATA	Receive data (used in both HDQ and 1-Wire modes)	R	0x00

Table 20-13. Register Call Summary for Register HDQ_RX_DATA

HDQ/1-Wire Functional Description
- Description: [0]
- 1-Wire Single-Bit Mode Operation: [1]

HDQ/1-Wire Basic Programming Model
- Read Operation: [2] [3]
- Read Operation: [4] [5]
- 1-Wire Bit Mode Operation: [6]
- System Idle Mode: [7]

HDQ/1-Wire Register Manual
- HDQ/1-Wire Register Mapping Summary: [8]

Table 20-14. HDQ_CTRL_STATUS

Address Offset	0x00C		
Physical Address	0x480B 200C	**Instance**	HDQ/1-Wire
Description	This register provides status information about the module.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																								1_WIRE_SINGLE_BIT	INTERRUPTMASK	CLOCKENABLE	GO	PRESENCEDETECT	INITIALIZATION	DIR	MODE

Bits	Field Name	Description	Type	Reset
31:8	Reserved	Reads return 0s.	R	0x000000
7	1_WIRE_SINGLE_BIT	Single-bit mode for 1-Wire 0x0: Disabled 0x1: Enabled	RW	0
6	INTERRUPTMASK	Interrupt masking bit 0x0: Disable interrupts 0x1: Enable interrupts	RW	0
5	CLOCKENABLE	Power down mode bit 0x0: Disable clocks 0x1: Enable clocks	RW	0

Bits	Field Name	Description	Type	Reset
4	GO	Go bit Write 1 to send the appropriate commands. Bit returns to 0 after the command is complete.	RW	0
3	PRESENCEDETECT	Presence detect received, 1-Wire mode only 0x0: Not detected 0x1: Detected	R	0
2	INITIALIZATION	Write 1 to send initialization pulse. Bit returns to 0 after pulse is sent.	RW	0
1	DIR	DIR bit, determines if next command is read or write 0x0: Write 0x1: Read	RW	0
0	MODE	Mode selection bit 0x0: HDQ mode 0x1: 1-Wire mode	RW	0

Table 20-15. Register Call Summary for Register HDQ_CTRL_STATUS

HDQ/1-Wire Environment
- HDQ Protocol Initialization (Default): [0]

HDQ/1-Wire Functional Description
- HDQ/1-Wire Block Diagram: [1]
- Description: [2]
- Single-Bit Mode: [3]
- Description: [4] [5]
- 1-Wire Single-Bit Mode Operation: [6]
- Status Flags: [7]
- Power-Down Mode: [8]

HDQ/1-Wire Basic Programming Model
- Mode Selection: [9]
- Reset/Initialization: [10] [11] [12] [13] [14]
- Write Operation: [15] [16] [17]
- Read Operation: [18] [19] [20] [21] [22] [23] [24]
- Write Operation: [25] [26]
- Read Operation: [27] [28] [29] [30] [31] [32] [33]
- 1-Wire Bit Mode Operation: [34]
- Module Power-Down Mode: [35]

HDQ/1-Wire Use Cases and Tips
- Pad Configuration and HDQ/1-Wire clock and power management: [36]
- Interrupts Enable: [37]

HDQ/1-Wire Register Manual
- HDQ/1-Wire Register Mapping Summary: [38]

Table 20-16. HDQ_INT_STATUS

Address Offset	0x010		
Physical Address	0x480B 2010	**Instance**	HDQ/1-Wire
Description	This register controls interrupt status.		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																													TXCOMPLETE	RXCOMPLETE	TIMEOUT

Bits	Field Name	Description	Type	Reset
31:3	Reserved	Reads return 0s.	R	0x00000000
2	TXCOMPLETE	TX-complete interrupt flag Set to 1 if cause of interrupt. Set to 0 when register read.	R	0
1	RXCOMPLETE	Read-complete interrupt flag Set to 1 if cause of interrupt. Set to 0 when register read.	R	0
0	TIMEOUT	Presence detect/timeout interrupt flag In 1-Wire mode, set to 1 if slave's presence detected. In HDQ mode, set to 1 if timeout on read occurs. Set to 0 when register read.	R	0

Table 20-17. Register Call Summary for Register HDQ_INT_STATUS

HDQ/1-Wire Functional Description
- Description: [0] [1] [2]
- Interrupt Conditions: [3] [4] [5]
- Description: [6] [7]
- Interrupt Conditions: [8] [9] [10]
- Status Flags: [11]

HDQ/1-Wire Basic Programming Model
- Reset/Initialization: [12]
- Write Operation: [13]
- Read Operation: [14] [15]
- Write Operation: [16] [17]
- Read Operation: [18] [19]
- 1-Wire Bit Mode Operation: [20] [21]
- Module Power-Down Mode: [22] [23] [24]
- System Idle Mode: [25] [26] [27] [28]

HDQ/1-Wire Register Manual
- HDQ/1-Wire Register Mapping Summary: [29]

Table 20-18. HDQ_SYSCONFIG

Address Offset	0x014		
Physical Address	0x480B 2014	**Instance**	HDQ/1-Wire
Description	This register controls various bits.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																														SOFTRESET	AUTOIDLE

Bits	Field Name	Description	Type	Reset
31:2	Reserved	Reads return 0s.	R	0x00000000
1	SOFTRESET	Start soft reset sequence. 0x0: Disabled 0x1: Enabled	RW	0
0	AUTOIDLE	Interconnect idle 0x0: Module clock is free-running. 0x1: Module is in power saving mode: Clock is running only when module is accessed or inside logic is in function to process events.	RW	0

Table 20-19. Register Call Summary for Register HDQ_SYSCONFIG

HDQ/1-Wire Integration
- HDQ/1-Wire Reset Scheme: [0]

HDQ/1-Wire Functional Description
- Autoidle Mode: [1]

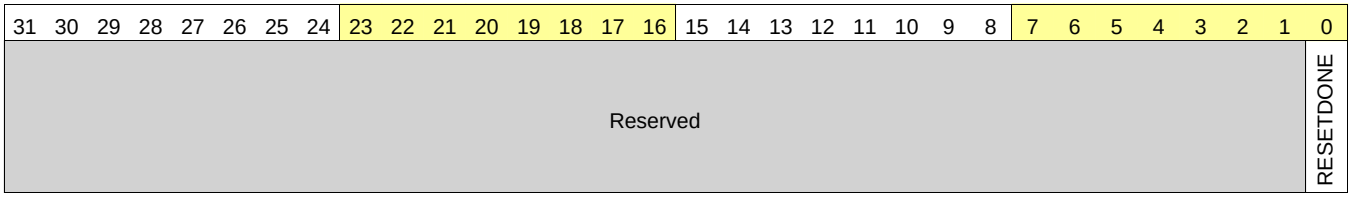
HDQ/1-Wire Basic Programming Model
- Power Management: [2]

HDQ/1-Wire Use Cases and Tips
- Pad Configuration and HDQ/1-Wire clock and power management: [3]
- HDQ/1-Wire Software Reset: [4] [5]

HDQ/1-Wire Register Manual
- HDQ/1-Wire Register Mapping Summary: [6]

Table 20-20. HDQ_SYSSTATUS

Address Offset	0x018		
Physical Address	0x480B 2018	**Instance**	HDQ/1-Wire
Description	This register monitors the reset sequence.		
Type	R		



Bits	Field Name	Description	Type	Reset
31:1	Reserved	Reads return 0s.	R	0x00000000
0	RESETDONE	Reset monitoring 0x0: The module is currently performing its reset. When the module is in power-down mode, set to 0 to indicate this fact. 0x1: The module has finished its reset.	R	0x-

Table 20-21. Register Call Summary for Register HDQ_SYSSTATUS

HDQ/1-Wire Use Cases and Tips
- HDQ/1-Wire Software Reset: [0] [1]

HDQ/1-Wire Register Manual
- HDQ/1-Wire Register Mapping Summary: [2]

Multi-Channel Buffered Serial Port

This chapter introduces the Multi-Channel Buffered Serial Port of the OMAP34xx multimedia device.

Topic **Page**

21.1 McBSP Overview 2920
21.2 McBSP Environment 2923
21.3 McBSP Integration 2932
21.4 McBSP Functional Description 2953
21.5 McBSP Basic Programming Model 2988
21.6 McBSP Use Case and Tips 3017
21.7 McBSP Register Manual 3020

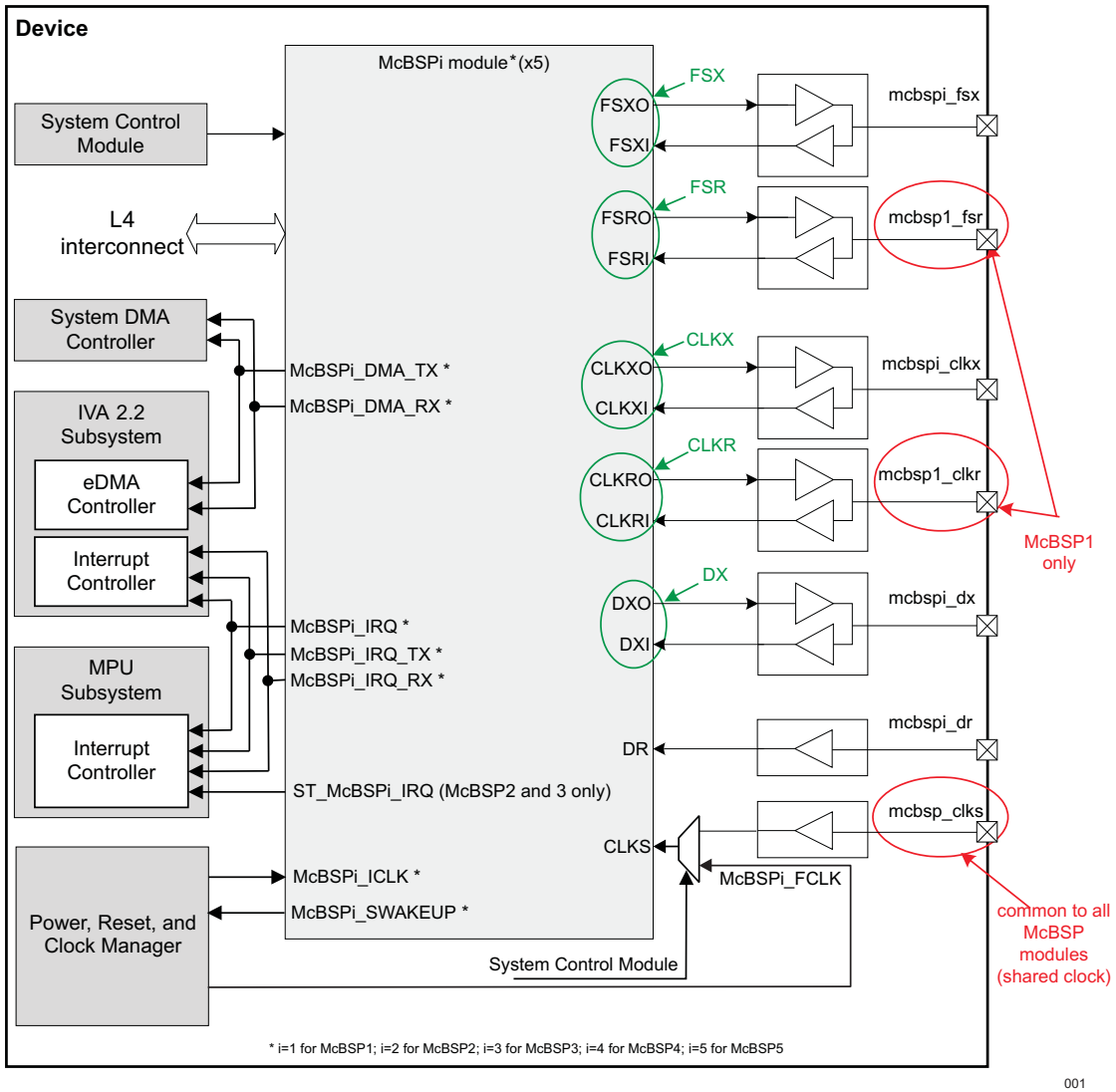
21.1 McBSP Overview

The multichannel buffered serial port (McBSP) provides a full-duplex direct serial interface between the device and other devices in a system such as other application chips (digital base band), audio and voice codec (TWL4030 device), etc. Because of its high level of versatility, it can accommodate to a wide range of peripherals and clocked frame oriented protocols (for details, see Section 21.1.1).

The device provides five instances of the McBSP module.

Figure 21-1 shows the McBSP overview in the device.

Figure 21-1. McBSP Highlight



21.1.1 McBSP Features

The main features of the McBSP modules are:

- L4 interconnect slave interface supports:
  - 32-bit data bus width
  - 32-bit access supported
  - 16- /8-bit access supported only by data registers

  - 10-bit address bus width
  - Burst mode not supported
  - Write nonposted transaction mode supported
- 128 × 32-bit words (512 bytes) for each buffer for transmit/receive operations (McBSP1, 3, 4, 5)
- 5K bytes (1024 x 32 bits for audio buffer + 256 × 32 bits for buffer) for each buffer for transmit/receive audio operations (McBSP2 only)
- Interrupts configurable in legacy mode (2 requests) or PRCM compliant (1 request)
- Transmit and receive DMA requests triggered with programmable FIFO thresholds
- SIDETONE core support: Audio loopback capability (McBSP2 and 3 only)
- Multidrop support
- Serial interface description
  - 6 pin configuration (McBSP 1 only)
  - 4 pin configuration (McBSP2, 3, 4, 5)
  - Full-duplex communication
  - Multichannel selection modes
    - Support to enable or block transfers in each of the channels
    - 128 channels for transmission and for reception
  - Direct interface to industry-standard codecs, analog interface chips (AICs), and other serially connected A/D and D/A devices:
    - Inter-IC sound (I2S) compliant devices
    - Pulse code modulation (PCM) devices
    - Time division multiplexed (TDM) bus devices

CAUTION

McBSP modules do not offer support for μ-law and A-law companding, two partitions mode dynamic reassignment, AC'97, and SPI protocol.

  - A wide selection of data sizes: 8, 12, 16, 20, 24, and 32 bits
  - Bit reordering (send/receive least significant bit [LSB])
- Clock and frame-synchronization generation support:
  - Independent clocking and framing for reception and for transmission up to 48 MHz
  - Support for external generation of clock signals and frame-synchronization (frame-sync) signals
  - A programmable sample rate generator for internal generation and control of clock signals and frame-sync signals
  - Programmable polarity for frame-sync pulses and for clock signals

Notes:

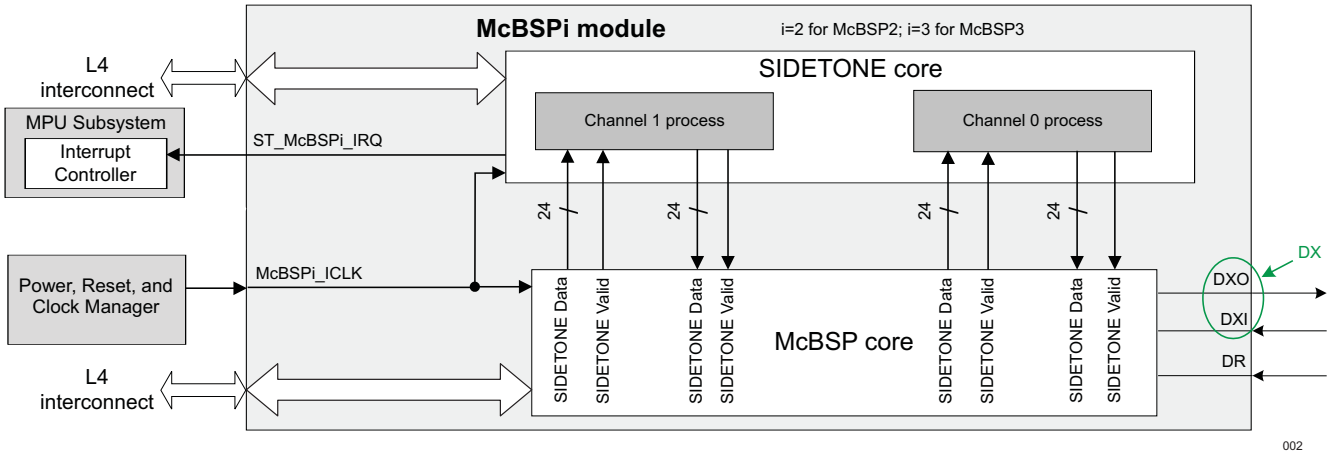
- McBSP modules do not support features such as re-transmit or re-receive of an erroneous frame or word.
- McBSP modules support dual phase frames to provide I2S fully compliant capabilities. But, this dual phase mode is limited at one channel (or word) for each phase instead of 128 channels max for single phase mode.

21.1.2 SIDETONE Core

The purpose of this section is to present the SIDETONE core implemented in McBSP2 and 3 modules. It is required that two of the audio input channels to be looped back, filtered, and mixed to the two corresponding audio output channels.

Data to be processed comes on two separate paths (one for each channel) through a simple interface. After filtering the data, gain is applied and outputted through a similar interface. For more details, see Figure 21-2).

Figure 21-2. SIDETONE Core Architecture



The SIDETONE core offers the following features:

- Filter coefficients are shared between the two channels (the filter coefficient is assumed to represent values in the range –1…+1)
- Gains are independent for the two channels (the gain coefficients are in the range –2…+2)
- The filter output is multiplied with the gain, and the result is rounded to output channel word width.
- The FIR filter length is 128 samples.
- Filtered loop back signals are added to the corresponding output signals, and saturation is applied if the result exceeds arithmetic range.
- Filter coefficients and gains are programmable:
  - Coefficients are changed while audio is stopped.
  - Gains are changed at any time (also during audio operation with SIDETONE enabled).

21.2 McBSP Environment

This section describes the intended functions for McBSP module from an environment point of view (that is, external connections). It presents the McBSP connectivity options, lists the possible interfaces, and details the protocol and data format used in each case.

21.2.1 McBSP Functions

The device provides five instances of the McBSP module, called McBSP1, McBSP2, McBSP3, McBSP4, and McBSP5.

List of recommended usage (non exhaustive) per McBSP modules in the device:

- McBSP1: Digital baseband (DBB) Data
- McBSP2: Audio data with audio buffer and SIDETONE feature
- McBSP3: Bluetooth voice data with SIDETONE feature
- McBSP4: DBB voice data
- McBSP5: Midi data

Table 21-1 describes the functions and the corresponding application fields.

Table 21-1. Functions Description

Function	Application field	Recommended McBSP module	Description
Control and Data	Digital Base Band (DBB) Data	McBSP1	Serial interface to transfer data
Audio Data	Audio Data with Audio Buffer	McBSP2	Audio interface to transfer audio data with Inter-IC Sound codec (I2S)
	Audio Data with Audio Buffer and SIDETONE feature		
	Midi Data	McBSP5	
Voice Data	Bluetooth Voice Data	McBSP3	Voice interface to transfer voice data with Pulse Code Modulation codec (PCM)
	Bluetooth Voice Data with SIDETONE feature		
	DBB Voice Data	McBSP4	

21.2.2 McBSP Signals Descriptions

The five McBSP modules consist of a data-flow path and a control path connected to external devices by a serial interface with 6 pins configuration (McBSP 1 only) or 4 pins configuration (McBSP2, 3, 4, 5).

For a McBSP module with 6 pins configuration, an internal loop back capability between Transmitter and Receiver clock signals, and both frame synchronisation signals enables using the McBSP module with 4 pins configuration. The related internal multiplexers are controlled through the System Control Module on the device (see Section 21.3).

Table 21-2. Input/Output Description

Pin Name[1]	I/O[2]	Description	Internal Signal Name	Reset Value	Control and Data	Audio Data	Voice Data
mcbsp_clks	I	External clock (shared by all McBSP modules)	CLKS	-	ü	ü	ü
mcbspi_dr	I	Receive serial data	DR	-	ü	ü	ü
mcbspi_dx	I/O[3]	Transmit serial data	DX	0	ü	ü	ü
mcbspi_clkx	I/O [3]	Transmit clock[4]	CLKX	0	ü	ü	ü
mcbspi_fsx	I/O [3]	Transmit frame synchronization [5]	FSX	0	ü	ü	ü
mcbsp1_clkr	I/O [3]	Receive clock[6]	CLKR	1	ü		
mcbsp1_fsr	I/O [3]	Receive frame synchronization [7]	FSR	0	ü		

(1) I = 1 to 5
(2) I = Input; O = Output
(3) For details of the input/output selection, see Section 21.2.3.1.
(4) This signal is also used as CLKR when it is configured as output.
(5) This signal is also used as FSR when it is configured as output.
(6) This signal is also used as CLKX when it is configured as input.
(7) This signal is also used as FSX when it is configured as input.

- The mcbsp_clks pin can be used to inject an external clock. This clock is used to generate control signals depending on the module internal configuration (see Section 21.4.3). The CLKS signal of the McBSP modules is linked to an external signal through the mcbsp_clks pin, but the CLKS signal can also be linked to an internal clock provided by PRCM of the device. For more information, see Section 21.3.
- Data are transmitted to external devices interfacing with McBSP modules via the mcbspi_dx pin. Data from those devices are received on the mcbspi_dr pin.

Note: The mcbspi_dx pin is an input/output signal to use the McBSP module in half-duplex mode.

- Control information is communicated via the following pins: mcbspi_clkx (transmit clock), mcbsp1_clkr (receive clock), mcbspi_fsx (transmit frame-sync), and mcbsp1_fsr (receive frame-sync).

CAUTION

External pins mcbsp1_clkr and mcbsp1_fsr are connected to pads only for McBSP1 module; other McBSP modules don't have these connections. For these modules, CLKR and FSR signals sources are mcbspi_clkx and mcbspi_fsx pins, respectively. Consequently, there is a light restriction on other McBSP modules when used in full-duplex mode. Both reception and transmission use the same clock signal and the same frame synchronization signal.

21.2.3 McBSP Functions Description

21.2.3.1 McBSP Modes

For all McBSP functions, McBSP modules can operate in master or slave mode. The difference between these modes is the definition of the source of McBSP clocks and McBSP frames synchronization:

- Master mode: McBSP module provides them to the external device
- Slave mode: McBSP module receives them from the external device

The choice between the two modes depends of technical data of the external device and the type of interface (protocols and data formats). For one McBSP module, there are four possible functions:

1. Transmit and receive master mode
2. Transmit and receive slave mode
3. Transmit master mode and receive slave mode
4. Transmit slave mode and receive master mode

Note: If the McBSP module has a serial interface with 4 pins configuration (McBSP2, 3, 4, 5), only modes 1 or 2 are possible.

Figure 21-3 shows the connection between the McBSP1 module (6 pins configuration) and an external device in transmit master mode and receive slave mode.

Figure 21-3. Mode Overview of McBSP1 Module

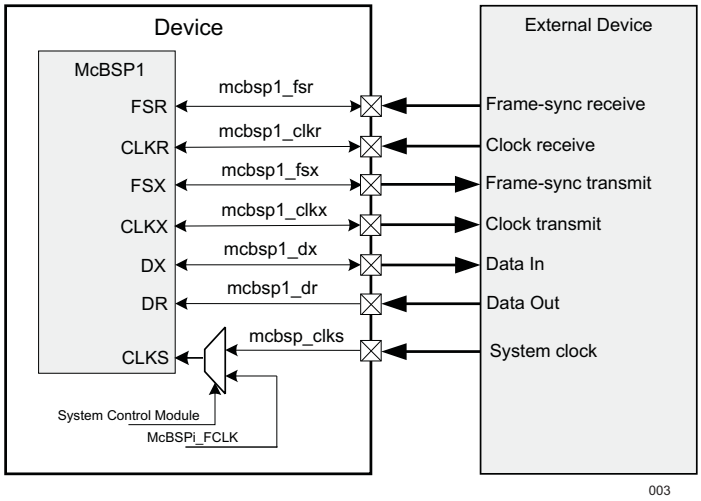
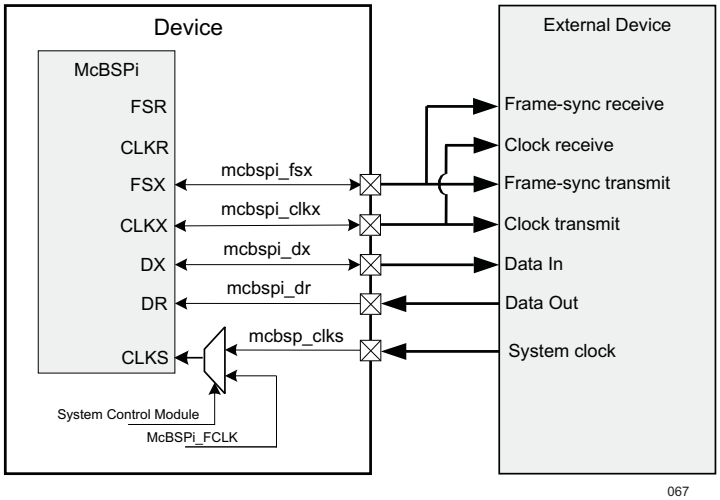


Figure 21-4 shows the connection between the McBSPi module, with I = 2, 3, 4 or 5 (4 pins configuration) and an external device in transmit and receive master mode.

Figure 21-4. Mode Overview of McBSPi Module

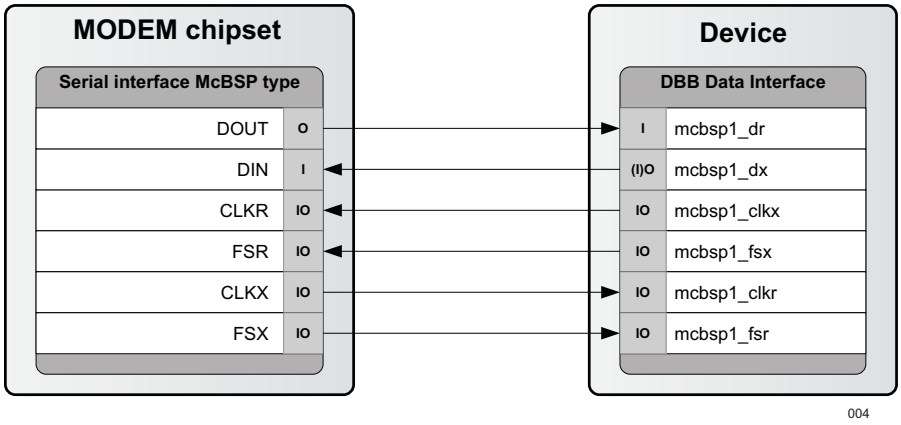


21.2.3.2 McBSP Functions

21.2.3.2.1 McBSP Function 1: Control and Data

In full-duplex mode (reception and transmission use independent clock signals and frame synchronization signals), the McBSP module can be used to exchange control and data with an external chipset, allowing the device to be interfaced with a modem device. Figure 21-5 shows typical connections between device and modem chipset to illustrate DBB data application.

Figure 21-5. DBB Data Application



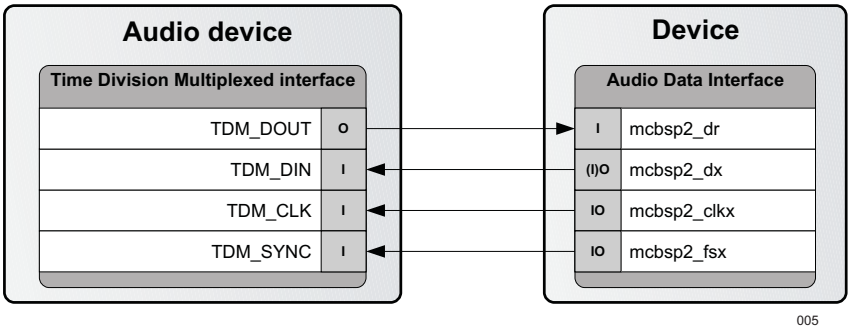
In Figure 21-5, the McBSP1 module is configured in transmit master mode and in receive slave mode.

21.2.3.2.2 McBSP Function 2: Audio Data

The McBSP module is connected to audio devices through the I2S interface. The I2S link serial interface is a TDM slot based serial interface that is used to transfer audio data. Those audio devices can be either AICs or other serially connected A/D and D/A devices.

Figure 21-6 shows typical connections between device and a typical device of analog audio interface (TWL4030 device) to illustrate Audio Data application. The typical device contains several audio analog inputs and outputs, as well as digital microphone inputs.

Figure 21-6. Audio Data Application



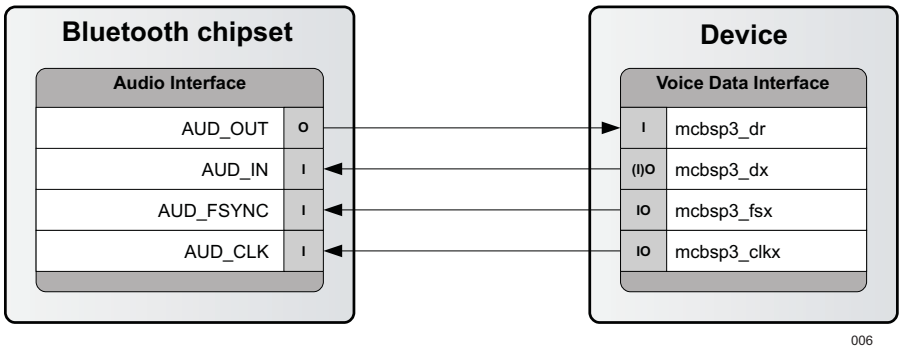
In Figure 21-6, the McBSP2 module is configured in transmit and receive master mode.

21.2.3.2.3 McBSP Function 3: Voice Data

The McBSP module is connected to a voice device through the PCM interface. The PCM link serial interface is a TDM slot based serial interface that is used to transfer voice data. The voice devices can be either modem chipsets, Bluetooth chipsets or others devices with voice data interface.

Figure 21-7 shows typical connections between device and Bluetooth chipset (TI BRF6300 or TI BRF6350) to illustrate voice data application.

Figure 21-7. Voice Data Application



In Figure 21-7, the McBSP3 module is configured in transmit and receive master mode.

21.2.4 McBSP Protocols and Data Formats

The McBSP module can use one of the three protocols with associated data formats:

- Serial protocol to exchange serial data
- Audio protocol to exchange the audio samples
- Voice protocol to exchange the voice samples

The McBSP modules offer the flexibility to modify the following parameters to adapt to the device features as described in the following subsections.

21.2.4.1 Words, Frames, and Phases Definitions

21.2.4.1.1 Words or Channels

The data bits are transferred (transmission or reception) in a group called a serial word or channel. The number of bits in a word (length) is programmable via bits field (McBSPi.MCBSPLP_RCR1_REG[7:5] RWDLEN1 field and McBSPi.MCBSPLP_RCR2_REG[7:5] RWDLEN2 field, McBSPi.MCBSPLP_XCR1_REG[7:5] XWDLEN1 field and McBSPi.MCBSPLP_XCR2_REG[7:5] XWDLEN2 field) and can be 8, 12, 16, 20, 24 or 32 bits (see Section 21.4.2.3). The McBSP module uses clock signals to control the time for each bit transfer. Data are sampled/driven on rising or falling edge of clock signals. This clock polarity is programmable via bits field of pin-control register (McBSPi.MCBSPLP_PCR_REG).

For more information, see Section 21.4.2.4.

21.2.4.1.2 Frames

One or more words (max 128) are transferred in a group called a frame. The McBSP module can transmit / receive a maximum of 128 words per frame, programmable via bits field of transmit and receive control registers (McBSPi.MCBSPLP_XCR1_REG/McBSPi.MCBSPLP_XCR2_REG and McBSPi.MCBSPLP_RCR1_REG/McBSPi.MCBSPLP_RCR2_REG). For more details, see Section 21.4.2.3.

All the words in a frame are sent in a continuous stream. However, there can be pauses between frame transfers. The McBSP module uses frame-synchronization signals to determine when each frame is received/transmitted. When a pulse occurs on a frame-synchronization signal, the McBSP module begins receiving/transmitting a frame of data. When the next pulse occurs, the McBSP module receives/transmits the next frame, and so on. Frame-synchronization pulse is active high or low. This pulse polarity is programmable via bits field of pin-control register (McBSPi.MCBSPLP_PCR_REG).

Each frame transfer can be delayed by 0, 1, or 2 clock cycles, depending on the value of bits for transmit and receive control registers (McBSPi.MCBSPLP_XCR2_REG and McBSPi.MCBSPLP_RCR2_REG). For more information, see Section 21.4.4.6.3 and Section 21.4.4.3.3.

21.2.4.1.3 Phases

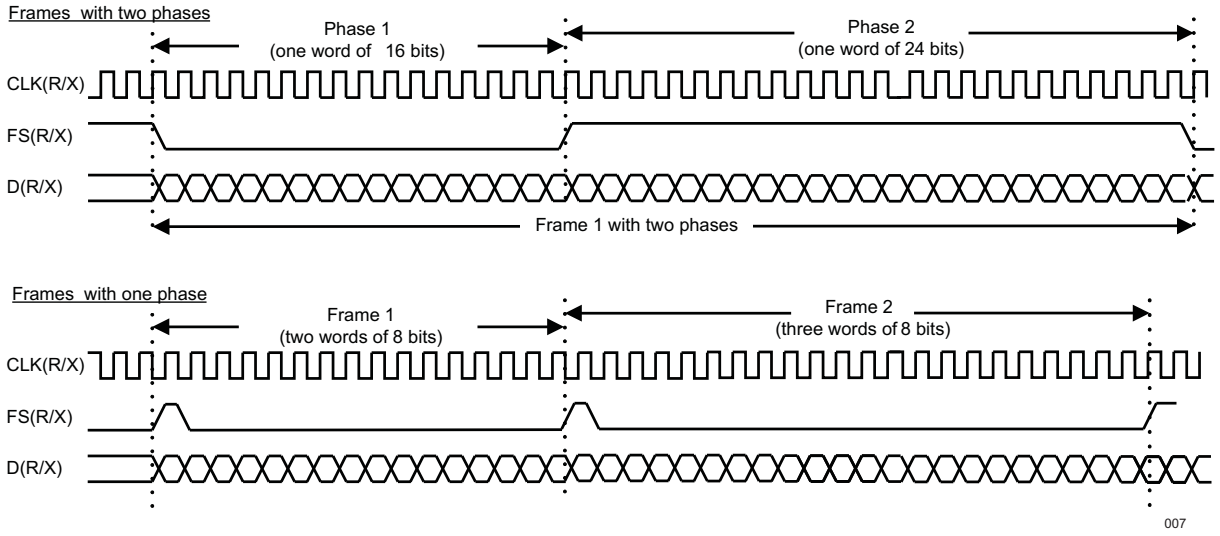
The McBSP module allows configuring each frame to contain one or two phases. The McBSP module supports dual phase frames to provide I2S fully compliant capabilities. These two phases represent left and right channels of audio stereo signals.

The limitation on dual phase frame is that the number of words per phase must be set to one for both first and second phase. But, the number of bits per word can be specified differently for each of the two phases of a frame, allowing greater flexibility in structuring data transfers.

For example, software may define a frame composed of a first phase with one 12-bit word and a second phase with one 16-bit word. This configuration allows the software to compose frames for custom applications. For more details, see Section 21.4.2.4.

Figure 21-8 shows signal activity for two possible reception/transmission scenarios.

Figure 21-8. McBSP Reception/Transmission Signal Activity



21.2.4.2 Serial Protocol and Data Formats

21.2.4.2.1 Protocol

The serial protocol is used to send and receive control data without specific formats. This allows McBSP module to accommodate all serial devices and their protocols.

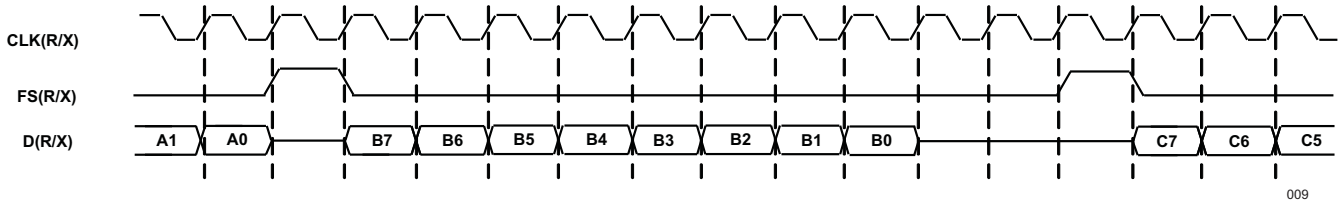
21.2.4.2.2 Data Format

Figure 21-9 shows typical operation of the McBSP clock and frame sync signals. Serial clocks CLKR and CLKX define the boundaries between bits for receive and transmit, respectively. Similarly, frame-sync signals FSR and FSX define the beginning of an element and/or frame transfer. The McBSP module allows the configuration of the following parameters for data and frame synchronization:

- Polarity of FSR, FSX, CLKX, and CLKR
- The number of words per frame
- The number of bits per word
- Whether subsequent frame synchronization restarts the serial data stream or is ignored
- The data delay from frame synchronization to first data bit which can be 0- , 1- , or 2-bit delays

The configuration is independent for receive and transmit parts. For more details and configuration examples, see Section 21.4 and Section 21.5.

Figure 21-9. Serial Data Formats



21.2.4.3 Audio Protocol and Data Formats

21.2.4.3.1 Protocol

The I2S protocol is used to send and receive audio data from 8 KHz up to 48 KHz sampling rate (frame-sync frequency), with 16 bits or 32 bits per words (Supported frequencies are 8, 11.025, 12, 16, 22.05, 24, 32, 44.1 and 48 KHz).

The frame-synchronization signal defines the frame length in the I2S protocol. Each frame consists of a fixed number of words. In dual-phase frame, the frame-synchronization signal is low for the left phase time slot and is high for the right phase time slot. In addition, the frame-synchronization signal is synchronous to the falling edge of the clock signal.

21.2.4.3.2 Data Formats

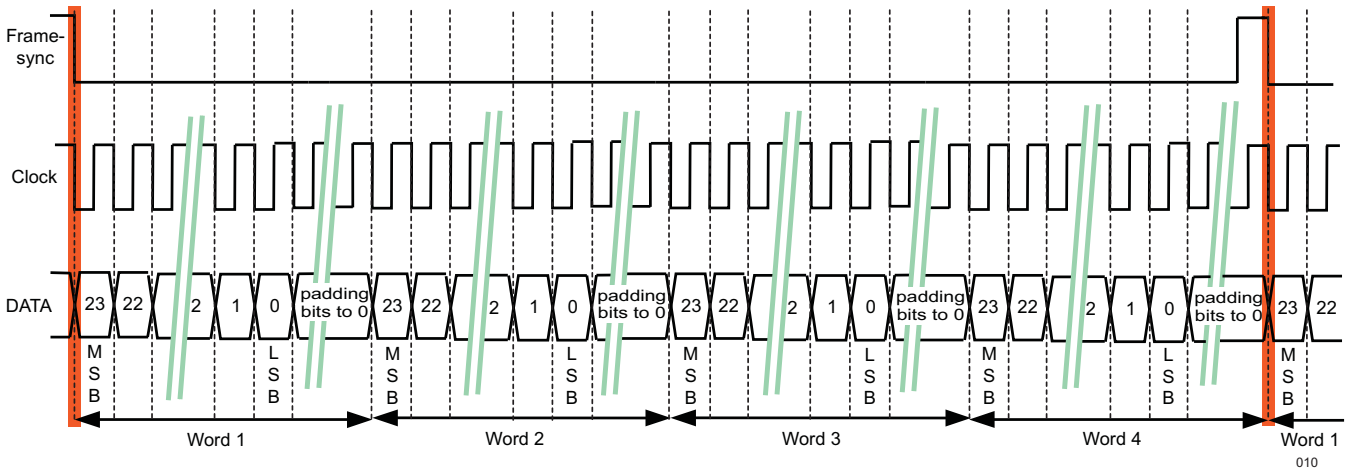
The I2S protocol supports TDM, I2S, left justified, and right justified data formats.

Bits of each word (sample) are clocked using clock signal. For each word, MSB is first. LSBs are padded to 0 when the data length (8, 12, 16, 20, or 24 bits) is less than the sample word width (16 or 32 bits).

21.2.4.3.2.1 TDM Data Format

Figure 21-10 shows that each frame of TDM data format is composed of four words (or channels).

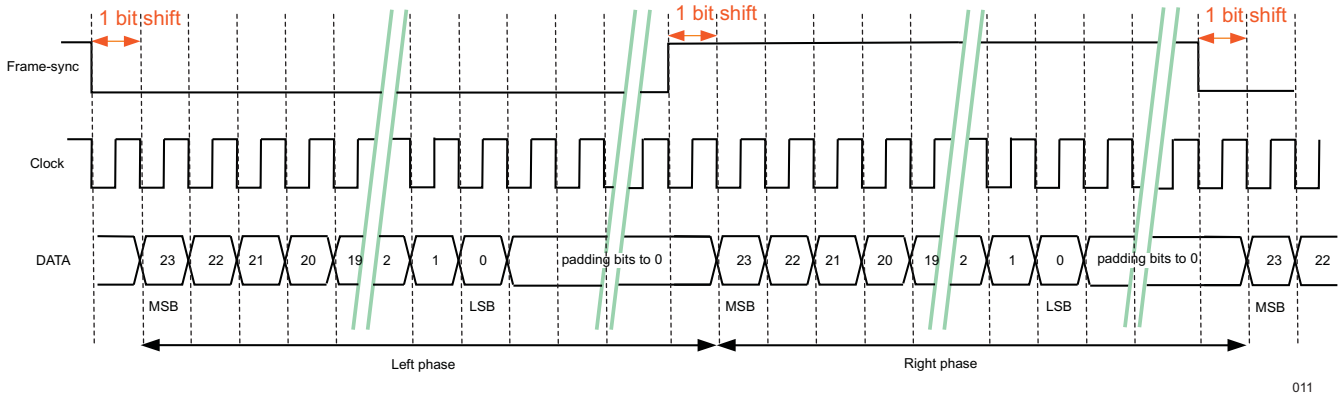
Figure 21-10. TDM Data Format; Word Width: 32 Bits; Data Length: 24 Bits



21.2.4.3.2.2 I2S Data Format

Figure 21-11 shows an example with 24 bits data (MSB first) and 8 padding bits at '0'.

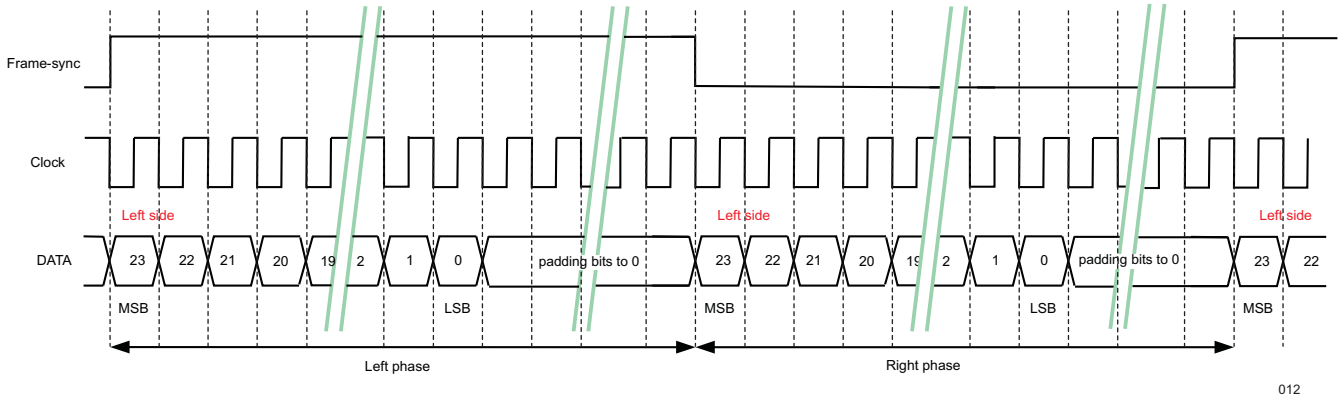
Figure 21-11. I2S Data Format; Word Width: 32 Bits; Data Length: 24 Bits



21.2.4.3.2.3 Left Justified Data Format

Figure 21-12 shows an example with 24 bits data (MSB first) and 8 padding bits at '0'.

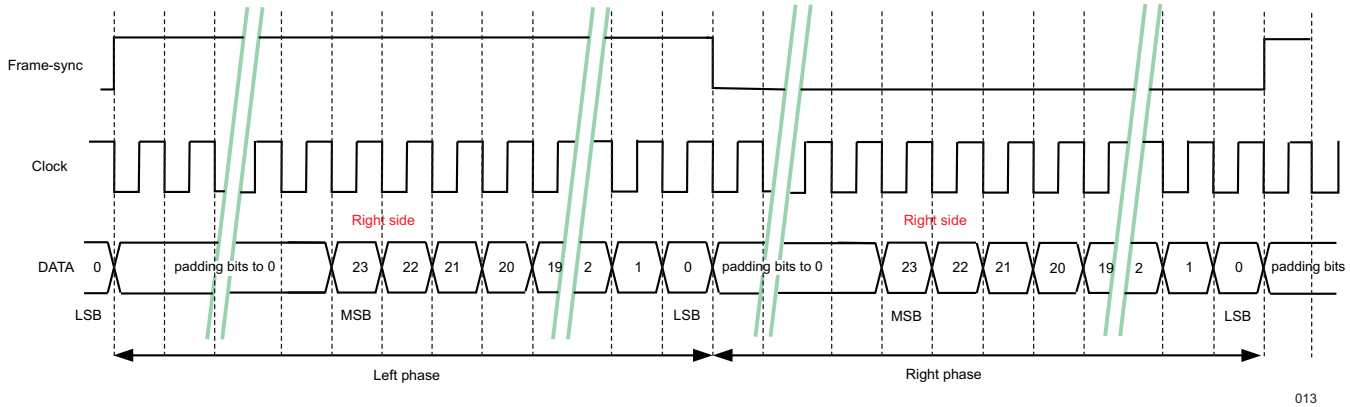
Figure 21-12. Left Justified Data Format; Word Width: 32 Bits; Data Length: 24 Bits



21.2.4.3.2.4 Right Justified Data Format

Figure 21-13 shows an example with 8 padding bits at '0' and 24 bits data (MSB first).

Figure 21-13. Right Justified Data Format; Word Width: 32 Bits; Data Length: 24 Bits



21.2.4.4 Voice Protocol and Data Formats

21.2.4.4.1 Protocol

The PCM protocol is intended to transfer voice data at 8 kHz (default narrowband mode) or 16 kHz (wideband mode) sample rates (frame-sync frequency). PCM protocol can act as a slave or master, and is used by the Bluetooth interface and the modem generic interface. The frame-synchronization defines the frame length in the PCM protocol. Bits are clocked using PCM clock signal, with MSB first.

21.2.4.4.2 Data Formats

Two modes are available for the PCM protocol: mode 1 and mode 2. For these both modes, it has two types of operations: Mono or stereo channels. The difference between PCM mode 1 and PCM mode 2 is in the way they use either the rising or the falling edge of clock signal, and the frame-synchronization polarity.

- PCM Mode 1: Input data is latched on the falling edge of the clock, and the transmitted data starts on the rising edge of the clock. Frame-synchronization pulse is active high.
- PCM Mode 2: Input data is latched on the falling edge of the clock, and the transmitted data starts on the falling edge of the clock. Frame-synchronization pulse is active low.

Figure 21-14 and Figure 21-15 shows an example of PCM protocol, mode 1 and mode 2, respectively, for a frame composed one word (width: 32 bits) with 16 bits data.

Figure 21-14. PCM Protocol - Mode 1 Data Format

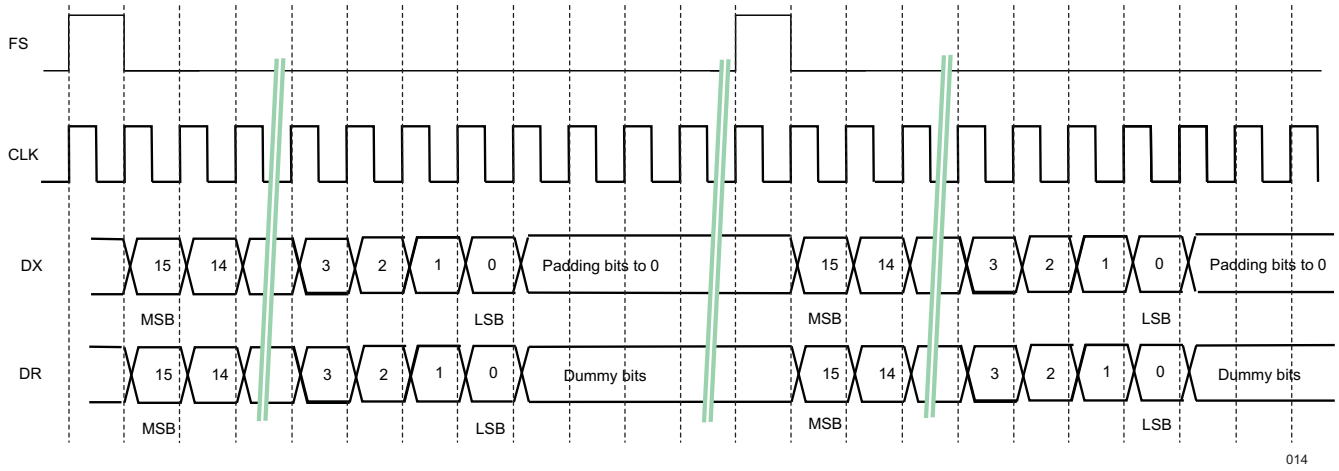
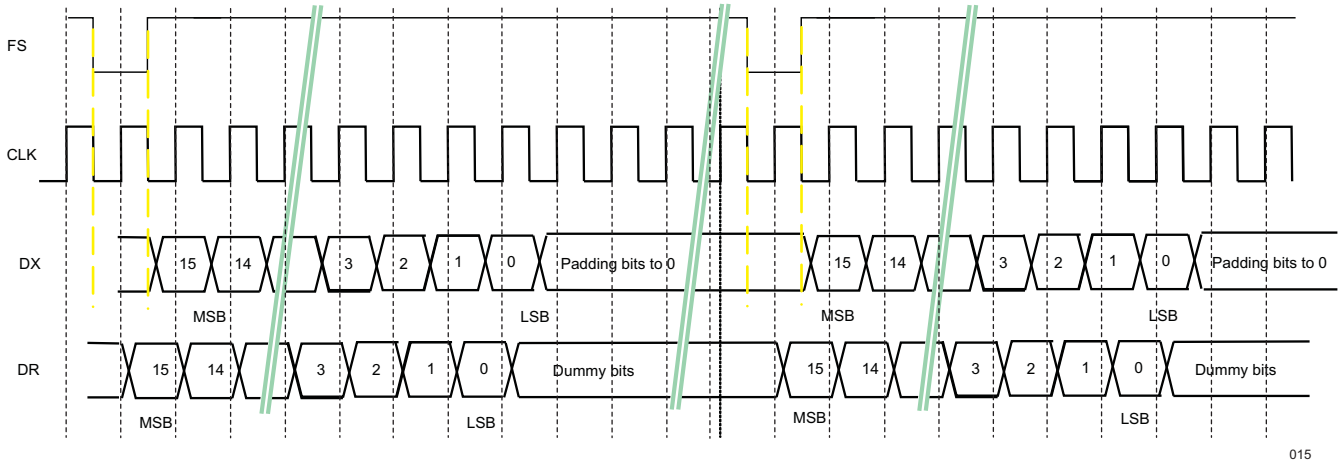


Figure 21-15. PCM Protocol - Mode 2 Data Format



21.3 McBSP Integration

This section describes the McBSP modules integration inside the device and all the details about signal source controls, clocks, resets, power management, and hardware requests.

McBSP modules are divided into 2 families: McBSP modules that are gated in CORE domain (McBSP1 and 5), and McBSP modules that are gated in PER domain (McBSP2, 3, and 4).

For more details on CORE and PER domain, see Chapter 4, *Power, Reset and Clock Management*.

Figure 21-16 through Figure 21-20 highlight the integration of the McBSP modules in the device including interrupt handlers, DMA requests, clock generators, and interconnections.

Figure 21-16. McBSP1 Integration

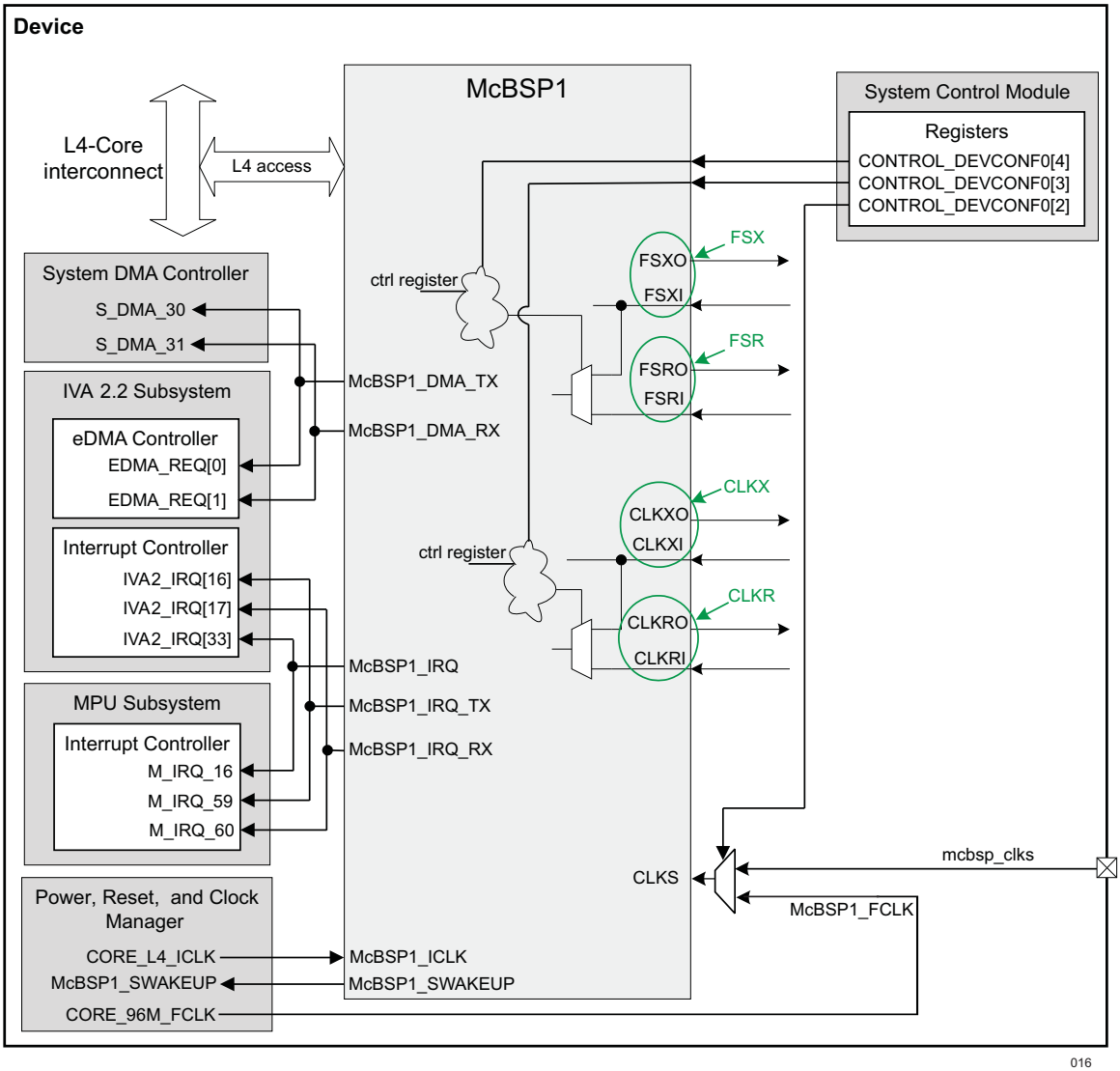


Figure 21-17. McBSP2 Integration

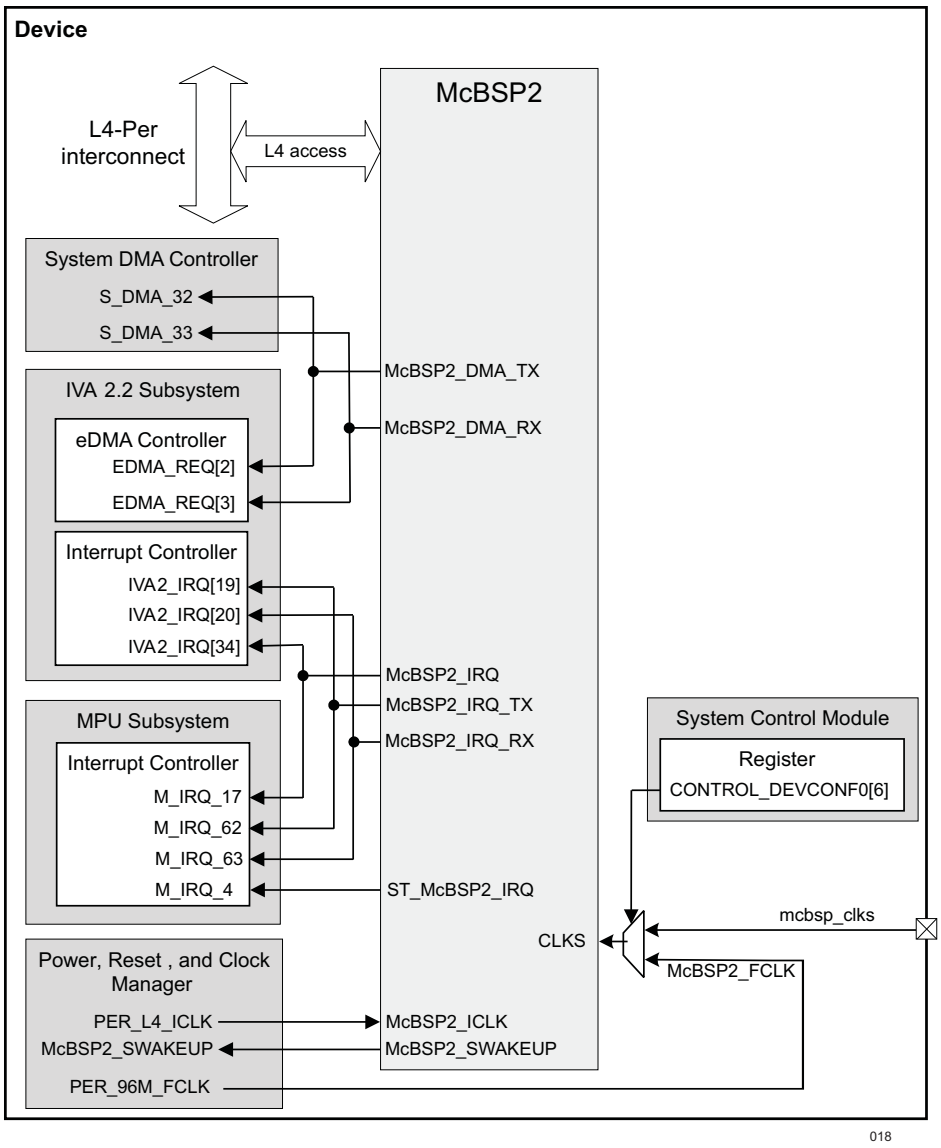


Figure 21-18. McBSP3 Integration

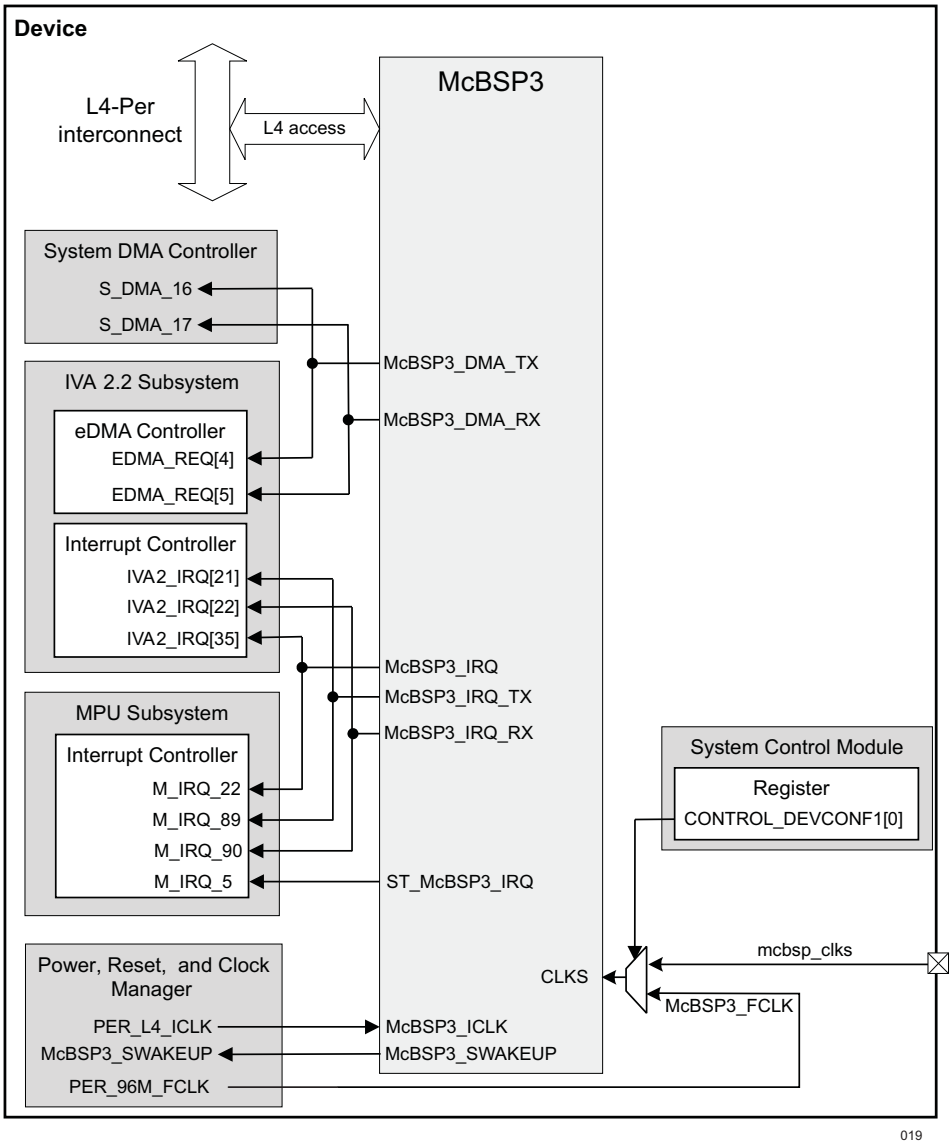


Figure 21-19. McBSP4 Integration

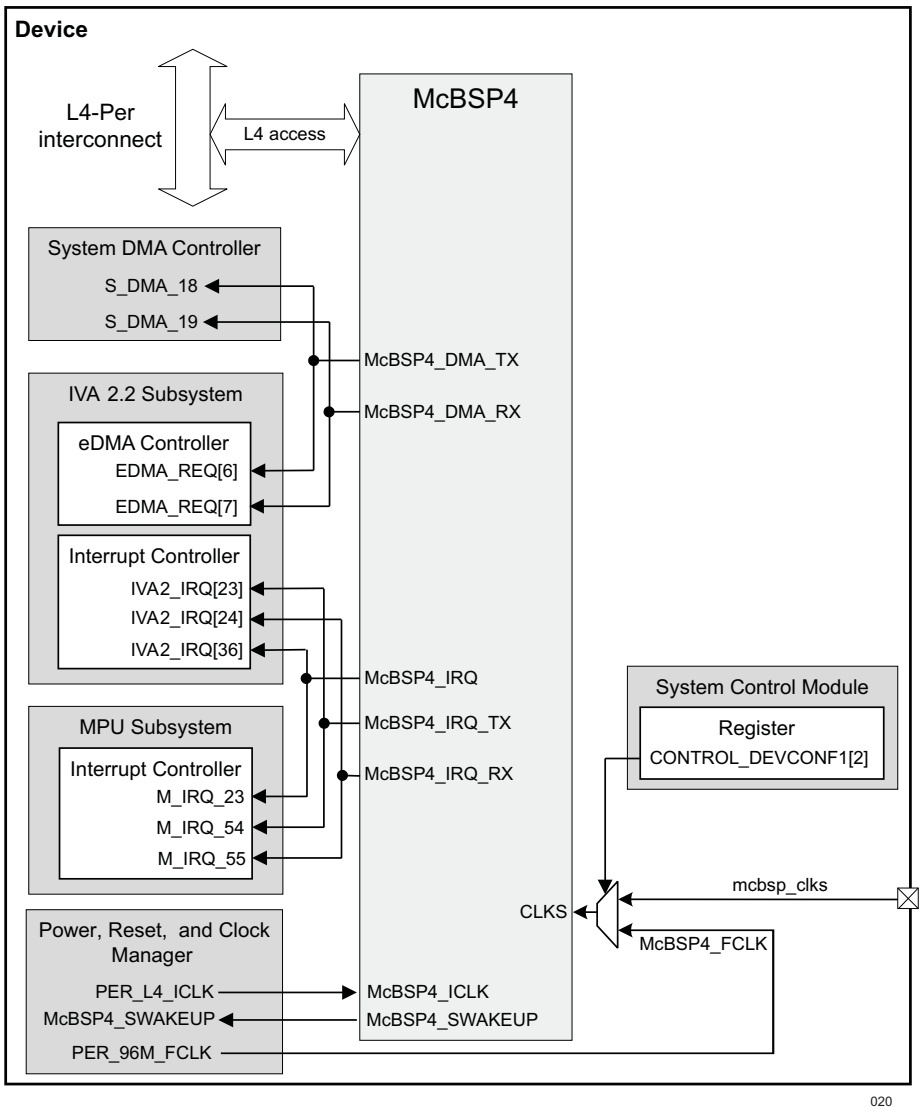
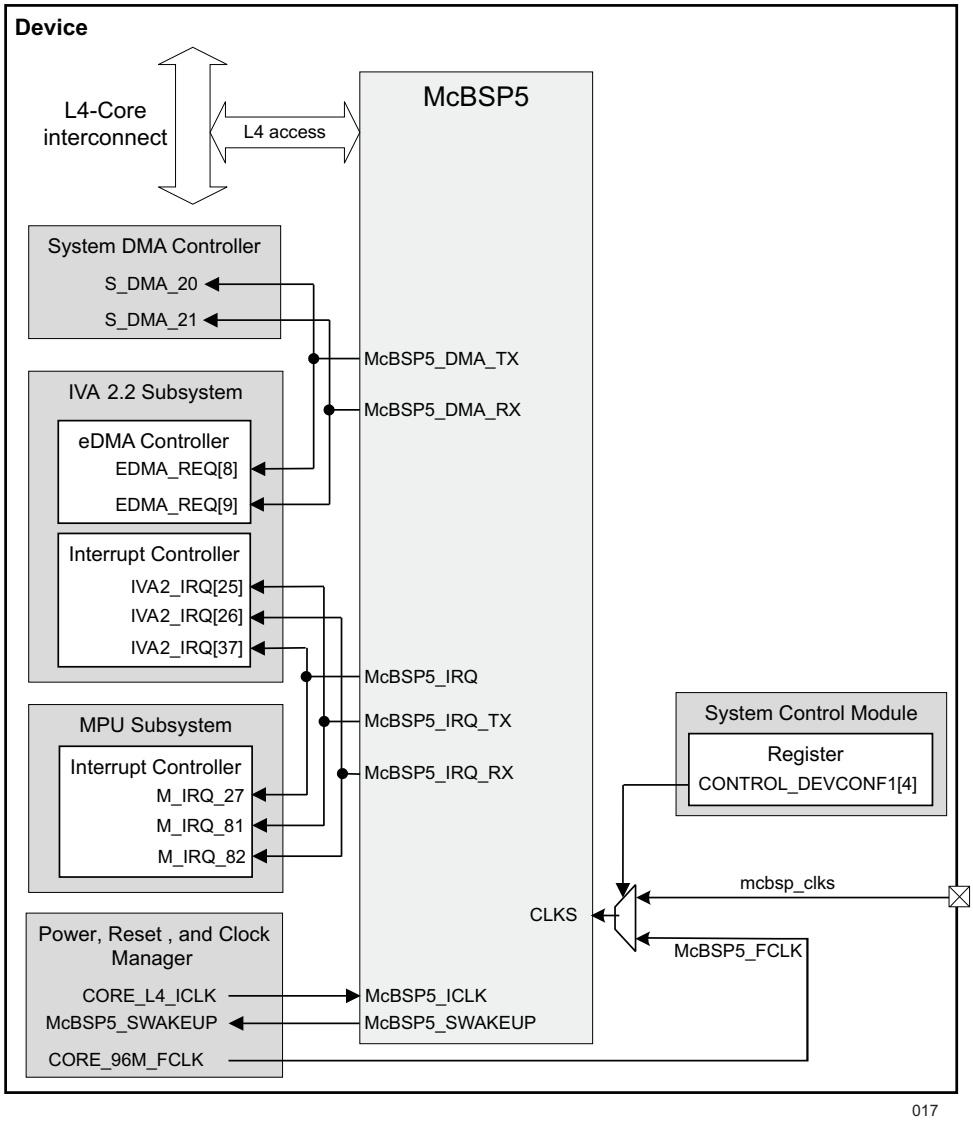


Figure 21-20. McBSP5 Integration



21.3.1 Signal Source Control

The FSR, CLKR and CLKS signals sources are defined by the system control module. The control registers of the system control module are used to select these signals sources.

21.3.1.1 McBSP1 Module (6 Pins Configuration)

The MCBSP1_CLKS bit of the CONTROL.CONTROL_DEVCONF0[2] register is used to select the McBSP1 module CLKS signal source:

- When set to ‘0’, the CLKS source is from the CORE_96M_FCLK
- When set to ‘1’, the CLKS source is from the mcbsp_clks pin

The MCBSP1_CLKR bit of the CONTROL_DEVCONF0[3] register is used to select the McBSP1 module CLKR signal source:

- When set to ‘0’, the CLKR source is from the CLKR input signal
- When set to ‘1’, the CLKR source is from the CLKX input signal

The MCBSP1_FSR bit of the CONTROL_DEVCONF0[4] register is used to select the McBSP1 module FSR signal source:

- When set to ‘0’, the FSR source is from the FSR input signal
- When set to ‘1’, the FSR source is from the FSX input signal

21.3.1.2 McBSP2, 3, 4, and 5 modules (4 pins configuration)

For these McBSPi modules, there are no external mcbspi_clkr and mcbspi_fxr pins (I=2, 3, 4, and 5).

Consequently, the system control module controls only the internal connection of CLKS input signals. The settings are explained in Section 21.3.2.2.2, Section 21.3.2.2.3, Section 21.3.2.2.4, and Section 21.3.2.2.5.

21.3.2 Clocking, Reset, and Power Management Scheme

21.3.2.1 Power Domain

McBSP1 and McBSP5 modules belong to the CORE domain, whereas the McBSP2, McBSP3, and McBSP4 modules belong to the PER (peripheral) domain. This separation of McBSP modules in two power domain allows major part of the device to be switched off while keeping active McBSP2, 3, and 4.

For more information about these power domains, see Chapter 4, *Power Reset and Clock Management*.

21.3.2.2 Clocks

There are two clock domains in the McBSP module:

- Functional clock domain
- Interface clock domain

Table 21-3. Clocking Signals Input to McBSP Module

Type	Name	Source	Description
Interface	McBSPi_ICLK	PER_L4_ICLK (McBSP2, 3, 4) (1)	The L4 interface clock is used for the module internal L4 interconnect slave interface and all depending parts of the Interface clock domain.
		CORE_L4_ICLK (McBSP1, 5) (1)	
Functional	CLKS	PER_96M_FCLK (McBSP2, 3, 4) (1)	McBSP module is running using either a functional clock generated internally (master mode) or supplied from its serial interface (slave mode) for the internal logic. Internal registers select the source of the functional clock and the divider ratio to apply. Functional clock after division in any mode is limited to maximum frequency divided by 2.
		CORE_96M_FCLK (McBSP1, 5) (1)	
		mcbsp_clks (external source common to all McBSP modules)	
Functional	CLKX	mcbspi_clkx (external source)	
Functional	CLKR (McBSP1 only)	mcbsp1_clkr (external source)	

(1) For more information about these sources, see Chapter 4, *Power Reset and Clock Management*.

21.3.2.2.1 McBSP1 Clocks

The McBSP1 module is clocked by a functional clock (CLKS, CLKX, or CLKR) and an interface clock (McBSP1_ICLK).

- The functional clock is used to generate control signals depending on the module internal configuration (see Section 21.4). For McBSP1 module, the functional clock comes from the CLKS signal, CLKX signal, or CLKR signal. The choice between these three clocks is defined by the SCLKME bit of the MCBSP1.MCBSPLP_PCR_REG[7] register and the CLKSM bit of the MCBSP1.MCBSPLP_SRGR2_REG[13] register.

  The CLKS signal of the McBSP1 module is linked to an internal clock (CORE_96M_FCLK) provided by PRCM, whereas the CLKS signal can also be linked to an external signal through the mcbsp_clks pin of the device boundary. The MCBSP1_CLKS bit of the CONTROL.CONTROL_DEVCONF0[2] register is used to select the McBSP1 module CLKS signal source:

  – 0: The CLKS source is from the CORE_96M_FCLK.

– 1: The CLKS source is from the mcbsp_clks pin.

For more information on this register , see Chapter 7, *System Control Module*.

Note: When the McBSP1 module does not require the PRCM functional clock anymore, the software can disable it at the PRCM level by setting the EN_MCBSP1 bit (PRCM.CM_FCLKEN1_CORE[9]) in the PRCM registers. The clock is effectively cut, provided the other modules that receive it do not require it. For more details, see Chapter 4, *Power Reset and Clock Management*.

At PRCM level, when all the conditions to shut-off CORE_96M_FCLK clock are met the PRCM automatically launches a **Hardware** handshake protocol to ensure McBSP1 is ready to have this clock switched off. Namely, the PRCM asserts an idle request to McBSP1. For more details, see Chapter 4, *Power Reset and Clock Management*.

The CLKX and CLKR signals are connected either by mcbsp1_clkx or mcbsp1_clkr pads. These signals are used like functional clocks by the intermediary of the sample rate generator (SRG).

- The McBSP1_ICLK runs at the L4 core interconnect clock speed. It is used to trigger access to the McBSP1 L4 interface and McBSP1 configuration interface via the MPU/IVA2.2 shared bus. It can also be an input clock for the McBSP sample-rate generator (clock divider), depending on the module configuration (see Section 21.4.3). Its source is the CORE_L4_ICLK signal.

Note: When the McBSP1 module does not require the interface clock anymore, the software can disable it at the PRCM level by setting the EN_MCBSP1 bit (PRCM.CM_ICLKEN1_CORE[9]) in the PRCM registers. The clock is effectively cut, provided the other modules that receive it do not require it. For more information, see Chapter 4, *Power Reset and Clock Management*.

At PRCM level, when all the conditions to shut-off CORE_L4_ICLK clock are met the PRCM automatically launches a hardware handshake protocol to ensure McBSP1 is ready to have this clock switched off. Namely, the PRCM asserts an idle request to McBSP1. For more details, see Chapter 4, *Power Reset and Clock Management*.

It is also possible to activate an autoidle mode for this clock (PRCM.CM_AUTOIDLE1_CORE[9] register AUTO_MCBSP1 bit set to 1). In this case, McBSP1_ICLK follows the CORE_L4 clock domain behavior on the device. For more information, see Chapter 4, *Power Reset and Clock Management*.

21.3.2.2.2 McBSP2 Clocks

The McBSP2 module is clocked by a functional clock (CLKS, CLKX or CLKR) and an interface clock (McBSP2_ICLK).

- The functional clock is used to generate control signals depending on the module internal configuration (see Section 21.4). For McBSP2 module, the functional clock comes from the CLKS signal CLKX signal, or CLKR signal. The choice between these three clocks is defined by the SCLKME bit of the MCBSP2.MCBSPLP_PCR_REG[7] register and the CLKSM bit of the MCBSP2.MCBSPLP_SRGR2_REG[13] register.

  The CLKS signal of the McBSP2 module is linked to an internal clock (PER_96M_FCLK) provided by PRCM, whereas the CLKS signal can also be linked to an external signal through the mcbsp_clks pin of the device boundary. The MCBSP2_CLKS bit of the CONTROL.CONTROL_DEVCONF0[6] register is used to select the McBSP2 module CLKS signal source:

  – 0: The CLKS source is from the PER_96M_FCLK.
  – 1: The CLKS source is from the mcbsp_clks pin.

  For more information, see Chapter 7, *System Control Module*.

Note: When the McBSP2 module does not require the functional clock anymore, the software can disable it at the PRCM level by setting the EN_MCBSP2 bit (PRCM.CM_FCLKEN_PER[0]) in the PRCM registers. The clock is effectively cut, provided the other modules that receive it do not require it. For more details, see Chapter 4, *Power Reset and Clock Management*.

At PRCM level, when all the conditions to shut-off PER_96M_FCLK clock are met the PRCM automatically launches a hardware handshake protocol to ensure McBSP2 is ready to have this clock switched off. Namely, the PRCM asserts an idle request to McBSP2. For more details, see Chapter 4, *Power Reset and Clock Management*.

Only, the CLKX signal is connected by mcbsp2_clkx pads. The CLKR signal is connected to the CLKX signal. These signals are used like functional clocks by the intermediary of the SRG.

- The McBSP2_ICLK runs at the L4 core interconnect clock speed. It is used to trigger access to the McBSP2 L4 interface and McBSP2 configuration interface via the MPU/IVA2.2 shared bus. It can also be an input clock for the McBSP sample-rate generator (clock divider), depending on the module configuration (see Section 21.4.3). Its source is either the PER_L4_ICLK signal.

Note: When the McBSP2 module does not require the interface clock anymore, the software can disable it at the PRCM level by setting the EN_MCBSP2 bit (PRCM.CM_ICLKEN_PER[0]) in the PRCM registers. The clock is effectively cut, provided the other modules that receive it do not require it. For more information, see Chapter 4, *Power Reset and Clock Management*.

At PRCM level, when all the conditions to shut-off PER_L4_ICLK clock are met the PRCM automatically launches a hardware handshake protocol to ensure McBSP2 is ready to have this clock switched off. Namely, the PRCM asserts an idle request to McBSP2. For more details, see Chapter 4, *Power Reset and Clock Management*.

It is also possible to activate an autoidle mode for this clock (PRCM.CM_AUTOIDLE_PER[0] register AUTO_MCBSP2 bit set to 1). In this case, McBSP2_ICLK follows the CORE_L4 clock domain behavior on the device. For more information, see Chapter 4, *Power Reset and Clock Management*.

21.3.2.2.3 McBSP3 Clocks

The McBSP3 module is clocked by a functional clock (CLKS, CLKX or CLKR) and an interface clock (McBSP3_ICLK).

- The functional clock is used to generate control signals depending on the module internal configuration (see Section 21.4). For McBSP3 module, the functional clock comes from the CLKS signal CLKX signal, or CLKR signal. The choice between these three clocks is defined by the SCLKME bit of the MCBSP3.MCBSPLP_PCR_REG[7] register and the CLKSM bit of the MCBSP3.MCBSPLP_SRGR2_REG[13] register.

  The CLKS signal of the McBSP3 module is linked to an internal clock (PER_96M_FCLK) provided by PRCM, whereas the CLKS signal can also be linked to an external signal through the mcbsp_clks pin of the device boundary. The MCBSP3_CLKS bit of the CONTROL.CONTROL_DEVCONF1[0] register is used to select the McBSP3 module CLKS signal source:

  – 0: The CLKS source is from the PER_96M_FCLK.
  – 1: The CLKS source is from the mcbsp_clks pin.

  For more information, see Chapter 7, *System Control Module*.

Note: When the McBSP3 module does not require the functional clock anymore, the software can disable it at the PRCM level by setting the EN_MCBSP3 bit (PRCM.CM_FCLKEN_PER[1]) in the PRCM registers. The clock is effectively cut, provided the other modules that receive it do not require it. For more details, see Chapter 4, *Power Reset and Clock Management*.

At PRCM level, when all the conditions to shut-off PER_96M_FCLK clock are met the PRCM automatically launches a hardware handshake protocol to ensure McBSP3 is ready to have this clock switched off. Namely, the PRCM asserts an idle request to McBSP3. For more details, see Chapter 4, *Power Reset and Clock Management*.

Only, the CLKX signal is connected by mcbsp3_clkx pads. The CLKR signal is connected to the CLKX signal. These signals are used like functional clocks by the intermediary of SRG.

- The McBSP3_ICLK runs at the L4 core interconnect clock speed. It is used to trigger access to the McBSP3 L4 interface and McBSP3 configuration interface via the MPU/IVA2.2 shared bus. It can also be an input clock for the McBSP sample-rate generator (clock divider), depending on the module configuration (see Section 21.4.3). Its source is either the PER_L4_ICLK signal.

Note: When the McBSP3 module does not require the interface clock anymore, the software can disable it at the PRCM level by setting the EN_MCBSP3 bit (PRCM.CM_ICLKEN_PER[1]) in the PRCM registers. The clock is effectively cut, provided the other modules that receive it do not require it. For more information, see Chapter 4, *Power Reset and Clock Management*.

At PRCM level, when all the conditions to shut-off PER_L4_ICLK clock are met the PRCM automatically launches a hardware handshake protocol to ensure McBSP3 is ready to have this clock switched off. Namely, the PRCM asserts an idle request to McBSP3. For more details, see Chapter 4, *Power Reset and Clock Management*.

It is also possible to activate an autoidle mode for this clock (PRCM.CM_AUTOIDLE_PER[1] register AUTO_MCBSP3 bit set to 1). In this case, McBSP3_ICLK follows the PER_L4 clock domain behavior on the device. For more information, see Chapter 4, *Power Reset and Clock Management*.

21.3.2.2.4 McBSP4 Clocks

The McBSP4 module is clocked by a functional clock (CLKS or CLKX) and an interface clock (McBSP4_ICLK).

- The functional clock is used to generate control signals depending on the module internal configuration (see Section 21.4). For McBSP4 module, the functional clock comes from the CLKS signal, the CLKX signal, or the CLKR signal. The choice between these three clocks is defined by the SCLKME bit of the MCBSP4.MCBSPLP_PCR_REG[7] register and the CLKSM bit of the MCBSP4.MCBSPLP_SRGR2_REG[13] register.

  The CLKS signal of the McBSP4 module is linked to an internal clock (PER_96M_FCLK) provided by PRCM, whereas the CLKS signal can also be linked to an external signal through the mcbsp_clks pin of the device boundary. The MCBSP4_CLKS bit of the CONTROL.CONTROL_DEVCONF1[2] register is used to select the McBSP4 module CLKS signal source:

  – 0: The CLKS source is from the PER_96M_FCLK.
  – 1: The CLKS source is from the mcbsp_clks pin.

  For more information, see Chapter 7, *System Control Module*.

Note: When the McBSP4 module does not require the functional clock anymore, the software can disable it at the PRCM level by setting the EN_MCBSP4 bit (PRCM.CM_FCLKEN_PER[2]) in the PRCM registers. The clock is effectively cut, provided the other modules that receive it do not require it. For more details, see Chapter 4, *Power Reset and Clock Management*.

At PRCM level, when all the conditions to shut-off PER_96M_FCLK clock are met the PRCM automatically launches a hardware handshake protocol to ensure McBSP4 is ready to have this clock switched off. Namely, the PRCM asserts an idle request to McBSP4. For more details, see Chapter 4, *Power Reset and Clock Management*.

Only the CLKX signal is connected by mcbsp4_clkx pads. The CLKR signal is connected to the CLKX signal. These signals are used like functional clocks by the intermediary of SRG.

- The McBSP4_ICLK runs at the L4 core interconnect clock speed. It is used to trigger access to the McBSP4 L4 interface and McBSP4 configuration interface via the MPU/IVA2.2 shared bus. It can also be an input clock for the McBSP sample-rate generator (clock divider), depending on the module configuration (see Section 21.4.3). Its source is either the PER_L4_ICLK signal.

Note: When the McBSP4 module does not require the interface clock anymore, the software can disable it at the PRCM level by setting the EN_MCBSP4 bit (PRCM.CM_ICLKEN_PER[2]) in the PRCM registers. The clock is effectively cut, provided the other modules that receive it do not require it. For more information, see Chapter 4, *Power Reset and Clock Management*.

At PRCM level, when all the conditions to shut-off PER_L4_ICLK clock are met the PRCM automatically launches a hardware handshake protocol to ensure McBSP4 is ready to have this clock switched off. Namely, the PRCM asserts an idle request to McBSP4. For more details, see Chapter 4, *Power Reset and Clock Management*.

It is also possible to activate an autoidle mode for this clock (PRCM.CM_AUTOIDLE_PER[2] register AUTO_MCBSP4 bit set to 1). In this case, McBSP4_ICLK follows the PER_L4 clock domain behavior on the device. For more information, see Chapter 4, *Power Reset and Clock Management*.

21.3.2.2.5 McBSP5 Clocks

The McBSP5 module is clocked by a functional clock (CLKS or CLKX) and an interface clock (McBSP5_ICLK).

- The functional clock is used to generate control signals depending on the module internal configuration (see Section 21.4). For McBSP5 module, the functional clock comes from the CLKS signal, the CLKX signal, or the CLKR signal. The choice between these three clocks is defined by the SCLKME bit of the MCBSP5.MCBSPLP_PCR_REG[7] register and the CLKSM bit of the MCBSP5.MCBSPLP_SRGR2_REG[13] register.

  The CLKS signal of the McBSP5 module is linked to an internal clock (CORE_96M_FCLK) provided by PRCM. The CLKS signal can also be linked to an external signal through the mcbsp_clks pin of the device boundary. The MCBSP5_CLKS bit of the CONTROL.CONTROL_DEVCONF1[4] register is used to select the McBSP5 module CLKS signal source:

  – 0: The CLKS source is from the CORE_96M_FCLK.
  – 1: The CLKS source is from the mcbsp_clks pin.

  For more information, see Chapter 7, *System Control Module*.

Note: When the McBSP5 module does not require the functional clock anymore, the software can disable it at the PRCM level by setting the EN_MCBSP5 bit (PRCM.CM_FCLKEN1_CORE[10]) in the PRCM registers. The clock is effectively cut, provided the other modules that receive it do not require it. For more details, see Chapter 4, *Power Reset and Clock Management*.

At PRCM level, when all the conditions to shut-off CORE_96M_FCLK clock are met the PRCM automatically launches a hardware handshake protocol to ensure McBSP5 is ready to have this clock switched off. Namely, the PRCM asserts an idle request to McBSP5. For more details, see Chapter 4, *Power Reset and Clock Management*.

Only, the CLKX signal is connected by mcbsp5_clkx pads. The CLKR signal is connected to the CLKX signal. These signals are used like functional clocks by the intermediary of SRG.

- The McBSP5_ICLK runs at the L4 core interconnect clock speed. It is used to trigger access to the McBSP5 L4 interface and McBSP5 configuration interface via the MPU/IVA2.2 shared bus. It can also be an input clock for the McBSP sample-rate generator (clock divider), depending on the module configuration (see Section 21.4.3). Its source is either the CORE_L4_ICLK signal.

Note: When the McBSP5 module does not require the interface clock anymore, the software can disable it at the PRCM level by setting the EN_MCBSP5 bit (PRCM.CM_ICLKEN1_CORE[10]) in the PRCM registers. The clock is effectively cut, provided the other modules that receive it do not require it. For more information, see Chapter 4, *Power Reset and Clock Management*.

At PRCM level, when all the conditions to shut-off CORE_L4_ICLK clock are met the PRCM automatically launches a hardware handshake protocol to ensure McBSP5 is ready to have this clock switched off. Namely, the PRCM asserts an idle request to McBSP5. For more details, see Chapter 4, *Power Reset and Clock Management*.

It is also possible to activate an autoidle mode for this clock (PRCM.CM_AUTOIDLE1_CORE[10] register AUTO_MCBSP5 bit set to 1). In this case, McBSP5_ICLK follows the CORE_L4 clock domain behavior on the device. For more information, see Chapter 4, *Power Reset and Clock Management*.

21.3.2.2.6 SIDETONE Clock

The SIDETONE feature, in the McBSP2 and McBSP3 modules, is clocked only by an interface clock (McBSP2_ICLK or McBSP3_ICLK).

CAUTION

See Section 21.3.2.2.2 and Section 21.3.2.2.3 for information on McBSP2_ICLK and McBSP3_ICLK clocks.

When the SIDETONE feature does not require the clock anymore, the software can disable it at the SIDETONE level by setting the McBSPi.ST_SYSCONFIG_REG[0] AUTOIDLE bit in SIDETONE registers. To conserve power, when SIDETONE feature is not active or there is no activity on SIDETONE feature, the McBSPi_ICLK clock supports an automatic gating that is enabled or disabled by setting the McBSPi.ST_SYSCONFIG_REG[0] AUTOIDLE bit.

- When this bit is asserted (set to 1), the McBSPi_ICLK clock auto-gating is enabled and this clock is disabled internally to the SIDETONE feature, thus reducing power consumption, but not to the McBSP module that contains this feature.
  After reset, the automatic clock gating is enabled; thus, this bit must be disabled by software for activated SIDETONE feature.
- When this bit is set to 0, the McBSPi_ICLK clock auto-gating is disabled and this clock is enabled. The SIDETONE feature can be used normally.

21.3.2.3 Hardware and Software Reset

McBSP1 and McBSP5 modules belong to the CORE domain and their reset signal is the CORE_RST signal from the PRCM module, whereas McBSP2, 3 and 4 modules belong to the PER domain and their reset signal is the PER_RST signal from the PRCM module.

For more details about these signals, see Chapter 4, *Power Reset and Clock Management*.

Table 21-4. Software Reset Signals to All McBSP Modules

Type	Bit Field	Register Source	Activation	Description
Software	SOFTRESET	MCBSPi.MCBSPLP_SYSCONFIG_REG[1]	Active high	McBSP global software reset
	RRST	MCBSPi.MCBSPLP_SPCR1_REG[0]	Active low	This resets and disables the receiver, including the RB.
	XRST	MCBSPi.MCBSPLP_SPCR2_REG[0]		This resets and disables the transmitter, including the XB.
	GRST	MCBSPi.MCBSPLP_SPCR2_REG[6]		SRG is reset
	FRST	MCBSPi.MCBSPLP_SPCR2_REG[7]		Frame-sync logic is reset. Frame-sync generated signal is not generated by the SRG

For more details about these signals, see Section 21.7. See Section 21.5.1.1, for a complete description of the McBSP initialization procedure.

21.3.2.4 Power Management

21.3.2.4.1 McBSP Operating States

Two operating states are defined for all the McBSP modules:

- ACTIVE state: The module is running synchronously on the interface and functional clock, interrupts and DMA requests can be generated according to the configuration (register, master or slave mode, etc) and the external signals.
- IDLE state: As part of the system power management, the PRCM module can request the McBSP modules to enter IDLE state. Depending on the configured acknowledgment mode: Force Idle, No Idle and Smart Idle modes, a McBSP module will effectively enter IDLE state or not. As soon as a McBSP module enters IDLE state, it doesn't have anymore activities except those unrelated to clock activity (for example wake-up features) and its clocks are likely to be switched off at PRCM level.

Note: Idle request and idle acknowledge are only internal signals, with no means to observe or to control. The signals generation and control is purely hardware (managed automatically by the PRCM and the McBSP module depending on the SIDLEMODE settings).

21.3.2.4.2 McBSP Acknowledgment Modes

During initialization or configuration of the McBSP module, the software must configure how the McBSP module will answer an IDLE solicitation from the PRCM module (that is, the way idle acknowledge will be asserted following an idle request assertion).

Each McBSP module can be configured via the MCBSPi.MCBSPLP_SYSCONFIG_REG[4:3] SIDLEMODE field as one of the following acknowledgment modes:

- Force Idle mode (SIDLEMODE bit = 0x0): An idle request is acknowledged unconditionally, regardless of the internal state of the module. The McBSP module immediately enters Idle state (no activity), interface and PRCM functional clock can be stopped, no interrupts and DMA requests can be generated. In this mode, the McBSP module freezes all the internal activity when the PRCM clocks are switched off by the PRCM module, leading to a potential loss of data.

CAUTION

In Force Idle mode, the wake-up feature is inhibited.

CAUTION

If the McBSP functional part, transmitter or receiver, is running within this period of time (the functional clock source is not the PRCM functional clock), the internal state of the McBSP module will not be idle (FSM states, processes, etc.), and when the McBSP module exits from the Force Idle state unexpected behavior may happen in both receiver and transmitter. To avoid this, both receive and transmit parts, must be disabled by software prior to idle request assertion (all functional clock external sources must be disabled).

- No Idle mode (SIDLEMODE bit = 0x1): An idle request is never acknowledged, meaning it will prevent the PRCM from switching off its related clocks and from putting in a lower power state than the power domain it belongs to. The McBSP module never enters Idle state (is active).
- Smart Idle mode (SIDLEMODE bit = 0x2): Acknowledgment to an idle request is given based on the internal activity of the McBSP module. The McBSP module is in a waiting state, interface / functional clocks can be stopped, no interrupt can be generated, a wake-up signal can be generated according to the configuration (See Section 21.3.2.4.4) and external signals.

Note: The value MCBSPi.MCBSPLP_SYSCONFIG_REG[4:3] SIDLEMODE field = 0x3 must not be used.

When configured in Smart Idle mode, McBSP module also offers an additional granularity on McBSPi_FCLK and McBSPi_ICLK gating. McBSPi.MCBSPLP_SYSCONFIG_REG[9:8] CLOCKACTIVITY bit field is used to determine which clock will be shut down (McBSPi_FCLK, McBSPi_ICLK, none of them or both of them).

CLOCKACTIVITY setting is used in the McBSP module to determine on which part of the module the conditions to acknowledge the PRCM idle request will be tested. As an example, if McBSPi_FCLK is said not to be shut down upon a PRCM idle request, this means McBSP module will only consider McBSPi_ICLK and the associated pending activities before acknowledging the request.

Note: Some of McBSP features are associated to McBSPi_ICLK and others to McBSPi_FCLK. Using the CLOCKACTIVITY along with the Smart Idle mode ensures that the features associated with the clock that will remain active are always enabled, even if McBSP has acknowledged an idle request. For more information, see Section 21.3.2.4.4.

Table 21-5 lists the value of the bit field and indicates if the interface (McBSPi_ICLK) and PRCM functional (McBSPi_FCLK) clocks can be switched-off or not when an idle request is received by McBSP module.

Table 21-5. State of Clocks When the Module is in Idle State

CLOCKACTIVITY Value	Interface Clock (McBSPi_ICLK)	PRCM Functional Clock (CORE_96M_FCLK or PER_96M_FCLK)
0b00	OFF	OFF
0b01	OFF	ON
0b10	ON	OFF
0b11	ON	ON

Note: OFF means this clock can be switched-off.

ON means this clock must be maintained during wake up period.

CAUTION

The PRCM module does not have any hardware mean to read CLOCKACTIVITY settings. It is thus software responsibility to ensure a consistent programming between McBSPi.MCBSPLP_SYSCONFIG_REG[9:8] CLOCKACTIVITY bit field and PRCM 96M_FCLK and L4_ICLK control bits (for more information about clocks, see Section 21.3.2.2, *Clocks*). If McBSP module is disabled in both CM_FCLKEN and CM_ICLKEN PRCM registers while CLOCKACTIVITY is set to 11, nothing prevents the PRCM module from asserting its idle request which will be acknowledged regardless of the features associated with the McBSP clocks. This may lead to unpredictable behaviors.

The software can disable all clocks at the McBSP module level by setting the IDLE_EN bit in McBSPi.MCBSPLP_PCR_REG[14] registers. The IDLE_EN bit allows stopping all the clocks in the McBSP module (legacy):

- When set to ‘0’, the McBSP module is running
- When set to ‘1’, the clocks in the McBSP module are shut off when both IDLE_EN =1 and his power domain is in idle mode.

21.3.2.4.3 Wake-Up Capability

When configured in Smart Idle mode, the sources for wake-up generation are a subset of the interrupt sources. The wake up sources are enabled by setting the McBSPi.MCBSPLP_SYSCONFIG_REG[2] ENAWAKEUP bit (wake up feature control):

- Set to ‘0’, wake up capability is disabled
- Set to ‘1’, wake up capability is enabled

The McBSPi_SWAKEUP signal is the McBSP module asynchronous wake-up signal sent to the PRCM module when a wake-up generation is requested.

The wake up configurations are defined by setting the corresponding bits in the McBSPi.MCBSPLP_WAKEUPEN_REG register.

21.3.2.4.3.1 Receive Wake-up

There are 4 receive possible wake up configurations:

- McBSPi.MCBSPLP_WAKEUPEN_REG[3] RRDYEN bit: The McBSP module asserts the McBSPi_SWAKEUP request when the receive buffer reaches the high threshold value (RTHRESHOLD value + 1) of the McBSPi.MCBSPLP_THRSH1_REG register. If the McBSPi.MCBSPLP_IRQENABLE_REG[3] RRDYEN bit is set to 1, the McBSP module sends an interrupt (McBSPi_IRQ) request to the MPU or IVA 2.2 subsystems when exiting from idle mode (interrupt will be asserted once the McBSPi.MCBSPLP_IRQSTATUS_REG[3] RRDY bit changes from ‘0’ to ‘1’, indicating that received data is ready to be read).
- McBSPi.MCBSPLP_WAKEUPEN_REG[2] REOFEN bit: The McBSP module asserts the McBSPi_SWAKEUP request at the end of the frame. If the McBSPi.MCBSPLP_IRQENABLE_REG[2] REOFEN bit is set to 1, the McBSP module sends an interrupt (McBSPi_IRQ) request to the MPU or IVA 2.2 subsystems when exiting idle mode.
- McBSPi.MCBSPLP_WAKEUPEN_REG[1] RFSREN bit: The McBSP module sends a McBSPi_SWAKEUP request to the PRCM module when a receive frame-sync pulse is detected while the McBSP module is in idle mode. If the McBSPi.MCBSPLP_IRQENABLE_REG[1] RFSREN bit is set to 1, the McBSP module sends an interrupt (McBSPi_IRQ) request to the MPU or IVA 2.2 subsystems when exiting idle mode.

- McBSPi.MCBSPLP_WAKEUPEN_REG[0] RSYNCERREN bit: The McBSP module asserts the McBSPi_SWAKEUP request when an unexpected receive frame-sync pulse is detected. If the McBSPi.MCBSPLP_IRQENABLE_REG[0] RSYNCERREN bit is set to 1, the McBSP module sends an interrupt (McBSPi_IRQ) request o the MPU or IVA 2.2 subsystems when exiting from idle mode (interrupt is asserted once the McBSPi.MCBSPLP_IRQSTATUS_REG[0] RSYNCERR bit changes from ‘0’ to ‘1’, indicating that a receive error occurred).

21.3.2.4.3.2 Transmit Wakeup

For transmit, there are also 5 possible wake-up configuration scenarios:

- McBSPi.MCBSPLP_WAKEUPEN_REG[14] XEMPTYEOFEN bit: The McBSP module asserts the McBSPi_SWAKEUP request when a complete frame was transmitted and the transmit buffer is empty. If the McBSPi.MCBSPLP_IRQENABLE_REG[14] XEMPTYEOFEN bit is set to 1, the McBSP module sends an interrupt (McBSPi_IRQ) request to the MPU or IVA 2.2 subsystems when exiting from idle mode.
- McBSPi.MCBSPLP_WAKEUPEN_REG[10] XRDYEN bit: The McBSP module asserts the McBSPi_SWAKEUP request when the transmit buffer reaches the high threshold value (XTHRESHOLD value + 1) of the McBSPi.MCBSPLP_THRSH2_REG register. If the McBSPi.MCBSPLP_IRQENABLE_REG[10] XRDYEN bit is set to 1, the McBSP module sends an interrupt (McBSPi_IRQ) request to the MPU or IVA 2.2 subsystems when exiting from idle mode (interrupt is asserted once the McBSPi.MCBSPLP_IRQSTATUS_REG[10] XRDY bit changes from ‘0’ to ‘1’, indicating that transmit buffer data is ready to accept new data).
- McBSPi.MCBSPLP_WAKEUPEN_REG[9] XEOFEN bit: The McBSP module asserts the McBSPi_SWAKEUP request at the end of the frame. If the McBSPi.MCBSPLP_IRQENABLE_REG[9] XEOFEN bit is set to 1, the McBSP module sends an interrupt (McBSPi_IRQ) request to the MPU or IVA 2.2 subsystems when exiting from idle mode.
- McBSPi.MCBSPLP_WAKEUPEN_REG[8] XFSXEN bit: The McBSP module sends a McBSPi_SWAKEUP request when a transmit frame-sync pulse is detected while the module is in idle mode. If the McBSPi.MCBSPLP_IRQENABLE_REG[8] XFSXEN bit is set to 1, the McBSP module sends an interrupt (McBSPi_IRQ) request to the MPU or IVA 2.2 subsystems when exiting from idle mode.
- McBSPi.MCBSPLP_WAKEUPEN_REG[7] XSYNCERREN bit: The McBSP module asserts the McBSPi_SWAKEUP request when an unexpected transmits frame-sync pulse is detected. If the McBSPi.MCBSPLP_IRQENABLE_REG[7] XSYNCERREN bit is set to 1, the McBSP module sends an interrupt (McBSPi_IRQ) request to the MPU or IVA 2.2 subsystems when exiting from idle mode (interrupt will be asserted once the McBSPi.MCBSPLP_IRQSTATUS_REG[7] XSYNCERR bit changes from ‘0’ to ‘1’, indicating that a transmit error occurred).

21.3.2.4.3.3 Notes

When mcbsp1_fsr/mcbspi_fsx pins is configured as an output, the FSR/FSX wake-up generation makes no sense (the module cannot be in Smart Idle mode).

Detection of RSYNCERR/XSYNCERR during idle mode can be used only when mcbsp1_fsr/mcbspi_fsx pins is configured as an input and the remote system knows to assert such an error to trigger the wake up of the McBSP module.

The module does not implement interrupt request (IRQ) assertion when configured as (GPIO). Pins that can be used to accept input signals and/or send output signals but are not linked to specific uses); also a wake up capability in this mode is not available.

21.3.2.4.4 Analysis of the Receiver Smart Idle Behavior

The analysis of the power mode behavior is shown in Table 21-6 :

In this table, the CLKRM bit is in the McBSPi.MCBSPLP_PCR_REG register on position 8, CLKXM bit in the McBSPi.MCBSPLP_PCR_REG[9] register, and CLOCKACTIVITY bit in the McBSPi.MCBSPLP_SYSCONFIG_REG[9:8] register.

The value X signifies that the bit value is not significant.

Table 21-6. McBSP Smart Idle Mode Configuration Behavior

CLKRM Bit	CLKXM Bit	McBSP Mode	Source of Functional Clock	CLOCKACTIVITY Bit	Behavior
0	0	Slave	outside	0bXX	The module acknowledges the idle request as soon as there is no pending DMA, interrupt request or transmit buffer threshold synchronization (only when wake-up event is set on transmit threshold reached), regardless of the CLOCKACTIVITY settings or receive and transmit activity.
0	1	Transmit Master	McBSPi_ICLK	0b0X	The McBSP will not acknowledge the idle request unless: • The transmit part is disabled (XDISABLE) or under software reset (XRST) and the receive part is not using the transmit loop-back clock (CLKR is not connected to the CLKX input pin). • Both transmit and receive parts are disabled (XDISABLE/RDISABLE) or under software reset (XRST/RRST) The idle acknowledge is asserted as soon as there is no pending DMA, interrupt request or transmit/receive buffer threshold synchronization (only when wake-up event is set on transmit/receive threshold reached) and the pending transmit and/or receive frames were completed in case of transmit and/or receive disable.
			CLKS	0bX0	
			McBSPi_ICLK	0b1X	The module acknowledges the idle request as soon as there is no pending DMA, interrupt request or transmit/receive buffer threshold synchronization (only when wake-up event is set on transmit/receive threshold reached).
			CLKS	0bX1	
			CLKR (outside)	0bXX	The module acknowledges the idle request as soon as there is no pending DMA, interrupt request or transmit/receive buffer threshold synchronization (only when wake-up event is set on transmit/receive threshold reached), regardless of the CLOCKACTIVITY settings.
1	0	Receive master	McBSPi_ICLK	0b0X	The McBSP will not acknowledge the idle request unless: The receive part is disabled (RDISABLE) or under software reset (RRST) The Idle acknowledge is asserted as soon as there is no pending DMA, interrupt request or transmit/receive buffer threshold synchronization (only when wake-up event is set on transmit/receive threshold reached) and the pending transmit and/or receive frames where completed in case of transmit and/or receive disable.
			CLKS	0bX0	
			McBSPi_ICLK	0b1X	The module acknowledges the idle request as soon as there is no pending DMA, interrupt request or transmit/receive buffer threshold synchronization (only when wake-up event is set on transmit/receive threshold reached).
			CLKS	0bX1	
			CLKX	0bXX	When CLKX is used as source (functional clock is provided from outside) then the module acknowledges the idle request as soon as there is no pending DMA, interrupt request or transmit/receive buffer threshold synchronization (only when wake-up event is set on transmit/receive threshold reached), regardless of the CLOCKACTIVITY settings.

Table 21-6. McBSP Smart Idle Mode Configuration Behavior (continued)

CLKRM Bit	CLKXM Bit	McBSP Mode	Source of Functional Clock	CLOCKACTIVITY Bit	Behavior
1	1	Transmit and Receive Master	McBSPi_ICLK	0b0X	The McBSP will not acknowledge the idle request unless: Both transmit and receive parts are disabled (XDISABLE/RDISABLE) or under software reset (XRST/RRST) The idle acknowledge is asserted as soon as there is no pending DMA, interrupt request or transmit/receive buffer threshold synchronization (only when wake-up event is set on transmit/receive threshold reached) and the pending transmit and/or receive frames were completed in case of transmit and/or receive disable. Note that no wake-up event is available in this mode since the entire McBSP and remote device activity is frozen.
			CLKS	0bX0	
			McBSPi_ICLK	0b1X	The module acknowledges the idle request as soon as there is no pending DMA, interrupt request or transmit/receive buffer threshold synchronization (only when wake-up event is set on transmit/receive threshold reached).
			CLKS	0bX1	

Note: The RFSREN/XFSXEN mode is suitable for wake-up generation when both clocks (PRCM functional and interface) are switched off and mcbsp1_fsr/mcbspi_fsx pins is configured as input. The frame-sync pulse is asynchronously detected during idle.

The RSYNCERREN/XSYNCERREN mode can be used to wake-up the McBSP module only by a remote module implementing such a feature, to trigger a wake up. This mode requires functional clock to be active.

21.3.3 Hardware Requests

21.3.3.1 DMA Requests

The DMA requests are shared between the IVA2.2 subsystem DMA controller (eDMA) and system DMA controller (sDMA). Each of the five McBSP modules can generate two DMA events:

- McBSPi_DMA_TX : McBSPi module transmit request
- McBSPi_DMA_RX : McBSPi module receive request

The following table summaries the DMA events with the mapping on both DMA controllers.

Table 21-7. McBSP DMA Requests

Request Name	Mapping	Destination	Description
McBSP1_DMA_TX	EDMA_REQ[0] S_DMA_30	IVA2.2 subsystem DMA controller system DMA controller	write (or transmit) request
McBSP1_DMA_RX	EDMA_REQ[1] S_DMA_31	IVA2.2 subsystem DMA controller system DMA controller	read (or receive) request
McBSP2_DMA_TX	EDMA_REQ[2] S_DMA_32	IVA2.2 subsystem DMA controller system DMA controller	write (or transmit) request
McBSP2_DMA_RX	EDMA_REQ[3] S_DMA_33	IVA2.2 subsystem DMA controller system DMA controller	read (or receive) request
McBSP3_DMA_TX	EDMA_REQ[4] S_DMA_16	IVA2.2 subsystem DMA controller system DMA controller	write (or transmit) request
McBSP3_DMA_RX	EDMA_REQ[5] S_DMA_17	IVA2.2 subsystem DMA controller system DMA controller	read (or receive) request
McBSP4_DMA_TX	EDMA_REQ[6] S_DMA_18	IVA2.2 subsystem DMA controller system DMA controller	write (or transmit) request
McBSP4_DMA_RX	EDMA_REQ[7] S_DMA_19	IVA2.2 subsystem DMA controller system DMA controller	read (or receive) request
McBSP5_DMA_TX	EDMA_REQ[8] S_DMA_20	IVA2.2 subsystem DMA controller system DMA controller	write (or transmit) request
McBSP5_DMA_RX	EDMA_REQ[9] S_DMA_21	IVA2.2 subsystem DMA controller system DMA controller	read (or receive) request

21.3.3.2 Interrupt Requests

21.3.3.2.1 McBSP Interrupt Requests

Each of the five McBSP modules can generate three interrupts, shared between the MPU subsystem and IVA2.2 subsystem interrupt controllers:

- McBSPi_IRQ: McBSPi module common interrupt request
- McBSPi_IRQ_TX: McBSPi module transmit interrupt request
- McBSPi_IRQ_RX: McBSPi module receive interrupt request

The following tables list the interrupt requests with the mapping.

Table 21-8. McBSP Common Interrupt Requests

Request Name	Mapping	Destination
McBSP1_IRQ	IVA2_IRQ[33] M_IRQ_16	IVA2.2 subsystem interrupt controller MPU subsystem interrupt controller
McBSP2_IRQ	IVA2_IRQ[34] M_IRQ_17	IVA2.2 subsystem interrupt controller MPU subsystem interrupt controller
McBSP3_IRQ	IVA2_IRQ[35] M_IRQ_22	IVA2.2 subsystem interrupt controller MPU subsystem interrupt controller
McBSP4_IRQ	IVA2_IRQ[36] M_IRQ_23	IVA2.2 subsystem interrupt controller MPU subsystem interrupt controller

Table 21-8. McBSP Common Interrupt Requests (continued)

Request Name	Mapping	Destination
McBSP5_IRQ	IVA2_IRQ[37] M_IRQ_27	IVA2.2 subsystem interrupt controller MPU subsystem interrupt controller

Table 21-9. McBSP Transmit Interrupt Requests

Request Name	Mapping	Destination
McBSP1_IRQ_TX	IVA2_IRQ[16] M_IRQ_59	IVA2.2 subsystem interrupt controller MPU subsystem interrupt controller
McBSP2_IRQ_TX	IVA2_IRQ[19] M_IRQ_62	IVA2.2 subsystem interrupt controller MPU subsystem interrupt controller
McBSP3_IRQ_TX	IVA2_IRQ[21] M_IRQ_89	IVA2.2 subsystem interrupt controller MPU subsystem interrupt controller
McBSP4_IRQ_TX	IVA2_IRQ[23] M_IRQ_54	IVA2.2 subsystem interrupt controller MPU subsystem interrupt controller
McBSP5_IRQ_TX	IVA2_IRQ[25] M_IRQ_81	IVA2.2 subsystem interrupt controller MPU subsystem interrupt controller

Table 21-10. McBSP Receive Interrupt Requests

Request Name	Mapping	Destination
McBSP1_IRQ_RX	IVA2_IRQ[17] M_IRQ_60	IVA2.2 subsystem interrupt controller MPU subsystem interrupt controller
McBSP2_IRQ_RX	IVA2_IRQ[20] M_IRQ_63	IVA2.2 subsystem interrupt controller MPU subsystem interrupt controller
McBSP3_IRQ_RX	IVA2_IRQ[22] M_IRQ_90	IVA2.2 subsystem interrupt controller MPU subsystem interrupt controller
McBSP4_IRQ_RX	IVA2_IRQ[24] M_IRQ_55	IVA2.2 subsystem interrupt controller MPU subsystem interrupt controller
McBSP5_IRQ_RX	IVA2_IRQ[26] M_IRQ_82	IVA2.2 subsystem interrupt controller MPU subsystem interrupt controller

An event can generate an interrupt request when the corresponding mask bit in the McBSPi.MCBSPLP_IRQENABLE_REG register is set to '1'.

Table 21-11 and Table 21-12 summarize the events causing the generation of an interrupt request.

Table 21-11. McBSP Transmit Interrupt Events

Event Name	Status Bit	Mask Bit	Description
Transmit buffer empty at end of frame	McBSPi.MCBSPLP_IRQSTATUS_REG[14] XEMPTYEOF	McBSPi.MCBSPLP_IRQENABLE_REG [14] XEMPTYEOFEN	This event happens when a complete frame was transmitted and the transmit buffer is empty .
Overflow	McBSPi.MCBSPLP_IRQSTATUS_REG[12] XOVFLSTAT	McBSPi.MCBSPLP_IRQENABLE_REG [12] XOVFLEN	This event happens when transmit data buffer is full and a new data is written in this buffer. The new data written is discarded.
Underflow	McBSPi.MCBSPLP_IRQSTATUS_REG[11] XUNDFLSTAT	McBSPi.MCBSPLP_IRQENABLE_REG [11] XUNDFLEN	This event happens when transmit data buffer is empty and a new data is required to be transmitted.
Threshold reached	McBSPi.MCBSPLP_IRQSTATUS_REG[10] XRDY	McBSPi.MCBSPLP_IRQENABLE_REG [10] XRDYEN	This event happens when the transmit buffer free locations are equal or above the THRSH2_REG value.
End of Frame	McBSPi.MCBSPLP_IRQSTATUS_REG[9] XEOF	McBSPi.MCBSPLP_IRQENABLE_REG [9] XEOFLEN	This event happens when a complete frame was transmitted.
Frame-sync	McBSPi.MCBSPLP_IRQSTATUS_REG[8] XFSX	McBSPi.MCBSPLP_IRQENABLE_REG [8] XFSXEN	This event happens when a new transmit frame synchronization is asserted.

Table 21-11. McBSP Transmit Interrupt Events (continued)

Event Name	Status Bit	Mask Bit	Description
Frame-sync error	McBSPi.MCBSPLP_IRQSTATUS_REG[7] XSYNCERR	McBSPi.MCBSPLP_IRQENABLE_REG [7] XSYNCERREN	This event happens when a transmit frame synchronization error is detected.

Table 21-12. McBSP Receive Interrupt Events

Event Name	Status bit	Mask bit	Description
Overflow	McBSPi.MCBSPLP_IRQSTATUS_REG[5] ROVFLSTAT	McBSPi.MCBSPLP_IRQENABLE_REG[5] ROVFLEN	This event happens when receive data buffer is full and a new data is written in this buffer. The new data written is discarded.
Underflow	McBSPi.MCBSPLP_IRQSTATUS_REG[4] XUNDFLSTAT	McBSPi.MCBSPLP_IRQENABLE_REG[4] RUNDFLEN	This event happens when read operation is performed to the receive data register while receive buffer is empty; data read while underflow condition is undefined.
Threshold reached	McBSPi.MCBSPLP_RQSTATUS_REG[3] RRDY	McBSPi.MCBSPLP_IRQENABLE_REG[3] RRDYEN	This event happens when the receive buffer occupied locations are equal or above the THRSH1_REG value.
End of Frame	McBSPi.MCBSPLP_IRQSTATUS_REG[2] REOF	McBSPi.MCBSPLP_IRQENABLE_REG[2] REOFLEN	This event happens when a complete frame was received.
Frame-sync	McBSPi.MCBSPLP_IRQSTATUS_REG[1] RFSR	McBSPi.MCBSPLP_IRQENABLE_REG[1] RFSREN	This event happens when a new receive frame synchronization is asserted.
Frame-sync error	McBSPi.MCBSPLP_IRQSTATUS_REG[0] RSYNCERR	McBSPi.MCBSPLP_IRQENABLE_REG[0] RSYNCERREN	This event happens when a receive frame synchronization error is detected.

Once an interrupt request is generated, the software must read the McBSPi.MCBSPLP_IRQSTATUS_REG register to check what event has caused the interrupt request generation, and acknowledge each processed event by writing a 1 to the corresponding bit in the McBSPi.MCBSPLP_IRQSTATUS_REG register.

Note: All Status bits can be cleared in two ways:

- If the corresponding mask bit is set to '1' (interrupt generation enabled), the status bit is cleared by writing a '1'.
- If the corresponding mask bit is cleared to '0' (interrupt generation disabled), the status bit is cleared when a new start or stop condition is detected on the receiver/transmitter.

21.3.3.2.2 SIDETONE_McBSP Interrupt Requests

The McBSP2 and McBSP3 can generate another interrupt to the MPU subsystem interrupt controller:

- **ST_McBSPi_IRQ:** SIDETONE interrupt request for McBSPi module (where I = 2, 3).

Table 21-13. SIDETONE_McBSP Interrupt Requests

Request Name	Mapping	Destination
ST_McBSP2_IRQ	M_IRQ_4	MPU subsystem interrupt controller
ST_McBSP3_IRQ	M_IRQ_5	MPU subsystem interrupt controller

Table 21-14. SIDETONE_McBSP Interrupt Events

Event Name	Status Bit	Mask Bit	Description
Over-run	McBSPi.ST_IRQSTATUS_REG[0] OVRRERROR	McBSPi.ST_IRQENABLE_REG[0] OVRRERROREN	This event happens when a new data to be processed arrives before the previous one has ended.

Once an interrupt request has been generated, the software must read the McBSPi.ST_IRQSTATUS_REG register to check what event caused the interrupt request generation, and acknowledge each processed event by writing a 1 to the corresponding bit in the McBSPi.ST_IRQSTATUS_REG register.

Note: The McBSPi.ST_IRQSTATUS_REG[0] OVRRERROR status bit can be cleared in two ways:

- If the McBSPi.ST_IRQENABLE_REG[0] OVRRERROREN mask bit is set to '1' (interrupt generation enabled), the status bit is cleared by writing a '1'.
- If the McBSPi.ST_IRQENABLE_REG[0] OVRRERROREN mask bit is cleared to '0' (interrupt generation disabled), the status bit is cleared when a new start or stop condition is detected on the SIDETONE channels.

21.4 McBSP Functional Description

This section is a functional description of the McBSP module.

21.4.1 Block Diagram

Figure 21-21, Figure 21-22, and Figure 21-23 show functional block diagrams of the five instances of the McBSP modules.

These figures regroup the McBSP modules by categories:

- McBSP module without audio buffer and SIDETONE core: McBSP1, McBSP4 and McBSP5
- McBSP module with audio buffer and SIDETONE core: McBSP2
- McBSP module without audio buffer, but with SIDETONE core: McBSP3

Figure 21-21. McBSP1, McBSP4 and McBSP5 Block Diagrams

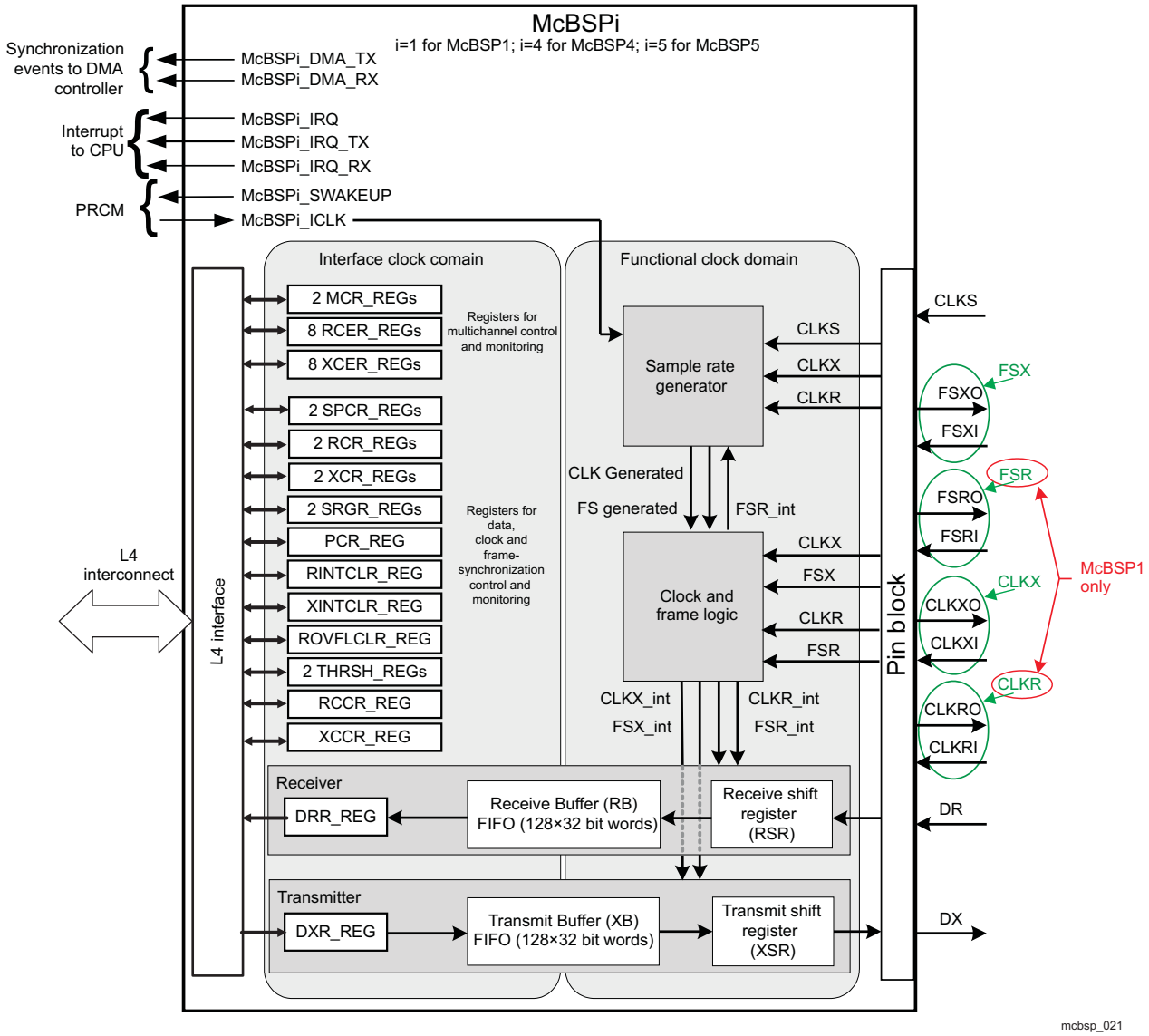


Figure 21-22. McBSP2 Block Diagram

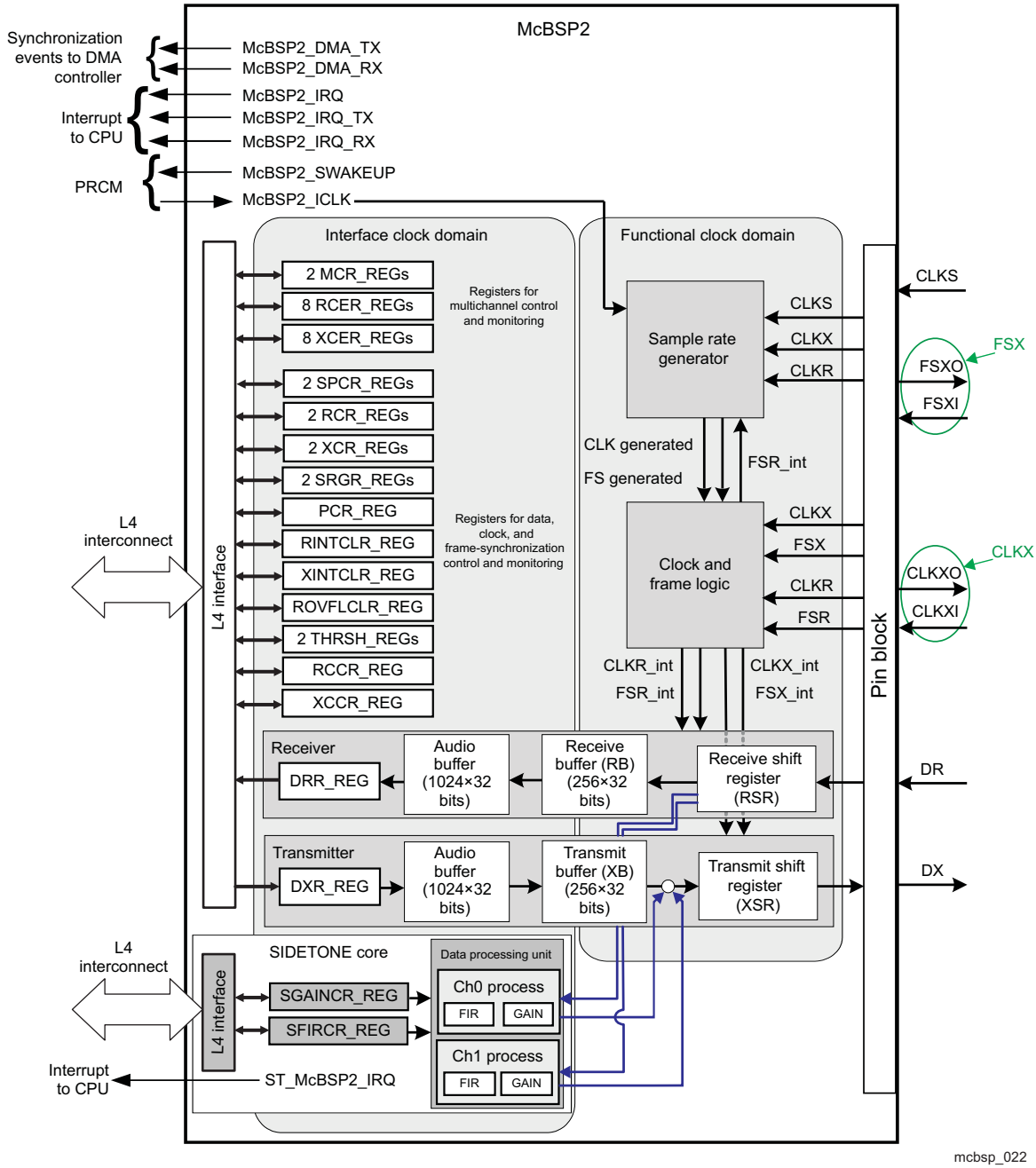
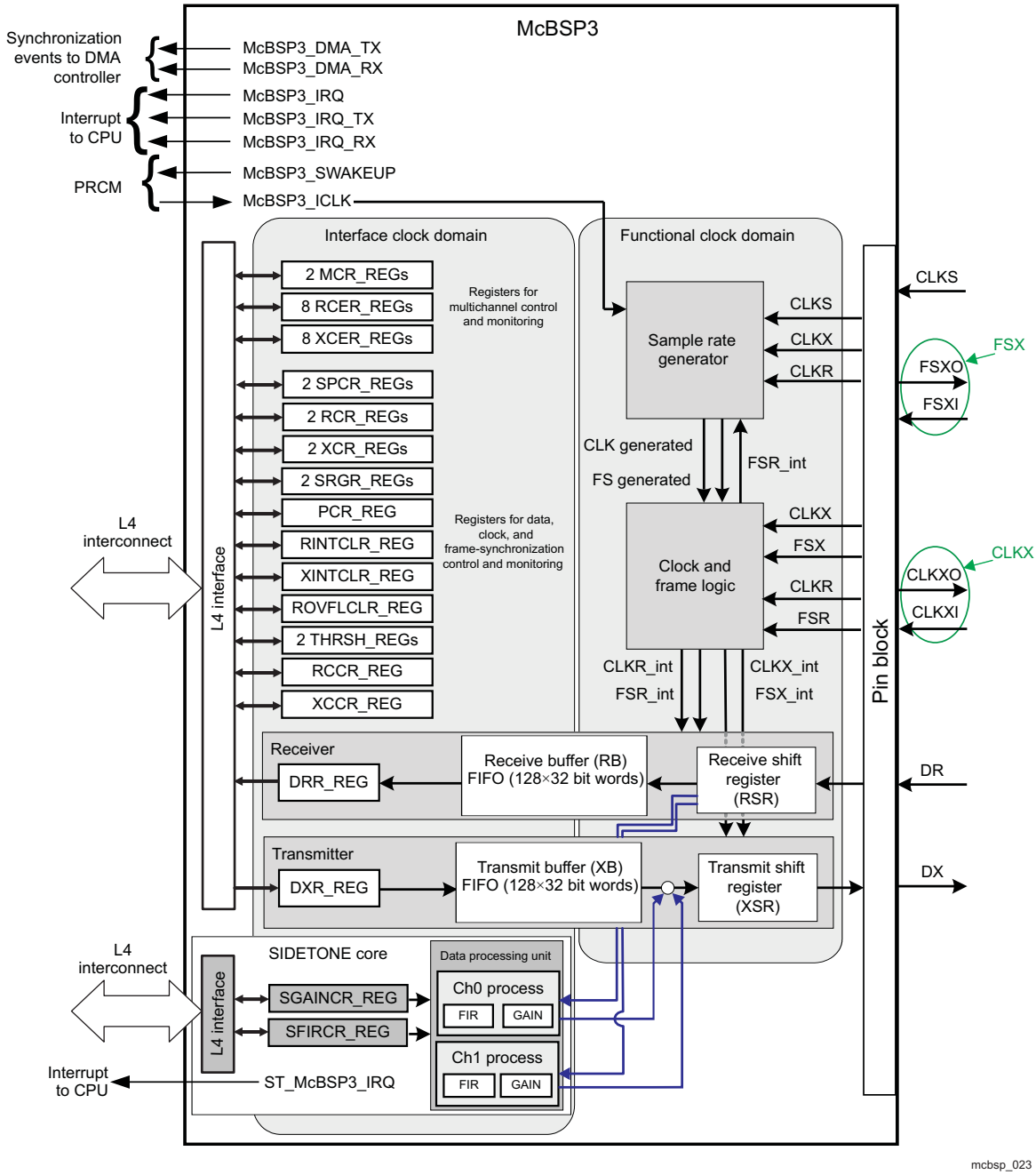


Figure 21-23. McBSP3 Block Diagram



21.4.2 McBSP Data Transfer Process

For McBSP1, McBSP3, McBSP4, and McBSP5 modules, receive and transmit operations are triple-buffered (512 bytes buffers organized in 32-bit words are used).

For the McBSP2 module, receive and transmit operations are quadruple-buffered (5K bytes buffers organized in 32-bit words are used). Receive and transmit buffers are separated in two memories: The same buffer as the others McBSP modules (1K bytes) and an audio buffer (4K bytes). The audio buffer is the largest one and is clocked by the interface clock, while the other buffer is used for synchronizing data to/from the audio buffer with the functional transmit/receive clock domains. Figure 21-25 shows this implementation in the McBSP2 data transfer paths.

All registers of McBSP data transfer paths are 32-bits wide. Figure 21-24 and Figure 21-25 illustrate the McBSP data transfer paths.

CAUTION

The McBSP registers (DRR_REG and DXR_REG) are limited to 32-bit data accesses (L4 Interconnect). 16-bit and 8-bit is not allowed and can corrupt register content.

Figure 21-24. McBSP Data Transfer Paths

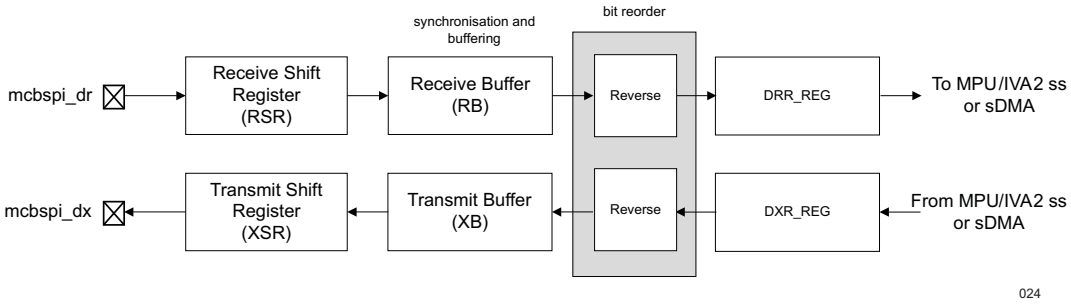
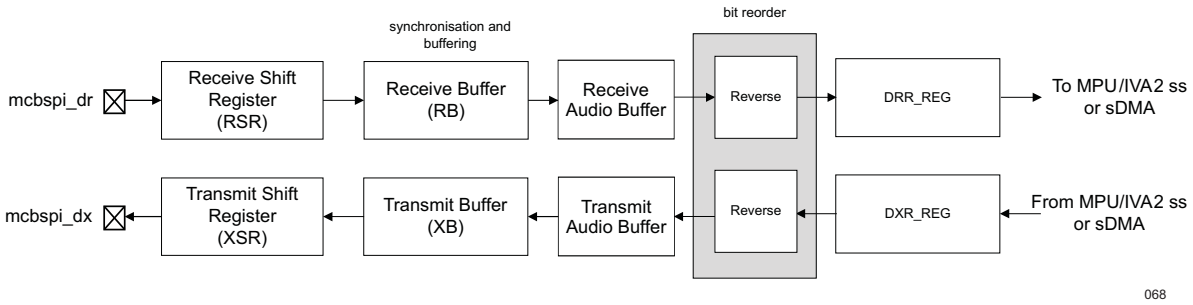


Figure 21-25. McBSP2 Data Transfer Paths



21.4.2.1 Data Transfer Process for 8- / 12- / 16- / 20- / 24- / 32-bits Long Words

CAUTION

For each data word length, one data occupies one 32-bit buffer word.

Receive data arrives on the mcbspi_dr pin and is shifted into the receive shift register (RSR). When a full word (depending on the data length configuration) is received, the content of the shift register is copied into the receive buffer (RB) if it is not full. When the RB threshold is reached the McBSP module asserts DMA or interrupt request and the RB content is then transferred (the sDMA or the eDMA controller reads the data receive register McBSPi.MCBSPLP_DRR_REG).

Transmit data is written by the MPU subsystem, the IVA2.2 subsystem or the DMA controller to data transmit register (McBSPi.MCBSPLP_DXR_REG) using the McBSPi.MCBSPLP_SPCR2_REG[1] XRDY bit enable input (when a byte is not enabled, the byte value in the memory will contain the previous written value). If there is no previous data in transmit shift register (XSR), the value from the transmit buffer (XB) is copied to XSR; otherwise, the content is copied to the XSR when the last bit of the previous data is shifted out on the mcbspi_dx pin.

21.4.2.2 Bit Reordering (Option to Transfer LSB First)

Generally, the McBSP module transmits or receives all data with the MSB first. However, some data protocols require the LSB to be transferred first.

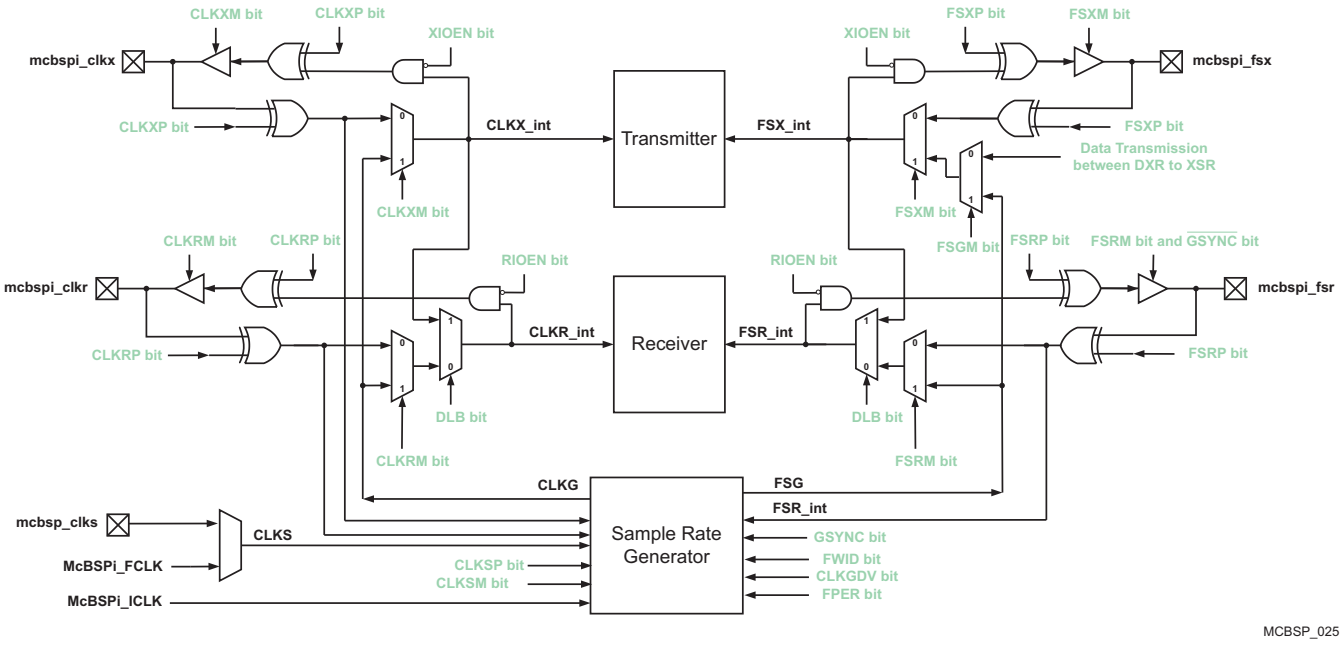
If you set McBSPi.MCBSPLP_XCR2_REG[4:3] XREVERSE=0b01, the bit ordering of the data words is reversed (LSB first) before being sent to the serial port. If you set McBSPi.MCBSPLP_RCR2_REG[4:3] RREVERSE=0b01, the bit ordering of the data words is reversed during reception (LSB first).

This feature is available for all the data formats from 8 up to 32-bit data length.

21.4.2.3 Clocking and Framing Data

This section explains basic concepts and terminology important for understanding how McBSP data transfers are timed and delimited.

Figure 21-26. Conceptual Block Diagram for Clock and Frame Generation



21.4.2.3.1 Clocking

Data is shifted one bit at a time from the mcbspi_dr pin to the RSRs or from the XSRs to the mcbspi_dx pin. The time for each bit transfer is controlled by the rising or falling edge of a clock signal.

The receive clock signal (CLKR_int) controls bit transfers from the mcbspi_dr pin to the RSRs. The transmit clock signal (CLKX_int) controls bit transfers from the XSRs to the mcbspi_dx pin. CLKR_int or CLKX_int signals can be derived from a pin at the boundary of the McBSP module (mcbspi_clkr and mcbspi_clkx respectively) or derived from inside the McBSP module (see Figure 21-26). The clocks source is selected by programming the McBSPi.MCBSPLP_PCR_REG[9] CLKXM bit and the McBSPi.MCBSPLP_PCR_REG[8] CLKRM bit respectively.

When the McBSPi.MCBSPLP_PCR_REG[9] CLKXM bit (transmitter clock mode) is set to:

- '0', CLKX_int is driven by an external clock and mcbspi_clkx is an input pin.
- '1', CLKX_int is driven by the internal sample rate generator and mcbspi_clkx is an output pin.

For the McBSPi.MCBSPLP_PCR_REG[8] CLKRM bit (receiver clock mode), see Table 21-15.

Table 21-15. Receiver Clock Mode

Value	Digital loop back mode	mcbspi_clkr pin	Description
0x0	McBSPi.MCBSPLP_XCCR_REG[5] DLB bit =0	input	CLKR_int is driven by an external clock
	McBSPi.MCBSPLP_XCCR_REG[5] DLB bit =1	High-impedance	CLKR_int is driven by CLKX_int. The CLKX_int is derived based on the CLKXM value.
0x1	McBSPi.MCBSPLP_XCCR_REG[5] DLB bit =0	output	CLKR_int is driven by the internal sample rate generator
	McBSPi.MCBSPLP_XCCR_REG[5] DLB bit =1	output	CLKR_int is driven by CLKX_int. The CLKX_int is derived based on the CLKXM value.

The polarities of CLKR and CLKX signals are configured in McBSPi.MCBSPLP_PCR_REG register.

The McBSPi.MCBSPLP_PCR_REG[1] CLKXP defines the transmit clock polarity:

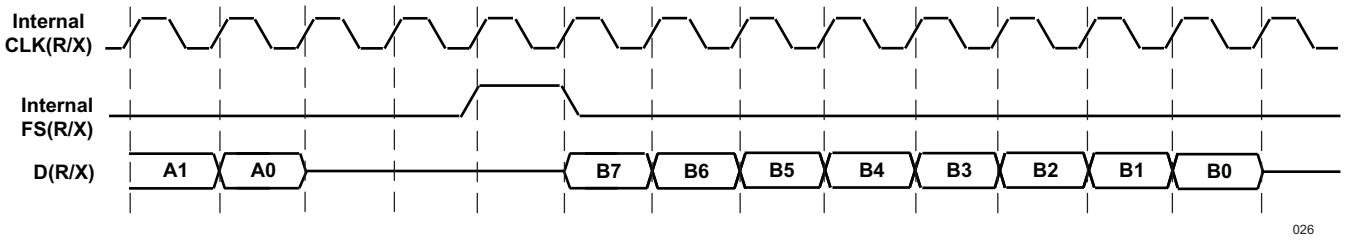
- When set to 0, transmit data driven on rising edge of CLKX signal
- When set to 1, transmit data driven on falling edge of CLKX signal

The McBSPi.MCBSPLP_PCR_REG[0] CLKRP defines the receive clock polarity:

- When set to 0, receive data sampled on falling edge of CLKX signal
- When set to 1, transmit data sampled on rising edge of CLKX signal

Figure 21-27 gives an example where the clock signal controls the timing of each bit transfer on the pin.

Figure 21-27. Clock Signal Control of Bit Transfer Timing



Note: The McBSP module is constrained to operate at an internal functional frequency of up to L4 interface frequency divided by 2. When driving CLKX or CLKR at the pin, choose an appropriate input clock frequency. When using the internal sample rate generator for CLKX/CLKR/CLKS, choose an appropriate input clock frequency (up to L4 interface frequency) and divide down value by programming the McBSPi.MCBSPLP_SRGR1_REG[7:0] CLKGDV bit field.

21.4.2.3.2 Serial Words

Bits traveling between a shift register (RSR or XSR) and a data pin (mcbspi_dr or mcbspi_dx) are transferred in a group called a serial word. The software defines how many bits are in a word by programming:

- For the receiver: the McBSPi.MCBSPLP_RCR1_REG[7:5] RWDLEN1 field and the McBSPi.MCBSPLP_RCR2_REG[7:5] RWDLEN2 field.
- For the transmitter: the McBSPi.MCBSPLP_XCR1_REG[7:5] XWDLEN1 field and the McBSPi.MCBSPLP_XCR2_REG[7:5] XWDLEN2 field.

The difference of use is explained to Section 21.4.2.4.1.

The various possibilities of word length are 8, 12, 16, 20, 24, and 32 bits (for field values, see Section 21.7)

Bits coming from the mcbspi_dr pin are held in the RSR until it holds a full serial word, then the word is passed to the RB and to the McBSPi.MCBSPLP_DRR_REG register.

During transmission, XSR accepts new data from XB after a full serial word has been passed from XSR to the mcbspi_dx pin.

In the example in Figure 21-27, an 8–bit word size was defined (see the transfer of the 8-bit word B).

21.4.2.3.3 Frames and Frame Synchronization

One or more words (up to 128) are transferred in a group called a frame. The software defines how many words are in a frame by programming:

- For the receiver: The McBSPi.MCBSPLP_RCR1_REG[14:8] RFRLEN1 field and the McBSPi.MCBSPLP_RCR2_REG[14:8] RFRLEN2 field.
- For the transmitter: The McBSPi.MCBSPLP_XCR1_REG[14:8] XFRLEN1 field and the McBSPi.MCBSPLP_XCR2_REG[14:8] XFRLEN2 field.

The difference between these registers is explained to Section 21.4.2.4.1. For the corresponding between field value and words number, see Section 21.7.

All the words in a frame are sent in a continuous stream. However, there can be pauses between frame transfers. The McBSP module uses frame-synchronization signals (FSG) to determine when each frame is received/transmitted. When a pulse occurs on a frame-synchronization signal, the McBSP module begins receiving/transmitting a frame of data. When the next pulse occurs, the McBSP module receives/transmits the next frame, and so on.

Pulses on the receive frame-synchronization (FSR_int) signal initiate frame transfers on mcbspi_dr. Pulses on the transmit frame-sync (FSX_int) signal initiate frame transfers on mcbspi_dx. FSR_int or FSX_int signals can be derived from a pin at the boundary of the McBSP module (mcbspi_fsr and mcbspi_fsx respectively) or derived from inside the McBSP module (see Figure 21-25). The frame-sync source is selected by programming the McBSPi.MCBSPLP_PCR_REG[11] FSXM bit and the McBSPi.MCBSPLP_PCR_REG[10] FSRM bit respectively.

When the McBSPi.MCBSPLP_PCR_REG[11] FSXM bit (transmitter frame-sync mode) is set to:

- ‘0’, FSX_int is derived from an external source and mcbspi_fsx is an input pin.
- ‘1’, FSX_int is determined by the McBSPi.MCBSPLP_SRGR2_REG[12] FSGM bit and mcbspi_fsx is an output pin.

For the McBSPi.MCBSPLP_PCR_REG[10] FSRM bit (receiver frame-sync mode), is set to:

- ‘0’, FSR_int is generated by an external source and mcbspi_fsr is an input pin
- ‘1’, FSR_int is generated internally by sample rate generator. The mcbspi_fsx is an output pin except when McBSPi.MCBSPLP_SRGR2_REG[15] GSYNC bit = 0x1

In the example in Figure 21-26, a one-word frame is transferred when a frame-synchronization pulse occurs. The polarities of FSR and FSX signals are programmable by bits on McBSPi.MCBSPLP_PCR_REG register.

The McBSPi.MCBSPLP_PCR_REG[3] FSXP defines the transmit frame-sync polarity:

- When set to ‘0’, frame-sync pulse FSX is active high
- When set to ‘1’, frame-sync pulse FSX is active low

The McBSPi.MCBSPLP_PCR_REG[2] FSRP defines the receive frame-sync polarity:

- When set to ‘0’, frame-sync pulse FSR is active high
- When set to ‘1’, frame-sync pulse FSR is active low

In McBSP operation, the inactive-to-active transition of the frame-synchronization signal indicates the start of the next frame. For this reason, the frame-synchronization signal may be high for an arbitrary number of clock cycles. Only after the signal is recognized to have gone inactive, and then active again, does the next frame synchronization occur.

21.4.2.3.4 Detecting Frame-Synchronization Pulses, Even in Reset State

The McBSP module can generate receive and transmit interrupts to the MPU/IVA2.2 subsystems to indicate specific events in the McBSP module. To facilitate detection of frame synchronization, these interrupts can be sent in response to frame-synchronization pulses (see Section 21.5 for further details).

Unlike other serial port interrupt modes, this mode can operate while the associated portion of the serial port is in reset (such as activating receive interrupt when the receiver is in reset). In this case, McBSPi.MCBSPLP_PCR_REG[0] FSRM bit/McBSPi.MCBSPLP_PCR_REG[1] FSXM bit and McBSPi.MCBSPLP_PCR_REG[2] FSRP bit/McBSPi.MCBSPLP_PCR_REG[3] FSXP bit still select the appropriate source and polarity of frame synchronization. Thus, even when the serial port is in the reset state, these signals are synchronized to the interface clock (McBSPi_ICLK) and then sent to the MPU/IVA2.2 subsystem in the form of receive interrupt and transmit interrupt at the point where they feed the receiver and transmitter of the serial port. Consequently, a new frame-synchronization pulse can be detected, and then, the MPU/IVA2.2 subsystem can take the serial port out of reset safely.

21.4.2.3.5 Ignoring Frame-Synchronization Pulses

The McBSP module ignores transmit and/or receive frame-synchronization pulses if the frame transfer was started by a previous frame-synchronization pulse (unexpected frame-synchronization pulses). The McBSP module does not support features like retransmit or re-receive of an erroneous frame or word. The receiver or transmitter ignores frame-synchronization pulses until the desired frame length or number of words is reached. For more details on unexpected frame-synchronization pulses, see Section 21.4.4.3, or Section 21.4.4.6

21.4.2.3.6 Frame Frequency

The frame frequency is determined by the period between frame-synchronization pulses and is defined as shown in the following equation:

Frame frequency = Clock frequency/(Number of clock cycles between two rising edges [or falling edges] of two consecutive frame synchronization pulses)

The frame frequency can be increased by decreasing the time between frame-synchronization pulses (limited only by the number of bits per frame). As the frame transmit frequency increases, the inactivity period between the data packets for adjacent transfers decreases to zero.

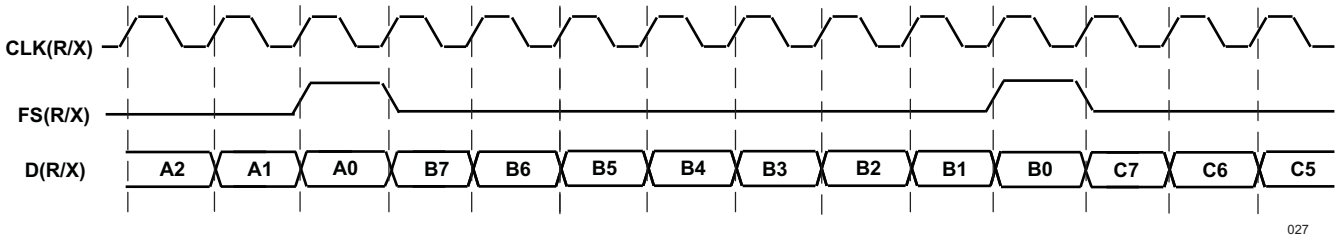
21.4.2.3.7 Maximum Frame Frequency

The minimum number of clock cycles between frame synchronization pulses is equal to the number of bits transferred per frame. The maximum frame frequency is defined as shown in the following equation:

Maximum frame frequency = clock frequency/Number of bits per frame

Figure 21-28 below shows the McBSP operating at maximum packet frequency. At maximum packet frequency, the data bits in consecutive packets are transmitted contiguously with no inactivity between bits.

Figure 21-28. McBSP Operating at Maximum Packet Frequency



If there is a 1–bit data delay as shown in Figure 21-28, the frame-synchronization pulse overlaps the last bit transmitted in the previous frame. Effectively, this permits a continuous stream of data, back-to-back transfers.

Note: For McBSPi.MCBSPLP_XCR2_REG[1:0] XDATDLY = 0x0 (0-bit data delay), the first bit of data is transmitted asynchronously to the internal transmit clock signal (CLKX_int). For more details, see Section 21.5.1.6.2.2.5.

21.4.2.4 Frame Phases (Dual-Phase Frame I2S Support)

The McBSP module allows you to configure each frame to contain one or two phases. The support for dual-phase frames is required to provide I2S fully compliant capabilities (audio left and right channels—stereo audio stream).

CAUTION

The limitation on dual-phase frame support is that the number of words per phase must be set to 1 for both first and second phase. It is the only possible value for word per frame when using the dual phase frame.

The number of bits per word can be specified differently for each of the two phases of a frame, allowing greater flexibility in structuring data transfers. For example, a user might define a frame as consisting of one phase containing one word of 16 bits, followed by a second phase consisting of one word of 32 bits. This configuration allows the user to compose frames for custom applications such as I2S protocol.

21.4.2.4.1 Number of Phases, Words, and Bits per Frame

Table 21-16 below shows which bit fields in the receive control registers (McBSPi.MCBSPLP_RCR1_REG and McBSPi.MCBSPLP_RCR2_REG) and in the transmit control registers (McBSPi.MCBSPLP_XCR1_REG and McBSPi.MCBSPLP_XCR2_REG) determine the number of phases per frame, the number of words per frame, and the number of bits per word for each phase, for both receiver and transmitter. The maximum number of words per frame is limited to 2 when using dual-phase frames (one word for each phase), and to 128 for a single-phase frame. The number of bits per word can be 8, 12, 16, 20, 24, or 32 bits.

The following legend applies to the table:

- RPHASE => McBSPi.MCBSPLP_RCR2_REG[15] RPHASE bit
- XPHASE => McBSPi.MCBSPLP_XCR2_REG[15] XPHASE bit
- RFRLEN1 => McBSPi.MCBSPLP_RCR1_REG[14:8] RFRLEN1 field
- RFRLEN2 => McBSPi.MCBSPLP_RCR2_REG[14:8] RFRLEN2 field
- XFRLEN1 => McBSPi.MCBSPLP_XCR1_REG[14:8] XFRLEN1 field
- XFRLEN2 => McBSPi.MCBSPLP_XCR2_REG[14:8] XFRLEN2 field
- RWDLEN1 => McBSPi.MCBSPLP_RCR1_REG[7:5] RWDLEN1 field
- RWDLEN2 => McBSPi.MCBSPLP_RCR2_REG[7:5] RWDLEN2 field
- XWDLEN1 => McBSPi.MCBSPLP_XCR1_REG[7:5] XWDLEN1 field
- XWDLEN2 => McBSPi.MCBSPLP_XCR2_REG[7:5] XWDLEN2 field

Table 21-16. Phases, Words and Bits per Frame Control Bit

Operation	Number of phases	Words per frame set with	Bits per word set with
Reception	1 (RPHASE=0)	RFRLEN1	RWDLEN1
Reception	2 (RPHASE=1)	RFRLEN1=0x0 and RFRLEN2=0x0	RWDLEN1 for phase 1 RWDLEN2 for phase 2
Transmission	1 (XPHASE=0)	XFRLEN1	XWDLEN1
Transmission	2 (XPHASE=1)	XFRLEN1=0x0 and XFRLEN2=0x0	XWDLEN1 for phase 1 XWDLEN2 for phase 2

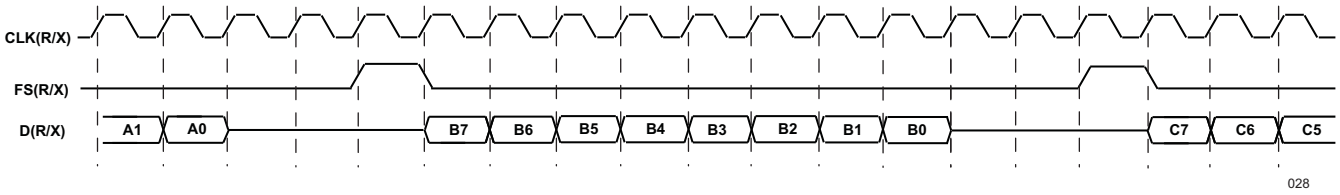
21.4.2.4.2 Single-Phase Frame Example

Figure 21-29 below shows an example of a single-phase data frame containing one 8-bit word. Because the transfer is configured for one data bit delay, the data on the mcbspi_dx and mcbspi_dr pins are available one clock cycle after FS(R/X) goes active. The following table shows the assumptions used in the example of this figure.

Table 21-17. Assumptions for the Single-Phase Frame Example

Assumption	Value	Bit or Field Name
Single-phase frame	'0'	McBSPi.MCBSPLP_RCR2_REG[15] RPHASE bit
		McBSPi.MCBSPLP_XCR2_REG[15] XPHASE bit
One word per frame	0x0	McBSPi.MCBSPLP_RCR1_REG[14:8] RFRLEN1 field
		McBSPi.MCBSPLP_XCR1_REG[14:8] XFRLEN1 field
8-bit word length	0x0	McBSPi.MCBSPLP_RCR1_REG[7:5] RWDLEN1 field
		McBSPi.MCBSPLP_XCR1_REG[7:5] XWDLEN1 field
word length in register2	ignored	McBSPi.MCBSPLP_RCR2_REG[14:8] RFRLEN2 field
		McBSPi.MCBSPLP_XCR2_REG[14:8] XWDLEN2 field
Receive data clocked on falling edge	'0'	McBSPi.MCBSPLP_PCR_REG[0] CLKRP bit
Transmit data clocked on rising edge		McBSPi.MCBSPLP_PCR_REG[1] CLKXP bit
Active–high frame–sync signals	'0'	McBSPi.MCBSPLP_PCR_REG[2] FSRP bit
		McBSPi.MCBSPLP_PCR_REG[3] FSXP bit
1-bit data delay	01b	McBSPi.MCBSPLP_RCR2_REG[1:0] RDATDLY field
		McBSPi.MCBSPLP_XCR2_REG[1:0] XDARDLY field

Figure 21-29. Single-Phase Frame for a McBSP Data Transfer



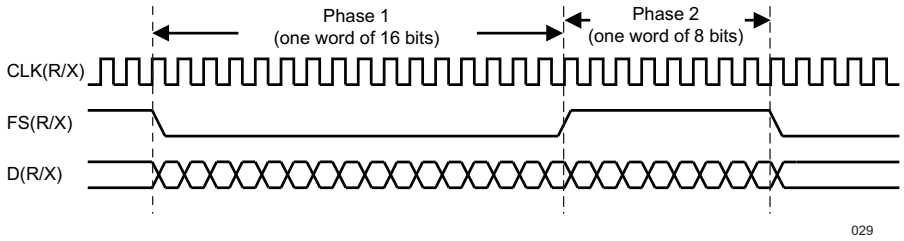
21.4.2.4.3 Dual-Phase Frame Example

Figure 21-30 below shows an example of a frame. The first phase consists of one word of 16 bits, followed by a second phase of one word of 8 bits. The entire bit stream in the frame is contiguous. There are no gaps between words/phases. Table 21-18 shows the assumptions used to the example in Figure 21-30.

Table 21-18. Assumptions for the Dual-Phase Frame Example

Assumption	Value	Bit or Field name
Single-phase frame	'1'	McBSPi.MCBSPLP_RCR2_REG[15] RPHASE bit
		McBSPi.MCBSPLP_XCR2_REG[15] XPHASE bit
One word per frame	0x0	McBSPi.MCBSPLP_RCR1_REG[14:8] RFRLEN1 field
		McBSPi.MCBSPLP_XCR1_REG[14:8] XFRLEN1 field
16-bit word length	0x0	McBSPi.MCBSPLP_RCR1_REG[7:5] RWDLEN1 field
		McBSPi.MCBSPLP_XCR1_REG[7:5] XWDLEN1 field
8-bit word length	0x2	McBSPi.MCBSPLP_RCR2_REG[14:8] RFRLEN2 field
		McBSPi.MCBSPLP_XCR2_REG[14:8] XWDLEN2 field
Receive data clocked on falling edge	'0'	McBSPi.MCBSPLP_PCR_REG[0] CLKRP bit
Transmit data clocked on rising edge		McBSPi.MCBSPLP_PCR_REG[1] CLKXP bit
Active-high frame-sync signals	'0'	McBSPi.MCBSPLP_PCR_REG[2] FSRP bit
		McBSPi.MCBSPLP_PCR_REG[3] FSXP bit
0-bit data delay	00b	McBSPi.MCBSPLP_RCR2_REG[1:0] RDATDLY field
		McBSPi.MCBSPLP_XCR2_REG[1:0] XDARDLY field

Figure 21-30. Dual-Phase Frame for a McBSP Data Transfer



21.4.2.5 McBSP Reception

This section explains the fundamental process of reception in the McBSP module. For details about how to program the McBSP receiver, see Section 21.5, Section 21.5.1.4, and Section 21.5.1.5.

Figure 21-31 and Figure 21-32 below show how reception occurs in the McBSP module. A description of the process follows the figures. Figure 21-31 shows the physical path for the data.

Figure 21-31. McBSP Reception Physical Data Path

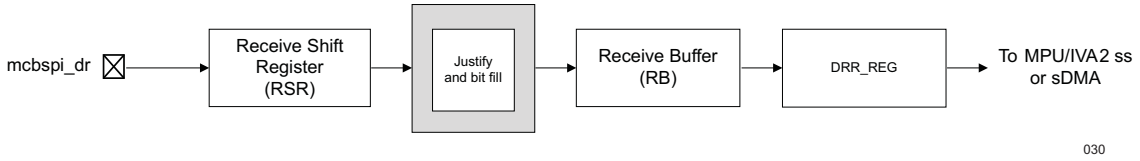
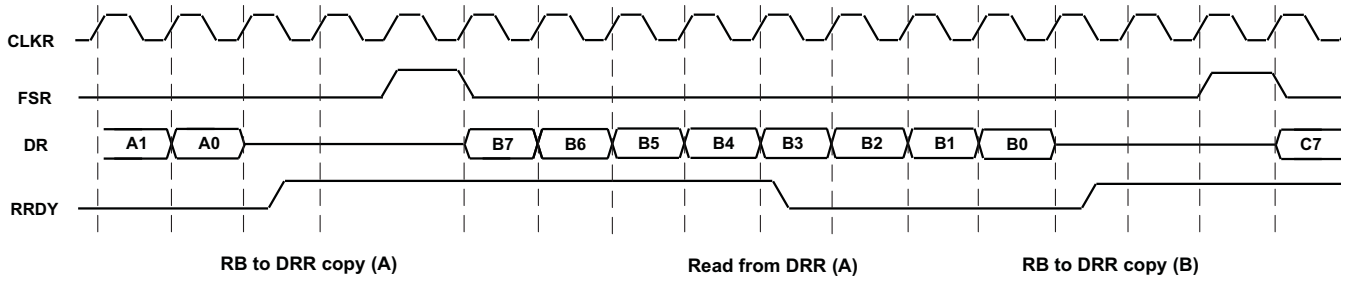


Figure 21-32 is a timing diagram showing signal activity for one possible reception scenario.

Figure 21-32. McBSP Reception Signal Activity



RRDY: Status of receiver ready bit (high is 1)

The following process describes how data travels from the mcbspi_dr pin to the MPU/IVA2.2 subsystem or to the sDMA controller:

1. The McBSP module waits for a receive frame-synchronization pulse on FSR_int.
2. When the pulse arrives, the McBSP module inserts the appropriate data delay that is selected with the McBSPi.MCBSPLP_RCR2_REG[1:0] RDATDLY bits. In the preceding timing diagram a 1-bit data delay is selected.
3. The McBSP module accepts data bits on the mcbspi_dr pin and shifts them into the RSR. For details on choosing a word length, see Section 21.5.1.5.2.2.2.
4. When a full word is received, the McBSP module copies the contents of the RSR to the RB, provided that RB is not full.
5. When the programmed receive threshold is reached (McBSPi.MCBSPLP_THRSH1_REG[10:0] RTHRESHOLD field), the McBSP module asserts the receiver ready bit (McBSPi.MCBSPLP_SPCR1_REG[1] RRDY bit). This indicates that receive data is ready to be read by the MPU/IVA2.2 subsystem or the sDMA controller by accessing McBSPi.MCBSPLP_DRR_REG register.

   The data copied from RB to McBSPi.MCBSPLP_DRR_REG is justified and bit filled according to the

McBSPi.MCBSPLP_SPCR1_REG[14:13] RJUST field.

6. The MPU/IVA2.2 subsystem or the sDMA controller reads the data from the data receive register. When the RB is empty, McBSPi.MCBSPLP_SPCR1_REG[1] RRDY bit is cleared.

21.4.2.6 McBSP Transmission

This section explains the fundamental process of transmission in the McBSP module. For details about how to program the McBSP transmitter, see Section 21.5, and Section 21.5.1.6.

Figures below show how transmission occurs in the McBSP module. A description of the process follows the figures. Figure 21-33 shows the physical path for the data.

Figure 21-33. McBSP Transmission Physical Data Path

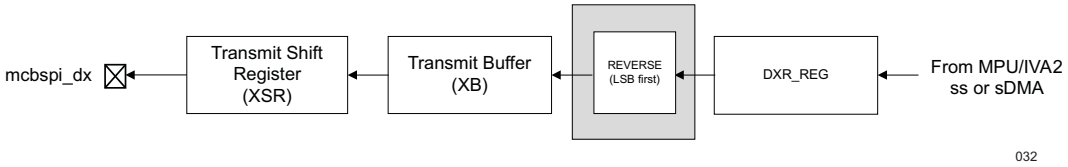
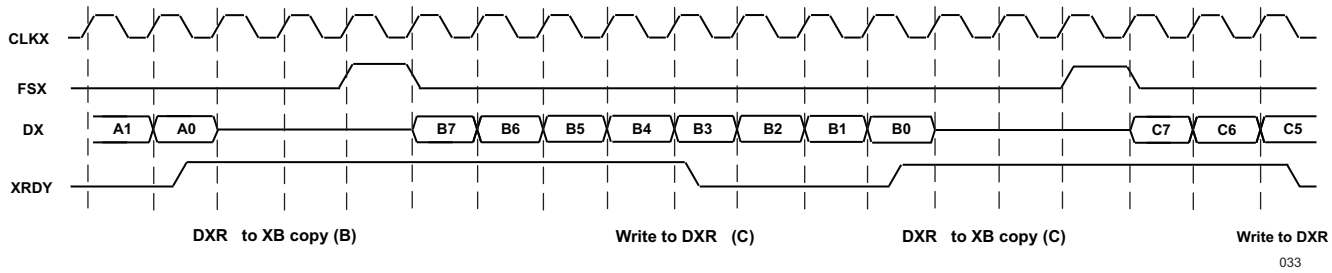


Figure 21-34 is a timing diagram showing signal activity for one possible transmission scenario.

Figure 21-34. McBSP Transmission Signal Activity



XRDY: Status of transmitter ready bit (high is 1)

1. The MPU/IVA2.2 subsystem or the sDMA controller writes data to the data transmit register (McBSPi.MCBSPLP_DXR_REG). When the XB is reached the transmitter ready bit (McBSPi.MCBSPLP_SPCR2_REG[1] XRDY bit) is cleared to indicate that the transmitter is not ready for new data. For details on choosing a word length, see Section 21.5.1.6.2.2.2.
2. When new data arrives in McBSPi.MCBSPLP_DXR_REG register, the McBSP module copies the content of the data transmit register to the XB. In addition, the transmit ready bit (McBSPi.MCBSPLP_SPCR2_REG[1] XRDY bit) is set as long as the buffer contains at least the transmit threshold number of free locations (McBSPi.MCBSPLP_THRSH2_REG[10:0] XTHRESHOLD field). This indicates that the transmitter is ready to accept new data from the MPU/IVA2.2 subsystem or the sDMA controller.
3. The McBSP module waits for a transmit frame-synchronization pulse on FSX_int.
4. When the pulse arrives, the McBSP module inserts the appropriate data delay that is selected with the McBSPi.MCBSPLP_XCR2_REG[1:0] XDATDLY field.

   In the preceding timing diagram, a 1-bit data delay is selected.
5. The McBSP module shifts data bits from the XSR to the mcbspi_dx pin.

21.4.2.7 Enable/Disable the Transmit and Receive Processes

The McBSP module has the option to stop-resume the transmit/receive process while the module is in functional mode (out of transmit/receive reset).

When the transmit/receive disable bit (McBSPi.MCBSPLP_XCCR_REG[0] XDISABLE/McBSPi.MCBSPLP_RCCR_REG[0] RDISABLE) is set, the McBSP module stops the transmit/receive operation at the next frame boundary (frame corruption avoided).

During the receive disable state, the frames that are sent (when FSR signal is asserted while receive disable) by the remote device are lost, and receive buffer overflow status bit (McBSPi.MCBSPLP_IRQSTATUS_REG[5] ROVFLSTAT) is not set. Also, the frames received by the remote device while McBSPi.MCBSPLP_XCCR_REG[0] XDISABLE bit is set (when FSX signal is asserted while transmit disable) are meaningless undefined data frames, and transmit buffer underflow status bit (McBSPi.MCBSPLP_IRQSTATUS_REG[11] XUNDFLSTAT) is not set. The presence of the frame synchronization, while transmit/receive process is disabled, can be checked by reading the transmit/receive Frame-sync interrupt status: McBSPi.MCBSPLP_IRQSTATUS_REG[8] XFSX / McBSPi.MCBSPLP_IRQSTATUS_REG[1] RFSR bits.

As soon as the McBSPi.MCBSPLP_XCCR_REG[0] XDISABLE/McBSPi.MCBSPLP_RCCR_REG[0] RDISABLE bit is cleared, the transmit/receive process resumes at the next frame boundary.

Note: It is not recommended to use this mechanism together with the possibility to interrogate the transmit/receive buffer status register (McBSPi.MCBSPLP_XBUFFSTAT_REG[7:0] XBUFFSTAT/McBSPi.MCBSPLP_RBUFFSTAT_REG[7:0] RBUFFSTAT field indicating the occupied/available buffer locations), since this register is an interface clock (McBSPi_ICLK) synchronous register and does not reflect the exact number of occupied/free locations available on the functional clock domain.

21.4.2.8 MCBSP Data Transfer Mode

Note: For all examples in this section, the configured CLKX edge is the rising edge (McBSPi.MCBSPLP_PCR_REG[1] CLKXP bit=0x0) and the configured CLKR edge is the falling edge (McBSPi.MCBSPLP_PCR_REG[0] CLKRP bit=0x0). There are the reset values.

In timing diagrams below, a 1-bit data delay is selected (McBSPi.MCBSPLP_RCR2_REG[1:0] RDATDLY field=0x01 and McBSPi.MCBSPLP_XCR2_REG[1:0] XDATDLY field=0x01), because data often follows a 1-cycle active frame-synchronisation.

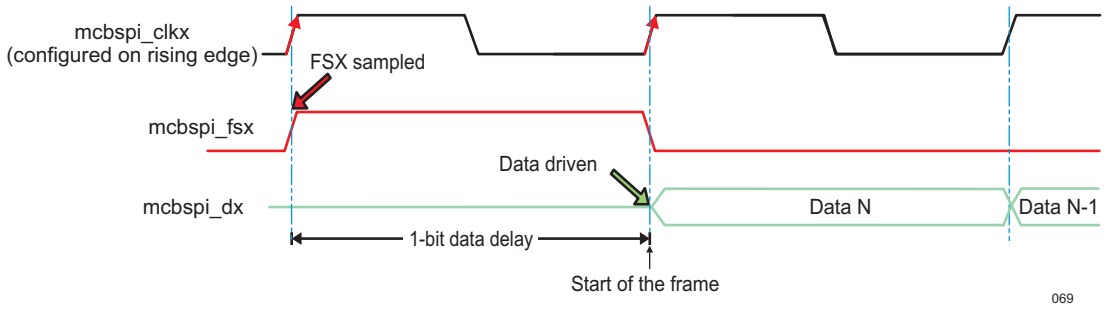
McBSP modules can support 2 edge selection modes for transmit and receive data transfer at the system level:

- The full cycle mode, for which one clock period is used to transfer the data, generated on one edge and captured on the same edge (one clock period later).
- The half cycle mode, for which one half clock period is used to transfer the data, generated on one edge and captured on the opposite edge (one half clock period later). Note that a new data is generated only every clock period, which permits to guarantee the required hold time.

21.4.2.8.1 Transmit Full Cycle Mode

When configured in full cycle mode (McBSPi.MCBSPLP_XCCR_REG[11] XFULL_CYCLE bit = 0x1), the FSX signal is sampled on the configured CLKX edge and the data is driven on the same configured edge. See Figure 21-35.

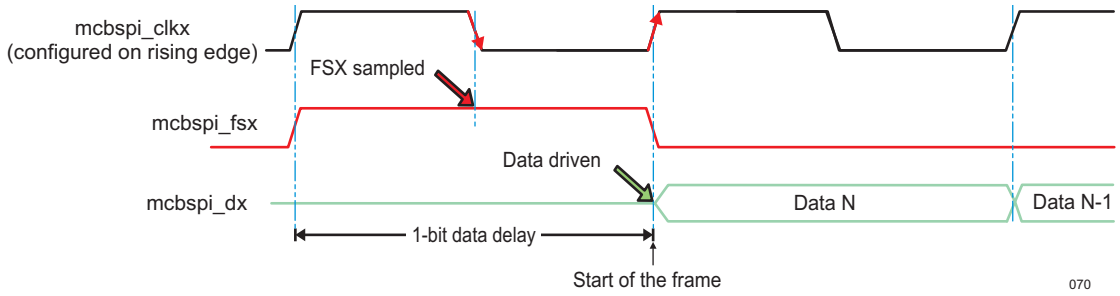
Figure 21-35. Transmit Full Cycle Timing Diagram



21.4.2.8.2 Transmit Half Cycle Mode

When configured in half cycle mode (McBSPi.MCBSPLP_XCCR_REG[11] XFULL_CYCLE bit = 0x0, reset value), the FSX signal is sampled on the opposite configured CLKX edge and the data is driven on the next configured edge. See Figure 21-36.

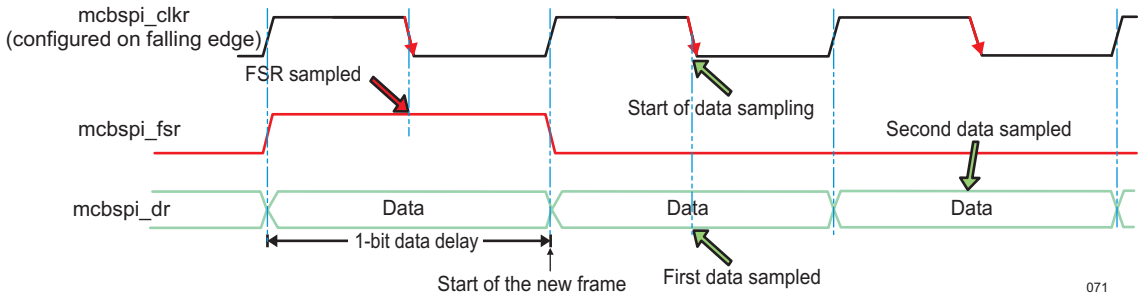
Figure 21-36. Transmit Half Cycle Timing Diagram



21.4.2.8.3 Receive Full Cycle Mode

When configured in full cycle mode (McBSPi.MCBSPLP_RCCR_REG[11] RFULL_CYCLE bit = 0x1, reset value), the FSR signal is sampled on the configured CLKR edge and the data is driven on the same configured edge. See Figure 21-37.

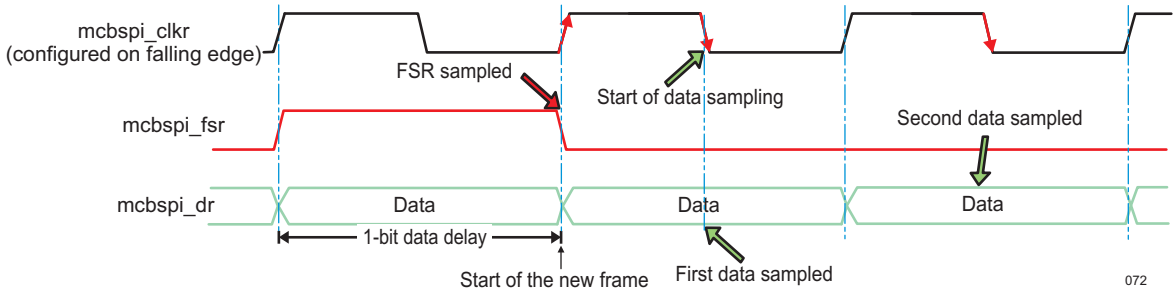
Figure 21-37. Receive Full Cycle Timing Diagram



21.4.2.8.4 Receive Half Cycle Mode

When configured in half cycle mode (McBSPi.MCBSPLP_RCCR_REG[11] RFULL_CYCLE bit = 0x0), the FSR signal is sampled on the opposite configured CLKR edge and the data is driven on the next configured edge. See Figure 21-38.

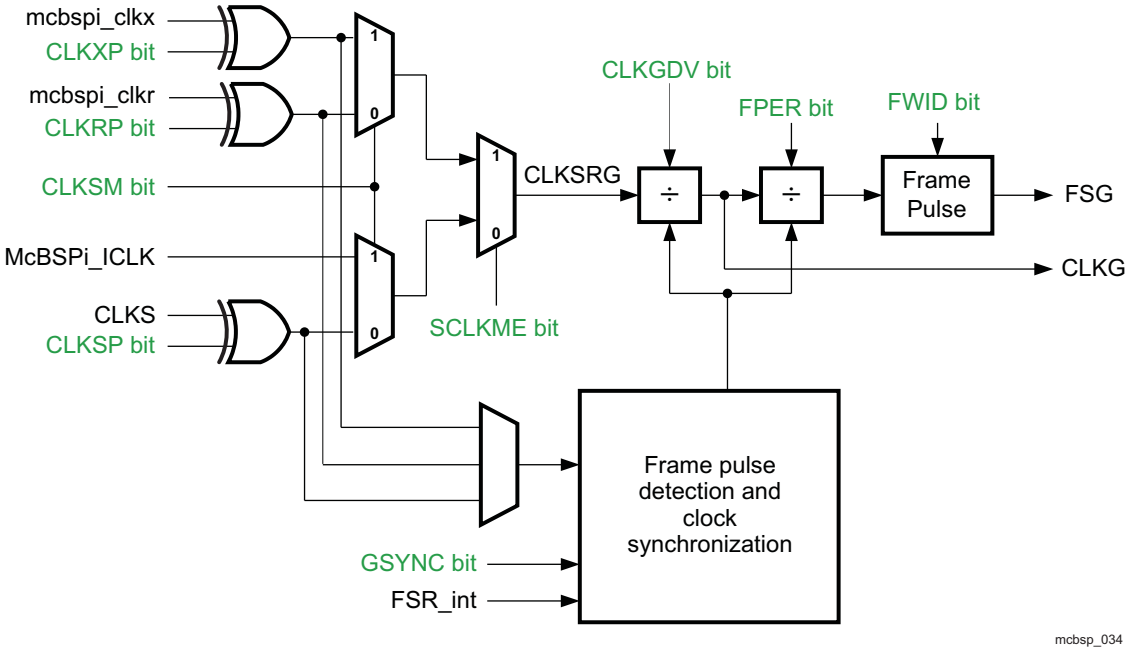
Figure 21-38. Receive Half Cycle Timing Diagram



21.4.3 McBSP SRG

The McBSP module contains an internal SRG that can be used to generate an internal data clock (CLKG) and an internal frame-synchronization signal (FSG). CLKG can be used for bit shifting on the data receive pin (mcbspi_dr) and/or the data transmit pin (mcbspi_dx). FSG can be used to initiate frame transfers on mcbspi_dr pin and/or mcbspi_dx pin. Figure 21-39 is a conceptual block diagram of the SRG.

Figure 21-39. Conceptual Block Diagram of the Sample Rate Generator



The source clock for the SRG (labeled CLKSRG in the diagram) can be supplied by either the interface clock (McBSPi_ICLK), or the functional clock (CLKS input), or by an external pin (mcbspi_clkx, or mcbspi_clkr). The source is selected with the McBSPi.MCBSPLP_PCR_REG[7] SCLKME bit and the McBSPi.MCBSPLP_SRGR2_REG[13] CLKSM bit.

If a pin or CLKS signal is used, the polarity of the incoming signal can be inverted with the appropriate polarity bit (McBSPi.MCBSPLP_SRGR2_REG[14] CLKSP bit, McBSPi.MCBSPLP_PCR_REG[1] CLKXP bit, or McBSPi.MCBSPLP_PCR_REG[0] CLKRP bit).

The SRG has a three-stage clock divider that gives CLKG and FSG programmability.

The three stages provide:

- Clock divide-down: The source clock (CLKSRG) is divided according to the McBSPi.MCBSPLP_SRGR1_REG[7:0] CLKGDV field to produce CLKG signal
- Frame period divide-down: CLKG is divided according to the McBSPi.MCBSPLP_SRGR2_REG[11:0] FPER field to control the period from the start of a frame-pulse to the start of the next pulse
- Frame-synchronization pulse-width countdown: CLKG cycles are counted according to the McBSPi.MCBSPLP_SRGR1_REG[15:8] FWID field to control the width of each frame-synchronization pulse

Note: The McBSP module cannot operate at an internal functional frequency faster than L4 interface frequency divided by 2. Choose an input clock frequency and a McBSPi.MCBSPLP_SRGR1_REG[7:0] CLKGDV value such that CLKG is less than or equal to L4 interface frequency divided by 2.

In addition to the three-stage clock divider, the sample rate generator has a frame-synchronization pulse detection and clock synchronization module that allows synchronization of the clock divide down with an incoming frame-synchronization pulse on the mcbspi_fsr pin. This feature is enabled or disabled with the McBSPi.MCBSPLP_SRGR2_REG[15] GSYNC bit.

CLKG is used as source to generate the output clocks CLKX/CLKR when the McBSPi.MCBSPLP_PCR_REG[9] CLKXM / McBSPi.MCBSPLP_PCR_REG[8] CLKRM bit indicates that the clock is an output. The output CLKX/CLKR is generated according to the clock polarity setting (see Figure 21-26).

For details on getting the sample rate generator ready for operation, see Section 21.5.

21.4.3.1 Clock Generation in the SRG

The SRG can produce a clock signal (CLKG) for use by the receiver, the transmitter, or both. Use of the SRG to drive clocking is controlled by the clock mode bits (McBSPi.MCBSPLP_PCR_REG[9] CLKXM and McBSPi.MCBSPLP_PCR_REG[8] CLKRM) and polarity mode bits (McBSPi.MCBSPLP_PCR_REG[1] CLKXP and McBSPi.MCBSPLP_PCR_REG[0] CLKRP).

When a clock mode bit is set to 1 (CLKRM=1 for reception, CLKXM=1 for transmission), the corresponding data clock (CLKR for reception, CLKX for transmission) is driven by the internal SRG output clock (CLKG) according to the polarity setting.

The effects of this setting on the McBSP module are partially affected by the use of the digital loopback (DLB) mode, the analog loop back (ALB) mode and by the synchronous receive/transmit setting, respectively, as described in Table 21-19. The ALB mode is selected with the McBSPi.MCBSPLP_SPCR1_REG[15] ALB bit. The DLB mode is selected with the McBSPi.MCBSPLP_XCCR_REG[5] DLB bit. The synchronous setting is controlled by input signals. These signals are defined by the control registers of the System Control Module (more details, refer Section 21.3.1)

When using the SRG as a clock source, make sure the SRG is enabled (McBSPi.MCBSPLP_SPCR2_REG[6] GRST bit =1).

Table 21-19. Effects of DLB and ALB Bits on Clock Modes

Mode Bit Settings		Effect
CLKRM=1	DLB=0 and ALB = 0 (Digital and analog loop back mode disabled)	mcbsp1_clkr is an output pin driven by the SRG output clock (CLKG).
	DLB=0 and ALB = 1 (Digital loop back mode disabled and Analog loop back mode enabled)	mcbsp1_clkr is an output pin driven by the SRG output clock (CLKG). The receiver functional part internal clock is driven by CLKX input signal provided by mcbspi_clkx pin. The source of CLKX depends on the CLKXM bit. The receive frame synchronization is driven by FSX input signal provided by mcbspi_fsx pin. The receive data is driven by the DX input loop-back pin (mcbspi_dx).
	DLB=1 & ALB = 0 (Digital loop back mode enabled and Analog loop back mode disabled)	The SRG and the frame synchronization generator must be enabled. The internal transmit and receive clocks are driven by the SRG (CLKG having the appropriate CLKXP polarity). The transmit and receive frame synchronization signals are driven by FSG (having the appropriate FSXP polarity). The transmit data is connected to the DR input data. Note that in digital loop back mode no serial link activity will be seen by the remote device.
	DLB=1 & ALB = 1 (reserved mode)	Undefined functionality.

Table 21-19. Effects of DLB and ALB Bits on Clock Modes (continued)

Mode Bit Settings		Effect
CLKXM= 1	DLB=0 & ALB = 0 (Digital & analog loop back mode disabled)	mcbspi_clkx is an output pin driven by the SRG output clock (CLKG).
	DLB=0 & ALB = 1 (Digital loop back mode disabled and Analog loop back mode enabled)	mcbspi_clkx is an output pin driven by the SRG output clock (CLKG).
	DLB=1 & ALB = 0 (Digital loop back mode enabled and Analog loop back mode disabled)	The SRG and the frame synchronization generator need to be enabled. The internal transmit and receive clocks are driven by the SRG (CLKG having the appropriate CLKXP polarity). The transmit and receive frame synchronization signals are driven by FSG (having the appropriate FSXP polarity). The transmit data is connected to the DR input data. Note that in digital loop back mode no serial link activity will be seen by the remote device.
	DLB=1 & ALB = 1 (reserved mode)	Undefined functionality.
	SCM.CONTROL_DEVCONF0[3] MCBSP1_CLKR bit =1 (synchronous setting and DLB = 0 & ALB = 0)	CLKX is an output pin driven by the SRG output clock (CLKG). CLKR is connected to the CLKX.

21.4.3.2 Frame Sync Generation in the SRG

The SRG can produce a frame-synchronization signal (FSG) for use by the receiver, the transmitter, or both.

If you want the receiver to use FSG for frame synchronization, make sure McBSPi.MCBSPLP_PCR_REG[10] FSRM bit =1. (When FSRM=0, receive frame synchronization is supplied via the mcbspi_fsr pin.)

If you want the transmitter to use FSG for frame synchronization, you must set:

- McBSPi.MCBSPLP_PCR_REG[11] FSXM = 1: This indicates that transmit frame synchronization is supplied by the McBSP module itself rather than from the mcbspi_fsx pin.
- MCBSPLP_SRGR2_REG[12] FSGM=1: This indicates that when FSXM=1, transmit frame synchronization is supplied by the SRG.

Note: When FSGM=0 and FSXM=1, the transmit frame-sync signal (FSX) is generated when XB is not empty. When FSGM = 0, McBSPi.MCBSPLP_SRGR2_REG[11:0] FPER and McBSPi.MCBSPLP_SRGR1_REG[15:8] FWID field are used to determine the frame synchronization period and width (external FSX is gated by the buffer empty condition).

In either case, the SRG must be enabled (McBSPi.MCBSPLP_SPCR2_REG[6] GRST bit=1) and the frame-synchronization logic in the SRG must be enabled (McBSPi.MCBSPLP_SPCR2_REG[7] FRST bit=0).

21.4.3.2.1 Choosing the Width of the Frame-sync Pulse

Each pulse on FSG has a programmable width. You program the McBSPi.MCBSPLP_SRGR1_REG[15:8] FWID field, and the resulting pulse width is (FWID+1)CLKG cycles, where CLKG is the output clock of the SRG. The range is from 1 to 256 clock periods.

21.4.3.2.2 Controlling the Period Between the Starting Edges of Frame Sync Pulses

You can control the amount of time from the starting edge of one FSG pulse to the starting edge of the next FSG pulse. This period is controlled in one of two ways, depending on the configuration of the SRG:

- If the SRG is using an external input clock and McBSPi.MCBSPLP_SRGR2_REG[15] GSYNC bit =1, FSG pulses in response to an inactive-to-active transition on the mcbspi_fsr pin. Thus, an external device controls the frame-synchronization period.

- Otherwise, the software program the McBSPi.MCBSPLP_SRGR2_REG[11:0] FPER field, and the resulting frame-synchronization period is (FPER+1)CLKG cycles, where CLKG is the output clock of the SRG. The range is from 1 to 4096 clock periods.

21.4.3.2.3 Keeping FSG Synchronized to an External Clock

When an external signal is selected to drive the SRG, the McBSPi.MCBSPLP_SRGR2_REG[15] GSYNC bit and the mcbspi_fsr pin can be used to configure the timing of FSG pulses.

McBSPi.MCBSPLP_SRGR2_REG[15] GSYNC=1 ensures that the McBSP module and an external device are dividing down the input clock with the same phase relationship.

If McBSPi.MCBSPLP_SRGR2_REG[15] GSYNC=1, an inactive-to-active transition on the mcbspi_fsr pin triggers a resynchronization of CLKG and generation of FSG.

21.4.3.3 Synchronizing SRG Outputs to an External Clock

The SRG can produce a clock signal (CLKG) and a FSG based on an input clock signal that is either the interface clock signal (McBSPi_ICLK), or the CLKS signal (PRCM functional clock or mcbsp_clks), or a signal at the mcbspi_clkr, or mcbspi_clkx pin. When an external clock (mcbsp_clks, or mcbspi_clkr, or mcbspi_clkx) is selected to drive the SRG, the McBSPi.MCBSPLP_SRGR2_REG[15] GSYNC bit and the mcbspi_fsr pin can be used to control the timing of CLKG and the pulsing of FSG relative to the chosen input clock. Make GSYNC=1 so that the McBSP module and an external device divide down the input clock with the same phase relationship.

If the McBSPi.MCBSPLP_SRGR2_REG[15] GSYNC bit=1:

- An inactive-to-active transition on the mcbspi_fsr pin triggers a resynchronization of CLKG signal and a pulsing of FSG signal.
- CLKG signal always begins with a high state after synchronization.
- FSR signal is always detected at the same edge of the input clock signal that generates CLKG signal, no matter how long the FSR pulse is.
- The McBSPi.MCBSPLP_SRGR2_REG[11:0] FPER field are ignored because the frame-synchronization period on FSG is determined by the arrival of the next frame-synchronization pulse on the mcbspi_fsr pin.

If the McBSPi.MCBSPLP_SRGR2_REG[15] GSYNC bit=0, CLKG signal runs freely and is not resynchronized, and the frame-synchronization period on FSG signal is determined by McBSPi.MCBSPLP_SRGR2_REG[11:0] FPER field.

21.4.3.3.1 Operating the Transmitter Synchronously with the Receiver

When the McBSPi.MCBSPLP_SRGR2_REG[15] GSYNC bit = 1, the transmitter can operate synchronously with the receiver, provided that the FSX signal is programmed to be driven by FSG signal (McBSPi.MCBSPLP_SRGR2_REG[12] FSGM = 1 and McBSPi.MCBSPLP_PCR_REG[11] FSXM = 1). If the FSR input signal has appropriate timing so that it can be sampled by the falling edge of CLKG signal, it can be used, instead, by setting McBSPi.MCBSPLP_PCR_REG[11] FSXM=0 and connecting FSR signal to FSX externally.

The SRG clock drives the transmit and receive clocking (McBSPi.MCBSPLP_PCR_REG[8] CLKRM bit and McBSPi.MCBSPLP_PCR_REG[9] CLKXM bit are set to 1). Therefore, the CLK(R/X) pin must not be driven by any other driving source.

21.4.3.3.2 Synchronization Examples

Figure 21-40 and Figure 21-41 show the clock and frame-synchronization operation with various polarities of CLKS (the chosen input clock) and FSR signals. These figures assume McBSPi.MCBSPLP_SRGR1_REG[15:8] FWID = 0x0, for an FSG pulse that is one CLKG cycle wide. The McBSPi.MCBSPLP_SRGR2_REG[11:0] FPER field are not programmed; the period from the start of a frame-synchronization pulse to the start of the next pulse is determined by the arrival of the next inactive-to-active transition on the mcbspi_fsr pin.

Each figure shows what happens to CLKG signal when the McBSPi.MCBSPLP_SRGR2_REG[15] GSYNC bit = 1 and if it is initially synchronized or if it is not initially synchronized. Figure 21-41 has a slower CLKG frequency (it has a larger divide-down value in the McBSPi.MCBSPLP_SRGR1_REG[7:0] CLKGDV field).

Figure 21-40. CLKG Synchronization and FSG Generation (GSYNC = 1 and CLKGDV = 0x1)

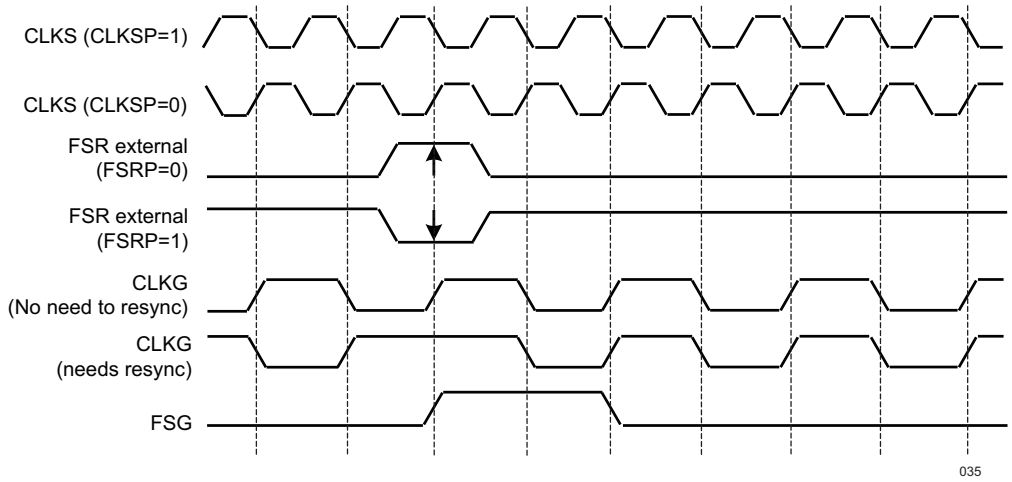
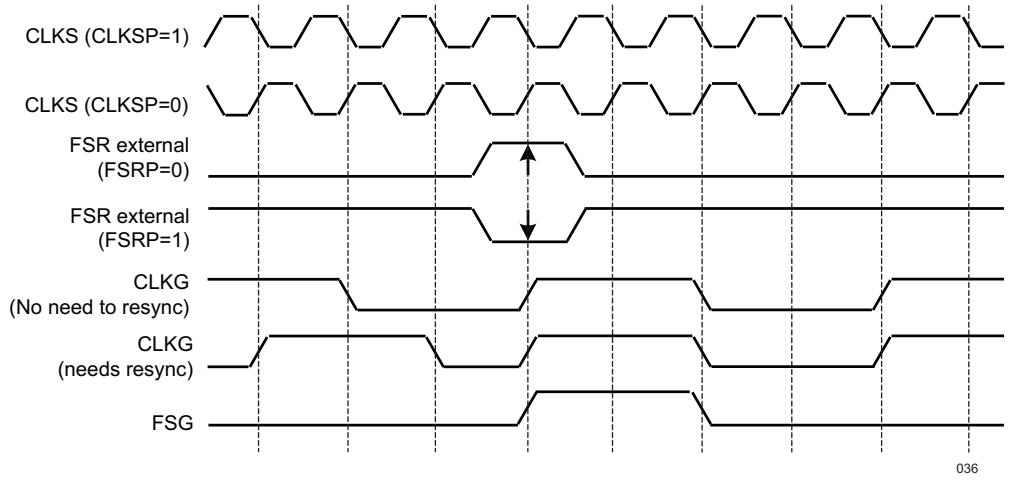


Figure 21-41. CLKG Synchronization and FSG Generation (GSYNC = 1 and CLKGDV = 0x3)



21.4.4 McBSP Exception/Error Conditions

21.4.4.1 Introduction

There are several serial port events that can constitute a system error. Any error conditions can be a source of an interrupt:

- Receiver overrun (McBSPi.MCBSPLP_IRQSTATUS_REG[5] ROVFLSTAT bit is set to 1, and legacy mode McBSPi.MCBSPLP_SPCR1_REG[2] RFULL bit is set to 1)

  This occurs when RB is full and RSR are full with another new word shifted in from mcbspi_dr. Therefore, McBSPi.MCBSPLP_IRQSTATUS_REG[5] ROVFLSTAT (McBSPi.MCBSPLP_SPCR1_REG[2] RFULL) indicates an error condition wherein any new data that can arrive at this time on mcbspi_dr replaces the contents of the RSR, and the previous word is lost. The RSR continues to be overwritten as long as new data arrives on mcbspi_dr and McBSPi.MCBSPLP_DRR_REG register is not read. For more details about overrun in the receiver, see Section 21.4.4.2.

- Unexpected receive frame-synchronization pulse (McBSPi.MCBSPLP_IRQSTATUS_REG[0] RSYNCERR bit is set to 1, and legacy mode McBSPi.MCBSPLP_SPCR1_REG[3] RSYNCERR bit is set to 1).

  This occurs during reception when an unexpected frame-synchronization pulse arrives. An unexpected frame-synchronization pulse is one that is supposed to begin the next frame transfer before all the bits of the current frame have been received. Such a pulse is ignored by the receiver, but sets the McBSPi.MCBSPLP_SPCR1_REG[3] RSYNCERR bit. For more details about receive frame-synchronization errors, see Section 21.4.4.3, *Unexpected Receive Frame-Sync Pulse*.
- Receiver underflow (McBSPi.MCBSPLP_IRQSTATUS_REG[4] RUNDFLSTAT bit is set to '1')

  This occurs when sDMA controller or MPU/IVA2.2 subsystem reads data from an empty receive buffer. For more details about underflow in the receiver, see Section 21.4.4.4.
- Transmitter underflow (McBSPi.MCBSPLP_IRQSTATUS_REG[11] XUNDFLSTAT bit is set to '1', and legacy mode McBSPi.MCBSPLP_SPCR2_REG[2] XEMPTY bit is set to '0')

  If a new frame-synchronization signal arrives when XB is empty, the previous data in the XSR is sent again. This procedure continues for every new frame-synchronization pulse that arrives until McBSPi.MCBSPLP_DXR_REG register is loaded with new data (and the XB is no longer empty). For more details about underflow in the transmitter, see Section 21.4.4.5.
- Unexpected transmit frame-synchronization pulse (McBSPi.MCBSPLP_IRQSTATUS_REG[7] XSYNCERR bit is set to '1', and legacy mode McBSPi.MCBSPLP_SPCR2_REG[3] XSYNCERR bit is set to '1')

  This occurs during transmission when an unexpected frame-synchronization pulse arrives. An unexpected pulse is one that is supposed to begin the next frame transfer before all the bits of the current frame have been transferred. Such a pulse is ignored by the transmitter, but sets the McBSPi.MCBSPLP_SPCR2_REG[3] XSYNCERR bit. For more details see Section 21.4.4.6.
- Transmitter overflow (McBSPi.MCBSPLP_IRQSTATUS_REG[12] XOVFLSTAT bit is set to '1')

  This occurs when sDMA controller or MPU/IVA2.2 subsystem writes data to a full XB. For more details about underflow in the receiver, see Section 21.4.4.7.

21.4.4.2 Overrun in the Receiver

When McBSPi.MCBSPLP_IRQSTATUS_REG[5] ROVFLSTAT bit set to '1', and McBSPi.MCBSPLP_SPCR1_REG[2] RFULL bit set to '1' (legacy mode) indicates that the receiver has experienced overrun and is in an error condition. Receive overrun is set when all of the following conditions are met:

1. McBSPi.MCBSPLP_DRR_REG is not read even if the McBSPi.MCBSPLP_IRQSTATUS_REG[3] RRDY bit is set (legacy mode) and DMA or interrupt request has been asserted.
2. RB is full
3. RSR is full

As previously described, data arriving on mcbspi_dr is continuously shifted into the Receive Shift Register (RSR). Once a complete word is shifted into the RSR, an RSR-to-RB copy can occur only if the RB is not full.

Either of the following events clears the legacy mode McBSPi.MCBSPLP_SPCR1_REG[2] RFULL bit and allows subsequent transfers to be read properly:

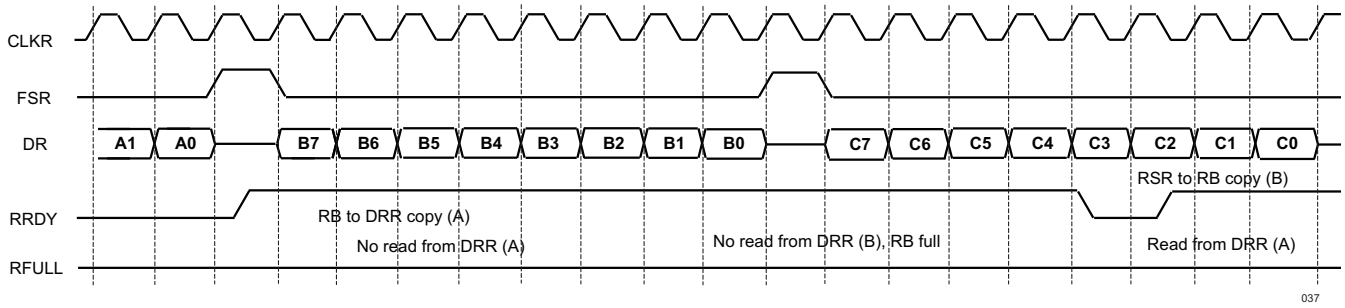
- The MPU/IVA2.2 subsystems or sDMA controller reads McBSPi.MCBSPLP_DRR_REG.
- The receiver is reset individually (McBSPi.MCBSPLP_SPCR1_REG[0] RRST bit =0) or as part of a global reset.

Another frame-synchronization pulse is required to restart the receiver.

According to the McBSPi.MCBSPLP_IRQENABLE_REG register setting, this condition can generate the McBSPi_IRQ line to be asserted low. Writing 1 to the corresponding bit in McBSPi.MCBSPLP_IRQSTATUS_REG register clears the interrupt.

Figure 21-42 shows the receive overrun condition.

Figure 21-42. Overrun in the McBSP Receiver



21.4.4.3 Unexpected Receive Frame-sync Pulse

21.4.4.3.1 Possible Responses to Receive Frame-sync Pulses

If a frame-synchronization pulse starts the transfer of a new frame before the current frame is fully received, this pulse is treated as an unexpected frame-synchronization pulse, and the receiver sets the receive frame-synchronization error bit McBSPi.MCBSPLP_IRQSTATUS_REG[0] RSYNCERR (and the legacy McBSPi.MCBSPLP_SPCR1_REG[3] RSYNCERR bit).

According to the McBSPi.MCBSPLP_IRQENABLE_REG register settings this condition can generate the McBSPi_IRQ line to be asserted low. Writing 1 to the corresponding bit in McBSPi.MCBSPLP_IRQSTATUS_REG register clears the interrupt.

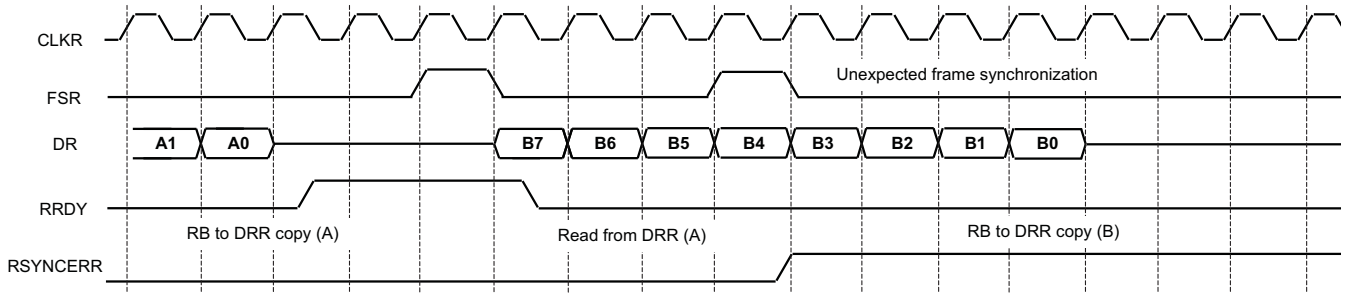
Using the legacy mode, McBSPi.MCBSPLP_SPCR1_REG[3] RSYNCERR bit can be cleared only by a receiver reset or by writing 0 to this bit. If you want the McBSP module to notify the MPU/IVA2.2 subsystem of receive frame-synchronization errors, set the legacy mode receive interrupt with the McBSPi.MCBSPLP_SPCR1_REG[5:4] RINTM field. When RINTM = 0b11, the McBSP module sends a receive interrupt (legacy mode) request to the MPU/IVA2.2 subsystems each time that RSYNCERR is set.

21.4.4.3.2 Example of an Unexpected Receive Frame-sync Pulse

Figure 21-43 shows an unexpected receive frame-synchronization pulse during normal operation of the serial port, with time intervals between data packets.

Note: The unexpected receive frame-synchronization pulse does not influence the data receive process, being ignored by the data receive state-machine.

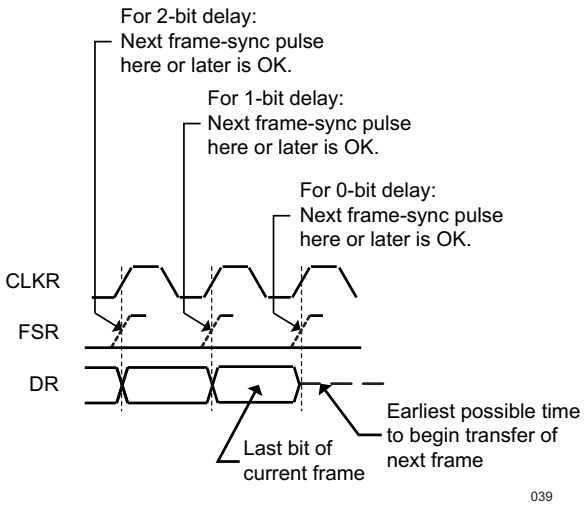
Figure 21-43. Unexpected Frame-sync Pulse During a McBSP Reception



21.4.4.3.3 Preventing Unexpected Receive Frame-sync Pulses

Each frame transfer can be delayed by 0, 1, or 2 CLKR cycles, depending on the value of the McBSPi.MCBSPLP_RCR2_REG[1:0] RDATDLY field. For each possible data delay, Figure 21-44 shows when a new frame-synchronization pulse on FSR can safely occur relative to the last bit of the current frame.

Figure 21-44. Proper Positioning of Receive Frame-sync Pulses



21.4.4.4 Underflow in the Receiver

The McBSP module indicates a receiver underflow condition by setting the McBSPi.MCBSPLP_IRQSTATUS_REG[4] RUNDFLSTAT bit. This error occurs when sDMA controller or MPU/IVA2.2 subsystem reads data from an empty RB this happens only if the MPU/IVA2.2 subsystem or sDMA controller does not respect the DMA length, does not wait for DMA request, or does not check the buffer status before reading data. According to the McBSPi.MCBSPLP_IRQENABLE_REG register settings this condition can generate the McBSPi_IRQ line to be asserted low. Writing 1 to the corresponding bit in McBSPi.MCBSPLP_IRQSTATUS_REG register clears the interrupt.

21.4.4.5 Underflow in the Transmitter

The McBSP module indicates a transmitter empty (or underflow) condition by setting the McBSPi.MCBSPLP_IRQSTATUS_REG[11] XUNDFLSTAT bit. Also the legacy mode McBSPi.MCBSPLP_SPCR2_REG[2] XEMPTY bit is cleared. Either of the following events activates XEMPTY bit (XEMPTY = 0):

- McBSPi.MCBSPLP_DXR_REG has not been loaded and XB is empty, and all bits of the data word in the XSR have been shifted out on the mcbspi_dx pin.
- The transmitter is reset (by forcing McBSPi.MCBSPLP_SPCR2_REG[0] XRST=0, or by an global reset) and is then restarted.

XEMPTY bit is deactivated (XEMPTY=1) when a new word in McBSPi.MCBSPLP_DXR_REG is transferred to Transmit Buffer (XB). If McBSPi.MCBSPLP_PCR_REG[11] FSXM=1 and McBSPi.MCBSPLP_SRGR2_REG[12] FSGM=0, the transmit frame-sync signal (FSX) is generated when Transmit Buffer (XB) is not empty. When McBSPi.MCBSPLP_SRGR2_REG[12] FSGM=0, McBSPi.MCBSPLP_SRGR2_REG[11:0] FPER and McBSPi.MCBSPLP_SRGR1_REG[15:8] FWID are used to determine the frame-synchronization period and width (external FSX is gated by the buffer empty condition). Otherwise, the transmitter waits for the next frame-synchronization pulse before sending out the next frame on mcbspi_dx.

When the transmitter is taken out of reset (McBSPi.MCBSPLP_SPCR2_REG[0] XRST=1), it is in a transmitter ready state (McBSPi.MCBSPLP_SPCR2_REG[1] XRDY bit =1) and transmitter empty (McBSPi.MCBSPLP_SPCR2_REG[2] XEMPTY=0) state. If McBSPi.MCBSPLP_DXR_REG is loaded by

the MPU/IVA2.2 subsystem or the sDMA controller before internal FSX goes active high, a valid XB-to-XSR transfer occurs. This allows for the first word of the first frame to be valid even before the transmit frame-synchronization pulse is generated or detected. Alternatively, if a transmit frame-synchronization pulse is detected before McBSPi.MCBSPLP_DXR_REG is loaded, zeros are output on mcbspi_dx.

The McBSPi.MCBSPLP_IRQSTATUS_REG[11] XUNDFLSTAT bit indicates a real underflow condition, in which the frame is corrupted due to lack of data availability during transmit process. According to the McBSPi.MCBSPLP_IRQENABLE_REG register settings this condition can generate the McBSPi_IRQ line to be asserted low. Writing 1 to the corresponding bit in McBSPi.MCBSPLP_IRQSTATUS_REG register clears the interrupt.

21.4.4.6 Unexpected Transmit Frame-sync Pulse

21.4.4.6.1 Possible Responses to Transmit Frame-sync Pulses

If a frame-synchronization pulse starts the transfer of a new frame before the current frame is fully transmitted, this pulse is treated as an unexpected frame-synchronization pulse, and the transmitter sets the transmit frame-synchronization error bit McBSPi.MCBSPLP_IRQSTATUS_REG[7] XSYNCERR (and the legacy McBSPi.MCBSPLP_SPCR2_REG[3] XSYNCERR bit).

According to the McBSPi.MCBSPLP_IRQENABLE_REG register settings, this condition can generate the McBSPi_IRQ line to be asserted low. Writing 1 to the corresponding bit in status register clears the interrupt.

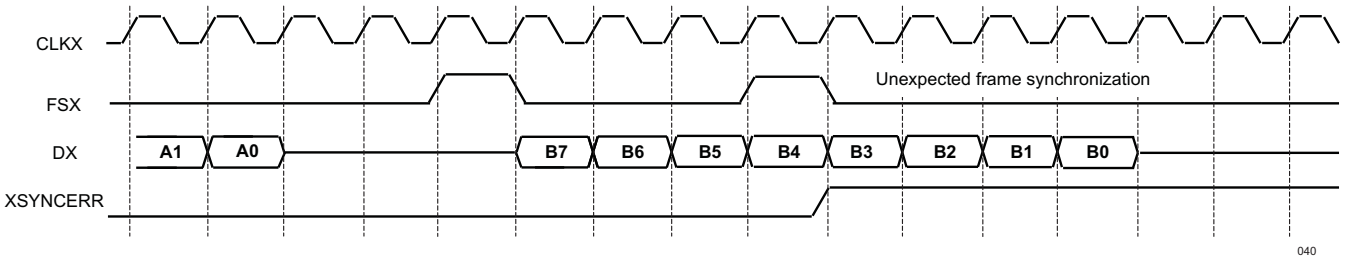
Using the legacy mode, McBSPi.MCBSPLP_SPCR2_REG[3] XSYNCERR bit can be cleared only by a transmitter reset or by a write of 0 to this bit. If you want the McBSP module to notify the MPU/IVA2.2 subsystem of frame-synchronization errors, you can set a special transmit interrupt mode with the McBSPi.MCBSPLP_SPCR2_REG[5:4] XINTM field. When XINTM=0b11, the McBSP module sends a transmit interrupt request to the MPU/IVA2.2 subsystem each time that XSYNCERR is set.

21.4.4.6.2 Example of Unexpected Transmit Frame-Synchronization Pulse

Figure 21-45 shows an unexpected transmit frame-synchronization pulse during normal operation of the serial port with intervals between the data packets.

Note: The unexpected transmit frame-synchronization pulse does not influence the data transmit process, being ignored by the data transmit state-machine.

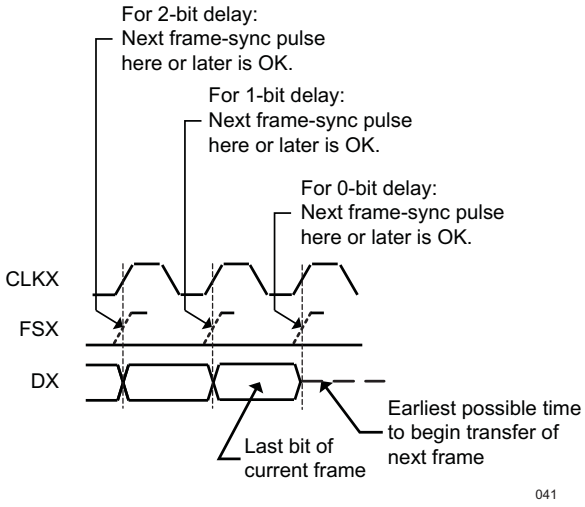
Figure 21-45. Unexpected Frame-sync Pulse During a McBSP Transmission



21.4.4.6.3 Preventing Unexpected Transmit Frame-sync Pulses

Each frame transfer can be delayed by 0, 1, or 2 CLKX cycles, depending on the value in the McBSPi.MCBSPLP_XCR2_REG[1:0] XDATDLY field. For each possible data delay, Figure 21-46 shows when a new frame-synchronization pulse on FSX can safely occur relative to the last bit of the current frame.

Figure 21-46. Proper Positioning of Transmit Frame-sync Pulses



21.4.4.7 Overflow in the Transmitter

The McBSP module indicates a transmitter overflow condition by setting the McBSPi.MCBSPLP_IRQSTATUS_REG[12] XOVFLSTAT bit. This error occurs when sDMA controller or MPU/IVA2.2 subsystem write data to a full XB (this may happen only if the MPU/IVA2.2 subsystem or sDMA controller does not respect the DMA length, does not wait for DMA request or does not check the buffer status before writing data). According to the McBSPi.MCBSPLP_IRQENABLE_REG register settings this condition can generate the McBSPi_IRQ line to be asserted low. Writing 1 to the corresponding bit in status register clears the interrupt.

21.4.5 McBSP DMA Configuration

The McBSP receive and transmit data DMA requests are active after the receive McBSPi.MCBSPLP_SPCR1_REG[0] RRST and transmit McBSPi.MCBSPLP_SPCR2_REG[0] XRST are released. After reset the default DMA threshold (and length) is one.

The receive and transmit DMA requests can be individually disabled by setting the McBSPi.MCBSPLP_RCCR_REG[3] RDMAEN, McBSPi.MCBSPLP_XCCR_REG[3] XDMAEN bits to 0. When disabling the DMA, the DMA request line is de-asserted even if a DMA transfer is pending and the DMA state-machine is not reset.

The DMA threshold and length configuration is done through McBSPi.MCBSPLP_THRSH1_REG and McBSPi.MCBSPLP_THRSH2_REG registers as follows:

- (THRSH1_REG + 1) value represents the required receive DMA request length (the length of the transfer is the same as the threshold value plus one). As long as the RB occupied locations level is above or equal to the THRSH1_REG value + 1, the DMA request is asserted. After transferring the configured (THRSH1_REG + 1) number of words, the receive DMA request is de-asserted and reasserted as soon as the conditions are met again.
- (THRSH2_REG + 1) value represents the required transmit DMA request length (the length of the transfer is the same as the threshold value plus one). As long as the XB free locations level is above or equal to the THRSH2_REG value + 1, the DMA request is asserted. After transferring the configured (THRSH2_REG + 1) number of words, the transmit DMA request is de-asserted and reasserted as soon as the conditions are met again.

Note: The MPU/IVA2.2 subsystem can decide not to use the DMA to transfer the data. In this case, the DMA must be disabled (or the DMA request can be ignored by MPU/IVA2.2 subsystem) and the common interrupt line (McBSPi_IRQ) can be used. The McBSPi.MCBSPLP_SPCR1_REG[1] RRDY bit for receive and McBSPi.MCBSPLP_SPCR2_REG[1] XRDY bit for transmit will indicate when the threshold values are reached. Also, by reading the receive buffer status McBSPi.MCBSPLP_RBUFFSTAT_REG register and transmit buffer status McBSPi.MCBSPLP_XBUFFSTAT_REG register, the MPU/IVA2.2 subsystem can decide to transfer data even if the threshold is not reached. This mechanism is useful on the last transfer on receive side when the threshold value is bigger than the occupied locations inside the receive buffer and the MPU/IVA2.2 subsystem needs to read this data. Since no interrupt or DMA request is asserted the only option in this case is to read the RB status register value and to transfer the remaining data without using the DMA or interrupt indication.

21.4.6 Multichannel Selection Modes

21.4.6.1 Channels, Blocks, and Partitions

A McBSP channel is a time slot for shifting in/out the bits of one serial word. The McBSP module supports up to 128 channels for reception and 128 channels for transmission. In the receiver and in the transmitter, the 128 available channels are divided into eight blocks that contain 16 contiguous channels each:

- Block 0: Channels 0–15
- Block 1: Channels 16–31
- Block 2: Channels 32–47
- Block 3: Channels 48–63
- Block 4: Channels 64–79
- Block 5: Channels 80–95
- Block 6: Channels 96–111
- Block 7: Channels 112–127

The blocks are assigned to partitions according to the selected partition mode. In the two-partition mode described in Section 21.4.6.6, you assign one even-numbered block (0, 2, 4, or 6) to partition A and one odd-numbered block (1, 3, 5, or 7) to partition B. In the 8-partition mode Section 21.4.6.4, blocks 0 through 7 are automatically assigned to partitions A through H, respectively.

The number of partitions for reception and the number of partitions for transmission are independent of each other. For example, it is possible to use two receive partitions (A and B) and eight transmit partitions (A–H).

21.4.6.2 Multichannel Selection

When a McBSP module uses a time-division multiplexed (TDM) data stream while communicating with other McBSP modules or serial devices, the McBSP module may need to receive and/or transmit on only a few channels. To save memory and bus bandwidth, you can use a multichannel selection mode to prevent data flow in some of the channels.

Each channel partition has a dedicated channel enable register. If the appropriate multichannel selection mode is on, each bit in the register controls whether data flow is allowed or prevented in one of the channels that is assigned to that partition.

The McBSP module has one receive multichannel selection mode Section 21.4.6.5, and three transmit multichannel selection modes Section 21.4.6.7.

21.4.6.3 Configuring a Frame for Multichannel Selection

Before enabling a multichannel selection mode, make sure you properly configure the data frame:

- Select a single-phase frame (McBSPi.MCBSPLP_RCR2_REG[15] RPHASE bit and McBSPi.MCBSPLP_XCR2_REG[15] XPHASE bit = 0). Each frame represents a TDM data stream.

- Set a frame length (in McBSPi.MCBSPLP_RCR1_REG[14:8] RFRLEN1 field and in McBSPi.MCBSPLP_XCR1_REG[14:8] XFRLEN1 field) that includes the highest-numbered channel to be used. For example, if you plan to use channels 0, 15, and 39 for reception, the receive frame length must be at least 40 (RFRLEN1 = 39). If RFRLEN1 = 39 in this case, the receiver creates 40 time slots per frame but only receives data during time slots 0, 15, and 39 of each frame.

21.4.6.4 Using Eight Partitions

For multichannel selection operation in the receiver and/or the transmitter, you can use eight partitions or two partitions (as previously described). If you choose the 8-partition mode (McBSPi.MCBSPLP_MCR1_REG[9] RMCME = 1 for reception, McBSPi.MCBSPLP_MCR2_REG[9] XMCME = 1 for transmission), McBSP channels are activated in the following order: A, B, C, D, E, F, G, H.

In response to a frame-synchronization pulse, the receiver or transmitter begins with the channels in partition A and then continues with the other partitions in order until the complete frame has been transferred. When the next frame-synchronization pulse occurs, the next frame is transferred, beginning with the channels in partition A.

In the 8-partition mode, the McBSPi.MCBSPLP_MCR1_REG[6:5] RPABLK and McBSPi.MCBSPLP_MCR2_REG[6:5] XPABLK, and McBSPi.MCBSPLP_MCR1_REG[8:7] RPBBLK and McBSPi.MCBSPLP_MCR2_REG[8:7] XPBBLK bit fields are ignored, and the 16 channel blocks are assigned to the partitions as shown in Table 21-20 through Table 21-21. These assignments cannot be changed. The tables also show the registers used to control the channels in the partitions.

Table 21-20. Eight Partitions – Receive Channel Assignment and Control

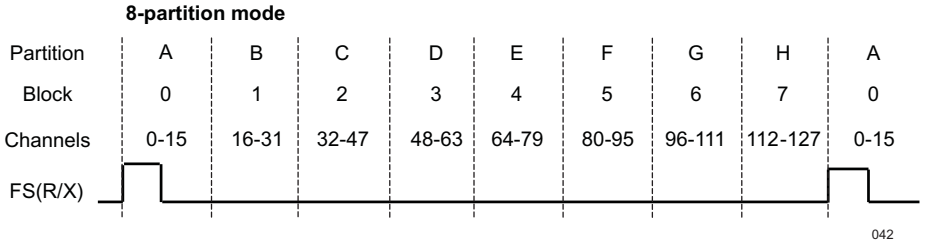
Receive Partition	Assigned Block of Receive Channels	Register Used for Channel Control
A	Block 0: Channels 0 through 15	McBSPi.MCBSPLP_RCERA_REG
B	Block 1: Channels 16 through 31	McBSPi.MCBSPLP_RCERB_REG
C	Block 2: Channels 32 through 47	McBSPi.MCBSPLP_RCERC_REG
D	Block 3: Channels 48 through 63	McBSPi.MCBSPLP_RCERD_REG
E	Block 4: Channels 64 through 79	McBSPi.MCBSPLP_RCERE_REG
F	Block 5: Channels 80 through 95	McBSPi.MCBSPLP_RCERF_REG
G	Block 6: Channels 96 through 111	McBSPi.MCBSPLP_RCERG_REG
H	Block 7: Channels 112 through 127	McBSPi.MCBSPLP_RCERH_REG

Table 21-21. Eight Partitions – Transmit Channel Assignment and Control

Transmit Partition	Assigned Block of Receive Channels	Register Used for Channel Control
A	Block 0: Channels 0 through 15	McBSPi.MCBSPLP_XCERA_REG
B	Block 1: Channels 16 through 31	McBSPi.MCBSPLP_XCERB_REG
C	Block 2: Channels 32 through 47	McBSPi.MCBSPLP_XCERC_REG
D	Block 3: Channels 48 through 63	McBSPi.MCBSPLP_XCERD_REG
E	Block 4: Channels 64 through 79	McBSPi.MCBSPLP_XCERE_REG
F	Block 5: Channels 80 through 95	McBSPi.MCBSPLP_XCERF_REG
G	Block 6: Channels 96 through 111	McBSPi.MCBSPLP_XCERG_REG
H	Block 7: Channels 112 through 127	McBSPi.MCBSPLP_XCERH_REG

Figure 21-47 shows an example of the McBSP using the 8-partition mode. In response to a frame-synchronization pulse, the McBSP module begins a frame transfer with partition A and then activates B, C, D, E, F, G, and H to complete a 128-word frame.

Figure 21-47. McBSP Data Transfer in 8-Partition Mode



21.4.6.5 Receive Multichannel Selection Mode

The McBSPi.MCBSPLP_MCR1_REG[0] RMCM bit determines whether all channels or only selected channels are enabled for reception.

- When RMCM = 0, all 128 receive channels are enabled and cannot be disabled.
- When RMCM = 1, the receive multichannel selection mode is enabled. In this mode:
  - Channels can be individually enabled or disabled. The enabled channels are those selected in the appropriate receive channel enable registers (McBSPi.MCBSPLP_RCERA_REG to McBSPi.MCBSPLP_RCERH_REG). The channels are assigned to the McBSPi.MCBSPLP_RCERA_REG to McBSPi.MCBSPLP_RCERH_REG registers depends on the number of receive channel partitions (2 or 8), as defined by the McBSPi.MCBSPLP_MCR1_REG[9] RMCME bit.
  - If a receive channel is disabled, any bits received in that channel are not transferred to the RB, and as a result, the receiver ready bit (RRDY) is not set. Therefore, no DMA synchronization event is generated and, if the receiver interrupt mode depends on RRDY (McBSPi.MCBSPLP_SPCR1_REG[5:4] RINTM = 0b00), no interrupt is generated.

As an example of how the McBSP module behaves in the receive multichannel selection mode, suppose you enable only channels 0, 15, and 39 and that the frame length is 40. The McBSP module:

1. Accepts bits shifted in from the mcbspi_dr pin in channel 0
2. Ignores bits received in channels 1–14
3. Accepts bits shifted in from the mcbspi_dr pin in channel 15
4. Ignores bits received in channels 16–38
5. Accepts bits shifted in from the mcbspi_dr pin in channel 39

21.4.6.6 Using Two Partitions (Legacy Only)

For multichannel selection operation in the receiver and/or the transmitter, you can use two partitions or eight partitions. If you choose the 2-partition mode (McBSPi.MCBSPLP_MCR1_REG[9] RMCME = 0 for reception, McBSPi.MCBSPLP_MCR2_REG[9] XMCME = 0 for transmission), McBSP channels are activated using an alternating scheme. In response to a frame-synchronization pulse, the receiver or transmitter begins with the channels in partition A and then alternates between partitions B and A until the complete frame has been transferred. When the next frame-synchronization pulse occurs, the next frame is transferred beginning with the channels in partition A.

For reception, any two of the eight receive-channel blocks can be assigned to receive partitions A and B, which means up to 32 receive channels can be enabled at any given point. Similarly, any two of the eight transmit-channel blocks (up 32 enabled transmit channels) can be assigned to transmit partitions A and B.

For reception:

- Assign an even-numbered channel block (0, 2, 4, or 6) to receive partition A by writing to the McBSPi.MCBSPLP_MCR1_REG[6:5] RPABLK field. In the receive multichannel selection mode, the channels in this partition are controlled by receive channel enable register A (McBSPi.MCBSPLP_RCERA_REG).

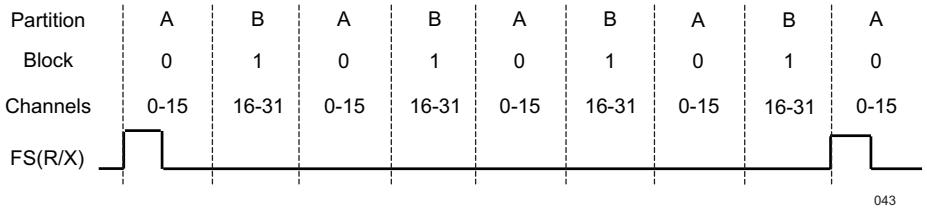
- Assign an odd-numbered block (1, 3, 5, or 7) to receive partition B with the McBSPi.MCBSPLP_MCR1_REG[8:7] RPBBLK field. In the receive multichannel selection mode, the channels in this partition are controlled by receive channel enable register B (McBSPi.MCBSPLP_RCERB_REG).

For transmission:

- Assign an even-numbered channel block (0, 2, 4, or 6) to transmit partition A by writing to the McBSPi.MCBSPLP_MCR2_REG[6:5] XPABLK fields. In one of the transmit multichannel selection modes, the channels in this partition are controlled by transmit channel enable register A (McBSPi.MCBSPLP_XCERA_REG).
- Assign an odd-numbered block (1, 3, 5, or 7) to transmit partition B with the McBSPi.MCBSPLP_MCR1_REG[8:7] XPBBLK field. In one of the transmit multichannel selection modes, the channels in this partition are controlled by transmit channel enable register B (McBSPi.MCBSPLP_XCERB_REG).

Figure 21-48 shows an example of alternating between the channels of partition A and the channels of partition B. Channels 0–15 have been assigned to partition A, and channels 16–31 have been assigned to partition B. In response to a frame-synchronization pulse, the McBSP module begins a frame transfer with partition A and then alternates between partitions B and A until the complete frame is transferred.

Figure 21-48. Alternating Between Partitions A and B Channels



21.4.6.7 Transmit Multichannel Selection Modes

The McBSPi.MCBSPLP_MCR2_REG[1:0] XMCM field determine whether all channels or only selected channels are enabled and unmasked for transmission. The McBSP module has three transmit multichannel selection modes (XMCM = 0b01, XMCM = 0b10, and XMCM = 0b11), which are described in Table 21-22.

Table 21-22. Selecting a Transmit Multichannel Selection Mode With the XMCM Bit Field

XMCM	Transmit Multichannel Selection Mode
0b00	No transmit multichannel selection mode is on. All channels are enabled and unmasked. No channels can be disabled or masked.
0b01	All channels are disabled unless they are selected in the appropriate transmit channel enable registers (McBSPi.MCBSPLP_XCERA_REG/McBSPi.MCBSPLP_XCERH_REG). If enabled, a channel in this mode is also unmasked. The McBSPi.MCBSPLP_MCR2_REG[9] XMCME bit determines whether 32 channels or 128 channels are selectable in the McBSPi.MCBSPLP_XCERA_REG/McBSPi.MCBSPLP_XCERH_REG registers.
0b10	All channels are enabled, but they are masked unless they are selected in the appropriate transmit channel enable registers (McBSPi.MCBSPLP_XCERA_REG/McBSPi.MCBSPLP_XCERH_REG). The McBSPi.MCBSPLP_MCR2_REG[9] XMCME bit determines whether 32 channels or 128 channels are selectable in the McBSPi.MCBSPLP_XCERA_REG/McBSPi.MCBSPLP_XCERH_REG registers.
0b11	This mode is used for symmetric transmission and reception. All channels are disabled for transmission unless they are enabled for reception in the appropriate receive channel enable registers (McBSPi.MCBSPLP_RCERA_REG/McBSPi.MCBSPLP_RCERH_REG). Once enabled, they are masked unless they are also selected in the appropriate transmit channel enable registers (McBSPi.MCBSPLP_XCERA_REG/McBSPi.MCBSPLP_XCERH_REG). The McBSPi.MCBSPLP_MCR2_REG[9] XMCME bit determines whether 32 channels or 128 channels are selectable in McBSPi.MCBSPLP_RCERA_REG/McBSPi.MCBSPLP_RCERH_REG registers and McBSPi.MCBSPLP_XCERA_REG/McBSPi.MCBSPLP_XCERH_REG registers.

As an example of how the McBSP module behaves in a transmit multichannel selection mode, suppose that XMCM = 0b01 (all channels disabled unless individually enabled) and that you have enabled only channels 0, 15, and 39. Suppose also that the frame length is 40. The McBSP module:

1. Shifts data to the mcbspi_dx pin in channel 0
2. Places the mcbspi_dx pin in the high impedance state in channels 1–14
3. Shifts data to the mcbspi_dx pin in channel 15
4. Places the mcbspi_dx pin in the high impedance state in channels 16–38
5. Shifts data to the mcbspi_dx pin in channel 39

21.4.6.7.1 Disabling/Enabling Versus Masking/Unmasking

For transmission, a channel can be:

- Enabled and unmasked (transmission can begin and can be completed)
- Enabled but masked (transmission can begin but cannot be completed)
- Disabled (transmission cannot occur)

The definitions in Table 21-23 explain the channel control options:

Table 21-23. McBSP Channel Control Options

Enabled channel	A channel that can begin transmission by passing data from the data transmit register (McBSPi.MCBSPLP_DXR_REG) to the XSR through XB.
Masked channel	A channel that cannot complete transmission. The mcbspi_dx pin is held in the high impedance state; data cannot be shifted out on the mcbspi_dx pin. In systems where symmetric transmit and receive provide software benefits, this feature allows transmit channels to be disabled on a shared serial bus. A similar feature is not needed for reception because multiple receptions cannot cause serial bus contention.
Disabled channel	A channel that is not enabled. A disabled channel is also masked. Because no DXR–to–XB copy occurs, the McBSPi.MCBSPLP_SPCR2_REG[1] XRDY bit is not set. Therefore, no DMA synchronization event is generated, and if the transmit interrupt mode depends on XRDY (McBSPi.MCBSPLP_SPCR2_REG[5:4] XINTM=00b), no interrupt is generated. The McBSPi.MCBSPLP_SPCR2_REG[2] XEMPTY bit is not affected.
Unmasked channel	A channel that is not masked. Data in the XSR(s) is shifted out on the mcbspi_dx pin.

21.4.6.7.2 Activity on McBSP Pins for Different Values of XMCM

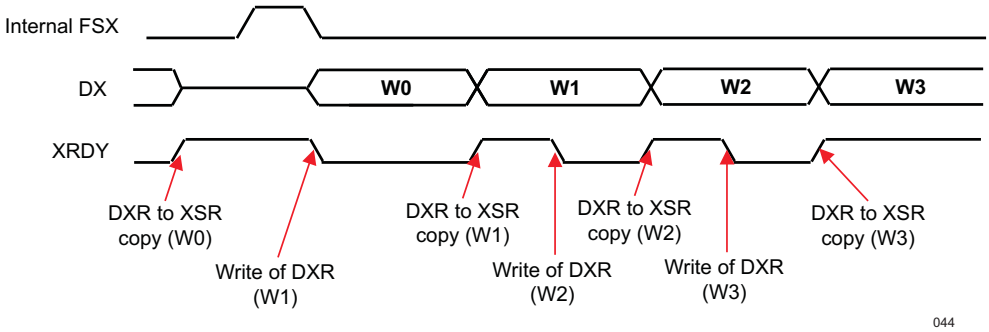
Figure 21-49 shows the activity on the McBSP pins for the various McBSPi.MCBSPLP_MCR2_REG[1:0] XMCM values. In all cases, the transmit frame is configured as follows:

- XPHASE=0: Single–phase frame (required for multichannel selection modes)
- XFRLEN1=0b0000011: 4 words per frame
- XWDLEN1=0b000: 8 bits per word
- XMCME=0: 2–partition mode (only partitions A and B used)

In the case where McBSPi.MCBSPLP_MCR2_REG[1:0] XMCM=0b11, transmission and reception are symmetric, which means the corresponding bits for the receiver (RPHASE, RFRLEN1, RWDLEN1, and RMCME) must have the same values as XPHASE, XFRLEN1, and XWDLEN1, respectively.

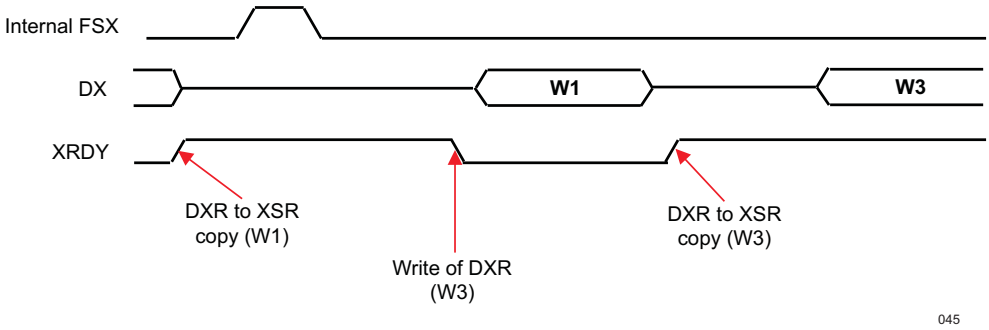
In Figure 21-49, the arrows showing where the various events occur are only sample indications. Wherever possible, there is a time window in which these events can occur.

Figure 21-49. Activity on McBSP Pins When XMCM=0b00



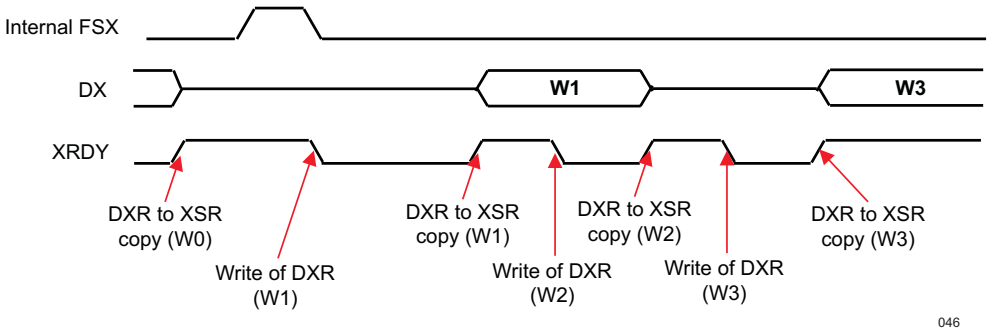
If XMCM = 0b00, all channels are enabled and unmasked. Words W0, W1, W2, and W3 are written to the XB, then, from the XB, there are transferred by mcbspi_dx.

Figure 21-50. Activity on McBSP Pins When XMCM=0b01



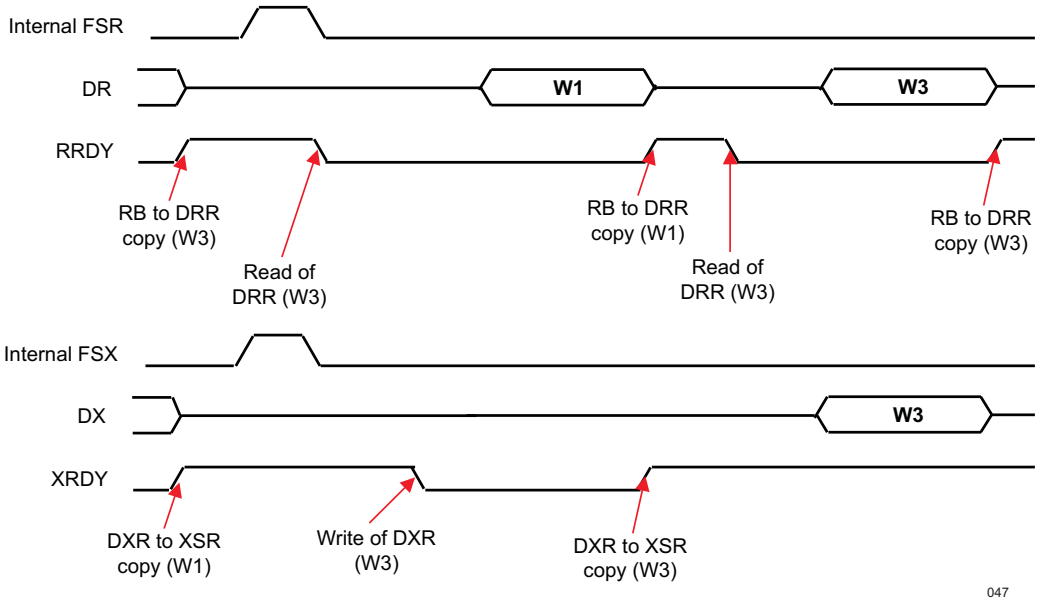
In Figure 21-50 if XMCM = 0b01, XPABLK = 0b00, and XCERA = 0b1010, only channels 1 and 3 are enabled and unmasked. Words W1 and W3 are written to the XB, then, from the XB, there are transferred by mcbspi_dx.

Figure 21-51. Activity on McBSP Pins When XMCM=0b10



In Figure 21-51 if XMCM = 0b10, XPABLK = 0b00, and XCERA = 0b1010, all channels are enabled, only 1 and 3 unmasked. Words W0, W1, W2, and W3 are written to the XB, but only W1 and W3, from the XB, there are transferred by mcbspi_dx.

Figure 21-52. Activity on McBSP Pins When XMCM=0b11



In Figure 21-52 if XMCM = 0b11, RPABLK = 0b00, XPABLK = 0bX, RCERA = 0b1010 and XCERA = 0b1000, channels 1 and 3 are enabled in receive and transmit mode, but only 3 unmasked. Words W1 and W3 are written to the XB, but only W3, from the XB, there are transferred by mcbspi_dx.

21.4.7 SIDETONE Mode (ALP)

21.4.7.1 Introduction

In multimedia-rich mobile communication devices, loopback signals from audio inputs to audio outputs are renamed. A traditional example is the telephone SIDETONE (that is, the telephone user expects to also hear his own voice in the earpiece).

Some of the features using the loopback have strict delay requirements.

It is required that two of the audio input channels be looped back, filtered, and mixed to the corresponding two audio output channels. The SIDETONE mode filters and applies gain to each sample received.

21.4.7.2 SIDETONE Interface

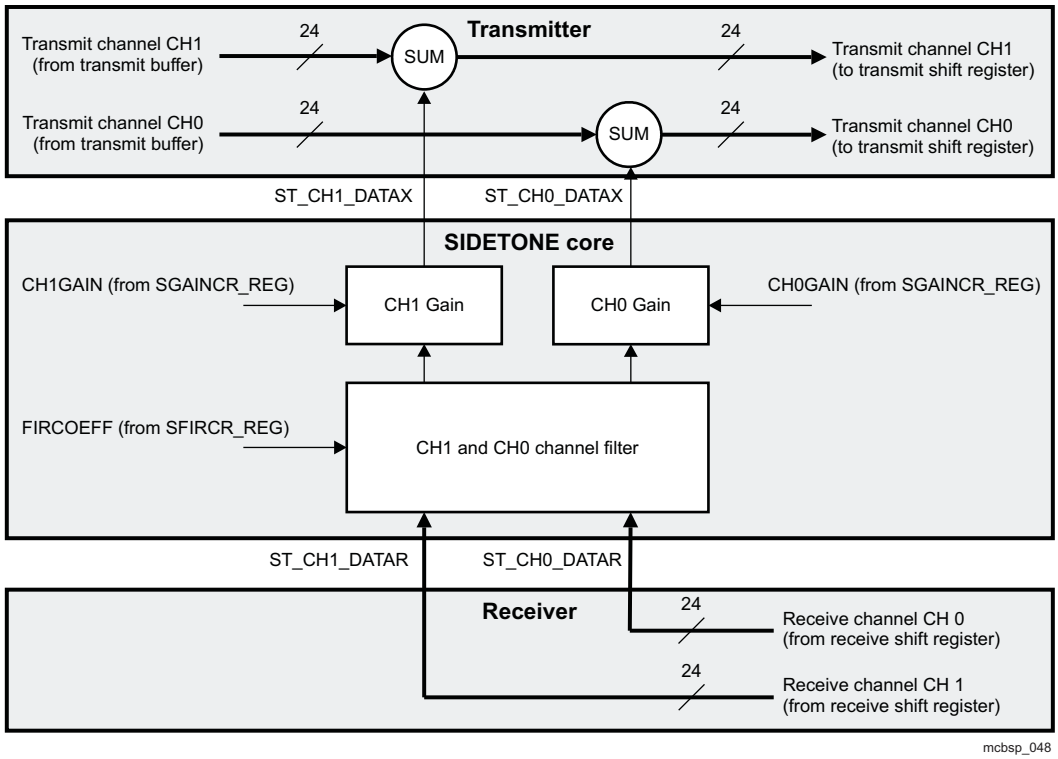
The data from digital microphone, two (out of four) channels, can be configured to be input in the external SIDETONE core. After filtering, the data from digital microphone data is mixed and sent out to the speaker output channels using two (out of eight) configured output channels (separate configuration bits are used). The McBSP module synchronizes the incoming data (filtered by the external SIDETONE core). The transmit and receive part of the McBSP module are not required to operate on the same functional frequency.

The SIDETONE interface offers the following features:

- Send out two 24-bit data channels for the configured SIDETONE received channels (channels can be the same)
- Send out two control signals to indicate to the SIDETONE module that the data is valid (toggle signals which are changing the value from 0 to 1 or 1 to 0 each time a new data is available)
- Receive two 24-bit filtered data channels from the external SIDETONE module and send these channels to the configured transmit channels after mixing the data with the incoming data (from the L4 interface). The sum between the incoming data and SIDETONE loopback data is a saturated sum.
- Receive two control signals to indicate that the SIDETONE module filtered data is available (toggle signals which are changing the value from 0 to 1 or 1 to 0 each time a new data is available).

Figure 21-53 shows the SIDETONE external module data path:

Figure 21-53. SIDETONE Data Path



Before you enable a SIDETONE selection mode, make sure you properly configure the data frame for multichannel and SIDETONE mode:

- Select a single–phase frame (McBSPi.MCBSPLP_RCR2_REG[15] RPHASE bit and McBSPi.MCBSPLP_XCR2_REG[15] XPHASE bit=0). Each frame represents a TDM data stream.
- Set McBSPi.MCBSPLP_MCR1_REG[0] RMCM=1 to select multichannel mode enable.
- Set a frame length (in McBSPi.MCBSPLP_RCR1_REG[14:8] RFRLEN1 bit field and in McBSPi.MCBSPLP_XCR1_REG[14:8] XFRLEN1 bit field) that includes the highest–numbered channel to be used (a maximum of 4 channels can be used in this configuration).
- Set a word length (in McBSPi.MCBSPLP_RCR1_REG[7:5] RWDLEN1 bit field and in McBSPi.MCBSPLP_XCR1_REG[7:5] XWDLEN1 bit field) to be either 16, 24 or 32 (see note).
- Select the input/output channels configured as SIDETONE channels and enable SIDETONE by setting the McBSPi.MCBSPLP_SSELCR_REG[1:0] ICH0ASSIGN/McBSPi.MCBSPLP_SSELCR_REG[6:4] OCH0ASSIGN , McBSPi.MCBSPLP_SSELCR_REG[3:2] ICH1ASSIGN/McBSPi.MCBSPLP_SSELCR_REG[9:7] OCH1ASSIGN fields (2 out of 4 channels external SIDETONE assignment) and McBSPi.MCBSPLP_SSELCR_REG[10] SIDETONEEN bit to 1.

Note: Word width in the loop is 24 bits. If input channel word width is less than 24 bits, LSBs of the samples are zero padded. If input channel word width is more than 24 bits, samples are truncated.

Figure 21-54 describes the data exchange protocol between McBSP module and SIDETONE core:

Figure 21-54. McBSP to SIDETONE Data Exchange

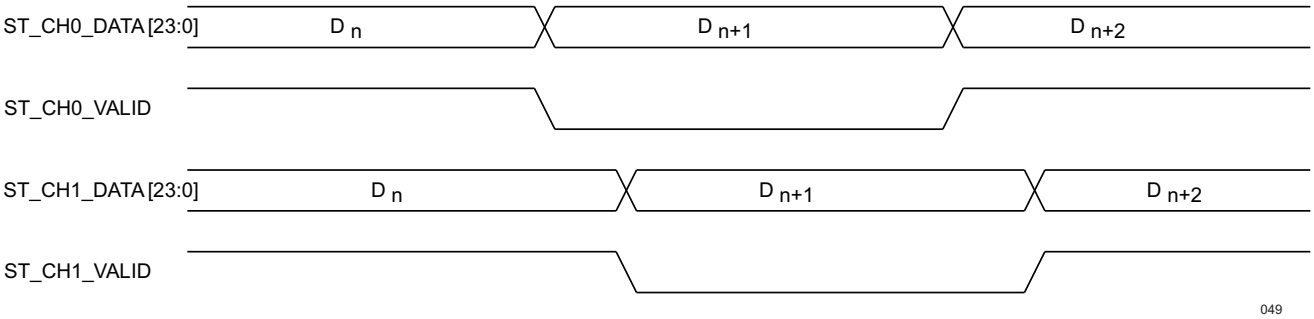
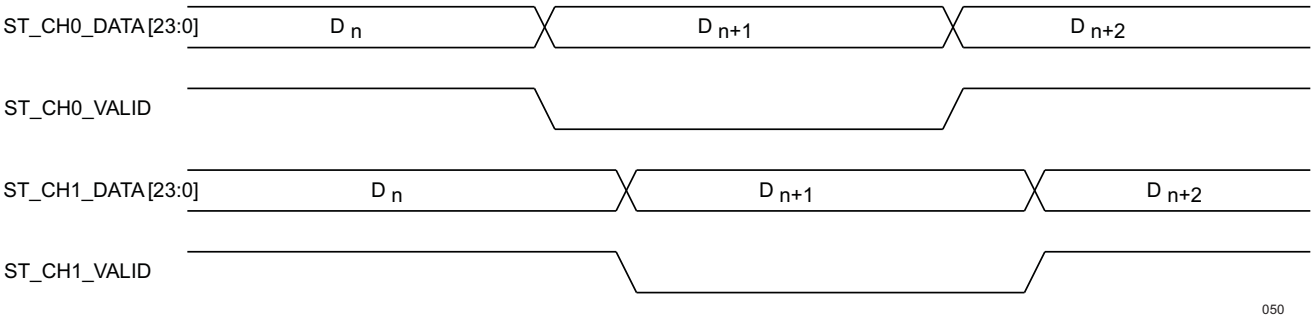


Figure 21-55 describes the data exchange protocol between SIDETONE core and McBSP module:

Figure 21-55. SIDETONE to McBSP Data Exchange



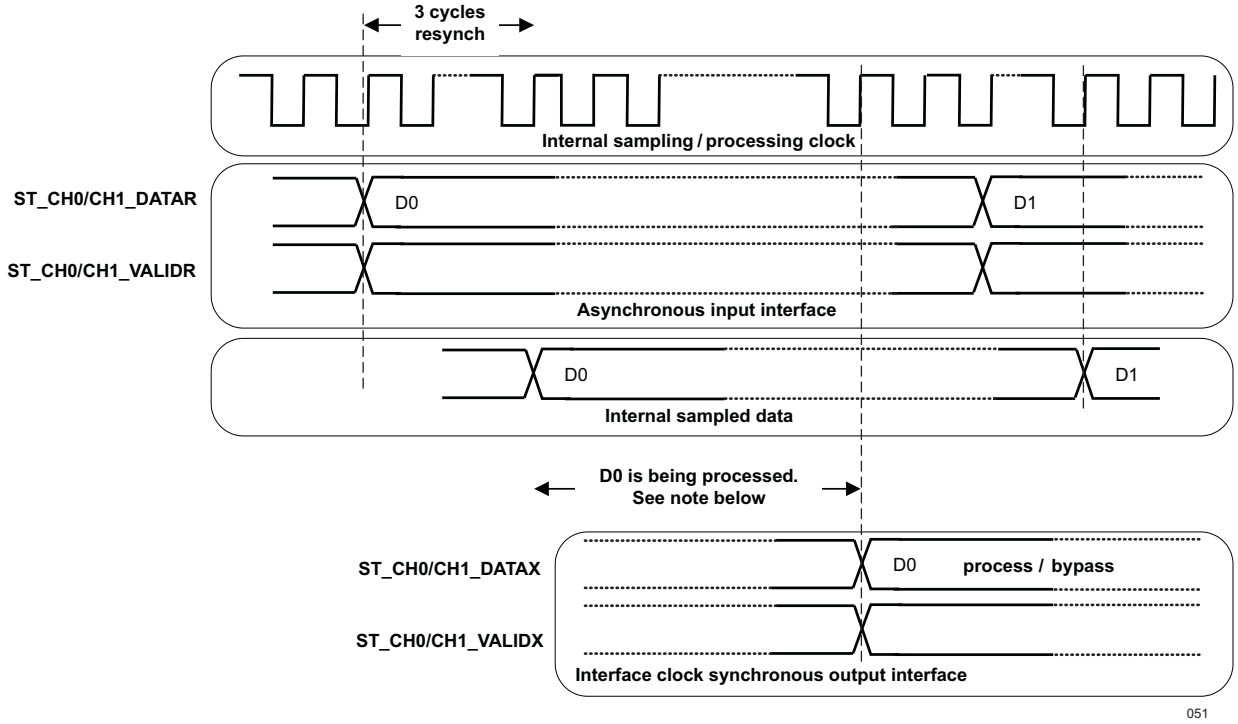
Note: To use the same input channel as source for both SIDETONE output channels the McBSPi.MCBSPLP_SSELCR_REG[1:0] ICH0ASSIGN should be equal to McBSPi.MCBSPLP_SSELCR_REG[3:2] ICH1ASSIGN. The McBSP module does not support two input channels to be assigned to only one SIDETONE output channel.

21.4.7.3 Data Processing Path

The SIDETONE core receives the data to be processed through two 24-bits parallel data interfaces, one for each audio channel. When enabled through McBSPi.ST_SSELCR_REG[0] SIDETONEEN bit-field, the module applies the filtering and gain functions to each data item (frame) and outputs it through two similar 24-bits parallel data interfaces.

Figure 21-56 below describes how the data is sampled and output through the dedicated data interfaces.

Figure 21-56. SIDETONE Processed Data Interfaces



Note: The processing time needed is 132 clock cycles (SIDETONE enabled). The total number of cycles needed for outputting the new processed data upon arrival of a sample is maximum 135 (including synchronizations). When not enabled, the data response takes maximum 5 clock cycles.

The ST_CH0_VALIDR and ST_CH1_VALIDR are 1 bit inputs and their toggling information is used to signal that new data is valid for sampling from ST_CH0_DATAR / ST_CH1_DATAR.

The ST_CH0_VALIDX and ST_CH1_VALIDX are 1 bit outputs and their toggling information is used to signal that new data is valid on the output data bus ST_CH0_DATAX / ST_CH1_DATAX.

21.4.7.4 Data Processing

The processing consists in 2 stages filtering and applying gain to signal. The whole process takes 132 clock cycles.

If a new frame comes too early causing the interrupt assertion, the current frame will be completely processed and this new frame will be ignored. Consequent frames will also be ignored until the current frame processing has ended.

21.4.7.4.1 Filtering

A 128 length FIR filter scheme is used. The module must process the two channels in parallel so two instances of filter will work in parallel sharing only the same coefficients.

Samples data are signed values in interval (-1..1) in Q23 format, negative values are expressed in 2's complement. Coefficients are also signed values in interval (-1..1) in Q15 format, negative values are expressed in 2's complement too.

The module handles overflows in the sums of the product but the user should choose the coefficients with the sum of magnitudes in absolute value is smaller than 1, otherwise the user must ensure that gain applying brings the samples value below |1|, to avoid saturation.

$|C0| + |C1| + \ldots + |C127| < 1$

CAUTION

The coefficients cannot be loaded while the filtering is active (SIDETONE enabled).

21.4.7.4.2 Applying Gain

Each output sample will be multiplied with the gain value specified in the McBSPi.ST_SGAINCR_REG. The gain is independent for each channel and can be modified anytime through L4-interface with immediate effect. Gain values are in the interval (-2..2) in Q1.14 format, negative values are expressed in 2's complement.

The user must choose the gain's value according to the magnitude of the samples to avoid overflow in the final product between the FIR data and gain value. If an overflow occurs, the output data is saturated.

21.4.7.4.3 Enabling SIDETONE

When the SIDETONE operation is enabled (McBSPi.ST_SSELCR_REG[0] SIDETONEEN is 1), the module starts collecting samples received through the data interface without returning any sample or toggling any of the ST_CH0/CH1_VALIDX outputs. The first 127 samples for each channel are only accumulated, no processed data is provided. The first processed frame comes after receiving the 128th sample (with the specific delay).

When the SIDETONE operation is disabled, data is output through the data interface as it comes on the data input interface with the resynchronization latency of maximum 5 clock cycles. Re-enabling it requires waiting another 128 samples before providing new processed samples (each transition of McBSPi.ST_SSELCR_REG[0] SIDETONEEN from 0 to 1 triggers this initialization process).

Note: After reset, the McBSPi.ST_SSELCR_REG[0] SIDETONEEN bit is 0.

21.4.7.4.4 FIR Accuracy

All the arithmetic inside the module is performed without any truncation/saturation until the last stage – the gain applying, where the result of the product between gain and FIR value is saturated to +/-1 if overflow occurs and the LSBs are truncated.

21.4.7.5 Interrupt Operation

The SIDETONE core has a single interrupt line and a single event that may trigger the interrupt. The event (an error one) occurs when the input interface data rate overflows the processing ability of the module. As described in the data processing path section, the SIDETONE core completes a frame processing in 132 cycles. If the input frame rate for any channel exceeds 1 frame/132 x interface clock period while SIDETONE is enabled , the McBSPi.ST_IRQSTATUS_REG[0] OVRRERROR bit in is set and, if McBSPi.ST_IRQENABLE_REG[0] OVRRERROREN bit is set to 1, the interrupt line is asserted.

21.5 McBSP Basic Programming Model

This section describes the programming model of typical McBSP module and SIDETONE core applications.

21.5.1 McBSP Core

CAUTION

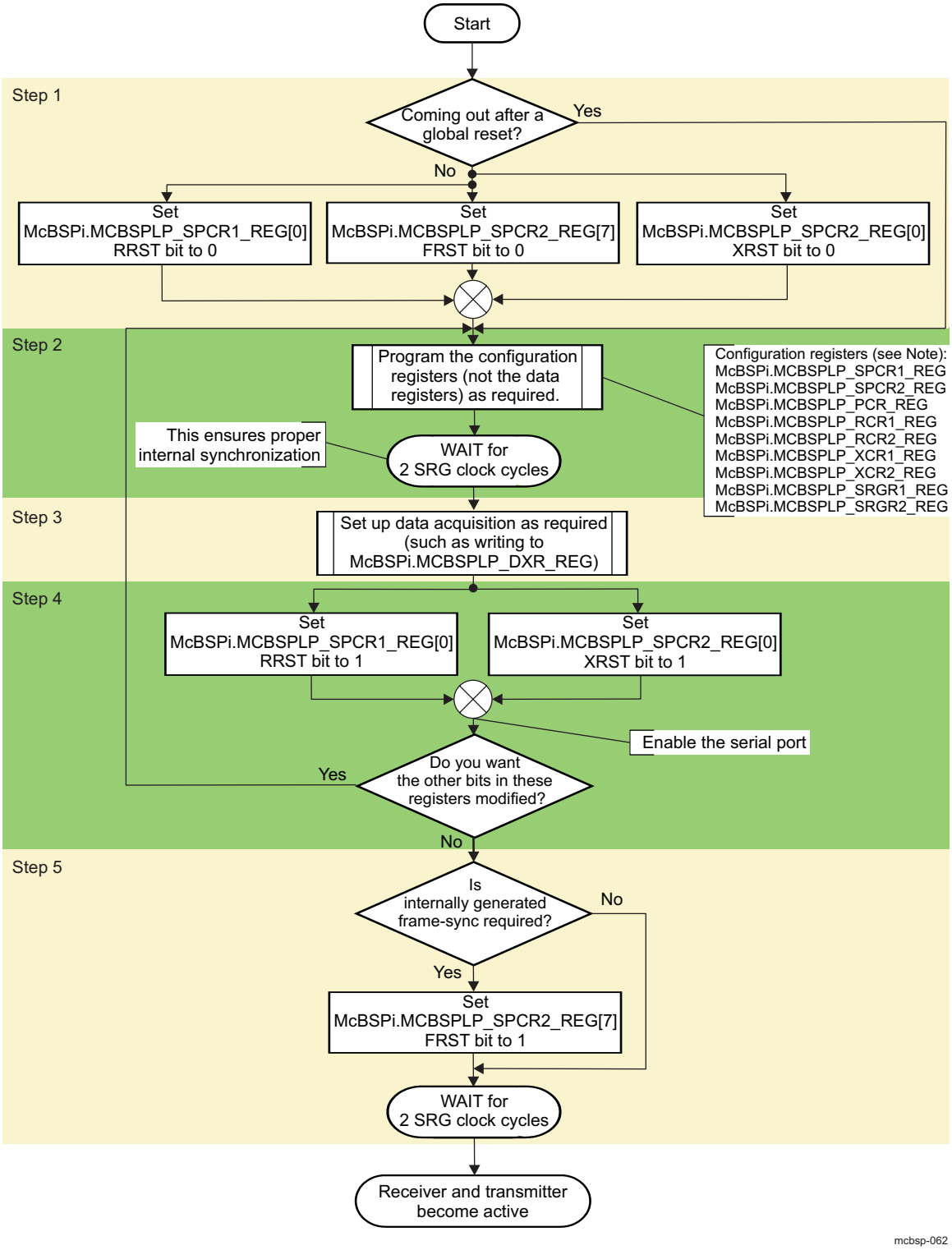
For all descriptions in this section, the McBSPi.MCBSPLP_XCCR_REG[11] XFULL_CYCLE bit and the McBSPi.MCBSPLP_RCCR_REG[11] RFULL_CYCLE bit are their reset value (XFULL_CYCLE bit=0 and RFULL_CYCLE bit=1).

21.5.1.1 McBSP Initialization Procedure

This procedure for reset/initialization can be applied in general when the Receiver or Transmitter has to be reset during its normal operation, and also when the Sample Rate Generator is not used for either operation.

Figure 21-57 shows the serial port initialization procedure for master mode.

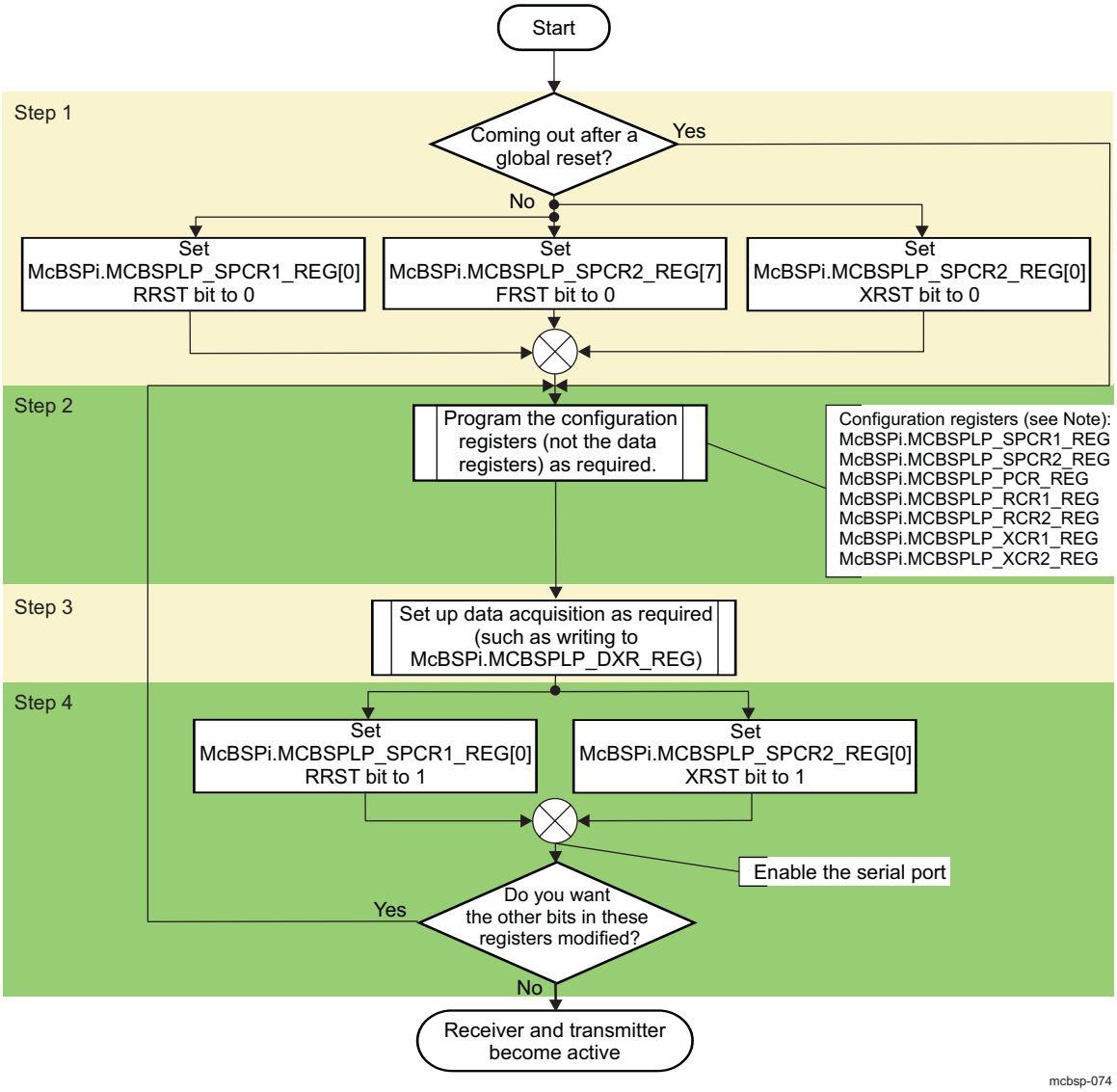
Figure 21-57. Flow Diagram of McBSP Initialization Procedure for Master Mode



Alternatively, on write (step 1 or 4), the transmitter and receiver can be placed in or taken out of reset by modifying the McBSPi.MCBSPLP_SPCR2_REG[0] XRST bit and the McBSPi.MCBSPLP_SPCR1_REG[0] RRST bit, respectively.

Figure 21-58 shows the flow diagram of the McBSP initialization procedure for slave mode.

Figure 21-58. Flow Diagram of McBSP Initialization Procedure for Slave Mode



Notes:

- The necessary duration of the active-low period of XRST or RRST is at least two CLKR/CLKX cycles.
- The appropriate bits in serial port configuration registers (McBSPi.MCBSPLP_SPCR1_REG, McBSPi.MCBSPLP_SPCR2_REG, McBSPi.MCBSPLP_PCR_REG, McBSPi.MCBSPLP_RCR1_REG, McBSPi.MCBSPLP_RCR2_REG, McBSPi.MCBSPLP_XCR1_REG, McBSPi.MCBSPLP_XCR2_REG, McBSPi.MCBSPLP_THRSH2_REG, McBSPi.MCBSPLP_XCCR_REG, McBSPi.MCBSPLP_SYSCONFIG_REG, McBSPi.MCBSPLP_SRGR1_REG, and McBSPi.MCBSPLP_SRGR2_REG) should only be modified when the affected portion of the serial port is in its reset state.
- In most cases, the data transmit register (McBSPi.MCBSPLP_DXR_REG) should be loaded by the MPU/IVA2 subsystem or the sDMA controller only when the transmitter is enabled (McBSPi.MCBSPLP_SPCR2_REG[0] XRST = 1). An exception to this rule is when these registers are used for loopback internal data.
- The bits of the channel control registers (McBSPi.MCBSPLP_MCR1_REG, McBSPi.MCBSPLP_MCR2_REG, McBSPi.MCBSPLP_RCER{A-H}_REG and McBSPi.MCBSPLP_XCER{A-H}_REG) can be modified at any time as long as they are not being used by the current reception/transmission in a multichannel selection mode.
- The SRG is reset by setting McBSPi.MCBSPLP_SPCR2_REG[6] GRST bit to 0.
- It is not necessary to wait if SRG is not used.
- Modification on-the-fly has no effect if a reset is not done first.

21.5.1.2 Reset and Initialization Procedure for the Sample Rate Generator

To reset and initialize the Sample Rate Generator:

- Place the McBSP Sample Rate Generator in RESET.

  During a Global RESET, the Sample Rate Generator, the Receiver, and the Transmitter reset bits (GRST, RRST, and XRST) are automatically forced to '0'. Otherwise, during normal operation, the Sample Rate Generator can be reset by making McBSPi.MCBSPLP_SPCR2_REG[6] GRST=0, provided that CLKG and/or FSG internal signal is not used by any portion of the McBSP module. Depending on the system needs, the software may also to reset the Receiver (McBSPi.MCBSPLP_SPCR1_REG[0] RRST=0) and reset the Transmitter (McBSPi.MCBSPLP_SPCR2_REG[0] XRST bit =0).
- Program the registers that affect the Sample Rate Generator.

  Program the Sample Rate Generator registers (McBSPi.MCBSPLP_SPCR1_REG and McBSPi.MCBSPLP_SPCR2_REG) as required for your application. Refer to Figure 21-59.

  If necessary, other control registers can be loaded with desired values provided the respective portion of the McBSP module (the Receiver or Transmitter) is in reset. Table 21-24 presents the McBSP configuration when one of the clock sources is selected, but others registers can be impacted in function of the user application.

Table 21-24. McBSP Configuration in Function of the SRG Clock Source Selected

SRG Clock Source Selected	Module Configuration	McBSPi.MCBSPLP_PCR_REG Configuration			
		CLKRM bit (1)	CLKXM bit	FSRM bit (1)	FSXM bit
McBSPi_ICLK or CLKS	Master Transmitter and Slave Receiver	0	1	0	1
	Master Receiver and Slave Transmitter	1	0	1	0
	Master Transmitter and Receiver	1	1	1	1
CLKR	Master Transmitter and Slave Receiver	0	1	0	1
CLKX	Master Receiver and Slave Transmitter	1	0	1	0

(1) This configuration is correct if McBSPi.MCBSPLP_XCCR_REG[5] DLB bit=0x0. When the DLB bit is set to 1, the CLKR clock (not the mcbspi_clkr pin) is driven by the CLKX clock, which is based on the CLKXM bit.

After the Sample Rate Generator registers are programmed, wait for 2 CLKSRG cycles. This ensures proper synchronization internally.

- Enable the Sample Rate Generator (take it out of reset).

  Set McBSPi.MCBSPLP_SPCR2_REG[6] GRST bit to1 to enable the Sample Rate Generator.

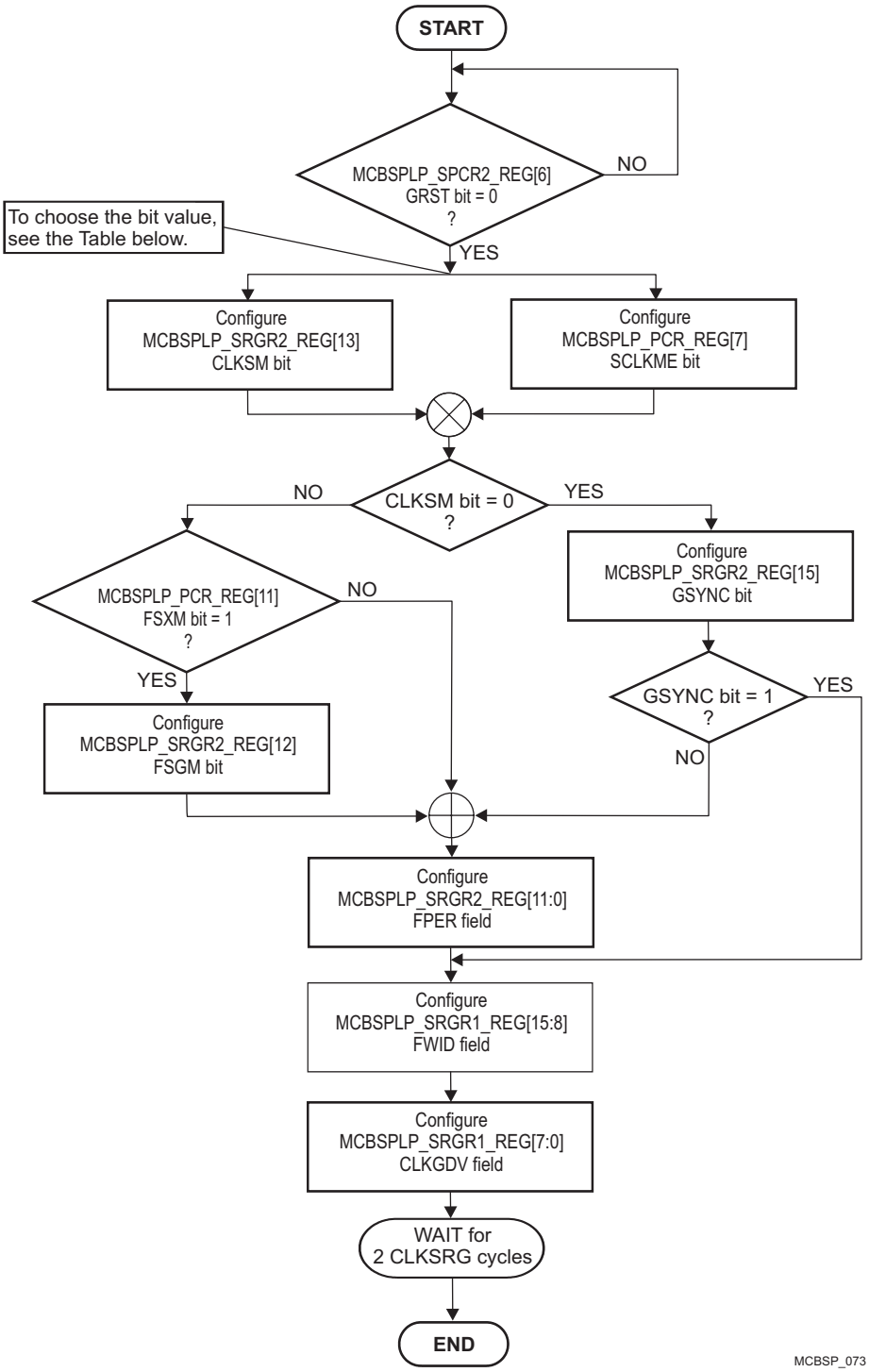
  After the Sample Rate Generator is enabled, wait for 2 CLKG cycles for the Sample Rate Generator logic to stabilize.

  On the next rising edge of CLKSRG, the CLKG signal transitions to '1' and starts clocking with a frequency equal to (input clock frequency)/(CLKGDV + 1).

- If necessary, enable the receiver and/or the transmitter.

  If necessary, remove the receiver and/or transmitter from reset by setting McBSPi.MCBSPLP_SPCR1_REG[0] RRST bit and/or McBSPi.MCBSPLP_SPCR2_REG[0] XRST =1.

Figure 21-59. Flow Diagram for the SRG Registers Programmation



The input clock is selected with the McBSPi.MCBSPLP_PCR_REG[7] SCLKME bit and the McBSPi.MCBSPLP_SRGR2_REG[13] CLKSM bit in one of the following configurations (see Table 21-25):

Table 21-25. Input Clock Selection for Sample Rate Generator

SCLKME bit	CLKSM bit	Input Clock for Sample Rate Generator
0	0	Signal on mcbsp_clks pin
0	1	McBSPi_ICLK clock
1	0	Signal on mcbsp_clkr pin
1	1	Signal on mcbsp_clkx pin

21.5.1.3 Data Transfer DMA Request Configuration

To configure the McBSP receive/transmit data DMA requests (McBSPi_DMA_RX and McBSPi_DMA_TX), perform the following procedure:

- Write the receive McBSPi.MCBSPLP_THRSH1_REG register with the required receive DMA request length (the length of the transfer is the same as the threshold value + 1). As long as the RB occupied locations level is above or equal to the THRSH1_REG value + 1, the DMA request will be asserted. After transferring the configured THRSH1_REG value + 1 number of words, the receive DMA request will be de-asserted and reasserted as soon as the conditions are met again.

  Note: In case of a number of transfers that exceed the number of the programmed DMA length the McBSP module will respond to the command, and will perform the transfer regardless of the receive buffer empty condition. When the receive buffer is empty a data transfer access will trigger a receive underflow interrupt, if enabled by McBSPi.MCBSPLP_IRQENABLE_REG[4] RUNDFLEN bit.

- Write the transmit McBSPi.MCBSPLP_THRSH2_REG register with the required transmit DMA request length (the length of the transfer is the same as the threshold value + 1). As long as the XB free locations level is above or equal to the THRSH2_REG value + 1, the DMA request will be asserted. After transferring the configured THRSH2_REG value + 1 number of words, the transmit DMA request will be de-asserted and reasserted as soon as the conditions are met again.

  Note: In case of a number of transfers that exceed the number of the programmed DMA length the McBSP module will respond to the command, and will perform the transfer regardless of the transmit buffer full condition. When the transmit buffer is full a data transfer access will trigger a transmit overflow interrupt, if enabled by McBSPi.MCBSPLP_IRQENABLE_REG[12] XOVFLEN bit.

21.5.1.4 Interrupt Configuration

The McBSP module offers two interrupt schemes:

- L4 compliant interrupt request scheme using a common receive/transmit interrupt request line
- The legacy interrupt compliant scheme using 3 interrupt lines: one for receive, one for transmit and the common interrupt line.

21.5.1.4.1 L4-Compliant Interrupt Line

The L4-compliant interrupt line can be configured by using the McBSPi.MCBSPLP_IRQENABLE_REG register. When the McBSPi.MCBSPLP_IRQSTATUS_REG bit is set and the corresponding McBSPi.MCBSPLP_IRQENABLE_REG bit is set to one, the interrupt line is asserted. Writing one to a bit in McBSPi.MCBSPLP_IRQSTATUS_REG register clears the bit.

There are several conditions, which can be configured to generate an interrupt as follows:

1. Transmit buffer empty at end of frame (McBSPi.MCBSPLP_IRQSTATUS_REG[14] XEMPTYEOF bit is set to one when a complete frame was transmitted and the transmit buffer is empty .
2. Transmit buffer overflow (McBSPi.MCBSPLP_IRQSTATUS_REG[12] XOVFLSTAT bit is set to one when transmit buffer overflow; the data written while overflow condition is discarded).
3. Transmit buffer underflow (McBSPi.MCBSPLP_IRQSTATUS_REG[11] XUNDFLSTAT bit is set to one when the transmit data buffer is empty, new data needs to be transmitted).

4. Transmit buffer threshold reached (McBSPi.MCBSPLP_IRQSTATUS_REG[10] XRDY bit is set to one when the transmit buffer free locations are equal or above the [THRSH2_REG + 1] value).
5. Transmit end of frame (McBSPi.MCBSPLP_IRQSTATUS_REG[9] XEOF is set to one when a complete frame was transmitted).
6. Transmit frame synchronization (McBSPi.MCBSPLP_IRQSTATUS_REG[8] XFSX bit is set to one when a new transmit frame synchronization is asserted).
7. Transmit frame synchronization Error (McBSPi.MCBSPLP_IRQSTATUS_REG[7] XSYNCERR is set to one when a transmit frame synchronization error is detected).
8. Receive buffer overflow (McBSPi.MCBSPLP_IRQSTATUS_REG[5] ROVFLSTAT bit is set to one when receive buffer overflow; the data which is written while overflow condition is discarded).
9. Receive buffer underflow (McBSPi.MCBSPLP_IRQSTATUS_REG[4] RUNDFLSTAT bit is set to one when read operation is performed to the receive data register while receive buffer is empty; data read while underflow condition is undefined).
10. Receive buffer threshold Reached (McBSPi.MCBSPLP_IRQSTATUS_REG[3] RRDY bit is set to one when the receive buffer occupied locations are equal or above the [THRSH1_REG + 1] value).
11. Receive end of frame (McBSPi.MCBSPLP_IRQSTATUS_REG[2] REOF is set to one when a complete frame was received).
12. Receive frame synchronization (McBSPi.MCBSPLP_IRQSTATUS_REG[1] RFSR bit is set to one when a new receive frame synchronization is asserted).
13. Receive frame synchronization error (McBSPi.MCBSPLP_IRQSTATUS_REG[0] RSYNCERR is set to one when a receive frame synchronization error is detected).

21.5.1.4.2 Legacy Interrupt Line

McBSPi_IRQ_TX and McBSPi_IRQ_RX are legacy interrupts. Not to be used for new development. McBSPi_IRQ (common interrupt line) should be preferred.

21.5.1.4.2.1 Set the receive interrupt line (legacy only)

The McBSPi.MCBSPLP_SPCR1_REG[5:4] RINTM bit field determines which event generates a receive interrupt request, McBSPi_IRQ_RX, to the MPU/IVA2.2 subsystem.

The receive interrupt informs the MPU/IVA2.2 subsystem of changes to the serial port status. Four options exist for configuring this interrupt.

- RINTM=0b00: The receive interrupt generated when the McBSPi.MCBSPLP_SPCR1_REG[1] RRDY bit changes from 0 to 1. Interrupt on every serial word by tracking the McBSPi.MCBSPLP_SPCR1_REG[1] RRDY bit. Regardless of the value of RINTM, RRDY bit can be read to detect the RRDY=1 condition.
- RINTM = 0b01: The receive interrupt generated by an end-of-frame condition in the receive multichannel selection mode. In any other serial transfer case, this setting is not applicable and, therefore, no interrupts are generated.
- RINTM = 0b10: The receive interrupt generated by a new receive frame-synchronization pulse. Interrupt on detection of receive frame-synchronization pulses. This generates an interrupt even when the receiver is in its reset state. This is done by synchronizing the incoming frame-synchronization pulse to the McBSPi_ICLK clock and sending it to the MPU/IVA2.2 subsystem via the receive interrupt .
- RINTM = 0b11: The receive interrupt generated when McBSPi.MCBSPLP_SPCR1_REG[3] RSYNCERR is set. Interrupt on frame-synchronization error. Regardless of the value of RINTM, RSYNCERR can be read to detect this condition. For information on using RSYNCERR, see Section 21.4.4.3.

The McBSP module also provides a common interrupt line McBSPi_IRQ, which can be used by setting the McBSPi.MCBSPLP_IRQENABLE register. All the above settings have equivalent enable bits in the McBSPi.MCBSPLP_IRQENABLE register to enable the common interrupt line:

- RRDYEN is equivalent with RINTM = 0 setting
- REOFEN is equivalent with RINTM = 0b01setting (the interrupt is generated by an end-of-frame condition regardless of the multichannel selection mode)
- RFSREN is equivalent with RINTM = 0b10 setting

- RSYNCERREN is equivalent with RINTM = 0b11 setting

This interrupt line has its own status register, McBSPi.MCBSPLP_IRQSTATUS_REG.

21.5.1.4.2.2 Set the transmit interrupt line (legacy only)

The McBSPi.MCBSPLP_SPCR2_REG[5:4] XINTM bit field determines which event generates a transmit interrupt request (McBSPi_IRQ_TX) to the MPU/IVA2.2 subsystem.

The transmit interrupt informs the MPU/IVA2.2 subsystem of changes to the serial port status. Four options exist for configuring this interrupt.

- XINTM = 0b00: The transmit interrupt generated when the McBSPi.MCBSPLP_SPCR2_REG[1] XRDY bit changes from 0 to 1. Interrupt on every serial word by tracking the XRDY bit. Regardless of the value of XINTM, XRDY bit can be read to detect the XRDY=1 condition.
- XINTM = 0b01: The transmit interrupt generated by an end-of-frame condition in the transmit multichannel selection mode. In any other serial transfer case, this setting is not applicable and, therefore, no interrupts are generated.
- XINTM = 0b10: The transmit interrupt generated by a new transmit frame-synchronization pulse. Interrupt on detection of transmit frame-synchronization pulses. This generates an interrupt even when the transmitter is in its reset state. This is done by synchronizing the incoming frame-synchronization pulse to the McBSPi_ICLK clock and sending it to the MPU/IVA2.2 subsystem via transmit interrupt.
- XINTM = 0b11: The transmit interrupt generated when McBSPi.MCBSPLP_SPCR2_REG[3] XSYNCERR is set. Interrupt on frame-synchronization error. Regardless of the value of XINTM, XSYNCERR bit can be read to detect this condition. For information on using XSYNCERR bit, see Section 21.4.4.6.

The McBSP module provides also a common interrupt line McBSPi_IRQ, which can be used by setting the McBSPi.MCBSPLP_IRQENABLE register. All the above settings have equivalent enable bits in the McBSPi.MCBSPLP_IRQENABLE register to enable the common interrupt line:

- XDYEN is equivalent with XINTM = 0b00 setting
- XEOFEN is equivalent with XINTM = 0b01setting (the interrupt is generated by an end-of-frame condition regardless of the multichannel selection mode)
- XFSXEN is equivalent with XINTM = 0b10 setting
- XSYNCERREN is equivalent with XINTM = 0b11 setting

This interrupt line has its own status register, McBSPi.MCBSPLP_IRQSTATUS_REG.

21.5.1.5 Receiver Configuration

To configure the McBSP receiver, perform the following procedure:

1. Place the McBSP receiver in reset .
2. Program the McBSP registers for the desired receiver operation.
3. Take the receiver out of reset.

These 3 steps are detailed in the following subsections.

21.5.1.5.1 Resetting (Step 1) and Enabling (Step 3) the Receiver

The first step of the receiver configuration procedure is to reset the receiver, and the last step is to enable the receiver (to take it out of reset).

The serial port can be reset in the following 2 ways:

1. A global reset places the receiver, transmitter, and SRG in reset. When the device reset is removed, McBSPi.MCBSPLP_SPCR2_REG[6] GRST, McBSPi.MCBSPLP_SPCR2_REG[7] FRST, McBSPi.MCBSPLP_SPCR1_REG[0] RRST and McBSPi.MCBSPLP_SPCR2_REG[0] XRST bits = 0, which keeps the entire serial port in the reset state.

2. The serial port receiver can be reset directly using the McBSPi.MCBSPLP_SPCR1_REG[0] RRST bit. If the SRG needs to be used, SRG must be reset directly using the McBSPi.MCBSPLP_SPCR2_REG[6] GRST bit. Similar operations with frame synchronization generator also require using the McBSPi.MCBSPLP_SPCR2_REG[7] FRST bit when the frame-sync signal must be generated.

To enable the receiver, the preceding bits, cleared to 0, must be set to 1.

21.5.1.5.2 Programming the McBSP Registers for the Desired Receiver Configuration (Step 2)

Figure 21-60 and Figure 21-61 lists important tasks to be performed when the software is configuring the McBSP receiver. Each task corresponds to one or more McBSP register bit fields.

Figure 21-60. Important Tasks to Configure the McBSP Receiver (Part 1)

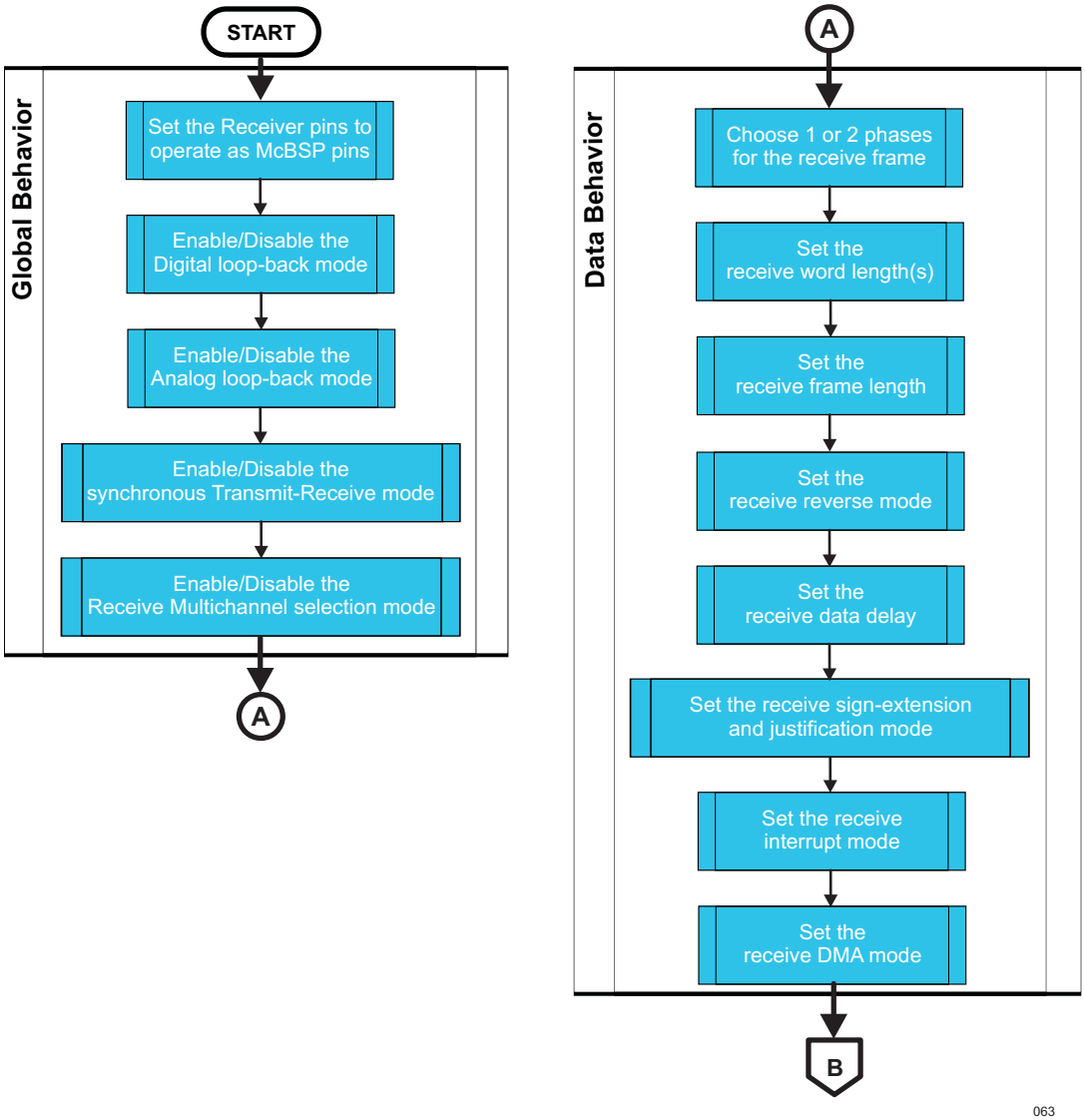
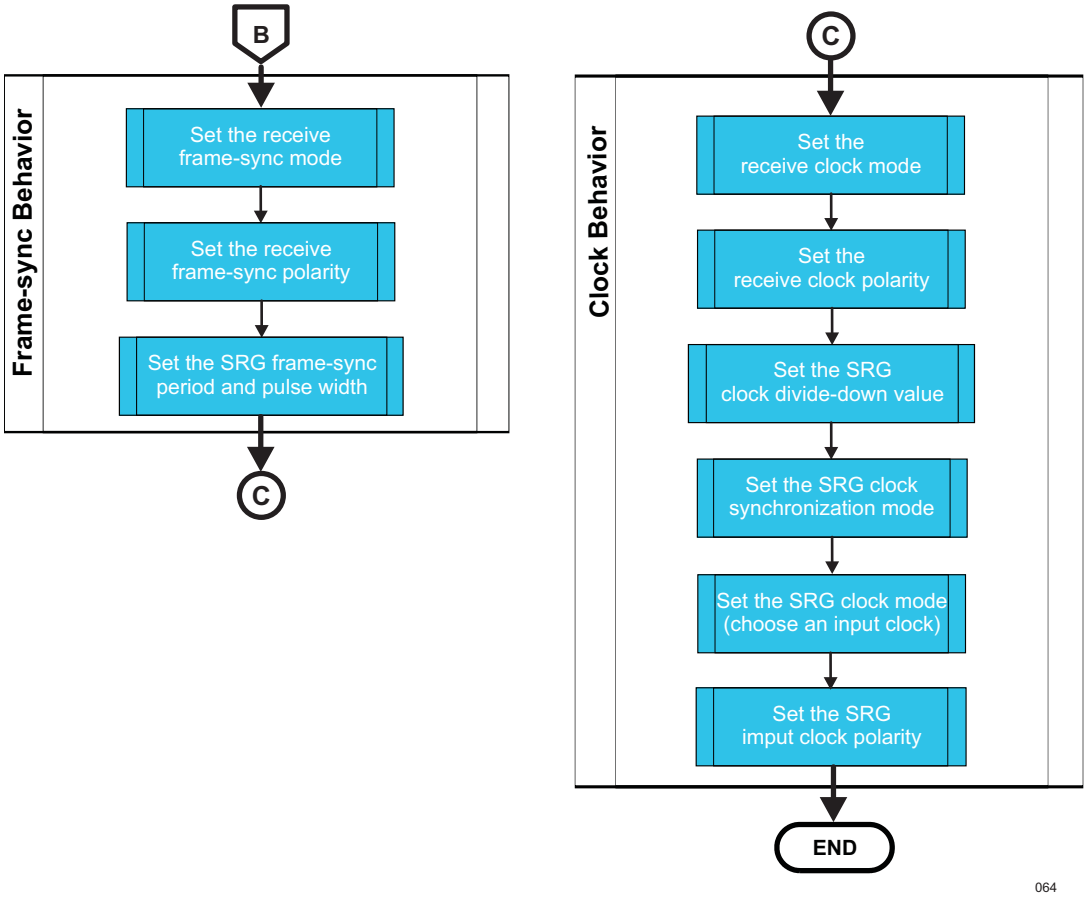


Figure 21-61. Important Tasks to Configure the McBSP Receiver (Part 2)



21.5.1.5.2.1 Global Behavior

21.5.1.5.2.1.1 Set the Receiver Pins to Operate as McBSP Pins

The McBSPi.MCBSPLP_PCR_REG[12] RIOEN bit determines whether the receiver pins are McBSP pins (RIOEN bit=0) or general–purpose I/O pins (RIOEN bit=1).

Please refer to Section 21.5.1.7 which describes how to use McBSP pins as GPIO pins.

21.5.1.5.2.1.2 Enable/Disable the Digital Loop Back Mode

The McBSPi.MCBSPLP_XCCR_REG[5] DLB bit determines whether the digital loopback mode is on or off.

In the digital loopback mode, the receive signals are connected internally through multiplexers to the corresponding transmit signals:

- DR signal is connected on DX signal to receive the transmitted data
- FSR is connected to FRX output signal
- CLKR is connected to the CLKX output signal

This mode allows testing of serial port; the McBSP module receives the data it transmits. This loopback mode is not done through pads, all output signals being disabled.

Note: That in digital loopback mode the SRG and the frame synchronization generator need to be enabled to generate the CLKX and FSX signals.

21.5.1.5.2.1.3 Enable/Disable the Analog Loopback Mode

The McBSPi.MCBSPLP_SPCR1_REG[15] ALB bit determines whether the analog loopback mode is on or off.

In the analog loopback mode, the receive signals are connected internally through multiplexers to the corresponding transmit loop back signals:

- DR is connected to transmit loop back data on DX input pin
- FSR is connected to FRX input pin
- CLKR is connected to the CLKX input pin

This Analog Loopback mode is also done to test the Input/Output buffers, through pads.

21.5.1.5.2.1.4 Enable/disable the synchronous transmit-receive mode (McBSP1 only)

The control registers of the System Control Module is used to configure the synchronous transmit-receive mode.

The MCBSP1_CLKR bit of the CONTROL_DEVCONF0[3] register is used to select the McBSP1 module CLKR signal source:

- When set to ‘0’, the CLKR source is from the CLKR input signal
- When set to ‘1’, the CLKR source is from the CLKX input signal

The MCBSP1_FSR bit of the CONTROL_DEVCONF0[4] register is used to select the McBSP1 module FSR signal source:

- When set to ‘0’, the FSR source is from the FSR input signal
- When set to ‘1’, the FSR source is from the FSX input signal

21.5.1.5.2.1.5 Enable/Disable the Receive Multichannel Selection Mode

The McBSPi.MCBSPLP_MCR1_REG[0] RMCM bit determines whether the receive multichannel selection mode is on or off.

For further details, see Section 21.4.6.5.

21.5.1.5.2.2 Data Behavior

21.5.1.5.2.2.1 Choose 1 or 2 Phases for the Receive Frame

The McBSPi.MCBSPLP_RCR2_REG[15] RPHASE bit determines whether the receive data frame has one (Single phase) or two phases (Dual phase). When dual-phase is selected the number of words per phase must be set to one.

21.5.1.5.2.2.2 Set the Receive Word Length(s)

The McBSPi.MCBSPLP_RCR1_REG[7:5] RWDLEN1 and McBSPi.MCBSPLP_RCR2_REG[7:5] RWDLEN2 bit fields determine how many bits are in each serial word in phase 1 and in phase 2, respectively, of the receive data frame.

If a single-phase frame is selected, RWDLEN1 selects the length for every serial word received in the frame.

If a dual–phase frame is selected, RWDLEN1 and RWDLEN2 must be set to select the both length. These both bits can have values different.

21.5.1.5.2.2.3 Set the Receive Frame Length

The receive frame length is the number of serial words in the receive frame.

The McBSPi.MCBSPLP_RCR1_REG[14:8] RFRLEN1 and McBSPi.MCBSPLP_RCR2_REG[14:8] RFRLEN2 bit fields determine how many serial words are in phase 1 and in phase 2, respectively, of the receive data frame.

If a dual-phase frame is selected (RPHASE=1), the frame length must be two words(one word for phase 1 plus the one word for phase 2). Others values must not be used.

The 7–bit RFRLEN1 field allows up to 128 words per phase when single-phase frame. See Table 21-26 below for a summary of how to calculate the frame length. This length corresponds to the number of words or logical time slots or channels per frame-synchronization pulse.

Program the RFRLEN fields with [W - 1], where W represents the number of words per phase. For example, to get a phase length of 128 words in phase 1, load 127 words into RFRLEN1.

Table 21-26. How to Calculate the Length of the Receive Frame

RPHASE	RFRLEN1	RFRLEN2	Frame Length
0	0 ≤ RFRLEN1 ≤ 127	Don't care	(RFRLEN1value +1) words
1	RFRLEN1 = 0	RFRLEN2 = 0	2 words

21.5.1.5.2.2.4 Set the Receive Reverse Mode

The McBSPi.MCBSPLP_RCR2_REG[4:3] RREVERSE bit field determines whether reverse (LSB first) data transfer option is chosen for McBSP reception.

For further information about reverse mode, see Section 21.4.2.2.

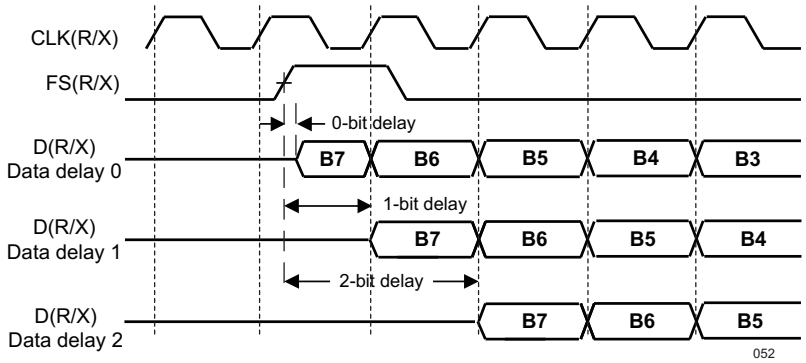
21.5.1.5.2.2.5 Set the Receive Data Delay

The McBSPi.MCBSPLP_RCR2_REG[1:0] RDATDLY bit field determines the length of the data delay for the receive frame.

The start of a frame is defined by the first clock cycle in which frame synchronization is active. The beginning of actual data reception or transmission with respect to the start of the frame can be delayed if required. This delay is called data delay.

McBSPi.MCBSPLP_RCR2_REG[1:0] RDATDLY specifies the data delay for reception. The range of programmable data delay is zero to two bit–clocks (RDATDLY=0b00–0b10), as shown in Figure 21-62 below. In this figure, the data transferred is an 8–bit value with bits labeled B7, B6, B5, and so on. Typically a 1-bit delay is selected, because data often follows a 1–cycle active frame-synchronization pulse.

Figure 21-62. Range of Programmable Data Delay



21.5.1.5.2.2.5.1 0-Bit Data Delay

Normally, a frame-synchronization pulse is detected or sampled with respect to an edge of internal serial clock CLK(R/X). Thus, on the following cycle or later (depending on the data delay value), data may be received or transmitted. However, in the case of 0–bit data delay, the data must be ready for reception and/or transmission on the same serial clock cycle.

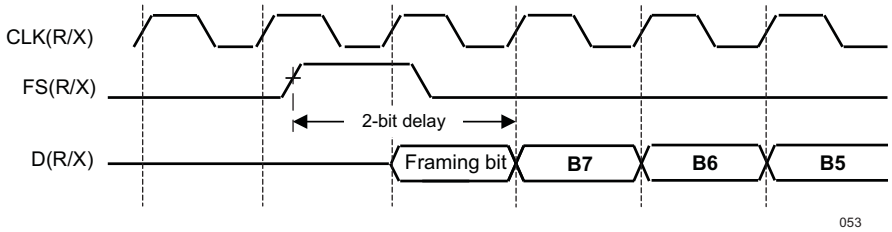
For reception, this problem is solved because receive data is sampled on the first falling edge of CLKR

where an active–high internal FSR is detected. However, data transmission must begin on the rising edge of the internal CLKX clock that generated the frame synchronization. Therefore, the first data bit is assumed to be present in XSR, and thus on mcbsp_dx. The transmitter then asynchronously detects the frame-synchronization signal (FSX) going active high and immediately starts driving the first bit to be transmitted on the mcbsp_dx pin.

21.5.1.5.2.2.5.2 2-Bit Data Delay

A data delay of two bit periods allows the serial port to interface to different types of T1 framing devices where the data stream is preceded by a framing bit. During reception of such a stream with data delay of two bits (framing bit appears after a 1–bit delay and data appears after a 2–bit delay), the serial port essentially discards the framing bit from the data stream, as shown in Figure 21-63. In this figure, the data transferred is an 8–bit value with bits labeled B7, B6, B5, and so on.

Figure 21-63. 2-Bit Data Delay Used to Skip a Framing Bit



21.5.1.5.2.2.6 Set the Receive Sign-Extension and Justification Mode

The McBSPi.MCBSPLP_SPCR1_REG[14:13] RJUST bit field determines whether data received by the McBSP module is sign–extended or not and how it is justified.

RJUST bit selects whether data in RB is right- or left-justified (with respect to the MSB) in McBSPi.MCBSPLP_DRR_REG register and whether unused bits in McBSPi.MCBSPLP_DRR_REG are filled with zeroes or with sign bits.

Table 21-27 and Table 21-28 show the effects of various RJUST values; the effect on an example 12–bit receive–data value 0xABC, and the effect on an example 20–bit receive–data value 0xABCDE, respectively.

Table 21-27. Example: Use of RJUST Bit Field With 12-bit Data Value 0xABC

RJUST	Justification	Extension	Value in DRR_REG
0b00	Right	Zero fill MSBs	0x0000 0ABC
0b01	Right	Sign extend data into MSBs	0xFFFF FABC
0b10	Left	Zero fill LSBs	0xABC0 0000
0b11	Reserved	Reserved	Reserved

Table 21-28. Example: Use of RJUST Bit Field With 20-bit Data Value 0xABCDE

RJUST	Justification	Extension	Value in DRR_REG
0b00	Right	Zero fill MSBs	0x000A BCDE
0b01	Right	Sign extend data into MSBs	0xFFFA BCDE
0b10	Left	Zero fill LSBs	0xABCD E000
0b11	Reserved	Reserved	Reserved

21.5.1.5.2.2.7 Set the Receive Interrupt Mode

Please refer to Section 21.5.1.4.2.1.

21.5.1.5.2.2.8 Set the Receive DMA Mode

The McBSP receive-data DMA requests (McBSPi_DMA_RX) are active after the McBSPi.MCBSPLP_SPCR1_REG [0] RRST bit is released. After reset, the default DMA threshold (and length) is 1.

The receive DMA requests can be disabled by setting the McBSPi.MCBSPLP_RCCR_REG[3] RDMAEN bit to 0. When disabling the DMA, the DMA request line is deasserted even if a DMA transfer is pending, and the DMA state-machine is not reset.

To configure the McBSP receive data DMA requests, perform the following:

- Write the receive McBSPi.MCBSPLP_THRSH1_REG register with the required receive DMA request length (the length of the transfer is the same as the threshold value + 1).
- As long as the occupied locations level in the RB is above or equal to the THRSH1_REG value + 1, the DMA request is asserted.
- After transferring the configured (THRSH1_REG value + 1) number of words, the receive DMA request is deasserted, and then reasserted as soon as the conditions are met again.

21.5.1.5.2.3 Frame-Sync Behavior

21.5.1.5.2.3.1 Set the Receive Frame-Sync Mode

McBSPi.MCBSPLP_PCR_REG[10] FSRM bit, McBSPi.MCBSPLP_SRGR2_REG[15] GSYNC bit, McBSPi.MCBSPLP_SPCR1_REG[15] ALB bit and McBSPi.MCBSPLP_XCCR_REG[5] DLB bit field are used to determine the source for receive frame synchronization and the function of the mcbsp_fsr pin.

Table 21-29 below shows how you can select various sources to provide the receive frame-synchronization signal and the effect on the mcbsp_fsr pin. The polarity of the signal on the mcbsp_fsr pin is determined by the McBSPi.MCBSPLP_PCR_REG[2] FSRP bit.

Table 21-29. FSRM and GSYNC Effects on Frame-Sync Signal and mcbsp_fsr Pin

FSRM	GSYNC	Source of Receive Frame Synchronization	MCBSPLP.FSR Pin Status
0	0 or 1	An external frame synchronization signal enters the McBSP module through the mcbsp_fsr pin. The signal is then inverted as determined by FSRP bit before being used as internal FSR.	Input
1	0	Internal FSR is driven by the SRG frame-synchronization signal (FSG).	Output. FSG is inverted as determined by FSRP bit before being driven out on the mcbsp_fsr pin.
1	1	Internal FSR is driven by the SRG frame-synchronization signal (FSG).	Input. The external frame-synchronization input on the mcbsp_fsr pin is used to synchronize CLKG and generate FSG pulses.

In digital loop-back mode (DLB=1), the transmit frame-synchronization signal is used as the receive frame-synchronization signal.

Also in the analog loop back mode (ALB=1), the internal receive clock signal (CLKR), and the internal receive frame-synchronization signal (FSR) are internally connected to their transmit counterparts, CLKX and FSX.

For more details on clock and frame-sync configuration, see Section 21.4.3.

21.5.1.5.2.3.2 Set the Receive Frame-Sync Polarity

The McBSPi.MCBSPLP_PCR_REG[2] FSRP bit determines whether frame-synchronization pulses are active high or active low on the mcbsp_fsr pin.

Receive frame-synchronization pulses can be generated internally by the SRG or driven by an external source. The source of frame synchronization is selected by programming the McBSPi.MCBSPLP_PCR_REG[10] FSRM bit. FSR is also affected by the McBSPi.MCBSPLP_SRGR2_REG[15] GSYNC bit. For information about the effects of FSRM and GSYNC, see Section 21.5.1.5.2.3.1, *Set the Receive Frame-Sync Mode*.

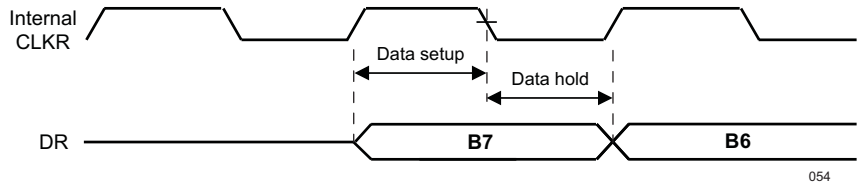
When FSR and FSX are inputs (FSXM = FSRM = 0, external frame-synchronization pulses), the McBSP module detects them on the internal falling edge of clock, internal CLKR, and internal CLKX, respectively. The receive data arriving at the mcbspi_dr pin is also sampled on the falling edge of internal CLKR. These internal clock signals are either derived from an external source via CLK(R/X) pins or driven by the SRG clock (CLKG) internal to the McBSP module.

When FSR and FSX are outputs, implying that they are driven by the SRG, they are generated (transition to their active state) on the rising edge of the internal clock, CLK(R/X). Similarly, data on the mcbsp_dx pin is output on the rising edge of internal CLKX.

FSRP, FSXP, CLKRP, and CLKXP bit fields in the pin control register (McBSPi.MCBSPLP_PCR_REG) configure the polarities of the FSR, FSX, CLKR, and CLKX signals, respectively. All frame-synchronization signals (internal FSR, internal FSX) that are internal to the serial port are active high. If the serial port is configured for external frame synchronization (FSR/FSX are inputs to McBSP), and FSRP = FSXP = 1, the external active-low frame-synchronization signals are inverted before being sent to the receiver (internal FSR) and transmitter (internal FSX). Similarly, if internal synchronization (FSR/FSX are output pins and GSYNC = 0) is selected, the internal active-high frame-synchronization signals are inverted, if the polarity bit FS(R/X)P = 1, before being sent to the FS(R/X) pin.

Figure 21-64 shows how data clocked by an external serial device using a rising edge can be sampled by the McBSP receiver on the falling edge of the same clock.

Figure 21-64. Data Externally Clocked on a Rising Edge and Sampled on a Falling Edge



21.5.1.5.2.3.3 Set the SRG Frame-Sync Period and Pulse Width

The SRG can produce a clock signal, CLKG, and FSG. If the SRG is supplying receive or transmit frame synchronization, you must program the bit fields FPER and FWID.

McBSPi.MCBSPLP_SRGR2_REG[11:0] FPER bit field is used to set the SRG frame-sync period and McBSPi.MCBSPLP_SRGR1_REG[15:8] FWID bit field is used to set the SRG pulse width.

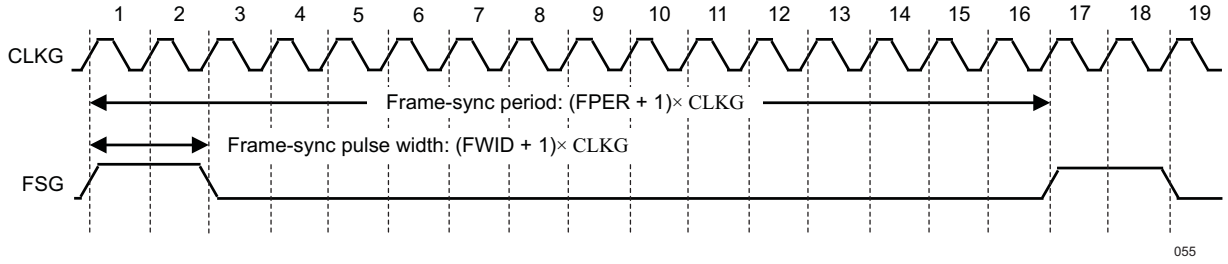
On FSG, the period from the start of a frame-synchronization pulse to the start of the next pulse is (FPER+1) CLKG cycles. The 12 bits of FPER allow a frame-synchronization period of 1 to 4096 CLKG cycles, which allows up to 4096 data bits per frame. When GSYNC=1, FPER is a don't care value.

Each pulse on FSG has a width of (FWID+1) CLKG cycles. The eight bits of FWID allow a pulse width of 1 to 256 CLKG cycles. It is recommended that FWID be programmed to a value less than the programmed word length.

The values in FPER and FWID are loaded into separate down–counters. The 12–bit FPER counter counts down the generated clock cycles from the programmed value (4095 maximum) to 0. The 8–bit FWID counter counts down from the programmed value (255 maximum) to 0.

Figure 21-65 shows a frame-synchronization period of 16 CLKG periods (FPER=15 or 00001111b) and a frame-synchronization pulse with an active width of 2 CLKG periods (FWID=1).

Figure 21-65. Frame of Period 16 CLKG Periods and Active Width of 2 CLKG Periods



When the SRG comes out of reset, FSG is in its inactive state. Then, when GRST=1 and FSGM=1, a frame-synchronization pulse is generated. The frame width value (FWID+1) is counted down on every CLKG cycle until it reaches 0, at which time FSG goes low. At the same time, the frame period value (FPER+1) is also counting down. When this value reaches 0, FSG goes high, indicating a new frame.

21.5.1.5.2.4 Clock Behavior

21.5.1.5.2.4.1 Set the receive clock mode

McBSPi.MCBSPLP_PCR_REG[8] CLKRM bit, McBSPi.MCBSPLP_SPCR1_REG[15] ALB bit and McBSPi.MCBSPLP_XCCR_REG[5] DLB bit are used to set the receive clock mode.

Table 21-30 shows how to select various sources to provide the receive clock signal and affect the mcbsp_clkr pin. The McBSPi.MCBSPLP_PCR_REG[0] CLKRP bit determines the polarity of the signal on the mcbsp_clkr pin.

Table 21-30. CLKRM Effect on Receive Clock Signal and mcbsp_clkr Pin

CLKRM	Source of Receive Clock	mcbsp_clkr Pin Status
0	The mcbsp_clkr pin is an input driven by an external clock. The external clock signal is inverted as determined by CLKRP bit before being used.	Input
1	The SRG clock (CLKG) drives internal CLKR.	Output. CLKG, inverted as determined by CLKRP, is driven out on the mcbsp_clkr pin.

In the digital loop-back mode (DLB=1) or analog loop-back mode (ALB = 1), the transmit clock signal is used as the receive clock signal. For more details on clock configuration, see Section 21.4.3.1.

21.5.1.5.2.4.2 Set the Receive Clock Polarity

McBSPi.MCBSPLP_PCR_REG[0] CLKRP bit is used to set the receive clock polarity.

The receive clock polarity bit, CLKRP, sets the edge used to sample received data. The receive data is always sampled on the falling edge of internal CLKR. Therefore, if CLKRP=1 and external clocking is selected (CLKRM=0 and CLKR is an input pin), the external rising–edge triggered input clock on CLKR is inverted to a falling–edge triggered clock before being sent to the receiver. If CLKRP=1 and internal clocking is selected (CLKRM=1), the internal falling–edge triggered clock is inverted to a rising–edge triggered clock before being sent out on the mcbsp_clkr pin.

Note: CLKRP=CLKXP in a system where the same clock (internal or external) is used to clock the receiver and transmitter. The receiver uses the opposite edge as the transmitter to ensure valid setup and hold of data around this edge.

21.5.1.5.2.4.3 Set the SRG Clock Divide-Down Value

McBSPi.MCBSPLP_SRGR1_REG[7:0] CLKGDV bit field is used to set the SRG clock divide-down value.

The first divider stage generates the serial data bit clock from the input clock. This divider stage utilizes a counter, preloaded by CLKGDV, that contains the divide ratio value.

The output of the first divider stage is the data bit clock, which is output as CLKG and which serves as the input for the second and third stages of the divider.

CLKG has a frequency equal to 1/(CLKGDV+1) of SRG input clock. Thus, the sample generator input clock frequency is divided by a value between 1 and 256. The CLKG duty cycle is 50%.

21.5.1.5.2.4.4 Set the SRG Clock Synchronization Mode

McBSPi.MCBSPLP_SRGR2_REG[15] GSYNC bit is used to set the SRG clock synchronization mode.

For more information about the clock synchronization feature, see Section 21.4.3.

21.5.1.5.2.4.5 Set the SRG Clock Mode (Choose an Input Clock)

McBSPi.MCBSPLP_PCR_REG[7] SCLKME bit and McBSPi.MCBSPLP_SRGR2_REG[13] CLKSM bit are used to set the SRG clock mode.

The SRG can produce a clock signal (CLKG) for use by the receiver, the transmitter, or both, but CLKG is derived from an input clock.

For further details about the clock synchronization feature, see Section 21.4.3.

21.5.1.5.2.4.6 Set the SRG Input Clock Polarity

McBSPi.MCBSPLP_SRGR2_REG[14] CLKSP bit, McBSPi.MCBSPLP_PCR_REG[1] CLKXP bit and McBSPi.MCBSPLP_PCR_REG[0] CLKRP bit are used to set the SRG input clock polarity.

The SRG can produce a clock signal (CLKG) and a frame-synchronization signal (FSG) for use by the receiver, the transmitter, or both. To produce CLKG and FSG, the SRG must be driven by an input clock signal derived from the McBSP_FCLK clock or from an external clock on the mcbsp_clks, mcbsp_clkx, or mcbsp_clkr pin. If you use a pin, choose a polarity for that pin by using the appropriate polarity bit (CLKSP for the mcbsp_clks pin, CLKXP for the mcbsp_clkx pin, CLKRP for the mcbsp_clkr pin). The polarity determines whether the rising or falling edge of the input clock generates transitions on CLKG and FSG.

21.5.1.6 Transmitter Configuration

To configure the McBSP transmitter, perform the following procedure:

1. Place the McBSP transmitter in reset
2. Program the McBSP registers for the desired transmitter operation
3. Take the transmitter out of reset

These 3 steps are described in more details in the sub-sections below.

21.5.1.6.1 Resetting (Step 1) and Enabling (Step 3) the Transmitter

The first step of the transmitter configuration procedure is to reset the transmitter, and the last step is to enable the transmitter (to take it out of reset).

The serial port can be reset in the following two ways:

1. A global reset places the receiver, transmitter, and SRG in reset. When the device reset is removed, McBSPi.MCBSPLP_SPCR2_REG[6] GRST, McBSPi.MCBSPLP_SPCR2_REG[7] FRST, McBSPi.MCBSPLP_SPCR1_REG[0] RRST and McBSPi.MCBSPLP_SPCR2_REG[0] XRST bits = 0, which keeps the entire serial port in the reset state.
2. The serial port receiver can be reset directly using the McBSPi.MCBSPLP_SPCR2_REG[0] XRST bit. If the SRG needs to be used, SRG must be reset directly using the McBSPi.MCBSPLP_SPCR2_REG[6] GRST bit. Similar operation with the Frame Synchronization Generator also requires using the McBSPi.MCBSPLP_SPCR2_REG[7] FRST bit when the frame-sync signal must be generated.

To enable the transmitter, the preceding bits, cleared to 0, must be set to 1.

21.5.1.6.2 Programming the McBSP Registers for the Desired Transmitter Operation (Step 2)

Figure 21-66 and Figure 21-67 list important tasks to be performed when configuring the McBSP transmitter. Each task corresponds to one or more McBSP register bit fields.

Figure 21-66. Important Tasks to Configure the McBSP Transmitter (Part 1)

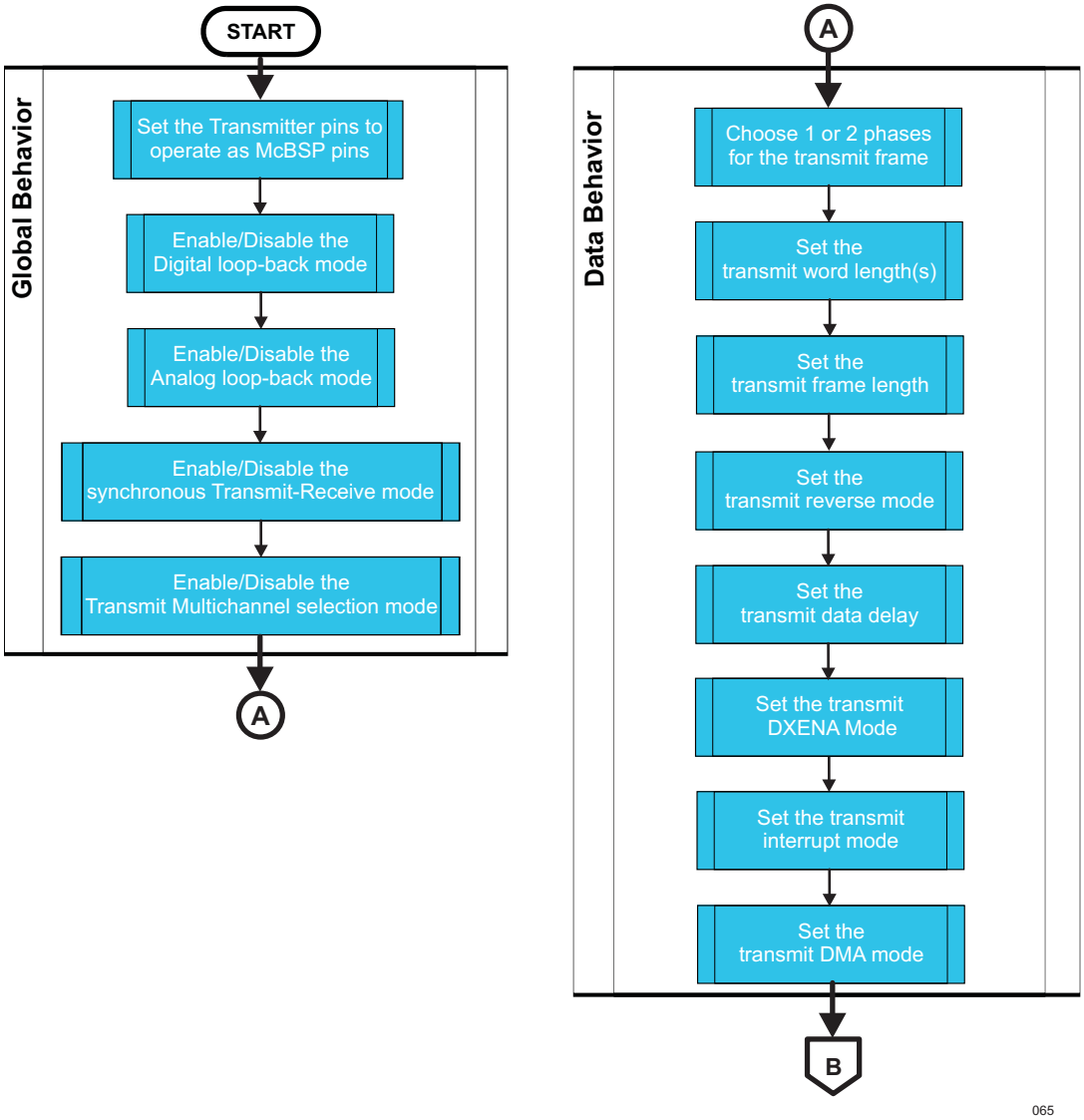
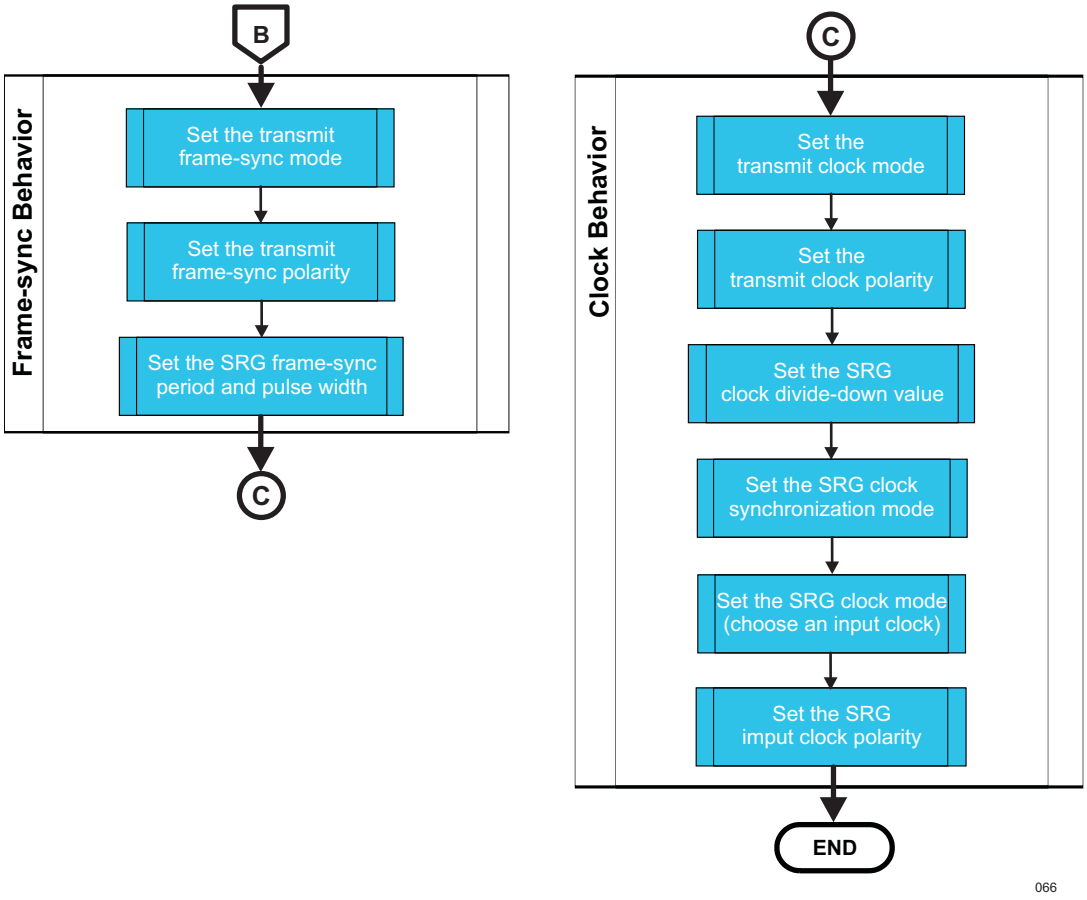


Figure 21-67. Important Tasks to Configure the McBSP Transmitter (Part 2)



21.5.1.6.2.1 Global Behavior

21.5.1.6.2.1.1 Set the Transmitter Pins to Operate as McBSP Pins

McBSPi.MCBSPLP_PCR_REG[13] XIOEN bit determines whether the transmitter pins are McBSP pins (XIOEN=0) or general–purpose I/O pins (XIOEN=1).

Refer to Section 21.5.1.7 which describes how to use McBSP pins as GPIO pins.

21.5.1.6.2.1.2 Enable/Disable the Digital Loop-Back Mode

Refer to it, Section 21.5.1.5.2.1.2.

21.5.1.6.2.1.3 Enable/Disable the Analog Loop-Back Mode

Refer to it, Section 21.5.1.5.2.1.3.

21.5.1.6.2.1.4 Enable/Disable the Synchronous Transmit-Receive Mode (McBSP1 only)

Refer to it, Section 21.5.1.5.2.1.4.

21.5.1.6.2.1.5 Enable/Disable the Transmit Multichannel Selection

McBSPi.MCBSPLP_MCR2_REG[1:0] XMCM bit field determines whether the transmit multichannel selection mode is on or off.

Refer to Section 21.4.6.7.

21.5.1.6.2.2 Data Behavior

21.5.1.6.2.2.1 Choose 1 or 2 Phases for the Transmit Frame

McBSPi.MCBSPLP_XCR2_REG[15] XPHASE bit determines whether the transmit data frame has one (Single phase) or two phases (Dual phase). When dual-phase is selected the number of words per phase must be set to one.

21.5.1.6.2.2.2 Set the Transmit Word Length(s)

McBSPi.MCBSPLP_XCR1_REG[7:5] XWDLEN1 and McBSPi.MCBSPLP_XCR2_REG[7:5] XWDLEN2 bit fields determine how many bits are in each serial word in phase 1 and in phase 2, respectively, of the transmit data frame.

If a single–phase frame is selected, XWDLEN1 selects the length for every serial word received in the frame.

If a dual–phase frame is selected, XWDLEN1 and XWDLEN2 must be set to select the both length. These both bits can have values different.

21.5.1.6.2.2.3 Set the Transmit Frame Length

The transmit frame length is the number of serial words in the transmit frame.

McBSPi.MCBSPLP_XCR1_REG[14:8] XFRLEN1 and McBSPi.MCBSPLP_XCR2_REG[14:8] XFRLEN2 bit fields determine how many serial words are in phase 1 and in phase 2, respectively, of the transmit data frame.

If a dual–phase frame is selected (XPHASE=1), the frame length is 2 words, the length of phase 1 (one word) plus the length of phase 2 (one word). Others values must not be used.

The 7–bit XFRLEN fields allow up to 128 words per phase. See Table 21-31 for a summary of how to calculate the frame length. This length corresponds to the number of words or logical time slots or channels per frame-synchronization pulse.

Program the XFRLEN fields with [W minus 1], where W represents the number of words per phase. For the example, if you want a phase length of 128 words in phase 1, load 127 into XFRLEN1.

Table 21-31. How to Calculate the Length of the Transmit Frame

XPHASE	XFRLEN1	XFRLEN2	Frame Length
0	0 ≤ XFRLEN1 ≤ 127	Don't care	(XFRLEN1value +1) words
1	XFRLEN1= 0	XFRLEN2 = 0	2 words

21.5.1.6.2.2.4 Set the Transmit Reverse Mode

McBSPi.MCBSPLP_XCR2_REG[4:3] XREVERSE bit field determines whether reverse (LSB first) data transfer option is chosen for McBSP reception.

For additional information about reverse mode, see Section 21.4.2.2.

21.5.1.6.2.2.5 Set the Transmit Data Delay

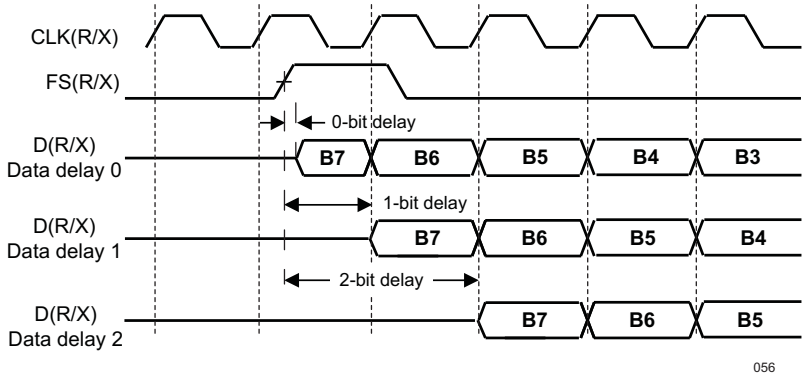
McBSPi.MCBSPLP_XCR2_REG[1:0] XDATDLY bit field determines the length of the data delay for the transmit frame.

The start of a frame is defined by the first clock cycle in which frame synchronization is found to be active. The beginning of actual data transmission with respect to the start of the frame can be delayed if required. This delay is called data delay.

XDATDLY specifies the data delay for reception. The range of programmable data delay is zero to two bit–clocks (XDATDLY=00b–10b), as shown in Figure 21-68. In this figure, the data transferred is an 8–bit value with bits labeled B7, B6, B5, and so on.

Note: Typically a 1–bit delay is selected, because data often follows a 1–cycle active frame-synchronization pulse.

Figure 21-68. Range of Programmable Data Delay



21.5.1.6.2.2.5.1 0-Bit Data Delay:

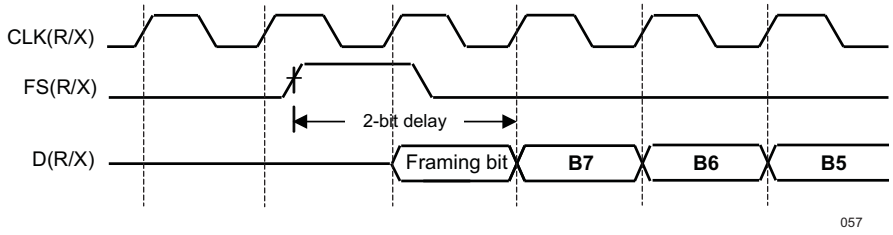
Normally, a frame-synchronization pulse is detected or sampled with respect to an edge of internal serial clock CLK(R/X). Thus, on the following cycle or later (depending on the data delay value), data may be received or transmitted. However, in the case of 0–bit data delay, the data must be ready for reception and/or transmission on the same serial clock cycle.

For reception, this problem is solved because receive data is sampled on the first falling edge of CLKR where an active–high internal FSR is detected. However, data transmission must begin on the rising edge of the internal CLKX clock that generated the frame synchronization. Therefore, the first data bit is assumed to be present in XSR1, and thus on mcbspi_dx. The transmitter then asynchronously detects the frame-synchronization signal (FSX) going active high and immediately starts driving the first bit to be transmitted on the mcbsp_dx pin.

21.5.1.6.2.2.5.2 2-Bit Data Delay:

A data delay of two bit periods allows the serial port to interface to different types of T1 framing devices where the data stream is preceded by a framing bit. During reception of such a stream with data delay of two bits (framing bit appears after a 1–bit delay and data appears after a 2–bit delay), the serial port essentially discards the framing bit from the data stream, as shown in Figure 21-69. The data transferred is an 8–bit value with bits labeled B7, B6, B5, and so on.

Figure 21-69. 2-Bit Data Delay Used to Skip a Framing Bit



21.5.1.6.2.2.6 Set the Transmit DXENA Mode

McBSPi.MCBSPLP_SPCR1_REG[7] DXENA bit is used to set the transmit DXENA (DX delay enable) mode.

The DXENA bit controls the delay enabler on the mcbsp_dx pin. Set DXENA to enable an extra delay for

turn–on time. This bit does not control the data itself, so only the first bit is delayed (the delay is given by a combinatorial delay buffer). The inserted delay: 7 ns, 14 ns (default), 20 ns or 28 ns can be set using the McBSPi.MCBSPLP_XCCR_REG[13:12] DXENDLY field. If you tie together the mcbsp_dx pins of multiple McBSP modules, make sure DXENA=1, to avoid having more than one McBSP transmitting on the data line at one time.

21.5.1.6.2.2.7 Set the Transmit Interrupt Mode

Refer to Section 21.5.1.4.2.2.

21.5.1.6.2.2.8 Set the Transmit DMA Mode

The McBSP transmit data DMA requests (McBSPi_DMA_TX) are active after the transmit McBSPi.MCBSPLP_SPCR2_REG[0] XRST bit is released. After reset, the default DMA threshold (and length) is 1.

The transmit DMA requests can be disabled by setting the McBSPi.MCBSPLP_XCCR_REG[3] XDMAEN bit to 0. When disabling the DMA, the DMA request line is deasserted even if a DMA transfer is pending, and the DMA state-machine is not reset.

To configure the McBSP transmit data DMA requests, follow this procedure:

- Write the transmit McBSPi.MCBSPLP_THRSH2_REG register with the required transmit DMA request length (the length of the transfer is the same as the threshold value + 1).
- As long as the free locations level in XB is above or equal to the THRSH2_REG value + 1, the DMA request is asserted.
- After transferring the configured (THRSH2_REG value + 1) number of words, the transmit DMA request is deasserted, and then reasserted as soon as the conditions are met again.

21.5.1.6.2.3 Frame-Synchronization Behavior

21.5.1.6.2.3.1 Set the Transmit Frame-Sync Mode

McBSPi.MCBSPLP_PCR_REG[11] FSXM bit and McBSPi.MCBSPLP_SRGR2_REG[12] FSGM bit are used to set the transmit frame-sync mode.

Table 21-32 shows how FSXM and FSGM select the source of transmit frame-synchronization pulses. The three choices are:

1. External frame-synchronization input
2. Sample rate generator frame-synchronization signal (FSG)
3. Sample rate generator frame-synchronization signal (FSG) gated by the transmit buffer XB empty condition.

Table 21-32 also shows the effect of each bit setting on the mcbsp_fsx pin. The FSXP bit determines the polarity of the signal on the mcbsp_fsx pin.

Table 21-32. How FSXM and FSGM Select the Source of Transmit Frame-Sync Pulses

FSXM	FSGM	Source of Transmit Frame Synchronization	mcbsp_fsx Pin Status
0	0 or 1	An external FSG enters the McBSP through the mcbsp_fsx pin. The signal is then inverted by FSXP bit before being used as internal FSX.	Input
1	1	Internal FSX is driven by the SRG FSG.	Output. FSG is inverted by FSXP bit before being driven out on mcbsp_fsx pin.
1	0	A XB empty condition causes the McBSP not to generate a transmit frame-synchronization pulse. The frame synchronization is generated taking into account the FWID and FPER bits as long as the transmit buffer is not empty. When the buffer is empty the generated frame synchronization signal is gated.	Output. The generated frame-synchronization pulse is inverted as determined by FSXP bit before being driven out on mcbsp_fsx pin.

If the SRG creates a frame-synchronization signal (FSG) that is derived from an external input clock, the GSYNC bit determines whether FSG is kept synchronized with pulses on the mcbsp_fsr pin. For more details, see Section 21.4.3.3.

In the digital loopback mode (DLB=1) or analog loopback mode (ALB = 1), the transmit frame synchronization signal is used as the receive frame synchronization signal. For more details on frame-sync configuration, see Section 21.4.3.2.

21.5.1.6.2.3.2 Set the Transmit Frame-Sync Polarity

McBSPi.MCBSPLP_PCR_REG[3] FSXP bit determines whether frame-synchronization pulses are active high or active low on the mcbsp_fsx pin.

Transmit frame-synchronization pulses can be generated internally by the SRG or driven by an external source. The source of frame synchronization is selected by programming the McBSPi.MCBSPLP_PCR_REG[11] FSXM bit. FSX is also affected by the McBSPi.MCBSPLP_SRGR2_REG[12] FSGM bit. For information about the effects of FSXM and FSGM, see Section 21.5.1.6.2.3.1).

When FSR and FSX are inputs (FSXM=FSRM=0, external frame-synchronization pulses), the McBSP module detects them on the internal falling edge of clock, internal CLKR, and internal CLKX, respectively. The receive data arriving at the mcbsp_dr pin is also sampled on the falling edge of internal CLKR. These internal clock signals are either derived from external source via CLK(R/X) pins or driven by the SRG clock (CLKG) internal to the McBSP module.

When FSR and FSX are outputs, implying that they are driven by the SRG, they are generated (transition to their active state) on the rising edge of internal clock, CLK(R/X). Similarly, data on the mcbsp_dx pin is output on the rising edge of internal CLKX.

FSRP, FSXP, CLKRP, and CLKXP in the pin control register (PCR_REG) configure the polarities of the FSR, FSX, CLKR, and CLKX signals, respectively. All FSG (internal FSR, internal FSX) that are internal to the serial port are active high. If the serial port is configured for external frame synchronization (FSR/FSX are inputs to McBSP) and FSRP=FSXP=1, the external active-low frame-synchronization signals are inverted before being sent to the receiver (internal FSR) and transmitter (internal FSX). Similarly, if internal synchronization (FSR/FSX are output pins and GSYNC=0) is selected and the polarity bit FS(R/X)P=1, the internal active–high FSG are inverted before being sent to the FS(R/X) pin.

21.5.1.6.2.3.3 Set the SRG Frame-Sync Period and Pulse Width

See Section 21.5.1.5.2.3.3.

21.5.1.6.2.4 Clock Behavior

21.5.1.6.2.4.1 Set the Transmit Clock Mode

The McBSPi.MCBSPLP_PCR_REG[9] CLKXM bit is used to set the transmit clock mode.

Table 21-33 shows how the CLKXM bit selects the transmit clock and the corresponding status of the mcbsp_clkx pin. The CLKXP bit determines the polarity of the signal on the mcbsp_clkx pin.

Table 21-33. CLKXM Bit Effect on Transmit Clock and MCBSPLP.CLKX Pin

CLKXM	Source of Transmit Clock	mcbsp_clkx Pin Status
0	Internal CLKX is driven by an external clock on the mcbsp_clkx pin. CLKX is inverted as determined by CLKXP before being used.	Input
1	Internal CLKX is driven by the SRG clock, CLKG.	Output. CLKG, inverted as determined by CLKXP, is driven out on mcbsp_clkx.

If the SRG creates a clock signal (CLKG) that is derived from an external input clock, the GSYNC bit determines whether CLKG is kept synchronized with pulses on the mcbsp_fsr pin.

In the digital loopback mode (DLB=1) or analog loopback mode (ALB = 1), the transmit frame synchronization signal is used as the receive frame synchronization signal. For more details on clock configuration, see Section 21.4.3.1.

21.5.1.6.2.4.2 Set the Transmit Clock Polarity

The McBSPi.MCBSPLP_PCR_REG[1] CLKXP bit is used to set the transmit clock polarity.

On the transmit side, the transmit clock polarity bit, CLKXP, sets the edge used to shift and clock out transmit data. Data is always transmitted on the rising edge of internal CLKX. If CLKXP=1 and external clocking is selected (CLKXM=0 and CLKX is an input), the external falling–edge triggered input clock on CLKX is inverted to a rising–edge triggered clock before being sent to the transmitter. If CLKXP=1 and internal clocking is selected (CLKXM=1 and CLKX is an output pin), the internal (rising–edge triggered) clock, internal CLKX, is inverted before being sent out on the mcbsp_clkx pin.

Similarly, the receiver can reliably sample data that is clocked with a rising edge clock (by the transmitter). The receive clock polarity bit, CLKRP, sets the edge used to sample received data. The receive data is always sampled on the falling edge of internal CLKR. Therefore, if CLKRP=1 and external clocking is selected (CLKRM=0 and CLKR is an input pin), the external rising–edge triggered input clock on CLKR is inverted to a falling–edge triggered clock before being sent to the receiver. If CLKRP=1 and internal clocking is selected (CLKRM=1), the internal falling–edge triggered clock is inverted to a rising–edge triggered clock before being sent out on the mcbsp_clkr pin.

Note: CLKRP=CLKXP in a system where the same clock (internal or external) is used to clock the receiver and transmitter. The receiver uses the opposite edge as the transmitter to ensure valid setup and hold of data around this edge.

21.5.1.6.2.4.3 Set the SRG Clock Divide-Down Value

See Section 21.5.1.5.2.4.3.

21.5.1.6.2.4.4 Set the SRG Clock Synchronization Mode

See Section 21.5.1.5.2.4.4.

21.5.1.6.2.4.5 Set the SRG Clock Mode (Choose an Input Clock)

See Section 21.5.1.5.2.4.5.

21.5.1.6.2.4.6 Set the SRG Input Clock Polarity

See Section 21.5.1.5.2.4.6.

21.5.1.7 General-Purpose I/O on the McBSP Pins (Legacy Only)

Table 21-34 summarizes how to use the McBSP pins as general–purpose I/O pins. All the bits mentioned in the table except XRST and RRST bits are in the pin control register (McBSPi.MCBSPLP_PCR_REG). McBSPi.MCBSPLP_SPCR2_REG[0] XRST bit and McBSPi.MCBSPLP_SPCR1_REG[0] RRST bit are in the serial port control registers.

To use receiver pins mcbsp_clkr, mcbsp_fsr, and mcbsp_dr as general–purpose I/O pins rather than as serial port pins, you must set two conditions:

1. The receiver of the serial port is in reset (McBSPi.MCBSPLP_SPCR1_REG[0] RRST bit =0).
2. General–purpose I/O is enabled for the serial port receiver (McBSPi.MCBSPLP_PCR_REG[12] RIOEN=1).

The mcbsp_clkr and mcbsp_fsr pins can be individually configured as either input or output pins with the CLKRM and FSRM bits, respectively. The mcbsp_dr pin can only be an input pin. Table 21-34 shows, which bits in McBSPi.MCBSPLP_PCR_REG are used to read from/write to these pins.

For the transmitter pins mcbsp_clkx, mcbsp_fsx, and mcbsp_dx, you must meet two conditions:

1. The transmitter of the serial port is in reset (McBSPi.MCBSPLP_SPCR2_REG[0] XRST=0).
2. General–purpose I/O is enabled for the serial port transmitter (McBSPi.MCBSPLP_PCR_REG[12] XIOEN=1).

The mcbsp_clkx and mcbsp_fsx pins can be individually configured as input or output pins with the CLKXM and FSXM bits, respectively. The mcbsp_dx pin can only be an output pin. Table 21-34 shows the bits in McBSPi.MCBSPLP_PCR_REG used to read from/write to these pins.

For the mcbsp_clks pin (common to all McBSP modules), all of the reset and I/O enable conditions must be met:

1. Both the receiver and transmitter of the serial port are in reset (RRST=0 and XRST=0).
2. General–purpose I/O is enabled for both the receiver and the transmitter (RIOEN=1 and XIOEN=1).

The mcbsp_clks pin can only be an input pin. To read the status of the signal on the mcbsp_clks pin, read the McBSPi.MCBSPLP_PCR_REG[6] CLKS_STAT bit.

Table 21-34. Using McBSP Pins for General-Purpose I/O

Pin	General–Purpose use enabled by this bit combination	Selected as output when	Output value driven from this bit	Selected as input when	Input value read from this bit
mcbsp_clkx	XRST = 0 XIOEN = 1	CLKXM = 1	CLKXP	CLKXM = 0	CLKXP
mcbsp_fsx	XRST = 0 XIOEN = 1	FSXM = 1	FSXP	FSXM = 0	FSXP
mcbsp_dx	XRST = 0 XIOEN = 1	Always	DX_STAT	Never	Does not apply
mcbsp_clkr	RRST = 0 RIOEN = 1	CLKRM = 1	CLKRP	CLKRM = 0	CLKRP
mcbsp_fsr	RRST = 0 RIOEN = 1	FSRM = 1	FSRP	FSRM = 0	FSRP
mcbsp_dr	RRST = 0 RIOEN = 1	Never	Does not apply	Always	DR_STAT
mcbsp_clks	RRST = XRST = 0 RIOEN = XIOEN = 1	Never	Does not apply	Always	CLKS_STAT

21.5.1.8 Data Packing Examples

This section describes two ways to implement data packing in the McBSP: Using frame length and word length, and using word length and ignoring frame sync pulses.

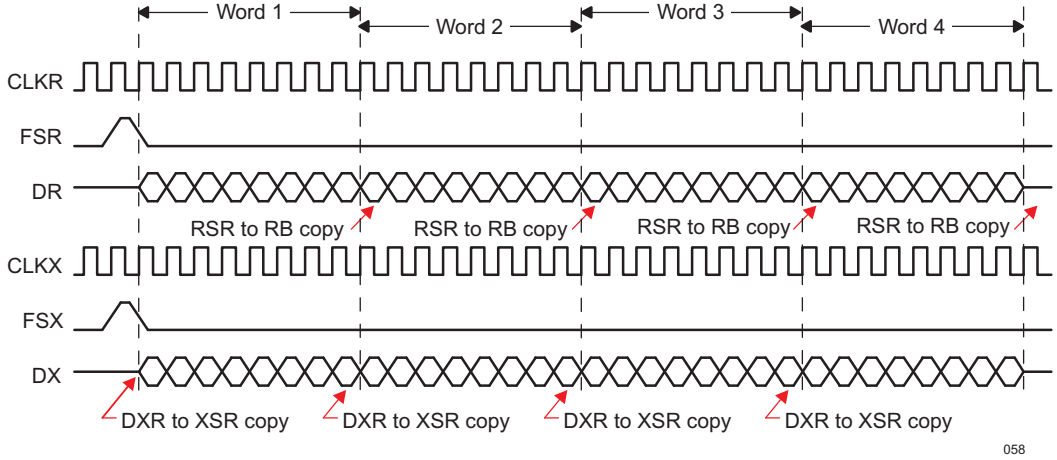
21.5.1.8.1 Data Packing Using Frame Length and Word Length

Frame length and word length can be manipulated to pack data effectively. For example, four 8-bit words are transferred in a single-phase frame, as shown in Figure 21-70. In this case:

- (R/X)PHASE = 0: Single-phase frame
- (R/X)FRLEN1 = 00011b: 4-word frame
- (R/X)WDLEN1 = 101b: 32-bit words

Four 8-bit data words are transferred to and from the McBSP by the MPU/IVA2.2 subsystems or the sDMA controller. Thus, four reads from McBSPi.MCBSPLP_DRR_REG and four writes to McBSPi.MCBSPLP_DXR_REG are necessary for each frame.

Figure 21-70. Four 8-bit Data Words Transferred To/From McBSP Module

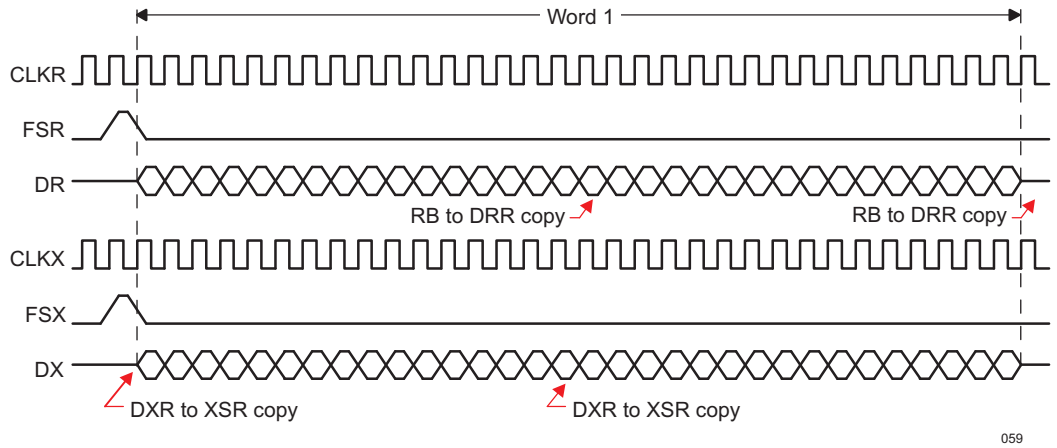


This data can also be treated as a single-phase frame consisting of one 32-bit data word, as shown in Figure 21-71. In this case:

- (R/X)PHASE = 0: Single-phase frame
- (R/X)FRLEN1 = 0000000b: 1-word frame
- (R/X)WDLEN1 = 000b: 8-bit words

Two 16-bit data words are transferred to and from the McBSP by the MPU/IVA2.2 subsystems or the sDMA controller. Thus, two reads from McBSPi.MCBSPLP_DRR_REG and two writes to McBSPi.MCBSPLP_DXR_REG are necessary for each frame. This results in only half the number of transfers, as compared to the previous case. This manipulation reduces the percentage of bus time required for serial port data movement.

Figure 21-71. One 32-bit Data Word Transferred To/From McBSP Module



21.5.1.8.2 Data Packing Using Word Length and the Frame-Sync Ignore Function

When there are multiple words per frame, data can be packed by increasing the word length (defining a serial word with more bits) and by ignoring frame-sync pulses. Figure 21-72 shows the McBSP operating at the maximum packet frequency. Here, each frame has only one 8-bit word. Notice the frame-sync pulse that initiates each frame transfer for reception and for transmission. For reception, this configuration requires one read operation for each word. For transmission, this configuration requires one write operation for each word.

Figure 21-72. 8-bit Data Words Transferred at Maximum Packet Frequency

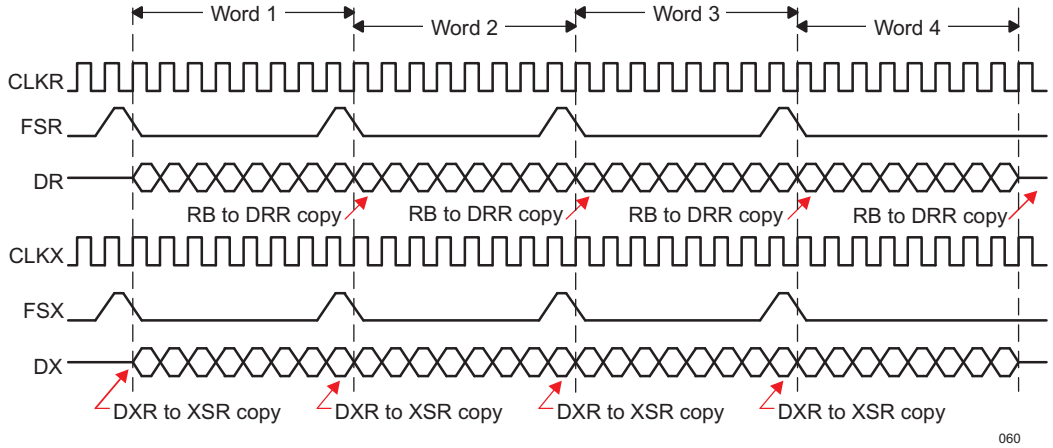
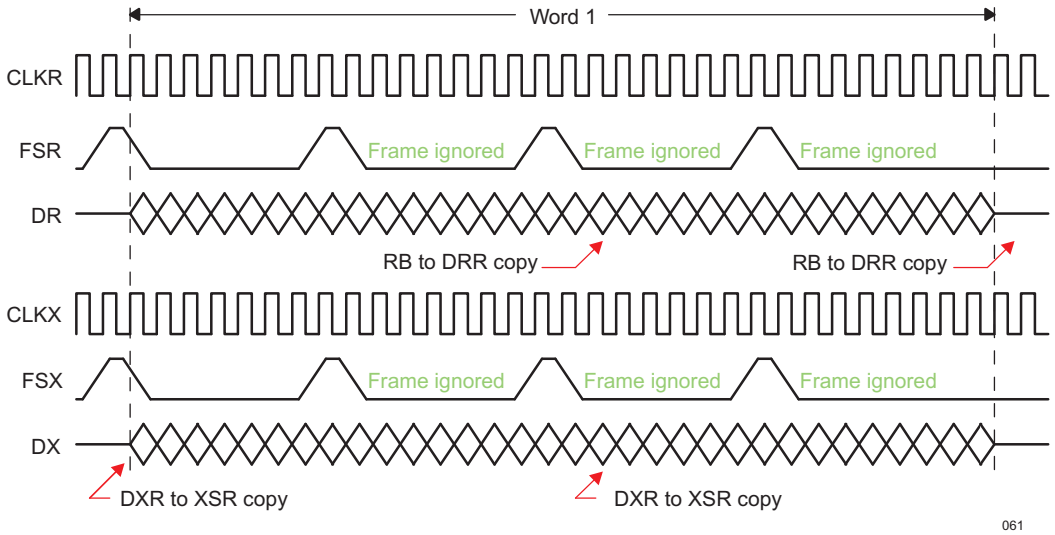


Figure 21-73 shows the McBSP configured to treat this data stream as a continuous 32-bit word. In this example, the McBSP responds to an initial frame-sync pulse. However, the McBSP ignores subsequent pulses. Only one read transfer or one write transfer is required every 32 bits. This configuration effectively reduces the required bus bandwidth to one-fourth the bandwidth needed to transfer four 8-bit words.

Figure 21-73. Configuring the Data Stream as a Continuous 32-bit Word



21.5.2 SIDETONE Feature

21.5.2.1 SIDETONE Activation Procedure

Before you enable a SIDETONE selection mode, make sure you properly configure the data frame for multichannel mode of the McBSP module:

- Select a single–phase frame (McBSPi.MCBSPLP_RCR2_REG[15] RPHASE bit and McBSPi.MCBSPLP_XCR2_REG[15] XPHASE bit=0). Each frame represents a TDM data stream.
- Set to 1 the McBSPi.MCBSPLP_MCR1_REG[0] RMCM bit, to select multichannel mode enable.
- Set a frame length (McBSPi.MCBSPLP_RCR1_REG[14:8] RFRLEN1 bit field and McBSPi.MCBSPLP_XCR1_REG[14:8] XFRLEN1 bit field) that includes the highest–numbered channel to be used (a maximum of 4 channels can be used in this configuration).
- Set a word length (McBSPi.MCBSPLP_RCR1_REG[7:5] RWDLEN1 bit field and McBSPi.MCBSPLP_XCR1_REG[7:5] XWDLEN1 bit field) to be either 16, 24 or 32 (see the note

below).

- Select the input/output channels configured as SIDETONE channels by setting the following bit fields listed in Table 21-35:

Table 21-35. Selection of the SIDETONE Input and Output Channels

Bit field	Description
McBSPi.MCBSPLP_SSELCR_REG[9:7] OCH1ASSIGN	Map the CH1 data for the speaker out channels to one of the McBSP channels (1 out of 8 channels)
McBSPi.MCBSPLP_SSELCR_REG[6:4] OCH0ASSIGN	Map the CH0 data for the speaker out channels to one of the McBSP channels (1 out of 8 channels)
McBSPi.MCBSPLP_SSELCR_REG[3:2] ICH1ASSIGN	Map the CH1 data from the digital microphone channels to one of the McBSP channels (1 out of 4 channels)
McBSPi.MCBSPLP_SSELCR_REG[1:0] ICH0ASSIGN	Map the CH0 data from the digital microphone channels to one of the McBSP channels (1 out of 4 channels)

- Enable SIDETONE by setting 2 bits to 1:
  1. In McBSP module, the McBSPi.MCBSPLP_SSELCR_REG[10] SIDETONEEN bit
  2. In SIDETONE core, the McBSPi.ST_SSELCR_REG[0] SIDETONEEN bit

Note: Word width in the loop is 24 bits. If input channel word width is less than 24 bits, LSBs of the samples are zero padded. If input channel word width is more than 24 bits, samples are truncated.

21.5.2.2 SIDETONE Initialization Procedure

The SIDETONE core initialization procedure is as follows:

1. Write 1 in McBSPi.ST_SSELCR_REG[1] COEFFWREN bit to enable loading of the FIR coefficients.
2. Load one by one the FIR coefficients performing 128 write accesses to McBSPi.ST_SFIRCR_REG[15:0] FIRCOEFF bit field, the McBSPi.ST_SFIRCR_REG[0] FIRCOEFF bit is loaded first. To ensure the completion of loading, check the status bit-field, McBSPi.ST_SSELCR_REG[2] COEFFWRDONE bit.
3. Set-up gain values for both channels writing desired values inside McBSPi.ST_SGAINCR_REG[31:16] CH1GAIN bit field for the second sidetone channel and McBSPi.ST_SGAINCR_REG[15:0] CH0GAIN bit field for the first sidetone channel.

21.5.2.3 SIDETONE FIR Coefficients Writing

Writing the coefficients is only possible when the SIDETONE is disabled.

1. Write 1 in McBSPi.ST_SSELCR_REG[1] COEFFWREN bit to enable writing of the FIR coefficients, or write 0, and then write 1 in the COEFFWREN bit to reset the write process.
2. Perform 128 write accesses in 32/16 LSB bit mode on the McBSPi.ST_SFIRCR_REG[15:0] FIRCOEFF bit field. The first write action following the previous step sets the coefficient index 0.
3. Check that the write is done by reading McBSPi.ST_SSELCR_REG[2] COEFFWRDONE bit (it becomes 1 after the 128th coefficient is written).

21.5.2.4 SIDETONE FIR Coefficients Reading

Reading the coefficients is only possible when the SIDETONE is disabled.

- Write 0 in McBSPi.ST_SSELCR_REG[1] COEFFWREN bit to enable reading of the FIR coefficients, or write 1, and then write 0 in the COEFFWREN bit to reset the read process.
- Perform 128 read accesses from McBSPi.ST_SFIRCR_REG[15:0] FIRCOEFF bit field. Each read returns coefficients one by one. The first read following previous step returns coefficient index 0.

21.6 McBSP Use Case and Tips

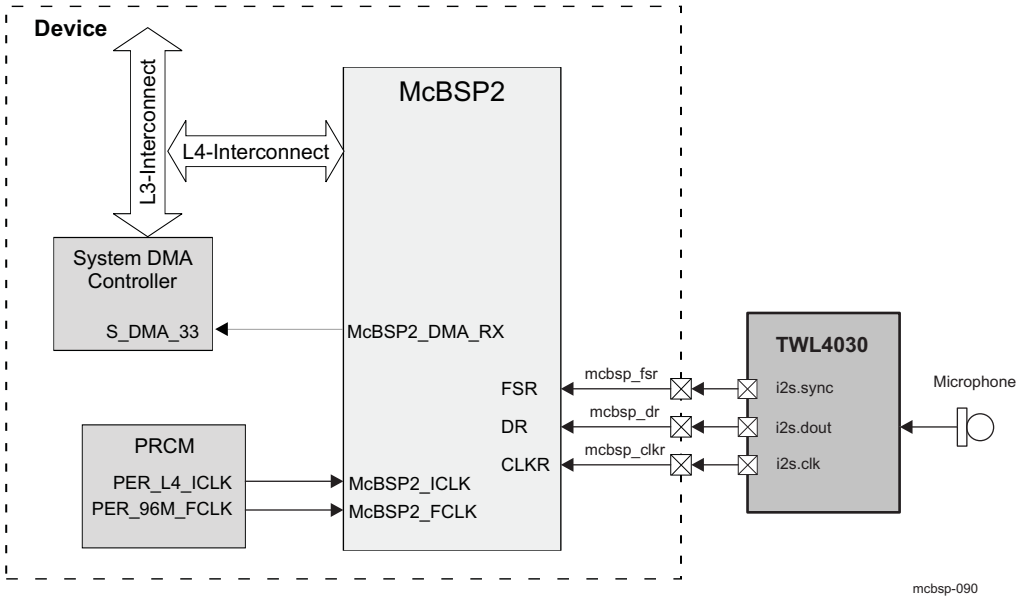
21.6.1 Camcorder Use Case: How to Configure the McBSP2 to Receive Voice Stream From the TWL4030 Device

21.6.1.1 Overview

In the camcorder use case, the McBSP2 is used to transfer voice from the TWL4030 device to external DRAM through the sDMA controller. The McBSP2 module is directly connected to the TWL4030 device in slave mode (that is, the TWL4030 generates clock and FSR signals to manage data.

Figure 21-74 is an overview of the voice path.

Figure 21-74. Overview

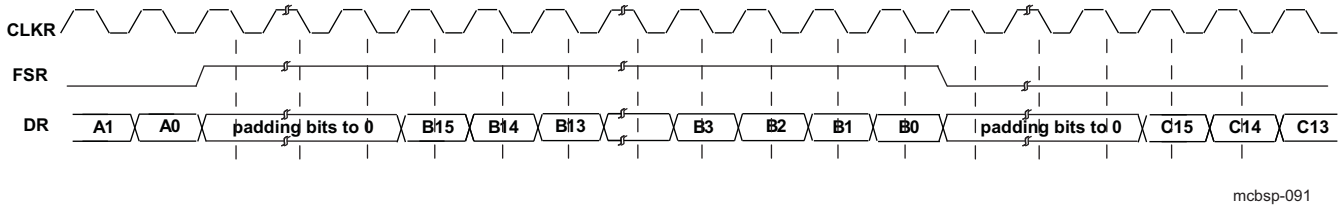


The TWL4030 device samples audio data at 16 kHz/16 bits in stereo. In the camcorder user case, a monophonic microphone is plugged on one channel of the TWL4030 device. Although only one channel is used, both left and right audio channels are transmitted to the McBSP. Moreover, the 16-bit channels are transmitted on two 32-bit words for this use case. The result is an I2S format with the following properties:

- Word width = 16 bits
- Sample length = 2 x 32 bits (two channels at 16 bits over 32 bits)
- Right-justified data format (zero-fill MSBs)

The TWL4030 device transmits two 32-bit frames that correspond to the two channels of a stereo signal, one on the low level of the frame synchronization signal, and the other on the high level. The microphone is monophonic; therefore, the McBSP2 is configured to receive only the right channel. Figure 21-75 shows the input data, CLKR, and FSR signal from the TWL4030 device.

Figure 21-75. I2S Format



21.6.1.2 Programming Flow

The McBSP2 is configured as a slave receiver device driven by the TWL4030 device. It is configured to receive input data (see Figure 21-75). When the configuration is complete, the sDMA controller can transfer data from MCBSPLP_DRR_REG to the external DRAM.

First, the correct functional clock must be selected in the system control module. The MCBSP2_CLKS bit of the CONTROL.CONTROL_DEVCONF0[6] register is set to 0 to make PER_96M_FCLK the functional clock of the module (CLKS).

Next, the McBSP2 clocks are enabled on the PRCM:

- Enable the functional clock: PRCM.CM_FLCKEN_PER[0] = 1
- Enable the interface clock: PRCM.CM_ILCKEN_PER[0] = 1

Then, the initialization procedure is followed (see Figure 21-57):

1. Put the receiver and frame-sync generator under reset:
   - MCBSPLP_SPCR1_REG[0] RRST = 0
   - MCBSPLP_SPCR2_REG[7] FRST = 0
2. Program the configuration registers as required:
   a. Set one phase for the receive frame (only one channel is taken in account):
      MCBSPLP_RCR2_REG[15] RPHASE = 0
   b. Set the receive data delay to 0-bit data delay:
      MCBSPLP_RCR2_REG[1:0] RDATDLY = 0
   c. Set Receive Word Length to 32 bits:
      MCBSPLP_RCR1_REG[7:5] RWDLEN1 = 5
   d. Set the number of words per frame to 1:
      MCBSPLP_RCR1_REG[14:8] RFRLEN1 = 0
   e. Set the threshold to one-half of the FIFO size (that is, 640):
      MCBSPLP_THRSH1_REG[10:0] RTHRESHOLD = 0x280
   f. Set the receive sign-extension and justification mode to right-justify and zero-fill MSBs in MCBSPLP_DRR_REG:
      MCBSPLP_SPCR1_REG[14:13] RJUST = 0
   g. Set the receive frame-synchronization mode, frame-synchronization pulses generated by an external device. FSR is an input pin.
      MCBSPLP_PCR_REG[10] FSRM = 0
   h. Set the receive frame-synchronization polarity to active high: The channel on the high level of FSR will be received.
      MCBSPLP_PCR_REG[2] FSRP = 0
   i. Set the receive clock mode: Receive clock is an input driven by an external clock with CLKR as an input pin.
      MCBSPLP_PCR_REG[8] CLKR = 0
   j. Set the receive clock polarity: Receive data sampled on the falling edge of CLKR.
      MCBSPLP_PCR_REG[0] CLKRP = 0
   k. Wait for two clock cycles.
3. Not applicable in receiver mode.
4. Release the receiver reset:
   MCBSPLP_SPCR1_REG[0] RRST = 1
5. Internally generated frame-sync Is not required:
   a. The frame-sync generator reset is not released.
   b. Wait for two clock cycles.

Table 21-36 lists the registers involved during a transfer.

Table 21-36. McBSP2 Registers Print

Register Name	Address	Value	Value Description
MCBSPLP_SPCR1_REG	0x4902 2014	0x0000 0001	Receiver started
MCBSPLP_SPCR2_REG	0x4902 2010	0x0000 0000	Frame-synch generator under reset
MCBSPLP_RCR1_REG	0x4902 201C	0x0000 00a0	32-bit word length and 1 word per frame
MCBSPLP_RCR2_REG	0x4902 2018	0x0000 0000	0-bit data delay
MCBSPLP_THRSH1_REG	0x4902 2094	0x0000 007f	FIFO threshold
MCBSPLP_PCR_REG	0x4902 2048	0x0000 0000	Pin configuration

21.7 McBSP Register Manual

Table 21-37 shows the base address and address space for the OMAP34xx module instances.

Table 21-37. OMAP34xx Instance Summary

Module Name	Base Address (hex)	Size
McBSP1	0x4807 4000	4K bytes
McBSP5	0x4809 6000	4K bytes
McBSP2	0x4902 2000	4K bytes
McBSP3	0x4902 4000	4K bytes
McBSP4	0x4902 6000	4K bytes
SIDETONE_McBSP2	0x4902 8000	4K bytes
SIDETONE_McBSP3	0x4902 A000	4K bytes

21.7.1 McBSP Register Mapping Summary

CAUTION

The McBSP data registers (that is, DXR_REG for data transmit and DRR_REG for data receive) support 8/16/32-bit data accesses. All other McBSP registers are limited to 32-bit data accesses. 16-bit and 8-bit data accesses are not allowed and can corrupt register contents.

Table 21-38. McBSP1 Registers Mapping Summary

Register Name	Type	Register Width (Bits)	Address Offset	Physical Address
MCBSPLP_DRR_REG	R	32	0x0000 0000	0x4807 4000
MCBSPLP_DXR_REG	W	32	0x0000 0008	0x4807 4008
MCBSPLP_SPCR2_REG	RW	32	0x0000 0010	0x4807 4010
MCBSPLP_SPCR1_REG	RW	32	0x0000 0014	0x4807 4014
MCBSPLP_RCR2_REG	RW	32	0x0000 0018	0x4807 4018
MCBSPLP_RCR1_REG	RW	32	0x0000 001C	0x4807 401C
MCBSPLP_XCR2_REG	RW	32	0x0000 0020	0x4807 4020
MCBSPLP_XCR1_REG	RW	32	0x0000 0024	0x4807 4024
MCBSPLP_SRGR2_REG	RW	32	0x0000 0028	0x4807 4028
MCBSPLP_SRGR1_REG	RW	32	0x0000 002C	0x4807 402C
MCBSPLP_MCR2_REG	RW	32	0x0000 0030	0x4807 4030
MCBSPLP_MCR1_REG	RW	32	0x0000 0034	0x4807 4034
MCBSPLP_RCERA_REG	RW	32	0x0000 0038	0x4807 4038
MCBSPLP_RCERB_REG	RW	32	0x0000 003C	0x4807 403C
MCBSPLP_XCERA_REG	RW	32	0x0000 0040	0x4807 4040
MCBSPLP_XCERB_REG	RW	32	0x0000 0044	0x4807 4044
MCBSPLP_PCR_REG	RW	32	0x0000 0048	0x4807 4048
MCBSPLP_RCERC_REG	RW	32	0x0000 004C	0x4807 404C
MCBSPLP_RCERD_REG	RW	32	0x0000 0050	0x4807 4050
MCBSPLP_XCERC_REG	RW	32	0x0000 0054	0x4807 4054
MCBSPLP_XCERD_REG	RW	32	0x0000 0058	0x4807 4058
MCBSPLP_RCERE_REG	RW	32	0x0000 005C	0x4807 405C
MCBSPLP_RCERF_REG	RW	32	0x0000 0060	0x4807 4060
MCBSPLP_XCERE_REG	RW	32	0x0000 0064	0x4807 4064

Table 21-38. McBSP1 Registers Mapping Summary (continued)

Register Name	Type	Register Width (Bits)	Address Offset	Physical Address
MCBSPLP_XCERF_REG	RW	32	0x0000 0068	0x4807 4068
MCBSPLP_RCERG_REG	RW	32	0x0000 006C	0x4807 406C
MCBSPLP_RCERH_REG	RW	32	0x0000 0070	0x4807 4070
MCBSPLP_XCERG_REG	RW	32	0x0000 0074	0x4807 4074
MCBSPLP_XCERH_REG	RW	32	0x0000 0078	0x4807 4078
MCBSPLP_REV_REG	R	32	0x0000 007C	0x4807 407C
MCBSPLP_RINTCLR_REG	RW	32	0x0000 0080	0x4807 4080
MCBSPLP_XINTCLR_REG	RW	32	0x0000 0084	0x4807 4084
MCBSPLP_ROVFLCLR_REG	RW	32	0x0000 0088	0x4807 4088
MCBSPLP_SYSCONFIG_REG	RW	32	0x0000 008C	0x4807 408C
MCBSPLP_THRSH2_REG	RW	32	0x0000 0090	0x4807 4090
MCBSPLP_THRSH1_REG	RW	32	0x0000 0094	0x4807 4094
MCBSPLP_IRQSTATUS_REG	RW	32	0x0000 00A0	0x4807 40A0
MCBSPLP_IRQENABLE_REG	RW	32	0x0000 00A4	0x4807 40A4
MCBSPLP_WAKEUPEN_REG	RW	32	0x0000 00A8	0x4807 40A8
MCBSPLP_XCCR_REG	RW	32	0x0000 00AC	0x4807 40AC
MCBSPLP_RCCR_REG	RW	32	0x0000 00B0	0x4807 40B0
MCBSPLP_XBUFFSTAT_REG	R	32	0x0000 00B4	0x4807 40B4
MCBSPLP_RBUFFSTAT_REG	R	32	0x0000 00B8	0x4807 40B8
MCBSPLP_SSELCR_REG	RW	32	0x0000 00BC	0x4807 40BC
MCBSPLP_STATUS_REG	R	32	0x0000 00C0	0x4807 40C0

Table 21-39. McBSP5 Registers Mapping Summary

Register Name	Type	Register Width (Bits)	Address Offset	Physical Address
MCBSPLP_DRR_REG	R	32	0x0000 0000	0x4809 6000
MCBSPLP_DXR_REG	W	32	0x0000 0008	0x4809 6008
MCBSPLP_SPCR2_REG	RW	32	0x0000 0010	0x4809 6010
MCBSPLP_SPCR1_REG	RW	32	0x0000 0014	0x4809 6014
MCBSPLP_RCR2_REG	RW	32	0x0000 0018	0x4809 6018
MCBSPLP_RCR1_REG	RW	32	0x0000 001C	0x4809 601C
MCBSPLP_XCR2_REG	RW	32	0x0000 0020	0x4809 6020
MCBSPLP_XCR1_REG	RW	32	0x0000 0024	0x4809 6024
MCBSPLP_SRGR2_REG	RW	32	0x0000 0028	0x4809 6028
MCBSPLP_SRGR1_REG	RW	32	0x0000 002C	0x4809 602C
MCBSPLP_MCR2_REG	RW	32	0x0000 0030	0x4809 6030
MCBSPLP_MCR1_REG	RW	32	0x0000 0034	0x4809 6034
MCBSPLP_RCERA_REG	RW	32	0x0000 0038	0x4809 6038
MCBSPLP_RCERB_REG	RW	32	0x0000 003C	0x4809 603C
MCBSPLP_XCERA_REG	RW	32	0x0000 0040	0x4809 6040
MCBSPLP_XCERB_REG	RW	32	0x0000 0044	0x4809 6044
MCBSPLP_PCR_REG	RW	32	0x0000 0048	0x4809 6048
MCBSPLP_RCERC_REG	RW	32	0x0000 004C	0x4809 604C
MCBSPLP_RCERD_REG	RW	32	0x0000 0050	0x4809 6050
MCBSPLP_XCERC_REG	RW	32	0x0000 0054	0x4809 6054
MCBSPLP_XCERD_REG	RW	32	0x0000 0058	0x4809 6058
MCBSPLP_RCERE_REG	RW	32	0x0000 005C	0x4809 605C

Table 21-39. McBSP5 Registers Mapping Summary (continued)

Register Name	Type	Register Width (Bits)	Address Offset	Physical Address
MCBSPLP_RCERF_REG	RW	32	0x0000 0060	0x4809 6060
MCBSPLP_XCERE_REG	RW	32	0x0000 0064	0x4809 6064
MCBSPLP_XCERF_REG	RW	32	0x0000 0068	0x4809 6068
MCBSPLP_RCERG_REG	RW	32	0x0000 006C	0x4809 606C
MCBSPLP_RCERH_REG	RW	32	0x0000 0070	0x4809 6070
MCBSPLP_XCERG_REG	RW	32	0x0000 0074	0x4809 6074
MCBSPLP_XCERH_REG	RW	32	0x0000 0078	0x4809 6078
MCBSPLP_REV_REG	R	32	0x0000 007C	0x4809 607C
MCBSPLP_RINTCLR_REG	RW	32	0x0000 0080	0x4809 6080
MCBSPLP_XINTCLR_REG	RW	32	0x0000 0084	0x4809 6084
MCBSPLP_ROVFLCLR_REG	RW	32	0x0000 0088	0x4809 6088
MCBSPLP_SYSCONFIG_REG	RW	32	0x0000 008C	0x4809 608C
MCBSPLP_THRSH2_REG	RW	32	0x0000 0090	0x4809 6090
MCBSPLP_THRSH1_REG	RW	32	0x0000 0094	0x4809 6094
MCBSPLP_IRQSTATUS_REG	RW	32	0x0000 00A0	0x4809 60A0
MCBSPLP_IRQENABLE_REG	RW	32	0x0000 00A4	0x4809 60A4
MCBSPLP_WAKEUPEN_REG	RW	32	0x0000 00A8	0x4809 60A8
MCBSPLP_XCCR_REG	RW	32	0x0000 00AC	0x4809 60AC
MCBSPLP_RCCR_REG	RW	32	0x0000 00B0	0x4809 60B0
MCBSPLP_XBUFFSTAT_REG	R	32	0x0000 00B4	0x4809 60B4
MCBSPLP_RBUFFSTAT_REG	R	32	0x0000 00B8	0x4809 60B8
MCBSPLP_SSELCR_REG	RW	32	0x0000 00BC	0x4809 60BC
MCBSPLP_STATUS_REG	R	32	0x0000 00C0	0x4809 60C0

Table 21-40. McBSP2 Registers Mapping Summary

Register Name	Type	Register Width (Bits)	Address Offset	Physical Address
MCBSPLP_DRR_REG	R	32	0x0000 0000	0x4902 2000
MCBSPLP_DXR_REG	W	32	0x0000 0008	0x4902 2008
MCBSPLP_SPCR2_REG	RW	32	0x0000 0010	0x4902 2010
MCBSPLP_SPCR1_REG	RW	32	0x0000 0014	0x4902 2014
MCBSPLP_RCR2_REG	RW	32	0x0000 0018	0x4902 2018
MCBSPLP_RCR1_REG	RW	32	0x0000 001C	0x4902 201C
MCBSPLP_XCR2_REG	RW	32	0x0000 0020	0x4902 2020
MCBSPLP_XCR1_REG	RW	32	0x0000 0024	0x4902 2024
MCBSPLP_SRGR2_REG	RW	32	0x0000 0028	0x4902 2028
MCBSPLP_SRGR1_REG	RW	32	0x0000 002C	0x4902 202C
MCBSPLP_MCR2_REG	RW	32	0x0000 0030	0x4902 2030
MCBSPLP_MCR1_REG	RW	32	0x0000 0034	0x4902 2034
MCBSPLP_RCERA_REG	RW	32	0x0000 0038	0x4902 2038
MCBSPLP_RCERB_REG	RW	32	0x0000 003C	0x4902 203C
MCBSPLP_XCERA_REG	RW	32	0x0000 0040	0x4902 2040
MCBSPLP_XCERB_REG	RW	32	0x0000 0044	0x4902 2044
MCBSPLP_PCR_REG	RW	32	0x0000 0048	0x4902 2048
MCBSPLP_RCERC_REG	RW	32	0x0000 004C	0x4902 204C
MCBSPLP_RCERD_REG	RW	32	0x0000 0050	0x4902 2050
MCBSPLP_XCERC_REG	RW	32	0x0000 0054	0x4902 2054

Table 21-40. McBSP2 Registers Mapping Summary (continued)

Register Name	Type	Register Width (Bits)	Address Offset	Physical Address
MCBSPLP_XCERD_REG	RW	32	0x0000 0058	0x4902 2058
MCBSPLP_RCERE_REG	RW	32	0x0000 005C	0x4902 205C
MCBSPLP_RCERF_REG	RW	32	0x0000 0060	0x4902 2060
MCBSPLP_XCERE_REG	RW	32	0x0000 0064	0x4902 2064
MCBSPLP_XCERF_REG	RW	32	0x0000 0068	0x4902 2068
MCBSPLP_RCERG_REG	RW	32	0x0000 006C	0x4902 206C
MCBSPLP_RCERH_REG	RW	32	0x0000 0070	0x4902 2070
MCBSPLP_XCERG_REG	RW	32	0x0000 0074	0x4902 2074
MCBSPLP_XCERH_REG	RW	32	0x0000 0078	0x4902 2078
MCBSPLP_REV_REG	R	32	0x0000 007C	0x4902 207C
MCBSPLP_RINTCLR_REG	RW	32	0x0000 0080	0x4902 2080
MCBSPLP_XINTCLR_REG	RW	32	0x0000 0084	0x4902 2084
MCBSPLP_ROVFLCLR_REG	RW	32	0x0000 0088	0x4902 2088
MCBSPLP_SYSCONFIG_REG	RW	32	0x0000 008C	0x4902 208C
MCBSPLP_THRSH2_REG	RW	32	0x0000 0090	0x4902 2090
MCBSPLP_THRSH1_REG	RW	32	0x0000 0094	0x4902 2094
MCBSPLP_IRQSTATUS_REG	RW	32	0x0000 00A0	0x4902 20A0
MCBSPLP_IRQENABLE_REG	RW	32	0x0000 00A4	0x4902 20A4
MCBSPLP_WAKEUPEN_REG	RW	32	0x0000 00A8	0x4902 20A8
MCBSPLP_XCCR_REG	RW	32	0x0000 00AC	0x4902 20AC
MCBSPLP_RCCR_REG	RW	32	0x0000 00B0	0x4902 20B0
MCBSPLP_XBUFFSTAT_REG	R	32	0x0000 00B4	0x4902 20B4
MCBSPLP_RBUFFSTAT_REG	R	32	0x0000 00B8	0x4902 20B8
MCBSPLP_SSELCR_REG	RW	32	0x0000 00BC	0x4902 20BC
MCBSPLP_STATUS_REG	R	32	0x0000 00C0	0x4902 20C0

Table 21-41. McBSP3 Registers Mapping Summary

Register Name	Type	Register Width (Bits)	Address Offset	Physical Address
MCBSPLP_DRR_REG	R	32	0x0000 0000	0x4902 4000
MCBSPLP_DXR_REG	W	32	0x0000 0008	0x4902 4008
MCBSPLP_SPCR2_REG	RW	32	0x0000 0010	0x4902 4010
MCBSPLP_SPCR1_REG	RW	32	0x0000 0014	0x4902 4014
MCBSPLP_RCR2_REG	RW	32	0x0000 0018	0x4902 4018
MCBSPLP_RCR1_REG	RW	32	0x0000 001C	0x4902 401C
MCBSPLP_XCR2_REG	RW	32	0x0000 0020	0x4902 4020
MCBSPLP_XCR1_REG	RW	32	0x0000 0024	0x4902 4024
MCBSPLP_SRGR2_REG	RW	32	0x0000 0028	0x4902 4028
MCBSPLP_SRGR1_REG	RW	32	0x0000 002C	0x4902 402C
MCBSPLP_MCR2_REG	RW	32	0x0000 0030	0x4902 4030
MCBSPLP_MCR1_REG	RW	32	0x0000 0034	0x4902 4034
MCBSPLP_RCERA_REG	RW	32	0x0000 0038	0x4902 4038
MCBSPLP_RCERB_REG	RW	32	0x0000 003C	0x4902 403C
MCBSPLP_XCERA_REG	RW	32	0x0000 0040	0x4902 4040
MCBSPLP_XCERB_REG	RW	32	0x0000 0044	0x4902 4044
MCBSPLP_PCR_REG	RW	32	0x0000 0048	0x4902 4048
MCBSPLP_RCERC_REG	RW	32	0x0000 004C	0x4902 404C

Table 21-41. McBSP3 Registers Mapping Summary (continued)

Register Name	Type	Register Width (Bits)	Address Offset	Physical Address
MCBSPLP_RCERD_REG	RW	32	0x0000 0050	0x4902 4050
MCBSPLP_XCERC_REG	RW	32	0x0000 0054	0x4902 4054
MCBSPLP_XCERD_REG	RW	32	0x0000 0058	0x4902 4058
MCBSPLP_RCERE_REG	RW	32	0x0000 005C	0x4902 405C
MCBSPLP_RCERF_REG	RW	32	0x0000 0060	0x4902 4060
MCBSPLP_XCERE_REG	RW	32	0x0000 0064	0x4902 4064
MCBSPLP_XCERF_REG	RW	32	0x0000 0068	0x4902 4068
MCBSPLP_RCERG_REG	RW	32	0x0000 006C	0x4902 406C
MCBSPLP_RCERH_REG	RW	32	0x0000 0070	0x4902 4070
MCBSPLP_XCERG_REG	RW	32	0x0000 0074	0x4902 4074
MCBSPLP_XCERH_REG	RW	32	0x0000 0078	0x4902 4078
MCBSPLP_REV_REG	R	32	0x0000 007C	0x4902 407C
MCBSPLP_RINTCLR_REG	RW	32	0x0000 0080	0x4902 4080
MCBSPLP_XINTCLR_REG	RW	32	0x0000 0084	0x4902 4084
MCBSPLP_ROVFLCLR_REG	RW	32	0x0000 0088	0x4902 4088
MCBSPLP_SYSCONFIG_REG	RW	32	0x0000 008C	0x4902 408C
MCBSPLP_THRSH2_REG	RW	32	0x0000 0090	0x4902 4090
MCBSPLP_THRSH1_REG	RW	32	0x0000 0094	0x4902 4094
MCBSPLP_IRQSTATUS_REG	RW	32	0x0000 00A0	0x4902 40A0
MCBSPLP_IRQENABLE_REG	RW	32	0x0000 00A4	0x4902 40A4
MCBSPLP_WAKEUPEN_REG	RW	32	0x0000 00A8	0x4902 40A8
MCBSPLP_XCCR_REG	RW	32	0x0000 00AC	0x4902 40AC
MCBSPLP_RCCR_REG	RW	32	0x0000 00B0	0x4902 40B0
MCBSPLP_XBUFFSTAT_REG	R	32	0x0000 00B4	0x4902 40B4
MCBSPLP_RBUFFSTAT_REG	R	32	0x0000 00B8	0x4902 40B8
MCBSPLP_SSELCR_REG	RW	32	0x0000 00BC	0x4902 40BC
MCBSPLP_STATUS_REG	R	32	0x0000 00C0	0x4902 40C0

Table 21-42. McBSP4 Registers Mapping Summary

Register Name	Type	Register Width (Bits)	Address Offset	Physical Address
MCBSPLP_DRR_REG	R	32	0x0000 0000	0x4902 6000
MCBSPLP_DXR_REG	W	32	0x0000 0008	0x4902 6008
MCBSPLP_SPCR2_REG	RW	32	0x0000 0010	0x4902 6010
MCBSPLP_SPCR1_REG	RW	32	0x0000 0014	0x4902 6014
MCBSPLP_RCR2_REG	RW	32	0x0000 0018	0x4902 6018
MCBSPLP_RCR1_REG	RW	32	0x0000 001C	0x4902 601C
MCBSPLP_XCR2_REG	RW	32	0x0000 0020	0x4902 6020
MCBSPLP_XCR1_REG	RW	32	0x0000 0024	0x4902 6024
MCBSPLP_SRGR2_REG	RW	32	0x0000 0028	0x4902 6028
MCBSPLP_SRGR1_REG	RW	32	0x0000 002C	0x4902 602C
MCBSPLP_MCR2_REG	RW	32	0x0000 0030	0x4902 6030
MCBSPLP_MCR1_REG	RW	32	0x0000 0034	0x4902 6034
MCBSPLP_RCERA_REG	RW	32	0x0000 0038	0x4902 6038
MCBSPLP_RCERB_REG	RW	32	0x0000 003C	0x4902 603C
MCBSPLP_XCERA_REG	RW	32	0x0000 0040	0x4902 6040
MCBSPLP_XCERB_REG	RW	32	0x0000 0044	0x4902 6044

Table 21-42. McBSP4 Registers Mapping Summary (continued)

Register Name	Type	Register Width (Bits)	Address Offset	Physical Address
MCBSPLP_PCR_REG	RW	32	0x0000 0048	0x4902 6048
MCBSPLP_RCERC_REG	RW	32	0x0000 004C	0x4902 604C
MCBSPLP_RCERD_REG	RW	32	0x0000 0050	0x4902 6050
MCBSPLP_XCERC_REG	RW	32	0x0000 0054	0x4902 6054
MCBSPLP_XCERD_REG	RW	32	0x0000 0058	0x4902 6058
MCBSPLP_RCERE_REG	RW	32	0x0000 005C	0x4902 605C
MCBSPLP_RCERF_REG	RW	32	0x0000 0060	0x4902 6060
MCBSPLP_XCERE_REG	RW	32	0x0000 0064	0x4902 6064
MCBSPLP_XCERF_REG	RW	32	0x0000 0068	0x4902 6068
MCBSPLP_RCERG_REG	RW	32	0x0000 006C	0x4902 606C
MCBSPLP_RCERH_REG	RW	32	0x0000 0070	0x4902 6070
MCBSPLP_XCERG_REG	RW	32	0x0000 0074	0x4902 6074
MCBSPLP_XCERH_REG	RW	32	0x0000 0078	0x4902 6078
MCBSPLP_REV_REG	R	32	0x0000 007C	0x4902 607C
MCBSPLP_RINTCLR_REG	RW	32	0x0000 0080	0x4902 6080
MCBSPLP_XINTCLR_REG	RW	32	0x0000 0084	0x4902 6084
MCBSPLP_ROVFLCLR_REG	RW	32	0x0000 0088	0x4902 6088
MCBSPLP_SYSCONFIG_REG	RW	32	0x0000 008C	0x4902 608C
MCBSPLP_THRSH2_REG	RW	32	0x0000 0090	0x4902 6090
MCBSPLP_THRSH1_REG	RW	32	0x0000 0094	0x4902 6094
MCBSPLP_IRQSTATUS_REG	RW	32	0x0000 00A0	0x4902 60A0
MCBSPLP_IRQENABLE_REG	RW	32	0x0000 00A4	0x4902 60A4
MCBSPLP_WAKEUPEN_REG	RW	32	0x0000 00A8	0x4902 60A8
MCBSPLP_XCCR_REG	RW	32	0x0000 00AC	0x4902 60AC
MCBSPLP_RCCR_REG	RW	32	0x0000 00B0	0x4902 60B0
MCBSPLP_XBUFFSTAT_REG	R	32	0x0000 00B4	0x4902 60B4
MCBSPLP_RBUFFSTAT_REG	R	32	0x0000 00B8	0x4902 60B8
MCBSPLP_SSELCR_REG	RW	32	0x0000 00BC	0x4902 60BC
MCBSPLP_STATUS_REG	R	32	0x0000 00C0	0x4902 60C0

21.7.2 SIDETONE Register Mapping Summary

Table 21-43. SIDETONE_McBSP2 Registers Mapping Summary

Register Name	Type	Register Width (Bits)	Address Offset	Physical Address
ST_REV_REG	R	32	0x0000 0000	0x4902 8000
ST_SYSCONFIG_REG	RW	32	0x0000 0010	0x4902 8010
ST_IRQSTATUS_REG	RW	32	0x0000 0018	0x4902 8018
ST_IRQENABLE_REG	RW	32	0x0000 001C	0x4902 801C
ST_SGAINCR_REG	RW	32	0x0000 0024	0x4902 8024
ST_SFIRCR_REG	RW	32	0x0000 0028	0x4902 8028
ST_SSELCR_REG	RW	32	0x0000 002C	0x4902 802C

Table 21-44. SIDETONE_McBSP3 Registers Mapping Summary

Register Name	Type	Register Width (Bits)	Address Offset	Physical Address
ST_REV_REG	R	32	0x0000 0000	0x4902 A000
ST_SYSCONFIG_REG	RW	32	0x0000 0010	0x4902 A010

Table 21-44. SIDETONE_McBSP3 Registers Mapping Summary (continued)

Register Name	Type	Register Width (Bits)	Address Offset	Physical Address
ST_IRQSTATUS_REG	RW	32	0x0000 0018	0x4902 A018
ST_IRQENABLE_REG	RW	32	0x0000 001C	0x4902 A01C
ST_SGAINCR_REG	RW	32	0x0000 0024	0x4902 A024
ST_SFIRCR_REG	RW	32	0x0000 0028	0x4902 A028
ST_SSELCR_REG	RW	32	0x0000 002C	0x4902 A02C

21.7.3 McBSP Register Description

Table 21-45. MCBSPLP_DRR_REG

Address Offset	0x0000 0000		
Physical Address	0x4807 4000	**Instance**	McBSP1
	0x4809 6000		McBSP5
	0x4902 2000		McBSP2
	0x4902 4000		McBSP3
	0x4902 6000		McBSP4
Description	McBSPLP data receive register		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DRR																															

Bits	Field Name	Description	Type	Reset
31:0	DRR	Data receive register	R	0x00000000

Table 21-46. Register Call Summary for Register MCBSPLP_DRR_REG

McBSP Functional Description
- Data Transfer Process for 8- / 12- / 16- / 20- / 24- / 32-bits Long Words: [0]
- Clocking and Framing Data: [1]
- McBSP Reception: [2] [3]
- Introduction: [4]
- Overrun in the Receiver: [5] [6]

McBSP Basic Programming Model
- Receiver Configuration: [7] [8]
- Data Packing Examples: [9] [10]

McBSP Use Case and Tips
- Programming Flow: [11] [12]

McBSP Register Manual
- McBSP Register Mapping Summary: [13] [14] [15] [16] [17]

Table 21-47. MCBSPLP_DXR_REG

Address Offset	0x0000 0008		
Physical Address	0x4807 4008	**Instance**	McBSP1
	0x4809 6008		McBSP5
	0x4902 2008		McBSP2
	0x4902 4008		McBSP3
	0x4902 6008		McBSP4
Description	McBSPLP data transmit register		
Type	W		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DXR																															

Bits	Field Name	Description	Type	Reset
31:0	DXR	Data transmit register	W	0x00000000

Table 21-48. Register Call Summary for Register MCBSPLP_DXR_REG

McBSP Functional Description

- Data Transfer Process for 8- / 12- / 16- / 20- / 24- / 32-bits Long Words: [0]
- McBSP Transmission: [1] [2]
- Introduction: [3]
- Underflow in the Transmitter: [4] [5] [6] [7]
- Transmit Multichannel Selection Modes: [8]

McBSP Basic Programming Model

- McBSP Initialization Procedure: [9]
- Data Packing Examples: [10] [11]

McBSP Register Manual

- McBSP Register Mapping Summary: [12] [13] [14] [15] [16]

Table 21-49. MCBSPLP_SPCR2_REG

Address Offset	0x0000 0010		
Physical Address	0x4807 4010	**Instance**	McBSP1
	0x4809 6010		McBSP5
	0x4902 2010		McBSP2
	0x4902 4010		McBSP3
	0x4902 6010		McBSP4
Description	McBSPLP serial port control register 2		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																						FREE	SOFT	FRST	GRST	XINTM	XSYNCERR	XEMPTY	XRDY	XRST	

Bits	Field Name	Description	Type	Reset
31:10	RESERVED	Read returns 0x0.	R	0x000000
9	FREE	Free Running Mode (When this bit is set, the module ignores the Msuspend input) 0x0: Free running mode is disabled 0x1: Free running mode is enabled	RW	0x0

Bits	Field Name	Description	Type	Reset
8	SOFT	Soft Bit 0x0: SOFT Mode is disabled: the McBSP module stops its activity immediately following MSuspend assertion. 0x1: SOFT Mode is enabled: the McBSP module freezes its state after completion of the current operation when MSuspend is asserted.	RW	0x0
7	FRST	Frame-Sync Generator Reset 0x0: Frame-synchronization logic is reset. Frame-sync signal FSG is not generated by the sample-rate generator 0x1: Frame-sync signal FSG is generated after (FPER+1) number of CLKG clocks; i.e., all frame counters are loaded with their programmed values	RW	0x0
6	GRST	Sample-Rate Generator Reset 0x0: SRG is reset 0x1: SRG is pulled out of reset. CLKG is driven as per programmed value in SRG registers (SRGR[1,2])	RW	0x0
5:4	XINTM	Transmit Interrupt Mode (legacy) 0x0: Transmit interrupt is driven by XRDY 0x1: Transmit interrupt generated by end-of-frame 0x2: Transmit interrupt generated by a new frame synchronization 0x3: Transmit interrupt generated by XSYNCERR	RW	0x0
3	XSYNCERR	Transmit Synchronization Error (writing 0 to this bit clear the legacy transmit interrupt if asserted due to XSYNCERROR condition) 0x0: No synchronization error 0x1: Synchronization error detected by McBSP	RW	0x0
2	XEMPTY	Transmit Shift Register XSR Empty 0x0: XSR is empty 0x1: XSR is not empty	R	0x0
1	XRDY	Transmitter ready 0x0: Transmitter is not ready. 0x1: Transmitter is ready for new data in DXR	R	0x0
0	XRST	Transmitter reset. This resets and enables the transmitter. 0x0: The serial port transmitter is disabled and in reset state. 0x1: The serial port transmitter is enabled.	RW	0x0

Table 21-50. Register Call Summary for Register MCBSPLP_SPCR2_REG

McBSP Integration
- Hardware and Software Reset: [0] [1] [2]

McBSP Functional Description
- Data Transfer Process for 8- / 12- / 16- / 20- / 24- / 32-bits Long Words: [3]
- McBSP Transmission: [4] [5]
- Clock Generation in the SRG: [6]
- Frame Sync Generation in the SRG: [7] [8]
- Introduction: [9] [10] [11]
- Underflow in the Transmitter: [12] [13] [14] [15] [16]
- Unexpected Transmit Frame-sync Pulse: [17] [18] [19]
- McBSP DMA Configuration: [20] [21]
- Transmit Multichannel Selection Modes: [22] [23] [24]

Table 21-50. Register Call Summary for Register MCBSPLP_SPCR2_REG (continued)

McBSP Basic Programming Model

- McBSP Initialization Procedure: [25] [26] [27] [28]
- Reset and Initialization Procedure for the Sample Rate Generator: [29] [30] [31] [32] [33]
- Interrupt Configuration: [34] [35] [36]
- Receiver Configuration: [37] [38] [39] [40] [41]
- Transmitter Configuration: [42] [43] [44] [45] [46] [47] [48]
- General-Purpose I/O on the McBSP Pins (Legacy Only): [49] [50]

McBSP Use Case and Tips

- Programming Flow: [51] [52]

McBSP Register Manual

- McBSP Register Mapping Summary: [53] [54] [55] [56] [57]

Table 21-51. MCBSPLP_SPCR1_REG

Address Offset	0x0000 0014		
Physical Address	0x4807 4014	**Instance**	McBSP1
	0x4809 6014		McBSP5
	0x4902 2014		McBSP2
	0x4902 4014		McBSP3
	0x4902 6014		McBSP4
Description	McBSPLP serial port control register 1		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																ALB	RJUST	RESERVED						DXENA	RESERVED	RINTM		RSYNCERR	RFULL	RRDY	RRST

Bits	Field Name	Description	Type	Reset
31:16	RESERVED	Read returns 0x0.	R	0x0000
15	ALB	Analog loopback Mode	RW	0x0
		0x0: Analog loopback mode disabled		
		0x1: Analog loopback mode enabled		
14:13	RJUST	Receive Sign-Extension and Justification Mode	RW	0x0
		0x0: Right-justify and zero-fill MSBs in DRR		
		0x1: Right-justify and sign-extend MSBs in DRR		
		0x2: Left-justify and zero-fill LSBs in DRR		
		0x3: Reserved		
12:8	RESERVED	Read returns 0x0.	R	0x00
7	DXENA	DX Enabler	RW	0x0
		0x0: DX enabler is off		
		0x1: DX enabler is on		
6	RESERVED	Read returns 0x0.	R	0x0

Bits	Field Name	Description	Type	Reset
5:4	RINTM	Receive Interrupt Mode (legacy) 0x0: Receive Interrupt driven by RRDY (i.e. end of word) and end of frame in A-bis mode 0x1: Receive Interrupt generated by end-of-block or end-of-frame in multichannel operation 0x2: Receive Interrupt generated by a new frame synchronization 0x3: Receive Interrupt generated by RSYNCERR	RW	0x0
3	RSYNCERR	Receive Synchronization Error (writing 0 to this bit clear the legacy receive interrupt if asserted due to RSYNCERROR condition) 0x0: No synchronization error 0x1: Synchronization error detected by McBSP	RW	0x0
2	RFULL	Receive Shift Register (RSR) Full 0x0: DRR is not read, RB is full and RSR is also full with new word 0x1: RB is not in overrun condition	R	0x0
1	RRDY	Receiver Ready 0x0: Receiver is not ready 0x1: Receiver is ready with data to be read from DRR	R	0x0
0	RRST	Receiver reset. This resets and enables the receiver. 0x0: The serial port receiver is disabled and in reset state. 0x1: The serial port receiver is enabled.	RW	0x0

Table 21-52. Register Call Summary for Register MCBSPLP_SPCR1_REG

McBSP Integration
- Hardware and Software Reset: [0]

McBSP Functional Description
- McBSP Reception: [1] [2] [3]
- Clock Generation in the SRG: [4]
- Introduction: [5] [6] [7] [8]
- Overrun in the Receiver: [9] [10] [11]
- Unexpected Receive Frame-sync Pulse: [12] [13] [14]
- McBSP DMA Configuration: [15] [16]
- Receive Multichannel Selection Mode: [17]

McBSP Basic Programming Model
- McBSP Initialization Procedure: [18] [19]
- Reset and Initialization Procedure for the Sample Rate Generator: [20] [21] [22]
- Interrupt Configuration: [23] [24] [25] [26]
- Receiver Configuration: [27] [28] [29] [30] [31] [32] [33]
- Transmitter Configuration: [34] [35]
- General-Purpose I/O on the McBSP Pins (Legacy Only): [36] [37]

McBSP Use Case and Tips
- Programming Flow: [38] [39] [40] [41]

McBSP Register Manual
- McBSP Register Mapping Summary: [42] [43] [44] [45] [46]
- McBSP Register Description: [47]

Table 21-53. MCBSPLP_RCR2_REG

Address Offset	0x0000 0018		
Physical Address	0x4807 4018	**Instance**	McBSP1
	0x4809 6018		McBSP5
	0x4902 2018		McBSP2
	0x4902 4018		McBSP3
	0x4902 6018		McBSP4
Description	McBSPLP receive control register 2		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																RPHASE	RFRLEN2							RWDLEN2			RREVERSE		RESERVED		RDATDLY

Bits	Field Name	Description	Type	Reset
31:16	RESERVED	Read returns 0x0.	R	0x0000
15	RPHASE	Receive Phases 0x0: Single-phase frame 0x1: Dual-phase frame	RW	0x0
14:8	RFRLEN2	Receive Frame Length 2 Single-phase frame selected: RFRLEN2=don't care Dual-phase frame selected: RFRLEN2=000 0000 - 1 word per second phase (other values are reserved)	RW	0x00
7:5	RWDLEN2	Receive Word Length 2 0x0: 8 bits 0x1: 12 bits 0x2: 16 bits 0x3: 20 bits 0x4: 24 bits 0x5: 32 bits 0x6: Reserved (do not use) 0x7: Reserved (do not use)	RW	0x0
4:3	RREVERSE	Receive reverse mode. 0x0: Data transfer starts with MSB first. 0x1: Data transfer starts with LSB first. 0x2: Reserved (do not use) 0x3: Reserved (do not use)	RW	0x0
2	RESERVED	Read returns 0x0.	R	0x0
1:0	RDATDLY	Receive Data Delay 0x0: 0-bit data delay 0x1: 1-bit data delay 0x2: 2-bit data delay 0x3: Reserved	RW	0x0

Table 21-54. Register Call Summary for Register MCBSPLP_RCR2_REG

McBSP Environment

- Words, Frames, and Phases Definitions: [0] [1] [2]

Table 21-54. Register Call Summary for Register MCBSPLP_RCR2_REG (continued)

McBSP Functional Description

- Bit Reordering (Option to Transfer LSB First): [3]
- Clocking and Framing Data: [4] [5]
- Frame Phases (Dual-Phase Frame I2S Support): [6] [7] [8] [9] [10] [11] [12] [13] [14] [15]
- McBSP Reception: [16]
- MCBSP Data Transfer Mode: [17]
- Unexpected Receive Frame-sync Pulse: [18]
- Configuring a Frame for Multichannel Selection: [19]
- SIDETONE Interface: [20]

McBSP Basic Programming Model

- McBSP Initialization Procedure: [21]
- Receiver Configuration: [22] [23] [24] [25] [26] [27]
- SIDETONE Activation Procedure: [28]

McBSP Use Case and Tips

- Programming Flow: [29] [30] [31]

McBSP Register Manual

- McBSP Register Mapping Summary: [32] [33] [34] [35] [36]

Table 21-55. MCBSPLP_RCR1_REG

Address Offset	0x0000 001C		
Physical Address	0x4807 401C	**Instance**	McBSP1
	0x4809 601C		McBSP5
	0x4902 201C		McBSP2
	0x4902 401C		McBSP3
	0x4902 601C		McBSP4
Description	McBSPLP receive control register 1		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																RFRLEN1								RWDLEN1			RESERVED				

Bits	Field Name	Description	Type	Reset
31:15	RESERVED	Read returns 0x0.	R	0x00000
14:8	RFRLEN1	Receive Frame Length 1 Single-phase frame selected: RFRLEN1=000 0000 - 1 word per frame RFRLEN1=000 0001 - 2 words per frame RFRLEN1=111 1111 - 128 words per frame Dual-phase frame selected: RFRLEN1=000 0000 - 1 word per phase (other values are reserved)	RW	0x00
7:5	RWDLEN1	Receive Word Length 1 0x0: 8 bits 0x1: 12 bits 0x2: 16 bits 0x3: 20 bits 0x4: 24 bits 0x5: 32 bits 0x6: Reserved (do not use) 0x7: Reserved (do not use)	RW	0x0
4:0	RESERVED	Read returns 0x0.	R	0x00

Table 21-56. Register Call Summary for Register MCBSPLP_RCR1_REG

McBSP Environment
- Words, Frames, and Phases Definitions: [0] [1]

McBSP Functional Description
- Clocking and Framing Data: [2] [3]
- Frame Phases (Dual-Phase Frame I2S Support): [4] [5] [6] [7] [8] [9] [10]
- Configuring a Frame for Multichannel Selection: [11]
- SIDETONE Interface: [12] [13]

McBSP Basic Programming Model
- McBSP Initialization Procedure: [14]
- Receiver Configuration: [15] [16]
- SIDETONE Activation Procedure: [17] [18]

McBSP Use Case and Tips
- Programming Flow: [19] [20] [21]

McBSP Register Manual
- McBSP Register Mapping Summary: [22] [23] [24] [25] [26]

Table 21-57. MCBSPLP_XCR2_REG

Address Offset	0x0000 0020		
Physical Address	0x4807 4020	**Instance**	McBSP1
	0x4809 6020		McBSP5
	0x4902 2020		McBSP2
	0x4902 4020		McBSP3
	0x4902 6020		McBSP4
Description	McBSPLP transmit control register 2		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																XPHASE	XFRLEN2							XWDLEN2			XREVERSE		RESERVED		XDATDLY

Bits	Field Name	Description	Type	Reset
31:16	RESERVED	Read returns 0x0.	R	0x0000
15	XPHASE	Transmit Phases 0x0: Single-phase frame 0x1: Dual-phase frame	RW	0x0
14:8	XFRLEN2	Transmit Frame Length 2 Single-phase frame selected: XFRLEN2=don't care Dual-phase frame selected: XFRLEN2=000 0000 - 1 word per second phase (other values are reserved)	RW	0x00
7:5	XWDLEN2	Transmit Word Length 2 0x0: 8 bits 0x1: 12 bits 0x2: 16 bits 0x3: 20 bits 0x4: 24 bits 0x5: 32 bits 0x6: Reserved (do not use) 0x7: Reserved (do not use)	RW	0x0

Bits	Field Name	Description	Type	Reset
4:3	XREVERSE	Transmit reverse mode. 0x0: Data transfer starts with MSB first. 0x1: Data transfer starts with LSB first. 0x2: Reserved (do not use) 0x3: Reserved (do not use)	RW	0x0
2	RESERVED	Read returns 0x0.	R	0x0
1:0	XDATDLY	Transmit Data Delay 0x0: 0-bit data delay 0x1: 1-bit data delay 0x2: 2-bit data delay 0x3: Reserved	RW	0x0

Table 21-58. Register Call Summary for Register MCBSPLP_XCR2_REG

McBSP Environment
- Words, Frames, and Phases Definitions: [0] [1] [2]

McBSP Functional Description
- Bit Reordering (Option to Transfer LSB First): [3]
- Clocking and Framing Data: [4] [5] [6]
- Frame Phases (Dual-Phase Frame I2S Support): [7] [8] [9] [10] [11] [12] [13] [14] [15] [16]
- McBSP Transmission: [17]
- MCBSP Data Transfer Mode: [18]
- Unexpected Transmit Frame-sync Pulse: [19]
- Configuring a Frame for Multichannel Selection: [20]
- SIDETONE Interface: [21]

McBSP Basic Programming Model
- McBSP Initialization Procedure: [22]
- Transmitter Configuration: [23] [24] [25] [26] [27]
- SIDETONE Activation Procedure: [28]

McBSP Register Manual
- McBSP Register Mapping Summary: [29] [30] [31] [32] [33]

Table 21-59. MCBSPLP_XCR1_REG

Address Offset	0x0000 0024		
Physical Address	0x4807 4024	**Instance**	McBSP1
	0x4809 6024		McBSP5
	0x4902 2024		McBSP2
	0x4902 4024		McBSP3
	0x4902 6024		McBSP4
Description	McBSPLP transmit control register 1		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																XFRLEN1								XWDLEN1			RESERVED				

Bits	Field Name	Description	Type	Reset
31:15	RESERVED	Read returns 0x0.	R	0x00000
14:8	XFRLEN1	Transmit Frame Length 1 Single-phase frame selected: XFRLEN1=000 0000 - 1 word per frame XFRLEN1=000 0001 - 2 words per frame XFRLEN1=111 1111 - 128 words per frame Dual-phase frame selected: XFRLEN1=000 0000 - 1 word per phase (other values are reserved)	RW	0x00
7:5	XWDLEN1	Transmit Word Length 1 0x0: 8 bits 0x1: 12 bits 0x2: 16 bits 0x3: 20 bits 0x4: 24 bits 0x5: 32 bits 0x6: Reserved (do not use) 0x7: Reserved (do not use)	RW	0x0
4:0	RESERVED	Read returns 0x0.	R	0x00

Table 21-60. Register Call Summary for Register MCBSPLP_XCR1_REG

McBSP Environment
- Words, Frames, and Phases Definitions: [0] [1]

McBSP Functional Description
- Clocking and Framing Data: [2] [3]
- Frame Phases (Dual-Phase Frame I2S Support): [4] [5] [6] [7] [8] [9] [10]
- Configuring a Frame for Multichannel Selection: [11]
- SIDETONE Interface: [12] [13]

McBSP Basic Programming Model
- McBSP Initialization Procedure: [14]
- Transmitter Configuration: [15] [16]
- SIDETONE Activation Procedure: [17] [18]

McBSP Register Manual
- McBSP Register Mapping Summary: [19] [20] [21] [22] [23]

Table 21-61. MCBSPLP_SRGR2_REG

Address Offset	0x0000 0028		
Physical Address	0x4807 4028	**Instance**	McBSP1
	0x4809 6028		McBSP5
	0x4902 2028		McBSP2
	0x4902 4028		McBSP3
	0x4902 6028		McBSP4
Description	McBSPLP SRG register 2		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																GSYNC	CLKSP	CLKSM	FSGM	FPER											

Bits	Field Name	Description	Type	Reset
31:16	RESERVED	Read returns 0x0.	R	0x0000
15	GSYNC	Sample Rate Generator Synchronization Only used when the external clock (CLKS) drives the SRG clock (CLKSM=0) 0x0: The SRG clock (CLKG) is free running. 0x1: The SRG clock (CLKG) is running. But CLKG is resynchronized and frame-sync signal (FSG) is generated only after detecting the receive frame-synchronization signal (FSR). Also, frame period, FPER, is a don't care because the period is dictated by the external frame-sync pulse.	RW	0x0
14	CLKSP	CLKS Polarity Clock Edge Select Only used when the external clock CLKS drives the SRG clock (CLKSM=0). 0x0: Rising edge of CLKG and FSG. 0x1: Falling edge of CLKG and FSG.	RW	0x0
13	CLKSM	McBSP SRG Clock Mode 0x0: SCLKME=0: SRG clock derived from the CLKS pin. SCLKME=1: SRG clock derived from the CLKR input pin. 0x1: SCLKME=0: SRG clock derived from the McBSPi_ICLK clock. SCLKME=1: SRG clock derived from the CLKX input pin.	RW	0x1
12	FSGM	Sample Rate Generator Transmit Frame-Synchronization Mode Used when FSXM=1 in the PCR. 0x0: Transmit frame-sync signal (FSX) is generated when transmit buffer is not empty. When FSGM=0, FPER and FWID are used to determine the frame synchronization period and width (external FSX is gated by the buffer empty condition). 0x1: Transmit frame-sync signal driven by the SRG frame-sync signal, FSG.	RW	0x0
11:0	FPER	Frame Period. This value + 1 determines when the next frame-sync signal becomes active. Range: 1 to 4096 CLKG periods	RW	0x000

Table 21-62. Register Call Summary for Register MCBSPLP_SRGR2_REG

McBSP Integration
- Clocks: [0] [1] [2] [3] [4]

McBSP Functional Description
- Clocking and Framing Data: [5] [6]
- McBSP SRG: [7] [8] [9] [10]
- Frame Sync Generation in the SRG: [11] [12] [13] [14] [15] [16] [17]
- Synchronizing SRG Outputs to an External Clock: [18] [19] [20] [21] [22] [23] [24] [25] [26]
- Underflow in the Transmitter: [27] [28] [29]

McBSP Basic Programming Model
- McBSP Initialization Procedure: [30]
- Reset and Initialization Procedure for the Sample Rate Generator: [31]
- Receiver Configuration: [32] [33] [34] [35] [36] [37]
- Transmitter Configuration: [38] [39]

McBSP Register Manual
- McBSP Register Mapping Summary: [40] [41] [42] [43] [44]

Table 21-63. MCBSPLP_SRGR1_REG

Address Offset	0x0000 002C		
Physical Address	0x4807 402C	**Instance**	McBSP1
	0x4809 602C		McBSP5
	0x4902 202C		McBSP2
	0x4902 402C		McBSP3
	0x4902 602C		McBSP4
Description	McBSPLP SRG register 1		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																FWID								CLKGDV							

Bits	Field Name	Description	Type	Reset
31:16	RESERVED	Read returns 0x0.	R	0x0000
15:8	FWID	Frame Width. This value + 1 determines the width of the frame-sync pulse, FSG, during its active period. Range: 1 to 256 CLKG periods.	RW	0x00
7:0	CLKGDV	Sample Rate Generator Clock Divider This value is used as the divide-down number to generate the required SRG clock frequency. Default value is 1.	RW	0x01

Table 21-64. Register Call Summary for Register MCBSPLP_SRGR1_REG

McBSP Functional Description
- Clocking and Framing Data: [0]
- McBSP SRG: [1] [2] [3]
- Frame Sync Generation in the SRG: [4] [5]
- Synchronizing SRG Outputs to an External Clock: [6] [7]
- Underflow in the Transmitter: [8]

McBSP Basic Programming Model
- McBSP Initialization Procedure: [9]
- Receiver Configuration: [10] [11]

McBSP Register Manual
- McBSP Register Mapping Summary: [12] [13] [14] [15] [16]

Table 21-65. MCBSPLP_MCR2_REG

Address Offset	0x0000 0030		
Physical Address	0x4807 4030	**Instance**	McBSP1
	0x4809 6030		McBSP5
	0x4902 2030		McBSP2
	0x4902 4030		McBSP3
	0x4902 6030		McBSP4
Description	McBSPLP multi channel register 2		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																						XMCME	XPBBLK	XPABLK	RESERVED						XMCM

Bits	Field Name	Description	Type	Reset
31:10	RESERVED	Read returns 0x0.	R	0x000000
9	XMCME	Transmit multichannel partition mode determines whether only 32 channels or all 128 channels are to be individually selectable. XMCME is only applicable if channels can be individually disabled/enabled or masked/unmasked for transmission (XMCM is nonzero). 0x0: 2-partition mode: Only partitions A and B are used. You can control up to 32 channels in the transmit multichannel selection mode selected with the XMCM bits. If XMCM = 01b or 10b, assign 16 channels to partition A with the XPABLK bits. Assign 16 channels to partition B with the XPBBLK bits. If XMCM = 11b (for symmetric transmission and reception), assign 16 channels to receive partition A with the RPABLK bits. Assign 16 channels to receive partition B with the RPBBLK bits. You control the channels with the appropriate transmit channel enable registers: XCERA: Channels in partition A XCERB: Channels in partition B 0x1: 8-partition mode: All partitions (A through H) are used. You can control up to 128 channels in the transmit multichannel selection mode selected with the XMCM bits. You control the channels with the appropriate transmit channel enable registers: XCERA: Channels 0 through 15 XCERB: Channels 16 through 31 XCERC: Channels 32 through 47 XCERD: Channels 48 through 63 XCERE: Channels 64 through 79 XCERF: Channels 80 through 95 XCERG: Channels 96 through 111 XCERH: Channels 112 through 127	RW	0x0
8:7	XPBBLK	Transmit Partition B Block (legacy) 0x0: Block 1. Channel 16 to channel 31 0x1: Block 3. Channel 48 to channel 63 0x2: Block 5. Channel 80 to channel 95 0x3: Block 7. Channel 112 to channel 127	RW	0x0
6:5	XPABLK	Transmit Partition A Block (legacy) 0x0: Block 0. Channel 0 to channel 15 0x1: Block 2. Channel 32 to channel 47 0x2: Block 4. Channel 64 to channel 79 0x3: Block 6. Channel 96 to channel 111	RW	0x0
4:2	RESERVED	Reserved	R	0x0

Bits	Field Name	Description	Type	Reset
1:0	XMCM	Transmit Multichannel Selection Enable 0x0: All channels enabled without masking (DX is always driven during transmission of data). 0x1: All channels disabled and therefore masked by default. Required channels are selected by enabling XP(A/B)BLK and XCER(A/B) appropriately. Also, these selected channels are not masked and therefore DX is always driven. 0x2: All channels enabled, but masked. Selected channels enabled via XP(A/B)BLK and XCER(A/B) are unmasked. 0x3: All channels disabled and therefore masked by default. Required channels are selected by enabling RP(A/B)BLK and RCER(A/B) appropriately. Selected channels can be unmasked by RP(A/B)BLK and XCER(A/B). This mode is used for symmetric transmit and receive operation.	RW	0x0

Table 21-66. Register Call Summary for Register MCBSPLP_MCR2_REG

McBSP Functional Description
- Using Eight Partitions: [0] [1] [2]
- Using Two Partitions (Legacy Only): [3] [4]
- Transmit Multichannel Selection Modes: [5] [6] [7] [8] [9] [10]

McBSP Basic Programming Model
- McBSP Initialization Procedure: [11]
- Transmitter Configuration: [12]

McBSP Register Manual
- McBSP Register Mapping Summary: [13] [14] [15] [16] [17]

Table 21-67. MCBSPLP_MCR1_REG

Address Offset	0x0000 0034		
Physical Address	0x4807 4034	**Instance**	McBSP1
	0x4809 6034		McBSP5
	0x4902 2034		McBSP2
	0x4902 4034		McBSP3
	0x4902 6034		McBSP4
Description	McBSPLP multi channel register 1		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																						RMCME	RPBBLK	RPABLK			RESERVED				RMCM

Bits	Field Name	Description	Type	Reset
31:10	RESERVED	Read returns 0x0.	R	0x000000
9	RMCME	Receive multichannel partition mode determines whether only 32 channels or all 128 channels are to be individually selectable. RMCME is only applicable if channels can be individually enabled or disabled for reception (RMCM = 1). 0x0: 2-partition mode. Only partitions A and B are used. You can control up to 32 channels in the receive multichannel selection mode (RMCM = 1). Assign 16 channels to partition A with the RPABLK bits. Assign 16 channels to partition B with the RPBBLK bits. You control the channels with the appropriate receive channel enable registers: RCERA: Channels in partition A RCERB: Channels in partition B 0x1: 8-partition mode: All partitions (A through H) are used. You can control up to 128 channels in the receive multichannel selection mode. You control the channels with the appropriate receive channel enable registers: RCERA: Channels 0 through 15 RCERB: Channels 16 through 31 RCERC: Channels 32 through 47 RCERD: Channels 48 through 63 RCERE: Channels 64 through 79 RCERF: Channels 80 through 95 RCERG: Channels 96 through 111 RCERH: Channels 112 through 127	RW	0x0
8:7	RPBBLK	Receive Partition B Block (legacy) 0x0: Block 1. Channel 16 to channel 31 0x1: Block 3. Channel 48 to channel 63 0x2: Block 5. Channel 80 to channel 95 0x3: Block 7. Channel 112 to channel 127	RW	0x0
6:5	RPABLK	Receive Partition A Block (legacy) 0x0: Block 0. Channel 0 to channel 15 0x1: Block 2. Channel 32 to channel 47 0x2: Block 4. Channel 64 to channel 79 0x3: Block 6. Channel 96 to channel 111	RW	0x0
4:1	RESERVED	Read returns 0x0.	R	0x0
0	RMCM	Receive Multichannel Selection Enable 0x0: All 128 channels 0x1: All channels disabled by default. Required channels are selected by enabling RP(A/B)BLK and RCER(A/B) appropriately	RW	0x0

Table 21-68. Register Call Summary for Register MCBSPLP_MCR1_REG

McBSP Functional Description

- Using Eight Partitions: [0] [1] [2]
- Receive Multichannel Selection Mode: [3] [4]
- Using Two Partitions (Legacy Only): [5] [6] [7] [8]
- SIDETONE Interface: [9]

McBSP Basic Programming Model

- McBSP Initialization Procedure: [10]
- Receiver Configuration: [11]
- SIDETONE Activation Procedure: [12]

Table 21-68. Register Call Summary for Register MCBSPLP_MCR1_REG (continued)

McBSP Register Manual

- McBSP Register Mapping Summary: [13] [14] [15] [16] [17]

Table 21-69. MCBSPLP_RCERA_REG

Address Offset	0x0000 0038		
Physical Address	0x4807 4038	**Instance**	McBSP1
	0x4809 6038		McBSP5
	0x4902 2038		McBSP2
	0x4902 4038		McBSP3
	0x4902 6038		McBSP4
Description	McBSPLP receive channel enable register partition A		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																RCERA															

Bits	Field Name	Description	Type	Reset
31:16	RESERVED	Read returns 0x0.	R	0x0000
15:0	RCERA	Receive Channel Enable RCERA n=0 Disables reception of n-th channel in an even-numbered block in partition A RCERA n=1 Enables reception of n-th channel in an even-numbered block in partition A	RW	0x0000

Table 21-70. Register Call Summary for Register MCBSPLP_RCERA_REG

McBSP Functional Description

- Using Eight Partitions: [0]
- Receive Multichannel Selection Mode: [1] [2]
- Using Two Partitions (Legacy Only): [3]
- Transmit Multichannel Selection Modes: [4] [5]

McBSP Register Manual

- McBSP Register Mapping Summary: [6] [7] [8] [9] [10]

Table 21-71. MCBSPLP_RCERB_REG

Address Offset	0x0000 003C		
Physical Address	0x4807 403C	**Instance**	McBSP1
	0x4809 603C		McBSP5
	0x4902 203C		McBSP2
	0x4902 403C		McBSP3
	0x4902 603C		McBSP4
Description	McBSPLP receive channel enable register partition B		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																RCERB															

Bits	Field Name	Description	Type	Reset
31:16	RESERVED	Read returns 0x0.	R	0x0000
15:0	RCERB	Receive Channel Enable RCERB n=0 Disables reception of n-th channel in a even-numbered block in partition B RCERB n=1 Enables reception of n-th channel in a even-numbered block in partition B	RW	0x0000

Table 21-72. Register Call Summary for Register MCBSPLP_RCERB_REG

McBSP Functional Description • Using Eight Partitions: [0] • Using Two Partitions (Legacy Only): [1]
McBSP Register Manual • McBSP Register Mapping Summary: [2] [3] [4] [5] [6]

Table 21-73. MCBSPLP_XCERA_REG

Address Offset	0x0000 0040		
Physical Address	0x4807 4040	**Instance**	McBSP1
	0x4809 6040		McBSP5
	0x4902 2040		McBSP2
	0x4902 4040		McBSP3
	0x4902 6040		McBSP4
Description	McBSPLP transmit channel enable register partition A		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																XCERA															

Bits	Field Name	Description	Type	Reset
31:16	RESERVED	Read returns 0x0.	R	0x0000
15:0	XCERA	Transmit Channel Enable XCERA n=0 Disables transmission of n-th channel in an event-numbered block in partition A XCERA n=1 Enables transmission of n-th channel in an event-numbered block in partition A	RW	0x0000

Table 21-74. Register Call Summary for Register MCBSPLP_XCERA_REG

McBSP Functional Description • Using Eight Partitions: [0] • Using Two Partitions (Legacy Only): [1] • Transmit Multichannel Selection Modes: [2] [3] [4] [5] [6] [7]
McBSP Register Manual • McBSP Register Mapping Summary: [8] [9] [10] [11] [12]

Table 21-75. MCBSPLP_XCERB_REG

Address Offset	0x0000 0044		
Physical Address	0x4807 4044	**Instance**	McBSP1
	0x4809 6044		McBSP5
	0x4902 2044		McBSP2
	0x4902 4044		McBSP3
	0x4902 6044		McBSP4
Description	McBSPLP transmit channel enable register partition B		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																XCERB															

Bits	Field Name	Description	Type	Reset
31:16	RESERVED	Read returns 0x0.	R	0x0000
15:0	XCERB	Transmit Channel Enable XCERB n=0 Disables transmission of n-th channel in an even-numbered block in partition B XCERB n=1 Enables transmission of n-th channel in an even-numbered block in partition B	RW	0x0000

Table 21-76. Register Call Summary for Register MCBSPLP_XCERB_REG

McBSP Functional Description

- Using Eight Partitions: [0]
- Using Two Partitions (Legacy Only): [1]

McBSP Register Manual

- McBSP Register Mapping Summary: [2] [3] [4] [5] [6]

Table 21-77. MCBSPLP_PCR_REG

Address Offset	0x0000 0048		
Physical Address	0x4807 4048	**Instance**	McBSP1
	0x4809 6048		McBSP5
	0x4902 2048		McBSP2
	0x4902 4048		McBSP3
	0x4902 6048		McBSP4
Description	McBSPLP pin control register		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																IDLE_EN	XIOEN	RIOEN	FSXM	FSRM	CLKXM	CLKRM	SCLKME	CLKS_STAT	DX_STAT	DR_STAT	FSXP	FSRP	CLKXP	CLKRP	

Bits	Field Name	Description	Type	Reset
31:15	RESERVED	Read returns 0x0.	R	0x00000
14	IDLE_EN	Idle enable. This bit allows stopping all the clocks in the MCBSP module. (legacy) 0x0: The McBSP is running 0x1: The clocks in the McBSP are shut off when both IDLE_EN=1 and his power domain is in idle mode (Force idle or Smart idle)	RW	0x0

Bits	Field Name	Description	Type	Reset
13	XIOEN	Transmit General Purpose I/O Mode only when XRST=0 in SPCR[1,2] (legacy) 0x0: DX, FSX and CLKX are configured as serial port pins and do not function as general-purpose I/Os. 0x1: DX pin is a general purpose output. FSX and CLKX are general purpose I/Os. These serial port pins do not perform serial port operation.	RW	0x0
12	RIOEN	Receive General Purpose I/O Mode when RRST=0 in SPCR[1,2] (legacy) 0x0: DR, FSR, CLKR and CLKS are configured as serial port pins and do not function as general-purpose I/Os. 0x1: DR and CLKS pins are general purpose inputs; FSR and CLKR are general purpose I/Os. These serial port pins do not perform serial port operation. The CLKS pin is affected by a combination of RRST and RIOEN signals of the receiver.	RW	0x0
11	FSXM	Transmit Frame-Synchronization Mode 0x0: Frame-synchronization signal derived from an external source 0x1: Frame synchronization is determined by the SRG frame-synchronization mode bit FSGM in SRGR2.	RW	0x0
10	FSRM	Receive Frame-Synchronization Mode 0x0: Frame-Synchronization pulses generated by an external device. FSR is an input pin. 0x1: Frame synchronization generated internally by SRG. FSR is an output pin except when GSYNC=1 in SRGR.	RW	0x0
9	CLKXM	Transmitter Clock Mode When digital loop-back mode set (McBSPi.MCBSPLP_XCCR_REG[5] DLB=1), CLKXM bit is ignored. The internal transmit clock (not the mcbspi_clkx pin) is driven by the internal SRG and mcbspi_clkx pin is in high-impedance. 0x0: Transmitter clock is driven by an external clock with CLKX as an input pin. 0x1: CLKX is an output pin and is driven by the internal SRG.	RW	0x0
8	CLKRM	Receiver Clock Mode When digital loop-back mode set (McBSPi.MCBSPLP_XCCR_REG[5] DLB=1), CLKRM bit is ignored. The internal receive lock (not the mcbsp1_clkr pin) is driven by the internal SRG and mcbsp1_clkr pin is in high-impedance. 0x0: Receive clock is an input driven by an external clock with CLKR as an input pin. 0x1: CLKR is an output pin and is driven by the internal SRG.	RW	0x0
7	SCLKME	The frequency of CLKG is: CLKG frequency = (Input clock frequency) / (CLKGDV + 1) SCLKME is used in conjunction with the CLKSM bit to select the input clock: 0x0: CLKSM = 0: Signal on CLKS pin CLKSM = 1: McBSPi_ICLK clock 0x1: CLKSM = 0: Signal on CLKR pin CLKSM = 1: Signal on CLKX pin	RW	0x0

Bits	Field Name	Description	Type	Reset
6	CLKS_STAT	CLKS pin status. Reflects value on CLKS pin when selected as a general purpose input. (legacy) 0x0: The signal on the CLKS pin is low 0x1: The signal on the CLKS pin is high	R	0x0
5	DX_STAT	DX pin status. Reflects value driven on to DX pin when selected as a general purpose output. (legacy) 0x0: Drive the signal on the DX pin low 0x1: Drive the signal on the DX pin high	RW	0x0
4	DR_STAT	DR pin status. Reflects value on DR pin when selected as a general purpose input. (legacy) 0x0: The signal on DR pin is low 0x1: The signal on DR pin is high	R	0x0
3	FSXP	Transmit Frame-Synchronization Polarity 0x0: Frame-synchronization pulse FSX is active high 0x1: Frame-synchronization pulse FSX is active low	RW	0x0
2	FSRP	Receive Frame-Synchronization Polarity 0x0: Frame-synchronization pulse FSR is active high 0x1: Frame-synchronization pulse FSR is active low	RW	0x0
1	CLKXP	Transmit Clock Polarity 0x0: Transmit data driven on rising edge of CLKX 0x1: Transmit data driven on falling edge of CLKX	RW	0x0
0	CLKRP	Receive Clock Polarity 0x0: Receive data sampled on falling edge of CLKR 0x1: Receive data sampled on rising edge of CLKR	RW	0x0

Table 21-78. Register Call Summary for Register MCBSPLP_PCR_REG

McBSP Environment
- Words, Frames, and Phases Definitions: [0] [1]

McBSP Integration
- Clocks: [2] [3] [4] [5] [6]
- Power Management: [7] [8] [9]

McBSP Functional Description
- Clocking and Framing Data: [10] [11] [12] [13] [14] [15] [16] [17] [18] [19] [20] [21] [22] [23] [24] [25] [26] [27]
- Frame Phases (Dual-Phase Frame I2S Support): [28] [29] [30] [31] [32] [33] [34] [35]
- MCBSP Data Transfer Mode: [36] [37]
- McBSP SRG: [38] [39] [40] [41] [42]
- Clock Generation in the SRG: [43] [44] [45] [46]
- Frame Sync Generation in the SRG: [47] [48]
- Synchronizing SRG Outputs to an External Clock: [49] [50] [51] [52]
- Underflow in the Transmitter: [53]

McBSP Basic Programming Model
- McBSP Initialization Procedure: [54]
- Reset and Initialization Procedure for the Sample Rate Generator: [55] [56]
- Receiver Configuration: [57] [58] [59] [60] [61] [62] [63] [64] [65] [66] [67] [68]
- Transmitter Configuration: [69] [70] [71] [72] [73] [74]
- General-Purpose I/O on the McBSP Pins (Legacy Only): [75] [76] [77] [78] [79] [80]

McBSP Use Case and Tips
- Programming Flow: [81] [82] [83] [84] [85]

Table 21-78. Register Call Summary for Register MCBSPLP_PCR_REG (continued)

McBSP Register Manual

- McBSP Register Mapping Summary: [86] [87] [88] [89] [90]

Table 21-79. MCBSPLP_RCERC_REG

Address Offset	0x0000 004C		
Physical Address	0x4807 404C	**Instance**	McBSP1
	0x4809 604C		McBSP5
	0x4902 204C		McBSP2
	0x4902 404C		McBSP3
	0x4902 604C		McBSP4
Description	McBSPLP receive channel enable register partition C		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																RCERC															

Bits	Field Name	Description	Type	Reset
31:16	RESERVED	Read returns 0x0.	R	0x0000
15:0	RCERC	Receive Channel Enable RCERC n=0 Disables reception of n-th channel in an even-numbered block in partition C RCERC n=1 Enables reception of n-th channel in an even-numbered block in partition C	RW	0x0000

Table 21-80. Register Call Summary for Register MCBSPLP_RCERC_REG

McBSP Functional Description

- Using Eight Partitions: [0]

McBSP Register Manual

- McBSP Register Mapping Summary: [1] [2] [3] [4] [5]

Table 21-81. MCBSPLP_RCERD_REG

Address Offset	0x0000 0050		
Physical Address	0x4807 4050	**Instance**	McBSP1
	0x4809 6050		McBSP5
	0x4902 2050		McBSP2
	0x4902 4050		McBSP3
	0x4902 6050		McBSP4
Description	McBSPLP receive channel enable register partition D		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																RCERD															

Bits	Field Name	Description	Type	Reset
31:16	RESERVED	Read returns 0x0.	R	0x0000
15:0	RCERD	Receive Channel Enable RCERD n=0 Disables reception of n-th channel in an even-numbered block in partition D RCERD n=1 Enables reception of n-th channel in an even-numbered block in partition D	RW	0x0000

Table 21-82. Register Call Summary for Register MCBSPLP_RCERD_REG

McBSP Functional Description • Using Eight Partitions: [0]
McBSP Register Manual • McBSP Register Mapping Summary: [1] [2] [3] [4] [5]

Table 21-83. MCBSPLP_XCERC_REG

Address Offset	0x0000 0054		
Physical Address	0x4807 4054	**Instance**	McBSP1
	0x4809 6054		McBSP5
	0x4902 2054		McBSP2
	0x4902 4054		McBSP3
	0x4902 6054		McBSP4
Description	McBSPLP transmit channel enable register partition C		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																XCERC															

Bits	Field Name	Description	Type	Reset
31:16	RESERVED	Read returns 0x0.	R	0x0000
15:0	XCERC	Transmit Channel Enable XCERC n=0 Disables transmission of n-th channel in an event-numbered block in partition C XCERC n=1 Enables transmission of n-th channel in an event-numbered block in partition C	RW	0x0000

Table 21-84. Register Call Summary for Register MCBSPLP_XCERC_REG

McBSP Functional Description • Using Eight Partitions: [0]
McBSP Register Manual • McBSP Register Mapping Summary: [1] [2] [3] [4] [5]

Table 21-85. MCBSPLP_XCERD_REG

Address Offset	0x0000 0058		
Physical Address	0x4807 4058	**Instance**	McBSP1
	0x4809 6058		McBSP5
	0x4902 2058		McBSP2
	0x4902 4058		McBSP3
	0x4902 6058		McBSP4
Description	McBSPLP transmit channel enable register partition D		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																XCERD															

Bits	Field Name	Description	Type	Reset
31:16	RESERVED	Read returns 0x0.	R	0x0000
15:0	XCERD	Transmit Channel Enable XCERD n=0 Disables transmission of n-th channel in an even-numbered block in partition D XCERD n=1 Enables transmission of n-th channel in an even-numbered block in partition D	RW	0x0000

Table 21-86. Register Call Summary for Register MCBSPLP_XCERD_REG

McBSP Functional Description • Using Eight Partitions: [0]
McBSP Register Manual • McBSP Register Mapping Summary: [1] [2] [3] [4] [5]

Table 21-87. MCBSPLP_RCERE_REG

Address Offset	0x0000 005C		
Physical Address	0x4807 405C	**Instance**	McBSP1
	0x4809 605C		McBSP5
	0x4902 205C		McBSP2
	0x4902 405C		McBSP3
	0x4902 605C		McBSP4
Description	McBSPLP receive channel enable register partition E		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																RCERE															

Bits	Field Name	Description	Type	Reset
31:16	RESERVED	Read returns 0x0.	R	0x0000
15:0	RCERE	Receive Channel Enable RCERE n=0 Disables reception of n-th channel in an even-numbered block in partition E RCERE n=1 Enables reception of n-th channel in an even-numbered block in partition E	RW	0x0000

Table 21-88. Register Call Summary for Register MCBSPLP_RCERE_REG

McBSP Functional Description • Using Eight Partitions: [0]
McBSP Register Manual • McBSP Register Mapping Summary: [1] [2] [3] [4] [5]

Table 21-89. MCBSPLP_RCERF_REG

Address Offset	0x0000 0060		
Physical Address	0x4807 4060	**Instance**	McBSP1
	0x4809 6060		McBSP5
	0x4902 2060		McBSP2
	0x4902 4060		McBSP3
	0x4902 6060		McBSP4
Description	McBSPLP receive channel enable register partition F		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																RCERF															

Bits	Field Name	Description	Type	Reset
31:16	RESERVED	Read returns 0x0.	R	0x0000
15:0	RCERF	Receive Channel Enable RCERF n=0 Disables reception of n-th channel in an even-numbered block in partition F RCERF n=1 Enables reception of n-th channel in an even-numbered block in partition F	RW	0x0000

Table 21-90. Register Call Summary for Register MCBSPLP_RCERF_REG

McBSP Functional Description
- Using Eight Partitions: [0]

McBSP Register Manual
- McBSP Register Mapping Summary: [1] [2] [3] [4] [5]

Table 21-91. MCBSPLP_XCERE_REG

Address Offset	0x0000 0064		
Physical Address	0x4807 4064	**Instance**	McBSP1
	0x4809 6064		McBSP5
	0x4902 2064		McBSP2
	0x4902 4064		McBSP3
	0x4902 6064		McBSP4
Description	McBSPLP transmit channel enable register partition E		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																XCERE															

Bits	Field Name	Description	Type	Reset
31:16	RESERVED	Read returns 0x0.	R	0x0000
15:0	XCERE	Transmit Channel Enable XCERE n=0 Disables transmission of n-th channel in an event-numbered block in partition E XCERE n=1 Enables transmission of n-th channel in an event-numbered block in partition E	RW	0x0000

Table 21-92. Register Call Summary for Register MCBSPLP_XCERE_REG

McBSP Functional Description
- Using Eight Partitions: [0]

Table 21-92. Register Call Summary for Register MCBSPLP_XCERE_REG (continued)

McBSP Register Manual
• McBSP Register Mapping Summary: [1] [2] [3] [4] [5]

Table 21-93. MCBSPLP_XCERF_REG

Address Offset	0x0000 0068		
Physical Address	0x4807 4068	**Instance**	McBSP1
	0x4809 6068		McBSP5
	0x4902 2068		McBSP2
	0x4902 4068		McBSP3
	0x4902 6068		McBSP4
Description	McBSPLP transmit channel enable register partition F		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																XCERF															

Bits	Field Name	Description	Type	Reset
31:16	RESERVED	Read returns 0x0.	R	0x0000
15:0	XCERF	Transmit Channel Enable XCERF n=0 Disables transmission of n-th channel in an even-numbered block in partition F XCERF n=1 Enables transmission of n-th channel in an even-numbered block in partition F	RW	0x0000

Table 21-94. Register Call Summary for Register MCBSPLP_XCERF_REG

McBSP Functional Description
• Using Eight Partitions: [0]
McBSP Register Manual
• McBSP Register Mapping Summary: [1] [2] [3] [4] [5]

Table 21-95. MCBSPLP_RCERG_REG

Address Offset	0x0000 006C		
Physical Address	0x4807 406C	**Instance**	McBSP1
	0x4809 606C		McBSP5
	0x4902 206C		McBSP2
	0x4902 406C		McBSP3
	0x4902 606C		McBSP4
Description	McBSPLP receive channel enable register partition G		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																RCERG															

Bits	Field Name	Description	Type	Reset
31:16	RESERVED	Read returns 0x0.	R	0x0000
15:0	RCERG	Receive Channel Enable RCERG n=0 Disables reception of n-th channel in an even-numbered block in partition G RCERG n=1 Enables reception of n-th channel in an even-numbered block in partition G	RW	0x0000

Table 21-96. Register Call Summary for Register MCBSPLP_RCERG_REG

McBSP Functional Description • Using Eight Partitions: [0]
McBSP Register Manual • McBSP Register Mapping Summary: [1] [2] [3] [4] [5]

Table 21-97. MCBSPLP_RCERH_REG

Address Offset	0x0000 0070		
Physical Address	0x4807 4070	**Instance**	McBSP1
	0x4809 6070		McBSP5
	0x4902 2070		McBSP2
	0x4902 4070		McBSP3
	0x4902 6070		McBSP4
Description	McBSPLP receive channel enable register partition H		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																RCERH															

Bits	Field Name	Description	Type	Reset
31:16	RESERVED	Read returns 0x0.	R	0x0000
15:0	RCERH	Receive Channel Enable RCERH n=0 Disables reception of n-th channel in an even-numbered block in partition H RCERH n=1 Enables reception of n-th channel in an even-numbered block in partition H	RW	0x0000

Table 21-98. Register Call Summary for Register MCBSPLP_RCERH_REG

McBSP Functional Description • Using Eight Partitions: [0] • Receive Multichannel Selection Mode: [1] [2] • Transmit Multichannel Selection Modes: [3] [4]
McBSP Register Manual • McBSP Register Mapping Summary: [5] [6] [7] [8] [9]

Table 21-99. MCBSPLP_XCERG_REG

Address Offset	0x0000 0074		
Physical Address	0x4807 4074	**Instance**	McBSP1
	0x4809 6074		McBSP5
	0x4902 2074		McBSP2
	0x4902 4074		McBSP3
	0x4902 6074		McBSP4
Description	McBSPLP transmit channel enable register partition G		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																XCERG															

Bits	Field Name	Description	Type	Reset
31:16	RESERVED	Read returns 0x0.	R	0x0000
15:0	XCERG	Transmit Channel Enable XCERG n=0 Disables transmission of n-th channel in an event-numbered block in partition G XCERG n=1 Enables transmission of n-th channel in an event-numbered block in partition G	RW	0x0000

Table 21-100. Register Call Summary for Register MCBSPLP_XCERG_REG

McBSP Functional Description • Using Eight Partitions: [0]
McBSP Register Manual • McBSP Register Mapping Summary: [1] [2] [3] [4] [5]

Table 21-101. MCBSPLP_XCERH_REG

Address Offset	0x0000 0078		
Physical Address	0x4807 4078	**Instance**	McBSP1
	0x4809 6078		McBSP5
	0x4902 2078		McBSP2
	0x4902 4078		McBSP3
	0x4902 6078		McBSP4
Description	McBSPLP transmit channel enable register partition H		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																XCERH															

Bits	Field Name	Description	Type	Reset
31:16	RESERVED	Read returns 0x0.	R	0x0000
15:0	XCERH	Transmit Channel Enable XCERH n=0 Disables transmission of n-th channel in an even-numbered block in partition H XCERH n=1 Enables transmission of n-th channel in an even-numbered block in partition H	RW	0x0000

Table 21-102. Register Call Summary for Register MCBSPLP_XCERH_REG

McBSP Functional Description • Using Eight Partitions: [0] • Transmit Multichannel Selection Modes: [1] [2] [3] [4] [5] [6]
McBSP Register Manual • McBSP Register Mapping Summary: [7] [8] [9] [10] [11]

Table 21-103. MCBSPLP_REV_REG

Address Offset	0x0000 007C		
Physical Address	0x4807 407C	**Instance**	McBSP1
	0x4809 607C		McBSP5
	0x4902 207C		McBSP2
	0x4902 407C		McBSP3
	0x4902 607C		McBSP4
Description	MCBSPLP Revision number register		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																								REV							

Bits	Field Name	Description	Type	Reset
31:8	RESERVED	Read returns 0x0.	R	0x000000
7:0	REV	IP revision [7:4] Major revision [3:0] Minor revision examples: 0x10 for 1.0, 0x21 for 2.1	R	See (1)

(1) TI internal data

Table 21-104. Register Call Summary for Register MCBSPLP_REV_REG

McBSP Register Manual

- McBSP Register Mapping Summary: [0] [1] [2] [3] [4]

Table 21-105. MCBSPLP_RINTCLR_REG

Address Offset	0x0000 0080		
Physical Address	0x4807 4080	**Instance**	McBSP1
	0x4809 6080		McBSP5
	0x4902 2080		McBSP2
	0x4902 4080		McBSP3
	0x4902 6080		McBSP4
Description	McBSPLP receive interrupt clear		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RINTCLR																															

Bits	Field Name	Description	Type	Reset
31:0	RINTCLR	Read from this register will clear the IRQ generated by receive end-of-frame indication or mcbsp1_fsr detection. Write to this register has no effect. (legacy)	RW	0x00000000

Table 21-106. Register Call Summary for Register MCBSPLP_RINTCLR_REG

McBSP Register Manual

- McBSP Register Mapping Summary: [0] [1] [2] [3] [4]

Table 21-107. MCBSPLP_XINTCLR_REG

Address Offset	0x0000 0084		
Physical Address	0x4807 4084	**Instance**	McBSP1
	0x4809 6084		McBSP5
	0x4902 2084		McBSP2
	0x4902 4084		McBSP3
	0x4902 6084		McBSP4
Description	McBSPLP transmit interrupt clear (legacy)		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
XINTCLR																															

Bits	Field Name	Description	Type	Reset
31:0	XINTCLR	Read from this register will clear the IRQ generated by transmit end-of-frame indication or mcbspi_fsx detection (I=1:5). Write to this register has no effect. (legacy)	RW	0x00000000

Table 21-108. Register Call Summary for Register MCBSPLP_XINTCLR_REG

McBSP Register Manual

- McBSP Register Mapping Summary: [0] [1] [2] [3] [4]

Table 21-109. MCBSPLP_ROVFLCLR_REG

Address Offset	0x0000 0088		
Physical Address	0x4807 4088	**Instance**	McBSP1
	0x4809 6088		McBSP5
	0x4902 2088		McBSP2
	0x4902 4088		McBSP3
	0x4902 6088		McBSP4
Description	McBSPLP receive overflow interrupt clear		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ROVFLCLR																															

Bits	Field Name	Description	Type	Reset
31:0	ROVFLCLR	Read from this register will clear the IRQ generated by the receive overflow condition. Write to this register has no effect. ((legacy)	RW	0x00000000

Table 21-110. Register Call Summary for Register MCBSPLP_ROVFLCLR_REG

McBSP Register Manual

- McBSP Register Mapping Summary: [0] [1] [2] [3] [4]

Table 21-111. MCBSPLP_SYSCONFIG_REG

Address Offset	0x0000 008C		
Physical Address	0x4807 408C	**Instance**	McBSP1
	0x4809 608C		McBSP5
	0x4902 208C		McBSP2
	0x4902 408C		McBSP3
	0x4902 608C		McBSP4
Description	McBSPLP System Configuration register		
Type	RW		

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10	9 8	7 6 5	4 3	2	1	0
RESERVED			CLOCKACTIVITY	RESERVED	SIDLEMODE	ENAWAKEUP	SOFTRESET	RESERVED

Bits	Field Name	Description	Type	Reset
31:10	RESERVED	Read returns 0x0.	R	0x000000
9:8	CLOCKACTIVITY	0x0: The McBSPi_ICLK clock can be switched off. The PRCM functional clock can be switched off. 0x1: The McBSPi_ICLK clock must be maintained during wakeup. The PRCM functional clock can be switched off. 0x2: The McBSPi_ICLK clock can be switched off. The PRCM functional clock must be maintained during wakeup. 0x3: The McBSPi_ICLK clock must be maintained during wakeup. The PRCM functional clock must be maintained during wakeup.	RW	0x0
7:5	RESERVED	Read returns 0x0.	R	0x0
4:3	SIDLEMODE	Slave interface power management, idle request / acknowledge control: 0x0: Force-idle. An idle request is acknowledged unconditionally. 0x1: No-idle. An idle request is never acknowledged. 0x2: Smart-idle. Acknowledgment to an idle request is given based on the internal activity of the module 0x3: Reserved	RW	0x0
2	ENAWAKEUP	Wake-up feature control: 0x0: Wake-up is disabled 0x1: Wake-up capability is enabled	RW	0x0
1	SOFTRESET	McBSP global software reset 0x0: NO soft reset 0x1: Soft reset triggered	RW	0x0
0	RESERVED	Read returns 0x0.	R	0x0

Table 21-112. Register Call Summary for Register MCBSPLP_SYSCONFIG_REG

McBSP Integration

- Hardware and Software Reset: [0]
- Power Management: [1] [2] [3] [4] [5] [6]

Table 21-112. Register Call Summary for Register MCBSPLP_SYSCONFIG_REG (continued)

McBSP Basic Programming Model • McBSP Initialization Procedure: [7]
McBSP Register Manual • McBSP Register Mapping Summary: [8] [9] [10] [11] [12]

Table 21-113. MCBSPLP_THRSH2_REG

Address Offset	0x0000 0090		
Physical Address	0x4807 4090	**Instance**	McBSP1
	0x4809 6090		McBSP5
	0x4902 2090		McBSP2
	0x4902 4090		McBSP3
	0x4902 6090		McBSP4
Description	McBSPLP transmit buffer threshold (DMA or IRQ trigger)		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																					XTHRESHOLD										

Bits	Field Name	Description	Type	Reset
31:11	RESERVED	Read returns 0x0.	R	0x000000
10:0[(1)]	XTHRESHOLD	Transmit buffer threshold value. The DMA request (if enabled) of interrupt assertion (if enabled) is triggered if the number of free locations in the transmit buffer are above or equal to the XTHRESHOLD value + 1. Also, this value (XTHRESHOLD value + 1) indicates the number of words transferred during a transmit data DMA request, if transmit DMA is enabled.	RW	0x00

(1) XTHRESHOLD is an 11-bit field for McBSP2 only. For other McBSPs, XTHRESHOLD is an 8-bit field (bits 7 to 10 are reserved). In other words, the other McBSPs are limited to a FIFO width of 0x7F.

Table 21-114. Register Call Summary for Register MCBSPLP_THRSH2_REG

McBSP Integration • Power Management: [0]
McBSP Functional Description • McBSP Transmission: [1] • McBSP DMA Configuration: [2]
McBSP Basic Programming Model • McBSP Initialization Procedure: [3] • Data Transfer DMA Request Configuration: [4] • Transmitter Configuration: [5]
McBSP Register Manual • McBSP Register Mapping Summary: [6] [7] [8] [9] [10]

Table 21-115. MCBSPLP_THRSH1_REG

Address Offset	0x0000 0094		
Physical Address	0x4807 4094	**Instance**	McBSP1
	0x4809 6094		McBSP5
	0x4902 2094		McBSP2
	0x4902 4094		McBSP3
	0x4902 6094		McBSP4
Description	McBSPLP receive buffer threshold (DMA or IRQ trigger)		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																					RTHRESHOLD										

Bits	Field Name	Description	Type	Reset
31:11	RESERVED	Read returns 0x0.	R	0x000000
10:0[(1)]	RTHRESHOLD	Receive buffer threshold value. The DMA request (if enabled) of interrupt assertion (if enabled)is triggered if the number of occupied locations in the receive buffer are above or equal to the RTHRESHOLD value + 1. Also, this value (RTHRESHOLD value + 1) indicates the number of words transferred during a receive data DMA request, if receive DMA is enabled.	RW	0x00

(1) RTHRESHOLD is an 11-bit field for McBSP2 only. For other McBSPs, RTHRESHOLD is an 8-bit field (bits 7 to 10 are reserved). In other words, the other McBSPs are limited to a FIFO width of 0x7F.

Table 21-116. Register Call Summary for Register MCBSPLP_THRSH1_REG

McBSP Integration
- Power Management: [0]

McBSP Functional Description
- McBSP Reception: [1]
- McBSP DMA Configuration: [2]

McBSP Basic Programming Model
- Data Transfer DMA Request Configuration: [3]
- Receiver Configuration: [4]

McBSP Use Case and Tips
- Programming Flow: [5] [6]

McBSP Register Manual
- McBSP Register Mapping Summary: [7] [8] [9] [10] [11]

Table 21-117. MCBSPLP_IRQSTATUS_REG

Address Offset	0x0000 00A0		
Physical Address	0x4807 40A0	**Instance**	McBSP1
	0x4809 60A0		McBSP5
	0x4902 20A0		McBSP2
	0x4902 40A0		McBSP3
	0x4902 60A0		McBSP4
Description	McBSPLP Interrupt Status register (OCP compliant IRQ line)		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																XEMPTYEOF	RESERVED	XOVFLSTAT	XUNDFLSTAT	XRDY	XEOF	XFSX	XSYNCERR	RESERVED	ROVFLSTAT	RUNDFLSTAT	RRDY	REOF	RFSR	RSYNCERR	

Bits	Field Name	Description	Type	Reset
31:15	RESERVED	Read returns 0x0.	R	0x00000
14	XEMPTYEOF	Transmit Buffer Empty at end of frame (XEMPTYEOF is set to one when a complete frame was transmitted and the transmit buffer is empty). Writing 1 to this bit clears the bit. 0x0: Transmit buffer NOT Empty at end of frame 0x1: Transmit buffer Empty at end of frame; Writing 1 to this bit clears the bit.	RW	0x0
13	RESERVED	Read returns 0x0.	R	0x0
12	XOVFLSTAT	Transmit Buffer Overflow (XOVFLSTAT bit is set to one when transmit buffer overflow; the data which is written while overflow condition is discarded). Writing 1 to this bit clears the bit. 0x0: Transmit buffer NOT overflow 0x1: Transmit buffer overflow; Writing 1 to this bit clears the bit.	RW	0x0
11	XUNDFLSTAT	Transmit Buffer Underflow (XUNDFLSTAT bit is set to one when the transmit data buffer is empty new data is required to be transmitted). Writing 1 to this bit clears the bit. 0x0: the transmit data buffer is NOT empty new data is required to be transmitted. 0x1: the transmit data buffer is empty new data is required to be transmitted. Writing 1 to this bit clears the bit.	RW	0x0
10	XRDY	Transmit Buffer Threshold Reached (XRDY bit is set to one when the transmit buffer free locations are equal or above the THRSH2_REG value). Writing 1 to this bit clears the bit. 0x0: Transmit buffer occupied locations are below the THRSH2_REG value). 0x1: Transmit buffer occupied locations are equal or above the THRSH2_REG value). Writing 1 to this bit clears the bit.	RW	0x0
9	XEOF	Transmit End Of Frame (XEOF is set to one when a complete frame was transmitted). Writing 1 to this bit clears the bit. 0x0: complete frame was NOT transmitted 0x1: complete frame was transmitted; Writing 1 to this bit clears the bit.	RW	0x0
8	XFSX	Transmit Frame Synchronization (XFSX bit is set to one when a new transmit frame synchronization is asserted). Writing 1 to this bit clears the bit. 0x0: new transmit frame synchronization is NOT asserted 0x1: new transmit frame synchronization is asserted; Writing 1 to this bit clears the bit.	RW	0x0

Bits	Field Name	Description	Type	Reset
7	XSYNCERR	Transmit Frame Synchronization Error (XSYNCERR is set to one when a transmit frame synchronization error is detected). Writing 1 to this bit clears the bit. 0x0: Transmit frame synchronization error is NOT detected 0x1: Transmit frame synchronization error is detected. Writing 1 to this bit clears the bit.	RW	0x0
6	RESERVED	Read returns 0x0.	R	0x0
5	ROVFLSTAT	Receive Buffer Overflow (ROVFLSTAT bit is set to one when receive buffer overflow; the data which is written while overflow condition is discarded). Writing 1 to this bit clears the bit. 0x0: receive buffer NOT overflow 0x1: receive buffer overflow; Writing 1 to this bit clears the bit.	RW	0x0
4	RUNDFLSTAT	Receive Buffer Underflow (RUNDFLSTAT bit is set to one when read operation is performed to the receive data register while receive buffer is empty; data read while underflow condition is undefined). Writing 1 to this bit clears the bit. 0x0: read operation is performed to the receive data register while receive buffer is NOT empty 0x1: read operation is performed to the receive data register while receive buffer is empty; Writing 1 to this bit clears the bit.	RW	0x0
3	RRDY	Receive Buffer Threshold Reached (RRDY bit is set to one when the receive buffer occupied locations are equal or above the THRSH1_REG value). Writing 1 to this bit clears the bit. 0x0: receive buffer occupied locations are below the THRSH1_REG value). 0x1: receive buffer occupied locations are equal or above the THRSH1_REG value). Writing 1 to this bit clears the bit.	RW	0x0
2	REOF	Receive End Of Frame (REOF is set to one when a complete frame was received). Writing 1 to this bit clears the bit. 0x0: complete frame was NOT received 0x1: complete frame was received; Writing 1 to this bit clears the bit.	RW	0x0
1	RFSR	Receive Frame Synchronization (RFSR bit is set to one when a new receive frame synchronization is asserted). Writing 1 to this bit clears the bit. 0x0: new receive frame synchronization is NOT asserted 0x1: new receive frame synchronization is asserted; Writing 1 to this bit clears the bit.	RW	0x0
0	RSYNCERR	Receive Frame Synchronization Error (RSYNCERR is set to one when a receive frame synchronization error is detected). Writing 1 to this bit clears the bit. 0x0: receive frame synchronization error is NOT detected 0x1: receive frame synchronization error is detected. Writing 1 to this bit clears the bit.	RW	0x0

Table 21-118. Register Call Summary for Register MCBSPLP_IRQSTATUS_REG

McBSP Integration
- Power Management: [0] [1] [2] [3]
- Interrupt Requests: [4] [5] [6] [7] [8] [9] [10] [11] [12] [13] [14] [15] [16] [17]

McBSP Functional Description
- Enable/Disable the Transmit and Receive Processes: [18] [19] [20] [21]
- Introduction: [22] [23] [24] [25] [26] [27] [28]
- Overrun in the Receiver: [29] [30] [31]
- Unexpected Receive Frame-sync Pulse: [32] [33]
- Underflow in the Receiver: [34] [35]
- Underflow in the Transmitter: [36] [37] [38]
- Unexpected Transmit Frame-sync Pulse: [39]
- Overflow in the Transmitter: [40]

McBSP Basic Programming Model
- Interrupt Configuration: [41] [42] [43] [44] [45] [46] [47] [48] [49] [50] [51] [52] [53] [54] [55] [56] [57]

McBSP Register Manual
- McBSP Register Mapping Summary: [58] [59] [60] [61] [62]

Table 21-119. MCBSPLP_IRQENABLE_REG

Address Offset	0x0000 00A4		
Physical Address	0x4807 40A4	**Instance**	McBSP1
	0x4809 60A4		McBSP5
	0x4902 20A4		McBSP2
	0x4902 40A4		McBSP3
	0x4902 60A4		McBSP4
Description	McBSPLP Interrupt Enable register (OCP compliant IRQ line)		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																XEMPTYEOFEN	RESERVED	XOVFLEN	XUNDFLEN	XRDYEN	XEOFEN	XFSXEN	XSYNCERREN	RESERVED	ROVFLEN	RUNDFLEN	RRDYEN	REOFEN	RFSREN	RSYNCERREN	

Bits	Field Name	Description	Type	Reset
31:15	RESERVED	Read returns 0x0.	R	0x00000
14	XEMPTYEOFEN	Transmit buffer empty at end of frame enable bit. 0x0: Transmit buffer Empty at end of frame NOT enabled 0x1: Transmit buffer Empty at end of frame enabled	RW	0x0
13	RESERVED	Read returns 0x0.	R	0x0
12	XOVFLEN	Transmit Buffer Overflow enable bit. 0x0: Transmit Buffer Overflow NOT enabled 0x1: Transmit Buffer Overflow enabled	RW	0x0
11	XUNDFLEN	Transmit Buffer Underflow enable bit. 0x0: Transmit Buffer Underflow NOT enabled 0x1: Transmit Buffer Underflow enabled	RW	0x0
10	XRDYEN	Transmit Buffer Threshold Reached enable bit. 0x0: Transmit Buffer Threshold Reached NOT enabled 0x1: Transmit Buffer Threshold Reached enabled	RW	0x0

Bits	Field Name	Description	Type	Reset
9	XEOFEN	Transmit End Of Frame enable bit. 0x0: Transmit End Of Frame NOT enabled 0x1: Transmit End Of Frame enabled	RW	0x0
8	XFSXEN	Transmit Frame Synchronization enable bit. 0x0: Transmit Frame Synchronization NOT enabled 0x1: Transmit Frame Synchronization enabled	RW	0x0
7	XSYNCERREN	Transmit Frame Synchronization Error enable bit. 0x0: Transmit Frame Synchronization Error NOT enabled 0x1: Transmit Frame Synchronization Error enabled	RW	0x0
6	RESERVED	Read returns 0x0.	R	0x0
5	ROVFLEN	Receive Buffer Overflow enable bit. 0x0: Receive Buffer Overflow NOT enabled 0x1: Receive Buffer Overflow enabled	RW	0x0
4	RUNDFLEN	Receive Buffer Underflow enable bit. 0x0: Receive Buffer Underflow NOT enabled 0x1: Receive Buffer Underflow enabled	RW	0x0
3	RRDYEN	Receive Buffer Threshold enable bit. 0x0: Receive Buffer Threshold NOT enabled 0x1: Receive Buffer Threshold enabled	RW	0x0
2	REOFEN	Receive End Of Frame enable bit. 0x0: Receive End Of Frame NOT enabled 0x1: Receive End Of Frame enabled	RW	0x0
1	RFSREN	Receive Frame Synchronization enable bit. RW 0x0: Receive Frame Synchronization NOT enabled 0x1: Receive Frame Synchronization enabled	RW	0x0
0	RSYNCERREN	Receive Frame Synchronization Error enable bit. 0x0: Receive Frame Synchronization Error NOT enabled 0x1: Receive Frame Synchronization Error enabled	RW	0x0

Table 21-120. Register Call Summary for Register MCBSPLP_IRQENABLE_REG

McBSP Integration
- Power Management: [0] [1] [2] [3] [4] [5] [6] [7] [8]
- Interrupt Requests: [9] [10] [11] [12] [13] [14] [15] [16] [17] [18] [19] [20] [21] [22]

McBSP Functional Description
- Overrun in the Receiver: [23]
- Unexpected Receive Frame-sync Pulse: [24]
- Underflow in the Receiver: [25]
- Underflow in the Transmitter: [26]
- Unexpected Transmit Frame-sync Pulse: [27]
- Overflow in the Transmitter: [28]

McBSP Basic Programming Model
- Data Transfer DMA Request Configuration: [29] [30]
- Interrupt Configuration: [31] [32]

McBSP Register Manual
- McBSP Register Mapping Summary: [33] [34] [35] [36] [37]

Table 21-121. MCBSPLP_WAKEUPEN_REG

Address Offset	0x0000 00A8		
Physical Address	0x4807 40A8	**Instance**	McBSP1
	0x4809 60A8		McBSP5
	0x4902 20A8		McBSP2
	0x4902 40A8		McBSP3
	0x4902 60A8		McBSP4
Description	McBSPLP Wakeup Enable register		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																XEMPTYEOFEN	RESERVED				XRDYEN	XEOFEN	XFSXEN	XSYNCERREN	RESERVED			RRDYEN	REOFEN	RFSREN	RSYNCERREN

Bits	Field Name	Description	Type	Reset
31:15	RESERVED	Read returns 0x0.	R	0x00000
14	XEMPTYEOFEN	Transmit buffer empty at end of frame WK enable bit. 0x0: Transmit buffer Empty at end of frame WK enable is NOT active 0x1: Transmit buffer Empty at end of frame WK enable is active	RW	0x0
13:11	RESERVED	Read returns 0x0.	R	0x0
10	XRDYEN	Transmit Buffer Threshold Reached WK enable bit. 0x0: Transmit Buffer Threshold WK enable is NOT active 0x1: Transmit Buffer Threshold WK enable is active	RW	0x0
9	XEOFEN	Transmit End Of Frame WK enable bit. 0x0: Transmit End Of Frame WK enable is NOT active 0x1: Transmit End Of Frame WK enable is active	RW	0x0
8	XFSXEN	Transmit Frame Synchronization WK enable bit. 0x0: Transmit Frame Synchronization WK enable is NOT active 0x1: Transmit Frame Synchronization WK enable is active	RW	0x0
7	XSYNCERREN	Transmit Frame Synchronization Error WK enable bit. 0x0: Transmit Frame Synchronization Error WK enable is NOT active 0x1: Transmit Frame Synchronization Error WK enable is active	RW	0x0
6:4	RESERVED	Read returns 0x0.	R	0x0
3	RRDYEN	Receive Buffer Threshold wake-up enable bit. 0x0: Receive Buffer Threshold WK enable is NOT active 0x1: Receive Buffer Threshold WK enable is active	RW	0x0
2	REOFEN	Receive End Of Frame WK enable bit. 0x0: Receive End Of Frame WK enable is NOT active 0x1: Receive End Of Frame WK enable is active	RW	0x0
1	RFSREN	Receive Frame Synchronization WK enable bit. 0x0: Receive Frame Synchronization WK enable is NOT active 0x1: Receive Frame Synchronization WK enable is active	RW	0x0

Bits	Field Name	Description	Type	Reset
0	RSYNCERREN	Receive Frame Synchronization Error WK enable bit. 0x0: Receive Frame Synchronization Error WK enable is NOT active 0x1: Receive Frame Synchronization Error WK enable is active	RW	0x0

Table 21-122. Register Call Summary for Register MCBSPLP_WAKEUPEN_REG

McBSP Integration
- Power Management: [0] [1] [2] [3] [4] [5] [6] [7] [8] [9]

McBSP Register Manual
- McBSP Register Mapping Summary: [10] [11] [12] [13] [14]

Table 21-123. MCBSPLP_XCCR_REG

Address Offset	0x0000 00AC		
Physical Address	0x4807 40AC	**Instance**	McBSP1
	0x4809 60AC		McBSP5
	0x4902 20AC		McBSP2
	0x4902 40AC		McBSP3
	0x4902 60AC		McBSP4
Description	McBSPLP transmit configuration control register		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																EXTCLKGATE	PPCONNECT	DXENDLY		XFULL_CYCLE	RESERVED			DLB	RESERVED	XDMAEN	RESERVED				XDISABLE

Bits	Field Name	Description	Type	Reset
31:16	RESERVED	Read returns 0x0.	R	0x0000
15	EXTCLKGATE	External clock gating enable (CLKX and FSX master only). When this bit is set and the transmit clock and FSX are set as output, the CLKX is enabled when FSX is active plus 3 clock cycles after (clock is provided for FWID + 4 clock cycles, assuming that the FSX width, active, is FWID + 1 clock cycles); outside this window the external transmit clock is gated. The receive use the same gated transmit clock and transmit frame synchronization signals regardless of the CLKRM/FSRM settings. When using this mode the frame synchronization signal must be active during reception of the entire frame (FWID must be programmed accordingly) to ensure the proper receive process, which requires at least 3 cycles after the frame complete to transfer the data into the receive buffer. 0x0: External clock gating disabled. 0x1: External clock gating enable.	RW	0x0

Bits	Field Name	Description	Type	Reset
14	PPCONNECT	Pair to pair connection. When set the DXENO pin is always set to 0, regardless of the frame boundary, setting the tree state buffer as output. 0x0: non Pair-to-pair connection. The DX pin will go to high-impedance state when there is no frame to transmit. 0x1: Pair-to-pair connection. When set, the DXENO pin is always set to 0, regardless of the frame boundary, setting the tree state buffer as output. This means the DX pin will be driven outside valid frame window. In that case, data sent by McBSP module during inactive channel are not guaranteed.	RW	0x0
13:12	DXENDLY	When McBSPi.MCBSPLP_SPCR1_REG[7] DXENA bit is set to one, this field selects the added delay as follow: 0x0: 18 ns 0x1: 26 ns (default) 0x2: 35 ns 0x3: 42 ns	RW	0x1
11	XFULL_CYCLE	Transmit full cycle mode select: 0x0: McBSP module operates in transmit half-cycle mode (transmit frame synchronization is sampled by the opposite edge of the clock used to drive transmit data) 0x1: McBSP module operates in transmit full-cycle mode (transmit frame synchronization is sampled by the same edge of the clock used to drive transmit data)	RW	0x0
10:6	RESERVED	Read returns 0x0.	R	0x00
5	DLB	Digital Loop-Back 0x0: No DLB 0x1: DLB	RW	0x0
4	RESERVED	Read returns 0x0.	R	0x0
3	XDMAEN	Transmit DMA Enable bit. When set to zero this bit will gate the external transmit DMA request, without resetting the DMA state machine. It is recommended to change this bit value only during transmit reset. 0x0: When set to zero this bit will gate the external transmit DMA request, 0x1: When set to one this bit will NOT gate the external transmit DMA request,	RW	0x1
2:1	RESERVED	Read returns 0x0.	R	0x0
0	XDISABLE	Transmit Disable bit. When this bit is set the transmit process will stop at the next frame boundary. 0x0: The transmit process will NOT stop at the next frame boundary. 0x1: The transmit process will stop at the next frame boundary.	RW	0x0

Table 21-124. Register Call Summary for Register MCBSPLP_XCCR_REG

McBSP Functional Description

- Clocking and Framing Data: [0] [1] [2] [3]
- Enable/Disable the Transmit and Receive Processes: [4] [5] [6]
- MCBSP Data Transfer Mode: [7] [8]
- Clock Generation in the SRG: [9]
- McBSP DMA Configuration: [10]

Table 21-124. Register Call Summary for Register MCBSPLP_XCCR_REG (continued)

McBSP Basic Programming Model

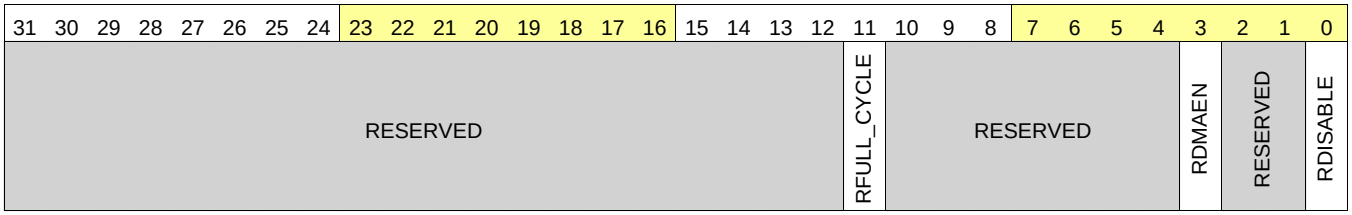
- McBSP Core: [11]
- McBSP Initialization Procedure: [12]
- Receiver Configuration: [13] [14] [15]
- Transmitter Configuration: [16] [17]

McBSP Register Manual

- McBSP Register Mapping Summary: [18] [19] [20] [21] [22]
- McBSP Register Description: [23] [24]

Table 21-125. MCBSPLP_RCCR_REG

Address Offset	0x0000 00B0		
Physical Address	0x4807 40B0	**Instance**	McBSP1
	0x4809 60B0		McBSP5
	0x4902 20B0		McBSP2
	0x4902 40B0		McBSP3
	0x4902 60B0		McBSP4
Description	McBSPLP receive configuration control register		
Type	RW		



Bits	Field Name	Description	Type	Reset
31:12	RESERVED	Read returns 0x0.	R	0x0000000
11	RFULL_CYCLE	Receive full cycle mode select: 0x0: McBSP module operates in receive half-cycle mode (receive frame synchronization is sampled by the opposite edge of the clock used to sample receive data) 0x1: McBSP module operates in receive full-cycle mode (receive frame synchronization is sampled by the same edge of the clock used to sample receive data)	RW	0x1
10:4	RESERVED	Read returns 0x0.	R	0x0000000
3	RDMAEN	Receive DMA Enable bit. When set to zero this bit will gate the external transmit DMA request, without resetting the DMA state machine. It is recommended to change this bit value only during receive reset. 0x0: When set to zero this bit will gate the external transmit DMA request 0x1: When set to one this bit will NOT gate the external transmit DMA request	RW	0x1
2:1	RESERVED	Read returns 0x0.	R	0x0
0	RDISABLE	Receive Disable bit. When this bit is set the receive process will stop at the next frame boundary. 0x0: the receive process will NOT stop at the next frame boundary. 0x1: When this bit is set the receive process will stop at the next frame boundary.	RW	0x0

Table 21-126. Register Call Summary for Register MCBSPLP_RCCR_REG

McBSP Functional Description

- Enable/Disable the Transmit and Receive Processes: [0] [1]
- MCBSP Data Transfer Mode: [2] [3]
- McBSP DMA Configuration: [4]

McBSP Basic Programming Model

- McBSP Core: [5]
- Receiver Configuration: [6]

McBSP Register Manual

- McBSP Register Mapping Summary: [7] [8] [9] [10] [11]

Table 21-127. MCBSPLP_XBUFFSTAT_REG

Address Offset	0x0000 00B4		
Physical Address	0x4807 40B4	**Instance**	McBSP1
	0x4809 60B4		McBSP5
	0x4902 20B4		McBSP2
	0x4902 40B4		McBSP3
	0x4902 60B4		McBSP4
Description	McBSPLP transmit buffer status		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																					XBUFFSTAT										

Bits	Field Name	Description	Type	Reset
31:11	RESERVED	Read returns 0x0.	R	0x000000
10:0[(1)]	XBUFFSTAT	Transmit Buffer Status (indicates the number of free locations in the transmit buffer). The XBUFFSTAT value reflects the buffer status on the L4 clock domain and it can be bigger than the real number of the free locations which are seen by the transmit state machine.	R	0x500[(2)]

(1) XBUFFSTAT is an 11-bit field for McBSP2 only. For other McBSPs, XBUFFSTAT is an 8-bit field (bits 7 to 10 are reserved).
(2) The reset value of XBUFFSTAT for other McBSPs is 0x080.

Table 21-128. Register Call Summary for Register MCBSPLP_XBUFFSTAT_REG

McBSP Functional Description

- Enable/Disable the Transmit and Receive Processes: [0]
- McBSP DMA Configuration: [1]

McBSP Register Manual

- McBSP Register Mapping Summary: [2] [3] [4] [5] [6]

Table 21-129. MCBSPLP_RBUFFSTAT_REG

Address Offset	0x0000 00B8		
Physical Address	0x4807 40B8	**Instance**	McBSP1
	0x4809 60B8		McBSP5
	0x4902 20B8		McBSP2
	0x4902 40B8		McBSP3
	0x4902 60B8		McBSP4
Description	McBSPLP receive buffer status		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																					RBUFFSTAT										

Bits	Field Name	Description	Type	Reset
31:11	RESERVED	Read returns 0x0.	R	0x000000
10:0[(1)]	RBUFFSTAT	Receive Buffer Status (indicates the number of occupied locations in the receive buffer). The RBUFFSTAT value reflects the buffer status on the L4 clock domain and it can be smaller than the real number of the occupied locations which are seen by the receive state machine.	R	0x00

(1) RBUFFSTAT is an 11-bit field in McBSP2. For other McBSPs, RBUFFSTAT is an 8-bit field (bits 7 to 10 are reserved).

Table 21-130. Register Call Summary for Register MCBSPLP_RBUFFSTAT_REG

McBSP Functional Description

- Enable/Disable the Transmit and Receive Processes: [0]
- McBSP DMA Configuration: [1]

McBSP Register Manual

- McBSP Register Mapping Summary: [2] [3] [4] [5] [6]

Table 21-131. MCBSPLP_SSELCR_REG

Address Offset	0x0000 00BC		
Physical Address	0x4807 40BC	**Instance**	McBSP1
	0x4809 60BC		McBSP5
	0x4902 20BC		McBSP2
	0x4902 40BC		McBSP3
	0x4902 60BC		McBSP4
Description	McBSPLP sidetone select register.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																					SIDETONEEN	OCH1ASSIGN			OCH0ASSIGN			ICH1ASSIGN		ICH0ASSIGN	

Bits	Field Name	Description	Type	Reset
31:11	RESERVED	Read returns 0x0.	R	0x000000
10	SIDETONEEN	Sidetone mode enable. 0x0: Sidetone disabled. 0x1: Sidetone enabled.	RW	0x0
9:7	OCH1ASSIGN	Map the data for the speaker out channels to one of the McBSP channels (1 out of 8 channels)	RW	0x1

Bits	Field Name	Description	Type	Reset
6:4	OCH0ASSIGN	Map the data for the speaker out channels to one of the McBSP channels (1 out of 8 channels)	RW	0x0
3:2	ICH1ASSIGN	Map the data from digital microphone channels to one of the McBSP channels (1 out of 4 channels)	RW	0x1
1:0	ICH0ASSIGN	Map the data from digital microphone channels to one of the McBSP channels (1out of 4 channels)	RW	0x0

Table 21-132. Register Call Summary for Register MCBSPLP_SSELCR_REG

McBSP Functional Description • SIDETONE Interface: [0] [1] [2] [3] [4] [5] [6]
McBSP Basic Programming Model • SIDETONE Activation Procedure: [7] [8] [9] [10] [11]
McBSP Register Manual • McBSP Register Mapping Summary: [12] [13] [14] [15] [16]

Table 21-133. MCBSPLP_STATUS_REG

Address Offset	0x0000 00C0		
Physical Address	0x4807 40C0	**Instance**	McBSP1
	0x4809 60C0		McBSP5
	0x4902 20C0		McBSP2
	0x4902 40C0		McBSP3
	0x4902 60C0		McBSP4
Description	McBSPLP status register.		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																															CLKMUXSTATUS

Bits	Field Name	Description	Type	Reset
31:1	RESERVED	Read returns 0x0.	R	0x00000000
0	CLKMUXSTATUS	When going to/exiting from idle mode, the clock for the interface domain is switched between McBSP_ICLK and McBSP_FCLK to allow functioning in idle mode. This bit indicates that the status of clock switching and accesses to the McBSP registers are delayed during clock switching. This information is relevant only for the McBSPLP having AUDIOBUFFER.	R	0x0

Table 21-134. Register Call Summary for Register MCBSPLP_STATUS_REG

McBSP Register Manual • McBSP Register Mapping Summary: [0] [1] [2] [3] [4]

21.7.4 SIDETONE Register Description

Table 21-135. ST_REV_REG

Address Offset	0x0000 0000		
Physical Address	0x4902 8000	**Instance**	SIDETONE_McBSP2
	0x4902 A000		SIDETONE_McBSP3
Description	SIDETONE Revision number register		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																								REV							

Bits	Field Name	Description	Type	Reset
31:8	RESERVED	Read returns 0x0.	R	0x000000
7:0	REV	IP revision [7:4] Major revision [3:0] Minor revision Examples: 0x10 for 1.0, 0x21 for 2.1	R	See (1)

(1) TI internal data

Table 21-136. Register Call Summary for Register ST_REV_REG

McBSP Register Manual
- SIDETONE Register Mapping Summary: [0] [1]

Table 21-137. ST_SYSCONFIG_REG

Address Offset	0x0000 0010		
Physical Address	0x4902 8010	**Instance**	SIDETONE_McBSP2
	0x4902 A010		SIDETONE_McBSP3
Description	SIDETONE System Configuration register		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																															AUTOIDLE

Bits	Field Name	Description	Type	Reset
31:1	RESERVED	Read returns 0x0.	R	0x00000000
0	AUTOIDLE	Automatic McBSPi_ICLK clock gating 0x0: McBSPi_ICLK clock auto-gating disabled. 0x1: McBSPi_ICLK clock auto-gating enabled.	RW	0x1

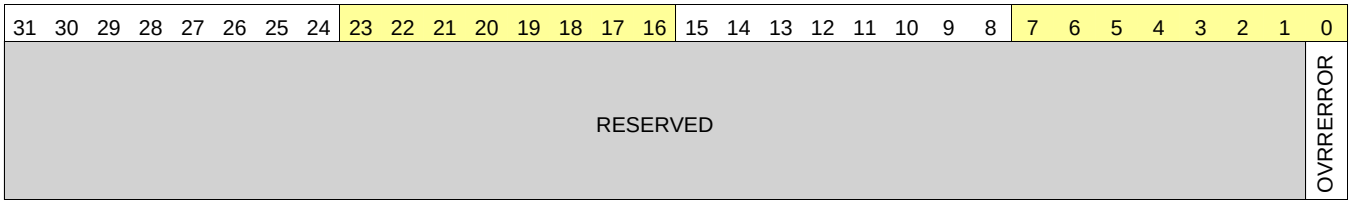
Table 21-138. Register Call Summary for Register ST_SYSCONFIG_REG

McBSP Integration
- Clocks: [0] [1]

McBSP Register Manual
- SIDETONE Register Mapping Summary: [2] [3]

Table 21-139. ST_IRQSTATUS_REG

Address Offset	0x0000 0018		
Physical Address	0x4902 8018	**Instance**	SIDETONE_McBSP2
	0x4902 A018		SIDETONE_McBSP3
Description	SIDETONE Interrupt Status Register		
Type	RW		



Bits	Field Name	Description	Type	Reset
31:1	RESERVED	Read returns 0x0.	R	0x00000000
0	OVRRERROR	Over-run error has occurred. New data to be processed has arrived before the previous one has ended. Writing 1 to this bit clears the bit.	RW	0x0

Table 21-140. Register Call Summary for Register ST_IRQSTATUS_REG

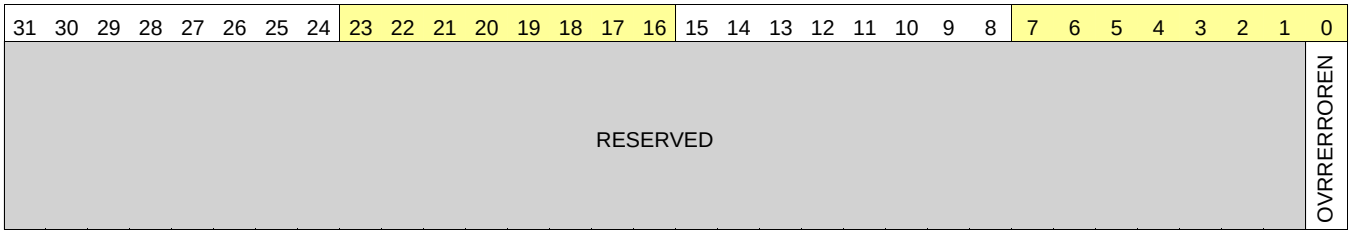
McBSP Integration
- Interrupt Requests: [0] [1] [2] [3]

McBSP Functional Description
- Interrupt Operation: [4]

McBSP Register Manual
- SIDETONE Register Mapping Summary: [5] [6]

Table 21-141. ST_IRQENABLE_REG

Address Offset	0x0000 001C		
Physical Address	0x4902 801C	**Instance**	SIDETONE_McBSP2
	0x4902 A01C		SIDETONE_McBSP3
Description	SIDETONE Interrupt enable register		
Type	RW		



Bits	Field Name	Description	Type	Reset
31:1	RESERVED	Read returns 0x0.	R	0x00000000
0	OVRRERROREN	Over-run error interrupt enable bit.	RW	0x0

Table 21-142. Register Call Summary for Register ST_IRQENABLE_REG

McBSP Integration
- Interrupt Requests: [0] [1] [2]

McBSP Functional Description
- Interrupt Operation: [3]

Table 21-142. Register Call Summary for Register ST_IRQENABLE_REG (continued)

McBSP Register Manual • SIDETONE Register Mapping Summary: [4] [5]

Table 21-143. ST_SGAINCR_REG

Address Offset	0x0000 0024		
Physical Address	0x4902 8024	**Instance**	SIDETONE_McBSP2
	0x4902 A024		SIDETONE_McBSP3
Description	Sidetone gain control register		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CH1GAIN																CH0GAIN															

Bits	Field Name	Description	Type	Reset
31:16	CH1GAIN	Second sidetone channel gain	RW	0x0000
15:0	CH0GAIN	First sidetone channel gain	RW	0x0000

Table 21-144. Register Call Summary for Register ST_SGAINCR_REG

McBSP Functional Description • Data Processing: [0]
McBSP Basic Programming Model • SIDETONE Initialization Procedure: [1] [2]
McBSP Register Manual • SIDETONE Register Mapping Summary: [3] [4]

Table 21-145. ST_SFIRCR_REG

Address Offset	0x0000 0028		
Physical Address	0x4902 8028	**Instance**	SIDETONE_McBSP2
	0x4902 A028		SIDETONE_McBSP3
Description	Sidetone FIR coefficients control register		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																FIRCOEFF															

Bits	Field Name	Description	Type	Reset
31:16	RESERVED	Read returns 0x0.	R	0x0000
15:0	FIRCOEFF	FIR coefficients control register (the coefficients are programmed by successive write sequence of all 128 FIR coefficients) The write sequence should start with the coefficient 0. In order to enable the write to this register the COEFFWREN bit in SSELCR_REG should be set to one. When this bit is set the read operation will return only the last written value. After a complete FIR coefficients write sequence the COEFFWREN should be set to zero. A read sequence from SFIRCR_REG while COEFFWREN is set to zero will return the coefficients values starting from 0 to 127. The write coefficient address is set to 0 by the change of COEFFWREN from 0 to 1. The read coefficient address is set to 0 by the change of COEFFWREN from 1 to 0	RW	0x0000

Table 21-146. Register Call Summary for Register ST_SFIRCR_REG

McBSP Basic Programming Model

- SIDETONE Initialization Procedure: [0] [1]
- SIDETONE FIR Coefficients Writing: [2]
- SIDETONE FIR Coefficients Reading: [3]

McBSP Register Manual

- SIDETONE Register Mapping Summary: [4] [5]

Table 21-147. ST_SSELCR_REG

Address Offset	0x0000 002C		
Physical Address	0x4902 802C	**Instance**	SIDETONE_McBSP2
	0x4902 A02C		SIDETONE_McBSP3
Description	Sidetone select register		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																													COEFFWRDONE	COEFFWREN	SIDETONEEN

Bits	Field Name	Description	Type	Reset
31:3	RESERVED	Read returns 0x0.	R	0x00000000
2	COEFFWRDONE	Write FIR coefficients completed. 0x0: FIR coefficients not loaded 0x1: FIR coefficients loaded	R	0x0

Bits	Field Name	Description	Type	Reset
1	COEFFWREN	Write enable FIR coefficients. 0x1: If a 0 to 1 transition on this bit occurs, the write coefficient index is reset. When this bit is 1, all coefficients can be written in SFIRCR_REG performing 128 write accesses with SIDETONEEN 1. First access writes coefficient index 0 Read access in this case returns the last written value. 0x0: If a 1 to 0 transition on this bit occurs, the read coefficient index is reset. When this bit is 0, all coefficients can be read from SFIRCR_REG by performing 128 read accesses with SIDETONEEN 0. First access reads coefficient index 0.	RW	0x0
0	SIDETONEEN	Sidetone mode enable. 0x0: Sidetone disabled 0x1: Sidetone enabled	RW	0x0

Table 21-148. Register Call Summary for Register ST_SSELCR_REG

McBSP Functional Description • Data Processing Path: [0] • Data Processing: [1] [2] [3]
McBSP Basic Programming Model • SIDETONE Activation Procedure: [4] • SIDETONE Initialization Procedure: [5] [6] • SIDETONE FIR Coefficients Writing: [7] [8] • SIDETONE FIR Coefficients Reading: [9]
McBSP Register Manual • SIDETONE Register Mapping Summary: [10] [11]

MMC/SD/SDIO Card Interface

This chapter describes the features and functions of the multimedia card/SD/SD I/O (MMC/SDIO) card interface for the OMAP34xx device.

Topic **Page**

22.1 MMC/SD/SDIO Overview 3076
22.2 MMC/SD/SDIO Environment 3080
22.3 MMC/SD/SDIO Integration 3088
22.4 MMC/SD/SDIO Functional Description 3096
22.5 MMC/SD/SDIO Basic Programming Model 3105
22.6 MMC/SD/SDIO Use Cases and Tips 3120
22.7 MMC/SD/SDIO Register Manual 3134

22.1 MMC/SD/SDIO Overview

The multimedia card high-speed/SD/SD I/O (MMC/SD/SDIO) host controller provides an interface between a local host (LH) such as a microprocessor unit (MPU) or digital signal processor (DSP) and either MMC, SD memory cards, or SDIO cards and handles MMC/SD/SDIO transactions with minimal LH intervention.

The application interface manages transaction semantics. The MMC/SD/SDIO host controller deals with MMC/SD/SDIO protocol at transmission level, data packing, adding cyclic redundancy checks (CRC), start/end bit, and checking for syntactical correctness.

The application interface can send every MMC/SD/SDIO command and either poll for the status of the adapter or wait for an interrupt request, which is sent back in case of exceptions or to warn of end of operation.

The application interface can read card responses or flag registers. It can also mask individual interrupt sources. All these operations can be performed by reading and writing control registers. The MMC/SD/SDIO host controller also supports two DMA channels.

There are three MMC/SD/SDIO host controllers inside the device: Figure 22-1 gives an overview of the MMC/SD/SDIO controller instances 1 and 3, and Figure 22-2 gives an overview of the MMC/SD/SDIO2 controller instance.

Figure 22-1. MMC/SD/SDIO1 and 3 Overview

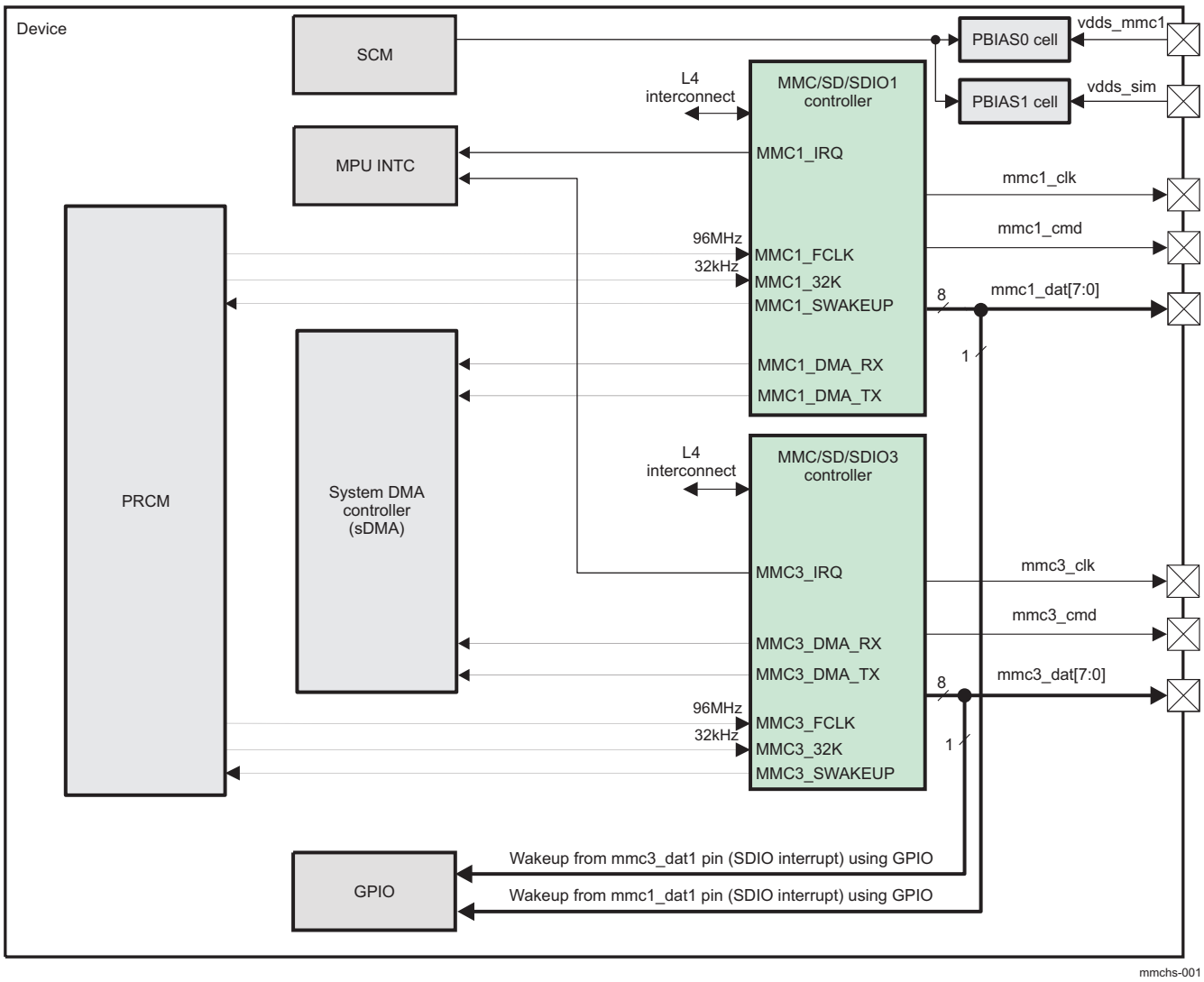
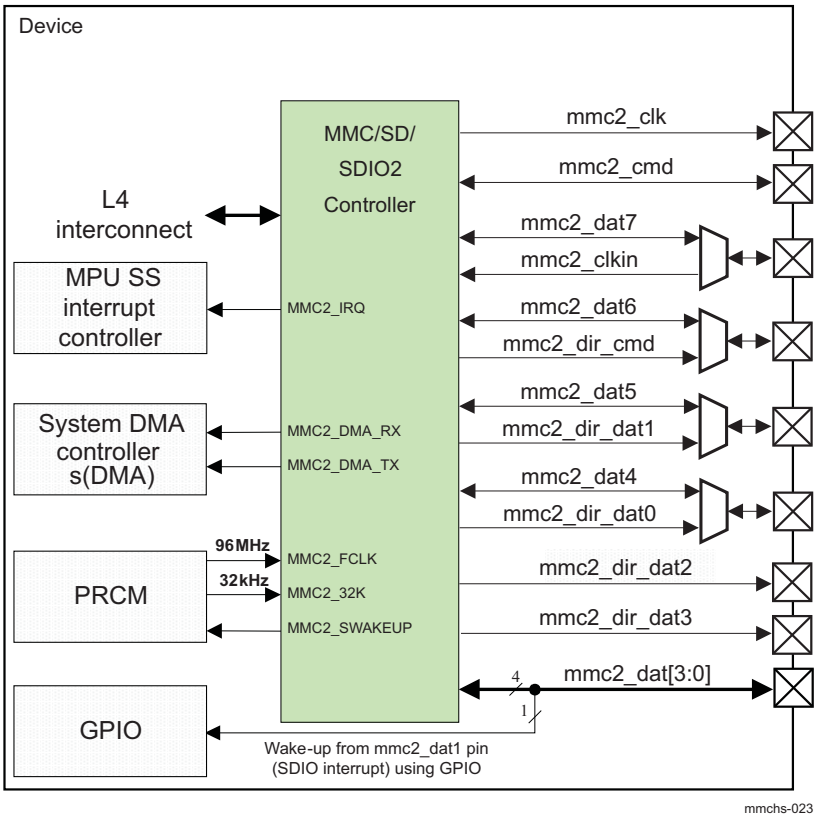


Figure 22-2. MMC/SD/SDIO2 Overview



22.1.1 MMC/SD/SDIO Features

The main features of the MMC/SD/SDIO host controller are:

- Full compliance with MMC command/response sets as defined in the *Multimedia Card System Specification,* v4.2 including high-capacity (size > 2GB) cards HC MMC.
- Full compliance with SD command/response sets as defined in the *SD Memory Card Specifications*, v2.0 including high-capacity SDHC cards up to 32 GB.
- Full compliance with SDIO command/response sets and interrupt/read-wait mode as defined in the *SDIO Card Specification, Part E1*, v1.10
- Compliance with sets as defined in the *SD Card Specification, Part A2, SD Host Controller Standard Specification*, v1.00
- Full compliance with MMC bus testing procedure as defined in the *Multimedia Card System Specification*, v4.2
- Full compliance with CE-ATA command/response sets as defined in the *CE-ATA Standard Specification*
- Full compliance with ATA for MMCA specification
- Flexible architecture allowing support for new command structure
- Support:
  - 1-bit or 4-bit transfer mode specifications for SD and SDIO cards
  - 1-bit, 4-bit, or 8-bit transfer mode specifications for MMC cards
- Built-in 1024-byte buffer for read or write
- 32-bit-wide access bus to maximize bus throughput
- Single interrupt line for multiple interrupt source events
- Two slave DMA channels (1 for TX, 1 for RX)
- Designed for low power

- Programmable clock generation
- Support SDIO Read Wait and Suspend/Resume functions
- Support Stop at block gap
- Support command completion signal (CCS) and command completion signal disable (CCSD) management as specified in the *CE-ATA Standard Specification*

The known limitations are as follows:

- No built-in hardware support for error correction codes (ECC). See the *Multimedia Card System Specification*, v4.2, and the *SD Memory Card Specifications*, v2.0, for details about ECC.
- The maximum block size defined in the *SD Memory Card Specifications*, v2.0, that the host driver can read and write to the buffer in the host controller is 2048 bytes. MMC supports a maximum block size of 1024 bytes. Up to 512 byte transfers, the buffer in MMC is considered as a double buffering with ping-pong management; half of the buffer can be written while the other part is read. For 512 to 1024 byte transfers, the entire buffer is dedicated to the transfer (read only or write only).

The differences between the MMC/SD/SDIO host controllers and a Standard SD host controller are defined by the *SD Card Specification, Part A2, SD Host Controller Standard Specification*, v1.00, as follows:

- The MMC/SD/SDIO host controllers support MMC cards.
- The MMC/SD/SDIO host controller is defined as a DMA slave device. Standard SD host controller is defined as DMA master controller that can start and stop a DMA transfer. MMC/SD/SDIO host controllers support DMA transfers through slave DMA requests.
- The clock divider in MMC/SD/SDIO host controller supports a wider range of frequency than specified in the *SD Memory Card Specifications*, v2.0. The MMC/SD/SDIO host controller supports odd and even clock ratio.
- The MMC/SD/SDIO host controller supports configurable busy timeout.

22.2 MMC/SD/SDIO Environment

One MMC/SD/SDIO host controller can support one MMC memory card, one SD memory card, or one SDIO card.

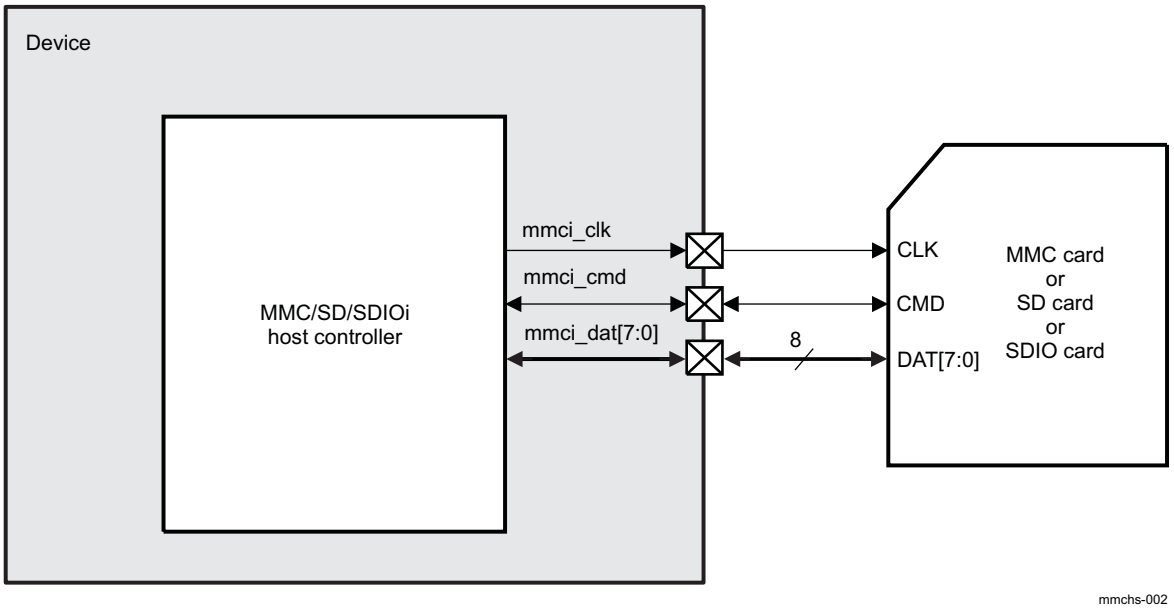
Other combinations (for example, two SD cards, one MMC card, and one SD card) are not supported through a single controller.

- The first controller (MMC/SD/SDIO1) integrates an internal transceiver that allows a direct connection to the MMC/SD/SDIO card (1.8 and 3V), without external transceiver.
- The second controller (MMC/SD/SDIO2) allows connecting MMC/SD/SDIO cards (only 1.8V cards) or an external device that uses the MMC/SD/SDIO interface (WLAN device for example). The second instance also supports an external transceiver and provides direction signals for data and command. Using an external transceiver device precludes 8-bit transfer mode.
- The third controller (MMC/SD/SDIO3) allows connecting MMC/SD/SDIO cards (only 1.8V cards) or an external device that uses the MMC/SD/SDIO interface (Wireless USB card for example). This interface is used without external transceiver.

22.2.1 MMC/SD/SDIO Connected to an MMC, an SD, or an SDIO Card

Figure 22-3 shows MMC/SD/SDIOi host controller, instance 1, 2 or 3, connected to an MMC, an SD, or an SDIO card and its related external connections.

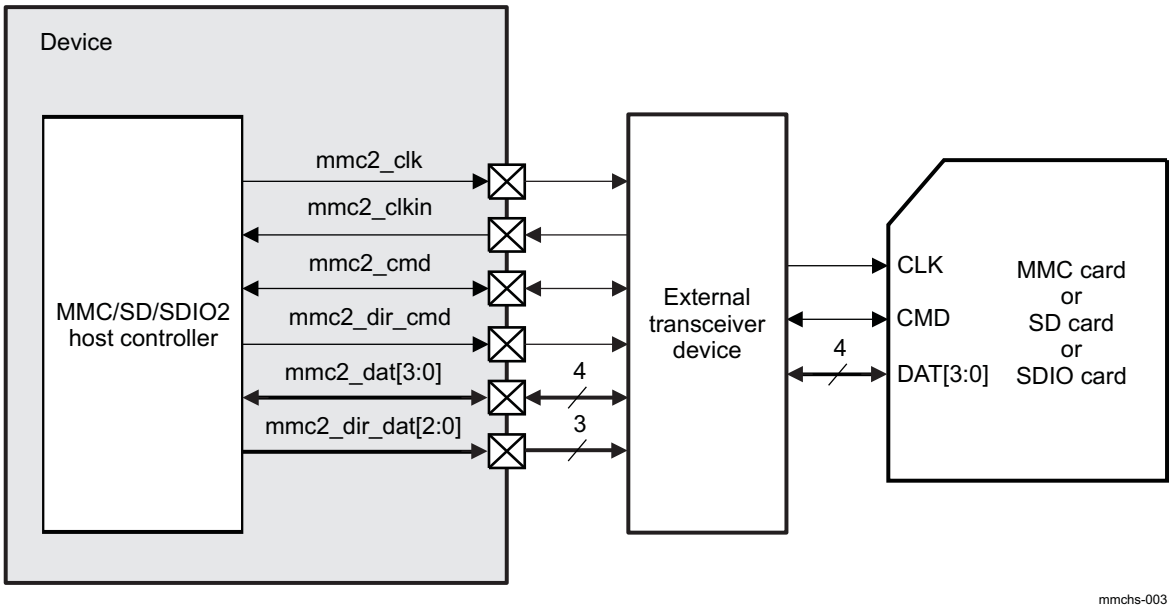
Figure 22-3. MMC/SD/SDIO Connected to an MMC, an SD, or an SDIO Card Without External Transceiver



22.2.2 MMC/SD/SDIO Connected to an MMC, an SD, or an SDIO Card Through an External Transceiver Device

This connection is supported only by the MMC/SD/SDIO2 host controller. Figure 22-4 shows the MMC/SD/SDIO2 host controller connected to an MMC, an SD, or an SDIO card through an external transceiver device.

Figure 22-4. MMC/SD/SDIO2 Connected to an MMC, an SD, or an SDIO Card with External Transceiver



22.2.3 MMC/SD/SDIO Functional Interfaces

22.2.3.1 Basic MMC/SD/SDIOi Pins Without External Transceiver

Figure 22-5 shows the MMC/SD/SDIOi host controller interface signals (instance 1, 2 or 3).

Figure 22-5. MMC/SD/SDIOi Interface Signals

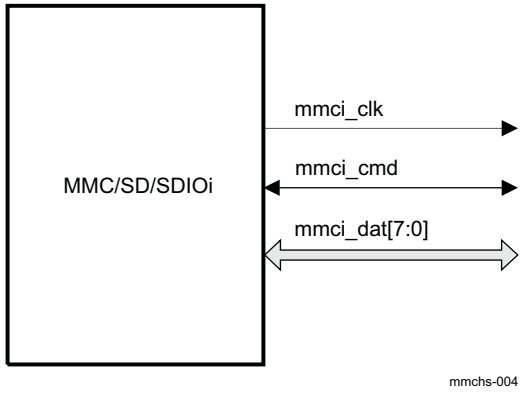


Table 22-1 describes the MMC/SD/SDIOi inputs/outputs.

Table 22-1. MMC/SD/SDIOi I/O Description

Signal Name	I/O[1]	Description	Reset Value
mmci_clk	O	External clock for MMC/SD/SDIO card[2]	0
mmci_cmd	I/O	Command signal	0
mmci_dat[7:0]	I/O	Data signals	0

(1) I = Input; O = Output
(2) This output signal is also used as re-timing input.

22.2.3.2 Basic MMC/SD/SDIO2 Pins with External Transceiver

Figure 22-6 shows the MMC/SD/SDIO2 host controller interface signals.

Figure 22-6. MMC/SD/SDIO2 Interface Signals

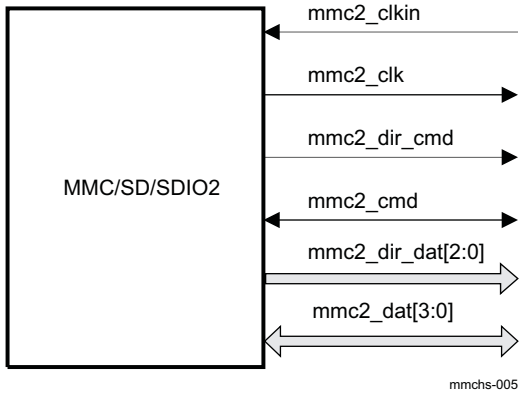


Table 22-2 describes the MMC/SD/SDIO2 inputs/outputs.

Table 22-2. MMC/SD/SDIO2 I/O Description

Signal Name	I/O	Description	Reset Value
mmc2_clk	O	External clock for MMC/SD/SDIO card	0
mmc2_clkin	I	Input clock from MMC/SD/SDIO card	0
mmc2_cmd	I/O	Command signal	0
mmc2_dir_cmd	O	Direction control for mmc2_cmd signal case an external transceiver is used (high when transmit, low when receive)	0
mmc2_dat[3:0]	I/O	Data signals	0
mmc2_dir_dat0	O	Direction control for mmc2_dat0 signal when an external transceiver is used (high when transmit, low when receive)	0
mmc2_dir_dat1	O	Direction control for mmc2_dat1 and mmc2_dat3 signal when an external transceiver is used (high when transmit, low when receive)	0
mmc2_dir_dat2	O	Direction control for mmc2_dat2 signal when an external transceiver is used (high when transmit, low when receive)	0
mmc2_dir_dat3	O	Direction control for mmc2_dat[7:4] signal when an external transceiver is used (high when transmit, low when receive). Unusable on the device because mmci_dat[7:4] are muxed with other direction control signals. See Chapter 7, *System Control Module* for further details on the pin multiplexing.	0

22.2.3.3 MMC/SD/SDIO Protocol and Data Format

The bus protocol between the MMC/SD/SDIOi host controller and the card is message-based. Each message is represented by one of the following parts:

Command: a command starts an operation. The command is transferred serially from the MMC/SD/SDIO host controller to the card on the mmci_cmd line.

Response: a response is an answer to a command. The response is sent from the card to the MMC/SD/SDIO host controller. It is transferred serially on the mmci_cmd line.

Data: data are transferred from the MMC/SD/SDIOi host controller to the card or from a card to the MMC/SD/SDIO host controller using the DATA lines.

Busy: the mmci_dat0 signal is maintained low by the card as far as it is programming the data received.

CRC status: CRC result is sent by the card through the mmci_dat0 line when executing a write transfer. In the case of transmission error, occurring on any of the active data lines, the card sends a negative CRC status on mmci_dat0. In the case of successful transmission, over all active data lines, the card sends a positive CRC status on mmci_dat0 and starts the data programming procedure.

22.2.3.3.1 Protocol

There are two types of data transfer:

- Sequential operation
- Block-oriented operation

There are specific commands for each type of operation (sequential or block-oriented).

See the *Multimedia Card System Specification*, v4.2, the *SD Memory Card Specifications*, v2.0, and the *SDIO Card Specification, Part E1*, August 2004, for details about commands and programming sequences supported by the MMC, SD, and SDIO cards.

Figure 22-7 and Figure 22-8 show how sequential operations are defined. Sequential operation is only for 1-bit transfer and initiates a continuous data stream. The transfer terminates when a stop command follows on the mmci_cmd line.

CAUTION

Stream commands are supported only by MMC cards.

Figure 22-7. Sequential Read Operation (MMC Cards Only)

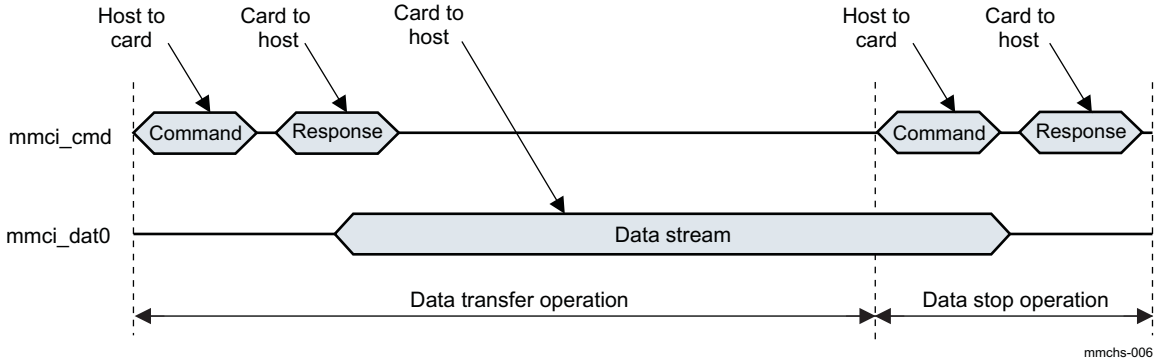


Figure 22-8. Sequential Write Operation (MMC Cards Only)

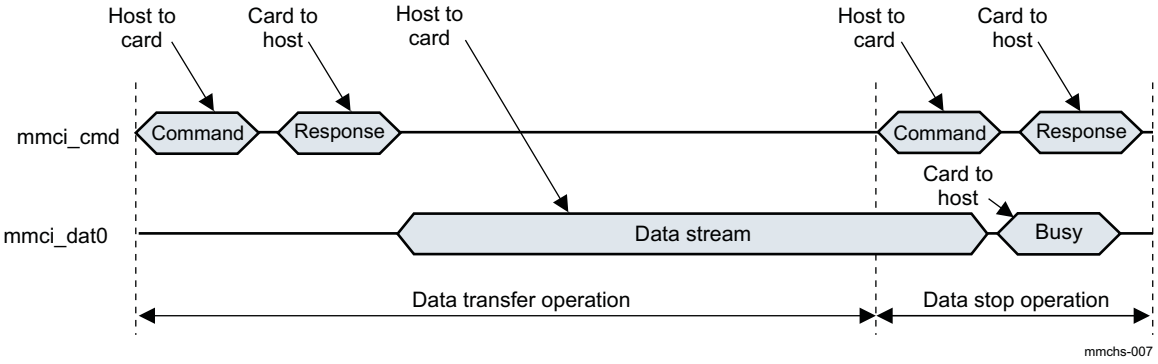


Figure 22-9 and Figure 22-10 show how multiple block-oriented operations are defined. A multiple block-oriented operation sends a data block plus CRC bits. The transfer terminates when a stop command follows on the mmci_cmd line. These operations are available for all kinds of cards.

Figure 22-9. Multiple Block Read Operation

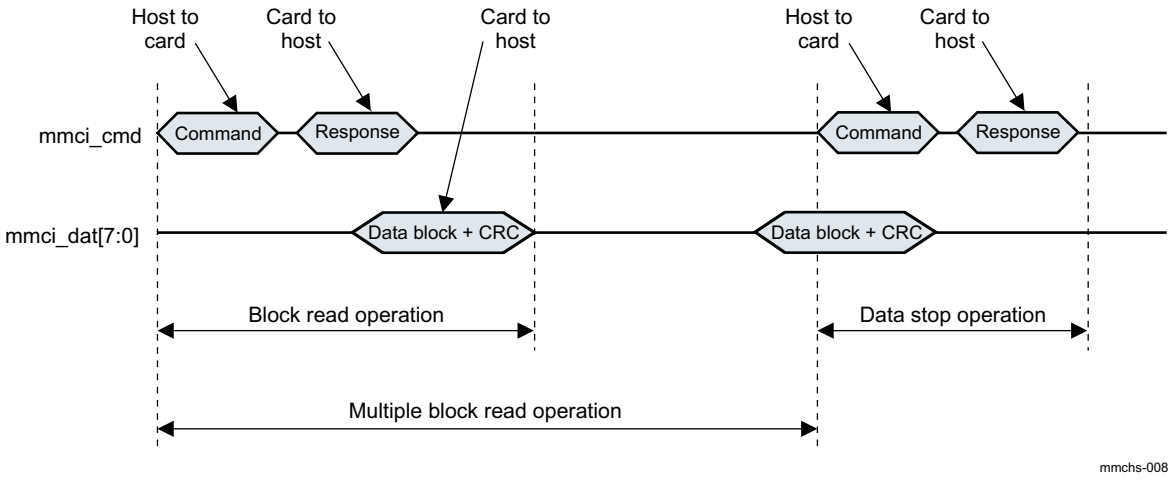
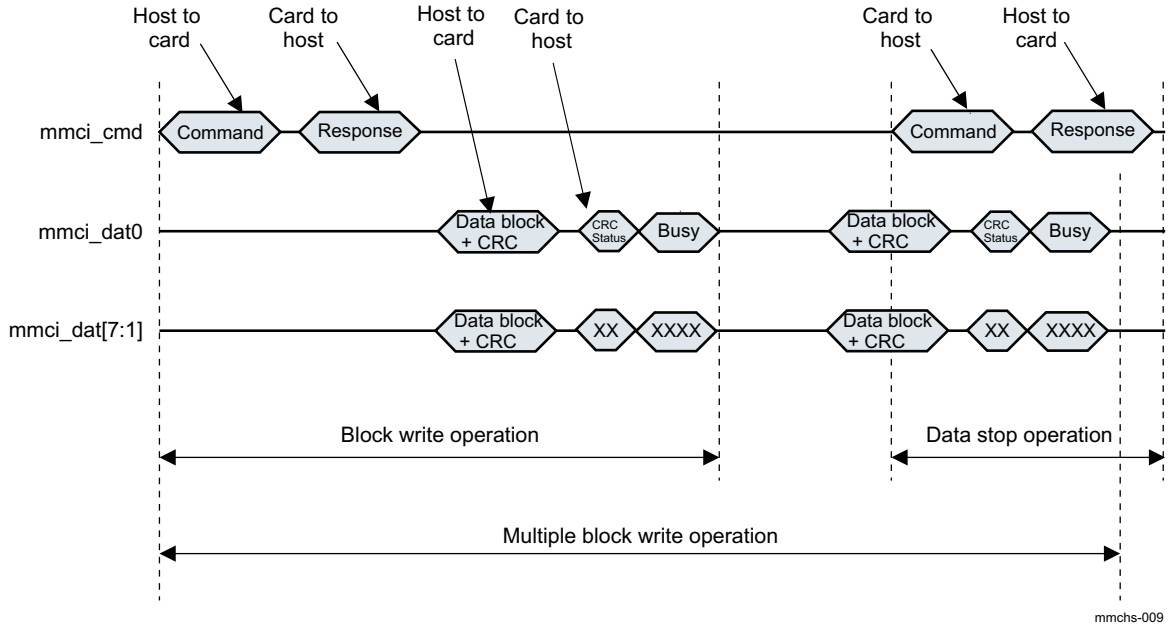


Figure 22-10. Multiple Block Write Operation with Card Busy Signal



Notes:

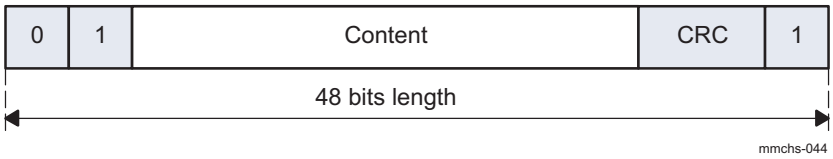
1. The card busy signal is not always generated by the card; the previous examples show a particular case.
2. It is software responsibility to do a software reset (set MMCi.MMCHS_SYSCTL[26] SRD bit to 0x1) after data timeout to ensure mmci_clk is stopped
3. For multiblock transfer, and especially for MMC cards, you can abort a transfer without using a stop command. Use a CMD23 before data transfer to define the number of blocks that will be transferred, then the transfer stops automatically after the last block (if the MMC card supports this feature).

22.2.3.3.2 Data Format

Coding Scheme for Command Token

Command tokens always start with 0 and end with 1. The second bit is a transmitter bit: 1 for a host command. The content is the command index (coded by 6 bits) and an argument (for example, an address), coded by 32 bits. The content is protected by 7-bit CRC checksum (see Figure 22-11).

Figure 22-11. Command Token Format



Coding Scheme for Response Token

Response tokens always start with 0 and end with a 1. The second bit is a transmitter bit: 0 for a card response. The content is different for each type of response (R1, R2, R3, R4, and R5 R6 [for SDIO]) and the content is protected by 7-bit CRC checksum (see Figure 22-12 and Figure 22-13). Depending on the type of commands sent to the card, the MMCHS_CMD register must be configured differently to avoid false CRC or index errors to be flagged on command response (see Table 22-3). For more details about response types, see the *Multimedia Card System Specification*, v4.2, the *SD Memory Card Specifications*, v2.0, or the *SDIO Card Specification, Part E1*, v1.10.

Figure 22-12. Response Token Format (R1, R3, R4, R5, R6)

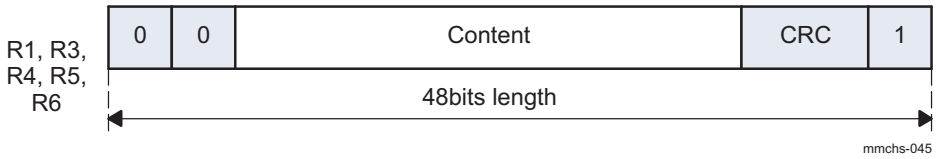


Figure 22-13. Response Token Format (R2)

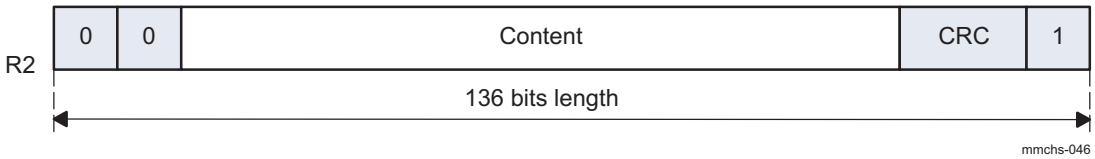


Table 22-3. Relation Between Configuration and Name of Response Type[(1)]

Response Type MMCi.MMCHS_CMD[17:16] RSP_TYPE	Index Check Enable MMCi.MMCHS_CMD[20] CICE	CRC Check Enable MMCi.MMCHS_CMD[19] CCCE	Name of Response Type
00	0	0	No Response
01	0	1	R2
10	0	0	R3 (R4 for SD cards)
10	1	1	R1, R6, R5
11	1	1	R1b, R5b

(1) The MMC/SD/SDIOi host controller assumes that both clocks may be switched off, whatever the value set in the MMCi.MMCHS_SYSCONFIG[9:8] CLOCKACTIVITY bit.

Coding Scheme for Data Token

Data tokens always start with 0 and end with 1 (see Figure 22-14, Figure 22-15, and Figure 22-16).

Figure 22-14. Data Token Format for 1-Bit Transfers

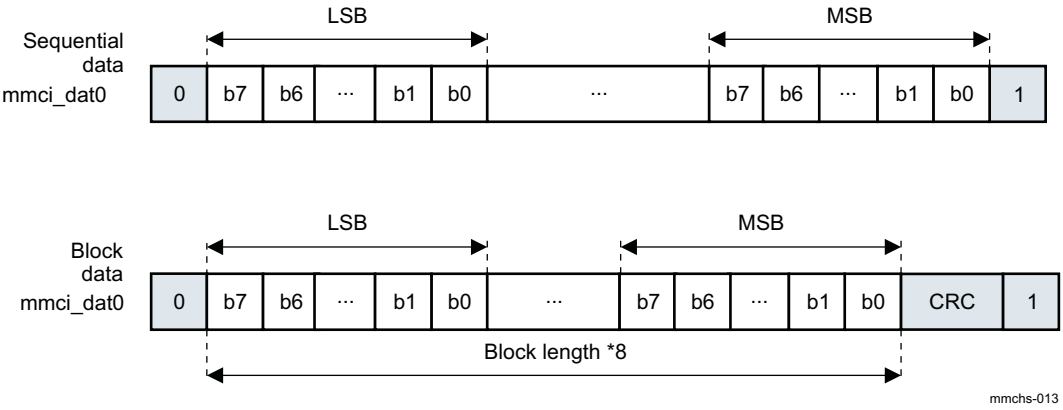


Figure 22-15. Data Token Format for 4-Bit Transfers

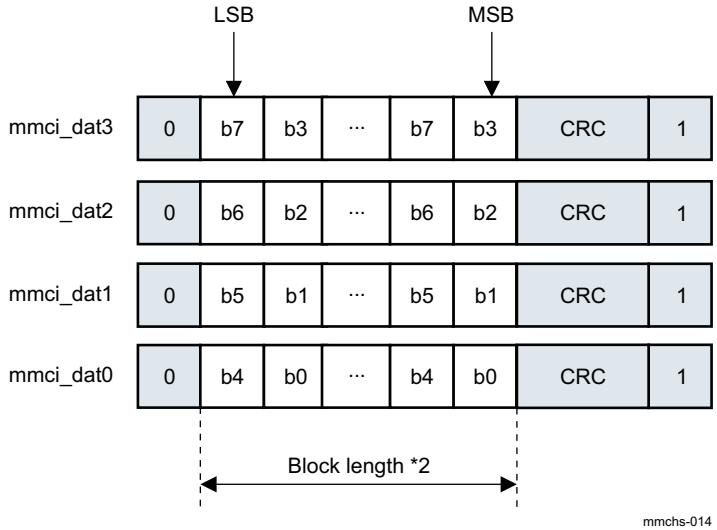
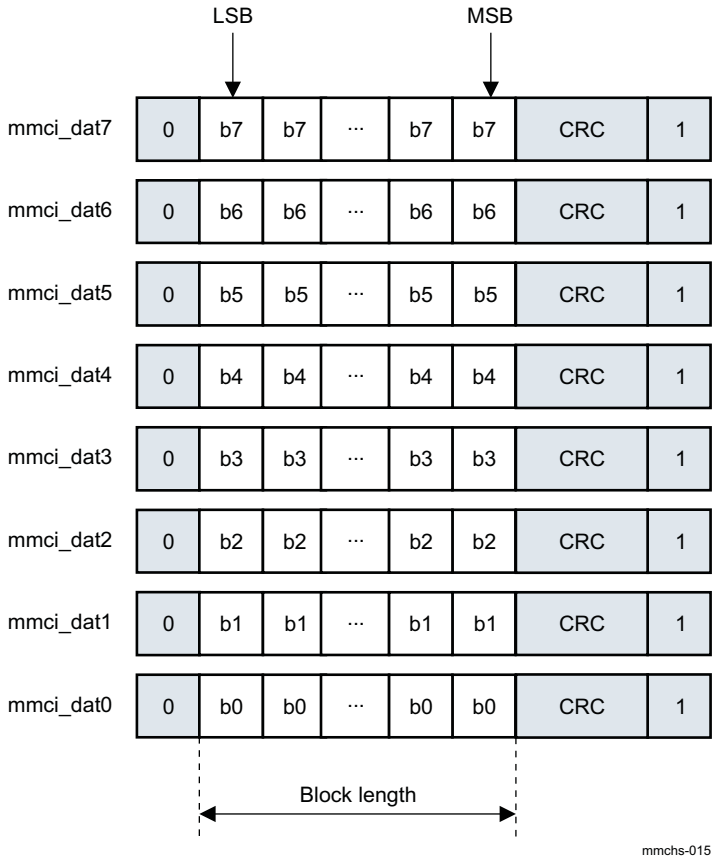


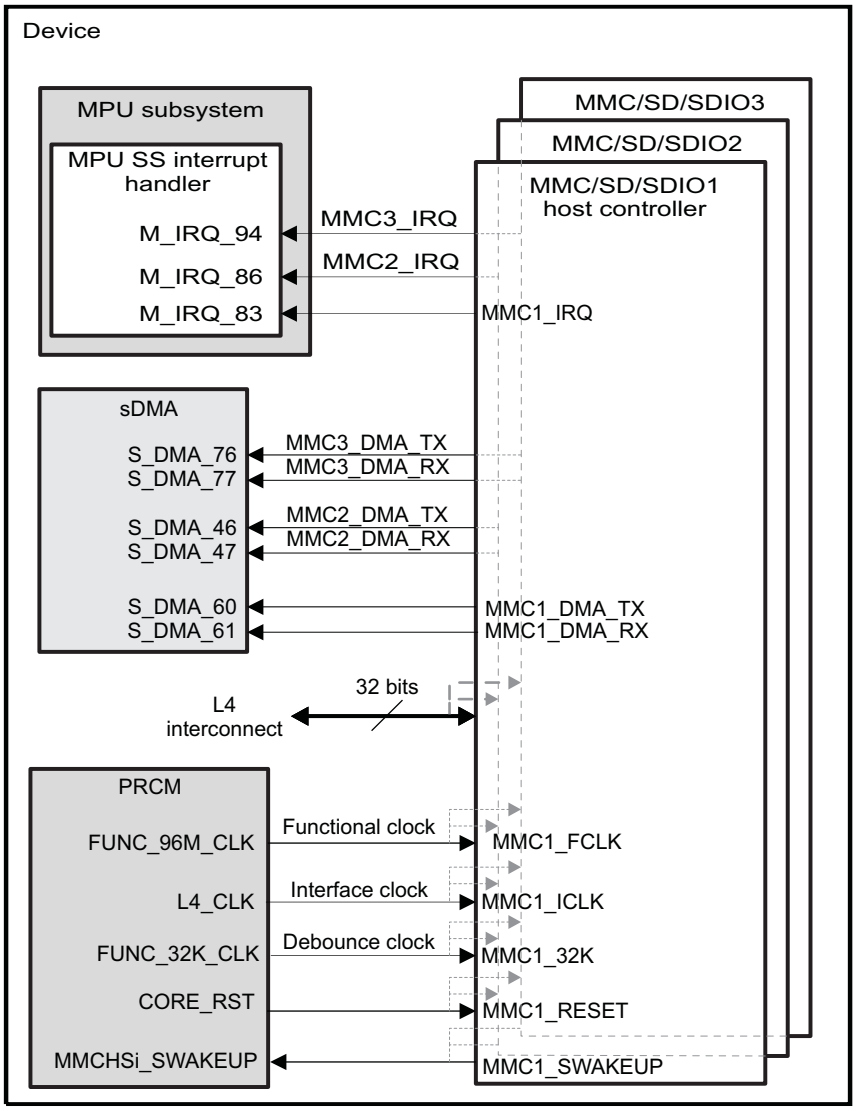
Figure 22-16. Data Token Format for 8-Bit Transfers



22.3 MMC/SD/SDIO Integration

Figure 22-17 shows the internal connections between the three instances of the MMC/SD/SDIO host controller and the other modules

Figure 22-17. MMC/SD/SDIO1 Integration



22.3.1 Clocking, Reset, and Power-Management Scheme

22.3.1.1 Clocks

22.3.1.1.1 Module Clocks

The MMC/SD/SDIO receives three clocks:

- A fixed functional clock of 96 MHz (the MMCi_FCLK) independent of the device external clock frequency
- An interface clock (the MMCi_ICLK) for interfacing with L4 interconnects (register accesses)
- A debounce clock, the MMCi_32K for reset process only

All these clocks are generated and controlled by the power, reset, and clock manager (PRCM) module (see Chapter 4, *Power Reset and Clock Management* for more information).

22.3.1.1.2 Power Management

The MMC/SD/SDIO host controller can enter into different modes and save power:

- Normal mode
- Idle mode

The two modes are mutually exclusive (the module can be in normal mode or in idle mode). The MMC/SD/SDIO host controller is compliant with the PRCM module handshake protocol.

Normal Mode

The autogating of interface and functional clocks occurs when the following conditions are met:

- The MMCi.MMCHS_SYSCONFIG[0] AUTOIDLE bit is set to 1 (I = 1 for MMC/SD/SDIO1, 2 for MMC/SD/SDIO2 and 3 for MMC/SD/SDIO3 instances).
- There is no transaction on the MMC interface.

The autogating of interface and functional clocks stops when the following conditions are met:

- A register access occurs through the L4 interconnect.
- A wake-up event occurs (an interrupt from a SDIO card).
- A transaction on the MMC/SD/SIO interface starts.

Then the MMC/SD/SDIO host controller enters in low-power state (MMCi_ICLK clock autogated) even if MMCi.MMCHS_SYSCONFIG[0] AUTOIDLE is set to 0.

The functional clock is internally switched off and only interconnect read and write accesses are allowed.

Idle Mode

The MMCi_ICLK and MMCi_FCLK clocks provided to MMC/SD/SDIO are switched off upon a PRCM module request. They are switched back upon module request.

The MMC/SD/SDIO host controller complies with the PRCM module handshaking protocol:

- Idle request from the system power manager
- Idle acknowledgment from the MMC/SD/SDIO host controller
- Wake-up request from the MMC/SD/SDIO host controller

The idle acknowledgment varies according to the MMCi.MMCHS_SYSCONFIG[4:3] SIDLEMODE bit field:

- 0x0: Force-idle mode. The MMC/SD/SDIO host controller acknowledges the system power manager request unconditionally.
- 0x1: No-idle mode. The MMC/SD/SDIO host controller ignores the system power manager request and behaves normally as if the request was not asserted.
- 0x2: Smart-idle mode. The MMC/SD/SDIO host controller acknowledges the system power manager request according to its internal state.

During the smart-idle mode period, the MMC/SD/SDIO host controller acknowledges that the MMCi_ICLK and MMCi_FCLK clocks may be switched off whatever the value set in the MMCi.MMCHS_SYSCONFIG[9:8] CLOCKACTIVITY field.

Transition from Normal Mode to Smart-Idle Mode

Smart-idle mode is enabled when the MMCi.MMCHS_SYSCONFIG[4:3] SIDLEMODE bit field is set to 0x2.

The MMC/SD/SDIOi host controller goes into idle mode when the PRCM issues an idle request, according to its internal activity.

During normal to idle mode transition, any access to the registers of the MMC/SD/SDIOi host controller generates an error as long as the MMCi_ICLK clock is alive. The PRCM.CM_IDLEST1_CORE[25] ST_MMC2 and (respectively the PRCM.CM_IDLEST1_CORE[24] ST_MMC1 bit) is set to 0x0 when the MMC/SD/SDIO2 module (respectively the MMC/SD/SDIO1 module) can be accessed.

The MMC/SD/SDIO host controller acknowledges the idle request from the PRCM after ensuring the following:

- The current multi/single-block transfer is completed.
- Any interrupt or DMA request is asserted.
- There is no card interrupt on mmci_dat[1] signal.

As long as the MMC/SD/SDIOi controller do not acknowledge the idle request, if an event occurs, the MMC/SD/SDIOi host controller can still generate an interrupt or a DMA request. In this case, the module ignores the idle request from the PRCM.

As soon as the MMC/SD/SDIOi controller acknowledges the idle request from the PRCM, the module does not assert any new interrupt or DMA request.

Wake-Up Event in Smart-Idle Mode

The wake-up feature is enabled when the following enable wake-up bits are set:

- MMCi.MMCHS_SYSCONFIG[2] ENAWAKEUP bit is set to 0x1
- MMCi.MMCHS_HCTL[24] IWE bit is set to 0x1
- MMCi.MMCHS_IE[8] CIRQ_ENABLE bit is set to 0x1

The wakeup is generated only in smart-idle mode only, when module is in idle mode.

Table 22-4 lists the supported cases in smart-idle mode.

Table 22-4. Smart Idle Mode and Wake-Up Capabilities

Mode	MMCi_ICLK clock	MMCi_FCLK clock	Wake-up Event
Card interrupt	May be switched off(1)	May be switched off(1)	The module sends an asynchronous wake-up request on detection of a card interrupt on mmci_dat[1] signal

(1) The MMC/SD/SDIOi host controller assumes that both clocks may be switched off, whatever the value set in the MMCi.MMCHS_SYSCONFIG[9:8] CLOCKACTIVITY bit.

Transition from Smart-Idle Mode to Normal Mode

The MMC/SD/SDIO host controller detects the end of the idle period when the PRCM deasserts the idle request.

For the wake-up event, there is a corresponding interrupt status in the MMCi.MMCHS_STAT register. The MMC/SD/SDIOi host controller operates the conversion between wake-up and interrupt (or DMA request) upon exit from smart-idle mode if the associated enable bit is set in the MMCi.MMCHS_ISE register.

Interrupts and wake-up events have independent enable/disable controls, accessible through the MMCi.MMCHS_HCTL and MMCi.MMCHS_ISE registers. The overall consistency must be ensured by software.

The interrupt status register MMCi.MMCHS_STAT is updated with the event that caused the wake-up in the CIRQ bit when the MMCi.MMCHS_IE[8] CIRQ_ENABLE associated bit is enabled.

Then, the wake-up event at the origin of the transition from smart-idle mode to normal mode is converted into its corresponding interrupt or DMA request. (The MMCi.MMCHS_STAT register is updated and the status of the interrupt signal changes.)

When the idle request from the PRCM is deasserted, the module switches back to normal mode. The module is fully operational.

Force-Idle Mode

Force-idle mode is enabled when the MMCi.MMCHS_SYSCONFIG[4:3] SIDLEMODE bit field is set to 0x0.

Force-idle mode is an idle mode where the MMC/SD/SDIOi host controller responds unconditionally to the idle request from the PRCM. Moreover, in this mode, the MMC/SD/SDIOi host controller unconditionally deasserts interrupts and DMA request lines asserted.

The transition from normal mode to force-idle mode does not affect the bits of the MMCi.MMCHS_STAT register.

In force-idle mode, the interrupt and DMA request lines are deasserted. MMCi_ICLK and MMCi_FCLK can be switched off.

CAUTION

In force-idle mode, an idle request from the PRCM during a command or a data transfer can lead to an unexpected and unpredictable result. When the module is idle, any access to the module generates an error as long as the MMCi_ICLK clock is alive.

The module exits the force-idle mode when the PRCM deasserts the idle request. Then the module switches back to normal mode. The module is fully operational. Interrupt and DMA request lines are optionally asserted one clock cycle later.

22.3.1.2 Resets

22.3.1.2.1 Hardware Reset

The module is reinitialized by the hardware when the active-low reset signal (CORE_RST), synchronous to the MMCi_ICLK clock is asserted on the input pin MMCi_RESET (see Chapter 4, *Power, Reset, and Clock Management* for more information).

A global status bit RESETDONE is provided in the status register MMCi.MMCHS_SYSSTATUS[0]. This bit is set to 1 when all the different clock domain resets (interface domain, functional domain, and 32K domain) have been released.

The MMCi.MMCHS_SYSSTATUS[0] RESETDONE bit can be monitored by the software to check if the module is ready-to-use after a hardware reset.

Note: Functional clock MMCi_FCLK, interface clock MMCi_ICLK, and debounce clock MMCi_32K must be provided to the module to allow the RESETDONE status bit to be set.

This hardware reset signal has a global reset action on the module. All configuration registers and all state-machines are reset in all clock domains.

22.3.1.2.2 Software Reset

The module is reinitialized by software through the MMCi.MMCHS_SYSCONFIG[1] SOFTRESET bit. This bit has the same action on the module logic as the hardware MMCi_RESET signal except for:

- Debounce logic
- MMCi.MMCHS_PSTATE, MMCi.MMCHS_CAPA, and MMCi.MMCHS_CUR_CAPA registers (see corresponding register descriptions)

The SOFTRESET bit is active high. The bit is automatically reinitialized to 0 by the hardware. The MMCi.MMCHS_SYSCTL[24] SRA bit has the same action as the SOFTRESET bit on the design.

The MMCi.MMCHS_SYSSTATUS[0] RESETDONE bit can be monitored by the software to check if the module is ready-to-use after a software reset.

Moreover, two partial software reset bits are provided:

- MMCi.MMCHS_SYSCTL[26] SRD bit
- MMCi.MMCHS_SYSCTL[25] SRC bit

These two reset bits are useful to reinitialize data or command processes respectively in case of line conflict. When set to 1, a reset process is automatically released when the reset completes:

- The MMCi.MMCHS_SYSCTL[26] SRD bit resets all finite state-machines and status management that handle data transfers on both the interface and functional side.
- The MMCi.MMCHS_SYSCTL[25] SRC bit resets all finite state-machines and status management that handle command transfers on both the interface and functional side.

22.3.1.3 Power Domain

MMC/SD/SDIOi power is supplied by the CORE power domain (see Chapter 4, *Power Reset and Clock Management* for more information).

When the MMC/SD/SDIOi power domain is off, the only way to wake up the power domain and different MMC/SD/SDIOi clocks is to monitor mmci_dat[1] input pin state via a different GPIO line for each MMC/SD/SDIO interface (see Chapter 25, *General-Purpose Interface*, for more information).

22.3.2 Hardware Requests

22.3.2.1 DMA Requests

The MMC/SD/SDIOi host controller can be interfaced with a DMA controller. At system level, the advantage is to discharge the LH of the data transfers. The module does not support wide DMA access (above 1024 bytes) for SD cards as specified in the *SD Card Specification, Part A2, SD Host Controller Standard Specification*, v1.00.

The DMA request is issued if the three following conditions are met:

- The MMCi.MMCHS_CMD[0] DE bit is set to 1 to trigger the initial DMA request (the write must be done when running the data transfer command).
- A command was emitted on the mmci_cmd line.
- There is enough space in the buffer of the MMC/SD/SDIOi host controller to write an entire block (BLEN writes).

DMA request lines are connected on the system DMA (sDMA) inputs:

- S_DMA_60 (MMC1_DMA_TX)
- S_DMA_61 (MMC1_DMA_RX)
- S_DMA_46 (MMC2_DMA_TX)
- S_DMA_47 (MMC2_DMA_RX)
- S_DMA_76 (MMC3_DMA_TX)
- S_DMA_77 (MMC3_DMA_RX)

22.3.2.1.1 DMA Receive Mode

In a DMA block read operation (single or multiple), the request signal MMCi_DMA_RX is asserted to its active level when a complete block is written in the buffer. The block size transfer is specified in the MMCi.MMCHS_BLK[10:0] BLEN field.

The MMCi_DMA_RX signal is deasserted to its inactive level when the sDMA has read one single word from the buffer.

Only one request is sent per block; the DMA controller can make a 1-shot read access or several DMA bursts, in which case the DMA controller must manage the number of burst accesses, according to block size BLEN field.

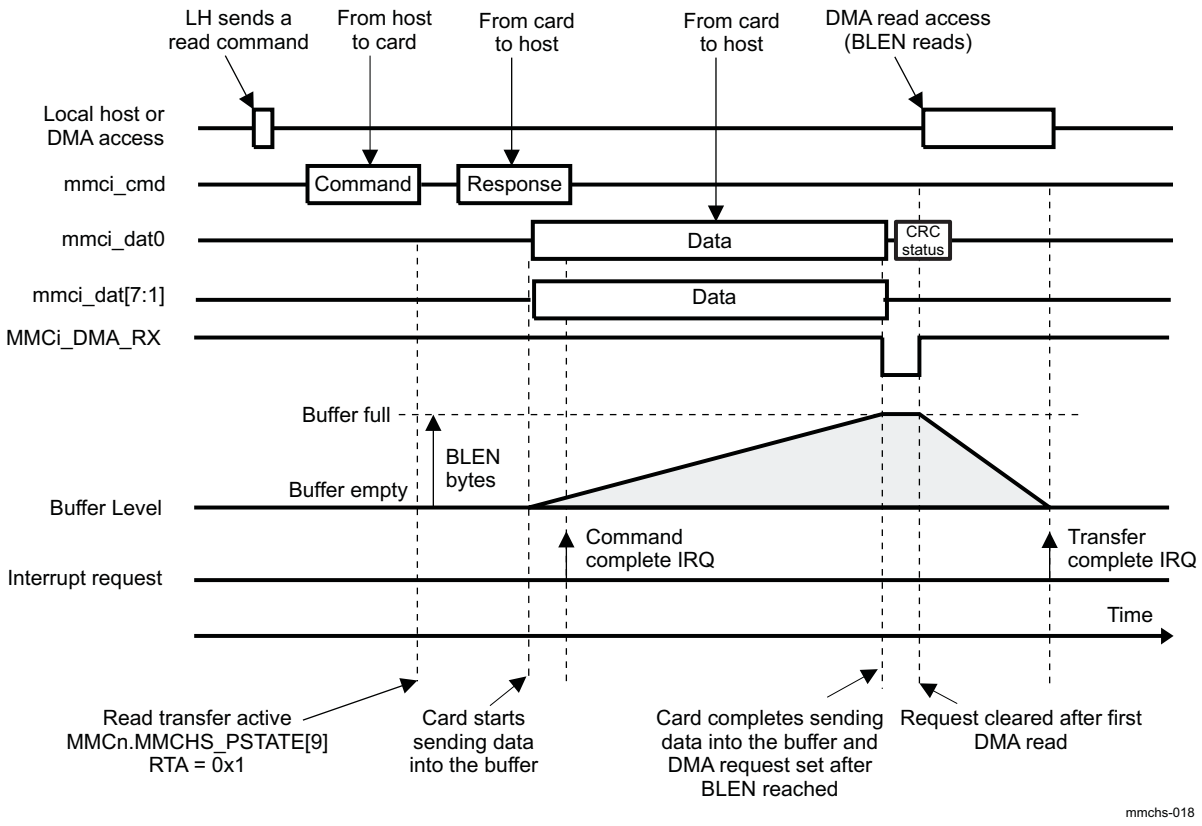
New DMA requests are internally masked if the sDMA has not read exactly BLEN bytes and a new complete block is not ready. As DMA accesses are in 32-bit, then the number of sDMA read is Integer(BLEN/4)+1.

The receive buffer never overflows. In multiple block transfers for block size above 512 bytes, when the buffer gets full, the mmci_clk clock signal (provided to the card) is momentarily stopped until the sDMA or the MPU performs a read access, which reads a complete block in the buffer.

Summary (see Figure 22-18):

- DMA transfer size = BLEN buffer size (maximum 1024 32-bit words) in one shot or by burst
- One DMA request per block

Figure 22-18. DMA Receive Mode



22.3.2.1.2 DMA Transmit Mode

In a DMA block write operation (single or multiple), the request signal MMCi_DMA_TX is asserted to its active level when a complete block is to be written to the buffer. The block size transfer is specified in the MMCi.MMCHS_BLK[10:0] BLEN field.

The MMCi_DMA_TX signal is deasserted to its inactive level when the sDMA has written one single word to the buffer.

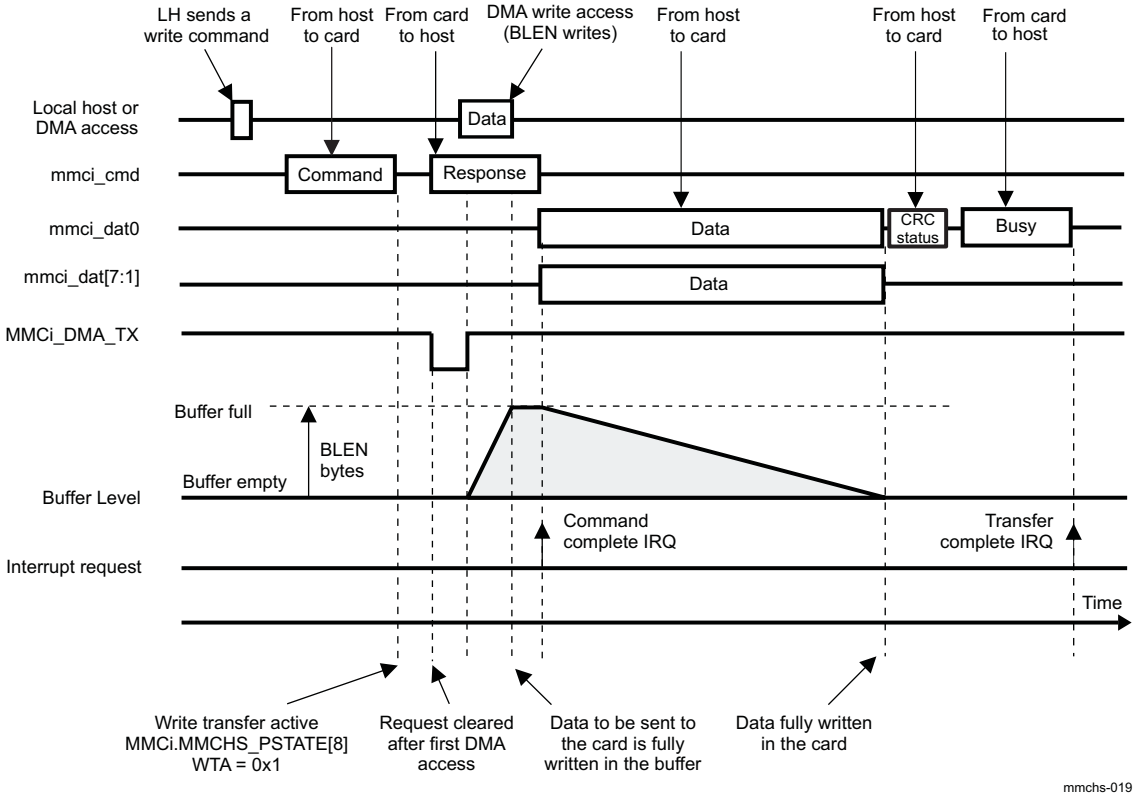
Only one request is sent per block; the DMA controller can make a 1-shot write access or multiple write DMA bursts, in which case the DMA controller must manage the number of burst accesses, according to block size BLEN field.

New DMA requests are internally masked if the sDMA has not written exactly BLEN bytes (as DMA accesses are in 32-bit, then the number of sDMA read is Integer(BLEN/4)+1) and if there is not enough memory space to write a complete block in the buffer.

Summary (see Figure 22-19):

- DMA transfer size = BLEN buffer size (maximum 1024 32-bit words) in one shot or by burst
- One DMA request per block

Figure 22-19. DMA Transmit Mode



22.3.2.2 Interrupt Requests

Several internal module events can generate an interrupt. Each interrupt has a status bit, an interrupt enable bit, and a signal status enable:

- The status of each type of interrupt is automatically updated in the MMCi.MMCHS_STAT register; it indicates which service is required.
- The interrupt status enable bits of the MMCi.MMCHS_IE register enable/disable the automatic update of the MMCi.MMCHS_STAT register on an event-by-event basis.
- The interrupt signal enable bits of the MMCi.MMCHS_ISE register enable/disable the transmission of an interrupt request on the interrupt line MMCi_IRQ (from the MMC/SD/SDIOi host controller to the MPU subsystem interrupt controller) on an event-by-event basis.

If an interrupt status is disabled in the MMCi.MMCHS_IE register, then the corresponding interrupt request is not transmitted, and the value of the corresponding interrupt signal enable in the MMCi.MMCHS_ISE register is ignored.

When an interrupt event occurs, the corresponding status bit is automatically set to 0x1 (the MMC/SD/SDIOi host controller updates the status bit) in the MMCi.MMCHS_STAT register. If later a mask is applied on the interrupt in the MMCi.MMCHS_ISE register, the interrupt request is deactivated.

When the interrupt source has not been serviced, if the interrupt status is cleared in the MMCi.MMCHS_STAT register and the corresponding mask is removed from the MMCi.MMCHS_ISE register, the interrupt status is not asserted again in the MMCi.MMCHS_STAT register and the MMC/SD/SDIOi host controller does not transmit an interrupt request.

CAUTION

If the buffer write ready interrupt (BWR) or the buffer read ready only interrupt (BRR) are not serviced and are cleared in the MMCi.MMCHS_STAT register, and the corresponding mask is removed, then the MMC/SD/SDIOi host controller will wait for the service of the interrupt without updating the status MMCi.MMCHS_STAT or transmitting an interrupt request.

The MMC/SD/SDIOi host controller supports interrupt-driven operation and polling.

There are one interrupt line for each instance of the MMC/SD/SDIOi host controller:

- MMC1_IRQ is connected to M_IRQ_83 for MMC/SD/SDIO1 instance.
- MMC2_IRQ is connected to M_IRQ_86 for MMC/SD/SDIO2 instance.
- MMC3_IRQ is connected to M_IRQ_94 for MMC/SD/SDIO3 instance.

22.3.2.2.1 Interrupt-Driven Operation

An interrupt enable bit must be set in the MMCi.MMCHS_IE register to enable the module internal source of interrupt.

When an interrupt event occurs, the single interrupt line is asserted and the LH must:

- Read the MMCi.MMCHS_STAT register to identify which event occurred.
- Write 1 into the corresponding bit of the MMCi.MMCHS_STAT register to clear the interrupt status and release the interrupt line (if a read is done after this write, this would return 0).

Note: In the MMCi.MMCHS_STAT register, Card Interrupt (CIRQ) and Error Interrupt (ERRI) bits cannot be cleared.
The MMCi.MMCHS_STAT[8] CIRQ status bit must be masked by disabling the MMCi.MMCHS_IE[8] CIRQ_ENABLE bit (set to 0x0), then the interrupt routine must clear SDIO interrupt source in SDIO card common control register (CCCR). See Chapter 5, *Interconnect* for more information.
The MMCi.MMCHS_STAT[15] ERRI bit is automatically cleared when all status bits in MMCi.MMCHS_STAT[31:16] are cleared.

22.3.2.2.2 Polling

When the interrupt capability of an event is disabled in the MMCi.MMCHS_ISE register, the interrupt line is not asserted:

- Software can poll the status bit in the MMCi.MMCHS_STAT register to detect when the corresponding event occurs.
- Writing 1 into the corresponding bit of the MMCi.MMCHS_STAT register clears the interrupt status and does not affect the interrupt line state.

Note: Refer to the previous note concerning CIRQ and ERRI bits clearing.

22.4 MMC/SD/SDIO Functional Description

22.4.1 Description

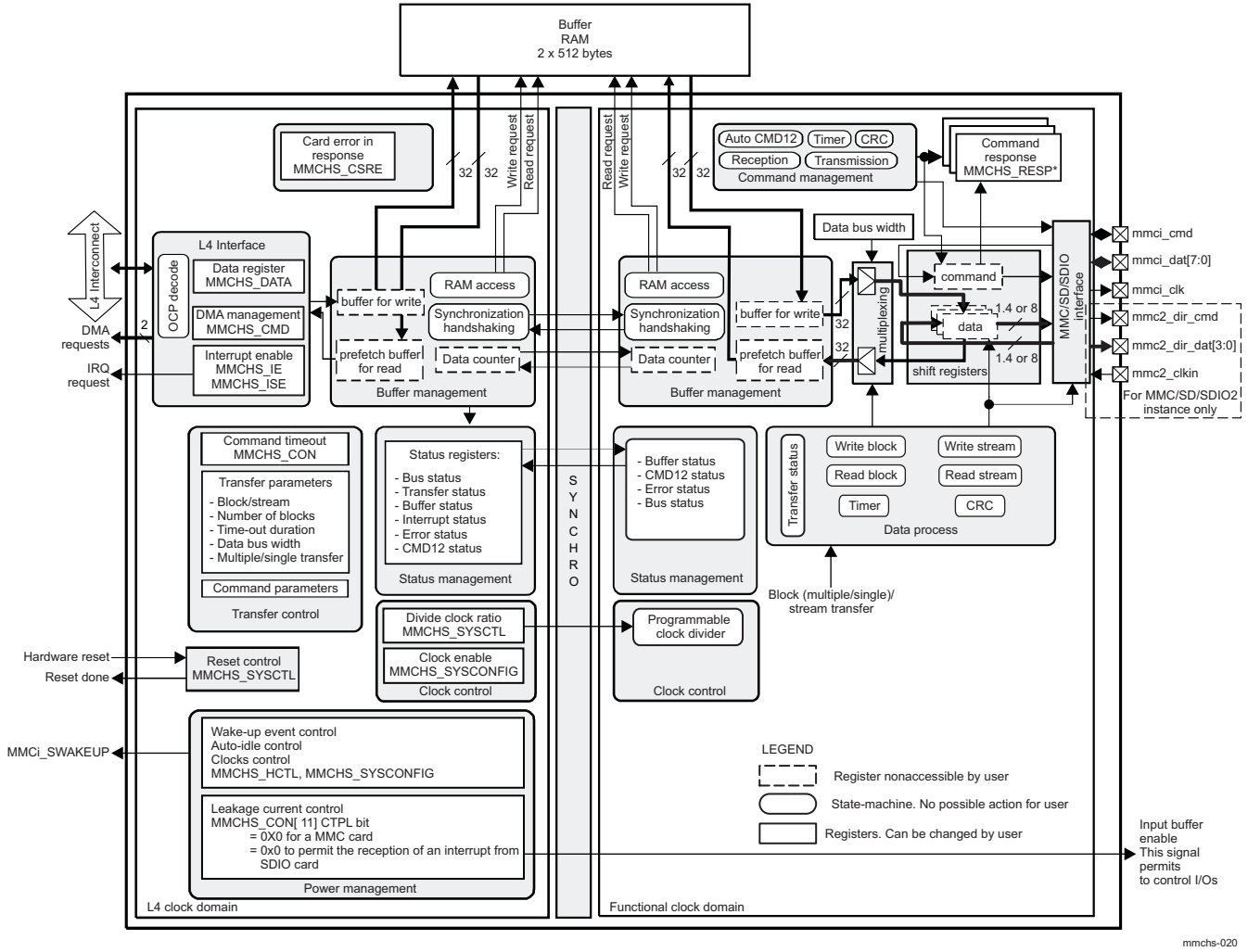
MMC/SD/SDIOi host controller is partitioned into two clock domains:

- The interface L4 clock domain
- The functional clock domain

Any two domains are considered asynchronous to each other, and exchanges between the two domains are synchronized through a synchronization stage and an asynchronous buffer. Data are transferred from one domain to other through the buffer (2*512 RAM).

Figure 22-20 shows a block diagram of the MMC/SD/SDIO host controller.

Figure 22-20. MMC/SD/SDIO Diagram



In the L4 clock domain, the user can:

- Control the transfer (configure parameters for the transfer)
- Control clock activities and reset
- Manage interrupts and hardware requests
- Consult different status:

  - Bus
  - Buffer
  - Interrupts
  - Errors
- Transmit and receive data through the buffer

In the functional clock domain, the internal mechanisms perform the transfers of data and commands.

For more detail see the command response registers (MMCi.MMCHS_RSPn registers: Table 22-52, Table 22-54, Table 22-56, and Table 22-58).

CAUTION

Read access to the command response registers is allowed only when the command process is completed.

22.4.2 Mode Selection

The MMC/SD/SDIO host controller can be use in two modes: MMC and SD/SDIO modes. It has been designed to be the most transparent with the type of card.

The type of the card connected is differentiated by the software initialization procedure. Software identifies the type of card connected during software initialization. For each given card type, there are corresponding commands. Some commands are not supported by all cards. See the *Multimedia Card System Specification*, v4.2, the *SD Memory Card Specifications*, v2.0, and the *SDIO Card Specification, Part E1*, v1.10, for more details.

The purpose of the module is to transfer commands and data, to whatever card is connected, respecting the protocol of the connected card.

Writes and reads to the card must respect the appropriate protocol of that card.

22.4.3 Buffer Management

22.4.3.1 Data Buffer

The MMC/SD/SDIOi host controller uses a data buffer divided into two 512-byte portions that are 32 bits wide by 128 words deep. This buffer transfers data from one data bus (Interconnect) to another data bus (SD SDIO or MMC card bus) and vice versa.

The buffer is the heart of the interface and ensures the transfer between the two interfaces (L4 and the card).

To enhance performance, the data buffer is completed by a prefetch register and a post-write buffer that are not accessible by the host controller.

The read access time of the prefetch register is faster than the one of the data buffer. The prefetch register allows data to be read from the data buffer at an increased speed by preloading data into the prefetch register.

The entry point of the data buffer for the two portions, the prefetch buffer and the post-write buffer, is the 32-bit register MMCi.MMCHS_DATA. A write access to the MMCi.MMCHS_DATA register followed by a read access from the MMCi.MMCHS_DATA register corresponds to a write access to the post-write buffer followed by a read access to the prefetch buffer. As a consequence, it is normal that the data of the write access to the MMCi.MMCHS_DATA register and the data of the read access to the MMCi.MMCHS_DATA register are different.

The number of 32-bit accesses to the MMCi.MMCHS_DATA register that are needed to read (or write) a data block with a size of MMCi.MMCHS_BLK[10:0] BLEN, and equals the rounded up result of BLEN divided by 4.

The maximum block size supported by the host controller is 1024 bytes. This value is hard-coded in the register MMCi.MMCHS_CAPA[17:16] MBL field and cannot be changed.

A read access to the MMCi.MMCHS_DATA register is allowed only when the buffer read enable status is set to 1 (MMCi.MMCHS_PSTATE[11] BRE); otherwise, a bad access (MMCi.MMCHS_STAT[29] BADA) is signaled.

A write access to the MMCi.MMCHS_DATA register is allowed only when the buffer write enable status is set to 1 (MMCi.MMCHS_PSTATE[10] BWE); otherwise, a bad access (MMCi.MMCHS_STAT[29] BADA) is signaled and the data is not written.

The data buffer has two modes of operation to store and read of the first and second portions of the data buffer:

- When the size of the data block to transfer is less than or equal to 512 bytes, meaning the value written in BLEN is less than or equal to 0x200, two data transfers can occur from one data bus to the other data bus and vice versa at the same time. The MMC/SD/SDIOi host controller uses the two portions of the data buffer in a ping pong manner so that storing and reading of the first and second portions of the data buffer are automatically interchanged from time to time so that data may be read from one portion (for instance, through a DMA read access on the interconnect bus) while data (for instance, from the card) is being stored into the other portion and vice versa. When BLEN is less than,

or equal to 0x200 (that is, less than, or equal to 512 bytes), each of the two portions of the buffer that can be used have a size of BLEN (that is, 32 bits x BLEN div by 4). Do not use a size greater than 2 times this value.

CAUTION

The MMCi.MMCHS_CMD[4] DDIR bit must be configured before a transfer to indicate the direction of the transfer.

Figure 22-21 and Figure 22-22 show the buffer management for a write and for a read, respectively.

Figure 22-21. Buffer Management for a Write

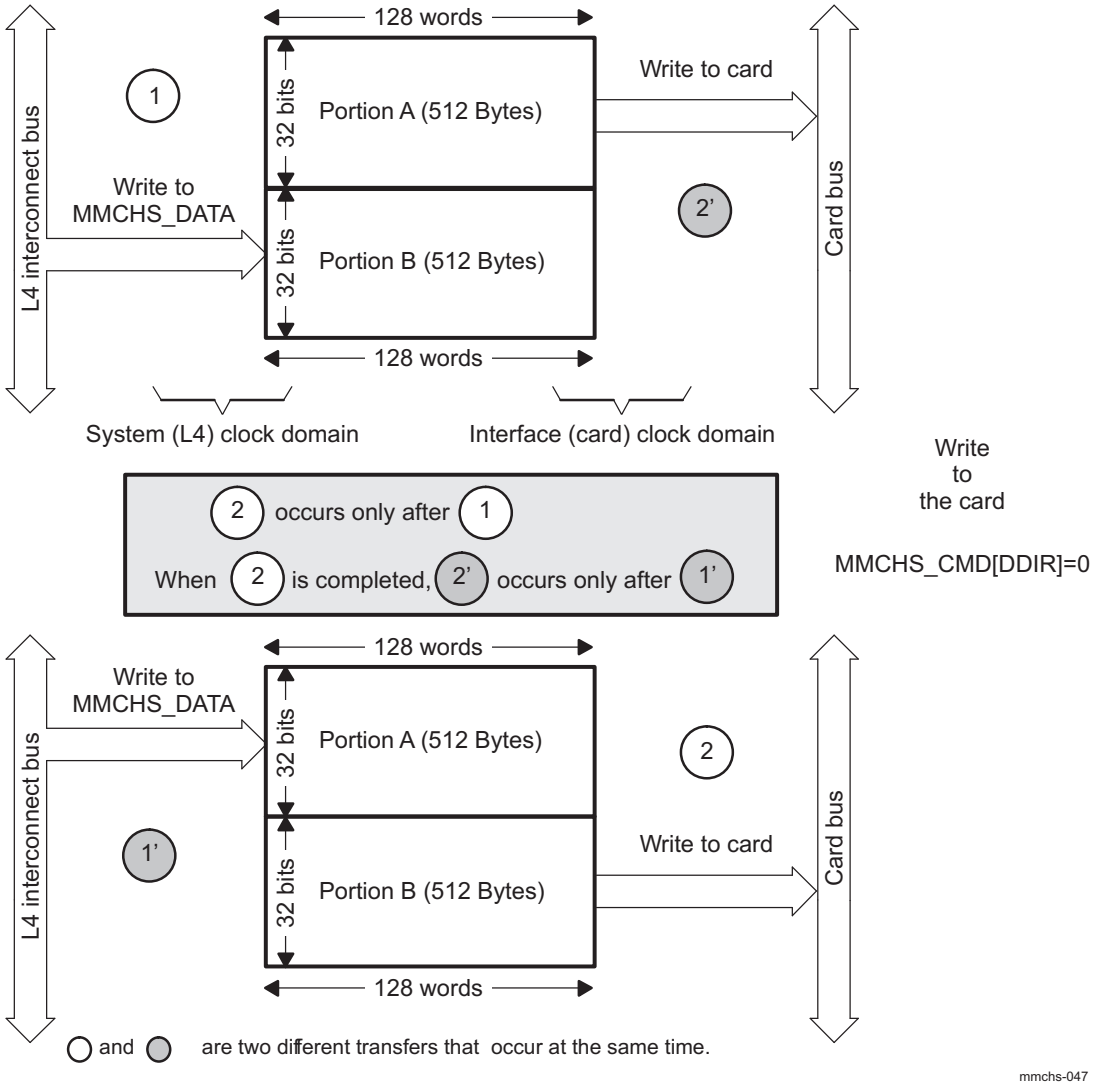
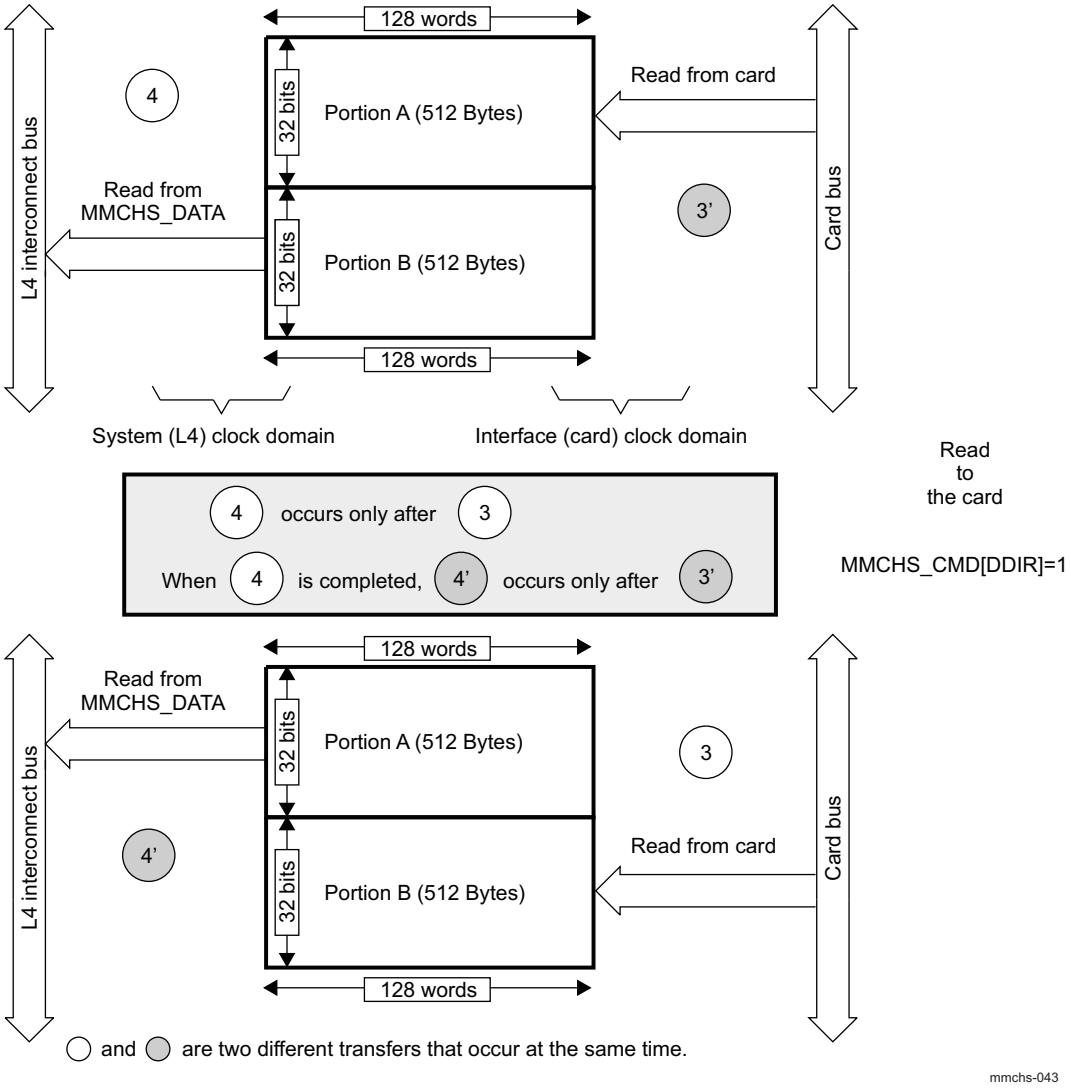


Figure 22-22. Buffer Management for a Read



- When the size of the data block to transfer is larger than 512 bytes, meaning the value written in BLEN is 0x201 or larger, only one data transfer can occur from one data bus to the other data bus at a time. The MMC/SD/SDIOi host controller uses the entire data buffer as a single 1024-byte portion. In this mode, a bad access (MMCi.MMCHS_STAT[29] BADA) is signaled when two data transfers occur from one data bus to the other data bus and vice versa at the same time.

22.4.3.1.1 Data Buffer Status

The data buffer status is defined in the following interrupt status register and status register:

- Interrupt status registers (see Table 22-68):
  - MMCi.MMCHS_STAT[29] BADA: Bad access to data space
  - MMCi.MMCHS_STAT[5] BRR: Buffer read ready
  - MMCi.MMCHS_STAT[4] BWR: Buffer write ready
- Status registers (see Table 22-62):
  - MMCi.MMCHS_PSTATE[11] BRE: Buffer read enable
  - MMCi.MMCHS_PSTATE[10] BWE: Buffer write enable

22.4.4 Transfer Process

The process of a transfer is dependent on the type of command. It can be with or without a response, with or without data.

22.4.4.1 Different Types of Commands

Different types of commands are specific to MMC, SD, or SDIO cards. See the *Multimedia Card System Specification*, v4.2, the *SD Memory Card Specifications*, v2.0, the *SDIO Card Specification, Part E1*, v1.10, or the *SD Card Specification, Part A2, SD Host Controller Standard Specification*, v1.00, for more details.

22.4.4.2 Different Types of Responses

Different types of responses are specific to MMC, SD, or SDIO cards. See the *Multimedia Card System Specification*, v4.2, the *SD Memory Card Specifications*, v2.0, the *SDIO Card Specification, Part E1*, v1.10, or the *SD Card Specification, Part A2, SD Host Controller Standard Specification*, v1.00, for more details.

Table 22-5 shows how the MMC, SD, and SDIO responses are stored in the MMCHS_RSPxx registers.

Table 22-5. MMC, SD, SDIO responses in the MMCHS_RSPxx registers

Kind of Response	Response Field	Response Register
R1, R1b (normal response), R3, R4, R5, R5b, R6	RESP[39:8] (1)	MMCHS_RSP10[31:0]
R1b (Auto CMD12 response)	RESP[39:8] (1)	MMCHS_RSP76[31:0]
R2	RESP[127:0] (1)	MMCHS_RSP76[31:0] MMCHS_RSP54[31:0] MMCHS_RSP32[31:0] MMCHS_RSP10[31:0]

(1) RESP refers to the command response format described in the specifications mentioned above.

When the host controller modifies part of the MMCHS_RSPxx registers, it preserves the unmodified bits.

The host controller stores the Auto CMD12 response in the MMCHS_RSP76[31:0] register because the Host Controller may have a multiple block data DAT line transfer executing concurrently with a command. This allows the host controller to avoid overwriting the Auto CMD12 response with the command response stored in MMCHS_RSP10 register and vice versa.

22.4.5 Transfer or Command Status and Errors Reporting

Flags in the MMC/SD/SDIOi host controller show status of communication with the card:

- A timeout (of a command, a data, or a response)
- A CRC

Error conditions generate interrupts. See Table 22-6 and register description for more details.

Table 22-6. CC and TC Values Upon Error Detected

Error hold in the MMCi.MMCHS_STAT register	CC	TC	Comments
29	BADA		No dependency with CC nor TC BADA is related to MMCHS_DATA register accesses. Its assertion is not dependent of the ongoing transfer.
28	CERR	1	CC is set upon CERR.
22	DEB	1	TC is set upon DEB.

Table 22-6. CC and TC Values Upon Error Detected (continued)

Error hold in the MMCi.MMCHS_STAT register		CC	TC	Comments
21	DCRC		1	TC is set upon DCRC.
20	DTO			DTO and TC are mutually exclusive DCRC and DEB cannot occur with DTO.
19	CIE	1		CC is set upon CIE.
18	CEB	1		CC is set upon CEB.
17	CCRC	1		CC can be set upon CCRC - See CTO comment
16	CTO			CTO and CC are mutually exclusive. CIE, CEB and CERR cannot occur with CTO. CTO can occur at the same time as CCRC: it indicates a command abort due to a contention on CMD line. In this case no CC appears.

22.4.6 Transfer Stop

Whenever a transfer is initiated, the transmission may be willed to stop whereas it is still not finished. Several cases can be faced depending on the transfer type:

- Multiple blocks oriented transfers (for which transfer length is known)
- Continuous stream transfers (which have an infinite length)

Note: Since the MMC/SD/SDIOi controller manages transfers based on a block granularity, the buffer will accept a block only if there is enough space to completely store it. Consequently, if a block is pending in the buffer, no command will be sent to the card because the card clock will be shut off by the controller.

The MMC/SD/SDIOi controller includes two features which makes a transfer stop more convenient and easier to manage:

- Auto CMD12 (for MMC and SD only).

  This feature is enabled by setting the MMCi.MMCHS_CMD[2] ACEN bit to 0x1 (this setting is relevant for a MMC/SD transfer with a known number of blocks to transfer). When the Auto CMD12 feature is enabled, the MMC/SD/SDIOi controller will automatically issue a CMD12 command when the expected number of blocks has been exchanged.
- Stop at block gap

  This feature is enabled by setting the MMCi.MMCHS_HCTL[16] SBGR bit to 0x1. When enabled, this capability holds the transfer on until the end of a block boundary. If a stop transmission is needed, software can use this pause to send a CMD12 to the card.

Note: For MMC and SD cards, the stop at block gap feature is not supported in READ mode.

For SDIO cards, this setting can be supported in READ mode if the card has a read wait capability.

Table 22-7 shows the common way to stop a transfer, indicating command to send and features to enable.

Table 22-7. MMC/SD/SDIOi Controller Transfer Stop Command Summary

		WRITE transfer		READ transfer	
		SD / MMC	SDIO	SD / MMC	SDIO
Single block		Transfer ends automatically Wait TC	Transfer ends automatically Wait TC	Transfer ends automatically Wait TC	Transfer ends automatically Wait TC
Multi blocks (finite or infinite)	Before the programmed block boundary	Send CMD12 Wait TC	Send CMD52 Wait TC	Send CMD12 Wait TC	Send CMD52 Wait TC
	Stop at the end of the transfer (finite transfer only)	Auto CMD12 active Transfer ends automatically Wait TC	Set MMCi.MMCHS_HCTL[16] SBGR bit to 0x1. Send CMD52 Wait TC	Auto CMD12 active Transfer ends automatically Wait TC	**If READ_WAIT supported** Stop at block gap Wait TC
					If READ_WAIT not supported Send CMD52 Wait TC

Note: The MMC/SD/SDIOi controller will send the stop command to the card on a block boundary, regardless the moment the command was written to the controller registers.

22.4.7 MMC CE-ATA Command Completion Disable Management

The MMC/SD/SDIOi host controller supports CE-ATA features, in particular the detection of command completion token. When a command that requires a command completion signal (MMCHS_CON[12] CEATA and MMCHS_CMD[2] ACEN set to 1) is launched, host system is no longer allowed to emit a new command in parallel of data transfer unless it is a command completion disable.

The settings to emit a command completion disable token follow:

- MMCHS_CON[12] CEATA is set to 1.
- MMCHS_CON[2] HR set to 1.
- Clear the MMCHS_ARG register.
- Write into MMCHS_CMD register with value 0x00000000.

When a command completion disable token was emitted (that is, MMCHS_STAT[0] CC received), the host system is again allowed to emit another type of command (for example a transfer abort command CMD12 to abort transfer).

A critical case can be met when command completion signal disable (CCSD) is emitted during the last data block transfer, the sequence on command line could be sent very close to command completion signal (CCS) token sent by the card.

Three cases can be met:

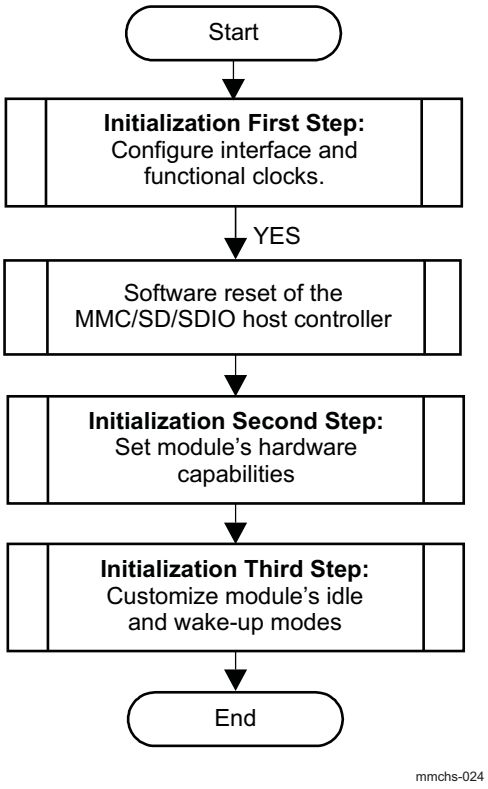
- CCS is receive just before CCSD is emitted:
  An interrupt CIRQ is generated with CCS detection, CCSD is transmitted to card then an interrupt CC is generated when CCSD ends. In this case, card consider the CCSD sequence.
- CCS is not generated or generated during the CCSD transfer:
  The CCS bit cannot be detected (conflict is not possible as they drive the same level on command line, then no CIRQ interrupt is generated; besides CC interrupt is generated when CCSD ends).
- CCS is generated without CCSD token required:
  Only the interrupt CIRQ is generated when CCS is detected.

22.5 MMC/SD/SDIO Basic Programming Model

22.5.1 MMC/SD/SDIO Host Controller Initialization Flow

Figure 22-23 shows the general boot process.

Figure 22-23. MMC/SD/SDIO Controller Meta Initialization Steps



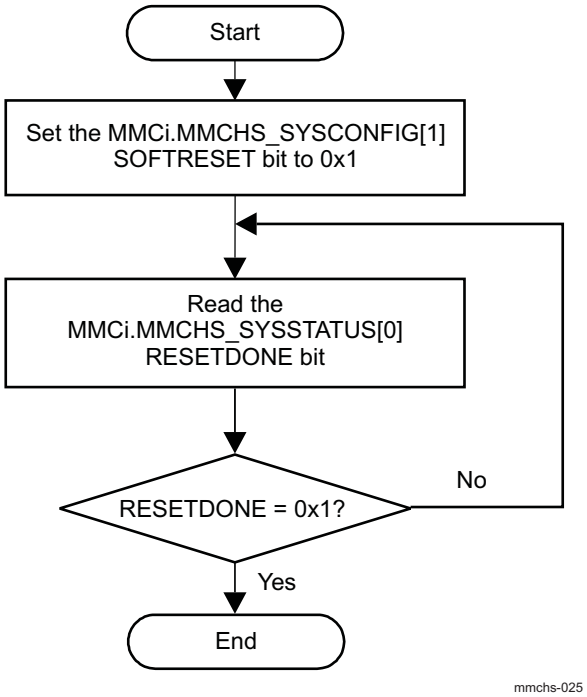
22.5.1.1 Enable Interface and Functional clock for MMC Controller

Prior to any MMCHS register access one must enable MMCHS interface clock and functional clock in PRCM module registers PRCM.CM_ICLKEN1_CORE and PRCM.CM_FCLKEN1_CORE. Please refer to Chapter 4, *Power, Reset, and Clock Management*.

22.5.1.2 MMCHS Soft Reset Flow

Figure 22-24 shows the soft reset process of MMCHS controller.

Figure 22-24. MMC/SD/SDIO Controller Software Reset Flow



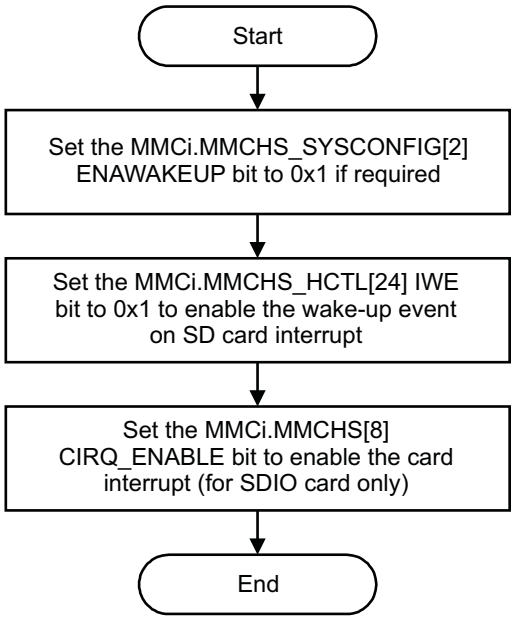
22.5.1.3 Set MMCHS Default Capabilities

Software must read capabilities (in boot ROM for instance) and is allowed to set (write) MMCi.MMCHS_CAPA[26:24] and MMCi.MMCHS_CUR_CAPA[23:0] registers before the MMC/SD/SDIO host driver is started.

22.5.1.4 Wake-Up Configuration

Figure 22-25 details MMCHS controller wake-up configuration.

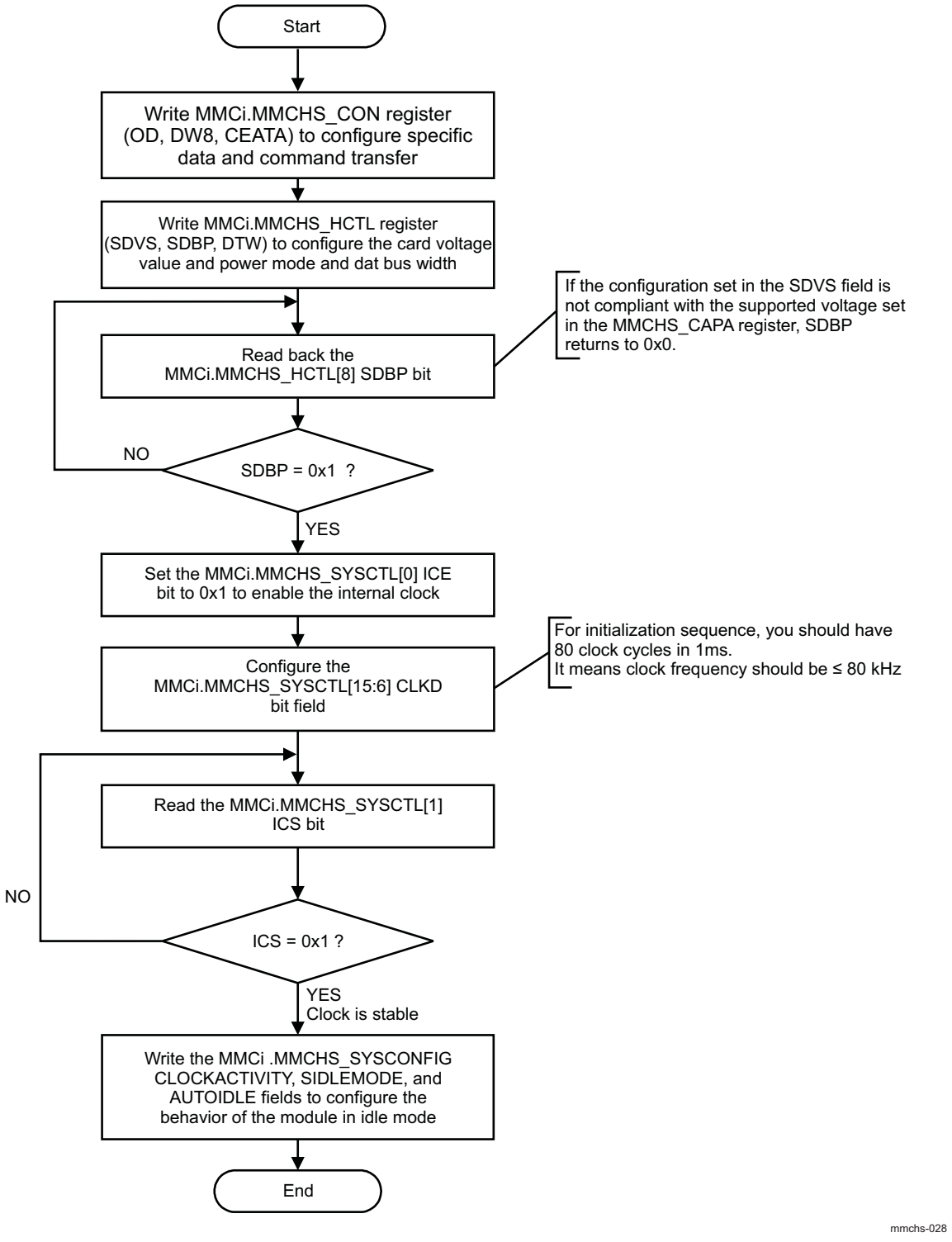
Figure 22-25. MMC/SD/SDIO Controller Wake-Up Configuration



22.5.1.5 MMC Host and Bus Configuration

Figure 22-26 details MMC bus configuration process.

Figure 22-26. MMC/SD/SDIO Controller Bus Configuration



22.5.2 Basic Operations for MMC/SD/SDIO Host Controller

The MMC/SD/SDIO host controller performs data transfers: data to card (referred to as write transfers) and data from card (referred to as read transfers).

The host controller requires transfers to run on a block-by-block basis, rather than on a DMA burst size basis. A single DMA request (or block request interrupt) is signaled for each block. Pipelining is supported as long as the block size is less than one half of the memory buffer size.

22.5.2.1 Card Detection, Identification, and Selection

Figure 22-27 and Figure 22-28 show the card identification and selection process.

Figure 22-27. MMC/SD/SDIO Controller Card Identification and Selection - Part 1

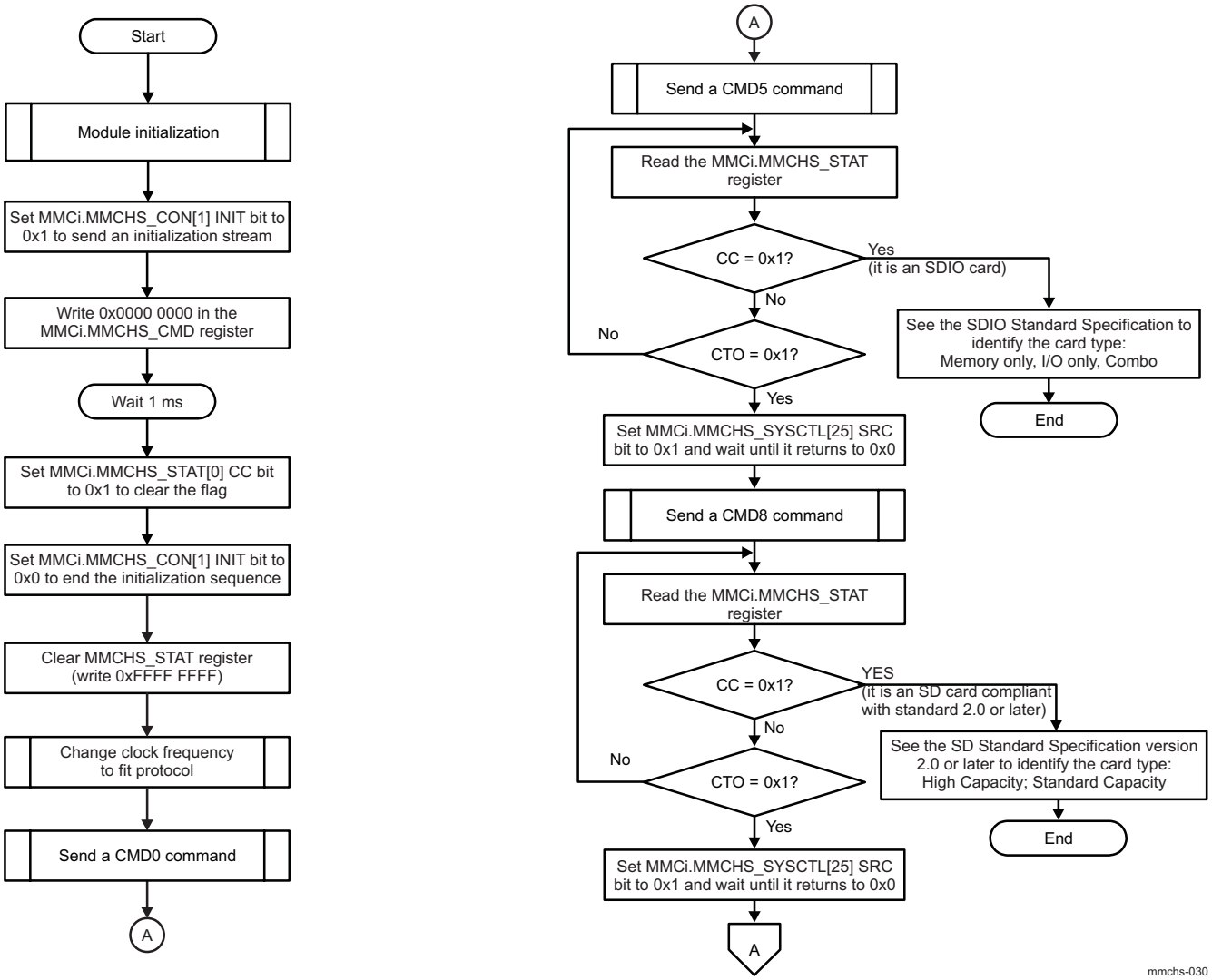
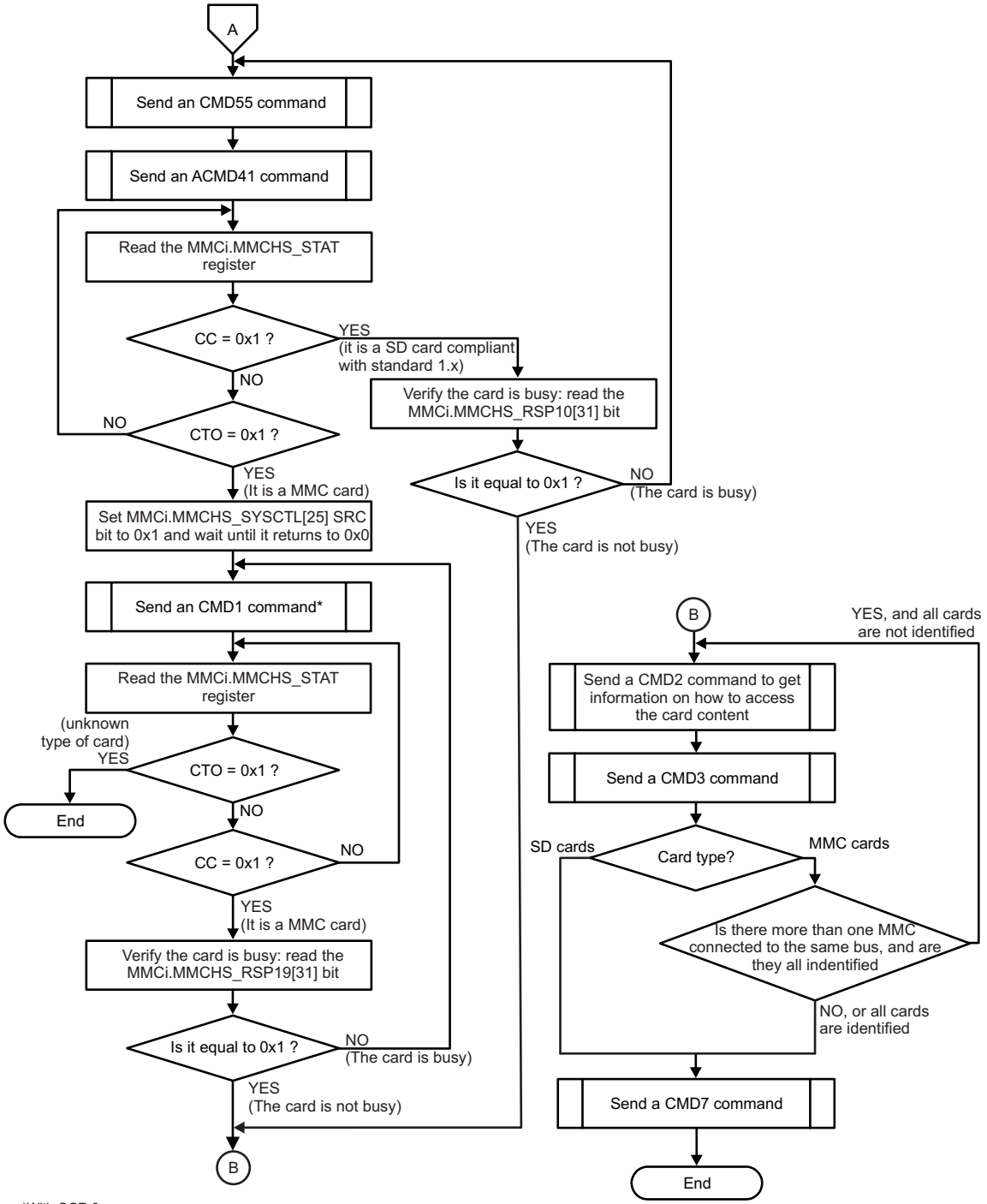


Figure 22-28. MMC/SD/SDIO Controller Card Identification and Selection - Part 2



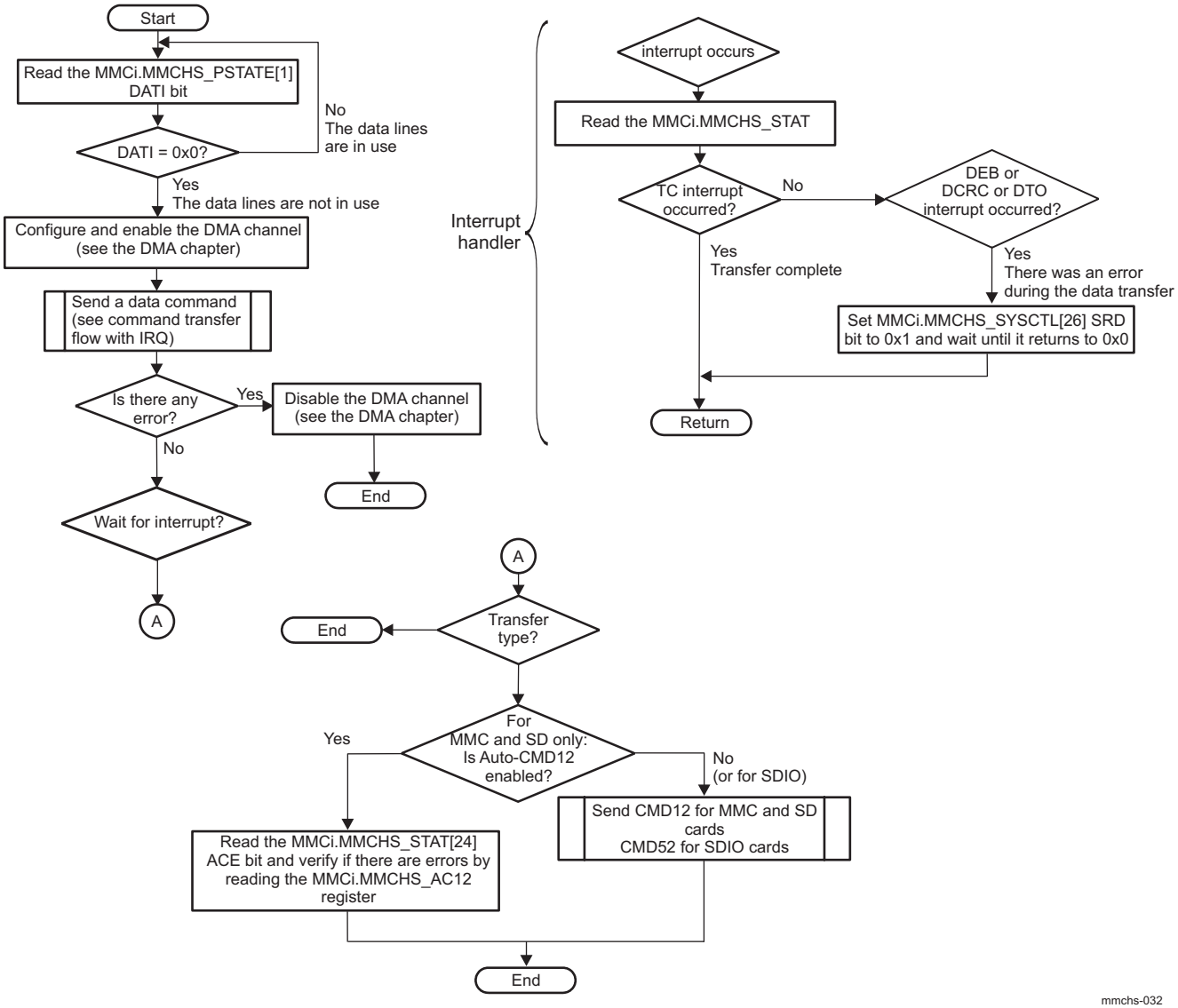
*With OCR 0.
In case of a CMD1 with OCR=0, a second CMD1 must be sent to the card with the "negociated" voltage.

108-031

22.5.2.2 Read/Write Transfer Flow in DMA Mode with Interrupt

Figure 22-29 describes the read and write protocol in DMA mode with interrupt signaling. Refer to Chapter 9, *DMA*, for more information on the DMA settings.

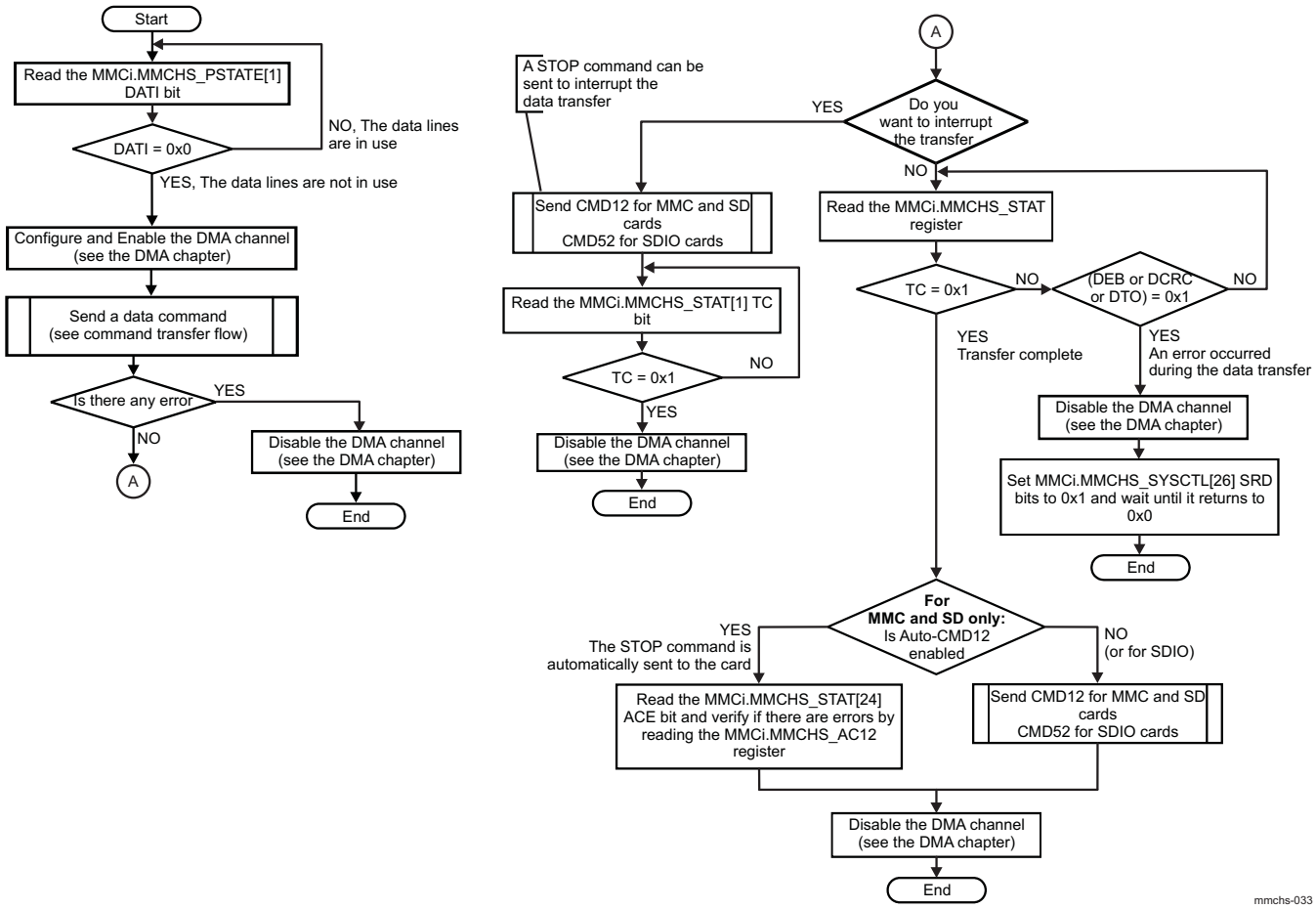
Figure 22-29. MMC/SD/SDIO Controller Read/Write Transfer Flow in DMA Mode with Interrupt



22.5.2.3 Read/Write Transfer Flow in DMA Mode with Polling

Figure 22-30 describes the read and write protocol in DMA mode. Refer to Chapter 9, *DMA*, for more information on the DMA settings.

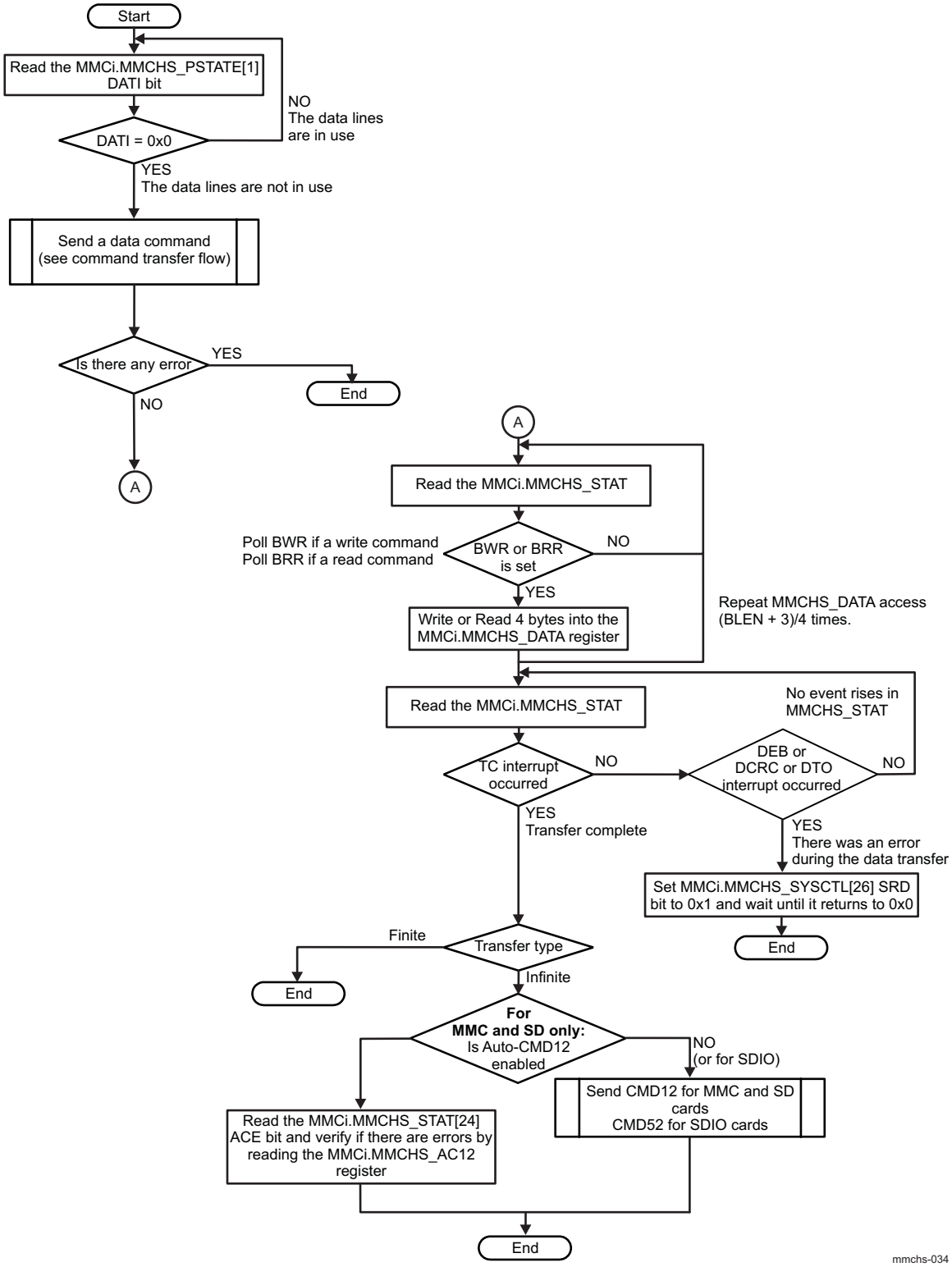
Figure 22-30. MMC/SD/SDIO Controller Read/Write Transfer Flow in DMA Mode with Polling



22.5.2.4 Read/Write Transfer Flow without DMA with Polling

Figure 22-31 describes a read/write transfer without using the DMA and with polling.

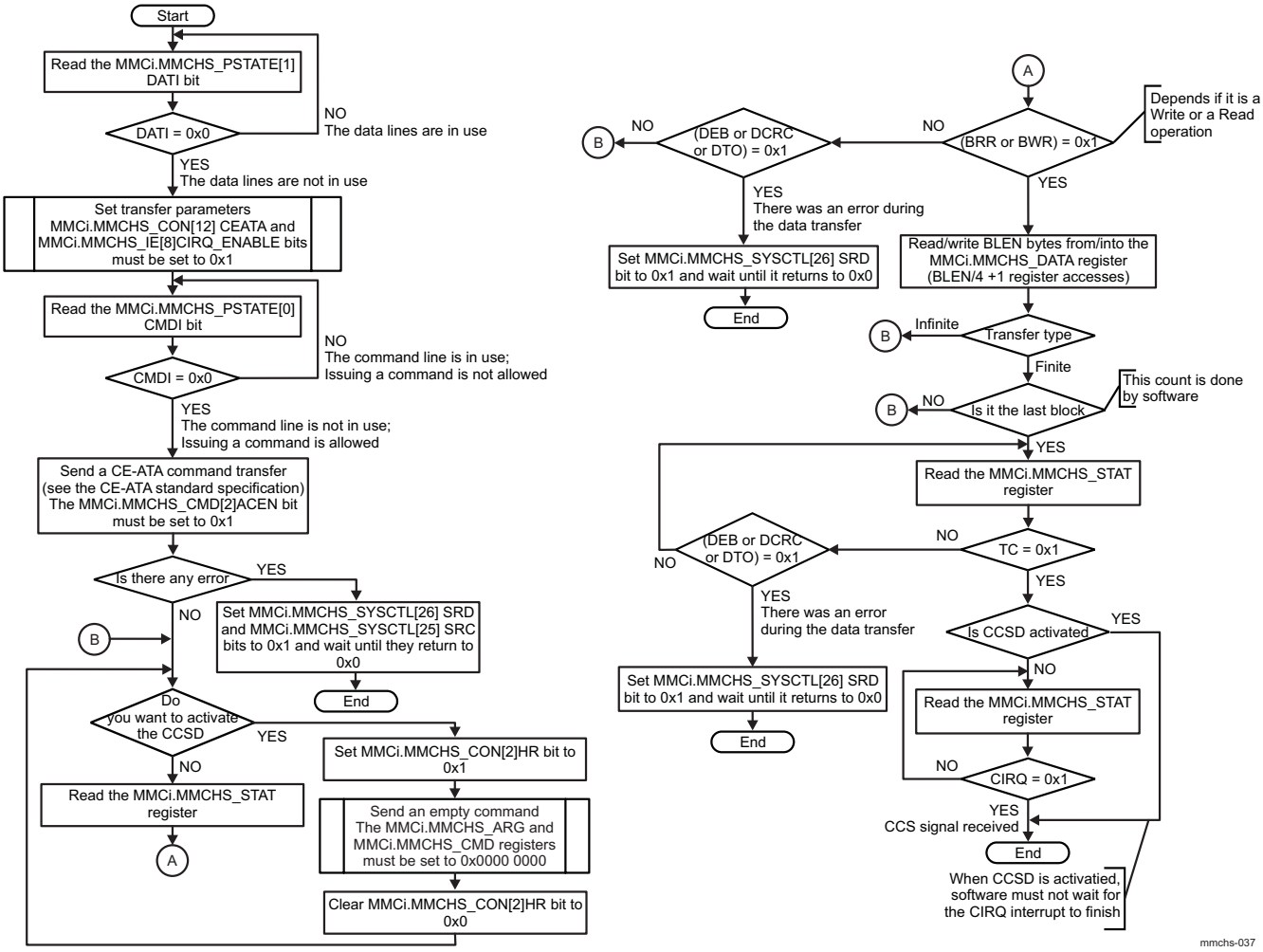
Figure 22-31. MMC/SD/SDIO Controller Read/Write Transfer Flow without DMA with Polling



22.5.2.5 Read/Write Transfer Flow in CE-ATA Mode

Figure 22-32 describes the read and write CE-ATA protocol when in Polling mode.

Figure 22-32. MMC/SD/SDIO Controller Read/Write in CE-ATA Mode



CAUTION

CE-ATA protocol is only supported by MMC cards.

In CE-ATA mode, issuing command during the transfer (except a CCSD command) is not allowed

In CE-ATA mode, infinite transfers are not allowed, only finite transfers are permitted.

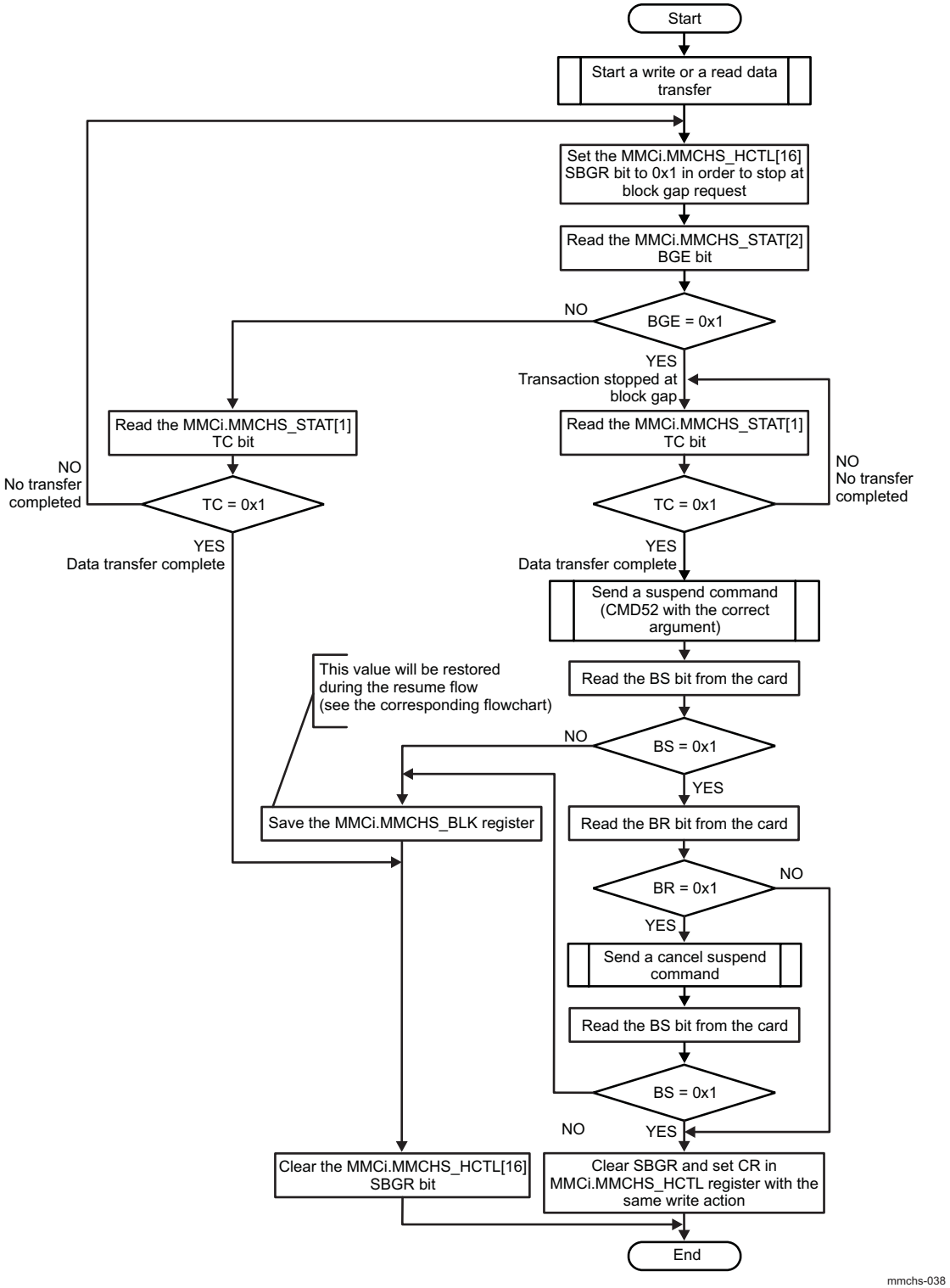
22.5.2.6 Suspend-Resume Flow

The suspend and resume feature is only supported by SDIO cards.

22.5.2.6.1 Suspend Flow

Figure 22-33 describes the suspend flow for SDIO cards.

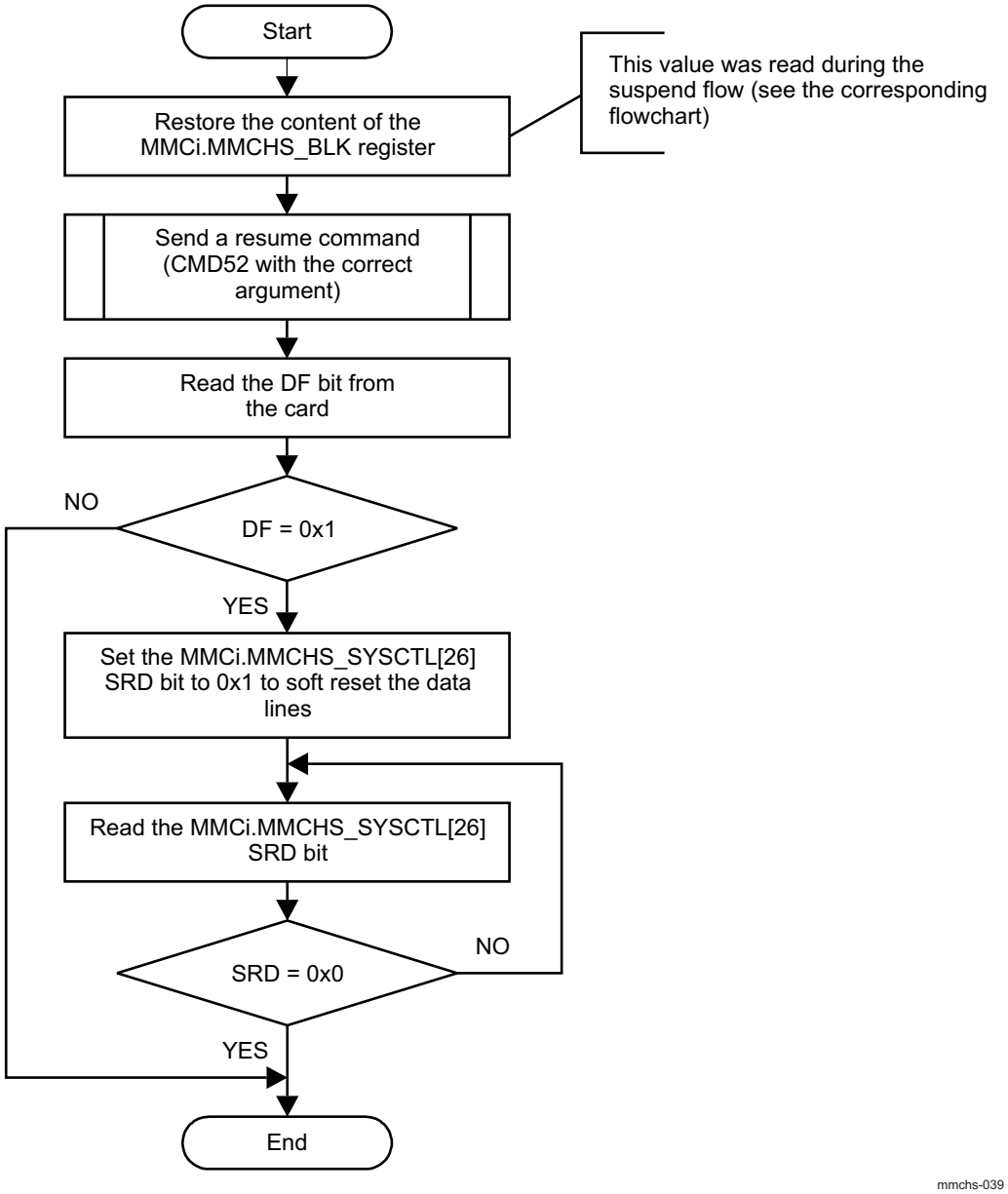
Figure 22-33. MMC/SD/SDIO Controller Suspend Flow



22.5.2.6.2 Resume Flow

Figure 22-34 describes the resume flow for SDIO cards.

Figure 22-34. MMC/SD/SDIO Controller Resume Flow



22.5.2.7 Basic Operations - Steps Detailed

22.5.2.7.1 Command Transfer Flow

Figure 22-35 describes how to send a command to the card using polling instead of interrupts for event signaling.

Figure 22-35. MMC/SD/SDIO Controller Command Transfer Flow with Polling

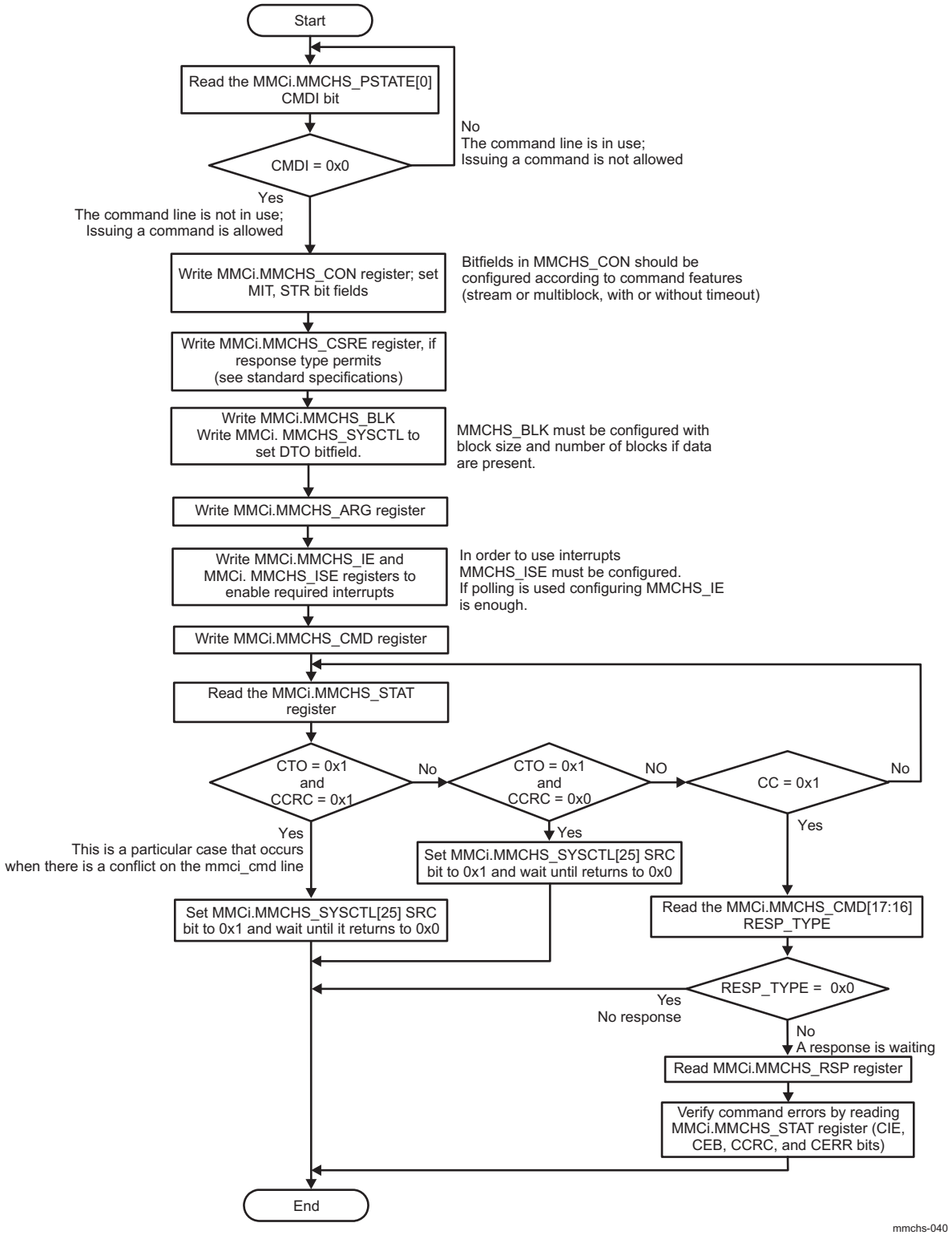
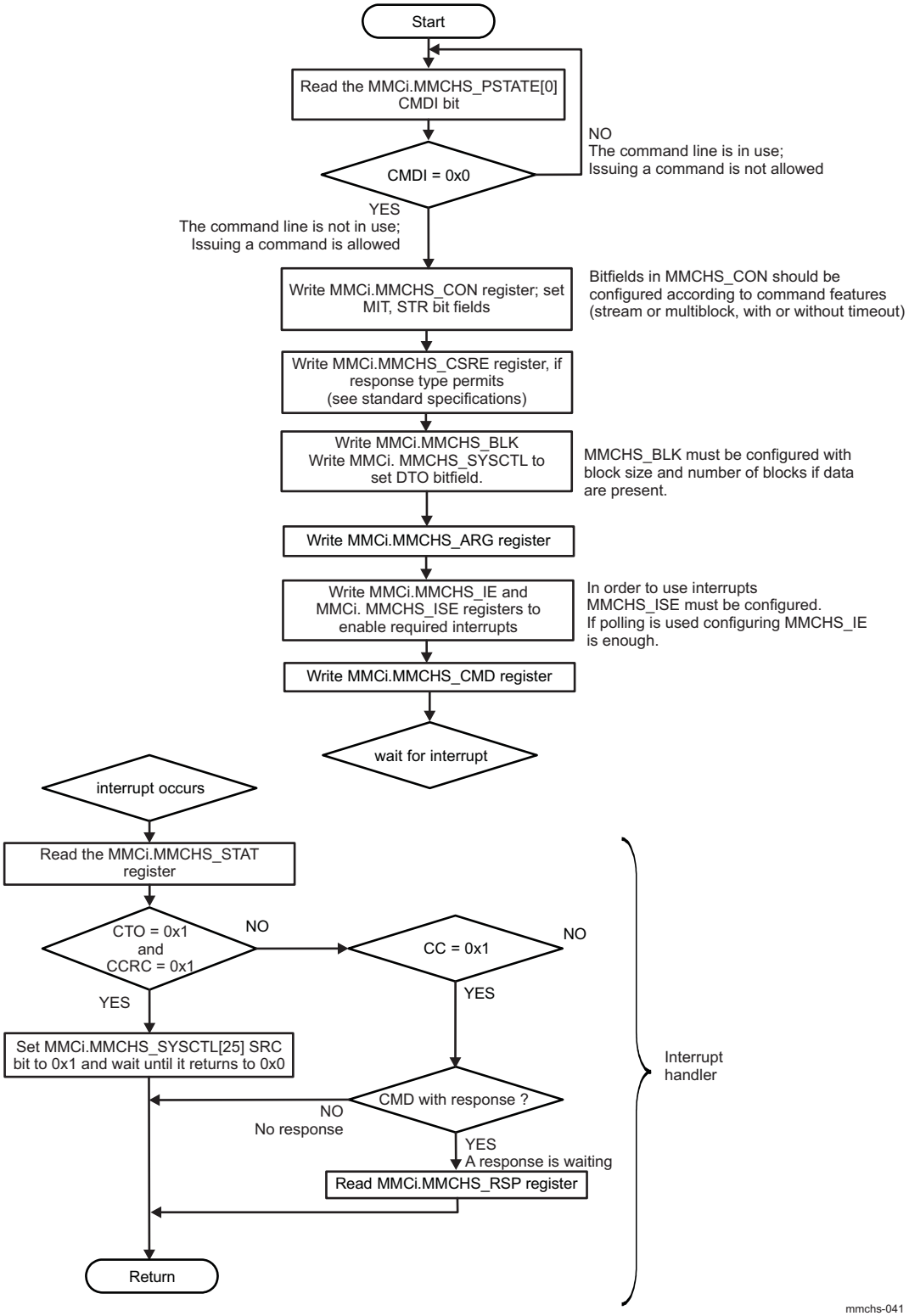


Figure 22-36 describes how to send a command to the card using interrupts for event signaling.

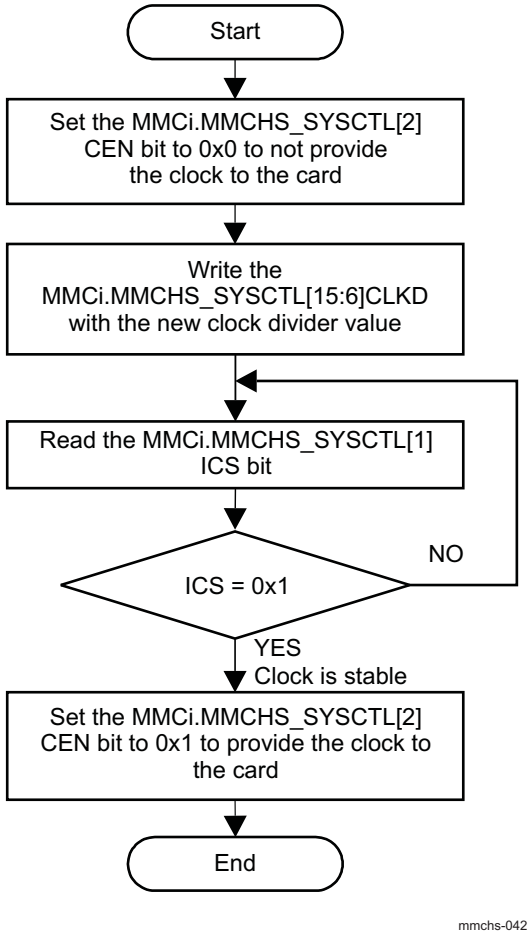
Figure 22-36. MMC/SD/SDIO Controller Command Transfer Flow with Interrupts



22.5.2.7.2 MMCHS Clock Frequency Change

Figure 22-37 describes the different steps that allow to change the MMC/SD/SDIO output clock frequency.

Figure 22-37. MMC/SD/SDIO Controller Clock Frequency Change Flow



22.5.3 MMC/SD/SDIO1 Bus Voltage Selection

The MMC/SD/SDIO1 controller can operate with two types of card voltages: 1.8 V and 3.0 V. For this reason, dual voltage pads are implemented on this interface. For technological concerns those pads must have an internal bias voltage reference to operate. The PBIAS_LITE module supplies this bias voltage, depending on the CONTROL.CONTROL_PBIAS_LITE register settings.

See Section 7.4.5, *Extended-Drain I/O Pin and PBIAS Cell*, for more information about the PBIAS_LITE cell.

Section 7.5.2, *Extended-Drain I/Os and PBIAS Cell Basic Programming Guide*, describes the steps involved in transitioning from 1.8 V to 3.0 V and from 3.0 V to 1.8 V, applicable to the MMC/SD/SDIO1 controller.

CAUTION

The BIAS voltage must be set using the procedure described in Section 7.5.2, *Extended-Drain I/Os and PBIAS Cell Basic Programming Guide*. Failure to follow this procedure can damage the MMCHS interface.

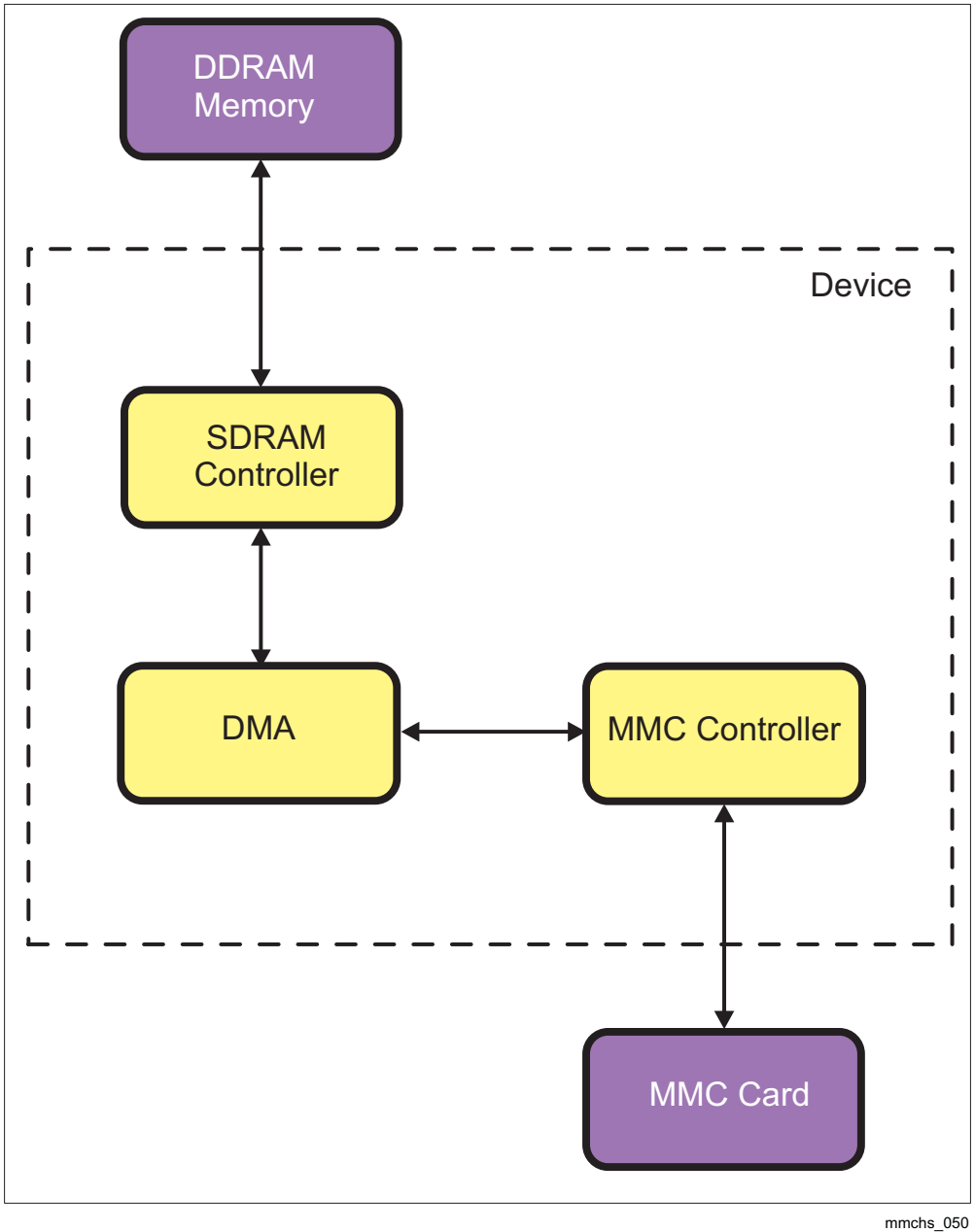
22.6 MMC/SD/SDIO Use Cases and Tips

22.6.1 MMCHS Controller Usage

22.6.1.1 Overview

The MMCHS controller is in charge of managing raw data storage in a MMC memory card. The MMCHS controller transfers data between DDRAM and MMC card. Figure 22-38 gives an overview of MMCHS controller position in the use case.

Figure 22-38. Overview



For the Camcorder use case, the MMCHS controller is configured to operate with the following features :

- High speed mode with a card clock frequency of 48 MHz.
- 8 data lines.

- 1.8 V and 3.0 V voltage capabilities.

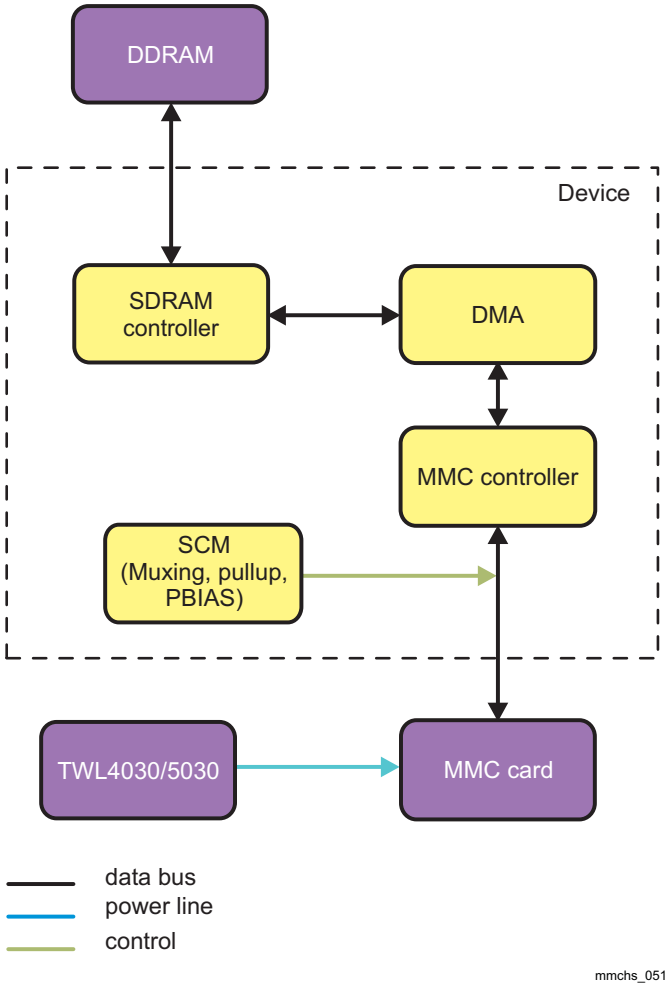
MMC card power supply is not provided by the MMCHS controller itself but rather by the companion device. please refer to I2C TRM output to learn how to set these voltage levels.

Only MMC1 controller is used in this configuration. Hence this document describes the output of the test carried out with MMC1 controller.

22.6.1.2 Environment

In order to operate in the correct manner, the MMCHS controller needs the System Control Module (SCM) to be configured with the right muxing mode and with the right pull up state. The MMC card needs to be powered with the right power level and this is done with the companion device. Figure 22-39 gives the environment picture of the MMCHS controller.

Figure 22-39. Environment



22.6.1.2.1 Command and Data Transfer Formats

When communicating with a MMC card, The MMCHS controller is always the master. The communication between the MMCHS controller and the MMC card always starts by sending a command. Both command and data transfers are started with a command.

The data transfer type used by the MMCHS controller is a finite multiple block transfer. Figure 22-40 illustrates a command transfer without data. Figure 22-41 and Figure 22-42 illustrate a multiple block transfer between the MMCHS controller and the MMC card.

Figure 22-40. Command Transfer

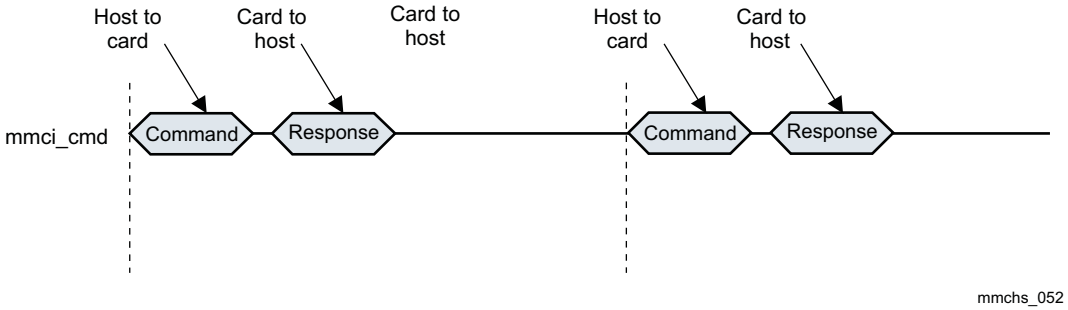


Figure 22-41. Data Read Transfer

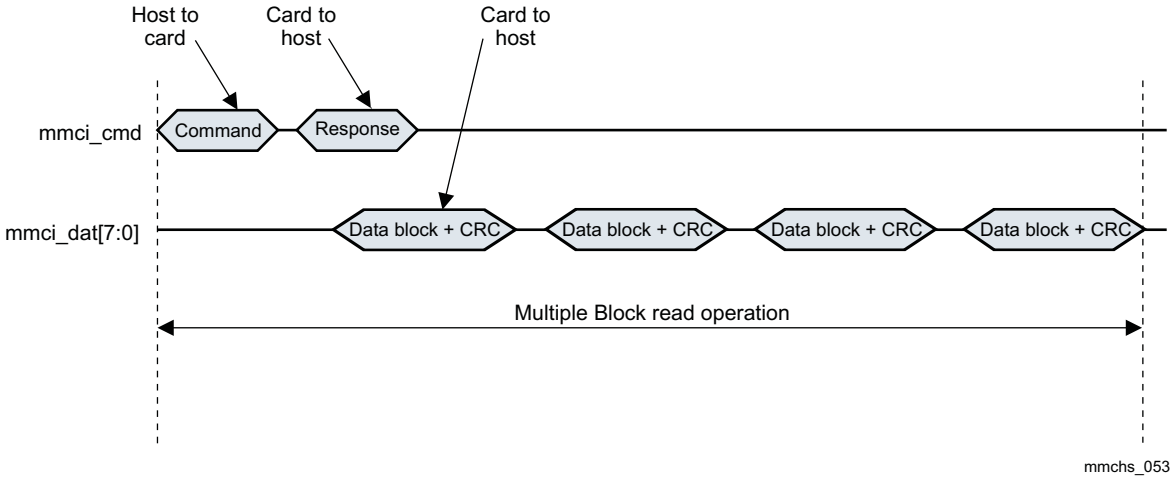
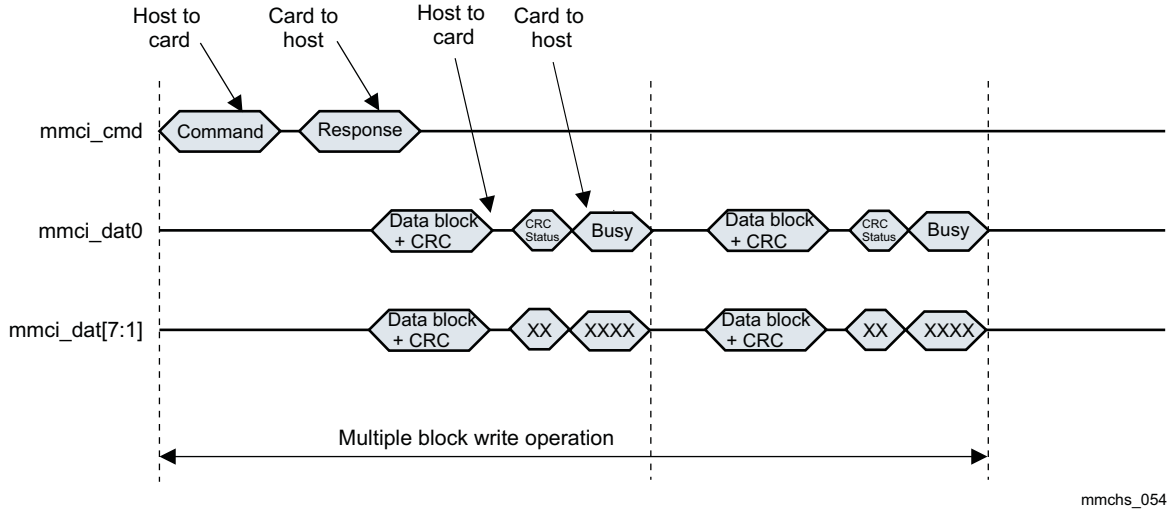


Figure 22-42. Data Write Transfer



22.6.1.3 Programming Flow

22.6.1.3.1 Initial Configuration

The initialization of the MMCHS controller is done through these steps :

1. MMCHS controller interface and functional clocks enabling.
2. MMCHS controller software reset.
3. MMCHS controller voltage capabilities initialization.
4. MMCHS controller default initialization.
5. MMCHS controller INIT procedure start.
6. MMCHS controller pre-card identification configuration.

   For more information about different steps the MMCHS controller goes through during initial configuration refer to Section 22.5, *MMC/SD/SDIO Basic Programming Model*.

22.6.1.3.1.1 MMCHS Controller Interface and Functional Clocks Enabling

To enable the interface and functional clocks of the MMCHS1 controller, the following steps must be done:

1. Enable the interface clock for the MMCHS1 controller (set the PRCM.CM_ICLKEN1_CORE[24] EN_MMCHS1).
2. Enable the functional clock for the MMCHS1 module (set the PRCM.CM_FCLKEN1_CORE[24] EN_MMCHS1).

Table 22-8 shows all PRCM registers to be configured to enable interface and functional clocks for MMCHS1 controller.

Table 22-8. Register Print for the MMCHS1 controller's clocks Initialization

Register Name	Register Address	Value	Value Description
PRCM.CM_ICLKEN1_CORE	0x4800 4A10	0x01000000	MMCHS1 interface clock enabled
PRCM.CM_FCLKEN1_CORE	0x4800 4A00	0x01000000	MMCHS1 functional clock enabled

22.6.1.3.1.2 MMCHS Controller Software Reset

In order to software reset the MMCHS1 controller, the following steps must be done:

1. Write 0x2 in MMCHS1.MMCHS_SYSCONFIG register.
2. Wait until MMCHS1.MMCHS_SYSSTATUS[0] RESETDONE turns 1.

Register Name	Register Address	Value	Value Description
MMCHS1.MMCHS_SYSCONFIG	0x4809 C010	0x00000002	Activate software reset
MMCHS1.MMCHS_SYSSTATUS	0x4809 C014	0x00000001	Reset is over.

22.6.1.3.1.3 MMCHS Controller Voltage Capabilities Initialization

MMCHS1 controller's voltage capabilities should be set in MMCHS1.MMCHS_CAPA. Refer to Table 22-9.

Table 22-9. MMCHS Controller Voltage Capabilities Initialization

Register Name	Register Address	Value	Value Description
MMCHS1.MMCHS_CAPA	0x4809 C140	current_value | 0x06000000	Activate VS18 and VS30 in MMCHS_CAPA register.

22.6.1.3.1.4 MMCHS Controller Default Initialization

Before sending any command, the MMCHS controller is configured with default values :

1. Default voltage support is set to 1.8 v in MMCHS1.MMCHS_HCTL[11:9] SDVS.
2. MMC bus is set to open drain in MMCHS1.MMCHS_CON[0] OD.

3. MMC data bus width is set to 1 in MMCHS1.MMCHS_HCTL[1] DTW.
4. MMC Card's power is off.
5. MMC Card's clock is on in MMCHS1.MMCHS_SYSCTL[0] ICE and MMCHS1.MMCHS_SYSCTL[2] CEN.
6. MMCHS controller bus power up in MMCHS1.MMCHS_HCTL[8] SDBP.
7. MMC card's clock frequency is set to 150 KHz in MMCHS1.MMCHS_SYSCTL[15:6] CLKD.

Table 22-10 shows the values that should be written in the right register pool.

Table 22-10. MMC Controller Default Initialization Values

Register Name	Register Address	Value	Value Description
MMCHS1.MMCHS_HCTL	0x4809C128	0x00000b00	data bus width = 1, voltage = 1.8v, MMC bus power is on (not card's power)
MMCHS1.MMCHS_SYSCTL	0x4809C12C	0x0000a007	card's clock enable and card's clock frequency divider.
MMCHS1.MMCHS_CON	0x4809C02C	0x00000001	Set MMC bus mode to open drain.

A small notice about MMCHS1.MMCHS_HCTL. Even if the value written in it is 0x00000b00, the value read from it is equal to 0x00000a00. This is due to the fact that the MMCHS controller in not in a card state mode which sets automatically MMCHS_HCTL[8] SDBP bit to 0.

22.6.1.3.1.5 MMCHS Controller INIT Procedure Start

Prior to issuing any command, the MMCHS controller has to execute a special INIT procedure. The MMCHS controller has to generate a clock during 1ms. During the INIT procedure, the MMCHS controller generates 80 clock periods. In order to keep the 1ms gap, the MMCHS controller should be configured to generate a clock whose frequency is smaller or equal to 80 KHz.

The INIt procedure is executed by setting MMCHS1.MMCHS_CON[1] INIT bit field to 1 and by sending a dummy command, writing 0x00000000 in MMCHS1.MMCHS_CMD register.

Table 22-11 shows the values that should be written in the right register pool.

Table 22-11. MMCHS Controller INIT Procedure Start

Register Name	Register Address	Value	Value Description
MMCHS1.MMCHS_CON	0x4809C02C	current_value | 0x00000002	sets MMCHS1.MMCHS_CON[1] INIT to 1
MMCHS1.MMCHS_CMD	0x4809C10C	0x00000000	sends dummy command.

22.6.1.3.1.6 MMCHS Controller Pre-card Identification Configuration

Before card identification starts, the MMCHS controller's configuration should change. MMC card's clock should now be 400 KHz according to MMC system spec requirements. Table 22-12 shows the values that should be written in the right register pool.

Table 22-12. MMCHS Controller Pre-Card Identification Configuration

Register Name	Register Address	Value	Value Description
MMCHS1.MMCHS_HCTL	0x4809C128	0x00000b00	data bus width = 1, voltage = 1.8v, MMC bus power is on (not card's power)
MMCHS1.MMCHS_SYSCTL	0x4809C12C	0x00003C07	card's clock enable and card's clock frequency divider.
MMCHS1.MMCHS_CON	0x4809C02C	0x00000001	Set MMC bus mode to open drain.

22.6.1.3.2 MMC Card Identification

MMC card identification is performed by issuing many MMC commands. Each command imposes certain configuration values in a pool of registers. The status of command transfer is read in MMCHS1.MMCHS_STAT register and command response if any is read from MMCHS1.MMCHS_RSP10, MMCHS1.MMCHS_RSP32, MMCHS1.MMCHS_RSP54 and MMCHS1.MMCHS_RSP76.

This TRM output describes a use case where interrupts are used to signal MMCHS controller status changes.

For more details about the card identification sequence, please refer to Section 22.5 *MMC/SD/SDIO Basic Programming Model*.

22.6.1.3.2.1 Sending CMD0

This command resets the MMC card (see Table 22-13).

Table 22-13. Sending CMD0

Register Name	Register Address	Value	Value Description
MMCHS1.MMCHS_CON	0x4809C02C	0x00000001	MMC bus is still in open drain state for broadcast.
MMCHS1.MMCHS_IE	0x4809C134	0x00040001	Enables CC and CEB events to occur.
MMCHS1.MMCHS_ISE	0x4809C138	0x00040001	Enables CC and CEB interrupts to rise.
MMCHS1.MMCHS_CMD	0x4809C10C	0x00000000	Sends CMD0 whose opcode is 0.

22.6.1.3.2.2 Sending CMD5

This command asks a SDIO card to send its operating conditions (see Table 22-14). This command will fail if there is no SDIO card. In case of success the response will be in MMCHS1.MMCHS_RSP10 register.

Table 22-14. Sending CMD5

Register Name	Register Address	Value	Value Description
MMCHS1.MMCHS_CON	0x4809C02C	0x00000001	MMC bus is still in open drain state for broadcast.
MMCHS1.MMCHS_IE	0x4809C134	0x00050001	Enables CC, CTO and CEB events to occur.
MMCHS1.MMCHS_ISE	0x4809C138	0x00050001	Enables CC, CTO and CEB interrupts to rise.
MMCHS1.MMCHS_CMD	0x4809C10C	0x05020000	Sends CMD5 whose opcode is 5 and response type is 48 bits.

22.6.1.3.2.2.1 Sending CMD8

This command asks a SD card version 2.X to send its operating conditions (see Table 22-15). This command will fail if there is no SD card verison 2.X. In case of success the response will be in MMCHS1.MMCHS_RSP10 register.

Table 22-15. Sending CMD8

Register Name	Register Address	Value	Value Description
MMCHS1.MMCHS_CON	0x4809C02C	0x00000001	MMC bus is still in open drain state for broadcast.

Table 22-15. Sending CMD8 (continued)

Register Name	Register Address	Value	Value Description
MMCHS1.MMCHS_IE	0x4809C134	0x100f0001	Enables CERR, CIE, CCRC, CC, CTO and CEB events to occur.
MMCHS1.MMCHS_ISE	0x4809C138	0x100f0001	Enables CERR, CIE, CCRC, CC, CTO and CEB interrupts to rise.
MMCHS1.MMCHS_CMD	0x4809C10C	0x81a0000	Sends CMD8 whose opcode is 8, response type is 48 bits with CICE and CCCE enabled.

22.6.1.3.2.3 Sending CMD55

This is a special command used to prevent the card that the following command is going to be an application one (see Table 22-16). This is used to prepare the issuing of ACMD41 (opcode = 41) that usually asks a SD card version 1.X to send its operating conditions. If no SD card version 1.X is connected to the MMCHS controller this command will fail. In case of success, the response will be received in MMCHS1.MMCHS_RSP10 register.

Table 22-16. Sending CMD55

Register Name	Register Address	Value	Value Description
MMCHS1.MMCHS_CON	0x4809C02C	0x00000001	MMC bus is still in open drain state for broadcast.
MMCHS1.MMCHS_IE	0x4809C134	0x100f0001	Enables CERR, CIE, CCRC, CC, CTO and CEB events to occur.
MMCHS1.MMCHS_ISE	0x4809C138	0x100f0001	Enables CERR, CIE, CCRC, CC, CTO and CEB interrupts to rise.
MMCHS1.MMCHS_CMD	0x4809C10C	0x371a0000	Sends CMD55 whose opcode is 55, response type is 48 bits with CICE and CCCE enabled.

22.6.1.3.2.4 Sending CMD1

Once the card response is available in register MMCHS1.MMCHS_RSP10, the software is responsible to compare Card OCR and Host OCR, and then send a second CMD1 command with the cross-checked OCR.

This way, the card is notified of the Operating Voltage to work with.

Table 22-17. Sending CMD1

Register Name	Register Address	Value	Value Description
MMCHS1.MMCHS_CON	0x4809C02C	0x00000001	MMC bus is still in open drain state for broadcast.
MMCHS1.MMCHS_IE	0x4809C134	0x00050001	Enables CC, CTO and CEB events to occur.
MMCHS1.MMCHS_ISE	0x4809C138	0x00050001	Enables CC, CTO and CEB interrupts to rise.
MMCHS1.MMCHS_CMD	0x4809C10C	0x01020000	Sends CMD1 whose opcode is 1 and response type is 48 bits.

Once the card response is available in register MMCHS1.MMCHS_RSP10, the software should compare card OCR and host OCR, and then send a second CMD1 command with the cross-checked OCR. This tells the card what operating voltage to use.

22.6.1.3.2.5 Sending CMD2

This command asks the MMC card to send its CID register's content (see Table 22-18). The response is 128 bit wide and is received in MMCHS1.MMCHS_RSP10, MMCHS1.MMCHS_RSP32, MMCHS1.MMCHS_RSP54 and MMCHS1.MMCHS_RSP76 registers.

Table 22-18. Sending CMD2

Register Name	Register Address	Value	Value Description
MMCHS1.MMCHS_CON	0x4809C02C	0x00000001	MMC bus is still in open drain state for broadcast.
MMCHS1.MMCHS_IE	0x4809C134	0x00070001	Enables CERR, CIE, CCRC, CC, CTO and CEB events to occur.
MMCHS1.MMCHS_ISE	0x4809C138	0x00070001	Enables CERR, CIE, CCRC, CC, CTO and CEB interrupts to rise.
MMCHS1.MMCHS_CMD	0x4809C10C	0x02090000	Sends CMD2 whose opcode is 2, response type is 136 bits with CCCE enabled.

22.6.1.3.2.6 Sending CMD3

This command sets MMC card address (see Table 22-19). Useful when MMCHS controller switches to addressed mode.

Table 22-19. Sending CMD3

Register Name	Register Address	Value	Value Description
MMCHS1.MMCHS_CON	0x4809C02C	0x00000001	MMC bus is still in open drain state for broadcast.
MMCHS1.MMCHS_IE	0x4809C134	0x100f0001	Enables CERR, CIE, CCRC, CC, CTO and CEB events to occur.
MMCHS1.MMCHS_ISE	0x4809C138	0x100f0001	Enables CERR, CIE, CCRC, CC, CTO and CEB interrupts to rise.
MMCHS1.MMCHS_CMD	0x4809C10C	0x031a0000	Sends CMD3 whose opcode is 3, response type is 48 bits with CICE and CCCE enabled.
MMCHS1.MMCHS_ARG	0x4809C108	0x00010000	MMCHS_ARG register carries MMC card's address. We choose to assign address 1 any other 16 bit wide value is valid.

22.6.1.3.3 MMC Bus Setting Change After Card Identification

After CMD3 command transfer is completed successfully, an auto-negotiation on voltage value an start. This is the frontier when the MMCHS controller should switch from identification mode to transfer mode. This impacts the controller in a way that it should change its bus state from open drain to push-pull. Table 22-20 gives several registers and their values.

Table 22-20. MMC Bus Setting Change Table

Register Name	Register Address	Value	Value Description
MMCHS1.MMCHS_CON	0x4809C02C	0x00000000	Bus is now in push-pull mode.
MMCHS1.MMCHS_HCTL	0x4809C128	0x00000B00	Bus power is on, 1.8V is selected.

Table 22-20. MMC Bus Setting Change Table (continued)

Register Name	Register Address	Value	Value Description
MMCHS1.MMCHS_SYSCTL	0x4809C12C	0x00003C07	MMCHS controller's internal clock is stable and enabled, MMC card's clock is on. Divider value is 240 which means that MMCHS controller is still supplying a 400 KHz clock.

22.6.1.3.4 Reading the CSD Register of a MMC Card

After settling on a voltage value, additional information must be read from the MMC card. This data is stored in MMC card CSD register. The card sends CSD register content after receiving CMD9 command. The CSD register holds important information on the card, MMC system specification version support, maximum clock speed support, memory capacity, minimum block length, read and write transfer latency timings.

22.6.1.3.4.1 Sending CMD9

This command asks the card to send its csd register's content (see Table 22-21). The 136 bit (128 bits are valid payload) response is received in MMCHS1.MMCHS_RSP10, MMCHS1.MMCHS_RSP32, MMCHS1.MMCHS_RSP54 and MMCHS1.MMCHS_RSP76 registers. CMD9 is an addressed command which means that card's address must be written in MMCHS1.MMCHS_ARG register before the command is issued.

Table 22-21. Sending CMD9

Register Name	Register Address	Value	Value Description
MMCHS1.MMCHS_CON	0x4809C02C	0x00000000	MMC bus is in push-pull mode.
MMCHS1.MMCHS_IE	0x4809C134	0x00070001	Enables CCRC, CC, CTO and CEB events to occur.
MMCHS1.MMCHS_ISE	0x4809C138	0x00070001	Enables CCRC, CC, CTO and CEB interrupts to rise.
MMCHS1.MMCHS_CMD	0x4809C10C	0x09090000	Sends CMD9 whose opcode is 9, response type is 136 bits with CCCE enabled.
MMCHS1.MMCHS_ARG	0x4809C108	0x00010000	MMCHS_ARG register carries MMC card's address. We choose to assign address 1 any other 16 bit wide value is valid.

After receiving and parsing CMD9 response, MMC card clock speed must change to take advantage to card's maximum speed. This has an impact on MMC bus as we should perform a clock frequency change. At this stage the maximum clock frequency will be 20 MHz (please refer to MMC system specification from www.mmca.org).

Clock frequency change procedure is performed in several steps. Please refer to Section 22.5, MMC/SD/SDIO Basic Programming Model, Section 22.5.2.7.2, *MMCHS Clock Frequency Change*.

Table 22-22 shows the value written in MMCHS1.MMCHS_SYSCTL register.

Table 22-22. MMCHS_SYSCTL Value

Register Name	Register Address	Value	Value Description
MMCHS1.MMCHS_SYSCTL	0x4809C12C	0x00000147	MMCHS controller's internal clock is stable and enabled, MMC card's clock is on. Divider value is 5 which means that MMCHS controller is supplying a 19.2 MHz clock < 20 MHz.

Another important parameter read from CSD register is MMC system specification version. If this parameter points to a value greater than or equal to 4, the MMC card is capable of a speed up to 52 MHz and a bus width up to 8 (1, 4 or 8 are the possible options). In order to enable these two extra features, MMCHS controller must issue a CMD6 command with specific argument.

A CMD6 command is issued in the data transfer mode after MMC card is selected. MMC card selection consists of sending CMD7 command with MMC card's address in command argument.

Table 22-23 shows the set of register impacted by CMD7 issue action.

Table 22-23. Sending CMD7

Register Name	Register Address	Value	Value Description
MMCHS1.MMCHS_CON	0x4809C02C	0x00000000	MMC bus is in push-pull mode.
MMCHS1.MMCHS_IE	0x4809C134	0x100f0001	Enables CERR, CIE, CCRC, CC, CTO and CEB events to occur.
MMCHS1.MMCHS_ISE	0x4809C138	0x100f0001	Enables CERR, CIE, CCRC, CC, CTO and CEB interrupts to rise.
MMCHS1.MMCHS_CMD	0x4809C10C	0x071a0000	Sends CMD7 whose opcode is 7, response type is 48 bits with CICE and CCCE enabled.
MMCHS1.MMCHS_ARG	0x4809C108	0x00010000	MMCHS_ARG register carries MMC card's address. We choose to assign address 1 any other 16 bit wide value is valid.

After a CMD7 transfer is complete, the MMC card is ready to receive a CMD6 command. CMD6 command is used to write a byte in MMC card extended CSD register (ext_csd). It is an IO access function. There are two write actions, the first one enables a specific data bus width in the card. For our use case we used maximum data bus width 8. The second one enables high speed feature in the card.

22.6.1.3.4.1.1 Setting Data Bus Width to 8

Table 22-24 shows the set of registers impacted by issuing CMD6.

Table 22-24. Setting Data Bus Width with CMD6

Register Name	Register Address	Value	Value Description
MMCHS1.MMCHS_CON	0x4809C02C	0x00000000	MMC bus is in push-pull mode.
MMCHS1.MMCHS_IE	0x4809C134	0x100f0001	Enables CERR, CIE, CCRC, CC, CTO and CEB events to occur.
MMCHS1.MMCHS_ISE	0x4809C138	0x100f0001	Enables CERR, CIE, CCRC, CC, CTO and CEB interrupts to rise.
MMCHS1.MMCHS_CMD	0x4809C10C	0x061b0000	Sends CMD6 whose opcode is 6, response type is 48 bits with busy, with CICE and CCCE enabled.
MMCHS1.MMCHS_ARG	0x4809C108	0x03b70200	(3 << 24) | (byte_address << 16) | (byte_value << 8). byte_address is the byte address in ext_csd register.

After issuing CMD6 completes successfully and MMC card leaves busy state, MMCHS controller should change its data bus width. This is done by changing MMCHS1.MMCHS_CON configuration value.

Table 22-25. MMCHS_CON Value

Register Name	Register Address	Value	Value Description
MMCHS1.MMCHS_CON	0x4809C02C	0x00000020	MMCHS controller's data bus width is set to 8.

22.6.1.3.4.1.2 Enable High Speed Feature

Table 22-26 shows the set of registers impacted by issuing CMD6 issuing.

Table 22-26. Enabling High Speed with CMD6

Register Name	Register Address	Value	Value Description
MMCHS1.MMCHS_CON	0x4809C02C	0x00000020	MMC bus is in push-pull mode. DW8 is enabled.
MMCHS1.MMCHS_IE	0x4809C134	0x100f0001	Enables CERR, CIE, CCRC, CC, CTO and CEB events to occur.
MMCHS1.MMCHS_ISE	0x4809C138	0x100f0001	Enables CERR, CIE, CCRC, CC, CTO and CEB interrupts to rise.
MMCHS1.MMCHS_CMD	0x4809C10C	0x061b0000	Sends CMD6 whose opcode is 6, response type is 48 bits with busy, with CICE and CCCE enabled.
MMCHS1.MMCHS_ARG	0x4809C108	0x03b90100	(3 << 24) | (byte_address << 16) | (byte_value << 8). byte_address is the byte address in ext_csd register.

After issuing CMD6 completes successfully and MMC card leaves busy state, MMCHS controller should now change its output clock to bring it to 48 MHz. 52 MHz, max frequency value supported by MMC card version 4 and above, is not supported because 96 MHz, MMCHS controller functional clock, is not a multiple of 52 MHz. We fall off to 48 MHz.

Table 22-27. MMCHS_SYSCTL Value

Register Name	Register Address	Value	Value Description
MMCHS1.MMCHS_SYSCTL	0x4809C12C	0x00000087	MMCHS controller's internal clock is stable and enabled, MMC card's clock is on. Divider value is 2 which means that MMCHS controller is supplying a 48 MHz clock.

22.6.1.3.5 MMC Write Transfer

Either data read or data write transfer uses DMA controller to perform memory (DDRAM) to/from MMCHS controller transfers. The DMA part is not described in this chapter, it is described in Chapter 9, *DMA*.

Before any data transfer begins, the card must selected by issuing CMD7 command (Table 22-23).

A write transfer is a finite multiple block write transfer. In order to perform a write transfer, the following steps must performed.

22.6.1.3.5.1 Send CMD16

Issuing CMD16 allows to set the block length. For our use case we decided to use a static block length of 512 bytes. The block length value is passed to the MMC card via MMCHS1.MMCHS_ARG register. The registers impacted by this operation are as follows:

Table 22-28. Setting Block Length

Register Name	Register Address	Value	Value Description
MMCHS1.MMCHS_CON	0x4809C02C	0x00000020	MMC bus is in push-pull mode. DW8 is enabled.
MMCHS1.MMCHS_IE	0x4809C134	0x100f0001	Enables CERR, CIE, CCRC, CC, CTO and CEB events to occur.
MMCHS1.MMCHS_ISE	0x4809C138	0x100f0001	Enables CERR, CIE, CCRC, CC, CTO and CEB interrupts to rise.
MMCHS1.MMCHS_CMD	0x4809C10C	0x101a0000	Sends CMD16 whose opcode is 16, response type is 48 bits, with CICE and CCCE enabled.
MMCHS1.MMCHS_ARG	0x4809C108	0x00000200	Block length is 512 = 0x200

22.6.1.3.5.2 Send CMD23

Issuing CMD23 allows to set the number of how many 512-byte blocks the MMC card should expect from the MMCHS controller. The number of blocks is passed to MMC card via MMCHS1.MMCHS_ARG register. The registers impacted by this operation are as follows:

Table 22-29. Setting Number of Blocks

Register Name	Register Address	Value	Value Description
MMCHS1.MMCHS_CON	0x4809C02C	0x00000020	MMC bus is in push-pull mode. DW8 is enabled.
MMCHS1.MMCHS_IE	0x4809C134	0x100f0001	Enables CERR, CIE, CCRC, CC, CTO and CEB events to occur.
MMCHS1.MMCHS_ISE	0x4809C138	0x100f0001	Enables CERR, CIE, CCRC, CC, CTO and CEB interrupts to rise.
MMCHS1.MMCHS_CMD	0x4809C10C	0x171a0000	Sends CMD23 whose opcode is 23, response type is 48 bits, with CICE and CCCE enabled.
MMCHS1.MMCHS_ARG	0x4809C108	0x00000008	Number of 512-byte blocks in 4 KB buffer is 8.

22.6.1.3.5.3 Send CMD25

Issuing CMD25 starts the finite, multiple block write transfer. Before the transfer starts, DMA controller should be configured for this operation. For more details about DMA configuration please refer to Chapter 9, *DMA*.

Table 22-30. CMD25 Issuing

Register Name	Register Address	Value	Value Description
MMCHS1.MMCHS_CON	0x4809C02C	0x00000020	MMC bus is in push-pull mode. DW8 is enabled.
MMCHS1.MMCHS_IE	0x4809C134	0x107f0013	Enables CERR, CIE, CCRC, CC, TC, BWR, CTO, DTO, DCRC, DEB and CEB events to occur.
MMCHS1.MMCHS_ISE	0x4809C138	0x107f0013	Enables CERR, CIE, CCRC, CC, TC, BWR, CTO, DTO, DCRC, DEB and CEB interrupts to rise.
MMCHS1.MMCHS_CMD	0x4809C10C	0x193a0023	Sends CMD25 whose opcode is 25, response type is 48 bits, with CICE, DP, MSBS, BCE, DE and CCCE enabled.

Table 22-30. CMD25 Issuing (continued)

Register Name	Register Address	Value	Value Description
MMCHS1.MMCHS_ARG	0x4809C108	0x00000000	Not used.
MMCHS1.MMCHS_BLK	0x4809C104	0x00080200	(number_blocks << 16) | (block_length)

22.6.1.3.6 MMC Read Transfer

Either data read or data write transfer uses DMA controller to perform memory (DDRAM) to/from MMCHS controller transfers. The DMA part is not described in this document, it is described in Chapter 9, *DMA*.

Before any data transfer begins, the card must selected by issuing CMD7 command (Table 22-23).

A read transfer is a finite multiple block write transfer. In order to perform a read transfer, the following steps must performed.

22.6.1.3.6.1 Send CMD16

Issuing CMD16 allows to set the block length. For our use case we decided to use a static block length of 512 bytes. The block length value is passed to the MMC card via MMCHS1.MMCHS_ARG register. See Table 22-28.

22.6.1.3.6.2 Send CMD23

Issuing CMD23 allows to set the number of how many 512-byte blocks the MMC card should expect from the MMCHS controller. The number of blocks is passed to MMC card via MMCHS1.MMCHS_ARG register. See Table 22-29.

22.6.1.3.6.3 Send CMD18

Issuing CMD18 starts the finite, multiple block read transfer. Before the transfer starts, DMA controller should be configured for this operation. For more details about DMA configuration please refer to Chapter 9, *DMA*.

Table 22-31. CMD18 Issuing

Register Name	Register Address	Value	Value Description
MMCHS1.MMCHS_CON	0x4809C02C	0x00000020	MMC bus is in push-pull mode. DW8 is enabled.
MMCHS1.MMCHS_IE	0x4809C134	0x107f0023	Enables CERR, CIE, CCRC, CC, TC, BRR, CTO, DTO, DCRC, DEB and CEB events to occur.
MMCHS1.MMCHS_ISE	0x4809C138	0x107f0023	Enables CERR, CIE, CCRC, CC, TC, BRR, CTO, DTO, DCRC, DEB and CEB interrupts to rise.
MMCHS1.MMCHS_CMD	0x4809C10C	0x123a0033	Sends CMD18 whose opcode is 18, response type is 48 bits, with CICE, DP, MSBS, BCE, DE, DDIR and CCCE enabled.
MMCHS1.MMCHS_ARG	0x4809C108	0x00000000	Not used.
MMCHS1.MMCHS_BLK	0x4809C104	0x00080200	(number_blocks << 16) | (block_length)

22.6.1.3.7 Dealing with High Capacity Cards

Unlike standard-capacity cards, memory addressing mode for high-capacity cards (SDHC and HC MMC) is in 512-byte block format instead of byte format. This means that byte and partial accesses are not allowed with high-capacity cards.

The 32-bit wide argument configured in the MMCHS_ARG register and sent in data transfer commands CMD17, CMD18, CMD24, and CMD25, carries the block index where the read/write should start and not the 32-bit byte address.

For high-capacity cards, the block size is fixed at 512 bytes. Any data transfer, when the data bus is involved, must be a multiple of 512 bytes.

The number of blocks for a high-capacity card, which determines the capacity of the card block_count x 512 bytes, is accessible through field C_SIZE in the card s CSD register version 2.0 for SD cards, or through the SEC_COUNT field in the card s EXT_CSD register for MMC cards compliant with MMC specification version 4.x.

To write/read a packet whose size is less than 512 bytes to/from a high-capacity card, write/read the whole 512-byte block that contains the range of bytes to modify/read.

22.7 MMC/SD/SDIO Register Manual

22.7.1 MMC/SD/SDIO Instance Summary

Table 22-32 lists the instance summary.

Table 22-32. Instance Summary

Module Name	Base Address	Size
MMCHS1	0x4809 C000	512 bytes
MMCHS2	0x480B 4000	512 bytes
MMCHS3	0x480A D000	512 bytes

22.7.2 MMC/SD/SDIO Registers Mapping Summary

Table 22-33 lists the MMC/SD/SDIO1 to MMC/SD/SDIO3 registers. Table 22-34 through Table 22-80 describe the register bits.

CAUTION

The MMC/SD/SDIOi registers are limited to 32-bit data accesses. 16-bit and 8-bit are not allowed and can corrupt register content.

Table 22-33. MMC/SD/SDIO1 Register Summary

Register	Type	Register Width (Bits)	Offset Address	MMCHS1 Physical Address	MMCHS2 Physical Address	MMCHS3 Physical Address
MMCHS_SYSCONFIG	RW	32	0x10	0x4809 C010	0x480B 4010	0x480A D010
MMCHS_SYSSTATUS	R	32	0x14	0x4809 C014	0x480B 4014	0x480A D014
MMCHS_CSRE	RW	32	0x24	0x4809 C024	0x480B 4024	0x480A D024
MMCHS_SYSTEST	RW	32	0x28	0x4809 C028	0x480B 4028	0x480A D028
MMCHS_CON	RW	32	0x2C	0x4809 C02C	0x480B 402C	0x480A D02C
MMCHS_PWCNT	RW	32	0x30	0x4809 C030	0x480B 4030	0x480A D030
MMCHS_BLK	RW	32	0x104	0x4809 C104	0x480B 4104	0x480A D104
MMCHS_ARG	RW	32	0x108	0x4809 C108	0x480B 4108	0x480A D108
MMCHS_CMD	RW	32	0x10C	0x4809 C10C	0x480B 410C	0x480A D10C
MMCHS_RSP10	R	32	0x110	0x4809 C110	0x480B 4110	0x480A D110
MMCHS_RSP32	R	32	0x114	0x4809 C114	0x480B 4114	0x480A D114
MMCHS_RSP54	R	32	0x118	0x4809 C118	0x480B 4118	0x480A D118
MMCHS_RSP76	R	32	0x11C	0x4809 C11C	0x480B 411C	0x480A D11C
MMCHS_DATA	RW	32	0x120	0x4809 C120	0x480B 4120	0x480A D120
MMCHS_PSTATE	R	32	0x124	0x4809 C124	0x480B 4124	0x480A D124
MMCHS_HCTL	RW	32	0x128	0x4809 C128	0x480B 4128	0x480A D128
MMCHS_SYSCTL	RW	32	0x12C	0x4809 C12C	0x480B 412C	0x480A D12C
MMCHS_STAT	RW	32	0x130	0x4809 C130	0x480B 4130	0x480A D130
MMCHS_IE	RW	32	0x134	0x4809 C134	0x480B 4134	0x480A D134
MMCHS_ISE	RW	32	0x138	0x4809 C138	0x480B 4138	0x480A D138
MMCHS_AC12	R	32	0x13C	0x4809 C13C	0x480B 413C	0x480A D13C
MMCHS_CAPA	RW	32	0x140	0x4809 C140	0x480B 4140	0x480A D140
MMCHS_CUR_CAPA	RW	32	ox148	0x4809 C148	0x480B 4148	0x480A D148

Table 22-33. MMC/SD/SDIO1 Register Summary (continued)

Register	Type	Register Width (Bits)	Offset Address	MMCHS1 Physical Address	MMCHS2 Physical Address	MMCHS3 Physical Address
MMCHS_REV	R	32	0x1FC	0x4809 C1FC	0x480B 41FC	0x480A D1FC

Table 22-34. MMCHS_SYSCONFIG

Address Offset	0x010		
Physical Address	0x4809 C010	**Instance**	MMCHS1
	0x480A D010		MMCHS3
	0x480B 4010		MMCHS2
Description	System Configuration Register This register allows controlling various parameters of the Interconnect interface.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																						CLOCKACTIVITY		Reserved			SIDLEMODE		ENAWAKEUP	SOFTRESET	AUTOIDLE

Bits	Field Name	Description	Type	Reset
31:10	Reserved	These bits are initialized to zero, and writes to them are ignored. Reads return 0	R	0x00000
9:8	CLOCKACTIVITY	Clocks activity during wake up mode period. Bit8: Interface clock Bit9: Functional clock 0x0: Interface and Functional clock may be switched off. 0x1: Interface clock is maintained. Functional clock may be switched-off. 0x2: Functional clock is maintained. Interface clock may be switched-off. 0x3: Interface and Functional clocks are maintained.	RW	0x0
7:5	Reserved	These bits are initialized to zero, and writes to them are ignored. Reads return 0	R	0
4:3	SIDLEMODE	Power management 0x0: If an idle request is detected, the MMC/SD/SDIO host controller acknowledges it unconditionally and goes in Inactive mode. Interrupt and DMA requests are unconditionally deasserted. 0x1: If an idle request is detected, the request is ignored and the module keeps on behaving normally. 0x2: If an idle request is detected, the module will switch to wake up mode based on its internal activity, and the wake up capability can be used if the wake up capability is enabled (bit MMCi.MMCHS_SYSCONFIG[2] ENAWAKEUP bit is set to 1). 0x3: Reserved - do not use	RW	0x2
2	ENAWAKEUP	Wake-up feature control 0x0: Wake-up capability is disabled 0x1: Wake-up capability is enabled	RW	1
1	SOFTRESET	Software reset. The bit is automatically reset by the hardware. During reset, it always returns 0. Read 0x0: Normal mode Write 0x0: No effect. Read 0x1: The module is reset.	RW	0

Bits	Field Name	Description	Type	Reset
		Write 0x1: Trigger a module reset.		
0	AUTOIDLE	Internal Clock gating strategy	RW	1
		0x0: Clocks are free-running		
		0x1: Automatic clock gating strategy is applied, based on the interconnect and MMC interface activity		

Table 22-35. Register Call Summary for Register MMCHS_SYSCONFIG

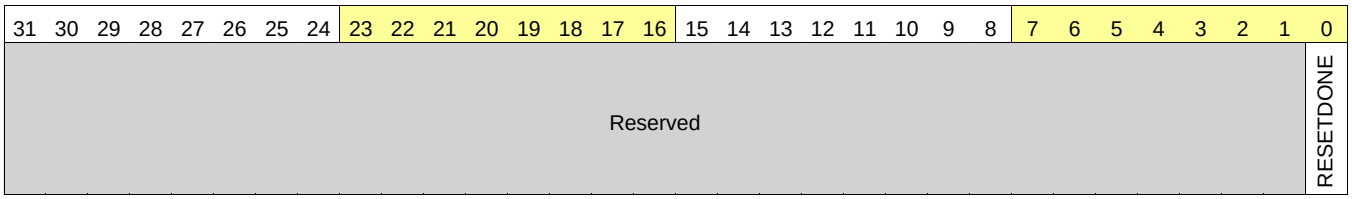
MMC/SD/SDIO Integration
- Clocks: [0] [1] [2] [3] [4] [5] [6]
- Resets: [7]

MMC/SD/SDIO Use Cases and Tips
- Programming Flow: [8] [9]

MMC/SD/SDIO Register Manual
- MMC/SD/SDIO Registers Mapping Summary: [10] [11] [12] [13] [14] [15] [16] [17]

Table 22-36. MMCHS_SYSSTATUS

Address Offset	0x014		
Physical Address	0x4809 C014	**Instance**	MMCHS1
	0x480A D014		MMCHS3
	0x480B 4014		MMCHS2
Description	System Status Register This register provides status information about the module excluding the interrupt status information		
Type	R		



Bits	Field Name	Description	Type	Reset
31:1	Reserved	Reserved bit field. Do not write any value.	R	0x00000000
0	RESETDONE	Internal Reset Monitoring Note: the debounce clock , the interface clock and the functional clock shall be provided to the MMC/SD/SDIO host controller to allow the internal reset monitoring.	R	0
		Read 0x0: Internal module reset is on-going		
		Read 0x1: Reset completed.		

Table 22-37. Register Call Summary for Register MMCHS_SYSSTATUS

MMC/SD/SDIO Integration
- Resets: [0] [1] [2]

MMC/SD/SDIO Use Cases and Tips
- Programming Flow: [3] [4]

MMC/SD/SDIO Register Manual
- MMC/SD/SDIO Registers Mapping Summary: [5]

Table 22-38. MMCHS_CSRE

Address Offset	0x024		
Physical Address	0x4809 C024	**Instance**	MMCHS1
	0x480A D024		MMCHS3
	0x480B 4024		MMCHS2
Description	Card status response error This register enables the host controller to detect card status errors of response type R1, R1b for all cards and of R5, R5b and R6 response for cards types SD or SDIO. When a bit MMCi.MMCHS_CSRE[I] is set to 1, if the corresponding bit at the same position in the response MMCi.MMCHS_RSP10[I] is set to 1, the host controller indicates a card error (MMCi.MMCHS_STAT[28] CERR bit) interrupt status to avoid the host driver reading the response register (MMCi.MMCHS_RSP10). Note: No automatic card error detection for autoCMD12 is implemented; the host system has to check autoCMD12 response register (MMCi.MMCHS_RSP76) for possible card errors.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CSRE																															

Bits	Field Name	Description	Type	Reset
31:0	CSRE	Card status response error	RW	0x00000000

Table 22-39. Register Call Summary for Register MMCHS_CSRE

MMC/SD/SDIO Register Manual

- MMC/SD/SDIO Registers Mapping Summary: [0] [1] [2]

Table 22-40. MMCHS_SYSTEST

Address Offset	0x028		
Physical Address	0x4809 C028	**Instance**	MMCHS1
	0x480A D028		MMCHS3
	0x480B 4028		MMCHS2
Description	System Test register This register is used to control the signals that connect to I/O pins when the module is configured in system test (SYSTEST) mode for boundary connectivity verification. Note: In SYSTEST mode, a write into MMCi.MMCHS_CMD register will not start a transfer. The buffer behaves as a stack accessible only by the local host (push and pop operations). In this mode, the Transfer Block Size (MMCi.MMCHS_BLK[10:0] BLEN bits) and the Blocks count for current transfer (MMCi.MMCHS_BLK[31:16] NBLK bits) are needed to generate a Buffer write ready interrupt (MMCi.MMCHS_STAT[4] BWR bit) or a Buffer read ready interrupt (MMCi.MMCHS_STAT[5] BRR bit) and DMA requests if enabled.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															OBI	SDCD	SDWP	WAKD	SSB	D7D	D6D	D5D	D4D	D3D	D2D	D1D	D0D	DDIR	CDAT	CDIR	MCKD

Bits	Field Name	Description	Type	Reset
31:17	Reserved	Reserved bit field. Do not write any value. Reads return 0.	R	0x00000
16	OBI	Out-Of-Band Interrupt (OBI) data value.	RW	0
		0x0: The Out-of-Band Interrupt pin is driven low.		
		0x1: The Out-of-Band Interrupt pin is driven high.		
15	SDCD	Card detect input signal (SDCD) data value	R	0
		0x0: The card detect pin is driven low.		
		0x1: The card detect pin is driven high.		
14	SDWP	Write protect input signal (SDWP) data value.	R	0

Bits	Field Name	Description		Type	Reset
		0x0:	The write protect pin SDWP is driven low.		
		0x1:	The write protect pin SDWP is driven high.		
13	WAKD	Wake request output signal data value.		RW	0
		Read 0x0:	No action. Returns 0.		
		Write 0x0:	The pin SWAKEUP is driven low.		
		Read 0x1:	No action. Returns 1.		
		Write 0x1:	The pin SWAKEUP is driven high.		
12	SSB	Set status bit This bit must be cleared prior attempting to clear a status bit of the interrupt status register (MMCi.MMCHS_STAT).		RW	0
		Read 0x0:	No action. Returns 0.		
		Write 0x0:	Clear this SSB bit field. Writing 0 does not clear already set status bits.		
		Read 0x1:	No action. Returns 1.		
		Write 0x1:	Force to 1 all status bits of the interrupt status register (MMCi.MMCHS_STAT) only if the corresponding bit field in the Interrupt signal enable register (MMCi.MMCHS_ISE) is set.		
11	D7D	DAT7 input/output signal data value.		RW	0
		Read 0x0:	If MMCi.MMCHS_SYSTEST[3] DDIR bit = 1 (input mode direction), returns the value on the DAT7 line (low). If MMCi.MMCHS_SYSTEST[3] DDIR bit = 0 (output mode direction), returns 0.		
		Write 0x0:	If MMCi.MMCHS_SYSTEST[3] DDIR bit = 0 (output mode direction), the DAT7 line is driven low. If MMCi.MMCHS_SYSTEST[3] DDIR bit = 1 (input mode direction), no effect.		
		Read 0x1:	If MMCi.MMCHS_SYSTEST[3] DDIR bit = 1 (input mode direction), returns the value on the DAT7 line (high) If MMCi.MMCHS_SYSTEST[3] DDIR bit = 0 (output mode direction), returns 1.		
		Write 0x1:	If MMCi.MMCHS_SYSTEST[3] DDIR bit = 0 (output mode direction), the DAT7 line is driven high. If MMCi.MMCHS_SYSTEST[3] DDIR bit = 1 (input mode direction), no effect.		
10	D6D	DAT6 input/output signal data value.		RW	0
		Read 0x0:	If MMCi.MMCHS_SYSTEST[3] DDIR bit = 1 (input mode direction), returns the value on the DAT6 line (low). If MMCi.MMCHS_SYSTEST[3] DDIR bit = 0 (output mode direction), returns 0.		
		Write 0x0:	If MMCi.MMCHS_SYSTEST[3] DDIR bit = 0 (output mode direction), the DAT6 line is driven low. If MMCi.MMCHS_SYSTEST[3] DDIR bit = 1 (input mode direction), no effect.		
		Read 0x1:	If MMCi.MMCHS_SYSTEST[3] DDIR bit = 1 (input mode direction), returns the value on the DAT6 line (high) If MMCi.MMCHS_SYSTEST[3] DDIR bit = 0 (output mode direction), returns 1.		
		Write 0x1:	If MMCi.MMCHS_SYSTEST[3] DDIR bit = 0 (output mode direction), the DAT6 line is driven high. If MMCi.MMCHS_SYSTEST[3] DDIR bit = 1 (input mode direction), no effect.		
9	D5D	DAT5 input/output signal data value.		RW	0

Bits	Field Name	Description	Type	Reset
		Read 0x0: If MMCi.MMCHS_SYSTEST[3] DDIR bit = 1 (input mode direction), returns the value on the DAT5 line (low). If MMCi.MMCHS_SYSTEST[3] DDIR bit = 0 (output mode direction), returns 0.		
		Write 0x0: If MMCi.MMCHS_SYSTEST[3] DDIR bit = 0 (output mode direction), the DAT5 line is driven low. If MMCi.MMCHS_SYSTEST[3] DDIR bit = 1 (input mode direction), no effect.		
		Read 0x1: If MMCi.MMCHS_SYSTEST[3] DDIR bit = 1 (input mode direction), returns the value on the DAT5 line (high) If MMCi.MMCHS_SYSTEST[3] DDIR bit = 0 (output mode direction), returns 1.		
		Write 0x1: If MMCi.MMCHS_SYSTEST[3] DDIR bit = 0 (output mode direction), the DAT5 line is driven high. If MMCi.MMCHS_SYSTEST[3] DDIR bit = 1 (input mode direction), no effect.		
8	D4D	DAT4 input/output signal data value.	RW	0
		Read 0x0: If MMCi.MMCHS_SYSTEST[3] DDIR bit = 1 (input mode direction), returns the value on the DAT4 line (low). If MMCi.MMCHS_SYSTEST[3] DDIR bit = 0 (output mode direction), returns 0.		
		Write 0x0: If MMCi.MMCHS_SYSTEST[3] DDIR bit = 0 (output mode direction), the DAT4 line is driven low. If MMCi.MMCHS_SYSTEST[3] DDIR bit = 1 (input mode direction), no effect.		
		Read 0x1: If MMCi.MMCHS_SYSTEST[3] DDIR bit = 1 (input mode direction), returns the value on the DAT4 line (high) If MMCi.MMCHS_SYSTEST[3] DDIR bit = 0 (output mode direction), returns 1.		
		Write 0x1: If MMCi.MMCHS_SYSTEST[3] DDIR bit = 0 (output mode direction), the DAT4 line is driven high. If MMCi.MMCHS_SYSTEST[3] DDIR bit = 1 (input mode direction), no effect.		
7	D3D	DAT3 input/output signal data value.	RW	0
		Read 0x0: If MMCi.MMCHS_SYSTEST[3] DDIR bit = 1 (input mode direction), returns the value on the DAT3 line (low). If MMCi.MMCHS_SYSTEST[3] DDIR bit = 0 (output mode direction), returns 0.		
		Write 0x0: If MMCi.MMCHS_SYSTEST[3] DDIR bit = 0 (output mode direction), the DAT3 line is driven low. If MMCi.MMCHS_SYSTEST[3] DDIR bit = 1 (input mode direction), no effect.		
		Read 0x1: If MMCi.MMCHS_SYSTEST[3] DDIR bit = 1 (input mode direction), returns the value on the DAT3 line (high) If MMCi.MMCHS_SYSTEST[3] DDIR bit = 0 (output mode direction), returns 1.		
		Write 0x1: If MMCi.MMCHS_SYSTEST[3] DDIR bit = 0 (output mode direction), the DAT3 line is driven high. If MMCi.MMCHS_SYSTEST[3] DDIR bit = 1 (input mode direction), no effect.		
6	D2D	DAT2 input/output signal data value.	RW	0
		Read 0x0: If MMCi.MMCHS_SYSTEST[3] DDIR bit = 1 (input mode direction), returns the value on the DAT2 line (low). If MMCi.MMCHS_SYSTEST[3] DDIR bit = 0 (output mode direction), returns 0.		

Bits	Field Name	Description	Type	Reset
		Write 0x0: If MMCi.MMCHS_SYSTEST[3] DDIR bit = 0 (output mode direction), the DAT2 line is driven low. If MMCi.MMCHS_SYSTEST[3] DDIR bit = 1 (input mode direction), no effect.		
		Read 0x1: If MMCi.MMCHS_SYSTEST[3] DDIR bit = 1 (input mode direction), returns the value on the DAT2 line (high) If MMCi.MMCHS_SYSTEST[3] DDIR bit = 0 (output mode direction), returns 1.		
		Write 0x1: If MMCi.MMCHS_SYSTEST[3] DDIR bit = 0 (output mode direction), the DAT2 line is driven high. If MMCi.MMCHS_SYSTEST[3] DDIR bit = 1 (input mode direction), no effect.		
5	D1D	DAT1 input/output signal data value.	RW	0
		Read 0x0: If MMCi.MMCHS_SYSTEST[3] DDIR bit = 1 (input mode direction), returns the value on the DAT1 line (low). If MMCi.MMCHS_SYSTEST[3] DDIR bit = 0 (output mode direction), returns 0.		
		Write 0x0: If MMCi.MMCHS_SYSTEST[3] DDIR bit = 0 (output mode direction), the DAT1 line is driven low. If MMCi.MMCHS_SYSTEST[3] DDIR bit = 1 (input mode direction), no effect.		
		Read 0x1: If MMCi.MMCHS_SYSTEST[3] DDIR bit = 1 (input mode direction), returns the value on the DAT1 line (high) If MMCi.MMCHS_SYSTEST[3] DDIR bit = 0 (output mode direction), returns 1.		
		Write 0x1: If MMCi.MMCHS_SYSTEST[3] DDIR bit = 0 (output mode direction), the DAT1 line is driven high. If MMCi.MMCHS_SYSTEST[3] DDIR bit = 1 (input mode direction), no effect.		
4	D0D	DAT0 input/output signal data value.	RW	0
		Read 0x0: If MMCi.MMCHS_SYSTEST[3] DDIR bit = 1 (input mode direction), returns the value on the DAT0 line (low). If MMCi.MMCHS_SYSTEST[3] DDIR bit = 0 (output mode direction), returns 0.		
		Write 0x0: If MMCi.MMCHS_SYSTEST[3] DDIR bit = 0 (output mode direction), the DAT0 line is driven low. If MMCi.MMCHS_SYSTEST[3] DDIR bit = 1 (input mode direction), no effect.		
		Read 0x1: If MMCi.MMCHS_SYSTEST[3] DDIR bit = 1 (input mode direction), returns the value on the DAT0 line (high) If MMCi.MMCHS_SYSTEST[3] DDIR bit = 0 (output mode direction), returns 1.		
		Write 0x1: If MMCi.MMCHS_SYSTEST[3] DDIR bit = 0 (output mode direction), the DAT0 line is driven high. If MMCi.MMCHS_SYSTEST[3] DDIR bit = 1 (input mode direction), no effect.		
3	DDIR	Control of the DAT[7:0] pins direction.	RW	0
		Read 0x0: No action. Returns 0.		
		Write 0x0: The DAT lines are outputs (host to card).		
		Read 0x1: No action. Returns 1.		
		Write 0x1: The DAT lines are inputs (card to host).		
2	CDAT	CMD input/output signal data value.	RW	0
		Read 0x0: If MMCi.MMCHS_SYSTEST[1] CDIR bit = 1 (input mode direction), returns the value on the CMD line (low). If MMCi.MMCHS_SYSTEST[1] CDIR bit = 0 (output mode direction), returns 0 .		

Bits	Field Name	Description		Type	Reset
		Write 0x0:	If MMCi.MMCHS_SYSTEST[1] CDIR bit = 0 (output mode direction), the CMD line is driven low. If MMCi.MMCHS_SYSTEST[1] CDIR bit = 1 (input mode direction), no effect.		
		Read 0x1:	If MMCi.MMCHS_SYSTEST[1] CDIR bit = 1 (input mode direction), returns the value on the CMD line (high) If MMCi.MMCHS_SYSTEST[1] CDIR bit = 0 (output mode direction), returns 1 .		
		Write 0x1:	If MMCi.MMCHS_SYSTEST[1] CDIR bit = 0 (output mode direction), the CMD line is driven high. If MMCi.MMCHS_SYSTEST[1] CDIR bit = 1 (input mode direction), no effect.		
1	CDIR	Control of the CMD pin direction.		RW	0
		Read 0x0:	No action. Returns 0.		
		Write 0x0:	The CMD line is an output (host to card).		
		Read 0x1:	No action. Returns 1.		
		Write 0x1:	The CMD line is an input (card to host) .		
0	MCKD	MMC clock output signal data value.		RW	0
		Read 0x0:	No action. Returns 0.		
		Write 0x0:	The output clock is driven low.		
		Read 0x1:	No action. Returns 1.		
		Write 0x1:	The output clock is driven high.		

Table 22-41. Register Call Summary for Register MMCHS_SYSTEST

MMC/SD/SDIO Register Manual

- MMC/SD/SDIO Registers Mapping Summary: [0] [1] [2] [3] [4] [5] [6] [7] [8] [9] [10] [11] [12] [13] [14] [15] [16] [17] [18] [19] [20] [21] [22] [23] [24] [25] [26] [27] [28] [29] [30] [31] [32] [33] [34] [35] [36] [37] [38] [39] [40] [41] [42] [43] [44] [45] [46] [47] [48] [49] [50] [51] [52] [53] [54] [55] [56] [57] [58] [59] [60] [61] [62] [63] [64] [65] [66] [67] [68] [69] [70] [71] [72]

Table 22-42. MMCHS_CON

Address Offset	0x02C		
Physical Address	0x4809 C02C	**Instance**	MMCHS1
	0x480A D02C		MMCHS3
	0x480B 402C		MMCHS2
Description	Configuration register This register is used: - to select the functional mode for any card. - to send an initialization sequence to any card. - to enable the detection on the mmci_dat[1] signal of a card interrupt for SDIO cards only. And also to configure: - specific data and command transfers for MMC cards only. - the parameters related to the card detect and write protect input signals.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															CLKEXTFREE	PADEN	OBIE	OBIP	CEATA	CTPL	DVAL		WPP	CDP	MIT	DW8	MODE	STR	HR	INIT	OD

Bits	Field Name	Description	Type	Reset
31:17	Reserved	Reserved bit field. Do not write any value	R	0x00000
16	CLKEXTFREE	External clock free running. This register is used to maintain card clock out of transfer transaction to enable slave module (for example to generate a synchronous interrupt on mmci_dat[1]). The Clock will be maintain only if MMCi.MMCHS_SYSCTL[2] CEN bit is set. 0x0: External card clock is cut off outside active transaction period. 0x1: External card clock is maintain even out of active transaction period only if MMCi.MMCHS_SYSCTL[2] CEN bit is set.	RW	0
15	PADEN	Control Power for MMC Lines. This register is only useful when MMC PADs contain power saving mechanism to minimize its leakage power. It works as a GPIO that directly control the ACTIVE pin of PADs. Excepted for mmci_dat[1], the signal is also combine outside the module with the dedicated power control MMCi.MMCHS_CON[11] CTPL bit. 0x0: ADPIDLE module pin is not forced, it is automatically generated by the MMC fsms. 0x1: ADPIDLE module pin is forced to active state.	RW	0
14	OBIE	Out-of-Band Interrupt Enable (MMC cards only). This bit enables the detection of Out-of-Band Interrupt on MMCOBI input pin. The usage of the Out-of-Band signal (OBI) is optional and depends on the system integration. 0x0: Out-of-Band interrupt detection disabled. 0x1: Out-of-Band interrupt detection enabled.	RW	0
13	OBIP	Out-of-Band Interrupt Polarity (MMC cards only). This bit selects the active level of the out-of-band interrupt coming from MMC cards. The usage of the Out-of-Band signal (OBI) is optional and depends on the system integration. 0x0: active high level. 0x1: active low level.	RW	0
12	CEATA	CE-ATA control mode (MMC cards compliant with CE-ATA): By default, this bit is set to 0. It is use to indicate that next commands are considered as specific CE-ATA commands that potentially use 'command completion' features. 0x0: Standard MMC/SD/SDIO mode. 0x1: CE-ATA mode. Next commands are considered as CE-ATA commands.	RW	0
11	CTPL	Control Power for mmci_dat[1] line (MMC and SD cards): By default, this bit is set to 0 and the host controller automatically disables all the input buffers outside of a transaction to minimize the leakage current. SDIO cards: When this bit is set to 1, the host controller automatically disables all the input buffers except the buffer of mmci_dat[1] outside of a transaction in order to detect asynchronous card interrupt on mmci_dat[1] line and minimize the leakage current of the buffers. 0x0: Disable all the input buffers outside of a transaction. 0x1: Disable all the input buffers except the buffer of mmci_dat[1] outside of a transaction.	RW	0
10:9	DVAL	Debounce filter value (All cards) This register is used to define a debounce period to filter the card detect input signal (SDCD). The usage of the card detect input signal (SDCD) is optional and depends on the system integration and the type of the connector housing that accommodates the card. 0x0: 33 us debounce period. 0x1: 231 us debounce period.	RW	0x3

Bits	Field Name	Description	Type	Reset
		0x2: 1 ms debounce period. 0x3: 8.4 ms debounce period.		
8	WPP	Write protect polarity (SD and SDIO cards only) This bit selects the active level of the write protect input signal (SDWP). The usage of the write protect input signal (SDWP) is optional and depends on the system integration and the type of the connector housing that accommodates the card. 0x0: Active high level. 0x1: Active low level.	RW	0
7	CDP	Card detect polarity (All cards) This bit selects the active level of the card detect input signal (SDCD). The usage of the card detect input signal (SDCD) is optional and depends on the system integration and the type of the connector housing that accommodates the card. 0x0: Active high level. 0x1: Active low level.	RW	0
6	MIT	MMC interrupt command (Only for MMC cards.) This bit must be set to 1, when the next write access to the command register (MMCi.MMCHS_CMD) is for writing a MMC interrupt command (CMD40) requiring the command timeout detection to be disabled for the command response. 0x0: Command timeout enabled 0x1: Command timeout disabled	RW	0
5	DW8	8-bit mode MMC select For SD/SDIO cards, this bit must be set to 0. For MMC card, this bit must be set following a valid SWITCH command (CMD6) with the correct value and extend CSD index written in the argument. Prior to this command, the MMC card configuration register (CSD and EXT_CSD) must be verified for compliancy with MMC standard specification. 0x0: 1-bit or 4-bit Data width (mmci_dat[0] or mmci_dat[3:0] used, MMC, SD cards) 0x1: 8-bit Data width (mmci_dat[7:0] used, MMC cards)	RW	0
4	MODE	Mode select (All cards) These bits select the functional mode. 0x0: Functional mode. Transfers to the MMC/SD/SDIO cards follow the card protocol. MMC clock is enabled. MMC/SD transfers are operated under the control of the MMCi.MMCHS_CMD register. 0x1: SYSTEST mode. The signal pins are configured as general-purpose input/output and the 1024-byte buffer is configured as a stack memory accessible only by the local host or system DMA. The pins retain their default type (input, output or in-out). SYSTEST mode is operated under the control of the SYSTEST register.	RW	0
3	STR	Stream command (Only for MMC cards). This bit must be set to 1 only for the stream data transfers (read or write) of the adtc commands. Stream read is a class 1 command (CMD11: READ_DAT_UNTIL_STOP). Stream write is a class 3 command (CMD20: WRITE_DAT_UNTIL_STOP). 0x0: Block oriented data transfer. 0x1: Stream oriented data transfer.	RW	0

Bits	Field Name	Description	Type	Reset
2	HR	Broadcast host response (Only for MMC cards). This register is used to force the host to generate a 48-bit response for bc command type. It can be used to terminate the interrupt mode by generating a CMD40 response by the core. In order to have the host response to be generated in open drain mode, the register MMCHS_CON[OD] must be set to 1. When MMCi.MMCHS_CON[12] CEATA bit is set to 1 and MMCi.MMCHS_ARG set to 0x00000000, when writing 0x00000000 into MMCi.MMCHS_CMD register, the host controller performs a 'command completion signal disable' token (i.e. mmci_cmd line held to '0' during 47 cycles followed by a 1). 0x0: The host does not generate a 48-bit response instead of a command. 0x1: The host generates a 48-bit response instead of a command or a command completion signal disable token.	RW	0
1	INIT	Send initialization stream (All cards). When this bit is set to 1, and the card is idle, an initialization sequence is sent to the card. An initialization sequence consists of setting the mmci_cmd line to 1 during 80 clock cycles. The initialization sequence is mandatory - but it is not required to do it through this bit - this bit makes it easier. Clock divider (MMCi.MMCHS_SYSCTL[15:6] CLKD bits) should be set to ensure that 80 clock periods are greater than 1ms. Note: in this mode, there is no command sent to the card and no response is expected. A command complete interrupt will be generated once the initialization sequence is completed. MMCi.MMCHS_STAT[0] CC bit can be polled. 0x0: The host does not send an initialization sequence. 0x1: The host sends an initialization sequence.	RW	0
0	OD	Card open drain mode (Only for MMC cards). This bit must be set to 1 for MMC card commands 1, 2, 3 and 40, and if the MMC card bus is operating in open-drain mode during the response phase to the command sent. Typically, during card identification mode when the card is either in idle, ready or ident state. It is also necessary to set this bit to 1, for a broadcast host response (see Broadcast host response register MMCi.MMCHS_CON[2] HR bit) 0x0: No Open Drain 0x1: Open Drain or Broadcast host response	RW	0

Table 22-43. Register Call Summary for Register MMCHS_CON

MMC/SD/SDIO Functional Description

- MMC CE-ATA Command Completion Disable Management: [0] [1] [2]

MMC/SD/SDIO Use Cases and Tips

- Programming Flow: [3] [4] [5] [6] [7] [8] [9] [10] [11] [12] [13] [14] [15] [16] [17] [18] [19] [20] [21] [22] [23] [24] [25] [26]

MMC/SD/SDIO Register Manual

- MMC/SD/SDIO Registers Mapping Summary: [27] [28] [29] [30] [31] [32] [33] [34] [35] [36] [37] [38] [39]

Table 22-44. MMCHS_PWCNT

Address Offset	0x030		
Physical Address	0x4809 C030	**Instance**	MMCHS1
	0x480A D030		MMCHS3
	0x480B 4030		MMCHS2
Description	Power counter register This register is used to program a mmc counter to delay command transfers after activating the PAD power, this value depends on PAD characteristics and voltage.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																PWRCNT															

Bits	Field Name	Description	Type	Reset
31:16	Reserved	These bits are initialized to zero, and writes to them are ignored. Reads return 0.	R	0x00
15:0	PWRCNT	Power counter register. This register is used to introduce a delay between the PAD ACTIVE pin assertion and the command issued. 0x0: No additional delay added. 0x1: TCF delay (card clock period). 0x2: TCF x 2 delay (card clock period). 0xFFFE: TCF x 65534 delay (card clock period). 0xFFFF: TCF x 65535 delay (card clock period).	RW	0x0000

Table 22-45. Register Call Summary for Register MMCHS_PWCNT

MMC/SD/SDIO Register Manual

- MMC/SD/SDIO Registers Mapping Summary: [0]

Table 22-46. MMCHS_BLK

Address Offset	0x104		
Physical Address	0x4809 C104	**Instance**	MMCHS1
	0x480A D104		MMCHS3
	0x480B 4104		MMCHS2
Description	Transfer Length Configuration register This register shall be used for any card.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
NBLK																Reserved					BLEN										

Bits	Field Name	Description	Type	Reset
31:16	NBLK	Blocks count for current transfer. This register is enabled when Block count Enable (MMCi.MMCHS_CMD[1] BCE bit) is set to 1 and is valid only for multiple block transfers. Setting the block count to 0 results no data blocks being transferred. Note: The host controller decrements the block count after each block transfer and stops when the count reaches zero. This register can be accessed only if no transaction is executing (i.e., after a transaction has stopped). Read operations during transfers may return an invalid value and write operation will be ignored. In suspend context, the number of blocks yet to be transferred can be determined by reading this register. When restoring transfer context prior to issuing a Resume command, The local host shall restore the previously saved block count.	RW	0x0000

Bits	Field Name	Description	Type	Reset
		0x0: Stop count 0x1: 1 block 0x2: 2 blocks 0xFFFF: 65535 blocks		
15:11	Reserved	Reserved bit field. Do not write any value	R	0x00
10:0	BLEN	Transfer Block Size. This register specifies the block size for block data transfers. Read operations during transfers may return an invalid value, and write operations are ignored. When a CMD12 command is issued to stop the transfer, a read of the BLEN field after transfer completion (MMCi.MMCHS_STAT[1] TC bit set to 1) will not return the true byte number of data length while the stop occurs but the value written in this register before transfer is launched. 0x0: No data transfer 0x1: 1 byte block length 0x2: 2 bytes block length 0x3: 3 bytes block length 0x1FF: 511 bytes block length 0x200: 512 bytes block length 0x3FF: 1023 bytes block length 0x400: 1024 bytes block length	RW	0x000

Table 22-47. Register Call Summary for Register MMCHS_BLK

MMC/SD/SDIO Integration • DMA Requests: [0] [1]
MMC/SD/SDIO Functional Description • Data Buffer: [2]
MMC/SD/SDIO Use Cases and Tips • Programming Flow: [3] [4]
MMC/SD/SDIO Register Manual • MMC/SD/SDIO Registers Mapping Summary: [5] [6] [7] [8] [9] [10] [11] [12]

Table 22-48. MMCHS_ARG

Address Offset	0x108		
Physical Address	0x4809 C108	**Instance**	MMCHS1
	0x480A D108		MMCHS3
	0x480B 4108		MMCHS2
Description	Command argument Register This register contains command argument specified as bit 39-8 of Command-Format These registers must be initialized prior to sending the command itself to the card (write action into the register MMCi.MMCHS_CMD register). Only exception is for a command index specifying stuff bits in arguments, making a write unnecessary.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ARG																															

Bits	Field Name	Description	Type	Reset
31:0	ARG	Command argument bits [31:0](1)	RW	0x00000000

(1) For CMD52, ARG has to be programmed with IO_RW_DIRECT[39:8]. Refer to SDIO specification.

Table 22-49. Register Call Summary for Register MMCHS_ARG

MMC/SD/SDIO Functional Description • MMC CE-ATA Command Completion Disable Management: [0]
MMC/SD/SDIO Use Cases and Tips • Programming Flow: [1] [2] [3] [4] [5] [6] [7] [8] [9] [10] [11] [12] [13] [14] [15] [16] [17] [18]
MMC/SD/SDIO Register Manual • MMC/SD/SDIO Registers Mapping Summary: [19] [20]

Table 22-50. MMCHS_CMD

Address Offset	0x10C		
Physical Address	0x4809 C10C	**Instance**	MMCHS1
	0x480A D10C		MMCHS3
	0x480B 410C		MMCHS2
Description	Command and transfer mode register MMCi.MMCHS_CMD[31:16] = the command register MMCi.MMCHS_CMD[15:0] = the transfer mode. This register configures the data and command transfers. A write into the most significant byte send the command. A write into MMCi.MMCHS_CMD[15:0] registers during data transfer has no effect. This register shall be used for any card.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								INDX								CMD_TYPE		DP		CICE		CCCE		Reserved		RSP_TYPE		Reserved										MSBS		DDIR		Reserved		ACEN		BCE		DE

Bits	Field Name	Description	Type	Reset
31:30	Reserved	Reserved bit field. Do not write any value	R	0x0
29:24	INDX	Command index Binary encoded value from 0 to 63 specifying the command number send to card 0x0: CMD0 or ACMD0 0x1: CMD1 or ACMD1 0x3F: CMD63 or ACMD63	RW	0x00
23:22	CMD_TYPE	Command type. This register specifies three types of special command: Suspend, Resume and Abort. These bits shall be set to 0b00 for all other commands. 0x0: Others Commands 0x1: Upon CMD52 "Bus Suspend" operation 0x2: Upon CMD52 "Function Select" operation 0x3: Upon CMD12 or CMD52 "I/O Abort" command	RW	0x0
21	DP	Data present select. This register indicates that data is present and mmci_dat line shall be used. It must be set to 0 in the following conditions: - Command using only mmci_cmd line -Command with no data transfer but using busy signal on mmci_dat[0] -Resume command 0x0: Command with no data transfer 0x1: Command with data transfer	RW	0

Bits	Field Name	Description	Type	Reset
20	CICE	Command Index check enable. This bit must be set to 1 to enable index check on command response to compare the index field in the response against the index of the command. If the index is not the same in the response as in the command, it is reported as a command index error (MMCi.MMCHS_STAT[19] CIE bit set to1) Note: The CICE bit cannot be configured for an Auto CMD12, then index check is automatically checked when this command is issued. 0x0: Index check disable 0x1: Index check enable	RW	0
19	CCCE	Command CRC check enable. This bit must be set to 1 to enable CRC7 check on command response to protect the response against transmission errors on the bus. If an error is detected, it is reported as a command CRC error (MMCi.MMCHS_STAT[17] CCRC bit set to 1). Note: The CCCE bit cannot be configured for an Auto CMD12, and then CRC check is automatically checked when this command is issued. 0x0: CRC7 check disable 0x1: CRC7 check enable	RW	0
18	Reserved	Reserved bit field. Do not write any value.	R	0
17:16	RSP_TYPE	Response type. This bits defines the response type of the command. 0x0: No response 0x1: Response Length 136 bits 0x2: Response Length 48 bits 0x3: Response Length 48 bits with busy after response	RW	0x0
15:6	Reserved	Reserved bit field. Do not write any value.	R	0x000
5	MSBS	Multi/Single block select. This bit must be set to 1 for data transfer in case of multi block command. For any others command this bit shall be set to 0. 0x0: Single block. If this bit is 0, it is not necessary to set the register MMCi.MMCHS_BLK[31:16] NBLK bits. 0x1: Multi block. When Block Count is disabled (MMCi.MMCHS_CMD[1] BCE bit is set to 0) in Multiple block transfers (MMCi.MMCHS_CMD[5] MSBS bit is set to 1), the module can perform infinite transfer.	RW	0
4	DDIR	Data transfer Direction. Select This bit defines either data transfer will be a read or a write. 0x0: Data Write (host to card) 0x1: Data Read (card to host)	RW	0
3	Reserved	Reserved bit field. Do not write any value.	R	0
2	ACEN	Auto CMD12 Enable. (SD cards only). When this bit is set to 1, the host controller issues a CMD12 automatically after the transfer completion of the last block. The Host Driver shall not set this bit to issue commands that do not require CMD12 to stop data transfer. For CE-ATA commands (MMCi.MMCHS_CON[12] CEATA bit set to 1), auto CMD12 is useless; therefore when this bit is set the mechanism to detect command completion signal, named CCS, interrupt is activated. 0x0: Auto CMD12 disable 0x1: Auto CMD12 enable or CCS detection enabled.	RW	0
1	BCE	Block Count Enable (Multiple block transfers only). This bit is used to enable the block count register (MMCHS_BLK[31:16] NBLK bits). When Block Count is disabled (MMCHS_CMD[1] BCE bit is set to 0) in Multiple block transfers (MMCHS_CMD[5] MSBS bits is set to 1), the module can perform infinite transfer. 0x0: Block count disabled for infinite transfer.	RW	0

Bits	Field Name	Description	Type	Reset
		0x1: Block count enabled for multiple block transfer with known number of blocks		
0	DE	DMA Enable. This bit is used to enable DMA mode for host data access. 0x0: DMA mode disable 0x1: DMA mode enable	RW	0

Table 22-51. Register Call Summary for Register MMCHS_CMD

MMC/SD/SDIO Environment
- MMC/SD/SDIO Protocol and Data Format: [0] [1] [2] [3]

MMC/SD/SDIO Integration
- DMA Requests: [4]

MMC/SD/SDIO Functional Description
- Data Buffer: [5]
- Transfer Stop: [6]
- MMC CE-ATA Command Completion Disable Management: [7] [8]

MMC/SD/SDIO Use Cases and Tips
- Programming Flow: [9] [10] [11] [12] [13] [14] [15] [16] [17] [18] [19] [20] [21] [22] [23] [24] [25]

MMC/SD/SDIO Register Manual
- MMC/SD/SDIO Registers Mapping Summary: [26] [27] [28] [29] [30] [31] [32] [33] [34] [35] [36] [37] [38] [39] [40] [41] [42] [43] [44] [45] [46]

Table 22-52. MMCHS_RSP10

Address Offset	0x110		
Physical Address	0x4809 C110	**Instance**	MMCHS1
	0x480A D110		MMCHS3
	0x480B 4110		MMCHS2
Description	Command response[31:0] Register This 32-bit register holds bits positions [31:0] of command response type R1/R1b/R2/R3/R4/R5/R5b/R6		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSP1																RSP0															

Bits	Field Name	Description	Type	Reset
31:16	RSP1	R1/R1b (normal response) /R3/R4/R5/R5b/R6 : Command Response [39:24] R2: Command Response [31:16]	R	0x0000
15:0	RSP0	R1/R1b (normal response) /R3/R4/R5/R5b/R6 : Command Response [23:8] R2: Command Response [15:0]	R	0x0000

Table 22-53. Register Call Summary for Register MMCHS_RSP10

MMC/SD/SDIO Functional Description
- Different Types of Responses: [0] [1] [2]

MMC/SD/SDIO Use Cases and Tips
- Programming Flow: [3] [4] [5] [6] [7] [8] [9] [10]

MMC/SD/SDIO Register Manual
- MMC/SD/SDIO Registers Mapping Summary: [11] [12] [13]

Table 22-54. MMCHS_RSP32

Address Offset	0x114		
Physical Address	0x4809 C114	**Instance**	MMCHS1
	0x480A D114		MMCHS3
	0x480B 4114		MMCHS2
Description	Command response[63:32] Register This 32-bit register holds bits positions [63:32] of command response type R2		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSP3																RSP2															

Bits	Field Name	Description	Type	Reset
31:16	RSP3	R2: Command Response [63:48]	R	0x0000
15:0	RSP2	R2: Command Response [47:32]	R	0x0000

Table 22-55. Register Call Summary for Register MMCHS_RSP32

MMC/SD/SDIO Functional Description • Different Types of Responses: [0]
MMC/SD/SDIO Use Cases and Tips • Programming Flow: [1] [2] [3]
MMC/SD/SDIO Register Manual • MMC/SD/SDIO Registers Mapping Summary: [4]

Table 22-56. MMCHS_RSP54

Address Offset	0x118		
Physical Address	0x4809 C118	**Instance**	MMCHS1
	0x480A D118		MMCHS3
	0x480B 4118		MMCHS2
Description	Command response[95:64] Register This 32-bit register holds bits positions [95:64] of command response type R2		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSP5																RSP4															

Bits	Field Name	Description	Type	Reset
31:16	RSP5	R2: Command Response [95:80]	R	0x0000
15:0	RSP4	R2: Command Response [79:64]	R	0x0000

Table 22-57. Register Call Summary for Register MMCHS_RSP54

MMC/SD/SDIO Functional Description • Different Types of Responses: [0]
MMC/SD/SDIO Use Cases and Tips • Programming Flow: [1] [2] [3]
MMC/SD/SDIO Register Manual • MMC/SD/SDIO Registers Mapping Summary: [4]

Table 22-58. MMCHS_RSP76

Address Offset	0x11C		
Physical Address	0x4809 C11C	**Instance**	MMCHS1
	0x480A D11C		MMCHS3
	0x480B 411C		MMCHS2
Description	Command response[127:96] Register This 32-bit register holds bits positions [127:96] of command response type R2		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSP7																RSP6															

Bits	Field Name	Description	Type	Reset
31:16	RSP7	R1b (Auto CMD12 response): Command Response [39:24] R2: Command Response [127:112]	R	0x0000
15:0	RSP6	R1b (Auto CMD12 response): Command Response [23:8] R2: Command Response [111:96]	R	0x0000

Table 22-59. Register Call Summary for Register MMCHS_RSP76

MMC/SD/SDIO Functional Description
- Different Types of Responses: [0] [1] [2]

MMC/SD/SDIO Use Cases and Tips
- Programming Flow: [3] [4] [5]

MMC/SD/SDIO Register Manual
- MMC/SD/SDIO Registers Mapping Summary: [6] [7] [8]

Table 22-60. MMCHS_DATA

Address Offset	0x120		
Physical Address	0x4809 C120	**Instance**	MMCHS1
	0x480A D120		MMCHS3
	0x480B 4120		MMCHS2
Description	Data Register This register is the 32-bit entry point of the buffer for read or write data transfers. The buffer size is 32bits x256(1024 bytes). Bytes within a word are stored and read in little endian format. This buffer can be used as two 512 byte buffers to transfer data efficiently without reducing the throughput. Sequential and contiguous access is necessary to increment the pointer correctly. Random or skipped access is not allowed. In little endian, if the local host accesses this register byte-wise or 16bit-wise, the least significant byte (bits [7:0]) must always be written/read first. The update of the buffer address is done on the most significant byte write for full 32-bit DATA register or on the most significant byte of the last word of block transfer. Example 1: Byte or 16-bit access Mbyteen[3:0]=0001 (1-byte) => Mbyteen[3:0]=0010 (1-byte) => Mbyteen[3:0]=1100 (2-bytes) OK Mbyteen[3:0]=0001 (1-byte) => Mbyteen[3:0]=0010 (1-byte) => Mbyteen[3:0]=0100 (1-byte) OK Mbyteen[3:0]=0001 (1-byte) => Mbyteen[3:0]=0010 (1-byte) => Mbyteen[3:0]=1000 (1-byte) Bad		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DATA																															

Bits	Field Name	Description	Type	Reset
31:0	DATA	Data Register [31:0] In functional mode (MMCI.MMCHS_CON[4] MODE bit set to the default value 0): A read access to this register is allowed only when the buffer read enable status is set to 1 (MMCi.MMCHS_PSTATE[11] BRE bit), otherwise a bad access (MMCi.MMCHS_STAT[29] BADA bit) is signaled. A write access to this register is allowed only when the buffer write enable status is set to 1 (MMCi.MMCHS_PSTATE[10] BWE bit), otherwise a bad access (MMCi.MMCHS_STAT[29] BADA bit) is signaled and the data is not written.	RW	0x00000000

Table 22-61. Register Call Summary for Register MMCHS_DATA

MMC/SD/SDIO Functional Description

- Data Buffer: [0] [1] [2] [3] [4] [5] [6] [7]
- Transfer or Command Status and Errors Reporting: [8]

MMC/SD/SDIO Register Manual

- MMC/SD/SDIO Registers Mapping Summary: [9] [10] [11] [12] [13]

Table 22-62. MMCHS_PSTATE

Address Offset	0x124		
Physical Address	0x4809 C124	**Instance**	MMCHS1
	0x480A D124		MMCHS3
	0x480B 4124		MMCHS2
Description	Present state register. The Host can get status of the Host Controller from this 32-bit read only register.		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							CLEV	DLEV				Reserved		Reserved		Reserved				BRE	BWE	RTA	WTA	Reserved					DLA	DATI	CMDI

Bits	Field Name	Description	Type	Reset
31:25	Reserved	Reserved bit field. Do not write any value.	R	0x00
24	CLEV	mmci_cmd line signal level. This status is used to check the mmci_cmd line level to recover from errors, and for debugging. The value of this register after reset depends on the mmci_cmd line level at that time. Read 0x0: The mmci_cmd line level is 0. Read 0x1: The mmci_cmd line level is 1.	R	-
23:20	DLEV	mmci_dat[3:0] line signal level mmci_dat[3] => bit 23 mmci_dat[2] => bit 22 mmci_dat[1] => bit 21 mmci_dat[0] => bit 20 This status is used to check mmci_dat line level to recover from errors, and for debugging. This is especially useful in detecting the busy signal level from mmci_dat[0]. The value of these registers after reset depends on the mmci_dat lines level at that time.	R	0x-
19	Reserved	Reserved bit field. Do not write any value	R	0
18	Reserved	Reserved bit field. Do not write any value This bit is not affected by soft reset.	R	1
17:16	Reserved	Reserved bit field. Do not write any value The value of these bits after soft reset is 0x0. These bits will be automatically set to 0x3 after debounce time. Debounce time is fixed to 256 x32kHz clock cycles.	R	00

Bits	Field Name	Description	Type	Reset
15:12	Reserved	Reserved bit field. Do not write any value	R	0x0
11	BRE	Buffer read enable. This bit is used for non-DMA read transfers. It indicates that a complete block specified by MMCi.MMCHS_BLK[10:0] BLEN bits has been written in the buffer and is ready to be read. It is set to 0 when the entire block is read from the buffer. It is set to 1 when a block data is ready in the buffer and generates the Buffer read ready status of interrupt (MMCi.MMCHS_STAT[5] BRR bit). Read 0x0: Read BLEN bytes disable Read 0x1: Read BLEN bytes enable. Readable data exists in the buffer.	R	0
10	BWE	Buffer Write enable. This status is used for non-DMA write transfers. It indicates if space is available for write data. Read 0x0: There is no room left in the buffer to write BLEN bytes of data. Read 0x1: There is enough space in the buffer to write BLEN bytes of data.	R	0
9	RTA	Read transfer active. This status is used for detecting completion of a read transfer. It is set to 1 after the end bit of read command or by activating a continue request (MMCi.MMCHS_HCTL[17] CR bit) following a stop at block gap request. This bit is set to 0 when all data have been read by the local host after last block or after a stop at block gap request. Read 0x0: No valid data on the mmci_dat lines. Read 0x1: Read data transfer on going.	R	0
8	WTA	Write transfer active. This status indicates a write transfer active. It is set to 1 after the end bit of write command or by activating a continue request (MMCi.MMCHS_HCTL[17] CR bit) following a stop at block gap request. This bit is set to 0 when CRC status has been received after last block or after a stop at block gap request. Read 0x0: No valid data on the mmci_dat lines. Read 0x1: Write data transfer on going.	R	0
7:3	Reserved	Reserved bit field. Do not write any value	R	0x00
2	DLA	mmci_dat line active. This status bit indicates whether one of the mmci_dat line is in use. In the case of read transactions (card to host): This bit is set to 1 after the end bit of read command or by activating continue request MMCi.MMCHS_HCTL[17] CR bit. This bit is set to 0 when the host controller received the end bit of the last data block or at the beginning of the read wait mode. In the case of write transactions (host to card): This bit is set to 1 after the end bit of write command or by activating continue request MMCi.MMCHS_HCTL[17] CR bit. This bit is set to 0 on the end of busy event for the last block; host controller must wait 8 clock cycles with line not busy to really consider not "busy state" or after the busy block as a result of a stop at gap request. Read 0x0: mmci_dat Line inactive Read 0x1: mmci_dat Line active	R	0
1	DATI	Command inhibit (mmci_dat). This status bit is generated if either mmci_dat line is active (MMCi.MMCHS_PSTATE[2] DLA bit) or Read transfer is active (MMCi.MMCHS_PSTATE[9] RTA bit) or when a command with busy is issued. This bit prevents the local host to issue a command. A change of this bit from 1 to 0 generates a transfer complete interrupt (MMCi.MMCHS_STAT[1] TC bit). Read 0x0: Issuing of command using the mmci_dat lines is allowed Read 0x1: Issuing of command using mmci_dat lines is not allowed	R	0

Bits	Field Name	Description	Type	Reset
0	CMDI	Command inhibit(mmci_cmd). This status bit indicates that the mmci_cmd line is in use. This bit is set to 0 when the most significant byte is written into the command register. This bit is not set when Auto CMD12 is transmitted. This bit is set to 0 in either the following cases: - After the end bit of the command response, excepted if there is a command conflict error (MMCi.MMCHS_STAT[17] CCRC bit or MMCi.MMCHS_STAT[18] CEB bit set to 1) or a Auto CMD12 is not executed (MMCi.MMCHS_AC12[0] ACNE bit). - After the end bit of the command without response (MMCi.MMCHS_CMD[17:16] RSP_TYPE bits set to "00"). In case of a command data error is detected (MMCi.MMCHS_STAT[19] CTO bit set to 1), this register is not automatically cleared. Read 0x0: Issuing of command using mmci_cmd line is allowed Read 0x1: Issuing of command using mmci_cmd line is not allowed	R	0

Table 22-63. Register Call Summary for Register MMCHS_PSTATE

MMC/SD/SDIO Integration • Resets: [0]
MMC/SD/SDIO Functional Description • Data Buffer: [1] [2] [3] [4]
MMC/SD/SDIO Register Manual • MMC/SD/SDIO Registers Mapping Summary: [5] [6] [7] [8] [9] [10] [11] [12] [13] [14] [15] [16] [17] [18] [19]

Table 22-64. MMCHS_HCTL

Address Offset	0x128		
Physical Address	0x4809 C128	**Instance**	MMCHS1
	0x480A D128		MMCHS3
	0x480B 4128		MMCHS2
Description	Control register. This register defines the host controls to set power, wake up and transfer parameters. MMCi.MMCHS_HCTL[31:24] = Wake-up control MMCi.MMCHS_HCTL[23:16] = Block gap control MMCi.MMCHS_HCTL[15:8] = Power control MMCi.MMCHS_HCTL[7:0] = Host control		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved				OBWE	REM	INS	IWE	Reserved				IBG	RWC	CR	SBGR	Reserved				SDVS			SDBP	Reserved						DTW	Reserved

Bits	Field Name	Description	Type	Reset
31:28	Reserved	Reserved bit field. Do not write any value.	R	0x00
27	OBWE	Wake-up event enable for 'Out-of-Band' Interrupt. This bit enables wake-up events for 'Out-of-Band' assertion. Wakeup is generated if the wake-up feature is enabled (MMCi.MMCHS_SYSCONFIG[2] ENAWAKEUP bit). The write to this register is ignored when MMCi.MMCHS_CON[14] OBIE bit is not set. 0x0: Disable wakeup on 'Out-of-Band' Interrupt. 0x1: Enable wakeup on 'Out-of-Band' Interrupt.	RW	0
26	REM	Wake-up event enable on SD card removal. This bit enables wake-up events for card removal assertion. Wakeup is generated if the wake-up feature is enabled (MMCi.MMCHS_SYSCONFIG[2] ENAWAKEUP bit). 0x0: Disable wakeup on card removal. 0x1: Enable wakeup on card removal.	RW	0

Bits	Field Name	Description	Type	Reset
25	INS	Wake-up event enable on SD card insertion This bit enables wake-up events for card insertion assertion. Wakeup is generated if the wake-up feature is enabled (MMCi.MMCHS_SYSCONFIG[2] ENAWAKEUP bit). 0x0: Disable wakeup on card insertion. 0x1: Enable wakeup on card insertion.	RW	0
24	IWE	Wake-up event enable on SD card interrupt. This bit enables wake-up events for card interrupt assertion. Wakeup is generated if the wake-up feature is enabled (MMCi.MMCHS_SYSCONFIG[2] ENAWAKEUP bit) and enable status bit is set (MMCi.MMCHS_IE[8] CIRQ_ENABLE bit). 0x0: Disable wakeup on card interrupt 0x1: Enable wakeup on card interrupt	RW	0
23:20	Reserved	Reserved bit field. Do not write any value	R	0x0
19	IBG	Interrupt block at gap. This bit is valid only in 4-bit mode of SDIO card to enable interrupt detection in the interrupt cycle at block gap for a multiple block transfer. For MMC cards and for SD card this bit should be set to 0. 0x0: Disable interrupt detection at the block gap in 4-bit mode 0x1: Enable interrupt detection at the block gap in 4-bit mode	RW	0
18	RWC	Read wait control. The read wait function is optional only for SDIO cards. If the card supports read wait, this bit must be enabled, then requesting a stop at block gap (MMCi.MMCHS_HCTL[16] SBGR bit) generates a read wait period after the current end of block. Be careful, if read wait is not supported it may cause a conflict on mmci_dat line. 0x0: Disable Read Wait Control. Suspend/Resume cannot be supported. 0x1: Enable Read Wait Control	RW	0
17	CR	Continue request. This bit is used to restart a transaction that was stopped by requesting a stop at block gap (MMCi.MMCHS_HCTL[16] SBGR bit). Set this bit to 1 restarts the transfer. The bit is automatically set to 0 by the host controller when transfer has restarted i.e. mmci_dat line is active (MMCi.MMCHS_PSTATE[2] DLA bit) or transferring data (MMCi.MMCHS_PSTATE[8] WTA bit). The Stop at block gap request must be disabled (MMCi.MMCHS_HCTL[16] SBGR bit =0) before setting this bit. 0x0: No affect 0x1: transfer restart	RW	0
16	SBGR	Stop at block gap request. This bit is used to stop executing a transaction at the next block gap. The transfer can restart with a continue request (MMCi.MMCHS_HCTL[17] CR bit) or during a suspend/resume sequence. In case of read transfer, the card must support read wait control. In case of write transfer, the host driver shall set this bit after all block data written. Until the transfer completion (MMCi.MMCHS_STAT[1] TC bit set to 1), the host driver shall leave this bit set to 1.If this bit is set, the local host shall not write to the data register (MMCi.MMCHS_DATA). 0x0: Transfer mode 0x1: Stop at block gap	RW	0
15:12	Reserved	Reserved bit field. Do not write any value.	R	0x0
11:9	SDVS	SD bus voltage select (All cards). The host driver should set these bits to select the voltage level for the card according to the voltage supported by the system (MMCi.MMCHS_CAPA[26] VS18 bit, MMCi.MMCHS_CAPA[25] VS30 bit, MMCi.MMCHS_CAPA[24] VS33 bit) before starting a transfer. 0x5: 1.8V (Typical) 0x6: 3.0V (Typical) 0x7: 3.3V (Typical)	RW	0x0

Bits	Field Name	Description	Type	Reset
		MMCHS2: This field must be set to 0x5. MMCHS3: This field must be set to 0x5.		
8	SDBP	SD bus power. Before setting this bit, the host driver shall select the SD bus voltage (MMCi.MMCHS_HCTL[11:9] SDVS bits). If the host controller detects the No card state, this bit is automatically set to 0. If the module is power off, a write in the command register (MMCi.MMCHS_CMD) will not start the transfer. A write to this bit has no effect if the selected SD bus voltage is not supported according to capability register (MMCi.MMCHS_CAPA[VS*]). 0x0: Power off 0x1: Power on	RW	0
7:2	Reserved	Reserved bit field. Do not write any value.	R	0x00
1	DTW	Data transfer width. For MMC card, this bit must be set following a valid SWITCH command (CMD6) with the correct value and extend CSD index written in the argument. Prior to this command, the MMC card configuration register (CSD and EXT_CSD) must be verified for compliance with *MMC standard specification 4.x* This register has no effect when the MMC 8-bit mode is selected (MMCi.MMCHS_CON[5] DW8 bit set to 1) For SD/SDIO cards, this bit must be set following a valid SET_BUS_WIDTH command (ACMD6) with the value written in bit 1 of the argument. Prior to this command, the SD card configuration register (SCR) must be verified for the supported bus width by the SD card. 0x0: 1-bit Data width (mmci_dat[0] used) 0x1: 4-bit Data width (mmci_dat[3:0] used)	RW	0
0	Reserved	Reserved bit field. Do not write any value.	R	0

Table 22-65. Register Call Summary for Register MMCHS_HCTL

MMC/SD/SDIO Integration • Clocks: [0] [1]
MMC/SD/SDIO Functional Description • Transfer Stop: [2] [3]
MMC/SD/SDIO Use Cases and Tips • Programming Flow: [4] [5] [6] [7] [8] [9] [10] [11]
MMC/SD/SDIO Register Manual • MMC/SD/SDIO Registers Mapping Summary: [12] [13] [14] [15] [16] [17] [18] [19] [20] [21] [22] [23] [24] [25] [26] [27] [28]

Table 22-66. MMCHS_SYSCTL

Address Offset	0x12C		
Physical Address	0x4809 C12C	**Instance**	MMCHS1
	0x480A D12C		MMCHS3
	0x480B 412C		MMCHS2
Description	SD system control register. This register defines the system controls to set software resets, clock frequency management and data timeout. MMCHS_SYSCTL[31:24] = Software resets MMCHS_SYSCTL[23:16] = Timeout control MMCHS_SYSCTL[15:0] = Clock control		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved					SRD	SRC	SRA	Reserved				DTO				CLKD								Reserved					CEN	ICS	ICE

Bits	Field Name	Description	Type	Reset
31:27	Reserved	Reserved bit field. Do not write any value.	R	0x00
26	SRD	Software reset for mmci_dat line. This bit is set to 1 for reset and released to 0 when completed .mmci_dat finite state machine in both clock domain are also reset. Here below are the registers cleared by the MMCHS_SYSCTL[26] SRD bit: MMCi.MMCHS_DATA MMCi.MMCHS_PSTATE: BRE, BWE, RTA, WTA, DLA and DATI MMCi.MMCHS_HCTL: SBGR and CR MMCi.MMCHS_STAT: BRR, BWR, BGE and TC Interconnect and MMC buffer data management is reinitialized. 0x0: Reset completed 0x1: Software reset for mmci_dat line	RW	0
25	SRC	Software reset for mmci_cmd line. This bit is set to 1 for reset and released to 0 when completed. mmci_cmd finite state machine in both clock domain are also reset. Here below the registers cleared by the MMCi.MMCHS_SYSCTL[25] SRC bit: MMCi.MMCHS_PSTATE: CMDI MMCi.MMCHS_STAT: CC Interconnect and MMC command status management is reinitialized. 0x0: Reset completed 0x1: Software reset for mmci_cmd line	RW	0
24	SRA	Software reset for all. This bit is set to 1 for reset , and released to 0 when completed. This reset affects the entire host controller except for the card detection circuit and capabilities registers. 0x0: Reset completed 0x1: Software reset for all the design	RW	0
23:20	Reserved	Reserved bit field. Do not write any value.	R	0x0
19:16	DTO	Data timeout counter value and busy timeout. This value determines the interval by which mmci_dat lines timeouts are detected. The host driver needs to set this bit field based on - the maximum read access time (NAC) (Refer to the SD Specification Part1 Physical Layer), - the data read access time values (TAAC and NSAC) in the card specific data register (CSD) of the card, - the timeout clock base frequency (MMCi.MMCHS_CAPA[5:0] TCF bits). If the card does not respond within the specified number of cycles, a data timeout error occurs (MMCi.MMCHS_STAT[20] DTO bit). The MMCi.MMCHS_SYSCTL[19,16] DTO bit field is also used to check busy duration, to generate busy timeout for commands with busy response or for busy programming during a write command. Timeout on CRC status is generated if no CRC token is present after a block write. 0x0: TCF x 2^13 0x1: TCF x 2^14 0xE: TCF x 2^27 0xF: Reserved	RW	0x0
15:6	CLKD	Clock frequency select These bits define the ratio between a reference clock frequency (system dependant) and the output clock frequency on the mmci_clk pin of either the memory card (MMC, SD or SDIO). 0x0: Clock Ref bypass 0x1: Clock Ref bypass 0x2: Clock Ref / 2 0x3: Clock Ref / 3 0x3FF: Clock Ref / 1023	RW	0x000
5:3	Reserved	Reserved bit field. Do not write any value	R	0x0
2	CEN	Clock enable. This bit controls if the clock is provided to the card or not. 0x0: The clock is not provided to the card . Clock frequency can be changed .	RW	0

Bits	Field Name	Description	Type	Reset
		0x1: The clock is provided to the card and can be automatically gated when MMCi.MMCHS_SYSCONFIG[0] AUTOIDLE bit is set to 1 (default value) . The host driver shall wait to set this bit to 1 until the Internal clock is stable (MMCi.MMCHS_SYSCTL[1] ICS bit).		
1	ICS	Internal clock stable (status)This bit indicates either the internal clock is stable or not. Read 0x0: The internal clock is not stable. Read 0x1: The internal clock is stable after enabling the clock (MMCi.MMCHS_SYSCTL[0] ICE bit) or after changing the clock ratio (MMCi.MMCHS_SYSCTL[15:6] CLKD bits).	R	0
0	ICE	Internal clock enable. This register controls the internal clock activity. In very low power state, the internal clock is stopped. Note: The activity of the debounce clock (used for wake-up events) and the interface clock (used for reads and writes to the module register map) are not affected by this register. 0x0: The internal clock is stopped (very low power state). 0x1: The internal clock oscillates and can be automatically gated when MMCi.MMCHS_SYSCONFIG[0] AUTOIDLE bit is set to 1 (default value) .	RW	0

Table 22-67. Register Call Summary for Register MMCHS_SYSCTL

MMC/SD/SDIO Environment • MMC/SD/SDIO Protocol and Data Format: [0]
MMC/SD/SDIO Integration • Resets: [1] [2] [3] [4] [5]
MMC/SD/SDIO Use Cases and Tips • Programming Flow: [6] [7] [8] [9] [10] [11] [12] [13] [14]
MMC/SD/SDIO Register Manual • MMC/SD/SDIO Registers Mapping Summary: [15] [16] [17] [18] [19] [20] [21] [22] [23] [24] [25] [26] [27] [28] [29]

Table 22-68. MMCHS_STAT

Address Offset	0x130		
Physical Address	0x4809 C130	**Instance**	MMCHS1
	0x480A D130		MMCHS3
	0x480B 4130		MMCHS2
Description	Interrupt status register The interrupt status regroups all the status of the module internal events that can generate an interrupt. MMCHS_STAT[31:16] = Error Interrupt Status MMCHS_STAT[15:0] = Normal Interrupt Status		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved		BADA	CERR	Reserved			ACE	Reserved	DEB	DCRC	DTO	CIE	CEB	CCRC	CTO	ERRI	Reserved					OBI	CIRQ	Reserved		BRR	BWR	Reserved	BGE	TC	CC

Bits	Field Name	Description	Type	Reset
31:30	Reserved	Reserved bit field. Do not write any value	R	0x0
29	BADA	Bad access to data space. This bit is set automatically to indicate a bad access to buffer when not allowed: - During a read access to the data register (MMCi.MMCHS_DATA) while buffer reads are not allowed (MMCi.MMCHS_PSTATE[11] BRE bit =0) - During a write access to the data register (MMCi.MMCHS_DATA) while buffer writes are not allowed (MMCi.MMCHS_PSTATE[10] BWE bit=0) Read 0x0: No Interrupt. Write 0x0: Status bit unchanged Read 0x1: Bad Access Write 0x1: Status is cleared	RW	0
28	CERR	Card error. This bit is set automatically when there is at least one error in a response of type R1, R1b, R6, R5 or R5b. Only bits referenced as type E (error) in status field in the response can set a card status error. An error bit in the response is flagged only if corresponding bit in card status response error MMCi.MMCHS_CSRE in set. There is no card error detection for autoCMD12 command. The host driver shall read MMCi.MMCHS_RSP76 register to detect error bits in the command response. Read 0x0: No Error Write 0x0: Status bit unchanged Read 0x1: Card error Write 0x1: Status is cleared	RW	0
27:25	Reserved	Reserved bit field. Do not write any value	R	0x0
24	ACE	Auto CMD12 error. This bit is set automatically when one of the bits in Auto CMD12 Error status register has changed from 0 to 1. Read 0x0: No Error Write 0x0: Status bit unchanged Read 0x1: AutoCMD12 error Write 0x1: Status is cleared	RW	0
23	Reserved	Reserved bit field. Do not write any value	R	0
22	DEB	Data End Bit error. This bit is set automatically when detecting a 0 at the end bit position of read data on mmci_dat line or at the end position of the CRC status in write mode. Read 0x0: No Error Write 0x0: Status bit unchanged Read 0x1: Data end bit error Write 0x1: Status is cleared	RW	0
21	DCRC	Data CRC Error. This bit is set automatically when there is a CRC16 error in the data phase response following a block read command or if there is a 3-bit CRC status different of a position "010" token during a block write command. Read 0x0: No Error. Write 0x0: Status bit unchanged Read 0x1: Data CRC error Write 0x1: Status is cleared	RW	0
20	DTO	Data timeout error. This bit is set automatically according to the following conditions: - Busy timeout for R1b, R5b response type. - Busy timeout after write CRC status. - Write CRC status timeout. - Read data timeout. Read 0x0: No error Write 0x0: Status bit unchanged	RW	0

Bits	Field Name	Description	Type	Reset
		Read 0x1: Time out Write 0x1: Status is cleared		
19	CIE	Command index error. This bit is set automatically when response index differs from corresponding command index previously emitted. It depends on the enable bit (MMCi.MMCHS_CMD[20] CICE). Read 0x0: No error Write 0x0: Status bit unchanged Read 0x1: Command index error Write 0x1: Status is cleared	RW	0
18	CEB	Command end bit error. This bit is set automatically when detecting a 0 at the end bit position of a command response. Read 0x0: No error Write 0x0: Status bit unchanged Read 0x1: Command end bit error Write 0x1: Status is cleared	RW	0
17	CCRC	Command CRC Error. This bit is set automatically when there is a CRC7 error in the command response depending on the enable bit (MMCi.MMCHS_CMD[19] CCCE). Read 0x0: No Error Write 0x0: Status bit unchanged Read 0x1: Command CRC error Write 0x1: Status is cleared	RW	0
16	CTO	Command Timeout Error. This bit is set automatically when no response is received within 64 clock cycles from the end bit of the command. For commands that reply within 5 clock cycles - the timeout is still detected at 64 clock cycles. Read 0x0: No error Write 0x0: Status bit unchanged Read 0x1: Time Out Write 0x1: Status is cleared	RW	0
15	ERRI	Error Interrupt. If any of the bits in the Error Interrupt Status register (MMCi.MMCHS_STAT[31:16]) are set, then this bit is set to 1. Therefore the host driver can efficiently test for an error by checking this bit first. Writes to this bit are ignored. Read 0x0: No Interrupt Read 0x1: Error interrupt event(s) occurred	R	0
14:10	Reserved	Reserved bit field. Do not write any value.	R	0x00
9	OBI	Out-Of-Band interrupt (This interrupt is only useful for MMC card). This bit is set automatically when MMCi.MMCHS_CON[14] OBIE bit is set and an Out-of-Band interrupt occurs on OBI pin. The interrupt detection depends on polarity controlled by MMCi.MMCHS_CON[13] OBIP bit. The Out-of-Band interrupt signal is a system specific feature for future use, this signal is not required for existing specification implementation. Read 0x0: No Out-Of-Band interrupt. Write 0x0: Status bit unchanged. Read 0x1: Interrupt Out-Of-Band occurs. Write 0x1: Status is cleared.	R	0

Bits	Field Name	Description	Type	Reset
8	CIRQ	Card interrupt. This bit is only used for SD and SDIO cards. In 1-bit mode, interrupt source is asynchronous (can be a source of asynchronous wakeup). In 4-bit mode, interrupt source is sampled during the interrupt cycle. In CE-ATA mode, interrupt source is detected when the card drives mmci_cmd line to zero during one cycle after data transmission end. All modes above are fully exclusive. The controller interrupt must be clear by setting MMCi.MMCHS_IE[8] CIRQ_ENABLE to 0, then the host driver must start the interrupt service with card (clearing card interrupt status) to remove card interrupt source. Otherwise the Controller interrupt will be reasserted as soon as MMCi.MMCHS_IE[8] CIRQ_ENABLE is set to 1. Writes to this bit are ignored. Read 0x0: No card interrupt Read 0x1: Generate card interrupt	R	0
7:6	Reserved	Reserved bit field. Do not write any value.	RW	00
5	BRR	Buffer read ready. This bit is set automatically during a read operation to the card (see class 2 - block oriented read commands) when one block specified by the MMCi.MMCHS_BLK[10:0] BLEN bit field is completely written in the buffer. It indicates that the memory card has filled out the buffer and that the local host needs to empty the buffer by reading it. Note: If the DMA receive-mode is enabled, this bit is never set; instead a DMA receive request to the main DMA controller of the system is generated. Read 0x0: Not Ready to read buffer Write 0x0: Status bit unchanged Read 0x1: Ready to read buffer Write 0x1: Status is cleared	RW	0
4	BWR	Buffer write ready. This bit is set automatically during a write operation to the card (see class 4 - block oriented write command) when the host can write a complete block as specified by MMCi.MMCHS_BLK[10:0] BLEN. It indicates that the memory card has emptied one block from the buffer and that the local host is able to write one block of data into the buffer. Note: If the DMA transmit mode is enabled, this bit is never set; instead, a DMA transmit request to the main DMA controller of the system is generated. Read 0x0: Not Ready to write buffer Write 0x0: Status bit unchanged Read 0x1: Ready to write buffer Write 0x1: Status is cleared	RW	0
3	Reserved	Reserved bit field. Do not write any value	R	0
2	BGE	Block gap event. When a stop at block gap is requested (MMCi.MMCHS_HCTL[16] SBGR bit), this bit is automatically set when transaction is stopped at the block gap during a read or write operation. This event does not occur when the stop at block gap is requested on the last block. In read mode, a 1-to-0 transition of the mmci_dat line active status (MMCi.MMCHS_PSTATE[2] DLA bit) between data blocks generates a Block gap event interrupt. Read 0x0: No block gap event Write 0x0: Status bit unchanged Read 0x1: Transaction stopped at block gap Write 0x1: Status is cleared	RW	0

Bits	Field Name	Description	Type	Reset
1	TC	Transfer completed. This bit is always set when a read/write transfer is completed or between two blocks when the transfer is stopped due to a stop at block gap request (MMCi.MMCHS_HCTL[16] SBGR bit). This bit is also set when exiting a command in a busy state (if the command has a busy notification capability). In Read mode: This bit is automatically set on completion of a read transfer (MMCi.MMCHS_PSTATE[9] RTA bit). In write mode: This bit is set automatically on completion of the mmci_dat line use (MMCi.MMCHS_PSTATE[2] DLA bit). Read 0x0: No transfer complete Write 0x0: Status bit unchanged Read 0x1: Data transfer complete Write 0x1: Status is cleared	RW	0
0	CC	Command complete. This bit is set when a 1-to-0 transition occurs in the register command inhibit (MMCi.MMCHS_PSTATE[0] CMDI bit) If the command is a type for which no response is expected, then the command complete interrupt is generated at the end of the command. A command timeout error (MMCi.MMCHS_STAT[16] CTO bit) has higher priority than command complete (MMCi.MMCHS_STAT[0] CC bit). If a response is expected but none is received, then a command timeout error is detected and signaled instead of the command complete interrupt. Read 0x0: No Command complete Write 0x0: Status bit unchanged Read 0x1: Command complete Write 0x1: Status is cleared	RW	0

Table 22-69. Register Call Summary for Register MMCHS_STAT

MMC/SD/SDIO Integration
- Clocks: [0] [1] [2] [3]
- Interrupt Requests: [4] [5] [6] [7] [8] [9] [10] [11] [12] [13] [14] [15] [16] [17] [18]

MMC/SD/SDIO Functional Description
- Data Buffer: [19] [20] [21] [22] [23] [24]
- Transfer or Command Status and Errors Reporting: [25]
- MMC CE-ATA Command Completion Disable Management: [26]

MMC/SD/SDIO Use Cases and Tips
- Programming Flow: [27]

MMC/SD/SDIO Register Manual
- MMC/SD/SDIO Registers Mapping Summary: [28] [29] [30] [31] [32] [33] [34] [35] [36] [37] [38] [39] [40] [41] [42] [43] [44] [45] [46] [47] [48] [49] [50] [51] [52] [53] [54] [55] [56]

Table 22-70. MMCHS_IE

Address Offset	0x134		
Physical Address	0x4809 C134	**Instance**	MMCHS1
	0x480A D134		MMCHS3
	0x480B 4134		MMCHS2
Description	Interrupt SD enable register This register allows to enable/disable the module to set status bits, on an event-by-event basis. MMCHS_IE[31:16] = Error Interrupt Status Enable MMCHS_IE[15:0] = Normal Interrupt Status Enable		
Type	RW		

31 30	29	28	27 26 25	24	23	22	21	20	19	18	17	16	15 14 13 12 11 10	9	8	7 6	5	4	3	2	1	0
Reserved	BADA_ENABLE	CERR_ENABLE	Reserved	ACE_ENABLE	Reserved	DEB_ENABLE	DCRC_ENABLE	DTO_ENABLE	CIE_ENABLE	CEB_ENABLE	CCRC_ENABLE	CTO_ENABLE	NULL	Reserved	OBI_ENABLE	CIRQ_ENABLE	Reserved	BRR_ENABLE	BWR_ENABLE	Reserved	BGE_ENABLE	TC_ENABLE	CC_ENABLE

Bits	Field Name	Description	Type	Reset
31:30	Reserved	Reserved bit field. Do not write any value.	R	0
29	BADA_ENABLE	Bad access to data space Interrupt Enable	RW	0
		0x0: Masked		
		0x1: Enabled		
28	CERR_ENABLE	Card error interrupt Enable	RW	0
		0x0: Masked		
		0x1: Enabled		
27:25	Reserved	Reserved bit field. Do not write any value	R	0x0
24	ACE_ENABLE	Auto CMD12 error Interrupt Enable	RW	0
		0x0: Masked		
		0x1: Enabled		
23	Reserved	Reserved bit field. Do not write any value	R	0
22	DEB_ENABLE	Data end bit error Interrupt Enable	RW	0
		0x0: Masked		
		0x1: Enabled		
21	DCRC_ENABLE	Data CRC error Interrupt Enable	RW	0
		0x0: Masked		
		0x1: Enabled		
20	DTO_ENABLE	Data timeout error Interrupt Enable	RW	0
		0x0: The data timeout detection is deactivated. The host controller provides the clock to the card until the card sends the data or the transfer is aborted.		
		0x1: The data timeout detection is enabled.		
19	CIE_ENABLE	Command index error Interrupt Enable	RW	0
		0x0: Masked		
		0x1: Enabled		
18	CEB_ENABLE	Command end bit error Interrupt Enable	RW	0
		0x0: Masked		
		0x1: Enabled		
17	CCRC_ENABLE	Command CRC error Interrupt Enable	RW	0
		0x0: Masked		
		0x1: Enabled		
16	CTO_ENABLE	Command timeout error Interrupt Enable	RW	0

Bits	Field Name	Description	Type	Reset
		0x0: Masked 0x1: Enabled		
15	NULL	Fixed to 0 The host driver shall control error interrupts using the Error Interrupt Signal Enable register. Writes to this bit are ignored.	R	0
14:10	Reserved	Reserved bit field. Do not write any value.	R	0x00
9	OBI_ENABLE	Out-of-Band interrupt Enable A write to this register when MMCi.MMCHS_CON[14] OBIE is set to '0' is ignored. 0x0: Masked 0x1: Enabled	RW	0
8	CIRQ_ENABLE	Card interrupt Enable A clear of this bit also clears the corresponding status bit. During 1-bit mode, if the interrupt routine does not remove the source of a card interrupt in the SDIO card, the status bit is reasserted when this bit is set to 1. This bit must be set to 1 when entering in smart idle mode to enable system to identity wakeup event and to allow controller to clear internal wakeup source. 0x0: Masked 0x1: Enabled	RW	0
7:6	Reserved	Reserved bit field. Do not write any value	RW	00
5	BRR_ENABLE	Buffer Read Ready Interrupt Enable 0x0: Masked 0x1: Enabled	RW	0
4	BWR_ENABLE	Buffer Write Ready Interrupt Enable 0x0: Masked 0x1: Enabled	RW	0
3	Reserved	Reserved bit field. Do not write any value.	R	0
2	BGE_ENABLE	Block Gap Event Interrupt Enable 0x0: Masked 0x1: Enabled	RW	0
1	TC_ENABLE	Transfer completed Interrupt Enable 0x0: Masked 0x1: Enabled	RW	0
0	CC_ENABLE	Command completed Interrupt Enable 0x0: Masked 0x1: Enabled	RW	0

Table 22-71. Register Call Summary for Register MMCHS_IE

MMC/SD/SDIO Integration
- Clocks: [0] [1]
- Interrupt Requests: [2] [3] [4] [5]

MMC/SD/SDIO Use Cases and Tips
- Programming Flow: [6] [7] [8] [9] [10] [11] [12] [13] [14] [15] [16] [17] [18] [19] [20]

MMC/SD/SDIO Register Manual
- MMC/SD/SDIO Registers Mapping Summary: [21] [22] [23] [24] [25] [26]

Table 22-72. MMCHS_ISE

Address Offset	0x138		
Physical Address	0x4809 C138	**Instance**	MMCHS1
	0x480A D138		MMCHS3
	0x480B 4138		MMCHS2
Description	Interrupt signal enable register This register allows to enable/disable the module internal sources of status, on an event-by-event basis. MMCHS_ISE[31:16] = Error Interrupt Signal Enable MMCHS_ISE[15:0] = Normal Interrupt Signal Enable		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved		BADA_SIGEN	CERR_SIGEN	Reserved			ACE_SIGEN	Reserved	DEB_SIGEN	DCRC_SIGEN	DTO_SIGEN	CIE_SIGEN	CEB_SIGEN	CCRC_SIGEN	CTO_SIGEN	NULL	Reserved						OBI_SIGEN	CIRQ_SIGEN	Reserved	Reserved	BRR_SIGEN	BWR_SIGEN	Reserved	BGE_SIGEN	TC_SIGEN	CC_SIGEN

Bits	Field Name	Description	Type	Reset
31:30	Reserved	Reserved bit field. Do not write any value.	R	0
29	BADA_SIGEN	Bad access to data space signal status Enable 0x0: Masked 0x1: Enabled	RW	0
28	CERR_SIGEN	Card error interrupt signal status Enable 0x0: Masked 0x1: Enabled	RW	0
27:25	Reserved	Reserved bit field. Do not write any value	R	0x0
24	ACE_SIGEN	Auto CMD12 error signal status Enable 0x0: Masked 0x1: Enabled	RW	0
23	Reserved	Reserved bit field. Do not write any value	R	0
22	DEB_SIGEN	Data end bit error signal status Enable 0x0: Masked 0x1: Enabled	RW	0
21	DCRC_SIGEN	Data CRC error signal status Enable 0x0: Masked 0x1: Enabled	RW	0
20	DTO_SIGEN	Data timeout error signal status Enable 0x0: Masked 0x1: Enabled	RW	0
19	CIE_SIGEN	Command index error signal status Enable 0x0: Masked 0x1: Enabled	RW	0
18	CEB_SIGEN	Command end bit error signal status Enable 0x0: Masked 0x1: Enabled	RW	0
17	CCRC_SIGEN	Command CRC error signal status Enable 0x0: Masked 0x1: Enabled	RW	0
16	CTO_SIGEN	Command timeout error signal status Enable 0x0: Masked 0x1: Enabled	RW	0

Bits	Field Name	Description	Type	Reset
15	NULL	Fixed to 0 The host driver shall control error interrupts using the Error Interrupt Signal Enable register. Writes to this bit are ignored	R	0
14:10	Reserved	Reserved bit field. Do not write any value	R	0x00
9	OBI_SIGEN	Out-Of-Band Interrupt signal status Enable. A write to this register when MMCHS_CON[14] OBIE bit is set to '0' is ignored. 0x0: Masked 0x1: Enabled	RW	0
8	CIRQ_SIGEN	Card interrupt signal status Enable 0x0: Masked 0x1: Enabled	RW	0
7	Reserved	Reserved bit field. Do not write any value	RW	0
6	Reserved	Reserved bit field. Do not write any value	RW	0
5	BRR_SIGEN	Buffer Read Ready signal status Enable 0x0: Masked 0x1: Enabled	RW	0
4	BWR_SIGEN	Buffer Write Ready signal status Enable 0x0: Masked 0x1: Enabled	RW	0
3	Reserved	Reserved bit field. Do not write any value	R	0
2	BGE_SIGEN	Black Gap Event signal status Enable 0x0: Masked 0x1: Enabled	RW	0
1	TC_SIGEN	Transfer completed signal status Enable 0x0: Masked 0x1: Enabled	RW	0
0	CC_SIGEN	Command completed signal status Enable 0x0: Masked 0x1: Enabled	RW	0

Table 22-73. Register Call Summary for Register MMCHS_ISE

MMC/SD/SDIO Integration
- Clocks: [0] [1]
- Interrupt Requests: [2] [3] [4] [5] [6]

MMC/SD/SDIO Use Cases and Tips
- Programming Flow: [7] [8] [9] [10] [11] [12] [13] [14] [15] [16] [17] [18] [19] [20] [21]

MMC/SD/SDIO Register Manual
- MMC/SD/SDIO Registers Mapping Summary: [22] [23] [24] [25]

Table 22-74. MMCHS_AC12

Address Offset	0x13C		
Physical Address	0x4809 C13C	**Instance**	MMCHS1
	0x480A D13C		MMCHS3
	0x480B 413C		MMCHS2
Description	Auto CMD12 Error Status Register. The host driver may determine which of the errors cases related to Auto CMD12 has occurred by checking this MMCi.MMCHS_AC12 register when an Auto CMD12 Error interrupt occurs. This register is valid only when Auto CMD12 is enabled (MMCi.MMCHS_CMD[2] ACEN bit) and Auto CMD12Error (MMCi.MMCHS_STAT[24] ACE bit) is set to 1. Note: These bits are automatically reset when starting a new adtc command with data.		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																								CNI	Reserved		ACIE	ACEB	ACCE	ACTO	ACNE

Bits	Field Name	Description	Type	Reset
31:8	Reserved	Reserved bit field. Do not write any value.	R	0x0
7	CNI	Command not issue by Auto CMD12 error. If this bit is set to 1, it means that pending command is not executed due to Auto CMD12 error: ACEB, ACCE, ACTO or ACNE. Read 0x0: Not error Read 0x1: Command not issued	R	0
6:5	Reserved	Reserved bit field. Do not write any value.	R	0x0
4	ACIE	Auto CMD12 index error. This bit is a set to 1 when response index differs from corresponding command auto CMD12 index previously emitted. This bit depends on the command index check enable (MMCi.MMCHS_CMD[20] CICE bit). Read 0x0: No error Read 0x1: Auto CMD12 Index Error	R	0
3	ACEB	Auto CMD12 end bit error. This bit is set to 1 when detecting a 0 at the end bit position of auto CMD12 command response. Read 0x0: No error Read 0x1: AutoCMD12 End bit Error	R	0
2	ACCE	Auto CMD12 CRC error. This bit is automatically set to 1 when a CRC7 error is detected in the auto CMD12 command response depending on the enable in the MMCi.MMCHS_CMD[19] CCCE bit. Read 0x0: No error Read 0x1: Auto CMD12 CRC Error	R	0
1	ACTO	Auto CMD12 timeout error. This bit is set to 1 if no response is received within 64 clock cycles from the end bit of the auto CMD12 command. Read 0x0: No error Read 0x1: Auto CMD12 Time Out	R	0
0	ACNE	Auto CMD12 not executed. This bit is set to 1 if multiple block data transfer command has started and if an error occurs in command before Auto CMD12 starts. Read 0x0: Auto CMD12 Executed Read 0x1: Auto CMD12 Not Executed	R	0

Table 22-75. Register Call Summary for Register MMCHS_AC12

MMC/SD/SDIO Register Manual

- MMC/SD/SDIO Registers Mapping Summary: [0] [1] [2]

Table 22-76. MMCHS_CAPA

Address Offset	0x140		
Physical Address	0x4809 C140	**Instance**	MMCHS1
	0x480A D140		MMCHS3
	0x480B 4140		MMCHS2
Description	Capabilities register. This register lists the capabilities of the MMC/SD/SDIO host controller.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved					VS18	VS30	VS33	SRS	DS	HSS	Reserved			MBL		Reserved		BCF						TCU	Reserved	TCF					

Bits	Field Name	Description	Type	Reset
31:27	Reserved	Reserved bit field. Do not write any value.	R	0x00
26	VS18	Voltage support 1.8V Initialization of this register (via a write access to this register) depends on the system capabilities. The host driver shall not modify this register after the initialization. This register is only reinitialized by a hard reset (via MMCi_RESET signal). Read 0x0: 1.8 V Not Supported Write 0x0: 1.8 V Not supported Read 0x1: 1.8 V Supported Write 0x1: 1.8 V Supported MMCHS1, 2 and 3: This bit must be set to 1.	RW	0
25	VS30	Voltage support 3.0V Initialization of this register (via a write access to this register) depends on the system capabilities. The host driver shall not modify this register after the initialization. This register is only reinitialized by a hard reset (via MMCi_RESET signal) Read 0x0: 3.0 V Not Supported Write 0x0: 3.0 V Not supported Read 0x1: 3.0 V Supported Write 0x1: 3.0 V Supported MMCHS1: This bit must be set to 1. MMCHS2 and 3: This bit must be left to 0.	RW	0
24	VS33	Voltage support 3.3V Initialization of this register (via a write access to this register) depends on the system capabilities. The host driver shall not modify this register after the initialization. This register is only reinitialized by a hard reset (via MMCi_RESET signal) Read 0x0: 3.3 V Not Supported Write 0x0: 3.3 V Not supported Read 0x1: 3.3 V Supported Write 0x1: 3.3 V Supported MMCHS1, 2 and 3: This bit must be left to 0.	RW	0
23	SRS	Suspend/Resume support (SDIO cards only)This bit indicates whether the host controller supports Suspend/Resume functionality. Read 0x0: The Host controller does not Suspend/Resume functionality. Read 0x1: The Host controller supports Suspend/Resume functionality.	R	1
22	DS	DMA support. This bit indicates that the Host Controller is able to use DMA to transfer data between system memory and the Host Controller directly. Read 0x0: DMA Not Supported Read 0x1: DMA Supported	R	1

Bits	Field Name	Description	Type	Reset
21	HSS	High speed support. This bit indicates that the host controller supports high speed operations and can supply an up-to-52 MHz clock to the card. Read 0x0: High Speed Not Supported Read 0x1: High Speed Supported	R	1
20:18	Reserved	Reserved bit field. Do not write any value.	R	0x0
17:16	MBL	Maximum block length. This value indicates the maximum block size that the host driver can read and write to the buffer in the host controller. The host controller supports 512 bytes and 1024 bytes block transfers. Read 0x0: 512 bytes Read 0x1: 1024 bytes Read 0x2: 2048 bytes	R	0x1
15:14	Reserved	Reserved bit field. Do not write any value	R	0x0
13:8	BCF	Base clock frequency for clock provided to the card. Read 0x0: The value indicating the base (maximum) frequency for the output clock provided to the card is system dependent and is not available in this register. Get the information via another method. See Chapter 4, *Power Reset and Clock Management*, for more information on the value of FUNC_96M_CLK clock signal.	R	0x00
7	TCU	Timeout clock unit. This bit shows the unit of base clock frequency used to detect Data Timeout Error (MMCi.MMCHS_STAT[20] DTO bit). Read 0x0: kHz Read 0x1: MHz	R	1
6	Reserved	Reserved. This bit is initialized to zero, and writes to it are ignored.	R	0
5:0	TCF	Timeout clock frequency. The timeout clock frequency is used to detect Data Timeout Error (MMCi.MMCHS_STAT[20] DTO bit). Read 0x0: The timeout clock frequency depends on the frequency of the clock provided to the card. The value of the timeout clock frequency is not available in this register. **Note:** You can have the timeout clock frequency by dividing FUNC_96M_CLK by the value of the MMCi.MMCHS_SYSCTL[15:6] CLKD bit field.	R	0x00

Table 22-77. Register Call Summary for Register MMCHS_CAPA

MMC/SD/SDIO Integration • Resets: [0]
MMC/SD/SDIO Functional Description • Data Buffer: [1]
MMC/SD/SDIO Basic Programming Model • Set MMCHS Default Capabilities: [2]
MMC/SD/SDIO Use Cases and Tips • Programming Flow: [3] [4] [5]
MMC/SD/SDIO Register Manual • MMC/SD/SDIO Registers Mapping Summary: [6] [7] [8] [9] [10] [11] [12]

Table 22-78. MMCHS_CUR_CAPA

Address Offset	0x148		
Physical Address	0x4809 C148	**Instance**	MMCHS1
	0x480A D148		MMCHS3
	0x480B 4148		MMCHS2
Description	Maximum current capabilities Register. This register indicates the maximum current capability for each voltage. The value is meaningful if the voltage support is set in the capabilities register (MMCi.MMCHS_CAPA). Initialization of this register (via a write access to this register) depends on the system capabilities. The host driver shall not modify this register after the initialization. This register is only reinitialized by a hard reset (via MMCi_RESET signal)		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								CUR_1V8								CUR_3V0								CUR_3V3							

Bits	Field Name	Description	Type	Reset
31:24	Reserved	Reserved. This bit is initialized to zero, and writes to it are ignored.	R	0x0
23:16	CUR_1V8	Maximum current for 1.8 V	RW	0x0
		Read 0x0 The maximum current capability for this voltage is not available. Feature not implemented.		
15:8	CUR_3V0	Maximum current for 3.0 V	RW	0x0
		Read 0x0 The maximum current capability for this voltage is not available. Feature not implemented.		
7:0	CUR_3V3	Maximum current for 3.3 V	RW	0x0
		Read 0x0 The maximum current capability for this voltage is not available. Feature not implemented.		

Table 22-79. Register Call Summary for Register MMCHS_CUR_CAPA

MMC/SD/SDIO Integration
- Resets: [0]

MMC/SD/SDIO Basic Programming Model
- Set MMCHS Default Capabilities: [1]

MMC/SD/SDIO Register Manual
- MMC/SD/SDIO Registers Mapping Summary: [2]

Table 22-80. MMCHS_REV

Address Offset	0x1FC		
Physical Address	0x4809 C1FC	**Instance**	MMCHS1
	0x480A D1FC		MMCHS3
	0x480B 41FC		MMCHS2
Description	Versions Register. This register contains the hard coded RTL vendor revision number, the version number of SD specification compliancy and a slot status bit. MMCi.MMCHS_REV[31:16] = Host controller version MMCi.MMCHS_REV[15:0] = Slot Interrupt Status		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
VREV								SREV								Reserved															SIS

Bits	Field Name	Description	Type	Reset
31:24	VREV	Vendor Version Number: IP revision [31:28] Major revision [27:24] Minor revision Examples: 0x10 for 1.0 0x21 for 2.1	R	See (1)
23:16	SREV	Specification Version Number. This status indicates the Standard SD Host Controller Specification Version. The upper and lower 4-bits indicate the version. Read 0x0: SD Host Specification Version 1.0	R	0x00
15:1	Reserved	Reserved bit field. Do not write any value	R	0x0000
0	SIS	Slot Interrupt Status. This status bit indicates the inverted state of interrupt signal for the module. By a power on reset or by setting a software reset for all (MMCi.MMCHS_SYSCTL[24] SRA), the interrupt signal shall be deasserted and this status shall read 0.	R	0

(1) TI internal data

Table 22-81. Register Call Summary for Register MMCHS_REV

MMC/SD/SDIO Register Manual

- MMC/SD/SDIO Registers Mapping Summary: [0] [1] [2]

Memory Stick PRO Host Controller

This chapter presents an overview of the Memory Stick PRO™ host controller in the OMAP34xx multimedia device.

Topic **Page**

23.1 Memory Stick PRO Host Controller Overview 3174

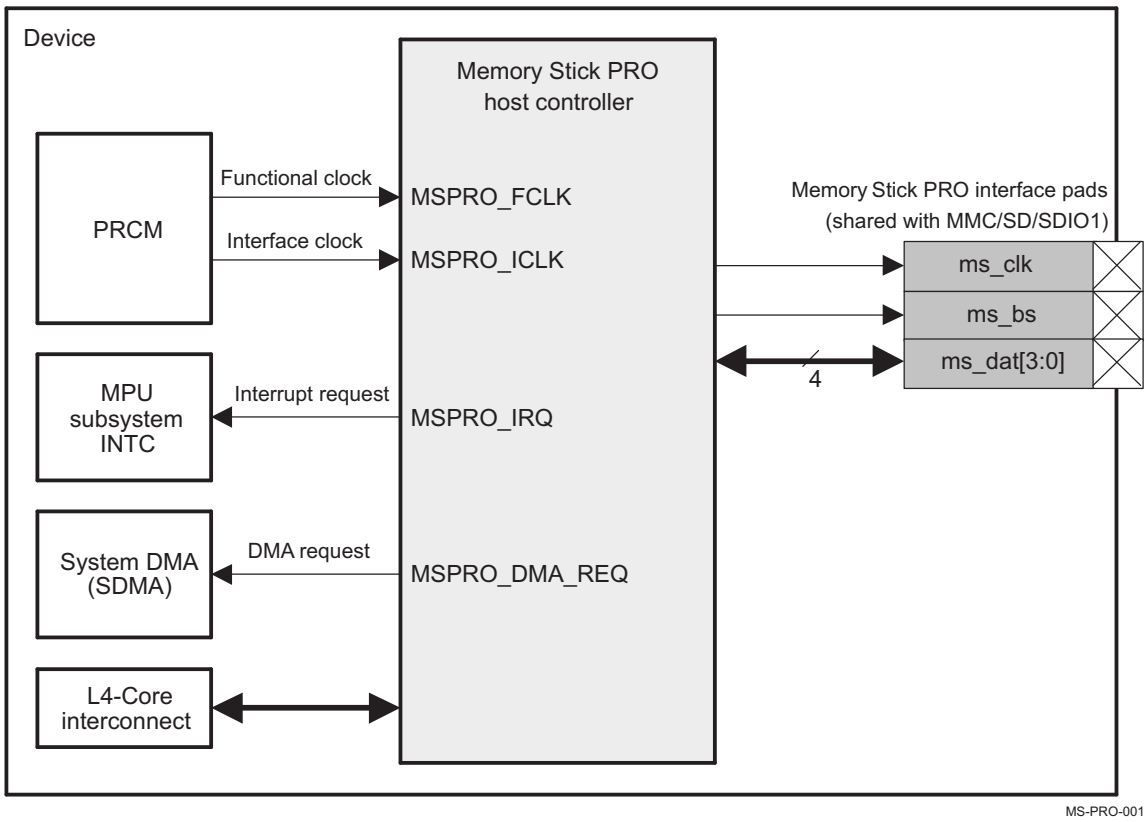
23.1 Memory Stick PRO Host Controller Overview

The Memory Stick PRO host controller provides the device with an interface between a microprocessor unit (MPU) local host (LH) and either a Memory Stick® or a Memory Stick PRO card.

The Memory Stick PRO host controller is connected to the L4-Core interconnect.

Figure 23-1 shows an overview of the Memory Stick PRO host controller.

Figure 23-1. Memory Stick PRO Host Controller Overview



23.1.1 Main Features

The Memory Stick PRO host controller includes the following main features:

- Supports Memory Stick v1.x and Memory Stick PRO
- Data transmit/receive internal first-in first-out (FIFO) (64 bits x 4)
- Cyclic redundancy check (CRC) circuit
- Memory Stick serial clock (Memory Stick v1.x: 19.2 MHz; Memory Stick PRO: 32 MHz with 96-MHz functional clock source)
- Direct memory access (DMA) supported
- Lower 16-/32-bit access possible
- Little endian (internally big endian)
- L4 interconnect support

Note: Other information about Memory Stick PRO host controller are not available in the public domain.

High-Speed USB Host Subsystem and High-Speed USB OTG Controller

This chapter describes the high-speed universal serial bus (USB) host subsystem and the high-speed USB On-The-Go (OTG) controller of the OMAP34xx multimedia device.

Topic **Page**

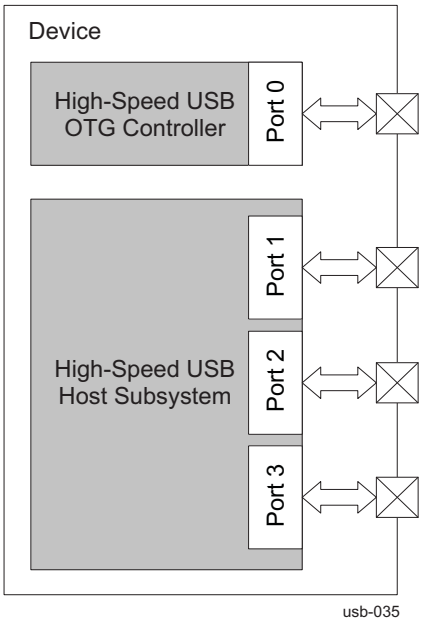
24.1 High-Speed USB OTG Controller 3177
24.2 High-Speed USB Host Subsystem 3198

Overview

The device contains two USB modules:

- A High-Speed USB OTG Controller (see Section 24.1, *High-Speed USB OTG Controller*)
- A High-Speed USB Host Subsystem (see Section 24.2, *High-Speed USB Host Subsystem*)

Figure 24-1. USB Modules Overview



24.1 High-Speed USB OTG Controller

Note: The High-Speed USB OTG Controller is an instantiation of the MUSBMHDRC from Mentor Graphics Corporation.

This document contains materials that are ©2003-2007 Mentor Graphics Corporation.

Mentor Graphics is a registered trademarks of Mentor Graphics Corporation or its affiliated companies in the United States and other countries.

24.1.1 High-Speed USB OTG Controller Overview

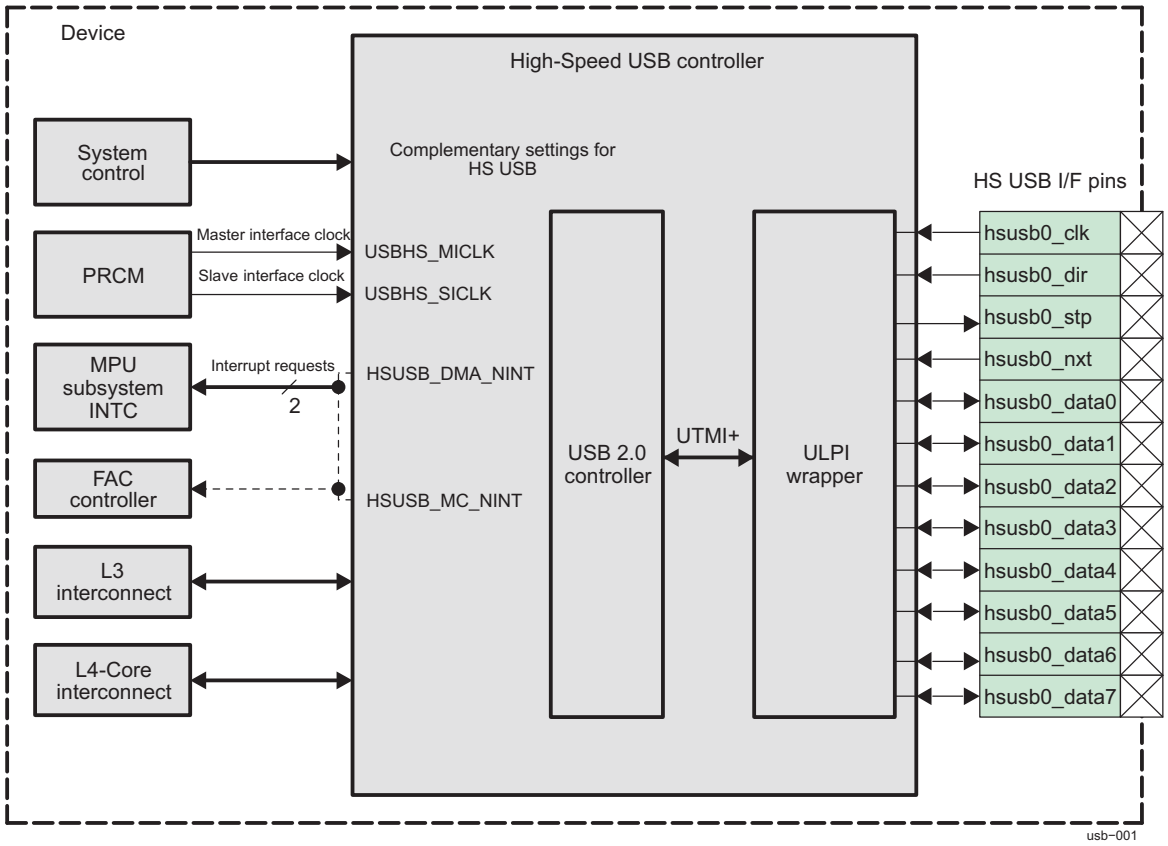
The OMAP34xx High-Speed USB controller is a high-speed, USB OTG dual-role-device (DRD) link controller supporting the following modes:

- USB 2.0 peripheral (function controller) in high/full speed (480/12 Mbps, respectively)
- USB 2.0 host in high/full/low speed (480/12/1.5 Mbps respectively), with one downstream port but multipoint capability when a hub is connected to it (split transaction support, etc.)
- USB 2.0 OTG DRD in high/full speed, with HNP and SRP support.

The high-speed USB controller supports a single USB port, which uses the ULPI interface mode, to connect to an off-chip transceiver (12-pin/8-bit data SDR mode).

Figure 24-2 highlights the high-speed USB controller.

Figure 24-2. High-Speed USB Controller Highlight



24.1.1.1 Main Features

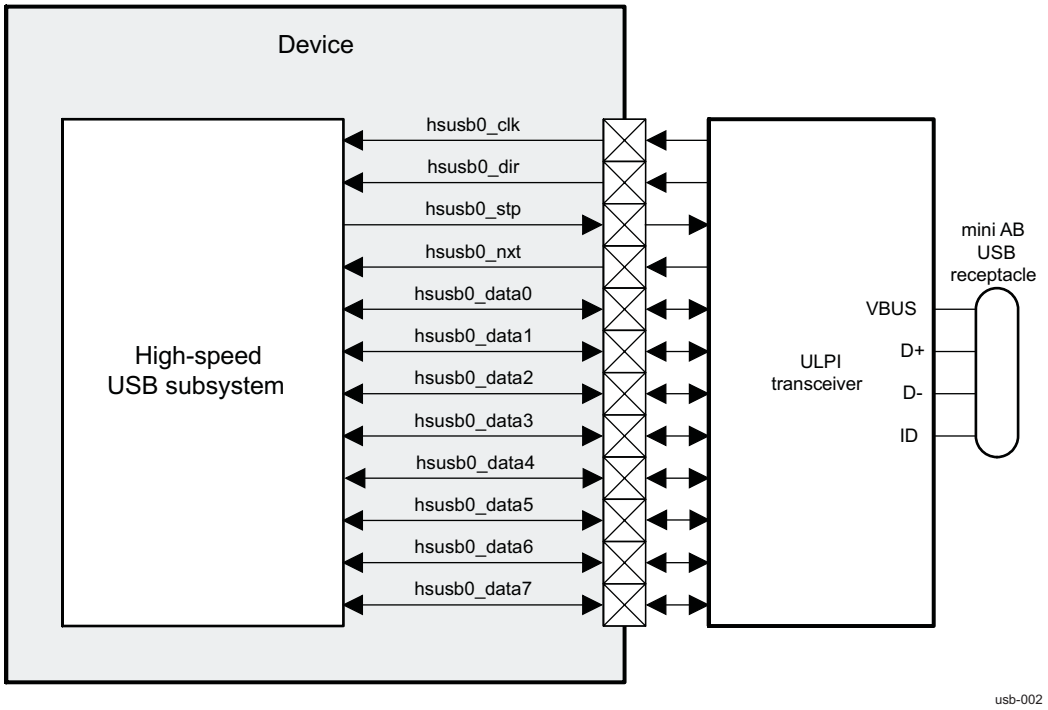
The high-speed USB controller has the following features:

- Operates either as the function controller of a high-/full-speed USB peripheral or as the host/peripheral in point-to-point or multipoint communications with other USB functions
- Complies with the USB 2.0 standard for high-speed (480 Mbps) functions and with the OTG supplement (Revision 1.0a)
- Includes all transfer types: Control, bulk, isochronous, interrupt
- Supports SRP and HNP
- Supports suspend/resume and remote wakeup
- Supports high-bandwidth isochronous and interrupt transfers
- Contains one PHY (physical layer) interface: ULPI interface (12/8-pin version)
- 15 transmit endpoints and 15 receive endpoints in addition to control endpoint 0
- Each endpoint has its own FIFO, with the following properties:
  - Implemented within a single, 16K-byte internal RAM
  - Can be dynamically sized by software
  - Can be configured to hold multiple packets (up to 8192 bytes per FIFO)
  - Can be accessed either by direct access or by DMA controller
- Software connect/disconnect option for peripheral
- Performs all transaction scheduling in hardware

24.1.2 High-Speed USB OTG Controller Environment

Figure 24-3 shows a typical application using the high-speed USB controller.

Figure 24-3. High-Speed USB Controller Typical Application System



The high-speed USB controller supports a single USB port, which uses the ULPI interface mode, to connect to an off-chip transceiver (12-pin/8-bit data SDR mode). A mini A-B external receptacle allows the connection of an external device.

The current implementation of ULPI includes a method for manual, software-controlled generation of PHY-side register accesses supporting the following features:

- Access to vendor-specific or optional PHY-side registers
- Carkit operation (including interrupt, PHY-side registers)
- Access to vendor ID, product ID, scratch and debug registers

Note: In the carkit mode, the interface must connect to the TI TWL4030 companion chip for proper use.

The 12-pin ULPI interface uses an 8-bit data bus with data synchronous to the rising edge of the PHY clock (SDR mode); the 8-pin ULPI uses a 4-bit data bus with data generated on both the rising and falling clock edge (DDR mode).

The TI high-speed USB controller supports only the 12-pin, 8-bit version ULPI interface.

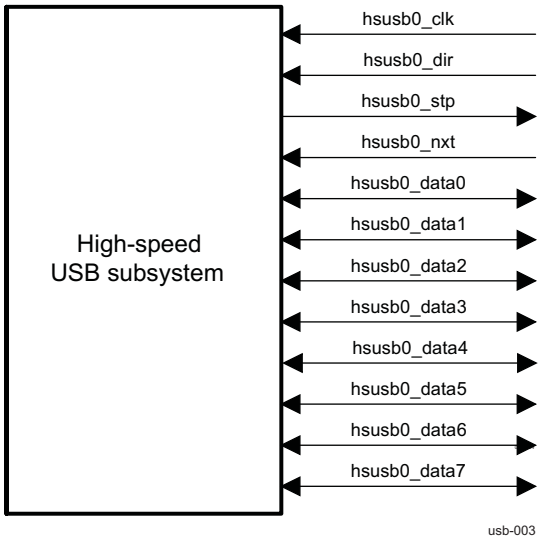
The 12-pin/8-bit data SDR ULPI interface is selected through the USBOTG.OTG_INTERFSEL[1:0] PHYSEL field and must be set to 0x1.

24.1.2.1 High-Speed USB Controller Functional Interfaces

24.1.2.1.1 Basic High-Speed USB Controller Pins

Figure 24-4 shows the high-speed USB controller functional interface.

Figure 24-4. High-Speed USB Controller Functional Interface Signals



24.1.2.1.2 High-Speed USB Controller Interface Description

Table 24-1 describes the I/O of the high-speed USB controller interface.

Table 24-1. Input/Output Description

Signal Name	I/O[1]	Description	Reset Value
hsusb0_clk	I	60-MHz clock input from ULPI PHY[2]	Unknown
hsusb0_dir	I	Data direction control from ULPI PHY	Unknown
hsusb0_stp	O	Stop signal to ULPI PHY	1
hsusb0_nxt	I	Next signal from ULPI PHY	Unknown
hsusb0_data0	I/O	Bidirectional DATA0	Unknown
hsusb0_data1	I/O	Bidirectional DATA1	Unknown
hsusb0_data2	I/O	Bidirectional DATA2	Unknown
hsusb0_data3	I/O	Bidirectional DATA3	Unknown
hsusb0_data4	I/O	Bidirectional DATA4	Unknown
hsusb0_data5	I/O	Bidirectional DATA5	Unknown
hsusb0_data6	I/O	Bidirectional DATA6	Unknown
hsusb0_data7	I/O	Bidirectional DATA7	Unknown

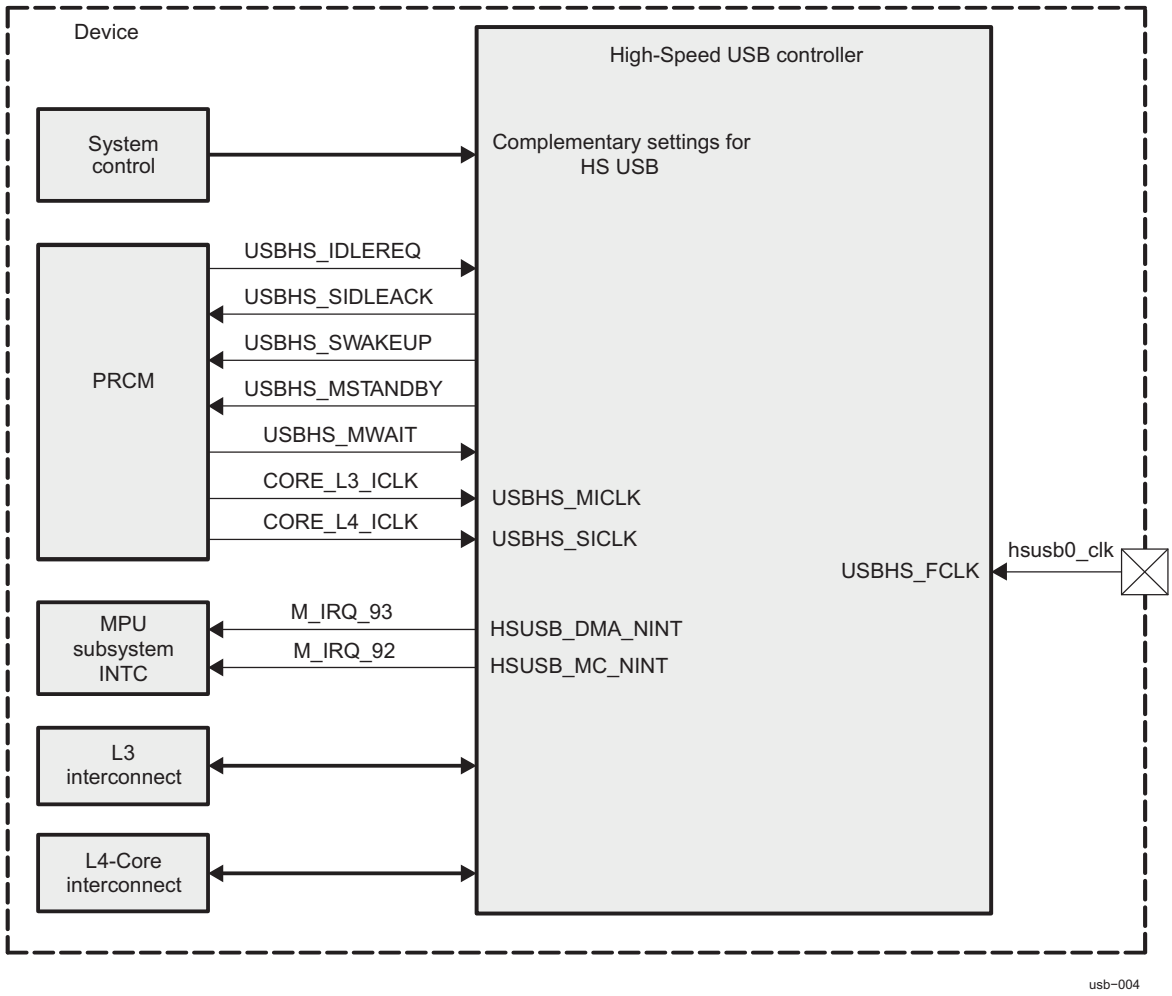
(1) I = Input; O = Output
(2) This signal is also used as re-timing input.

24.1.3 High-Speed USB OTG Controller Integration

The high-speed USB controller is connected to L3 interconnect master (initiator) and slave (target) interfaces and L4-Core interconnect. The L3 interconnect generates data traffic within the OMAP34xx device. The L4-Core interconnect is a configuration port for register setting.

Figure 24-5 highlights the high-speed USB controller integration in the OMAP34xx device.

Figure 24-5. High-Speed USB Controller Integration



24.1.3.1 Clocking, Reset, and Power-Management Scheme

24.1.3.1.1 Clocks

24.1.3.1.1.1 Module Clocks

Three clocks are provided to the high-speed USB controller, as shown in Table 24-2.

Table 24-2. USB Clocks

Attributes	Frequency	Name	Mapping	Comments
Functional clock	60 MHz	USBHS_FCLK	hsusb0_clk	Source is external transceiver (ULPI)
Master Interface clock	Depending on PRCM register settings	USBHS_MICLK	CORE_L3_ICLK	Source is PRCM module

Table 24-2. USB Clocks (continued)

Attributes	Frequency	Name	Mapping	Comments
Slave Interface clock	Depending on PRCM register settings	USBHS_SICLK	CORE_L4_ICLK	Source is PRCM module

24.1.3.1.1.2 Master Interface Clock

The master interface clock USBHS_MICLK comes from the PRCM module. This clock is controlled by the PRCM register bits PRCM.CM_ICLKEN1_CORE[4]:

0: Disabled
1: Enabled

and PRCM.CM_AUTOIDLE1_CORE[4] (enables/disables automatic control of the interface clock)—see Table 24-3.

Table 24-3. High-Speed USB Interface Clock

PRCM.CM_AUTOIDLE1_CORE[4]	PRCM.CM_ICLKEN1_CORE[4]	Interface Clock
0	0	Disabled
0	1	Enabled
1	0	Disabled
1	1	Automatic enabling/disabling

24.1.3.1.1.3 Functional Clock

The functional clock (60MHz), USBHS_FCLK, comes from the external ULPI transceiver through the hsusb0_clk input pin.

24.1.3.1.2 Resets

The high-speed USB controller can be reset either by the domain reset (hardware reset) or by setting a dedicated configuration bit (software reset) in the module.

24.1.3.1.2.1 Hardware Reset

The high-speed USB controller is attached to the CORE power domain: The CORE_RST signal resets the module (see Chapter 4, *Power, Reset, and Clock Management*). The hardware reset signal has a global reset action on the high-speed USB controller.

24.1.3.1.2.2 Software Reset

The high-speed USB controller has a software reset through the USBOTG.OTG_SYSCONFIG[1] SOFTRESET bit (0: Normal mode, 1: Module is reset).

This bit controls the software reset. Writing 1 to this bit resets the module. The bit value 1 remains until the reset is complete. When the software reset is complete, the SOFTRESET bit is automatically reset to 0 and has the same effect as the hardware reset.

24.1.3.1.3 Power-Management Scheme

24.1.3.1.3.1 Overview

To save dynamic power consumption, an efficient idle scheme in the device is based on the following:

- An efficient local autoclock gating for each module
- The implementation of control sideband signals between the PRCM module and each module

This enhanced idle control allows clocks to be activated/deactivated safely without complex software intervention. In both cases, the high-speed USB controller power management is applied only to the interface clock domain. The USB functional clock (60 MHz), USBHS_FCLK, is controlled by the transceiver and is responsible only for a very small percentage of the module overall power consumption.

The high-speed USB controller has both master (initiator) and slave (target) interfaces.

- As an initiator, the high-speed USB controller implements the standby handshake protocol to inform the PRCM module when it enters standby mode and does not generate traffic on the interconnect.
- As a target, the high-speed USB controller implements the IDLE handshake protocol to allow the PRCM module requiring it to enter idle mode.

24.1.3.1.3.2 System Power Management

Master Interface Power Management

The high-speed USB controller can choose to go to standby mode, in which case it stops generating transactions on the interconnect. The module standby leads the PRCM to disable the USB clocks to save power.

The high-speed USB controller has a MSTANDBY handshake mechanism with the PRCM module (see Figure 24-5).

The module is ready to enter standby mode (indicated by the MSTANDBY signal to the PRCM asserted) when there is no USB activity and the module is idle. It means the following:

- The module is committed not to start any new transaction on its master interface.
- The module is idle and, therefore, the power manager can start the procedure to turn off the interface clock, if needed. This procedure must be implemented using the slave power-management protocol.

The handshake mechanism lets the module to go to standby state based on the USBOTG.OTG_SYSCONFIG[13:12] MIDLEMODE field.

- Smart standby

  The high-speed USB controller is configured in smart-standby mode (USBOTG.OTG_SYSCONFIG[13:12] MIDLEMODE field = 0x2). The module is ready to enter standby mode (MSTANDBY is asserted) when there is no more activity on the USB master interface of the interconnect. MSTANDBY is asserted when the module is idle and deasserted when the module is activated by either an external USB event or an appropriate register access. The module then waits for MWAIT deassertion before a DMA transfer is started.
- Force standby

  The high-speed USB controller is configured in force-standby mode (USBOTG.OTG_SYSCONFIG[13:12] MIDLEMODE field = 0x0).
  - When the high-speed USB controller operates as a host: The USBOTG.POWER[1] SUSPENDMODE bit is set to 1 to bring the module to low-power mode (suspend mode). After this setting, the high-speed USB controller waits for its idle state. The USBOTG.OTG_FORCESTDBY[0] ENABLEFORCE bit must be set to 1 to assert MSTANDBY. Similarly, to release the MSTANDBY signal, an appropriate register access must be applied, which can be either of the following two cases:
    - Remote wakeup causes a RESUME interrupt
    - Set the USBOTG.POWER[3] RESET bit to 1.
    - Write 0 to the USBOTG.OTG_FORCESTDBY[0] ENABLEFORCE bit.

      OR
    - Set the USBOTG.POWER[3] RESET bit to 1.
    - Write 0 to the USBOTG.OTG_FORCESTDBY[0] ENABLEFORCE bit.
  - When the high-speed USB controller operates as a peripheral: When the USB bus is idle for 3 ms, a SUSPEND interrupt is generated by the high-speed USB controller. The USBOTG.OTG_FORCESTDBY[0] ENABLEFORCE bit must be set to 1 to enable the MSTANDBY signal. The high-speed USB controller then asserts MSTANDBY. Similarly, to release the MSTANDBY signal, an appropriate register access must be applied, which can be either of the following two cases:
    - Set the USBOTG.POWER[2] RESUME bit to 1.

- Write 0 to the USBOTG.OTG_FORCESTDBY[0] ENABLEFORCE bit.
  OR
- RESET interrupt is generated by the high-speed USB controller.
- Write 0 to the USBOTG.OTG_FORCESTDBY[0] ENABLEFORCE bit.

When MSTANDBY is deasserted as a consequence of the previous register access, the module waits for MWAIT deassertion before a DMA transfer is started.

- No standby

  The high-speed USB controller is configured in no-standby mode (USBOTG.OTG_SYSCONFIG[13:12] MIDLEMODE field = 0x1). The module never enters standby mode (that is, MSTANDBY is never asserted).

Table 24-4 describes the high-speed USB master interface power management modes.

Table 24-4. High-Speed USB Master Interface Power Management Modes

Power Management Mode Requested by the PRCM	USBOTG.OTG_SYSCONFIG[13:12] MIDLEMODE Field
Force-standby	0x0
No-standby	0x1
Smart-standby	0x2
Reserved	0x3

Slave Interface Power Management

The high-speed USB controller can be configured through the USBOTG.OTG_SYSCONFIG[4:3] SIDLEMODE field as one of the following acknowledgment modes:

- Smart-Idle Mode

  When the high-speed USB controller receives an IDLE request from the PRCM module:

  – The interface clock USBHS_ICLK is disabled (PRCM register bit PRCM.CM_ICLKEN1_CORE[4] set to 0) or under automatic control (PRCM register bits PRCM.CM_ICLKEN1_CORE[4] and PRCM.CM_AUTOIDLE1_CORE[4] both set to 1)

  – L4 interface clock idle transitions:

    Configured in smart-idle mode (USBOTG.OTG_SYSCONFIG[4:3] SIDLEMODE field = 0x2), the high-speed USB controller checks for no ongoing activity. The idle acknowledge then is asserted and the module waits for active system clock gating by the PRCM module (this occurs only when all peripherals supplied by the same L3 clock domain are also ready for idle).

  Once in idle mode (when the PRCM module gates the interface clock), the module has no activity, the interface clock paths are gated, no interrupt request can be generated, and the module is ready to issue a wake-up request. If a wake-up condition occurs, the high-speed USB controller exits from idle mode if the USBOTG.OTG_SYSCONFIG[2] ENABLEWAKEUP bit is set to 1 (wake-up capability enabled) and the PRCM register bit PRCM.PM_WKEN1_CORE[4] is also set to 1.

- Force-Idle Mode

  When the high-speed USB controller receives an IDLE request from the PRCM module:

  – The interface clock USBHS_ICLK is disabled (PRCM register bit PRCM.CM_ICLKEN1_CORE[4] set to 0) or under automatic control (PRCM register bits PRCM.CM_ICLKEN1_CORE[4] and PRCM.CM_AUTOIDLE1_CORE[4] both set to 1)

  – The L4 interface clock idle transitions:

    Configured in force-idle mode (USBOTG.OTG_SYSCONFIG[4:3] SIDLEMODE field = 0x0), the high-speed USB controller waits unconditionally for active system clock gating by the PRCM module (this occurs only when all peripherals supplied by the same L3 clock domain are also ready for idle).

  Once in idle mode (when the PRCM module gates the interface clock), the module has no activity, the interface clock paths are gated, no interrupt request can be generated, and the wake-up feature is totally inhibited.

- No-Idle Mode

  When the high-speed USB controller receives an IDLE request from the PRCM module:

– The interface clock(s) USBHS_ICLK are disabled (PRCM register bit PRCM.CM_ICLKEN1_CORE[4] set to 0) or under automatic control (PRCM register bits PRCM.CM_ICLKEN1_CORE[4] and PRCM.CM_AUTOIDLE1_CORE[4] both set to 1)
– L4 interface clock idle transitions
Configured in no-idle mode (USBOTG.OTG_SYSCONFIG[4:3] SIDLEMODE field = 0x1), the high-speed USB controller module does not go to idle mode and the idle acknowledge is never sent.

Table 24-5 describes the high-speed USB slave interface power management modes.

Table 24-5. High-Speed USB Slave Interface Power Management Modes

Power Management Mode Requested by the PRCM	USBOTG.OTG_SYSCONFIG[4:3] SIDLEMODE Field
Force-idle	0x0
No-idle	0x1
Smart-idle	0x2
Reserved	0x3

Note: The high-speed USB controller standby status can be checked by the PRCM module register bit CM_IDLEST1_CORE[4].

- 0: High-Speed USB is active.
- 1: High-Speed USB is in standby mode.

The high-speed USB controller wake-up status can be checked by the PRCM module register bit PM_WKST1_CORE[4].

- Read 0: Wakeup has not occurred or was masked.
- Read 1: Wakeup has occurred.
- Write 0: Status bit unchanged.
- Write 1: Status bit is cleared to 0.

24.1.3.1.3.3 Local Power Management

The high-speed USB controller has local power management by internal clock gating features:

- Internal interface clock autogating: Clock for the L3 interconnect logic can be gated when the module is not accessed, if the USBOTG.OTG_SYSCONFIG[0] AUTOIDLE bit is set. Otherwise, this logic is free-running on the interface clock. This bit is used to save power when the module is not used because of the multiplexing configuration selected at the chip level. This bit has precedence over all other internal configuration bits.

24.1.3.1.4 Power Domain

The high-speed USB controller is attached to the CORE power domain (see Chapter 4, *Power, Reset, and Clock Management*).

24.1.3.2 Hardware Requests

24.1.3.2.1 Interrupt Requests

Table 24-6 lists the interrupt lines that are driven out from the high-speed USB controller to the MPU subsystem INTC.

Table 24-6. High-Speed USB Interrupts

Name	Mapping	Comments
HS USB Controller Interrupt		
HSUSB_MC_NINT	M_IRQ_92	Destination is MPU subsystem interrupt controller
HS USB DMA Controller Interrupt		
HSUSB_DMA_NINT	M_IRQ_93	Destination is MPU subsystem interrupt controller

24.1.3.2.2 IDLE Handshake Protocol

The PRCM handles an IDLE handshake protocol for the high-speed USB controller. The IDLE handshake protocol allows the PRCM requiring the high-speed USB controller to enter idle mode. The high-speed USB controller acknowledges when it is ready.

24.1.3.2.3 MSTANDBY Handshake Protocol

The PRCM module handles an MSTANDBY handshake protocol for the high-speed USB controller, which initiates the MSTANDBY handshake to inform the PRCM module when it enters standby mode and does not generate traffic on interconnect.

24.1.3.2.4 Wake-Up Request

The high-speed USB controller generates a wake-up request signal (UBSHS_SWAKEUP) to the PRCM module.

24.1.4 High-Speed USB OTG Controller Functional Description

24.1.4.1 Inventra™ MUSBMHDRC

The core functionality of the module is provided by a third-party IP, the Inventra MUSBMHDRC from Mentor Graphics Corporation.

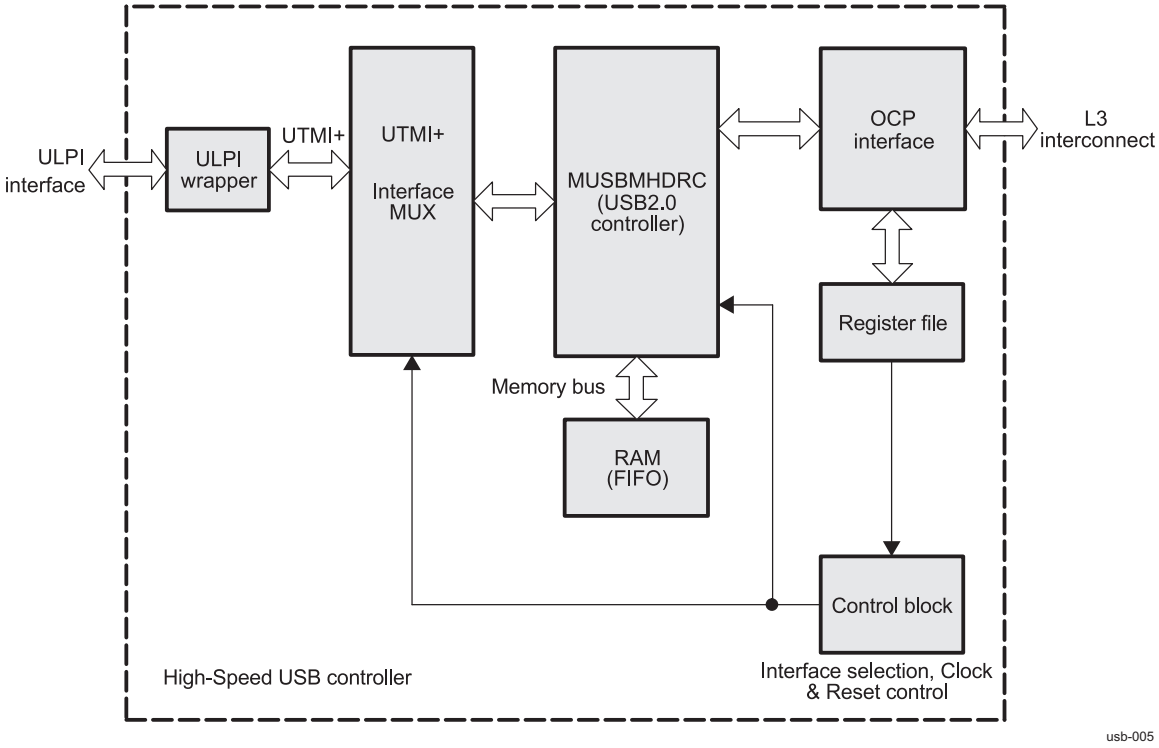
The high-speed USB controller is basically the appropriate adaptation/integration of this IP to comply with TI requirements.

The high-speed USB controller includes the following:

- MUSBMHDRC (USB 2.0 controller)
- Master and slave bridges for conversion AHB to L3 interconnect and L3 interconnect to AHB
- Built-in self-test (BIST) controller
- Two memories (total of 16Kbytes; available for various endpoints)
- Reset and clock generation
- DFT compliance with regards to latest specs

Figure 24-6 shows the high-speed USB controller.

Figure 24-6. High-Speed USB Controller



24.1.4.2 Configuration

The Mentor Graphics IP RTL MUSBMHDRC can be configured to generate some customized hardware. Those options are all hard-wired and cannot be reprogrammed in the case of the TI High-Speed USB controller.

The following options are selected:

- 8-bit internal data path processing width (mandated by the use of ULPI)
- 32-bit vcontrol/vstatus support
- External charge pump: Hacked
- Software connect/disconnect supported
- Little-endian byte ordering
- Eight DMA channels (with internal DMA initiator implemented)
- Dynamic FIFO sizing enabled
- 16K bytes RAM buffer
- 15 IN/OUT endpoints in addition to control endpoint 0
- IN/OUT bulk packet splitting enabled
- High-bandwidth IN/OUT isochronous support enabled (see *Universal Serial Bus Specification Revision 2.0*)

24.1.4.3 Basic Operation

This section provides an overview of the module basic operation.

To implement the most time-critical functions in hardware, the module provides all of the encoding, decoding, and checking required to send and receive USB packets—interrupting the MPU only when endpoint data is successfully transferred. Generally, the following steps are performed:

24.1.4.3.1 Module Initialization

First, the firmware must do the overall initialization of the module by configuring the interrupts, the DMA controller, and the individual endpoints.

The specific items of a configuration are for each endpoint:

- Direction TX/RX
- Speed: High/full/low
- Special host settings when used in host mode (function address/hub parameters, etc.)
- Transaction protocol: Control/isochronous/bulk/interrupt
- Maximum packet size, that can be transferred through the endpoint
- Whether DMA is required (DMA enable)
- DMA mode
- Start address of the endpoint FIFO within the RAM block
- Maximum packet size allowed
- Whether double-buffering is required. This and the previous option define the amount of space that must be allocated to the FIFO. In the case of double-buffering, the FIFO size is doubled; therefore, the maximum supported size is 8196 bytes for the endpoint FIFOs.

Note: On the system level, it must be ensured that the reset and the interface are selected before the 60-MHz functional clock (USBHS_FCLK) is applied to the module. Correct functionality is not ensured if either a reset deassertion or a change in the interface selection occurs when the functional clock USBHS_FCLK is already running.

24.1.4.3.1.1 MSTANDBY Deassertion

MSTANDBY is asserted during reset and continues to be 1 after reset. The high-speed USB controller does not perform master interface transactions until this signal is deasserted. Firmware must follow these steps to begin normal operations on the master interface bus:

- Write 0 to the USBOTG.OTG_FORCESTDBY[0] ENABLEFORCE bit.
- Put the high-speed USB controller into no-idle mode (USBOTG.OTG_SYSCONFIG[4:3] SIDLEMODE field = 0x1) and no-standby mode (USBOTG.OTG_SYSCONFIG[13:12] MIDLEMODE field = 0x1).

Even though ENABLEFORCE is 0, MSTANDBY remains asserted until the high-speed USB controller core is out of idle state.

24.1.4.3.2 Transaction Handling

Depending on whether the module is in the host or peripheral mode, an RX endpoint is assigned to IN or OUT transactions, respectively. Similarly, a TX endpoint is used for IN transactions in the peripheral mode and for OUT transactions in the host mode. When a transaction is complete, the appropriate endpoint interrupt is generated (if enabled).

When a transaction is handled through an RX endpoint, the received packet is placed in the RX FIFO, and the appropriate RX endpoint interrupt is generated. The packet can then be unloaded from the FIFO either manually (that is, by the MPU) or by the DMA. Although an RX endpoint is always active for a peripheral, in host mode the transaction must first be initiated by setting the appropriate request flag (REQPKT). This indicates to the transaction scheduler that there is an active transaction on this endpoint, which requires an IN token to be sent to the target function.

When a transaction is handled through a TX endpoint, the packet to send is loaded into the TX FIFO either manually (that is, by the MPU) or by the DMA. The transaction then must be initiated by setting the appropriate ready flag (TXPKTRDY) to indicate the availability of the new packet. When the packet is successfully sent (scheduled according to the protocol rules), the appropriate TX interrupt is generated.

24.1.4.4 Optional Features

This section gives a quick overview of the optional features that are available, depending on the endpoint configuration.

24.1.4.4.1 Double Packet Buffering

When double packet buffering is enabled (by setting the MSB of the FIFOSZ register two data packets can be stored in the FIFO. This option can be set for both the TX and RX endpoints in both the peripheral and host mode. For a TX endpoint, double-buffering means that up to two packets can be loaded into the FIFO awaiting transmission; for an RX endpoint, one packet can be received while another is being read. Double packet buffering is especially advisable for isochronous transactions to avoid underrun or overrun errors.

The high-speed USB controller provides dynamic FIFO sizing with an overall RAM size of 16K bytes, which can then be allocated to the different endpoints when the module is initialized. The maximum size of an endpoint FIFO is 4096 bytes for single packet buffering and 8192 bytes for double packet buffering. The firmware must ensure that a block of RAM is properly assigned to all TX and RX endpoints, considering the total RAM size and the maximum packet size set for the endpoint.

24.1.4.4.2 DMA

The high-speed USB controller has an internal DMA controller for efficient loading/unloading of the endpoint FIFOs.

If the DMA is enabled for a TX endpoint, a DMA transfer occurs whenever the endpoint can accept another packet in its FIFO (that is, the DMA controller loads the packets into the FIFO without processor intervention).

If the DMA is enabled for an RX endpoint, a DMA transfer occurs whenever the endpoint has a packet in its FIFO (that is, the DMA CONTROLLER unloads the packets from the FIFO without processor intervention).

The high-speed USB controller supports two DMA request modes, which also affects the generation of endpoint interrupts.

- DMA request mode 0 is especially advisable for isochronous transfers but can also be used for bulk and interrupt transfers.
- DMA request mode 1 is mainly valuable for bulk transfers, where typically a large block of data is split into a series of packets of the maximum size.

24.1.4.4.2.1 DMA Request Mode 0

For RX endpoints, the DMA transfer is initiated when a data packet is available in the endpoint FIFO. The appropriate endpoint interrupt is also generated after the packets are received.

For TX endpoints, the DMA transfer is initiated when the endpoint FIFO can accept a data packet. The appropriate endpoint interrupt is also generated to prompt the loading of the packets.

24.1.4.4.2.2 DMA Request Mode 1

For RX endpoints the DMA transfer is initiated only when the received packet is the maximum packet size (as set in the RXMAXP register for the corresponding endpoint). The appropriate endpoint interrupt is generated only when the received packet is a short packet (that is, one less than the maximum packet size); otherwise, it is suppressed.

For TX endpoints, the DMA transfer is initiated when the endpoint FIFO can accept a data packet. The appropriate endpoint interrupt is suppressed.

24.1.4.4.2.3 Internal DMA Controller

The high-speed USB controller has an internal DMA controller to perform DMA transfers acting as host on the L3 interconnect. The DMA controller has eight channels, which can be independently programmed based on the following configuration options:

- DMA enable
- Direction (DMA write/DMA read)
- DMA mode (transfer one or more packets)
- Interrupt enable
- Endpoint number
- L3 interconnect memory address (32 bits)
- Byte count

The DMA controller uses burst transfers on the L3 interconnect. DMA transfers can be 8-bit, 16-bit, or 32-bit as required. However, all transfers associated with one packet (except the last packet) must be of the same width so that the data is consistently byte-, word-, or double-word-aligned. The last transfer can contain fewer bytes than the previous transfer to complete an odd-byte or odd-word transfer.

24.1.4.5 Automatic Packet Splitting/Combining for Bulk Transfers

The high-speed USB controller offers the facility for bulk endpoints to store larger amounts of data in their FIFOs than can be transferred in a single USB operation. In other words, the module includes a configuration option that, if selected, allows larger data packets to be written to bulk TX endpoints, which are then split into packets of an appropriate size for transfer across the USB. A similar option exists for bulk RX endpoints, which, if selected, causes the module to combine the packet received across the USB into larger data packets before being read by the application software.

The firmware can enable these options by setting the corresponding bits (MPRXE and MPTXE) in the CONFIGDATA register. The necessary packet size information contains the payload size for one transaction and a multiplier defining the maximum number of USB packets. The payload is required to be either 8, 16, 32, 64, or (in case of high-speed transfers) 512 bytes; the multiplier can be any value up to 32.

24.1.4.6 High-Bandwidth Isochronous Endpoints

In the high-speed mode, isochronous endpoints can transfer up to three USB packets in any microframe with a payload of up to 1024 bytes in each packet, corresponding to a data transfer rate of up to 3072 bytes per microframe (see *Universal Serial Bus Specification Revision 2.0*).

For TX endpoints, the high-speed USB controller supports this by allowing loading data packets of up to 3072 bytes (that is, 3 x 1024 bytes) into the associated endpoint FIFO, which is then automatically split into USB packets of the maximum payload, or smaller to be transmitted in one microframe.

For RX endpoints, the module automatically combines all the USB packets received during a microframe into a single packet of up to 3072 bytes (that is, 3 x 1024 bytes) within the RX FIFO.

The number of USB packets transferred per microframe and the maximum payload in each packet is defined through the appropriate registers (RXMAXP and TXMAXP).

24.1.5 High-Speed USB OTG Controller Basic Programming Model

This section describes only the TI-specific programming model details.

24.1.5.1 High-Speed USB Controller Interface Selection

The TI High-Speed USB controller supports only the 12-pin/8-bit data version ULPI interface (see Table 24-15).

The 8-pin, 4-bit version ULPI interface and 8-bit UTMI+ Level 3 interface are not supported.

The 12-pin/8-bit data SDR ULPI interface is selected through the USBOTG.OTG_INTERFSEL[1:0] PHYSEL field set to 0x1.

24.1.5.2 Enable Simulation Acceleration Features

The USBOTG.OTG_SIMENABLE register is dedicated solely for simulation (see Table 24-17) and is used to reduce timer length and, hence, the time taken for the test bench to run.

This special mode can be active only during simulation and must be disabled for normal operation. The firmware must keep TM1 inactive (reset value) during normal operation. An accidental write to this register can cause a malfunction.

24.1.5.3 Enabling MSTANDBY in Force-Standby Mode

The USBOTG.OTG_FORCESTDBY[0] ENABLEFORCE bit controls MSTANDBY behavior in force-standby mode only (see Table 24-19). In this mode, only when the internal core is idle (the module has no activity; that is, the USB is in suspend state) and when 1 is written to this bit, MSTANDBY goes high. Similarly, when ENABLEFORCE is 0 and the internal core is also in a non-idle state, MSTANDBY is deasserted. This bit does not influence MSTANDBY behavior in all other modes, such as no-standby and smart-standby.

24.1.5.4 Power Management Basic Programming Model

This section describes the settings for optimal High-Speed USB controller power management, depending on the use of this module in the application.

Two registers are involved in power management: USBOTG.OTG_SYSCONFIG and USBOTG.OTG_FORCESTDBY.

On reset, the high-speed USB controller has the following configuration:

- Master interface power management is in force-standby mode (USBOTG.OTG_SYSCONFIG[13:12] MIDLEMODE field = 0x0).
- Slave interface power management is in force-idle mode (USBOTG.OTG_SYSCONFIG[4:3] SIDLEMODE field = 0x0).
- Internal clock autogating feature is disabled (USBOTG.OTG_SYSCONFIG[0] AUTOIDLE bit = 0).

- MSTANDBY signal assertion is enabled (USBOTG.OTG_FORCESTDBY[0] ENABLEFORCE bit = 1).

24.1.5.4.1 High-Speed USB Controller Not Used for Application

In this scenario, the high-speed USB controller is not used by the system software. Default settings must be changed to reduce power consumption. Enabling the internal clock autogating feature cuts off the module internal interface clock as soon as it is no longer required. This is done by setting the USBOTG.OTG_SYSCONFIG[0] AUTOIDLE bit to 1.

The optimal configuration when the high-speed USB controller is not used by the application is as follows:

- Master interface power management is in force-standby mode (USBOTG.OTG_SYSCONFIG[13:12] MIDLEMODE field = 0x0).
- Slave interface power management is in force-idle mode (USBOTG.OTG_SYSCONFIG[4:3] SIDLEMODE field = 0x0).
- Internal clock autogating feature is enabled (USBOTG.OTG_SYSCONFIG[0] AUTOIDLE bit = 1).
- MSTANDBY signal assertion enabled (USBOTG.OTG_FORCESTDBY[0] ENABLEFORCE bit = 1)

24.1.5.4.2 High-Speed USB Controller in Host Mode

When used as a host, the high-speed USB controller must be programmed as follows:

- Master interface power management in smart-standby mode
- Slave interface power management in smart-idle mode
- Internal clock autogating feature enabled
- MSTANDBY signal assertion disabled

The programming sequence must be as follows:

1. Write 0 to the USBOTG.OTG_FORCESTDBY[0] ENABLEFORCE bit to disable the MSTANDBY assertion before programming to smart-standby and smart-idle modes.
2. Set the USBOTG.OTG_SYSCONFIG[13:12] MIDLEMODE field to 0x2, the USBOTG.OTG_SYSCONFIG[4:3] SIDLEMODE field to 0x2, and the USBOTG.OTG_SYSCONFIG[0] AUTOIDLE bit to 0 to program the smart-standby and smart-idle modes. Ensure that internal clock autogating is not enabled while programming smart-idle mode.
3. Set the USBOTG.OTG_SYSCONFIG[0] AUTOIDLE bit to 1 to cut off the internal clocks to save power.

Note: The internal clock autogating feature and smart-idle mode must not be programmed simultaneously.

24.1.5.4.3 High-Speed USB Controller in Peripheral Mode

When used as a peripheral, the high-speed USB controller must be programmed as follows:

- Master interface power management in smart-standby mode
- Slave interface power management in smart-idle mode
- Internal clock autogating feature enabled
- MSTANDBY signal assertion disabled

The programming sequence must be as follows:

1. Write 0 to the USBOTG.OTG_FORCESTDBY[0] ENABLEFORCE bit to disable the MSTANDBY assertion before programming to smart-standby mode.
2. Set the USBOTG.OTG_SYSCONFIG[13:12] MIDLEMODE field to 0x2, the USBOTG.OTG_SYSCONFIG[4:3] SIDLEMODE field to 0x2, and the USBOTG.OTG_SYSCONFIG[0] AUTOIDLE bit to 0 to program the smart-standby and smart-idle modes. Ensure that the internal clock autogating is not enabled while programming the smart-idle mode..
3. Set the USBOTG.OTG_SYSCONFIG[0] AUTOIDLE bit to 1 to cut off the internal clocks to save power.

Note: The internal clock autogating feature and smart-idle mode must not be programmed simultaneously.

24.1.5.4.4 High-Speed USB Controller in Host/Peripheral Mode

When used as a host/peripheral, the high-speed USB controller must be programmed as follows:

- Master interface power management in smart-standby mode
- Slave interface power management in smart-idle mode
- Internal clock autogating feature enabled
- MSTANDBY signal assertion disabled

See Section 24.1.5.4.2 and Section 24.1.5.4.3 for the programming sequence.

As an application required to disable the master interface, the high-speed USB controller can also be programmed as follows:

- Master interface power management in force-standby mode
- Slave interface power management in smart-idle mode
- Internal clock autogating feature enabled
- MSTANDBY signal assertion enabled

The programming sequence must be as follows:

1. Write 0 to the USBOTG.OTG_FORCESTDBY[0] ENABLEFORCE bit to disable the MSTANDBY assertion before programming to smart-standby mode.
2. Set the USBOTG.OTG_SYSCONFIG[13:12] MIDLEMODE field to 0x0, the USBOTG.OTG_SYSCONFIG[4:3] SIDLEMODE field to 0x2, and the USBOTG.OTG_SYSCONFIG[0] AUTOIDLE bit to 0 to program the force-standby and smart-idle modes. Ensure that the internal clock autogating is not enabled while programming smart-idle mode.
3. Set the USBOTG.OTG_FORCESTDBY[0] ENABLEFORCE bit to 1 to enable the MSTANDBY assertion before enabling internal clock autogating feature.
4. Set the USBOTG.OTG_SYSCONFIG[0] AUTOIDLE bit to 1 to cut off the internal clocks to save power.

Note: The internal clock the autogating feature and smart-idle mode must not be programmed simultaneously.

24.1.6 High-Speed USB OTG Controller Register Manual

As this chapter describes only the TI-specific details, this section presents only the TI-specific registers of the high-speed USB controller.

Table 24-7 lists the base address and address space for the high-speed USB controller.

Table 24-7. High-Speed USB Controller Instance Summary

Module Name	Base Address	Size
USBHS	0x480A B000	4K bytes

CAUTION

The high-speed USB registers are limited to 32-bit data accesses; 16-bit and 8-bit accesses are not allowed and can corrupt register content.

24.1.6.1 High-Speed USB Register Mapping Summary

Table 24-8 lists the TI-specific High-Speed USB controller registers.

Table 24-8. TI-Specific High-Speed USB Controller Register Summary

Register Name	Type	Register Width (Bits)	Address Offset	USBHS Physical Address
OTG_REVISION	R	32	0x400	0x480A B400
OTG_SYSCONFIG	RW	32	0x404	0x480A B404
OTG_SYSSTATUS	R	32	0x408	0x480A B408
OTG_INTERFSEL	RW	32	0x40C	0x480A B40C
OTG_SIMENABLE	RW	32	0x410	0x480A B410
OTG_FORCESTDBY	RW	32	0x414	0x480A B414

24.1.6.2 Register Description

Table 24-9 through Table 24-19 describe the register bits.

Table 24-9. OTG_REVISION

Address Offset	0x400		
Physical Address	0x480A B400	**Instance**	USBHS
Description	Standard USB OTG HS core revision number		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																								OTG_REVISION							

Bits	Field Name	Description	Type	Reset
31:8	Reserved	Reserved	R	0x000000
7:0	OTG_REVISION	Revision number, BCD-encoded	R	See (1)

(1) TI internal data

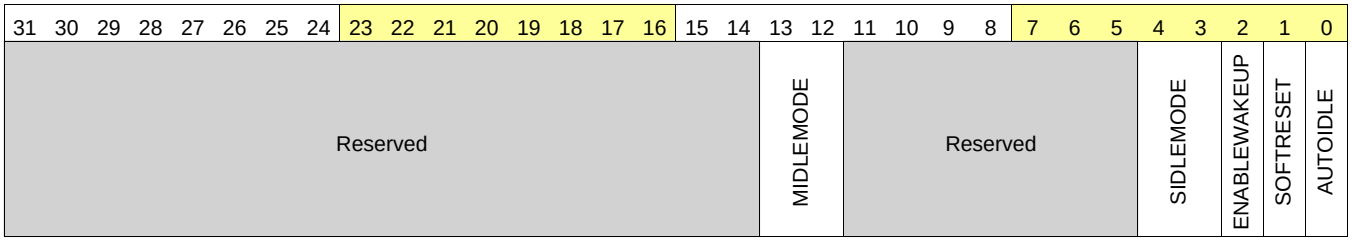
Table 24-10. Register Call Summary for Register OTG_REVISION

High-Speed USB OTG Controller

- High-Speed USB Register Mapping Summary: [0]
- Register Description: [1] [2]

Table 24-11. OTG_SYSCONFIG

Address Offset	0x404		
Physical Address	0x480A B404	**Instance**	USBHS
Description	Standard configuration		
Type	RW		



Bits	Field Name	Description	Type	Reset
31:14	Reserved	Reserved	R	0x00000
13:12	MIDLEMODE	Master interface power management control. Standby/wait control	RW	0x0
		0x0: Force Standby mode. Mstandby asserted unconditionally.		
		0x1: No standby mode. Mstandby never asserted.		
		0x2: Smart standby mode. Mstandby asserted when no more activity on the USB master.		
11:5	Reserved	Reserved	R	0x00
4:3	SIDLEMODE	Slave interface power management control. Req/ack control	RW	0x0
		0x0: Force-idle mode. Sidleack asserted after Midlereq assertion		
		0x1: No-idle mode. Sidleack never asserted.		
		0x2: Smart-Idle mode. Sidleack asserted after Midlereq assertion when no more activity on the USB.		
2	ENABLEWAKEUP	Enable wakeup capability	RW	0
		0x0: Wakeup is disabled.		
		0x1: Wakeup is enabled.		
1	SOFTRESET	Software reset bit	RW	0
		0x1: Starts softreset sequence.		
0	AUTOIDLE	Autoidle bit	RW	1
		0x0: Clock is always running.		
		0x1: When no activity on L3 interconnect, clock is cut off.		

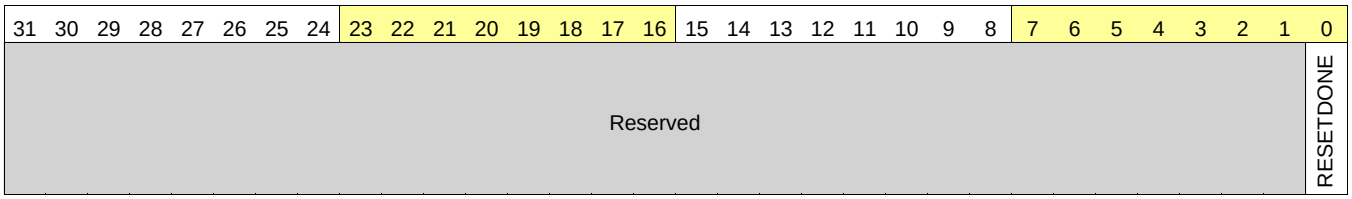
Table 24-12. Register Call Summary for Register OTG_SYSCONFIG

High-Speed USB OTG Controller

- Clocking, Reset, and Power-Management Scheme: [0] [1] [2] [3] [4] [5] [6] [7] [8] [9] [10] [11] [12]
- Basic Operation: [13] [14]
- Power Management Basic Programming Model: [15] [16] [17] [18] [19] [20] [21] [22] [23] [24] [25] [26] [27] [28] [29] [30] [31] [32] [33] [34]
- High-Speed USB Register Mapping Summary: [35]

Table 24-13. OTG_SYSSTATUS

Address Offset	0x408		
Physical Address	0x480A B408	**Instance**	USBHS
Description	Standard status		
Type	R		



Bits	Field Name	Description		Type	Reset
31:1	Reserved	Reserved		R	0x00000000
0	RESETDONE	Reset done		R	1
		Read 0x0:	Ongoing reset		
		Read 0x1:	Reset is finished.		

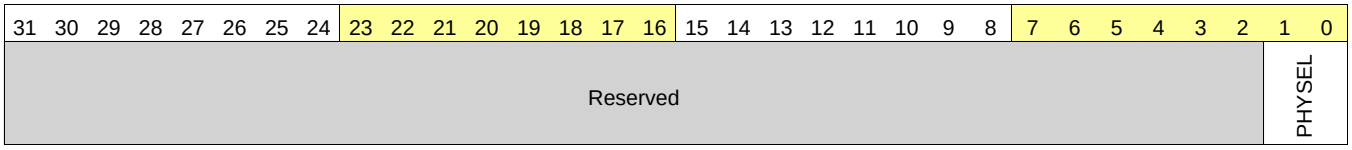
Table 24-14. Register Call Summary for Register OTG_SYSSTATUS

High-Speed USB OTG Controller

- High-Speed USB Register Mapping Summary: [0]

Table 24-15. OTG_INTERFSEL

Address Offset	0x40C		
Physical Address	0x480A B40C	**Instance**	USBHS
Description	USB OTG HS interface selection		
Type	RW		



Bits	Field Name	Description		Type	Reset
31:2	Reserved	Reserved		RW	0x00000000
1:0	PHYSEL	PHY interface selection		RW	0x1
		0x0:	PHY interface is 8-bit, UMTI+ Level 3. Not supported in OMAP34xx.		
		0x1:	PHY interface is 12-pin, 8-bit SDR ULPI.		
		0x2:	PHY interface is 8-pin, 4-bit DDR ULPI. Not supported in OMAP34xx.		

Table 24-16. Register Call Summary for Register OTG_INTERFSEL

High-Speed USB OTG Controller

- High-Speed USB OTG Controller Environment: [0]
- High-Speed USB Controller Interface Selection: [1]
- High-Speed USB Register Mapping Summary: [2]

Table 24-17. OTG_SIMENABLE

Address Offset	0x410		
Physical Address	0x480A B410	**Instance**	USBHS
Description	Enable simulation acceleration features. **WARNING: For simulations only, because those features have an impact on USB protocol.**		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																															TM1

Bits	Field Name	Description	Type	Reset
31:1	Reserved	Reserved	RW	0x00000000
0	TM1	Test Mode 1 enabling (timer shortcuts)	RW	0

Table 24-18. Register Call Summary for Register OTG_SIMENABLE

High-Speed USB OTG Controller

- Enable Simulation Acceleration Features: [0]
- High-Speed USB Register Mapping Summary: [1]

Table 24-19. OTG_FORCESTDBY

Address Offset	0x414		
Physical Address	0x480A B414	**Instance**	USBHS
Description	Enabling MSTANDBY in FORCESTANDBY mode.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																															ENABLEFORCE

Bits	Field Name	Description	Type	Reset
31:1	Reserved	Reserved	RW	0x00000000
0	ENABLEFORCE	Enabling MSTANDBY to go high	RW	1

Table 24-20. Register Call Summary for Register OTG_FORCESTDBY

High-Speed USB OTG Controller

- Clocking, Reset, and Power-Management Scheme: [0] [1] [2] [3] [4] [5]
- Basic Operation: [6]
- Enabling MSTANDBY in Force-Standby Mode: [7]
- Power Management Basic Programming Model: [8] [9] [10] [11] [12] [13] [14]
- High-Speed USB Register Mapping Summary: [15]

24.2 High-Speed USB Host Subsystem

Note: Copyright ©2004,2005, 2006, 2007, 2008 Synopsys, Inc. All rights reserved. Used with permission.

24.2.1 High-Speed USB Host Subsystem Overview

The high-speed universal serial bus (USB) host subsystem is composed of the high-speed multiport USB host controller and the USBTLL module.

The USB controller is a high-speed multiport USB2.0 host controller. It contains two independent, 3-port host controllers that operate in parallel:

- The EHCI controller, based on the *Enhanced Host Controller Interface (EHCI) specification for USB Release 1.0*, is in charge of high-speed traffic (480M bit/s), over the ULPI/UTMI interface
- The OHCI controller, based on the *Open Host Controller Interface (OHCI) specification for USB Release 1.0a*, is in charge of full-speed/low-speed traffic (12/1.5M bit/s, respectively), over a serial interface

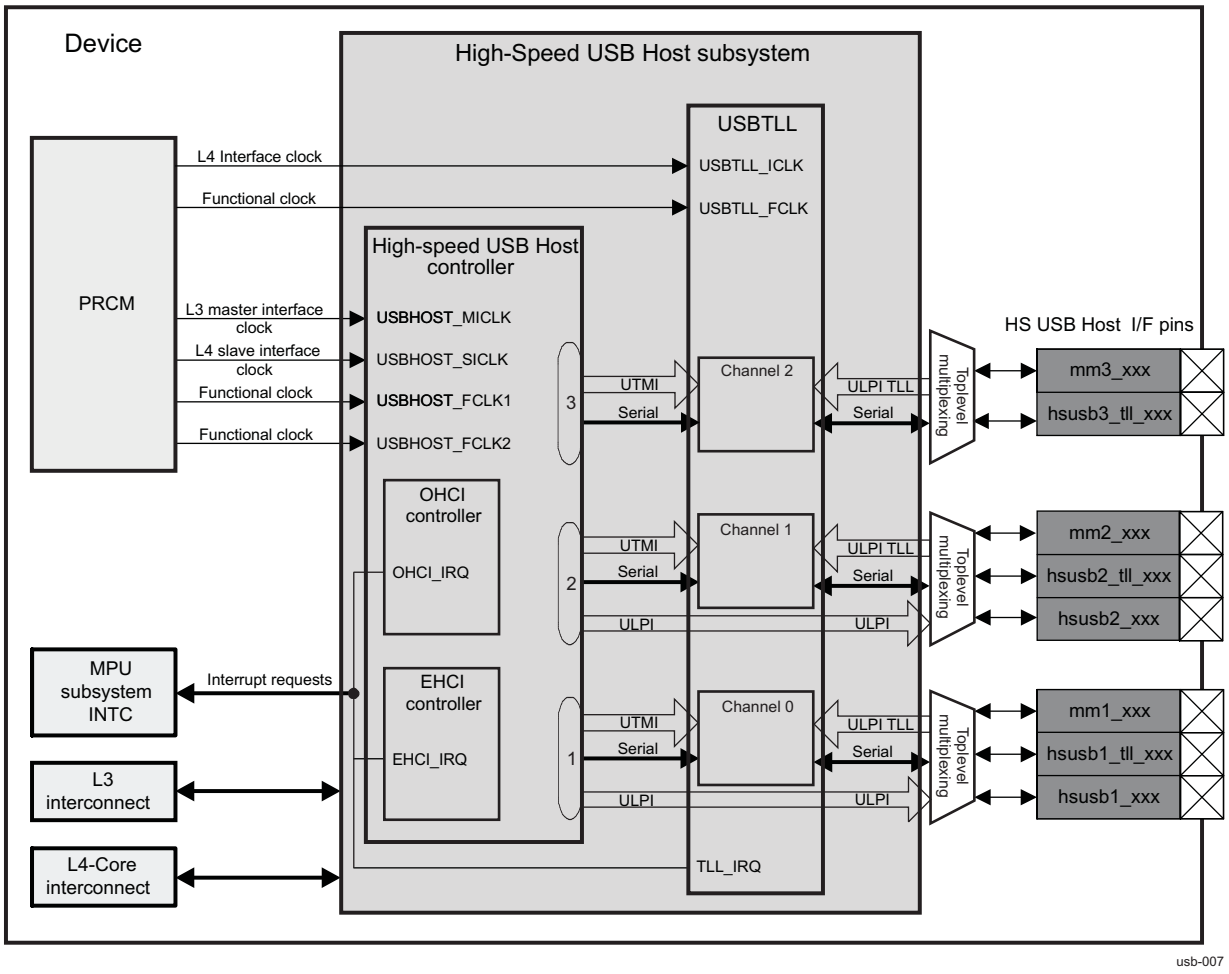
Each of the three external ports is owned by exactly one of the controllers at any time.

The USBTLL module is a high-speed USB UTMI low-pin interface (ULPI) transceiverless link logic (TLL) adapter. It implements a TLL compatible with a number of USB standard interface protocols. It consists of three channels, defined as independent USB path through the TLL module, which always converts the UTMI+ PHY interface protocol coming from the high-speed USB host controller.

Each USB port (1, 2, and 3) can connect either to an external-to-OMAP chip USB transceiver or directly using a transceiverless link to an external integrated circuit (IC) supporting the same TLL protocol.

Figure 24-7 highlights the high-speed USB host subsystem.

Figure 24-7. High-Speed USB Host Subsystem Highlight



24.2.1.1 Main Features

The high-speed USB host subsystem includes the following features:

- Multiport high-speed USB host controller:
  - Complies with the USB 2.0 standard for high-speed (480 Mbps) functions
  - USB 2.0 low-speed (1.5M bit/s), full-speed (12M bit/s), and high-speed (480M bit/s) operations
  - Three downstream ports (3-port root hub)
  - Complies with EHCI (high-speed host controller)
  - Complies with OHCI (low-speed/full-speed host controller)
  - Supports suspend/resume and remote wakeup
  - Interface with USBTLL port A on all ports (UTMI+)
    - 8-bit datapath
    - 60-MHz synchronous (on-chip) interface
  - Interface with ULPI PHYs (transceivers) on all ports (only two mapped)
    - 12-pin/8-bit data single data rate (SDR) mode
    - 60-MHz clock, generated by the host: ULPI "input" clocking mode

CAUTION

The HS USB host subsystem can only support the external charge pump of PHY (no support of internal charge pump for ULPI PHY)

  - Hardware-driven save-and-restore of the suspended host hardware context
  - Two interrupt lines
- USBTLL module
  - Three channels
  - Three ports (A, C, and D) by channel
  - Port A: PHY-side UTMI+ port. Connects to the local link controller. The UTMI “local” port is used in all configurations, (that is, the entire channel can be seen as a protocol converted from that port to one of the other, “remote” ports).
    - Compliant with UTMI+ (USB 2.0 Transceiver Macrocell Interface) version 1.0
    - 8-data-bit, 60-MHz UTMI (HS/FS/LS-capable)
    - UTMI+ Level 3 extensions
    - Vcontrol/Vstatus (from UTMI)
    - Serial FS/LS “6-pin” mode
  - Port C: PHY-side ULPI port. Connects to a remote (off-chip) ULPI link controller through I/O pads.
    - SDR and dual data rate (DDR) ULPI capable (resp 8/4-bit data width modes)
    - Supports optional 6-pin/3-pin serial modes
    - Supports optional input clocking mode
  - Port D: Serial multimode port. Connects to either a serial link controller (TLL modes) or a serial PHY (PHY interface modes).
    - Supports 6-pin unidirectional, 4-pin bidirectional, 3-pin bidirectional, 2-pin bidirectional modes
    - All modes are supported for TLL or PHY interface configuration.
    - Supports sideband signals (pullup/down control, speed/suspend enable, etc)
  - An interrupt line
- USB port signal pins interface supporting:
  - External USB transceivers
    - ULPI interface: 12-pin/8-bit data SDR version
    - Serial 6-pin PHY (transceiver) interfaces: 6-pin mode (TX: DAT/SE0 or TX: DP/DM unidirectional mode), 4-pin mode (DP/DM bidirectional mode), and 3-pin mode (DAT/SE0 bidirectional mode)
  - TLL mode
    - ULPI TLL interfaces: 12-pin/8-bit data SDR and 8-pin/4-bit data DDR versions
    - Serial 6-pin TLL interfaces: 6-pin mode (DAT/SE0 and DP/DM unidirectional modes), 4-pin mode, (DP/DM bidirectional mode), 3-pin mode (DAT/SE0 bidirectional mode) and 2-pin mode (DAT/SE0 and DP/DM bidirectional modes)

Table 24-21. USB Connectivity Modes

USB Connectivity Modes	Port 1	Port 2	Port 3
ULPI interface	√	√	
Serial 6-pin transceiver interfaces	√	√	√
ULPI TLL interfaces	√	√	√
Serial 6-pin TLL interfaces	√	√	√

Note: Only FS/LS USB transceivers can be connected to port 3; HS ULPI transceivers are not supported on port 3.

24.2.2 High-Speed USB Host Subsystem Environment

The high-speed USB host controller provides two kinds of interfaces for connection:

- ULPI interfaces for high-speed data transactions (up to 480M bit/s)
- Serial interfaces (with the use of the USBTLL module) for full- and low-speed data transactions (up to 12M bit/s)

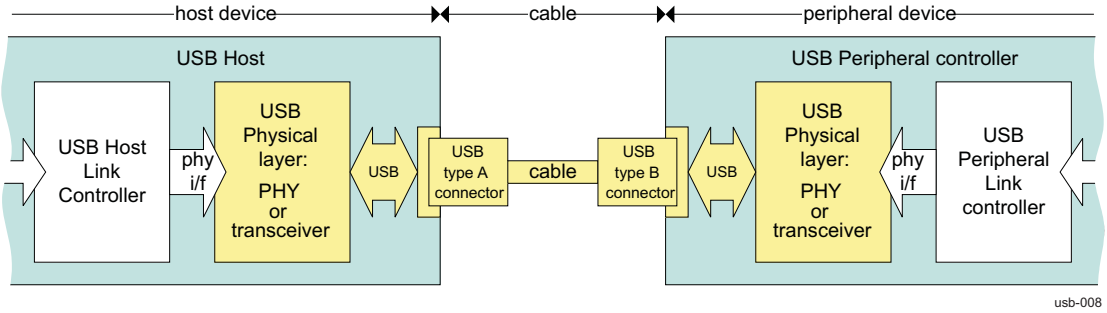
The high-speed USB host controller connects to either controllers (TLL modes) and/or transceivers. It supports the following configurations with the serial interfaces and ULPI interfaces:

- External USB transceiver configurations
  - ULPI interface: 12-pin/8-bit data SDR version
  - Serial 6-pin PHY (transceiver) interfaces: 6-pin mode (TX: DAT/SE0 or TX: DP/DM unidirectional mode), 4-pin mode (DP/DM bidirectional mode) and 3-pin mode (DAT/SE0 bidirectional mode)
- TLL configurations
  - ULPI TLL interfaces: 12-pin/8-bit data SDR and 8-pin/4-bit data DDR versions
  - Serial 6-pin TLL interfaces: 6-pin mode (DAT/SE0 and DP/DM unidirectional modes), 4-pin mode (DP/DM bidirectional mode), 3-pin mode (DAT/SE0 bidirectional mode), and 2-pin mode (DAT/SE0 and DP/DM bidirectional modes)

24.2.2.1 Standard USB Implementation: Transceiver Connection

From a logical point of view, a point-to-point USB connection is composed of several blocks, organized in protocol layers, and shown in Figure 24-8.

Figure 24-8. USB Connection



The host system (USB master) and the peripheral system (USB slave) connected through the USB cable include a link or controller (link layer) and a PHY or transceiver (physical layer). Each system talks to its own controller, which talks to its own transceiver, which is connected to the opposite side (transceiver) through an assembly of connectors, receptacles, and cable.

24.2.2.2 TLL Connection

The TLL feature enables connection of the high-speed USB host subsystem to an external, onboard USB peripheral controller, without using USB transceivers or associated circuitry. When TLL is used, the following components are removed:

- Both USB transceivers
- The series resistors
- Pullup and pulldown resistors
- VBUS switching components
- USB connectors and cables, typically used between a USB host controller and the downstream USB peripheral controller

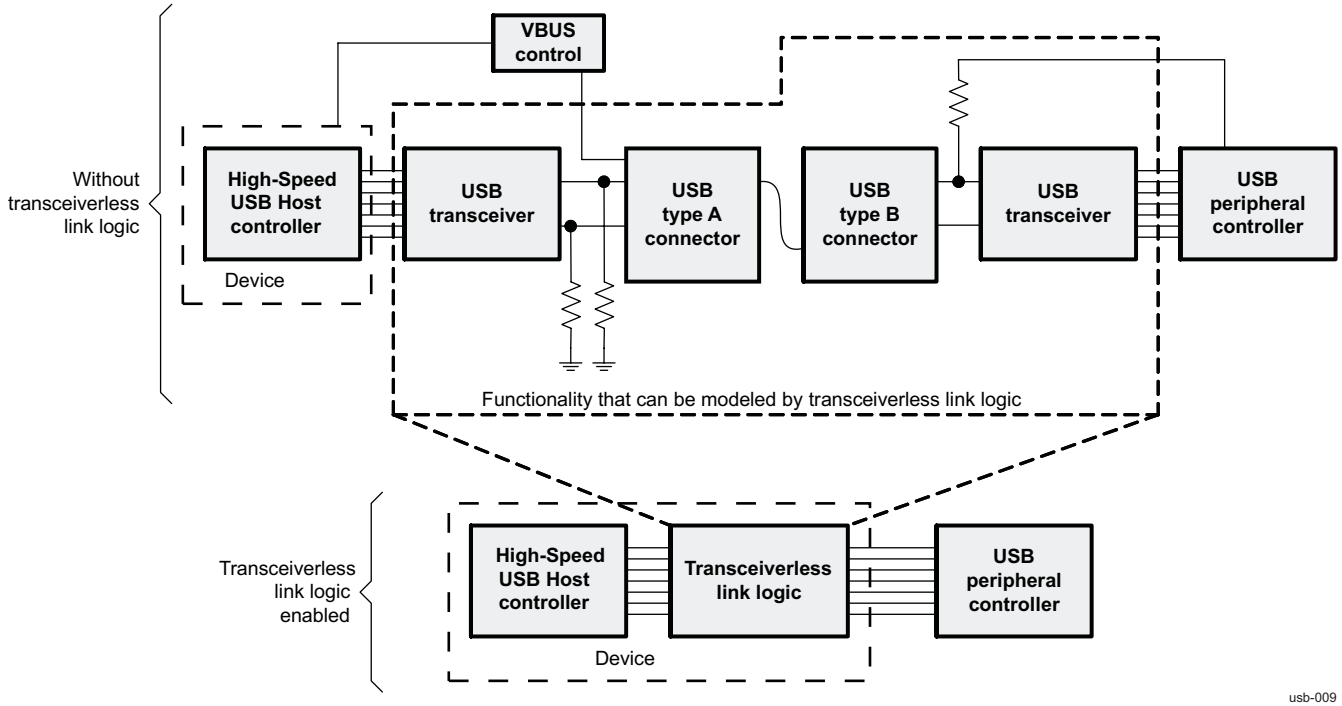
The TLL signaling system is not suitable for use across a cable. It is intended only for use when the device is used with an external USB integrated circuit (IC) that is on the same board.

When using the TLL, signals of the external USB IC pins, which typically connect to a USB transceiver, instead directly connect to the device pins. Signaling on these pins use CMOS levels. TLL logic can be used with external devices that support ULPI TLL interface and serial 6-pin TLL interface connectivity.

The TLL function in the device interprets the transmit control signals from the external USB IC and similar signals from the device USB host controller, and computes the equivalent USB differential-pair state. The computed differential-pair state is interpreted and the appropriate transceiver output signals are provided to the external USB IC and to the device USB host controller.

Figure 24-9 shows the device and how TLL can be compared to a typical USB implementation. The top portion of Figure 24-9 shows the a transceiver-based solution, and the bottom portion shows a transceiverless solution using TLL.

Figure 24-9. High-Speed USB Host Controller Connection—With and Without TLL



Note: The USB bus lines (D+/D–) no longer appear in subsequent figures: They are emulated by the TLL.

24.2.2.3 ULPI Interfaces

The high-speed USB host subsystem supports the following configurations with the ULPI interfaces:

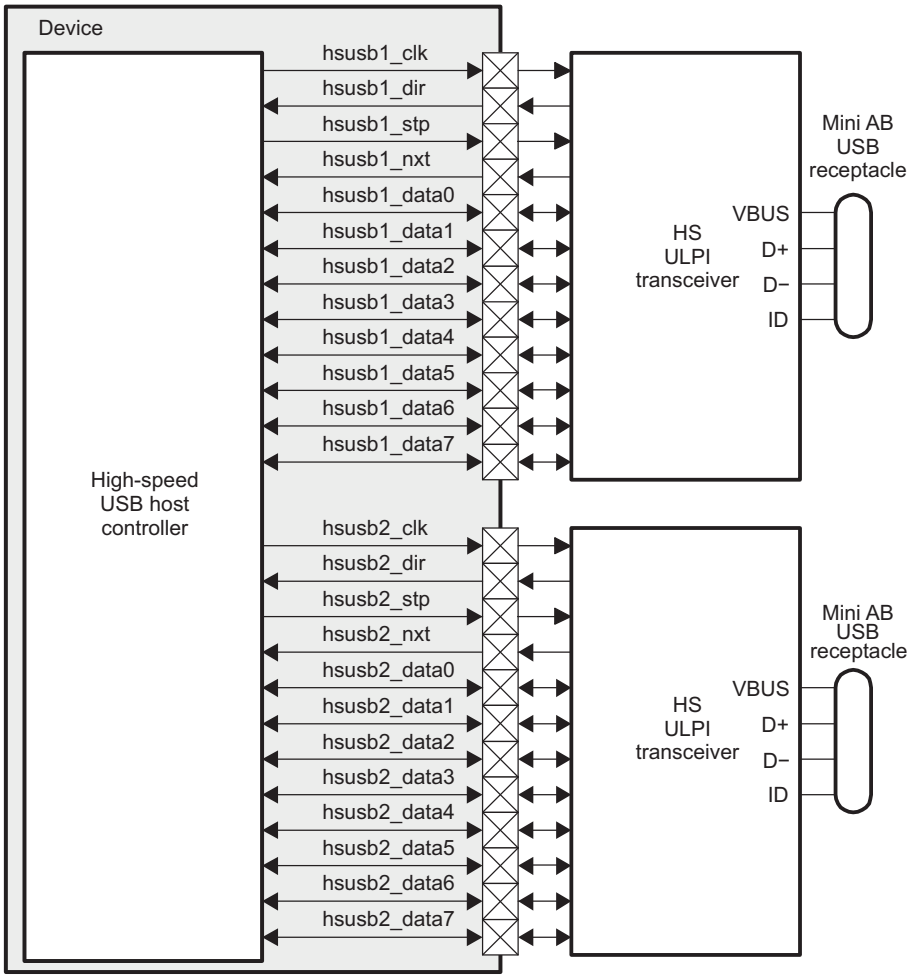
- External USB transceiver
  - ULPI interfaces: 12-pin/8-bit data SDR version
- TLL
  - ULPI TLL interfaces: 12-pin/8-bit data SDR and 8-pin/4-bit data DDR versions

The high-speed USB host subsystem supports USB ports, which use the ULPI interface mode to connect to an off-chip high-speed ULPI transceiver (12-pin/8-bit data SDR mode) for high- and full-speed data transactions (up to 480M bit/s). Using the ULPI interfaces in 12-pin/8-bit data version, the device and the transceiver achieve the USB function. A mini A-B external receptacle allows the connection of an external device.

The device supports TLL logic interfaces on its ports in the ULPI TLL interface mode. TLL modes enable glueless interconnect to another USB device port without a costly transceiver.

Figure 24-10 and Figure 24-11 show typical applications using the high-speed USB host subsystem with the ULPI, and the ULPI TLL, respectively.

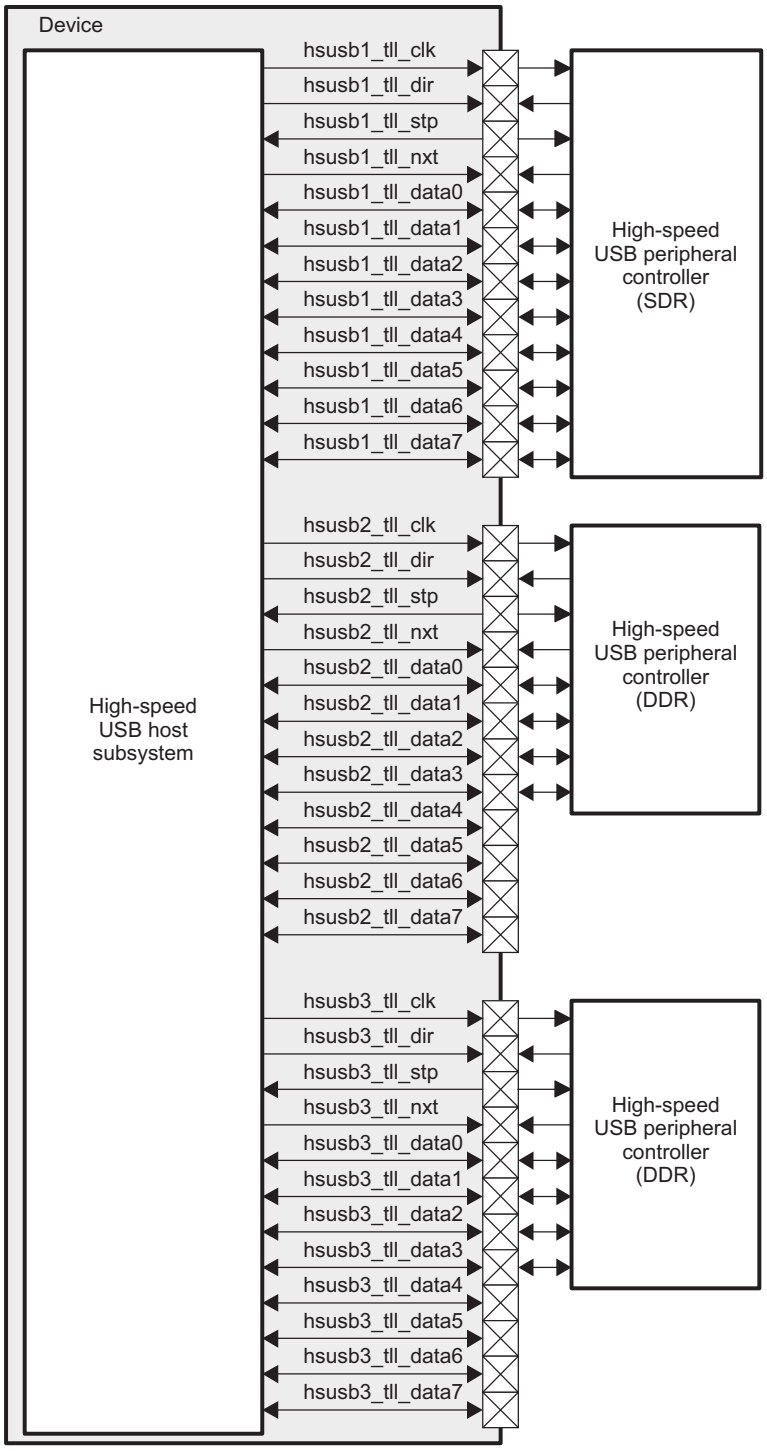
Figure 24-10. High-Speed USB Host Controller Typical Application System – ULPI Interfaces



ULPI Interfaces - USBTLL is bypassed and high-speed USB host controller ports 1 and 2 are connected directly to external transceivers

usb–010

Figure 24-11. High-Speed USB Host Subsystem Typical Application System - ULPI TLL Interfaces



ULPI TLL Interfaces -The high-speed USB host controller is coupled with the USBTLL module to compose the ULPITLL interface modes usb−032

The current implementation of the ULPI includes a method for manual, software-controlled generation of PHY-side register accesses. This implies support of the following features:

- Access to vendor-specific or optional PHY-side registers
- Access to vendor ID, product ID, and scratch and debug registers

The 12-pin ULPI interface uses an 8-bit data bus with data synchronous to the rising edge of the PHY (transceiver) clock (SDR mode), whereas the 8-pin ULPI uses a 4-bit data bus with data generated on both the rising and falling clock edges (DDR mode).

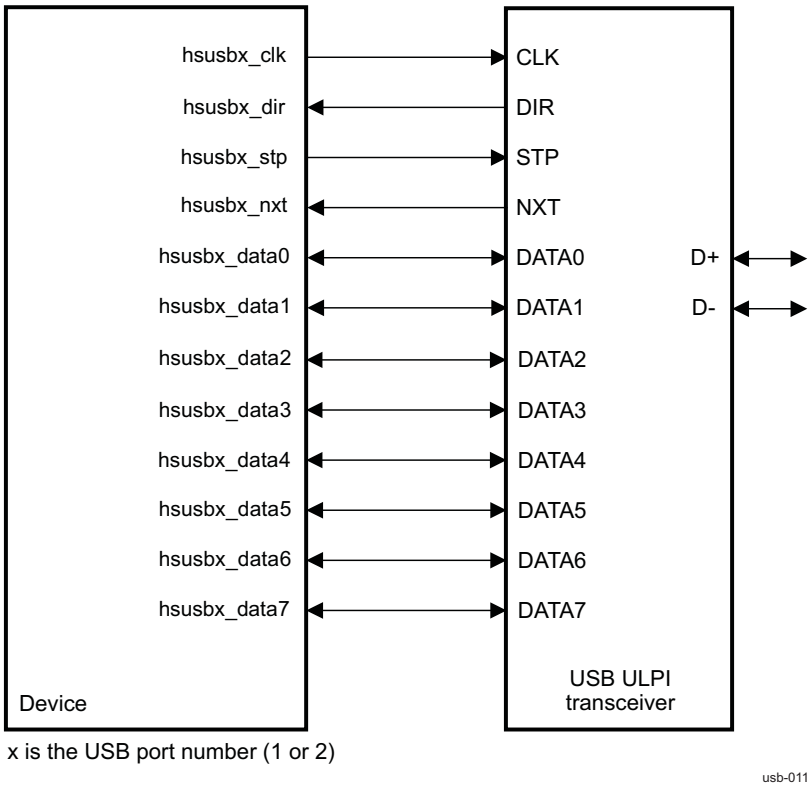
24.2.2.3.1 Transceiver Interface Configurations

The high-speed USB host subsystem supports only the 12-pin/8-bit data SDR version of the ULPI interface mode.

Note: In the device, only the ULPI ports 1 and 2 of the high-speed USB host controller are mapped and can be connected directly to external transceivers.

Figure 24-12 shows USB ports using the 12-pin/8-bit data SDR version of the ULPI interface mode.

Figure 24-12. ULPI Interfaces – 12-Pin/8-Bit Data SDR Version



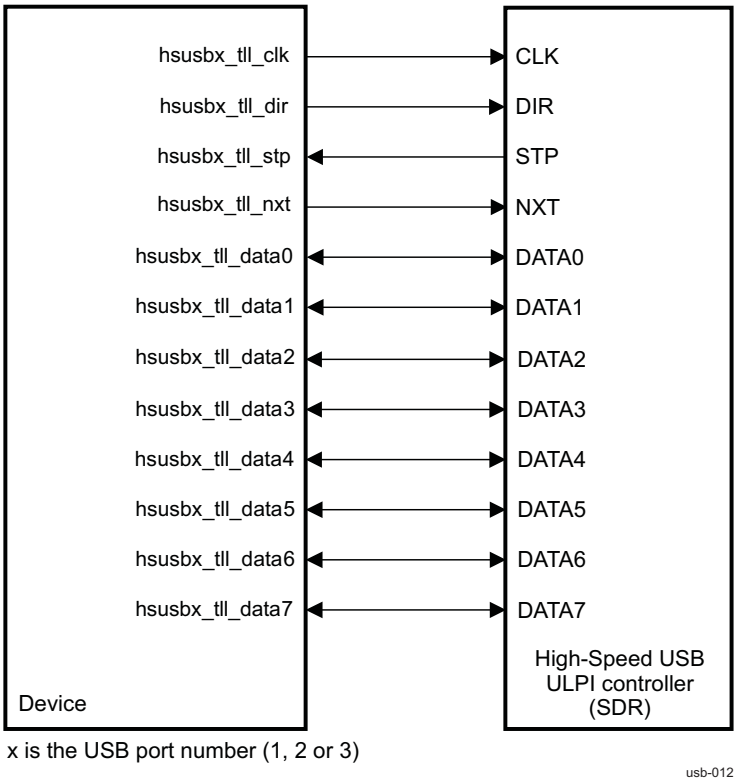
24.2.2.3.2 TLL Configurations

The high-speed USB host controller is coupled with the USBTLL module to compose the ULPI TLL interface modes.

The high-speed USB host subsystem supports the 12-pin/8-bit data SDR and 8-pin/4-bit data DDR versions of the ULPI TLL interface mode.

Figure 24-13 shows USB ports using the 12-pin/8-bit data SDR version of the ULPI TLL interface mode.

Figure 24-13. ULPI TLL Interfaces –12-Pin/8-Bit Data SDR Version

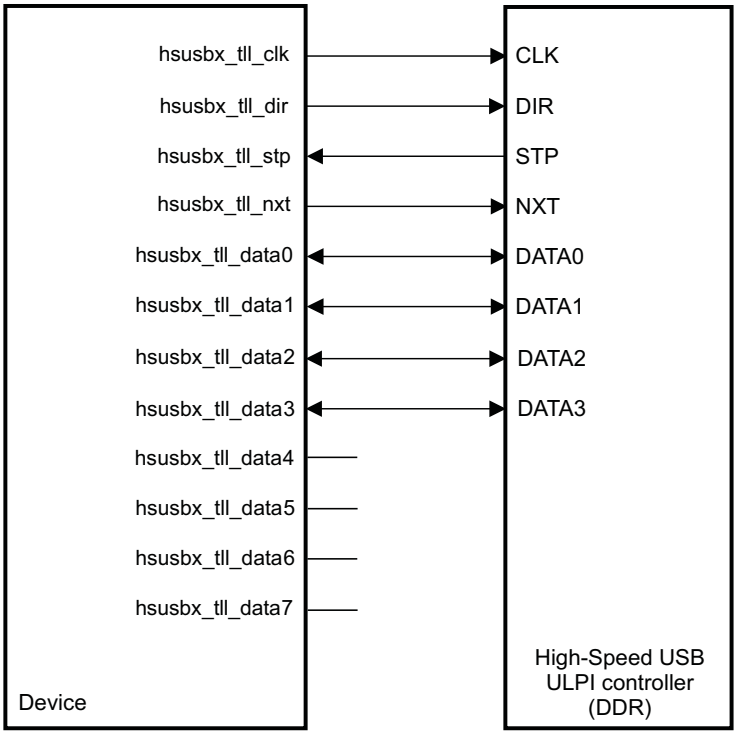


x is the USB port number (1, 2 or 3)

usb-012

Figure 24-14 shows USB ports using the 8-pin/4-bit data DDR version of the ULPI TLL interface mode.

Figure 24-14. ULPI TLL Interfaces – 8-Pin/4-Bit Data DDR Version



x is the USB port number (1, 2 or 3)

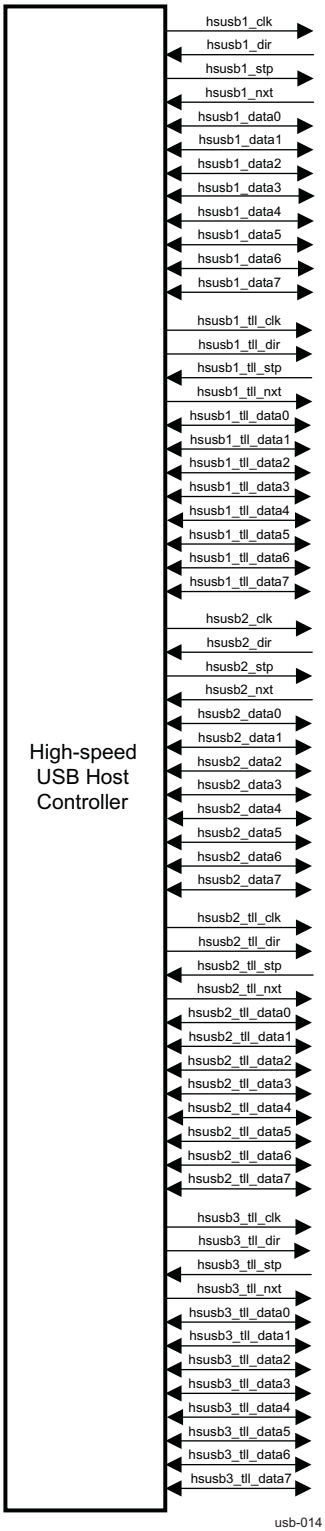
usb-013

24.2.2.3.3 High-Speed USB Host Subsystem Functional Interfaces

24.2.2.3.3.1 Basic High-Speed USB Host Subsystem Pins

Figure 24-15 shows the high-speed USB host controller ULPI functional interfaces.

Figure 24-15. High-Speed USB Host Subsystem Functional Interface Signals



24.2.2.3.3.2 High-Speed USB Host Subsystem Interface Description

Table 24-22 describes the I/O of the high-speed USB host subsystem ULPI interfaces.

Table 24-22. I/O Description

Signal Name	I/O[1]	Description	Reset Value
HSUSB1			
hsusb1_clk	O	60-MHz clock output to ULPI transceiver[2]	0
hsusb1_dir	I	Data direction control from ULPI transceiver	Unknown
hsusb1_stp	O	Stop signal to ULPI transceiver	1
hsusb1_nxt	I	Next signal from ULPI transceiver	Unknown
hsusb1_data0	I/O	Bidirectional DATA0	Unknown
hsusb1_data1	I/O	Bidirectional DATA1	Unknown
hsusb1_data2	I/O	Bidirectional DATA2	Unknown
hsusb1_data3	I/O	Bidirectional DATA3	Unknown
hsusb1_data4	I/O	Bidirectional DATA4	Unknown
hsusb1_data5	I/O	Bidirectional DATA5	Unknown
hsusb1_data6	I/O	Bidirectional DATA6	Unknown
hsusb1_data7	I/O	Bidirectional DATA7	Unknown
HSUSB1 TLL			
hsusb1_tll_clk	O	60-MHz clock output to ULPI transceiver[2]	0
hsusb1_tll_dir	O	Data direction control from ULPI transceiver	0
hsusb1_tll_stp	I	Stop signal to ULPI transceiver	Unknown
hsusb1_tll_nxt	O	Next signal from ULPI transceiver	0
hsusb1_tll_data0	I/O	Bidirectional DATA0	0
hsusb1_tll_data1	I/O	Bidirectional DATA1	0
hsusb1_tll_data2	I/O	Bidirectional DATA2	0
hsusb1_tll_data3	I/O	Bidirectional DATA3	0
hsusb1_tll_data4	I/O	Bidirectional DATA4	0
hsusb1_tll_data5	I/O	Bidirectional DATA5	0
hsusb1_tll_data6	I/O	Bidirectional DATA6	0
hsusb1_tll_data7	I/O	Bidirectional DATA7	0
HSUSB2			
hsusb2_clk	O	60-MHz clock output to ULPI transceiver[2]	0
hsusb2_dir	I	Data direction control from ULPI transceiver	Unknown
hsusb2_stp	O	Stop signal to ULPI transceiver	1
hsusb2_nxt	I	Next signal from ULPI transceiver	Unknown
hsusb2_data0	I/O	Bidirectional DATA0	Unknown
hsusb2_data1	I/O	Bidirectional DATA1	Unknown
hsusb2_data2	I/O	Bidirectional DATA2	Unknown
hsusb2_data3	I/O	Bidirectional DATA3	Unknown
hsusb2_data4	I/O	Bidirectional DATA4	Unknown
hsusb2_data5	I/O	Bidirectional DATA5	Unknown
hsusb2_data6	I/O	Bidirectional DATA6	Unknown
hsusb2_data7	I/O	Bidirectional DATA7	Unknown
HSUSB2 TLL			
hsusb2_tll_clk	O	60-MHz clock output to ULPI transceiver[2]	0
hsusb2_tll_dir	O	Data direction control from ULPI transceiver	0
hsusb2_tll_stp	I	Stop signal to ULPI transceiver	Unknown

(1) I = Input; O = Output
(2) This output signal is also used as re-timing input.

Table 24-22. I/O Description (continued)

Signal Name	I/O[1]	Description	Reset Value
hsusb2_tll_nxt	O	Next signal from ULPI transceiver	0
hsusb2_tll_data0	I/O	Bidirectional DATA0	0
hsusb2_tll_data1	I/O	Bidirectional DATA1	0
hsusb2_tll_data2	I/O	Bidirectional DATA2	0
hsusb2_tll_data3	I/O	Bidirectional DATA3	0
hsusb2_tll_data4	I/O	Bidirectional DATA4	0
hsusb2_tll_data5	I/O	Bidirectional DATA5	0
hsusb2_tll_data6	I/O	Bidirectional DATA6	0
hsusb2_tll_data7	I/O	Bidirectional DATA7	0
HSUSB3 TLL			
hsusb3_tll_clk	O	60-MHz clock output to ULPI transceiver[2]	0
hsusb3_tll_dir	O	Data direction control from ULPI transceiver	0
hsusb3_tll_stp	I	Stop signal to ULPI transceiver	Unknown
hsusb3_tll_nxt	O	Next signal from ULPI transceiver	0
hsusb3_tll_data0	I/O	Bidirectional DATA0	0
hsusb3_tll_data1	I/O	Bidirectional DATA1	0
hsusb3_tll_data2	I/O	Bidirectional DATA2	0
hsusb3_tll_data3	I/O	Bidirectional DATA3	0
hsusb3_tll_data4	I/O	Bidirectional DATA4	0
hsusb3_tll_data5	I/O	Bidirectional DATA5	0
hsusb3_tll_data6	I/O	Bidirectional DATA6	0
hsusb3_tll_data7	I/O	Bidirectional DATA7	0

ULPI (PHY) Interfaces and ULPI TLL Interfaces can not be used together: Either the ULPI (PHY) Interfaces or the ULPI TLL Interfaces are selected.

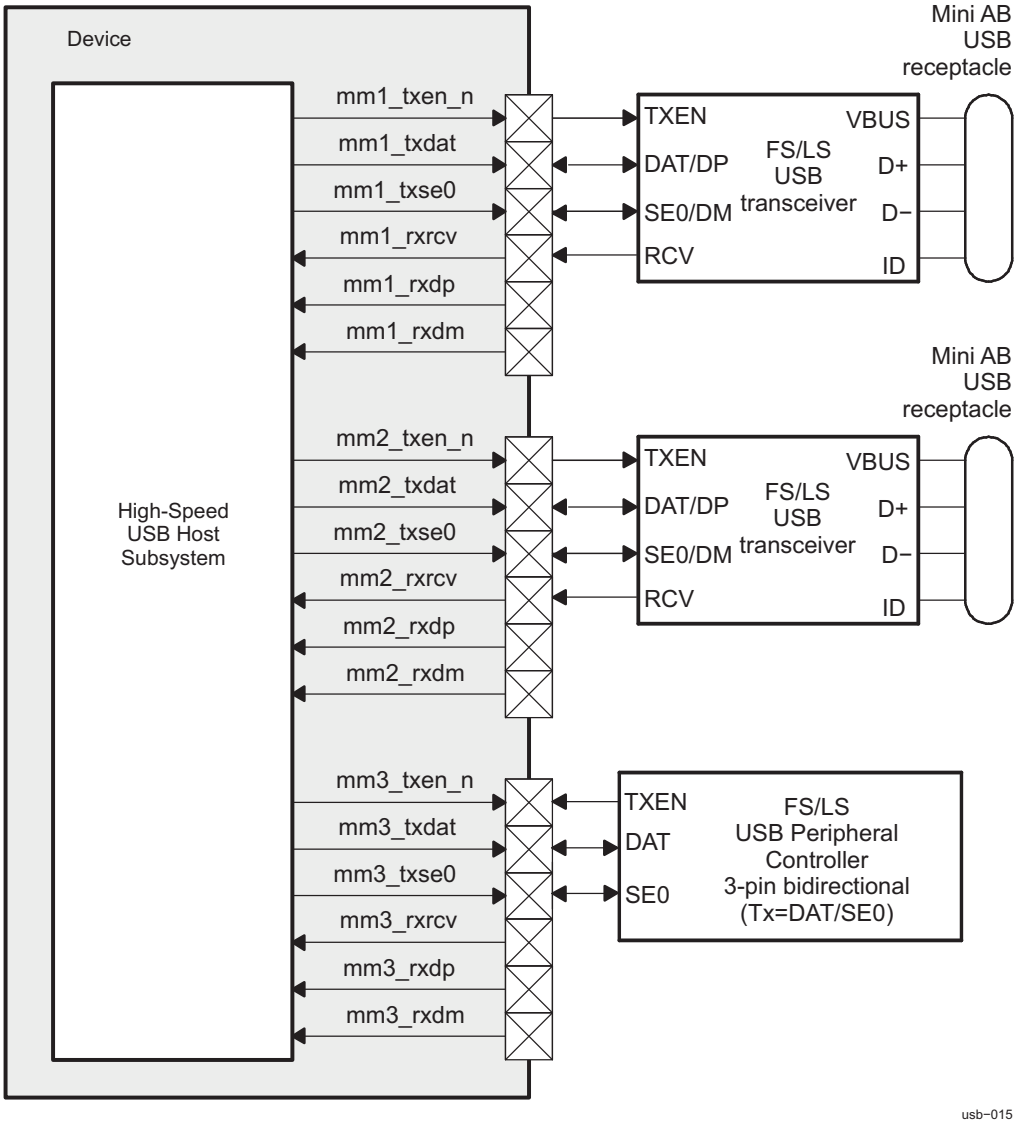
24.2.2.4 Serial Interfaces

The high-speed USB host subsystem supports the following configurations with the serial interfaces:

- External USB transceiver
  - Serial 6-pin PHY (transceiver) interfaces: 6-pin mode (TX: DAT/SE0 or TX: DP/DM unidirectional mode), 4-pin mode (DP/DM bidirectional mode), and 3-pin mode (DAT/SE0 bidirectional mode)
- TLL
  - Serial 6-pin TLL interfaces: 6-pin mode (DAT/SE0 and DP/DM unidirectional modes), 4-pin mode (DP/DM bidirectional mode), 3-pin mode (DAT/SE0 bidirectional mode), and 2-pin mode (DAT/SE0 and DP/DM bidirectional modes)

Figure 24-16 shows a typical application using the high-speed USB host subsystem with the serial interfaces.

Figure 24-16. High-Speed USB Host Subsystem Typical Application System



The high-speed USB host controller is coupled with the USBTLL module to compose the serial interface modes. The USBTLL module translates the parallel, synchronous UTMI+ (Level 3) protocol to a serial, asynchronous one. The benefit of the conversion is a simplified interface to the transceiver.

CAUTION

Only full- and low-speed data transactions are possible in serial mode. Transceiver interface is serial (its frequency is that of the actual USB line) and combinatorial (no clock is passed).

CAUTION

It is possible to change the polarity of the TXEN signal for each USB port using the following bits:

- CONTROL.CONTROL_WKUP_CTRL[2] M_FSUSB3_TXEN_N_OUT_POLARITY_CTRL
- CONTROL.CONTROL_WKUP_CTRL[1]MM_FSUSB3_TXEN_N_OUT_POLARITY_CTRL
- CONTROL.CONTROL_WKUP_CTRL[0] M_FSUSB3_TXEN_N_OUT_POLARITY_CTRL

Whether in TLL or transceiver configuration, the serial interface follows the same principles. It is limited to full/low speed, and that high speed requires a parallel interface.

24.2.2.4.1 Encoding in Serial Mode

24.2.2.4.1.1 Unidirectional

When a USB transceiver is connected to the device and used in 6-pin unidirectional DAT/SE0 encoding mode, the encoding described in Table 24-23 is used.

Table 24-23. Signaling Between High-Speed USB Host Subsystem and 6-Pin Unidirectional USB Transceiver (DAT/SE0 Signaling)

Logical Signal Name	Device Pin Direction	Transceiver Pin Direction	Description				
TXEN	Output	Input	When low or high (depending on the CONTROL.CONTROL_WKUP_CTRL[2, 1, and/or 0] MM_FSUSBi_TXEN_N_OUT_POLARITY_CTRL bit), the USB transceiver drives D+ and D-.				
DAT and SE0	Output	Input	Controls the values output by the USB transceiver on D+ and D- when TXEN is low/high; ignored when TXEN is high/low (depending on the TXEN signal polarity; see CONTROL.CONTROL_WKUP_CTRL[2, 1 and/or 0] MM_FSUSBi_TXEN_N_OUT_POLARITY_CTRL bit).				
			TXEN	DAT	SE0	D+	D-
			0	0	0	0	1
				1	0	1	0
				X	1	0	0
			1	X	X	Undriven	Undriven
RCV	Input	Output	Output from transceiver differential receiver				
			D+		D-		RCV
			0		0		X
			0		1		0
			1		0		1
			1		1		X
DP	Input	Output	Output from transceiver single-ended D+ signal receiver				
			D+				DP
			0				0
			1				1
DM	Input	Output	Output from transceiver single-ended D- signal receiver				
			D-				DM
			0				0
			1				1

When a USB transceiver is connected to the device and used in 6-pin unidirectional DP/DM encoding mode, the encoding described in Table 24-24 is used.

Table 24-24. Signaling Between High-Speed USB Host Subsystem and 6-Pin Unidirectional USB Transceiver (DP/DM Signaling)

Logical Signal Name	Device Pin Direction	Transceiver Pin Direction	Description
TXEN	Output	Input	When low or high (depending on the CONTROL.CONTROL_WKUP_CTRL[2, 1 and/or 0] MM_FSUSBi_TXEN_N_OUT_POLARITY_CTRL bit), the USB transceiver drives D+ and D-.
DAT and SE0	Output	Input	Controls the values output by the USB transceiver on D+ and D- when TXEN is low/high; ignored when TXEN is high/low (depending on the TXEN signal polarity; see the CONTROL.CONTROL_WKUP_CTRL[2, 1 and/or 0] MM_FSUSBi_TXEN_N_OUT_POLARITY_CTRL bit).

TXEN	DAT	SE0	D+	D-
0	0	0	0	1
	1	0	1	0
	X	1	0	0
1	X	X	Undriven	Undriven

RCV	Input	Output	Output from transceiver differential receiver

D+	D-	RCV
0	0	X
0	1	0
1	0	1
1	1	X

DP	Input	Output	Output from transceiver single-ended D+ signal receiver

D+	DP
0	0
1	1

DM	Input	Output	Output from transceiver single-ended D- signal receiver

D-	DM
0	0
1	1

24.2.2.4.1.2 Bidirectional

When a USB or USB OTG transceiver is connected to the device and is used in 3-pin bidirectional DAT/SE0 encoding mode, the encoding described in Table 24-25 is used.

Table 24-25. Signaling Between High-Speed USB Host Subsystem and 3-Pin Bidirectional USB Transceiver Using DAT/SE0 Signaling

Logical Signal Name	Device Pin Direction	Transceiver Pin Direction	Description
TXEN	Output	Input	When low or high (depending on the CONTROL.CONTROL_WKUP_CTRL[2, 1 and/or 0] MM_FSUSBi_TXEN_N_OUT_POLARITY_CTRL bit), the USB transceiver drives D+ and D-.

Table 24-25. Signaling Between High-Speed USB Host Subsystem and 3-Pin Bidirectional USB Transceiver Using DAT/SE0 Signaling (continued)

Logical Signal Name	Device Pin Direction	Transceiver Pin Direction	Description				
DAT and SE0	Output	Input	When TXEN is low/high (depending on the TXEN signal polarity; see the CONTROL.CONTROL_WKUP_CTRL[2, 1 and/or 0] MM_FSUSBi_TXEN_N_OUT_POLARITY_CTR bit), the device drives DAT and SE0 and the transceiver drives D+ and D- based on the values of DAT and SE0.				
	Output	Input	TXEN	DAT	SE0	D+	D-
			0	0	0	0	1
				1	0	1	0
				X	1	0	0
	Input	Output	TXEN	D+	D-	DAT	SE0
			1	0	0	0	1
				0	1	0	0
				1	0	1	0
				1	1	Undefined	Undefined

Note: The device does not support 3-wire bidirectional signaling using DP/DM signals.

When a USB or USB OTG transceiver is connected to the device and is used in 4-pin bidirectional DP/DM encoding mode, the encoding described in Table 24-26 is used.

Table 24-26. Signaling Between High-Speed USB Host Subsystem and 4-Pin Bidirectional USB Transceiver Using DP/DM Signaling

Logical Signal Name	Device Pin Direction	Transceiver Pin Direction	Description		
TXEN	Output	Input	When low or high (depending on the CONTROL.CONTROL_WKUP_CTRL[2, 1 and/or 0] MM_FSUSBi_TXEN_N_OUT_POLARITY_CTRL bit), the USB transceiver drives D+ and D-.		
DM			Value driven to or received from D-		
	Output	Input	TXEN	DM	D-
			0	0	0
			0	1	1
	Input	Output	TXEN	D-	DM
			1	0	0
			1	1	1
DP			Value driven to or received from D+		
	Output	Input	TXEN	DP	D+
			0	0	0
			0	1	1
	Input	Output	TXEN	D+	DP
			1	0	0
			1	1	1

Table 24-26. Signaling Between High-Speed USB Host Subsystem and 4-Pin Bidirectional USB Transceiver Using DP/DM Signaling (continued)

Logical Signal Name	Device Pin Direction	Transceiver Pin Direction	Description		
RCV	Input	Output	Output from transceiver single-ended D- signal receiver		
			D+	D-	RCV
			0	0	X
			0	1	0
			1	0	1
			1	1	X

Note: The device does not support 4-pin bidirectional signaling using DAT/SE0 signals.

24.2.2.4.2 Sideband Signals for Serial Modes

Serial interfaces only carry the USB data information. Sideband control and status (respectively, to/from the transceiver/TLL or to the bus lines themselves) require additional signals, which are usually implemented in a case-by-case, ad hoc way.

- Sideband control examples: FS/LS (slew rate control), transceiver suspend, connect (D+/D– pullup), pulldown enable, VBUS drive, etc.
- Sideband status example: VBUS level (VBUS valid, session valid, session end), etc.
- Sideband signal implementations: Dedicated lines (one per sideband information bit), serial bus + interrupt line with register-mapped control/status (I2C, UART, etc.)

Figure 24-17 and Figure 24-18 show system integration for sideband signals for two logically identical USB connections: One in transceiver configuration, and one in TLL configuration. Although the sideband (purple) arrows are all oriented from controller to transceiver in the two figures, the sideband information flow is bidirectional (that is, it flows from controller to transceiver [control] but also from transceiver to controller [status]).

Figure 24-17 shows the transceiver configuration, where each side connects the sideband signals to its own transceiver. On the device (containing the USBTLL module), the sideband is decoded/re-encoded. The sideband signals available at the device boundary (and the USBTLL module) are decoded from the standard UTMI+ interface.

- Sideband output signals from the USBTLL module (speed, suspend, puen, etc.)
- The software-driven VBUS reporting procedure is described in Section 24.2.4.2.4.1.1.

Figure 24-17. Serial Interface Sideband Integration - Transceiver Configuration

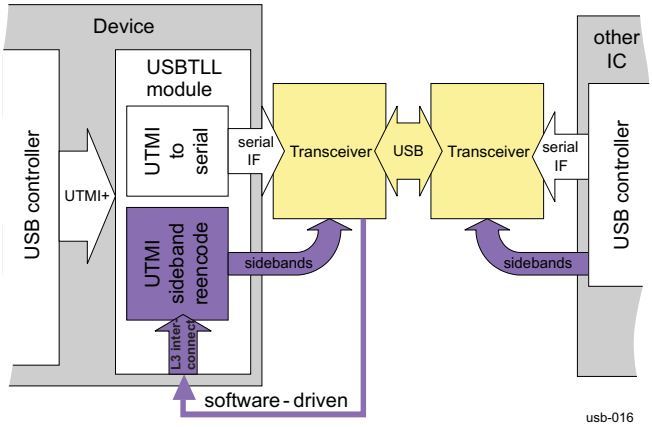
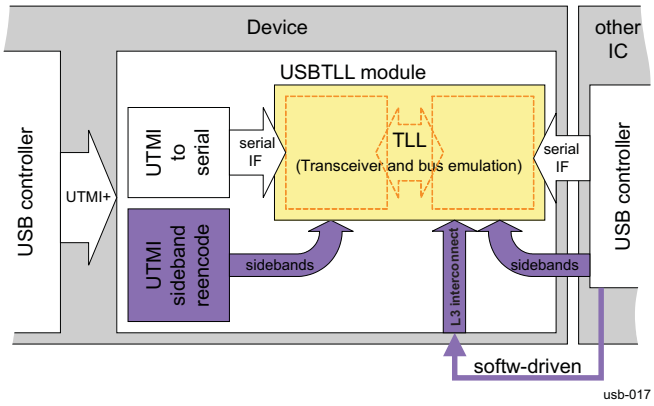


Figure 24-18 shows the TLL configuration, where both transceivers are actually emulated inside the USBTLL module.

- The transceiver of the local controller (left) is working with the sideband information to/from the UTMI+ port. This is internal to the USBTLL module.
- The transceiver of the remote controller (right) must communicate with its controller, located on another IC. This is done in two ways:
  - Sideband input signals at the TLL module boundary (tllpuen, tlldrvvbus, tllvbusvalid, etc.)
  - The software-driven VBUS control procedure is described in Section 24.2.4.2.4.2.2

Figure 24-18. Serial Interface Sideband Integration - TLL Configuration



24.2.2.4.3 Transceiver Interface Configurations

An external USB transceiver is required for each USB port used in the system. It converts between appropriate signaling for the high-speed USB host subsystem and appropriate signaling for the USB wire.

The serial interface mode of the high-speed USB host subsystem includes support for several types of USB transceivers. It provides signaling to up to three external USB transceivers.

Several types of external transceiver signaling are supported. Signaling between the high-speed USB subsystem in the serial-interface mode and the external USB transceiver for monitoring and controlling the differential USB signal can be done through a 6-, 4-, or 3-wire signaling interface, with two or more control signals provided either by additional signals or through an I^2C link.

The following subsections describe the transceiver interface modes supported by the high-speed USB host subsystem in the serial interface mode. In each case, the subsystem is connected to external transceivers, on the other side of which are the actual USB lines (D+/D–).

24.2.2.4.3.1 Unidirectional Transceiver Interface Modes: 6-Pin

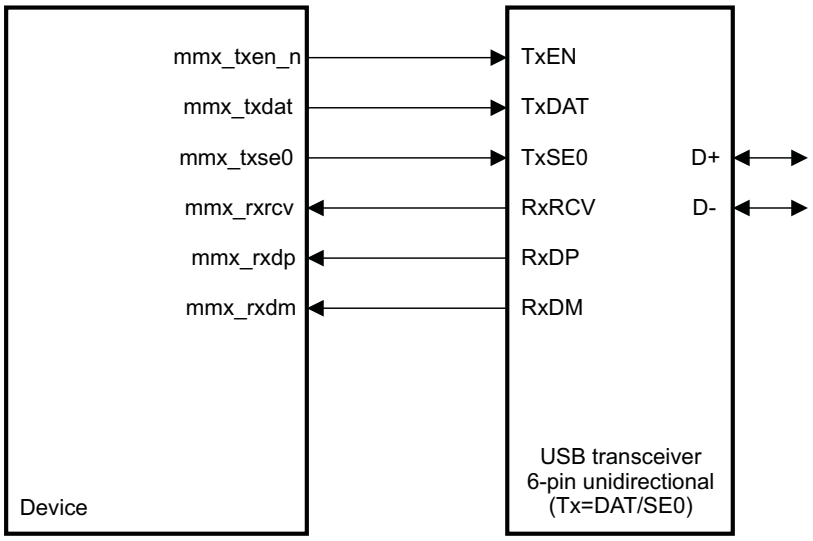
The 6-pin modes are the "natural" transceiver interface modes for the full-speed transceivers in the sense that they mirror the internal makeup of the transceivers.

Two encodings exist for TX: DAT/SE0 or DP/DM.

When a USB is connected to the device and used in 6-pin unidirectional DAT/SE0 signaling mode, the signaling described in Table 24-23 is used.

Figure 24-19 shows a USB port using DAT/SE0 encoding.

Figure 24-19. 6-Pin Unidirectional Using DAT/SE0 Signaling

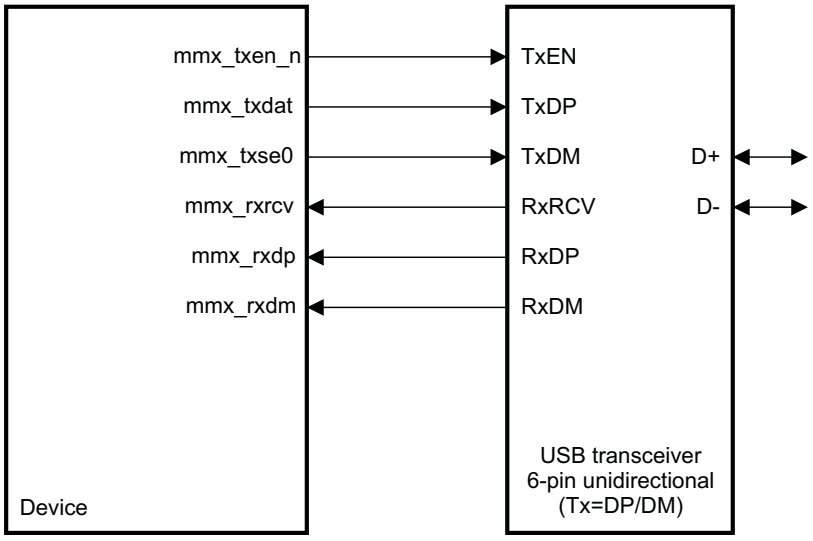


x is the USB port number (1, 2 or 3)

When a USB is connected to the device and used in 6-pin unidirectional DP/DM signaling mode, the signaling described in Table 24-24 is used.

Figure 24-20 shows a USB port using DP/DM encoding.

Figure 24-20. 6-Pin Unidirectional Using DP/DM Signaling



x is the USB port number (1, 2 or 3)

24.2.2.4.3.2 Bidirectional Transceiver Interface Modes: 3-Pin, 4-Pin

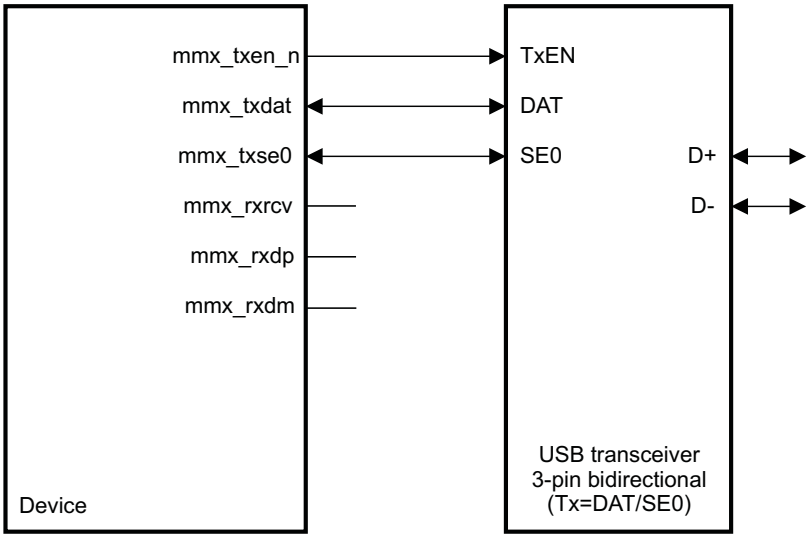
The bidirectional transceiver interface modes are pin-count optimizations of the unidirectional modes. They take advantage of the fact that a USB port is either sending or receiving at any given time, but never both. The TX and RX paths of the unidirectional mode can be multiplexed on bidirectional lines. To prevent glitches at TX/RX turnaround, the same encoding is used for both directions (DAT/SE0 or DP/DM).

The signaling listed in Table 24-25 is used when a USB transceiver is connected to the device and is used in 3-pin bidirectional DAT/SE0 signaling mode.

Note: The device does not support 3-wire bidirectional signaling using DP/DM signals.

Figure 24-21 shows a USB port using DAT/SE0 encoding.

Figure 24-21. 3-Pin Bidirectional Using DAT/SE0 Signaling

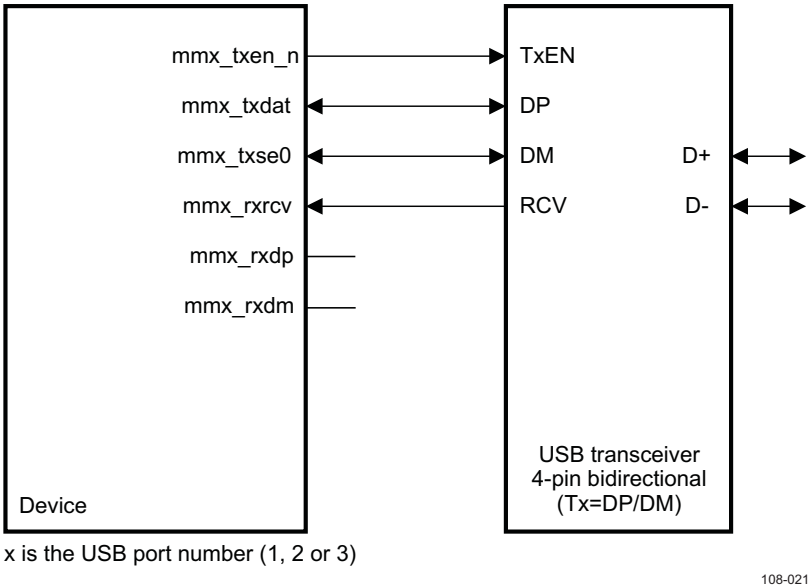


The signaling listed in Table 24-26 is used when a USB transceiver is connected to the device and is used in 4-pin bidirectional DP/DM signaling mode.

Note: The device does not support 4-pin bidirectional signaling using DAT/SE0 signals.

Figure 24-22 shows a USB port using DP/DM encoding.

Figure 24-22. 4-Pin Bidirectional Using DP/DM Signaling



24.2.2.4.4 TLL Configurations

The high-speed USB host subsystem supports unidirectional and bidirectional TLL logic interfaces on its ports. The TLL modes enable glueless interconnect to the USB device port of another device without needing a costly transceiver.

Serial interface modes are full- or low-speed only. Transceiver interface is serial (its frequency is that of the actual USB line) and combinatorial (no clock is passed).

24.2.2.4.4.1 Unidirectional TLL Modes

The 6-pin TLL configurations are mirror images of the 6-pin transceiver configurations presented above. The same signals are mapped on the same physical pins, but in the opposite directions.

Two possible modes exist, depending on the TX data encoding used by the external device.

Figure 24-23 shows an external device using DAT/SE0 encoding.

Figure 24-23. 6-Pin Unidirectional TLL Using DAT/SE0 Signaling

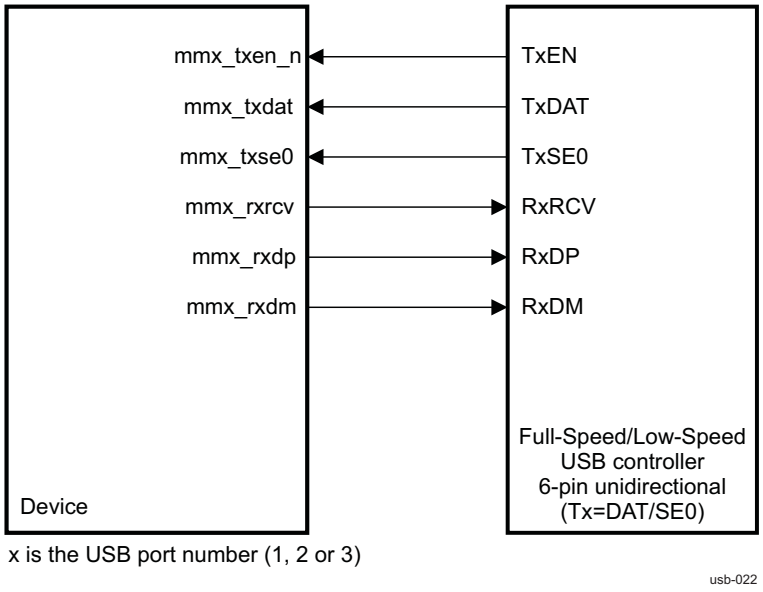
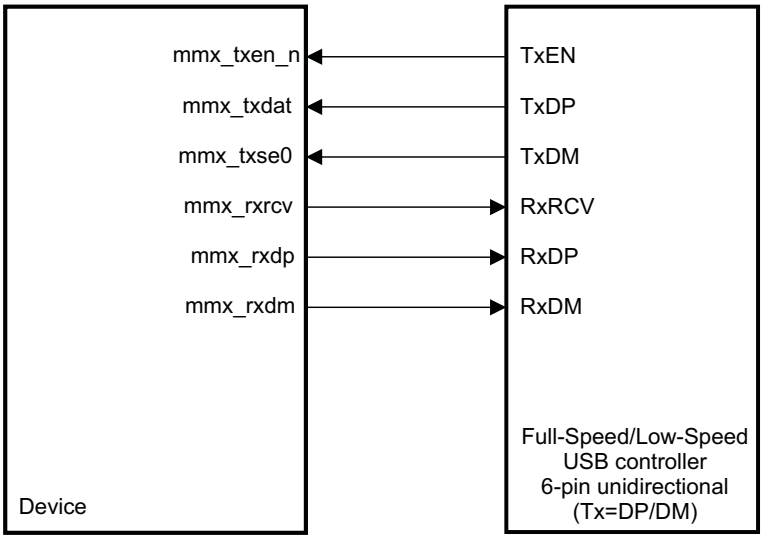


Figure 24-24 shows an external device using DP/DM encoding.

Figure 24-24. 6-Pin Unidirectional TLL Using DP/DM Signaling



x is the USB port number (1, 2 or 3)

usb-023

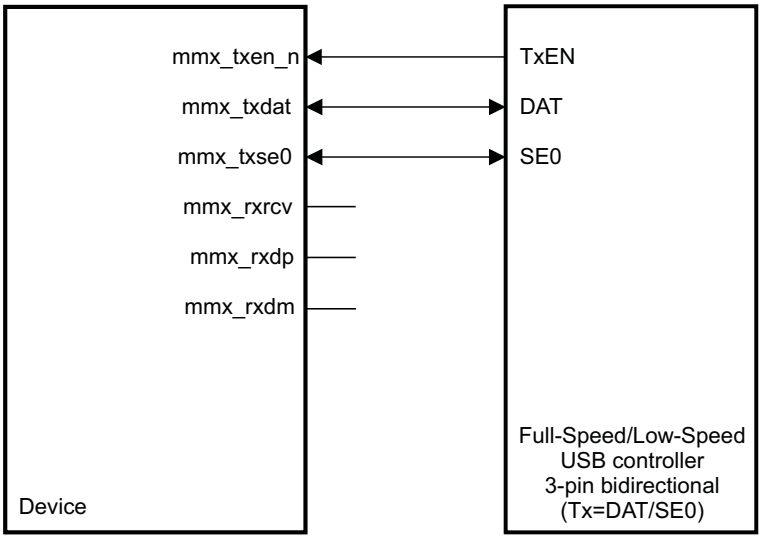
24.2.2.4.4.2 Bidirectional TLL Modes

The 3-pin/4-pin TLL configurations are mirror images of the 3-pin/4-pin transceiver configurations presented above. The same signals are mapped on the same physical pins, but in the opposite directions (bidirectional lines remain bidirectional).

Two possible modes exist, depending on the TX data encoding used by the external device.

Figure 24-25 shows an external device using DAT/SE0 encoding.

Figure 24-25. 3-Pin Bidirectional TLL Using DAT/SE0 Signaling

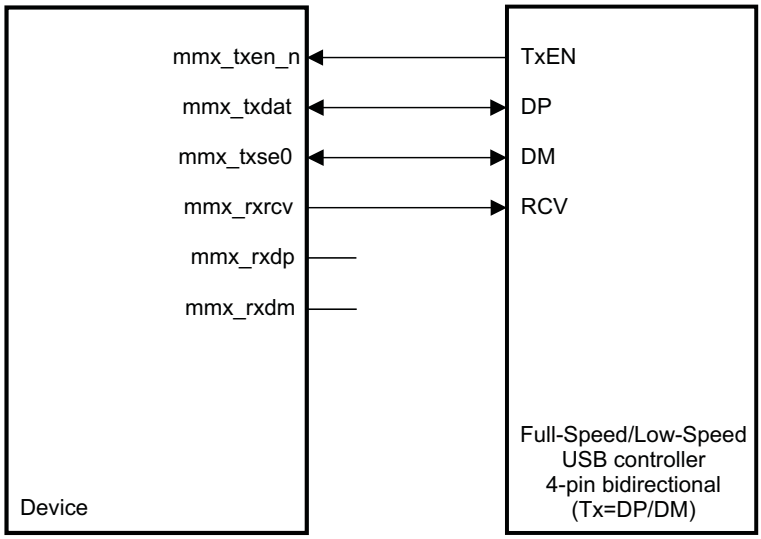


x is the USB port number (1, 2 or 3)

usb-024

Figure 24-26 shows an external device using DP/DM encoding.

Figure 24-26. 4-Pin Bidirectional TLL Using DP/DM Signaling



The 2-pin TLL configurations have unique specifications:

- They require pullups/pulldowns to operate, because the bidirectional lines are not driven at all times like the other serial transceiver interfaces described above. The connection of pull resistors depends on the speed of the controller.
- The module supports explicit 2-pin TLL modes, with either DAT/SE0 or DP/DM encoding.
- Non-TLL modes (that is, transceiver configuration mode) can be used to implement the 2-pin functionality, using a specific connectivity.

Figure 24-27 shows USB port using DP/DM encoding.

Figure 24-27. 2-Pin Bidirectional TLL Using DP/DM Encoding, With 4-Pin Bidirectional USB Device

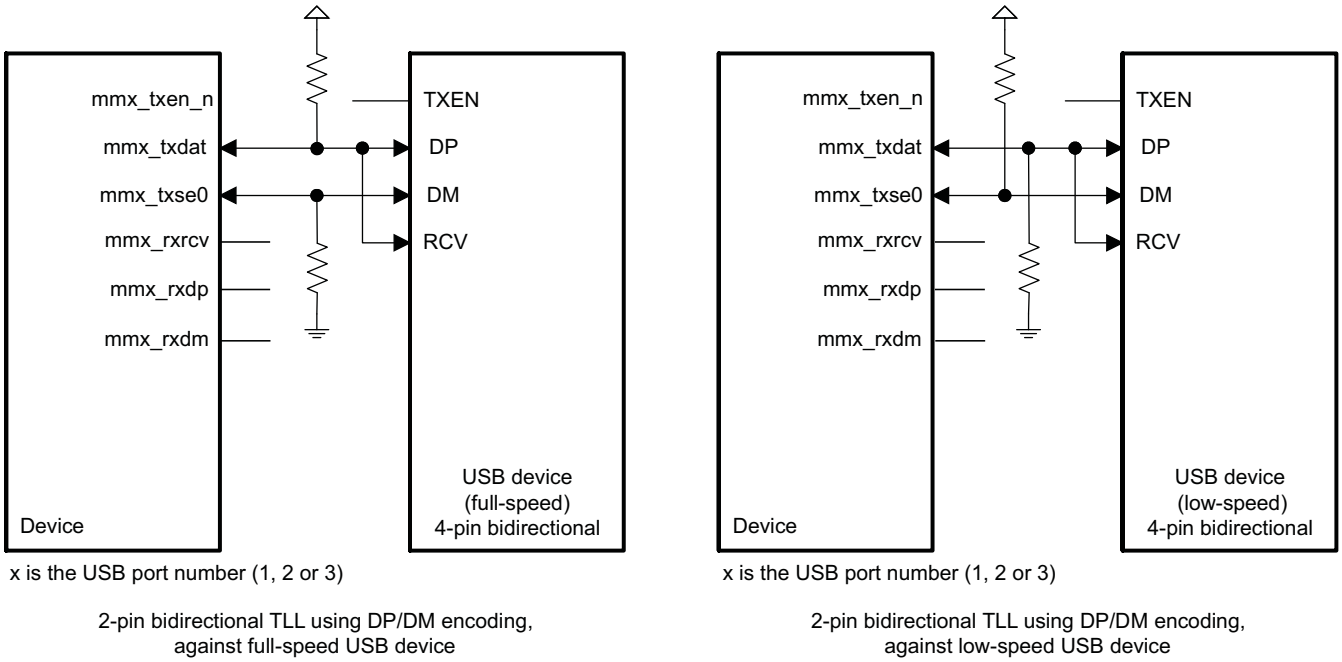


Table 24-27 shows the pullup/pulldown configuration for DP/DM encoding.

Table 24-27. Pullup/Pulldown Configuration for DP/DM Encoding

	Nonconnected Device (Any Speed)	Connected Low-Speed Device	Connected Full-Speed Device
DP	Pulldown	Pulldown	Pullup
DM	Pulldown	Pullup	Pulldown

Figure 24-28 shows a USB port using DAT/SE0 encoding.

Figure 24-28. 2-Pin Bidirectional TLL Using DAT/SE0 Encoding, With 3-Pin Bidirectional USB Device

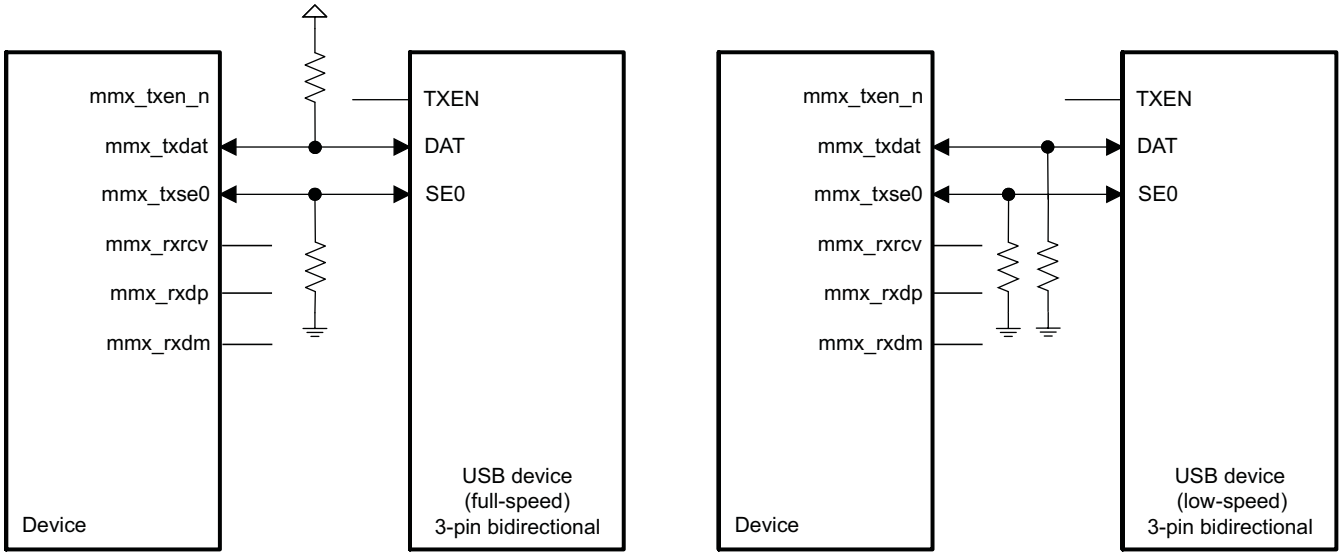


Table 24-28 shows pullup/pulldown configuration for DAT/SE0 encoding.

Table 24-28. Pullup/Pulldown Configuration for DAT/SE0 Encoding

	Nonconnected Device (Any Speed)	Connected Low-Speed Device	Connected Full-Speed Device
DAT	Pulldown	Pulldown	Pullup
SEO	Pullup	Pulldown	Pulldown

24.2.2.4.5 High-Speed USB Host Subsystem Interface Description

Table 24-29 describes the I/O of the high-speed USB host subsystem serial interfaces.

Table 24-29. I/O Description

Signal Name	I/O[(1)]	Description	Value at Reset
Multiple-Mode FS/LS Serial Interface: Port 1			
mm1_txse0	I/O	SE0 function in 3-pin bidirectional DAT/SE0 mode	0
	I/O	DM function in 4-pin bidirectional DP/DM mode	
	O	SE0 output in 6-pin unidirectional DAT/SE0 mode	
	O	DM output in 6-pin unidirectional DP/DM mode	
	I/O	SE0-TLL in 2-/3-pin bidirectional DAT/SE0 TLL mode	
	I/O	DM-TLL in 2-/4-pin bidirectional DP/DM TLL mode	

(1) I = Input, O = Output

Table 24-29. I/O Description (continued)

Signal Name	I/O[1]	Description	Value at Reset
Multiple-Mode FS/LS Serial Interface: Port 1			
	I	SE0-TLL input in 6-pin unidirectional DAT/SE0 TLL mode	
	I	DM-TLL input in 6-pin unidirectional DP/DM TLL mode	
mm1_txdat	I/O	DAT function in 3-pin bidirectional DAT/SE0 mode	Unknown
	I/O	DP function in 4-pin bidirectional DP/DM mode	
	O	DAT output in 6-pin unidirectional DAT/SE0 mode	
	O	DP output in 6-pin unidirectional DAT/SE0 mode	
	I/O	DAT-TLL in 2-/3-pin bidirectional DAT/SE0 TLL mode	
	I/O	DP-TLL in2-/4-pin bidirectional DP/DM TLL mode	
	I	DAT-TLL input in 6-pin unidirectional DAT/SE0 TLL mode	
	I	DP-TLL input in 6-pin unidirectional DP/DM TLL mode	
mm1_txen_n	O	Transmit enable in the 3-pin bidirectional DAT/SE0 or 4-pin bidirectional DP/DM or 6-pin unidirectional modes	1
	I	Transmit enable in the 3-pin bidirectional DAT/SE0 TLL or 4-pin bidirectional DP/DM TLL or 6-pin unidirectional TLL modes (not used in the 2-pin bidirectional TLL modes)	
mm1_rxrcv	I	Differential receiver signal input in the 4-pin bidirectional DP/DM or 6-pin unidirectional modes (not used in the 3-pin bidirectional DAT/SE0 mode)	Unknown
	O	Differential receiver signal output in the 4-pin bidirectional DP/DM TLL or 6-pin unidirectional TLL modes (not used in the 3-pin bidirectional DAT/SE0 TLL or 2-pin bidirectional TLL modes)	
mm1_rxdp	I	Single-ended DP receiver signal input in 6-pin unidirectional modes (not used in the 3-pin bidirectional DAT/SE0 or 4-pin bidirectional DP/DM modes)	Unknown
	O	Single-ended DP receiver signal output in 6-pin unidirectional TLL modes (not used in the 3-pin bidirectional DAT/SE0 TLL or 4-pin bidirectional DP/DM TLL or 2-pin bidirectional TLL modes)	
mm1_rxdm	I	Single-ended DM receiver signal input in 6-pin unidirectional TLL modes (not used in the 3-pin bidirectional DAT/SE0 or 4-pin bidirectional DP/DM or 2-pin bidirectional TLL modes)	Unknown
	O	Single-ended DM receiver signal output in 6-pin unidirectional TLL modes (not used in the 3-pin bidirectional DAT/SE0 TLL or 4-pin bidirectional DP/DM TLL or 2-pin bidirectional TLL modes)	
Multiple-mode FS/LS serial interface: Port 2			
mm2_txse0	I/O	SE0 function in 3-pin bidirectional DAT/SE0 mode	0
	I/O	DM function in 4-pin bidirectional DP/DM mode	
	O	SE0 output in 6-pin unidirectional DAT/SE0 mode	
	O	DM output in 6-pin unidirectional DP/DM mode	
	I/O	SE0-TLL in 2-/3-pin bidirectional DAT/SE0 TLL mode	
	I/O	DM-TLL in 2-/4-pin bidirectional DP/DM TLL mode	
	I	SE0-TLL input in 6-pin unidirectional DAT/SE0 TLL mode	
	I	DM-TLL input in 6-pin unidirectional DP/DM TLL mode	
mm2_txdat	I/O	DAT function in 3-pin bidirectional DAT/SE0 mode	Unknown
	I/O	DP function in 4-pin bidirectional DP/DM mode	
	O	DAT output in 6-pin unidirectional DAT/SE0 mode	
	O	DP output in 6-pin unidirectional DAT/SE0 mode	
	I/O	DAT-TLL in 2-/3-pin bidirectional DAT/SE0 TLL mode	
	I/O	DP-TLL in2-/4-pin bidirectional DP/DM TLL mode	
	I	DAT-TLL input in 6-pin unidirectional DAT/SE0 TLL mode	
	I	DP-TLL input in 6-pin unidirectional DP/DM TLL mode	

Table 24-29. I/O Description (continued)

Signal Name	I/O[1]	Description	Value at Reset
Multiple-Mode FS/LS Serial Interface: Port 1			
mm2_txen_n	O	Transmit enable in the 3-pin bidirectional DAT/SE0 or 4-pin bidirectional DP/DM or 6-pin unidirectional modes	1
	I	Transmit enable in the 3-pin bidirectional DAT/SE0 TLL or 4-pin bidirectional DP/DM TLL or 6-pin unidirectional TLL modes (not used in the 2-pin bidirectional TLL modes)	
mm2_rxrcv	I	Differential receiver signal input in the 4-pin bidirectional DP/DM or 6-pin unidirectional modes (not used in the 3-pin bidirectional DAT/SE0 mode)	Unknown
	O	Differential receiver signal output in the 4-pin bidirectional DP/DM TLL or 6-pin unidirectional TLL modes (not used in the 3-pin bidirectional DAT/SE0 TLL or 2-pin bidirectional TLL modes)	
mm2_rxdp	I	Single-ended DP receiver signal input in 6-pin unidirectional modes (not used in the 3-pin bidirectional DAT/SE0 or 4-pin bidirectional DP/DM modes)	Unknown
	O	Single-ended DP receiver signal output in 6-pin unidirectional TLL modes (not used in the 3-pin bidirectional DAT/SE0 TLL or 4-pin bidirectional DP/DM TLL or 2-pin bidirectional TLL modes)	
mn2_rxdm	I	Single-ended DM receiver signal input in 6-pin unidirectional TLL modes (not used in the 3-pin bidirectional DAT/SE0 or 4-pin bidirectional DP/DM or 2-pin bidirectional TLL modes)	Unknown
	O	Single-ended DM receiver signal output in 6-pin unidirectional TLL modes (not used in the 3-pin bidirectional DAT/SE0 TLL or 4-pin bidirectional DP/DM TLL or 2-pin bidirectional TLL modes)	
Multiple-mode FS/LS serial interface: Port 3			
mm3_txse0	I/O	SE0 function in 3-pin bidirectional DAT/SE0 mode	0
	I/O	DM function in 4-pin bidirectional DP/DM mode	
	O	SE0 output in 6-pin unidirectional DAT/SE0 mode	
	O	DM output in 6-pin unidirectional DP/DM mode	
	I/O	SE0-TLL in 2-/3-pin bidirectional DAT/SE0 TLL mode	
	I/O	DM-TLL in 2-/4-pin bidirectional DP/DM TLL mode	
	I	SE0-TLL input in 6-pin unidirectional DAT/SE0 TLL mode	
	I	DM-TLL input in 6-pin unidirectional DP/DM TLL mode	
mm3_txdat	I/O	DAT function in 3-pin bidirectional DAT/SE0 mode	Unknown
	I/O	DP function in 4-pin bidirectional DP/DM mode	
	O	DAT output in 6-pin unidirectional DAT/SE0 mode	
	O	DP output in 6-pin unidirectional DAT/SE0 mode	
	I/O	DAT-TLL in 2-/3-pin bidirectional DAT/SE0 TLL mode	
	I/O	DP-TLL in2-/4-pin bidirectional DP/DM TLL mode	
	I	DAT-TLL input in 6-pin unidirectional DAT/SE0 TLL mode	
	I	DP-TLL input in 6-pin unidirectional DP/DM TLL mode	
mm3_txen_n	O	Transmit enable in the 3-pin bidirectional DAT/SE0 or 4-pin bidirectional DP/DM or 6-pin unidirectional modes	1
	I	Transmit enable in the 3-pin bidirectional DAT/SE0 TLL or 4-pin bidirectional DP/DM TLL or 6-pin unidirectional TLL modes (not used in the 2-pin bidirectional TLL modes)	
mm3_rxrcv	I	Differential receiver signal input in the 4-pin bidirectional DP/DM or 6-pin unidirectional modes (not used in the 3-pin bidirectional DAT/SE0 mode)	Unknown
	O	Differential receiver signal output in the 4-pin bidirectional DP/DM TLL or 6-pin unidirectional TLL modes (not used in the 3-pin bidirectional DAT/SE0 TLL or 2-pin bidirectional TLL modes)	
mm3_rxdp	I	Single-ended DP receiver signal input in 6-pin unidirectional modes (not used in the 3-pin bidirectional DAT/SE0 or 4-pin bidirectional DP/DM modes)	Unknown

Table 24-29. I/O Description (continued)

Signal Name	I/O[1]	Description	Value at Reset
Multiple-Mode FS/LS Serial Interface: Port 1			
	O	Single-ended DP receiver signal output in 6-pin unidirectional TLL modes (not used in the 3-pin bidirectional DAT/SE0 TLL or 4-pin bidirectional DP/DM TLL or 2-pin bidirectional TLL modes)	
mm3_rxdm	I	Single-ended DM receiver signal input in 6-pin unidirectional modes (not used in the 3-pin bidirectional DAT/SE0 or 4-pin bidirectional DP/DM modes)	Unknown
	O	Single-ended DM receiver signal output in 6-pin unidirectional TLL modes (not used in the 3-pin bidirectional DAT/SE0 TLL or 4-pin bidirectional DP/DM TLL or 2-pin bidirectional TLL modes)	

24.2.3 High-Speed USB Host Subsystem Integration

This section describes the integration of the high-speed USB host subsystem.

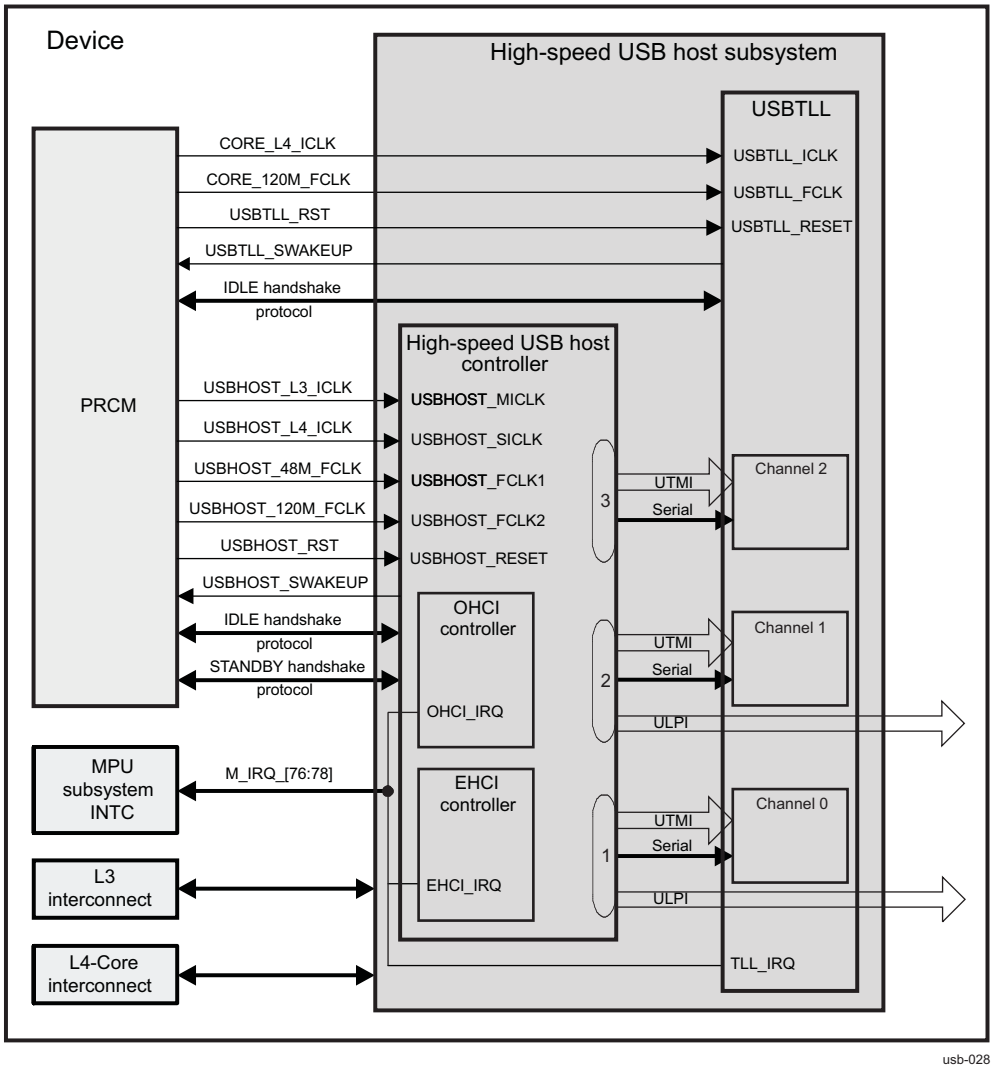
The high-speed USB host controller is connected to the L3 interconnect master (initiator) and L4-Core interconnect slave (target) interfaces. The USBTLL module is connected to the L4-Core interconnect slave (target) interface. The L3 interconnect is used to generate data traffic within the device. The L4-Core interconnect is a configuration port for register setting.

Three interrupts, M_IRQ_[78:76], are connected to the system.

The system control module (SCM) offers complementary settings for USB port connectivity modes.

Figure 24-29 highlights the high-speed USB host subsystem integration in the device.

Figure 24-29. High-Speed USB Subsystem Integration



24.2.3.1 Reset, Clocking, and Power-Management Scheme

The high-speed USB host controller belongs to the USBHOST power domain. As part of the USBHOST power domain, it is sensible to the USBHOST_RST reset signal issued by the PRCM and to USBHOST power domain-related transitions. For further details about USBHOST power domain implementation and USBHOST_RST signal, see Chapter 4, *Power, Reset, and Clock Management*.

The USBTLL module belongs to the CORE power domain. As part of the CORE power domain, it is sensible to the asynchronous USBTLL_RST reset signal issued by the PRCM and to CORE power domain-related transitions. For further details about CORE power domain implementation and USBTLL_RST signal, see Chapter 4, *Power, Reset, and Clock Management*.

24.2.3.1.1 High-Speed USB Host Subsystem Resets

Table 24-30 lists and describes all the high-speed USB host subsystem resets:

Table 24-30. High-Speed USB Host Subsystem Reset Description

Type	Name	Source	Polarity	Description
Software	USBHOST.UHH_SYSCONFIG[1] SOFTRESET bit	High-speed USB host controller internal software reset	Active high	Writing 1 to the SOFTRESET bit resets the module. The bit value of 1 remains until the reset is complete. When the software reset is complete, the SOFTRESET bit is automatically reset to 0 and has the same effect as the hardware reset.
Software	USBHOST.USBTLL_SYSCONFIG[1] SOFTRESET bit	USBTLL module internal software reset	Active high	Writing 1 to the SOFTRESET bit resets the module. The bit value of 1 remains until the reset is complete. When the software reset is complete, the SOFTRESET bit is automatically reset to 0 and has the same effect as the hardware reset.
Hardware	USBHOST_RESET	PRCM USBHOST_RST signal	Active low	The USBHOST_RST signal resets the module. The hardware reset signal has a global reset action on the high-speed USB host controller (see Chapter 4, *Power, Reset, and Clock Management*).
Hardware	USBTLL_RESET	PRCM USBTLL_RST signal	Active low	The USBTLL_RST signal resets asynchronously the module. The hardware reset signal has a global reset action on the USBTLL module (see Chapter 4, *Power, Reset, and Clock Management*).

24.2.3.1.1.1 Hardware Resets

The high-speed USB host controller is attached to the USBHOST power domain: The USBHOST_RST signal resets the module. The USBTLL module is attached to the CORE power domain: The USBTLL_RST signal resets asynchronously the module (see Chapter 4, *Power, Reset, and Clock Management*). The hardware reset signal has a global reset action on the modules.

24.2.3.1.1.2 Software Resets

The high-speed USB host controller and the USBTLL module have their own software-reset functionality through the USBHOST.UHH_SYSCONFIG[1] SOFTRESET bit (0: Normal mode; 1: Module is reset) for the high-speed USB host controller, and through the USBHOST.USBTLL_SYSCONFIG[1] SOFTRESET bit (0: Normal mode; 1: Module is reset) for the USBTLL module.

Writing 1 to the SOFTRESET bit resets the module. The bit value of 1 remains until the reset is complete. When the software reset is complete, the SOFTRESET bit is automatically reset to 0 and has the same effect as the hardware reset.

24.2.3.1.2 High-Speed USB Host Subsystem Clocks

The high-speed USB host controller operates from four clock domains:

- USBHOST_FCLK1 is a high-speed USB host controller functional clock. It is used to clock the OHCI and EHCI controller internal logic of the high-speed USB host controller. Its source is the PRCM USBHOST_48M_FCLK output clock. USBHOST_FCLK1 is controlled by the PRCM.PRCM.CM_FCLKEN_USBHOST[0] EN_USBHOST1 bit (0: Disabled; 1: Enabled).
- USBHOST_FCLK2 is a high-speed USB host controller functional clock. It is used to clock EHCI controller internal logic of the high-speed USB host controller. Its source is the PRCM USBHOST_120M_FCLK output clock. USBHOST_FCLK2 is controlled by the PRCM.PRCM.CM_FCLKEN_USBHOST[1] EN_USBHOST2 bit (0: Disabled; 1: Enabled).
- USBHOST_MICLK is the high-speed USB host controller L3 master interface clock. It is used to synchronize the high-speed USB host controller L3 port to L3 interconnect. All accesses from the

interconnect are synchronous to USBHOST_MICLK. Its source is the PRCM USBHOST_L3_ICLK output clock. USBHOST_MICLK is controlled by the PRCM.CM_ICLKEN_USBHOST[0] EN_USBHOST bit (0: Disabled; 1: Enabled) and the PRCM.CM_AUTOIDLE_USBHOST[0] AUTO_USBHOST bit (enables/disables automatic control of the interface clock).

- USBHOST_SICLK is a high-speed USB host controller L4 slave interface clock. It is used to synchronize the high-speed USB host controller L4 port to L4 interconnect. All accesses from the interconnect are synchronous to USBHOST_SICLK. Its source is the PRCM USBHOST_L4_ICLK output clock.

The USBTLL module operates from two clock domains:

- USBTLL_FCLK is the USBTLL module functional clock. It is used to clock the USBTLL module internal logic. Its source is the PRCM CORE_120M_FCLK output clock. UBTLL_FCLK is controlled by the PRCM.CM_FCLKEN3_CORE[2] EN_USBTLL bit (0: Disabled; 1: Enabled).
- USBTLL_ICLK is the USBTLL module interface clock. It is used to synchronize USBTLL module L4 port to L4 interconnect. All accesses from the interconnect are synchronous to USBTLL_ICLK. Its source is the PRCM CORE_L4_ICLK output clock. USBTLL_ICLK is controlled by the PRCM.CM_ICLKEN3_CORE[2] EN_USBTLL bit (0: Disabled; 1: Enabled) and the PRCM.CM_AUTOIDLE3_CORE[2] AUTO_USBTLL bit (enables/disables automatic control of the interface clock).

Table 24-31 summarizes the high-speed USB host subsystem clocks.

Table 24-31. High-Speed USB Host Subsystem Clocks

Attributes	Frequency	Name	Module	Mapping	Comments
Functional clock	48 MHz	USBHOST_ FCLK1	High-speed USB Host controller	USBHOST_48M_FCLK	Source is PRCM module
Functional clock	120 MHz	USBHOST_ FCLK2	High-speed USB Host controller	USBHOST_ 120M_FCLK	Source is PRCM module
L3 master interface clock	Depending on PRCM register settings	USBHOST_ MICLK	High-speed USB Host controller	USBHOST_L3_ICLK	Source is PRCM module
L4 slave interface clock	Depending on PRCM register settings	USBHOST_ SICLK	High-speed USB Host controller	USBHOST_L4_ICLK	Source is PRCM module
Functional clock	120 MHz	USBTLL_FCLK	USBTLL	CORE_120M_FCLK	Source is PRCM module
L4 interface clock	Depending on PRCM register settings	USBTLL_ICLK	USBTLL	CORE_L4_ICLK	Source is PRCM module

24.2.3.1.2.1 L3 Master Interface Clock

The L3 master interface clock (USBHOST_L3_ICLK) is used only by the high-speed USB host controller (USBHOST_MICLK) in the subsystem, and comes from the PRCM module. This clock is controlled by the PRCM register bits PRCM.CM_ICLKEN_USBHOST[0]:

0: Disabled
1: Enabled

and PRCM.CM_AUTOIDLE_USBHOST[0] (enables/disables automatic control of the interface clock)—see Table 24-32.

Table 24-32. High-Speed USB Controller L3 Master Interface Clock

PRCM.CM_AUTOIDLE_USBHOST[0]	PRCM.CM_ICLKEN_USBHOST[0]	Interface Clock
0	0	Disabled
0	1	Enabled
1	0	Disabled
1	1	Automatic enabling/disabling

CAUTION

The L3 master interface clock should not be below 30 MHz, ULPI clock divided by 2 (this can occur during DPLL3 relocking).

24.2.3.1.2.2 L4 Slave Interface Clock

The L4 slave interface clock (USBHOST_L4_ICLK) is used only by the high-speed USB host controller (USBHOST_SICLK) in the subsystem, and comes from the PRCM module.

24.2.3.1.2.3 L4 Interface Clock

The L4 interface clock (CORE_L4_ICLK) is used only by the USBTLL module (USBTLL_ICLK) in the subsystem, and comes from the PRCM module. This clock is controlled by the PRCM register bits PRCM.CM_ICLKEN3_CORE[2]:

0: Disabled
1: Enabled

and PRCM.CM_AUTOIDLE3_CORE[2] (enables/disables automatic control of the interface clock)—see Table 24-33.

Table 24-33. USBTLL Module Interface Clock

PRCM.CM_AUTOIDLE3_CORE[2]	PRCM.CM_ICLKEN3_CORE[2]	Interface Clock
0	0	Disabled
0	1	Enabled
1	0	Disabled
1	1	Automatic enabling/disabling

24.2.3.1.2.4 Functional Clocks

Two functional clocks are provided by the PRCM module to the high-speed USB host controller: USBHOST_FLCK1 running at 48 MHz (USBHOST_48M_FCLK) and USBHOST_FLCK2 running at 120 MHz (USBHOST_120M_FCLK).

The USBHOST_FLCK1 functional clock is controlled by the PRCM register bit PRCM.CM_FCLKEN_USBHOST[0]:

0: Disabled
1: Enabled

The USBHOST_FLCK2 functional clock is controlled by the PRCM register bit PRCM.CM_FCLKEN_USBHOST[1]:

0: Disabled
1: Enabled

The CORE_120M_FCLK functional clock is used only by the USBTLL module (USBTLL_FCLK) in the subsystem, and comes from the PRCM module. This clock is controlled by the PRCM register bit PRCM.CM_FCLKEN3_CORE[2]:

0: Disabled
1: Enabled

24.2.3.1.3 Power-Management Scheme

24.2.3.1.3.1 High-Speed USB Host Controller Power-Management Scheme

24.2.3.1.3.1.1 Overview

To save dynamic power consumption, an efficient idle scheme in the device is based on the following:

- An efficient local autoclock gating for each module
- The implementation of control sideband signals between the PRCM module and each module

This enhanced idle control allows clocks to be activated/deactivated safely without complex software intervention. In both cases, the high-speed USB host controller power management is applied only to the interface clock domain.

The high-speed USB host controller has both master (initiator) and slave (target) interfaces.

- As an initiator, the high-speed USB host controller implements the standby handshake protocol to inform the PRCM module when it enters standby mode and does not generate traffic on the interconnect.
- As a target, the high-speed USB host controller implements the IDLE handshake protocol to allow the PRCM module requiring it to enter idle mode.

Table 24-34 details the high-speed USB host controller module PRCM clock control bits.

Table 24-34. High-Speed USB Host Controller PRCM Clock Control Bits

Module Clock	Associated PRCM Clock Output	Enabled Bit	Autoidle Bit
USBHOST_FCLK1	USBHOST_48M_FCLK	PRCM.CM_FCLKEN_USBHOST[0] EN_USBHOST1 bit	N/A
USBHOST_FCLK2	USBHOST_120M_FCLK	PRCM.CM_FCLKEN_USBHOST[1] EN_USBHOST2 bit	N/A
USBHOST_MICLK	USBHOST_L3_ICLK	PRCM.CM_ICLKEN_USBHOST[0] EN_USBHOST bit	PRCM.CM_AUTOIDLE_USBHOST[0] AUTO_USBHOST bit

Notes:

- The PRCM USBHOST_48M_FCLK output is cut at PRCM level assuming all the modules that share it have been disabled in the corresponding register. Disabling the high-speed USB host controller is a necessary but not sufficient condition.
- The PRCM USBHOST_120M_FCLK output is cut at PRCM level assuming all the modules that share it have been disabled in the corresponding register. Disabling the high-speed USB host controller is a necessary but not sufficient condition.
- The PRCM USBHOST_L3_ICLK output is cut at PRCM level assuming all the modules that share it have been disabled in the corresponding register. Disabling the high-speed USB host controller is a necessary but not sufficient condition.
- The PRCM.CM_AUTOIDLE_USBHOST[0] AUTO_USBHOST bit is used to link/unlink the high-speed USB host controller from USBHOST_L3_ICLK-related clock domain transitions.
- For further details about source clocks gating and domain transitions, see Chapter 4, *Power, Reset, and Clock Management.*

24.2.3.1.3.1.2 L3 Master Interface Power Management

The high-speed USB host controller can go to standby mode, in which case it stops generating transactions on the interconnect. The module standby leads the PRCM to disable the USB clocks to save power.

The high-speed USB host controller has a MSTANDBY/WAIT handshake mechanism with the PRCM module (see Figure 24-29).

The module is ready to enter standby mode (indicated by the MSTANDBY signal to the PRCM asserted) when there is no USB activity and the module is idle. It means the following:

- The module is committed not to start any new transaction on its master interface.
- The whole module is idle and, therefore, the power manager can start the procedure to turn off the interface clock, if needed. This procedure must be implemented using the slave power-management protocol.

The handshake mechanism lets the module go to standby mode based on the USBHOST.UHH_SYSCONFIG[13:12] MIDLEMODE field.

Table 24-35. High-Speed USB Host Controller MIDLEMODE Settings

MIDLEMODE Value	Selected Mode	Description
0x0	Force-standby	The high-speed USB host controller enters standby mode unconditionally (MSTANDBY is asserted unconditionally).
0x1	No-standby	The high-speed USB host controller never enters standby mode (MSTANDBY is never asserted).
0x2	Smart-standby	The high-speed USB host controller is ready to enter standby mode (MSTANDBY is asserted) when there is no more activity on the USB master interface of the interconnect. MSTANDBY is asserted when the module is idle and deasserted when the module is activated by either an external USB event or an appropriate register access. The module then waits for MWAIT deassertion before a DMA transfer is started.

24.2.3.1.3.1.3 L4 Slave Interface Power Management

At PRCM level, when all the conditions to shut off the high-speed USB host controller output clocks are met (see Chapter 4, *Power, Reset, and Clock Management* for details), the PRCM module automatically launches a hardware handshake protocol to ensure the high-speed USB host controller is ready to have its clocks switched off. Namely, the PRCM asserts an IDLE request to the high-speed USB host controller. Although this handshake is completely hardware and out of any software control, the way in which the high-speed USB host controller acknowledges the PRCM IDLE request is configurable through the USBHOST.UHH_SYSCONFIG[4:3] SIDLEMODE bit field. Table 24-36 details SIDLEMODE settings and the related acknowledgment modes.

Table 24-36. High-Speed USB Host Controller SIDLEMODE Settings

SIDLEMODE Value	Selected Mode	Description
0x0	Force-idle	The high-speed USB host controller acknowledges unconditionally the IDLE request from the PRCM, regardless of its internal operations. Because such a mode does not prevent any loss of data when the clock is switched off, the mode must be used carefully.
0x1	No-idle	The high-speed USB host controller never acknowledges any IDLE request from the PRCM. This mode is safe from a module point of view as it ensures the clocks remain active; however, it is not efficient from a power-saving perspective because it does not allow the PRCM output clock to be shut off and thus the power domain to be set to a lower power state.
0x2	Smart-idle	The high-speed USB host controller acknowledges the IDLE request basing its decision on its internal activity. Namely, the acknowledge signal is asserted only when all pending transactions, IRQs or DMA requests are treated. This is the best approach for an efficient system power management.

When configured in smart-idle mode, the high-speed USB host controller also offers an additional granularity on its interface clock gating. The USBHOST.UHH_SYSCONFIG[9:8] CLOCKACTIVITY bit is used to control the interface clock internal gating while module is idle. Table 24-37 details the CLOCKACTIVITY settings.

Table 24-37. High-Speed USB Host Controller CLOCKACTIVITY Settings

CLOCKACTIVITY Value	Interface Clock Effect	Description
0	OFF	Interface clock is considered for generating the acknowledgment. This setting also means the interface clock is shut down upon PRCM IDLE request.
1	ON	Interface clock is not shut down upon PRCM IDLE request. The high-speed USB host controller can potentially acknowledge the IDLE request without checking the internal functionalities linked to its clock.

CAUTION

The PRCM does not have any hardware means to read CLOCKACTIVITY settings. Software ensures a consistent programming between the high-speed USB host controller CLOCKACTIVITY and PRCM interface clock control bit. Indeed, if the USBTLL module is disabled in the PRCM.CM_ICLKEN_USBHOST register while CLOCKACTIVITY is set to 1, nothing prevents the PRCM module from asserting its IDLE request, which is acknowledged regardless of the features associated to the USBTLL module interface clock. This may lead to unpredictable behaviors.

24.2.3.1.3.2 USBTLL Module Device Power-Management Scheme

From a global system power-management perspective, when one or both of the USBTLL module clocks are no longer required, the USBTLL module can be deactivated at PRCM level in the corresponding registers.

Table 24-38 details the USBTLL module PRCM clock control bits.

Table 24-38. USBTLL Module PRCM Clock Control Bits

Module Clock	Associated PRCM Clock Output	Enabled Bit	Autoidle Bit
USBTLL_FCLK	CORE_120M_FCLK	PRCM.CM_FCLKEN3_CORE[2] EN_USBTLL bit	N/A
USBTLL_ICLK	CORE_L4_ICLK	PRCM.CM_ICLKEN3_CORE[2] EN_USBTLL bit	PRCM.CM_AUTOIDLE3_CORE [2] AUTO_USBTLL bit

Notes:

- The PRCM CORE_120M_CLK output is cut at PRCM level assuming all the modules that share it have been disabled in the corresponding register. Disabling the USBTLL module is a necessary but not sufficient condition.
- The PRCM CORE_L4_ICLK output is cut at PRCM level assuming all the modules that share it have been disabled in the corresponding register. Disabling the USBTLL module is a necessary but not sufficient condition.
- The PRCM.CM_AUTOIDLE3_CORE[2] AUTO_USBTLL bit is used to link/unlink the USBTLL module from CORE_L4_ICLK-related clock domain transitions.
- For further details about source clocks gating and domain transitions, see Chapter 4, *Power, Reset, and Clock Management.*

At PRCM level, when all the conditions to shut off the USBTLL_FCLK or USBTLL_ICLK output clocks are met (see Chapter 4, *Power, Reset, and Clock Management* for details), the PRCM module automatically launches a hardware handshake protocol to ensure the USBTLL module is ready to have its clocks switched off. Namely, the PRCM asserts an IDLE request to the USBTLL module. Although this handshake is completely hardware and out of any software control, the way in which the USBTLL module acknowledges the PRCM IDLE request is configurable through the USBHOST.USBTLL_SYSCONFIG[4:3] SIDLEMODE bit field. Table 24-39 details SIDLEMODE settings and the related acknowledgment modes.

Table 24-39. USBTLL Module SIDLEMODE Settings

SIDLEMODE Value	Selected Mode	Description
0x0	Force-idle	The USBTLL module acknowledges unconditionally the IDLE request from the PRCM, regardless of its internal operations. Because such a mode does not prevent any loss of data when the clock is switched off, the mode must be used carefully.
0x1	No-idle	The USBTLL module never acknowledges any IDLE request from the PRCM. This mode is safe from a module point of view as it ensures the clocks remain active; however, it is not efficient from a power-saving perspective because it does not allow the PRCM output clock to be shut off and thus the power domain to be set to a lower power state.
0x2	Smart-idle	The USBTLL module acknowledges the IDLE request basing its decision on its internal activity. Namely, the acknowledge signal is asserted only when all pending transactions, IRQs or DMA requests are treated. This is the best approach for an efficient system power management.

When configured in smart-idle mode, the USBTLL module also offers an additional granularity on USBTLL_ICLK gating. The USBHOST.USBTLL_SYSCONFIG[9:8] CLOCKACTIVITY bit is used to control the USBTLL_ICLK clock internal gating while the module is idle. Table 24-40 details the CLOCKACTIVITY settings.

Table 24-40. USBTLL Module CLOCKACTIVITY Settings

CLOCKACTIVITY Value	USBTLL_ICLK Effect	Description
0	OFF	USBTLL_ICLK is considered for generating the acknowledgment. This setting also means USBTLL_ICLK is shut down upon PRCM IDLE request.
1	ON	USBTLL_ICLK is not shut down upon PRCM IDLE request. The USBTLL module can potentially acknowledge the IDLE request without checking the internal functionalities linked to its clock.

CAUTION

The PRCM does not have any hardware means to read CLOCKACTIVITY settings. Software ensures a consistent programming between the USBTLL module CLOCKACTIVITY and PRCM USBTLL_ICLK control bit. Indeed, if the USBTLL module is disabled in the PRCM.CM_ICLKEN3_CORE register while CLOCKACTIVITY is set to 1, nothing prevents the PRCM module from asserting its IDLE request, which is acknowledged regardless of the features associated to the USBTLL module interface clock. This may lead to unpredictable behaviors.

24.2.3.2 Hardware Requests

24.2.3.2.1 Interrupt Requests

Table 24-41 lists the interrupt lines that are driven out from the high-speed USB host controller to the microprocessor unit (MPU) subsystem interrupt controller (INTC).

Table 24-41. High-Speed USB Host Subsystem Interrupts

Name	Mapping	Comments
HS USB OHCI Host Controller Interrupt		
OHCI_IRQ	M_IRQ_76	Destination is MPU subsystem interrupt controller
HS USB EHCI Host Controller Interrupt		
EHCI_IRQ	M_IRQ_77	Destination is MPU subsystem interrupt controller
USBTLL Module Interrupt		
TLL_IRQ	M_IRQ_78	Destination is MPU subsystem interrupt controller

24.2.3.2.2 IDLE Handshake Protocol

The PRCM handles an IDLE handshake protocol for the high-speed USB host controller and the USBTLL module. The IDLE handshake protocol allows the PRCM requiring the high-speed USB host controller to enter idle mode. The module acknowledges when it is ready.

24.2.3.2.3 MSTANDBY Handshake Protocol

The PRCM module handles an MSTANDBY handshake protocol for the high-speed USB host controller, which initiates the MSTANDBY handshake to inform the PRCM module when it enters standby mode and does not generate traffic on interconnect.

24.2.3.2.4 Wake-Up Request

Wake-up request signal USBHOST_SWAKEUP is generated by the high-speed USB host controller to the PRCM module. Wake-up request signal USBTLL_SWAKEUP is generated by the USBTLL module to the PRCM module.

24.2.4 High-Speed USB Host Subsystem Functional Description

This section describes the functionality of the high-speed USB host subsystem by describing the high-speed USB host controller and the USBTLL module.

24.2.4.1 High-Speed USB Host Controller Functionality

The full details of the standard OHCI and EHCI host controller APIs (implemented by the current module) are not repeated here. For more information, see the following public specifications:

- *Open Host Controller Interface (OHCI) specification for USB Release 1.0a*
- *Enhanced Host Controller Interface (EHCI) specification for USB Release 1.0*

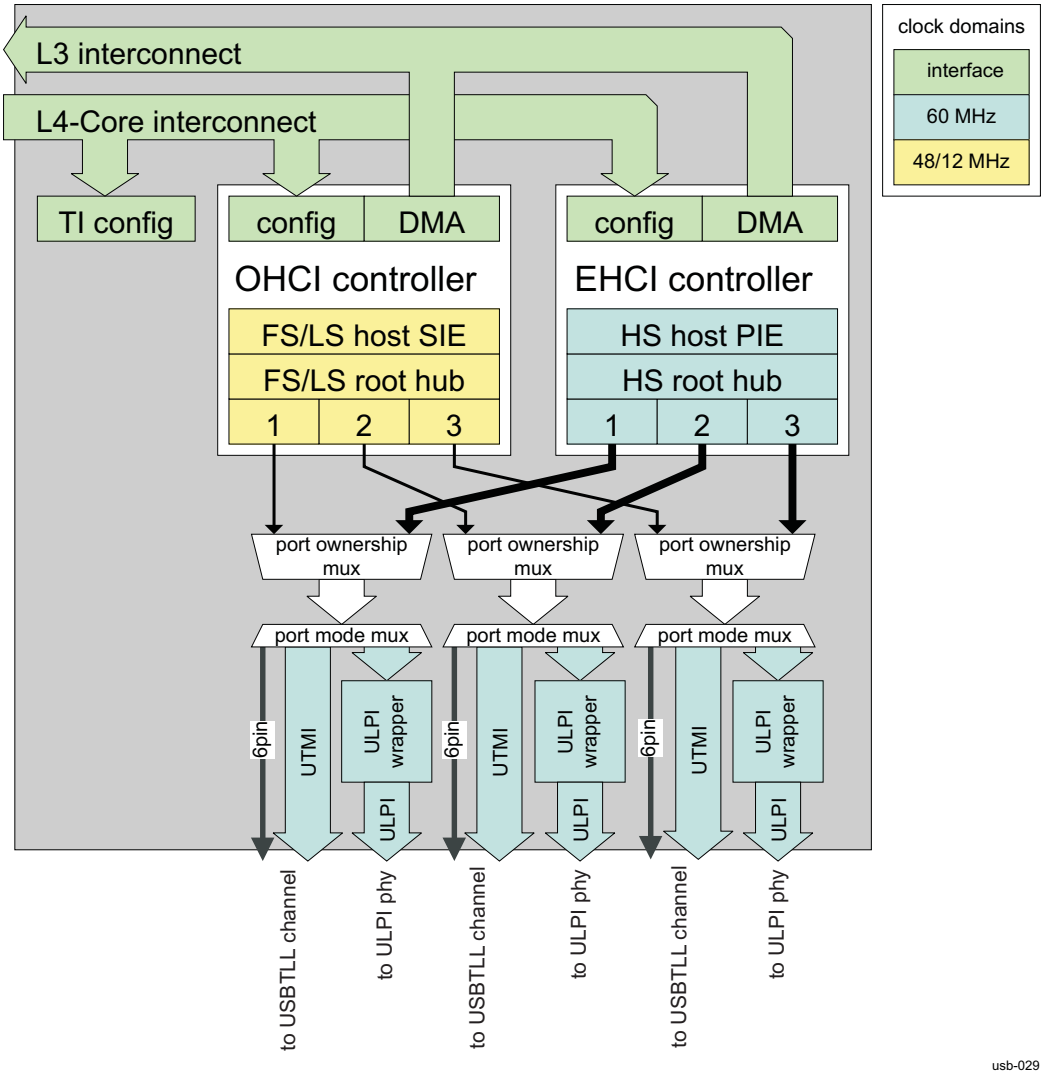
24.2.4.1.1 High-Speed USB Host Controller Architecture

Figure 24-30 shows an overview of the high-speed USB host controller internal architecture: It contains two independent, 3-port host controllers that operate in parallel: EHCI and OHCI. Each of the three external ports is owned by exactly one of the controllers at any point in time. Each port can work in several modes:

- When the port is owned by the OHCI (full-speed) host controller, the serial 6-pin interface mode is used.
- When the port is owned by the EHCI (high-speed) host controller, either the ULPI or the UTMI modes are used.

The L4-Core interconnect is used to configure the two controllers, as well as a general, TI-specific register bank. The L3 interconnect merges and arbitrates between the transactions generated by the controller respective DMA engines.

Figure 24-30. High-Speed USB Host Controller Architecture



24.2.4.1.2 OHCI Implementation Specifications

Some features of the OHCI API are optional and/or implementation-specific. The choices made in the current implementation, the high-speed USB host controller, are described below, and are reflected in the register descriptions (see Section 24.2.6.4.3, *OHCI Registers*). For all standard features, see the *Open Host Controller Interface (OHCI) specification for USB Release 1.0a*.

- USBHOST.HCFMINTERVAL[30:16] FSMPS field (FullSpeedMaxPacketSize) = 0x0000: Host will stop scheduling new packets 0 bit times before the end of the frame (that is, there is no scheduling overrun protection by default). To be updated by the software driver.
- USBHOST.HCRHDESCRIPTORA[7:0] NDP field (NumberDownstreamPorts) = 0x03 = 3 ports.
- USBHOST.HCRHDESCRIPTORA[9] NPS bit (NoPowerSwitching) = 0: Ports are power-switched by default.
- USBHOST.HCRHDESCRIPTORA[8] PSM bit (PowerSwitchingMode) = 1: Per-port power switching is supported, although PPCM default setup has all ports controlled globally (default must be = OCPM).
- USBHOST.HCRHDESCRIPTORA[31:24] POTPG field (PowerOnToPowerGood) = 0x0A = 10: Power rampup time is 10 x 2 ms = 20 ms.
- USBHOST.HCRHDESCRIPTORB[15:0] DR field (DeviceRemovable) = 0x0000: By default, no nonremovable devices (that is, devices attached to any of the ports) are removable.

- USBHOST.HCRHDESCRIPTORB[31:16] PPCM field (PortPowerControlMask) = 0x0000: By default, all ports are affected only by global power control.

24.2.4.1.3 UTMI Ports

The high-speed USB host controller supports N “downstream” ports, numbered from 1 through N. (In USB terminology, port 0 is necessarily an “upstream” port, and because the host is on “top” of the USB topological tree it has none). In the current implementation N = 3 (that is, available ports are 1, 2, 3).

The high-speed USB host controller is configured to be either in UTMI or in ULPI mode (see each port configuration with the USBHOST.UHH_HOSTCONFIG[12] P3_ULPI_BYPASS, USBHOST.UHH_HOSTCONFIG[11] P2_ULPI_BYPASS, and USBHOST.UHH_HOSTCONFIG[0] P1_ULPI_BYPASS bits).

In UTMI mode (see *USB 2.0 Transceiver Macrocell Interface specification Release 1.05*), all ports are in UTMI mode (that is, each port has its UTMI signal set broadcast the “outgoing“ packets — from the host to the peripherals) and gather the “incoming” ones (that is, from the addressed peripheral to the host). ULPI signal sets are undefined/don’t care on all ports.

In the device, the UTMI ports connect to the USBTLL module. The UTMI ports between the high-speed USB host controller and the USBTLL module are on-chip and remain invisible.

24.2.4.1.4 ULPI Ports

The high-speed USB host controller supports N “downstream” ports, numbered from 1 through N. (In USB terminology, port 0 is necessarily an “upstream” port, and because the host is on “top” of the USB topological tree it has none). In the current implementation N = 3 (that is, available ports are 1, 2, 3).

The high-speed USB host controller is configured to be either in UTMI or in ULPI mode (see each port configuration with the USBHOST.UHH_HOSTCONFIG[12] P3_ULPI_BYPASS, USBHOST.UHH_HOSTCONFIG[11] P2_ULPI_BYPASS, and USBHOST.UHH_HOSTCONFIG[0] P1_ULPI_BYPASS bits).

In ULPI mode , all ports are in ULPI mode (that is, each port has its ULPI signal set broadcast the “outgoing“ packets — from the host to the peripherals) and gather the “incoming” ones (that is, from the addressed peripheral to the host). UTMI signal sets are undefined/don’t care on all ports.

When in ULPI mode, the high-speed USB host controller is in charge of generating the (nominally 60-MHz) clock to the transceiver on the ULPI interface. This is called ULPI “input” clocking mode, because the ULPI protocol is transceiver-centric. The opposite mode, “output mode” (that is, the host receives the ULPI clock from the transceiver), is not supported and there is consequently no ULPI clock input.

In the device, the ULPI ports can only be connected directly to external transceivers. The USBTLL module is bypassed. USB traffic can be monitored directly on the USB lines.

24.2.4.1.5 Port Status

The USB port status is given through the USBHOST.UHH_HOSTCONFIG[10:8] bit field. The default value of the following bits is 1:

- USBHOST.UHH_HOSTCONFIG[8] P1_CONNECT_STATUS
- USBHOST.UHH_HOSTCONFIG[9] P2_CONNECT_STATUS
- USBHOST.UHH_HOSTCONFIG[10] P3_CONNECT_STATUS

CAUTION

These bits show the port status as connected after power on even though no USB device is connected. The USB host controller has operational status registers (for example, USBHOST.HCRHPORTSTATUS_1 for OHCI port 1) that indicate the correct port connect status. The USB driver software must read these status bits and check whether or not a port is connected. If the port is not connected, the USB driver software must reprogram the USBHOST.UHH_HOSTCONFIG bits to indicate the correct port connect status.

24.2.4.1.6 Save and Restore

The save-and-restore (SAR) mechanism can extract the hardware context of the high-speed USB host controller (after all USB activity has been suspended) before switching off (=save), save it to an external always-on memory, and reinject it later after the module has been switched on again and reset (=restore) seamlessly for the USB. Part of that context is composed of the register fields described in the current chapter. The rest of the context is composed of the "buried" flip-flops and memories (not accessible by software) like finite state-machine (FSM) states, buffer contents, and miscellaneous random logic bits.

The PRCM.PM_PWSTCTRL_USBHOSTE[4] SAVEANDRESTORE bit enables the SAR mechanism for the high-speed USB host controller (see Chapter 4, *Power, Reset, and Clock Management*). When set, the PRCM module initiates the save and/or the restore sequences at the appropriate time. When not set, the USB host is treated as a standard module, and the save/restore sequences do not occur.

24.2.4.1.7 Burst Control

To avoid buffer underflow bursts shall be enabled by writing 0x7 in USBHOST.UHH_HOSTCONFIG[4:2] and 0x0 in USBHOST.UHH_HOSTCONFIG[5] ENA_INCR_ALIGN.

24.2.4.2 USBTLL Module Functionality

The USBTLL module implements a TLL compatible with a number of USB standard interface protocols. Once the interface protocol has been selected during an initial configuration phase, USB operation should take place seamlessly (that is, as if actual transceivers were present). To ensure maximum compatibility, as many features as possible have been included, as described in the rest of this document. The basic principle is that all the software "handles" should be available and behave in a proper way, even if there is no actual functionality underneath.

The USBTLL module is integrated with the high-speed USB host controller in the device. The transceiver interfaces (UTMI ports) between the high-speed USB host controller and the USBTLL module are on-chip and remain invisible. The other transceiver interfaces go off-chip, where they can be connected to the other controllers (for example, peripherals) on another IC.

24.2.4.2.1 Channels and Ports

Following the same convention than UTMI and ULPI, the current specification is consistently PHY-centric (that is, directions are always given with respect to the transceiver emulated here by the TLL), and not with respect to the link controller: An "input" goes from the link controller to the TLL (transceiver emulator) (that is, it is an input for the USBTLL module. Reciprocally, an "output" goes from TLL (transceiver) to the link controller (that is, it is an output for the USBTLL module).

By convention, the local link controller is the controller integrated on the same IC as the USBTLL module: This is the high-speed USB host controller in the device. The remote link controller is the other controller, located off-chip (that is, on another IC). One controller is always the USB host, the other the USB peripheral, and they communicate through the USBTLL module.

A channel is defined as a independent USB path through the USBTLL module, which always converts the UTMI+ transceiver interface protocol coming from the local link controller (the high-speed USB host controller in the device). The number of channels of the USBTLL module is three in the device.

A USB port is a set of I/O signals that carry the data and control information from/to a USB line. Several port formats exist, with different capabilities. A channel has three ports. If the channel is active, two ports are active at a time, depending on the channel configuration. The mode remains static throughout USB operation. Table 24-42 summarizes the ports features.

24.2.4.2.2 Channel Architecture

Figure 24-31 shows the architecture of a single channel (of three) of the USBTLL module, and its integration in a USB system. The ports are indicated by the circled letters (A, C, and D). See the following descriptions:

- Full arrows represent parallel, synchronous, high-speed-capable interfaces.
- Line arrows represent serial, combinatorial, full-speed/low-speed-only interfaces.
- Arrows always point toward the PHY layer (actual transceiver or TLL, in yellow), away from the link controller.
- The arrow marked ULPI* represents the entire ULPI protocol (synchronous or not) except the 3-/6-pin serial TLL modes which are reoriented toward the serial TLL block.
- The USBTLL module cannot interface with a ULPI transceiver. This functionality is provided by the high-speed USB host controller.
- Top-level multiplexing is shown only for information. It is static and does not add functionality. The figure shows ULPI and serial ports implemented on different I/O pads for clarity, but a real implementation would typically reuse the same pads, because the interfaces cannot be active simultaneously

Figure 24-31. USBTLL Channel

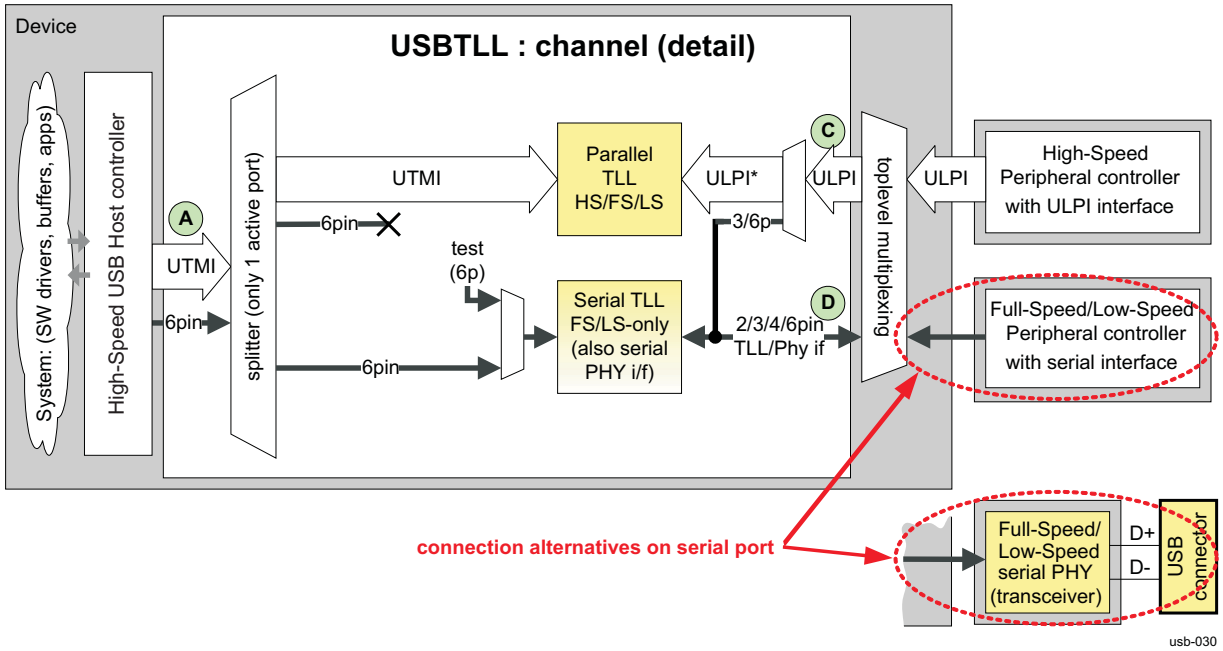


Table 24-42 summarizes the properties of each port.

Table 24-42. USBTLL Channel USB Ports

USBTLL Port	Port Description	Connect to Controller	Connect to Transceiver	Serial (Full-Speed/ Low-Speed only)	Parallel (High-Speed/ Full-Speed/Low-Speed)
A	PHY-side UTMI+	UTMI+ (L3)	No	6-pin	60-MHz UTMI
C	PHY-side ULPI	ULPI TLL	No	6-pin	60-MHz UTMI
D	Multimode Serial	6-/4-/3-/2-pin TLL	6-/4-/3-/2-pin	6-/4-/3-/2-pin TLL	No

24.2.4.2.2.1 Port A: PHY-side UTMI+ Port

Connects to the high-speed USB host controller in the device. The UTMI “local” port is used in all configurations (that is, the entire channel can be seen as a protocol converted from that port to one of the remote ports C or D).

- Compliant with UTMI+ version 1.0
- 8-data-bit, 60 MHz UTMI (HS/FS/LS-capable)
- UTMI+ Level 3 extensions
- Vcontrol/Vstatus (from UTMI)
- Serial FS/LS “6-pin” mode

24.2.4.2.2.2 Port C: PHY-Side ULPI Port

Connects to a remote (off-chip) ULPI controller through I/O pads.

- Compliant with ULPI version 1.1
- SDR and DDR ULPI capable (respectively, 8-bit/4-bit data width modes)
- Supports optional 6-pin/3-pin serial modes
- Supports optional input clocking mode

24.2.4.2.2.3 Port D: Serial Multimode Port

Connects to either a serial controller (TLL modes) or a serial transceiver (transceiver interface modes).

- Supports 6-pin (TX: DAT/SE0 or TX: DP/DM) unidirectional, 4-pin bidirectional, 3-pin bidirectional, 2-pin bidirectional modes
- All modes are supported for TLL or transceiver interface configuration.
- Supports sideband signals (pullup/down control, speed/suspend enable, etc.)

24.2.4.2.3 Channel Configuration

A channel configuration is a set of software settings that specifies the connection of two of the channel ports through the USBTLL module. USB data and control injected on one side (or port) comes out on the other side (or port) after a certain amount of processing, depending on the mode. Table 24-43, lists the modes.

All configurations connect the PHY UTMI port (attached to the high-speed USB host controller) to one of the other two ports (attached to a variety of transceivers or controllers on the pads side).

Table 24-43 describes the available modes and the software settings required for each. Channel i has the following settings:

- CHANMODE: USBHOST.TLL_CHANNEL_CONF_i[2:1] CHANMODE field
- FSLSMODE: USBHOST.TLL_CHANNEL_CONF_i[27:24] FSLSMODE field
- FSLSSERIALMODE_3PIN/6PIN: Either the ULPI PHY-side USBHOST.ULPI_INTERFACE_CTRL[1] FSLSSERIALMODE_3PIN bit or the USBHOST.ULPI_INTERFACE_CTRL[0] FSLSSERIALMODE_6PIN bit (only one can be set to 1 at a time)

Table 24-43. USBTLL Channel Configuration

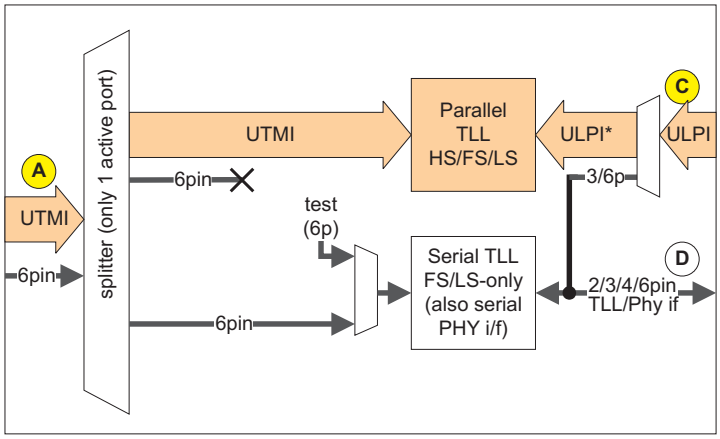
Configuration	Mode	CHAN MODE	FSLSMODE	Other Settings	Ports	Speed	Remote Port Connection
2	ULPI synchronous TLL	0	N/A	FSLSSERIALMODE_ 3PIN/6PIN = 0	A–C	HFL	ULPI link (peripheral controller)
4	Serial UTMI to serial ULPI TLL	0	N/A	FSLSSERIALMODE_ 3PIN/6PIN = 1	A–C	FL	ULPI link (peripheral controller) supporting 3-/6-pin mode
6	Serial UTMI to serial TLL	1	0x4 to 0x7; 0xA to 0xB	-	A–D	FL	Serial link (2-/3-/4-/6-pin)

Table 24-43. USBTLL Channel Configuration (continued)

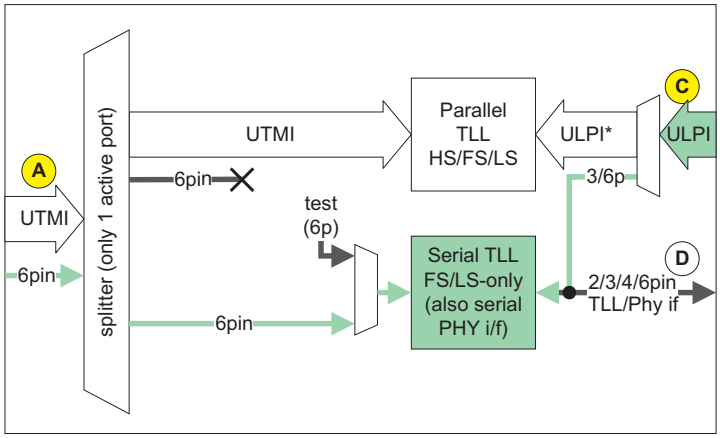
Configuration	Mode	CHAN MODE	FSLSMODE	Other Settings	Ports	Speed	Remote Port Connection
6	Serial UTMI to serial PHY	1	0x0 to 0x3	-	A–D	FL	Serial transceiver (2-/3-/4-/6-pin)

Figure 24-32. Per-Configuration Datapath Through USBTLL

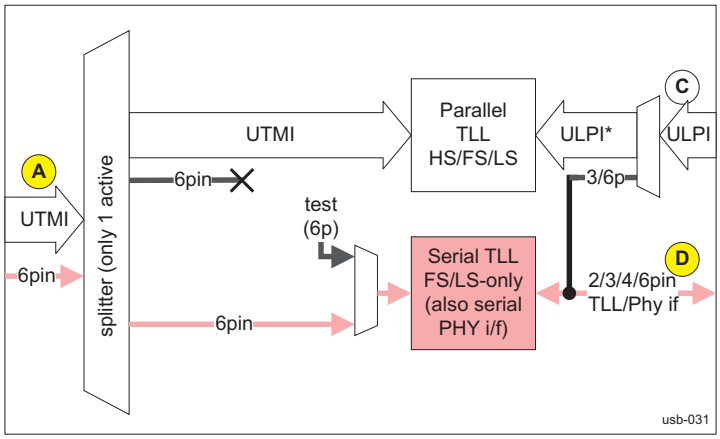
2: Sync UTMI to sync ULPI TLL config (HS/FS/LS)



4: Serial UTMI to serial ULPI TLL config (FS/LS)



6: Serial UTMI to serial ULPI, TLL or PHY config



24.2.4.2.4 VBUS Management and Emulations

In transceiver configurations, an actual USB cable is present, including an actual 5 V VBUS supply line. On the other hand, in TLL configurations, the physical USB lines are emulated and have no physical existence. This is especially true for the VBUS line, which distributes the 5-V power provided by the default host (or A-device) to the entire bus. VBUS is also used for signaling purposes, and those features must be emulated:

- A peripheral detects the presence of a host by detecting the presence of VBUS.
- USB OTG defines an elaborate voltage-sensing scheme to dynamically switch on and off VBUS (start and stop sessions). In the context of TLL, this bring no power saving compared to simple suspend.
- In particular, USB OTG uses VBUS as a wake-up source (VBUS-pulsing SRP) for the default peripheral (or B-device).

For more information on how sideband controls are integrated, see Figure 24-17 and Figure 24-18 and the related explanations.

24.2.4.2.4.1 VBUS Control and Status for Transceiver (Non-TLL) Configurations

In non-TLL modes, VBUS exists, and the problem is to propagate control and status to/from the actual VBUS manager IC (typically the transceiver itself).

Only serial transceiver configurations are concerned in the case of the high-speed USB host subsystem in the device.

24.2.4.2.4.1.1 VBUS Management in Serial Transceiver Configurations

VBUS management is not standardized in transceiver configurations. The chosen implementation is described below. See also Figure 24-17.

- VBUS control required for host and OTG operation (VBUS drive, VBUS pullup “charge”, VBUS pulldown “discharge”) is assumed to be taken care of separately from the USBTLL module (that is, by software and straight to the power IC, which can be the transceiver itself, especially in OTG cases).
- VBUS status must be sampled by the appropriate hardware (again, most of the time the transceiver itself) and reported by software to the USBTLL module, using the USBHOST.TLL_CHANNEL_CONF_i DRVVBUS and CHRGVBUS bits, as indicated in Table 24-44.

Table 24-44 lists the values to write to the USBHOST.TLL_CHANNEL_CONF_i register depending on the VBUS status observed by the transceiver on the actual VBUS line. The same register fields are also used in TLL configuration, and have been named according to that second configuration. In transceiver configurations the fields actual signification is:

- DRVVBUS: Set to 1 to report a VBUS level greater than *VBUS valid.*
- CHRGVBUS: Set to 1 to report a VBUS level greater than *Session valid*

Table 24-44. VBUS Level Software Reporting for Serial Transceiver Configuration

VBUS Status	USBHOST.TLL_CHANNEL_CONF_i[16] DRVVBUS Bit	USBHOST.TLL_CHANNEL_CONF_i[15] CHRGVBUS Bit
VBUS valid	1	1
Session valid (A/B)	0	1
Session not valid	0	0
Session end	0	0

24.2.4.2.4.2 VBUS Emulation for TLL Configurations

The TLL VBUS emulation sums up all actions on the VBUS line, obtains a voltage level, reported in the VBUS status bits following the protocol. The level depends on the immediate VBUS actions and has no memory of previous levels, whereas a real VBUS line behaves like an RC circuit and takes time to charge and discharge. This causes the following differences:

- The TLL level always jumps abruptly from session valid to session end (and back) with no transient time in between (where session is neither valid nor ended) as in real life.

- The Charge feature is used for VBUS-pulsing SRP, and is enabled long enough to go over the Session valid threshold, but without reaching VBUS valid. In the TLL the transition to Session valid is immediate, and VBUS valid is never reached even if the Charge is intentionally kept active.
- The Discharge feature is used in real life to accelerate the voltage drop of an undriven VBUS towards the session-end level. For TLL, this is therefore useless (although the UTMI input/ULPI register bit do exist, for compatibility), and always a “don’t care”

24.2.4.2.4.2.1 VBUS Emulation in ULPI TLL Modes

Table 24-45 summarizes the VBUS emulation in ULPI TLL modes. VBUS controls are writable, static PHY-side registers on the ULPI side, and input signals on the ULPI ports (port A). VBUS status bits are read-only, volatile PHY-side registers on the ULPI, and output signals on the ULPI ports (port A).

Table 24-45. Emulation of VBUS Levels for UTMI-to-ULPI TLL Mode

VBUS Controls (Actions)			VBUS Level	VBUS Status		
USBHOST.ULPI_OTG_CTRL[5] DRVVBUS Bit	USBHOST.ULPI_OTG_CTRL[4] CHRGVBUS Bit	USBHOST.ULPI_OTG_CTRL[3] DISCHRGVBUS Bit		USBHOST.ULPI_USB_INT_STATUS[1] VBUSVALID Bit	USBHOST.ULPI_USB_INT_STATUS[2] SESSVALID Bit	USBHOST.ULPI_USB_INT_STATUS[3] SESSEND Bit
1	X	X	VBUS valid	1	1	0
0	1	X	VBUS valid	0	1	0
0	0	X	Session end	0	0	1

24.2.4.2.4.2.2 VBUS Emulation in Serial TLL Modes

In serial TLL modes, VBUS status and control is implemented with ad-hoc sideband signals. See Figure 24-18.

VBUS control can be done in software, by writing to the following fields of the USBHOST.TLL_CHANNEL_CONF_i register:

- DRVVBUS: Set to 1 to drive VBUS to 5 V (for A-device or host)
- CHRGVBUS: Set to 1 to pullup VBUS (for SRP)
- There is no pulldown (discharge) control, because the emulated VBUS has no latency and VBUS level goes to the session end level as soon as it is neither driven nor pulled up.

Alternatively, VBUS drive can also be hardware-controlled through a dedicated input. (DRVVBUS register bit and input signal are actually ORed internally.)

VBUS status is available on dedicated output signals. If those outputs are not available at top level, a software alternative is to use the voltage status reported on the local controller (the high-speed USB host controller in the device) interface (through the standard UTMI+ sideband signals) and to pass it to the remote controller (a peripheral controller), by means of an ad hoc software-controller interface other than the USB itself. This is based on the fact that the level of VBUS is the same on both extremities of the bus (that is, it does not matter which side does the measurement).

24.2.4.2.5 Multimode Serial Port

The multimode serial port requires six bidirectional I/O pads to support all eight defined modes (selected in the USBHOST.TLL_CHANNEL_CONF_i[27:24] FSLSMODE field when field CHANMODE = 0x1 = UTMI-to-serial). Those modes are full-speed/low-speed only (that is, high-speed is not supported over a serial interface).

The pads are named TXEN, TXDAT, TXSE0, RXRCV, and RXDM after their functionality in standard 6-pin mode (mode 0). Each pad has an input, output, and output enable signal associated to it on the USBTLL entity.

Table 24-46 shows the functionality of each pad in each mode. USBTLL outputs are shown in yellow, inputs in blue, and bidirectional pads in green.

Table 24-46. Serial Mode Description, Signal Functionality

Usual Name	6-Pin Mode	6-Pin Mode (Alt)	3-Pin Mode	4-Pin Mode	6-Pin TLL Mode	6-Pin TLL (Alt) Mode	3-Pin TLL Mode	4-Pin TLL Mode	2-Pin TLL Mode	2-Pin TLL (Alt) Mode
USBHOST.TLL_CHANNEL_CONF_i[27:24] FSLSMODE field	0x0	0x1	0x2	0x3	0x4	0x5	0x6	0x7	0xA	0xB
TX encoding	DAT/SE0	DP/DM	DAT/SE0	DP/DM	DAT/SE0	DP/DM	DAT/SE0	DP/DM	DAT/SE0	DP/DM
RX encoding	DP/DM/RCV	DP/DM/RCV	DAT/SE0	DP/DM/RCV	DP/DM/RCV	DP/DM/RCV	DAT/SE0	DP/DM/RCV	DAT/SE0	DP/DM
Pin usage	Unidirectional	Unidirectional	Bidirectional	Bidirectional	Unidirectional	Unidirectional	Bidirectional	Bidirectional	Bidirectional	Bidirectional
Pin count	6	6	3	4	6 or 5[1]	6 or 5[1]	3	4 or 3[2]	2	2
I/O Pad Function Per Mode										
TXEN	TX Enable	TX Enable	TX Enable	TX Enable	TX Enable	TX Enable	TX Enable	TX Enable	N/C	N/C
TXDAT	TX Diff Data	TX SE Plus Data	TX/RX Diff Data	TX/RX SE Plus Data	TX Diff Data	TX SE Plus Data	TX/RX Diff Data	TX/RX Diff Data	TX/RX Diff Data	TX/RX SE Plus Data
TXSE0	TX force SE0	TX SE Minus Data	TX/RX force SE0	TX/RX SE Minus Data	TX force SE0	TX SE Minus Data	TX/RX force SE0	TX/RX force SE0	TX/RX force SE0	TX/RX SE Minus Data
RXRCV	RX Diff Data	RX Diff Data	N/C	RX Diff Data	RX Diff Data	RX Diff Data	N/C	RX Diff Data	N/C	N/C
RXDM	RX SE Plus Data	RX SE Plus Data	N/C	N/C	RX SE Plus Data	RX SE Plus Data	N/C	N/C	N/C	N/C
RXDM	RX SE Minus Data	RX SE Minus Data	N/C	N/C	RX SE Minus Data	RX SE Minus Data	N/C	N/C	N/C	N/C

(1) RXRCV and RXDM carry the same info: RXDM can drive both inputs of the remote controller and RXRCV kept unused
(2) Same remark on TXDAT (for outputs) and RXRCV: TXDAT only is enough

24.2.4.2.6 Attach/Connect Emulation for Serial TLL Modes

This section applies to all serial TLL modes:

- In UTMI-to-serial mode (USBHOST.TLL_CHANNEL_CONF_i[2:1] CHANMODE field = 0x1) for all TLL values of USBHOST.TLL_CHANNEL_i[27:24] FSLSMODE field (0x4 to 0x7; 0xA to 0xB)
- In UTMI-to-ULPI TLL mode (USBHOST.TLL_CHANNEL_CONF_i[2:1] CHANMODE field = 0x0) when the ULPI bus is switched to 6-pin serial or 3-pin serial modes

In those modes, the USB bus lines are emulated by USBTLL internal logic, and are never available on the outside. The pullup/pulldown actions described in the USB specification cannot be applied directly, and the USB cable cannot be physically attached.

Because serial modes do not specify a standard format for those sideband settings, a custom software-controlled one was implemented:

- USBHOST.TLL_CHANNEL_CONF_i[4] TLLATTACH bit emulates the physical attachment of the two controllers through a TLL cable.
  - When this bit is cleared, the local controller RX path only shows the local controller (the high-speed USB host controller) actions on the bus: TX driving, pullups, pulldowns (see below). The same thing applies for the remote controller RX path (except that test override is not available).
  - As soon as the bit is set, the actions of both sides are applied to the same bus and are resolved, similar to a real bus. The RX path for both sides shows the same bus state.
- USBHOST.TLL_CHANNEL_CONF_i[5] TLLCONNECT bit emulates the USB electrical connect (that is,

the pullup by the USB peripheral of one of the two USB lines [by a 1.5KOhm resistor]), which causes the linestate to transition from SE0 to J, which is detected by the USB host. The register bit is ORed with a USBTLL module input signal — the connect control can be software (L4-Core interconnect write access) or hardware (input level). The speed of the connection is determined by the TLLFULLSPEED bit below.

- USBHOST.TLL_CHANNEL_CONF_i[6] TLLFULLSPEED bit determines the speed (full or low) of the USB connect to be emulated. The connect enable (controlled as defined above) results in the pulling-up of either D+ (1 = full speed) or D- (0 = low speed): See Table 24-47.
- The 15kOhm pulldowns are implicit: Because they are supposed to be turned on at least on the host side of the bus, they do not require an additional control.

Note: Sideband control and status actions like pullups are included in parallel (that is, nonserial) standards (UTMI, ULPI), and do not require any custom additions.

Table 24-47. Pullup Enable Emulation in Serial TLL Modes

USBHOST.TLL_CHANNEL_CONF_i Fields		Input Signal	Resulting TLL Pullup Emulation	
TLLFULLSPEED	TLLCONNECT	USB State	D+ Pullup	D– Pullup
1	0	Full-speed unconnected	Off	Off
1	1	Full-speed connected	on	Off
0	0	Low-speed unconnected	Off	Off
0	1	Low-speed connected	Off	On

24.2.4.2.7 Save and Restore

The SAR mechanism can extract the hardware context of the USBTLL module (after all USB activity has been suspended) before switching off (=save), save it to an external always-on memory, and reinject it later after the module has been switched on again and reset (=restore) seamlessly for the USB. Part of that context is composed of the register fields described in the current chapter. The rest of the context is composed of the buried flip-flops and memories (not accessible by software) like FSM states, buffer contents, and miscellaneous random logic bits.

The PRCM.PM_PWSTCTRL_CORE[4] SAVEANDRESTORE bit enables the SAR mechanism for the USBTLL module (see Chapter 4, *Power, Reset, and Clock Management*). When set, the PRCM module initiates the save and/or the restore sequences at the appropriate time. When not set, the USB host is treated as a standard module, and the save/restore sequences do not occur.

Table 24-48 lists the USBTLL registers impacted by the SAR context.

Note: Because all addresses give access to the same physical register (that is, to the same piece of context), the ULPI registers with multiple accesses (write, set, clear) are listed only once in the table.

Table 24-48. USBTLL Registers Impacted by the SAR Context

Register Name	Comments on SAR Policy
USBTLL_SYSCONFIG	Except the SOFTRESET bit (write-only)
USBTLL_IRQENABLE	-
TLL_SHARED_CONF	Except the FCLK_REQ bit
TLL_CHANNEL_CONF_i	Except the FSLSLINESTATE field
ULPI_FUNCTION_CTRL_i	-
ULPI_INTERFACE_CTRL_i	-
ULPI_OTG_CTRL_i	-

Table 24-48. USBTLL Registers Impacted by the SAR Context (continued)

Register Name	Comments on SAR Policy
ULPI_USB_INT_EN_RISE_i	-
ULPI_USB_INT_EN_FALL_i	-
ULPI_USB_INT_STATUS_i	-
ULPI_VENDOR_INT_EN_i	-
ULPI_VENDOR_INT_STATUS_i	-

24.2.5 High-Speed USB Host Subsystem Basic Programming Model

24.2.5.1 Selecting and Configuring USB Connectivity

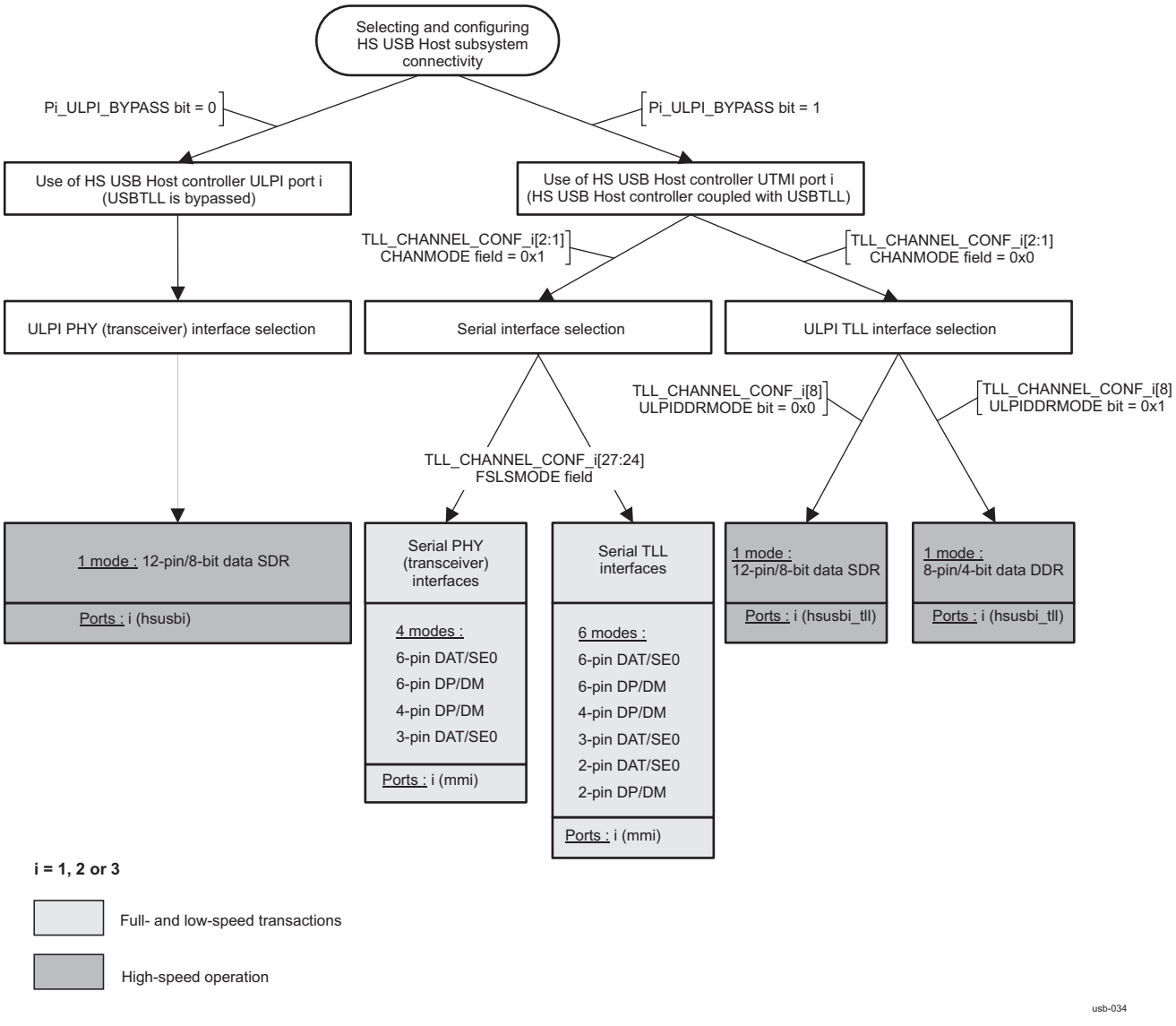
Perform the following steps to select the desired USB connectivity and configure the device accordingly.

The high-speed USB host subsystem provides three kinds of interfaces for connection:

- ULPI PHY interfaces for high-speed data transactions (up to 480 Mbps)
- Serial interfaces for full- and low-speed data transactions (up to 12 Mbps)
- ULPI TLL interfaces for high-speed data transactions (up to 480 Mbps)

Figure 24-33 shows how to select and configure the high-speed USB host subsystem connectivity.

Figure 24-33. Selecting and Configuring High-Speed USB Host Subsystem Connectivity



Notes:

- When the USBHOST.UHH_HOSTCONFIG[12] P3_ULPI_BYPASS bit is 0 (the port 3 ULPI transceiver interface selection), only the 12-pin/8-bit data SDR version of the ULPI interface mode is supported for port 3.
- When the USBHOST.UHH_HOSTCONFIG[11] P2_ULPI_BYPASS bit is 0, (the port 2 ULPI transceiver interface selection), only the 12-pin/8-bit data SDR version of the ULPI interface mode is supported for port 2.
- When the USBHOST.UHH_HOSTCONFIG[0] P1_ULPI_BYPASS bit is 0 (the port 1 ULPI transceiver interface selection), only the 12-pin/8-bit data SDR version of the ULPI interface mode is supported for port 1.
- When the USBHOST.UHH_HOSTCONFIG[12] P3_ULPI_BYPASS bit is 1, (the ULPI TLL interface selection and serial interface selection), there is no restriction and port 3 can be configured in any ULPI TLL or serial mode.
- When the USBHOST.UHH_HOSTCONFIG[11] P2_ULPI_BYPASS bit is 1 (the ULPI TLL interface selection and serial interface selection), there is no restriction and port 2 can be configured in any ULPI TLL or serial mode.
- When the USBHOST.UHH_HOSTCONFIG[0] P1_ULPI_BYPASS bit is 1 (ULPI TLL interface selection and serial interface selection), there is no restriction and port 1 can be configured in any ULPI TLL or serial mode.

24.2.5.1.1 ULPI Interface Selection

The high-speed USB host subsystem supports the following modes with the ULPI interfaces:

- External USB transceiver
  - ULPI interfaces: 12-pin/8-bit data version
- TLL
  - ULPI TLL interfaces: 12-pin/8-bit data version or 8-pin/4-bit data version

24.2.5.1.1.1 Transceiver Interfaces

The USBTLL module is bypassed and the high-speed USB host controller ports 1 and 2 are connected directly to external transceivers.

The high-speed USB host subsystem supports only the 12-pin/8-bit data SDR version of the ULPI interface mode. The high-speed USB host controller uses its ULPI ports (UTMI ports cannot be used). The USBHOST.UHH_HOSTCONFIG[12] P3_ULPI_BYPASS, USBHOST.UHH_HOSTCONFIG[11] P2_ULPI_BYPASS, or USBHOST.UHH_HOSTCONFIG[0] P1_ULPI_BYPASS bit must be cleared to 0, depending on the port used in ULPI mode.

24.2.5.1.1.2 TLL

The high-speed USB host controller is coupled with the USBTLL module to compose the ULPI TLL interface modes.

The high-speed USB host controller uses its UTMI ports (bypassing the ULPI ports). The USBHOST.UHH_HOSTCONFIG[12] P3_ULPI_BYPASS, USBHOST.UHH_HOSTCONFIG[11] P2_ULPI_BYPASS, or USBHOST.UHH_HOSTCONFIG[0] P1_ULPI_BYPASS bit must be set to 1, depending on the port used in UTMI mode.

At the USBTLL module level, a channel configuration for ULPI TLL interfaces must be set (see Section 24.2.4.2.3, *Channel Configurations*).

Two configurations are supported for ULPI TLL interfaces in the device:

- Configuration 2: ULPI synchronous TLL mode
- Configuration 4: Serial UTMI to serial ULPI TLL mode

In both configurations, the USBHOST.TLL_CHANNEL_CONF_i[2:1] CHANMODE field must be cleared to 0x0 (UTMI-to-ULPI TLL mode).

The selection of the ULPI TLL interface version, 12-pin/8-bit data version (SDR mode) or 8-pin/4-bit data version (DDR mode) is done through the USBHOST.TLL_CHANNEL_CONF_i[8] ULPIDDRMODE bit (0: SDR mode; 1: DDR mode).

24.2.5.1.2 Serial Interface Selection

The high-speed USB host subsystem supports the following modes with the serial interfaces:

- External USB transceiver configurations
  - Serial 6-pin PHY (transceiver) interfaces: 6-pin unidirectional (TX: DAT/SE0 or TX: DP/DM), 4-pin bidirectional and 3-pin bidirectional modes
- TLL configurations
  - Serial 6-pin TLL interfaces: 6-pin unidirectional (TX: DAT/SE0 or TX: DP/DM), 4-pin bidirectional, 3-pin bidirectional, and 2-pin bidirectional modes

The high-speed USB host controller is coupled with the USBTLL module to compose the serial interface modes.

The high-speed USB host controller uses its UTMI ports (bypassing the ULPI ports). The USBHOST.UHH_HOSTCONFIG[12] P3_ULPI_BYPASS, USBHOST.UHH_HOSTCONFIG[11] P2_ULPI_BYPASS, or USBHOST.UHH_HOSTCONFIG[0] P1_ULPI_BYPASS bit must be set to 1, depending on the port. In UTMI mode, all ports of the high-speed USB host controller are in UTMI mode.

At the USBTLL module level, a channel configuration for serial interfaces must be set (see Section 24.2.4.2.3, *Channel Configurations*).

One configuration is supported for serial interfaces in the device, the Configuration 6 including two modes:

- Serial UTMI to serial TLL
- Serial UTMI to serial PHY

The multimode-mode serial interface mode selection is done through the USBHOST.TLL_CHANNEL_CONF_i[27:24] FSLSMODE field only when the main channel mode is serial (USBHOST.TLL_CHANNEL_CONF_i[2:1] CHANMODE field = 0x1 = UTMI-to-serial mode).

Table 24-49. USB Connectivity Mode Description

Usual Name	6-Pin Mode	6-Pin Mode (Alt)	3-Pin Mode	4-Pin Mode	6-Pin TLL Mode	6-Pin TLL Mode (Alt)	3-Pin TLL Mode	4-Pin TLL Mode	2-Pin TLL Mode	2-Pin TLL (Alt) Mode
USBHOST.TLL_CHANNEL_CONF_i[27:24] FSLSMODE field	0x0	0x1	0x2	0x3	0x4	0x5	0x6	0x7	0xA	0xB
TX encoding	DAT/SE0	DP/DM	DAT/SE0	DP/DM	DAT/SE0	DP/DM	DAT/SE0	DP/DM	DAT/SE0	DP/DM
RX encoding	DP/DM/RCV	DP/DM/RCV	DAT/SE0	DP/DM/RCV	DP/DM/RCV	DP/DM/RCV	DAT/SE0	DP/DM/RCV	DAT/SE0	DP/DM
Pin usage	Unidirectional	Unidirectional	Bidirectional	Bidirectional	Unidirectional	Unidirectional	Bidirectional	Bidirectional	Bidirectional	Bidirectional
Pin count	6	6	3	4	6 or 5[(1)]	6 or 5[(1)]	3	4 or 3[(2)]	2	2

(1) RxRCV and RxDP carry the same info: RxDP can drive both inputs of the remote controller and RxRCV kept unused
(2) Same remark on TXDAT (for outputs) and RxRCV: TXDAT only is enough

24.2.5.2 USBTLL Registers

The USBTLL module contains two types of software-programmable registers.

24.2.5.2.1 TLL Control and Status Registers

These 32-bit registers configure the various channels. These registers are accessed by the MPU through the L4-Core interconnect. They are used mostly before the actual USB activity starts. These registers are:

- Standard registers for revision number, IRQ, clocking management, etc.
- TLL-specific registers

24.2.5.2.2 ULPI PHY-Side Registers

Each TLL channel emulates a ULPI transceiver and accordingly contains this set of 8-bit PHY-side registers, per ULPI specification. Those registers are:

- All ULPI-mandatory standard registers and fields
- A selection of ULPI-optional standard registers and fields, when relevant to the TLL context
- Vendor-specific registers, mapped at the addresses specified for that purpose in ULPI specification

Those registers are accessed by the external (that is, off-chip) link controller over the ULPI port of each channel, in the 0x100-byte ULPI address space, using the ULPI register access protocol.

Those registers are accessible by the L4-Core interconnect: The ULPI register sets of all channels are mapped side by side in the upper part of the L4-Core interconnect address space, where they can be accessed through byte accesses. In case of conflict between the two access modes, the access over ULPI will have priority, but both accesses will eventually complete correctly. For normal USB activity, all register accesses are expected to go over ULPI, and register changes caused by L4-Core interconnect accesses could compromise proper USB operation. The L4-Core interconnect access port is intended for:

- Miscellaneous test and debug
- Nonintrusive observation of ongoing USB operations (test)
- Context restore: During USB suspend periods, the USBTLL module can be switched off to save power. Upon resume, the ULPI register contents have been lost and can be restored over L4-Core interconnect before USB operations restarts, provided they have been saved elsewhere beforehand.

24.2.6 High-Speed USB Host Subsystem Register Manual

24.2.6.1 USBTLL ULPI PHY-Side Register Space

Each ULPI port emulates a separate ULPI transceiver and as such gives access to a single set of ULPI PHY-side registers, mapped in a separate ULPI register space as specified in the ULPI specification. The ULPI protocol defines two register access methods: Immediate and extended.

- The immediate space maps ULPI PHY-side registers in a 0x40- (64-) byte space (address is 6 bits wide). All ULPI PHY-side registers implemented in the USBTLL implementation are in the immediate space.
- The extended register space maps all ULPI PHY-side registers in a 0x100- (256-) byte space (address is 8 bits wide). The immediate space is remapped at the bottom of the extended space (that is, the extended access method can be used to access any ULPI register).

An access is recognized as extended by first pointing to a reserved dummy address 0x2F (EXTENDED_SET_ACCESS in Table 24-50). Immediate-mode accesses to this address over the ULPI interface are forbidden by the protocol and the USBTLL behavior is then undefined. Extended accesses to this address have no effect.

Some physical registers are accessible at more than one address, where write accesses perform different actions on the register value: (over-)write, set, clear. A read to any of the addresses returns the register value. The names of the set and clear registers are the write name postfixed with respectively _SET and _CLR. The register fields are described only once, at the write address (see Section 24.2.6.4, *EHCI Registers*).

Note: Some ULPI registers are cleared upon read.

Table 24-50. ULPI Registers Mapping Summary (For a Single ULPI Port)

Register Name	Type	Read Action	Write Action	Address Offset
VENDOR_ID_LO	R	-	-	0x0000 0000
VENDOR_ID_HI	R	-	-	0x0000 0001
PRODUCT_ID_LO	R	-	-	0x0000 0002
PRODUCT_ID_HI	R	-	-	0x0000 0003
FUNCTION_CTRL	RW	-	Overwrite	0x0000 0004
FUNCTION_CTRL_SET	RW	-	Set if 1	0x0000 0005
FUNCTION_CTRL_CLR	RW	-	Clear if 1	0x0000 0006
INTERFACE_CTRL	RW	-	Overwrite	0x0000 0007
INTERFACE_CTRL_SET	RW	-	Set if 1	0x0000 0008
INTERFACE_CTRL_CLR	RW	-	Clear if 1	0x0000 0009
OTG_CTRL	RW	-	Overwrite	0x0000 000A
OTG_CTRL_SET	RW	-	Set if 1	0x0000 000B
OTG_CTRL_CLR	RW	-	Clear if 1	0x0000 000C
USB_INT_EN_RISE	RW	-	Overwrite	0x0000 000D
USB_INT_EN_RISE_SET	RW	-	Set if 1	0x0000 000E
USB_INT_EN_RISE_CLR	RW	-	Clear if 1	0x0000 000F
USB_INT_EN_FALL	RW	-	Overwrite	0x0000 0010
USB_INT_EN_FALL_SET	RW	-	Set if 1	0x0000 0011
USB_INT_EN_FALL_CLR	RW	-	Clear if 1	0x0000 0012
USB_INT_STATUS	R	-	-	0x0000 0013
USB_INT_LATCH	R	Clear	-	0x0000 0014
DEBUG	R	-	-	0x0000 0015
SCRATCH_REGISTER	RW	-	Overwrite	0x0000 0016

Table 24-50. ULPI Registers Mapping Summary (For a Single ULPI Port) (continued)

Register Name	Type	Read Action	Write Action	Address Offset
SCRATCH_REGISTER_SET	RW	-	Set if 1	0x0000 0017
SCRATCH_REGISTER_CLR	RW	-	Clear if 1	0x0000 0018
EXTENDED_SET_ACCESS	Reserved	-	N/A	0x0000 002F
UTMI_VCONTROL_EN	RW	-	Overwrite	0x0000 0030
UTMI_VCONTROL_EN_SET	RW	-	Set if 1	0x0000 0031
UTMI_VCONTROL_EN_CLR	RW	-	Clear if 1	0x0000 0032
UTMI_VCONTROL_STATUS	RW	-	Overwrite	0x0000 0033
UTMI_VCONTROL_LATCH	R	Clear	-	0x0000 0034
UTMI_VSTATUS	RW	-	Overwrite	0x0000 0035
UTMI_VSTATUS_SET	RW	-	Set if 1	0x0000 0036
UTMI_VSTATUS_CLR	RW	-	Clear if 1	0x0000 0037
USB_INT_LATCH_NOCLR	R	-	-	0x0000 0038
VENDOR_INT_EN	RW	-	Overwrite	0x0000 003B
VENDOR_INT_EN_SET	RW	-	Set if 1	0x0000 003C
VENDOR_INT_EN_CLR	RW	-	Clear if 1	0x0000 003D
VENDOR_INT_STATUS	R	-	-	0x0000 003E
VENDOR_INT_LATCH	R	Clear	-	0x0000 003F

24.2.6.2 L4-Core Interconnect Register Space

Table 24-51 lists the base address and address space for the high-speed USB host subsystem.

Table 24-51. High-Speed USB Host Subsystem Instance Summary

Module Name	Base Address (hex)	Size
USBTLL	0x4806 2000	4096 bytes
UHH_config	0x4806 4000	1024 bytes
OHCI	0x4806 4400	1024 bytes
EHCI	0x4806 4800	1024 bytes

24.2.6.3 High-Speed USB Host Subsystem Register Mapping Summary

The USBTLL single L4-Core interconnect gives access to both the TLL control and status registers, and the ULPI PHY-side registers.

Table 24-52 lists all the USBTLL registers mapped by the L4-Core interconnect, for the maximum 8-channel configuration.

CAUTION

On ULPI PHY-side register access over L4-Core interconnect:

ULPI registers are byte-sized, and can only be accessed in this size. Attempts to access them over L4-Core interconnect using any other data size (16- or 32-bit) will complete without error (or any other warning), but will result in undefined behaviors.

The following cases can cause problems:

- If the ULPI register contents are defined as static (nonvolatile) by the software, a cache update may result in a burst of 32-bit access, with unwanted consequences.
- Some registers have adjacent overwrite, set, and clear addresses. An oversized write access could clear and set the same bit, which can have several results.
- Some registers are cleared on read: Oversized read accesses to adjacent memory locations could cause unwanted clears.

Table 24-52. USBTLL Registers Mapping Summary (L4-Core Interconnect Register Space)

Register Name	Type	Register Width (Bits)	Address Offset	Physical Address
USBTLL_REVISION	R	32	0x0000 0000	0x4806 2000
USBTLL_SYSCONFIG	RW	32	0x0000 0010	0x4806 2010
USBTLL_SYSSTATUS	R	32	0x0000 0014	0x4806 2014
USBTLL_IRQSTATUS	RW	32	0x0000 0018	0x4806 2018
USBTLL_IRQENABLE	RW	32	0x0000 001C	0x4806 201C
TLL_SHARED_CONF	RW	32	0x0000 0030	0x4806 2030
TLL_CHANNEL_CONF_i (1)	RW	32	0x0000 0040 + (0x04 * i)	0x4806 2040 + (0x04 * i)
ULPI_VENDOR_ID_LO_i (1)	R	8	0x0000 0800 + (0x100 * i)	0x4806 2800 + (0x100 * i)
ULPI_VENDOR_ID_HI_i (1)	R	8	0x0000 0001 + (0x100 * i)	0x4806 2801 + (0x100 * i)
ULPI_PRODUCT_ID_LO_i (1)	R	8	0x0000 0002 + (0x100 * i)	0x4806 2802 + (0x100 * i)
ULPI_PRODUCT_ID_HI_i (1)	R	8	0x0000 0003 + (0x100 * i)	0x4806 2803 + (0x100 * i)
ULPI_FUNCTION_CTRL_i (1)	RW	8	0x0000 0004 + (0x100 * i)	0x4806 2804 + (0x100 * i)
ULPI_FUNCTION_CTRL_SET_i (1)	RW	8	0x0000 0005 + (0x100 * i)	0x4806 2805 + (0x100 * i)
ULPI_FUNCTION_CTRL_CLR_i (1)	RW	8	0x0000 0006 + (0x100 * i)	0x4806 2806 + (0x100 * i)
ULPI_INTERFACE_CTRL_i (1)	RW	8	0x0000 0007 + (0x100 * i)	0x4806 2807 + (0x100 * i)
ULPI_INTERFACE_CTRL_SET_i (1)	RW	8	0x0000 0008 + (0x100 * i)	0x4806 2808 + (0x100 * i)
ULPI_INTERFACE_CTRL_CLR_i (1)	RW	8	0x0000 0009 + (0x100 * i)	0x4806 2809 + (0x100 * i)
ULPI_OTG_CTRL_i (1)	RW	8	0x0000 000A + (0x100 * i)	0x4806 280A + (0x100 * i)
ULPI_OTG_CTRL_SET_i (1)	RW	8	0x0000 000B + (0x100 * i)	0x4806 280B + (0x100 * i)
ULPI_OTG_CTRL_CLR_i (1)	RW	8	0x0000 000C + (0x100 * i)	0x4806 280C + (0x100 * i)
ULPI_USB_INT_EN_RISE_i (1)	RW	8	0x0000 000D + (0x100 * i)	0x4806 280D + (0x100 * i)
ULPI_USB_INT_EN_RISE_SET_i (1)	RW	8	0x0000 000E + (0x100 * i)	0x4806 280E + (0x100 * i)
ULPI_USB_INT_EN_RISE_CLR_i (1)	RW	8	0x0000 000F + (0x100 * i)	0x4806 280F + (0x100 * i)
ULPI_USB_INT_EN_FALL_i (1)	RW	8	0x0000 0010 + (0x100 * i)	0x4806 2810 + (0x100 * i)

(1) i = 0 to 2

Table 24-52. USBTLL Registers Mapping Summary (L4-Core Interconnect Register Space) (continued)

Register Name	Type	Register Width (Bits)	Address Offset	Physical Address
ULPI_USB_INT_EN_FALL_SET_i (1)	RW	8	0x0000 0011 + (0x100 * i)	0x4806 2811 + (0x100 * i)
ULPI_USB_INT_EN_FALL_CLR_i (1)	RW	8	0x0000 0012 + (0x100 * i)	0x4806 2812 + (0x100 * i)
ULPI_USB_INT_STATUS_i (1)	R	8	0x0000 0013 + (0x100 * i)	0x4806 2813 + (0x100 * i)
ULPI_USB_INT_LATCH_i (1)	R	8	0x0000 0014 + (0x100 * i)	0x4806 2814 + (0x100 * i)
ULPI_DEBUG_i (1)	R	8	0x0000 0015 + (0x100 * i)	0x4806 2815 + (0x100 * i)
ULPI_SCRATCH_REGISTER_i (1)	RW	8	0x0000 0016 + (0x100 * i)	0x4806 2816 + (0x100 * i)
ULPI_SCRATCH_REGISTER_SET_i (1)	RW	8	0x0000 0017 + (0x100 * i)	0x4806 2817 + (0x100 * i)
ULPI_SCRATCH_REGISTER_CLR_i (1)	RW	8	0x0000 0018 + (0x100 * i)	0x4806 2818 + (0x100 * i)
ULPI_EXTENDED_SET_ACCESS_i (1)	Reserved	8	0x0000 002F + (0x100 * i)	0x4806 282F + (0x100 * i)
ULPI_UTMI_VCONTROL_EN_i (1)	RW	8	0x0000 0030 + (0x100 * i)	0x4806 2830 + (0x100 * i)
ULPI_UTMI_VCONTROL_EN_SET_i (1)	RW	8	0x0000 0031 + (0x100 * i)	0x4806 2831 + (0x100 * i)
ULPI_UTMI_VCONTROL_EN_CLR_i (1)	RW	8	0x0000 0032 + (0x100 * i)	0x4806 2832 + (0x100 * i)
ULPI_UTMI_VCONTROL_STATUS_i (1)	RW	8	0x0000 0033 + (0x100 * i)	0x4806 2833 + (0x100 * i)
ULPI_UTMI_VCONTROL_LATCH_i (1)	R	8	0x0000 0034 + (0x100 * i)	0x4806 2834 + (0x100 * i)
ULPI_UTMI_VSTATUS_i (1)	RW	8	0x0000 0035 + (0x100 * i)	0x4806 2835 + (0x100 * i)
ULPI_UTMI_VSTATUS_SET_i (1)	RW	8	0x0000 0036 + (0x100 * i)	0x4806 2836 + (0x100 * i)
ULPI_UTMI_VSTATUS_CLR_i (1)	RW	8	0x0000 0037 + (0x100 * i)	0x4806 2837 + (0x100 * i)
ULPI_USB_INT_LATCH_NOCLR_i (1)	R	8	0x0000 0038 + (0x100 * i)	0x4806 2838 + (0x100 * i)
ULPI_VENDOR_INT_EN_i (1)	RW	8	0x0000 003B + (0x100 * i)	0x4806 283B + (0x100 * i)
ULPI_VENDOR_INT_EN_SET_i (1)	RW	8	0x0000 003C + (0x100 * i)	0x4806 283C + (0x100 * i)
ULPI_VENDOR_INT_EN_CLR_i (1)	RW	8	0x0000 003D + (0x100 * i)	0x4806 283D + (0x100 * i)
ULPI_VENDOR_INT_STATUS_i (1)	R	8	0x0000 003E + (0x100 * i)	0x4806 283E + (0x100 * i)
ULPI_VENDOR_INT_LATCH_i (1)	R	8	0x0000 003F + (0x100 * i)	0x4806 283F + (0x100 * i)

Table 24-53, Table 24-54, and Table 24-55 list the high-speed USB host controller registers.

Table 24-53. UHH_config Registers Mapping Summary

Register Name	Type	Register Width (Bits)	Address Offset	Physical Address
UHH_REVISION	R	32	0x0000 0000	0x4806 4000
UHH_SYSCONFIG	RW	32	0x0000 0010	0x4806 4010
UHH_SYSSTATUS	R	32	0x0000 0014	0x4806 4014
UHH_HOSTCONFIG	RW	32	0x0000 0040	0x4806 4040
UHH_DEBUG_CSR	RW	32	0x0000 0044	0x4806 4044

Table 24-54. OHCI Registers Mapping Summary

Register Name	Type	Register Width (Bits)	Address Offset	Physical Address
HCREVISION	R	32	0x0000 0000	0x4806 4400
HCCONTROL	RW	32	0x0000 0004	0x4806 4404
HCCOMMANDSTATUS	RW	32	0x0000 0008	0x4806 4408
HCINTERRUPTSTATUS	RW	32	0x0000 000C	0x4806 440C
HCINTERRUPTENABLE	RW	32	0x0000 0010	0x4806 4410

Table 24-54. OHCI Registers Mapping Summary (continued)

Register Name	Type	Register Width (Bits)	Address Offset	Physical Address
HCINTERRUPTDISABLE	RW	32	0x0000 0014	0x4806 4414
HCHCCA	RW	32	0x0000 0018	0x4806 4418
HCPERIODCURRENTED	R	32	0x0000 001C	0x4806 441C
HCCONTROLHEADED	RW	32	0x0000 0020	0x4806 4420
HCCONTROLCURRENTED	RW	32	0x0000 0024	0x4806 4424
HCBULKHEADED	RW	32	0x0000 0028	0x4806 4428
HCBULKCURRENTED	RW	32	0x0000 002C	0x4806 442C
HCDONEHEAD	R	32	0x0000 0030	0x4806 4430
HCFMINTERVAL	RW	32	0x0000 0034	0x4806 4434
HCFMREMAINING	R	32	0x0000 0038	0x4806 4438
HCFMNUMBER	R	32	0x0000 003C	0x4806 443C
HCPERIODICSTART	RW	32	0x0000 0040	0x4806 4440
HCLSTHRESHOLD	RW	32	0x0000 0044	0x4806 4444
HCRHDESCRIPTORA	RW	32	0x0000 0048	0x4806 4448
HCRHDESCRIPTORB	RW	32	0x0000 004C	0x4806 444C
HCRHSTATUS	RW	32	0x0000 0050	0x4806 4450
HCRHPORTSTATUS_1	RW	32	0x0000 0054	0x4806 4454
HCRHPORTSTATUS_2	RW	32	0x0000 0058	0x4806 4458
HCRHPORTSTATUS_3	RW	32	0x0000 005C	0x4806 445C

OHCI register descriptions conform to the OHCI USB standard: *Open Host controller Interface Specification for USB*, Release 1.0a.

For more information about these registers, or for new specification releases, search OHCI on www.usb.org.

Table 24-55. EHCI Registers Mapping Summary

Register Name	Type	Register Width (Bits)	Address Offset	Physical Address
HCCAPBASE	R	32	0x0000 0000	0x4806 4800
HCSPARAMS	R	32	0x0000 0004	0x4806 4804
HCCPARAMS	R	32	0x0000 0008	0x4806 4808
USBCMD	RW	32	0x0000 0010	0x4806 4810
USBSTS	RW	32	0x0000 0014	0x4806 4814
USBINTR	RW	32	0x0000 0018	0x4806 4818
FRINDEX	RW	32	0x0000 001C	0x4806 481C
CTRLDSSEGMENT	R	32	0x0000 0020	0x4806 4820
PERIODICLISTBASE	RW	32	0x0000 0024	0x4806 4824
ASYNCLISTADDR	RW	32	0x0000 0028	0x4806 4828
CONFIGFLAG	RW	32	0x0000 0050	0x4806 4850
PORTSC_i (1)	RW	32	0x0000 0054 + (0x04 * I)	0x4806 4854 + (0x04 * I)
INSNREG00	RW	32	0x0000 0090	0x4806 4890
INSNREG01	RW	32	0x0000 0094	0x4806 4894
INSNREG02	RW	32	0x0000 0098	0x4806 4898
INSNREG03	RW	32	0x0000 009C	0x4806 489C

(1) i = 0 to 2

Table 24-55. EHCI Registers Mapping Summary (continued)

Register Name	Type	Register Width (Bits)	Address Offset	Physical Address
INSNREG04	RW	32	0x0000 00A0	0x4806 48A0
INSNREG05_UTMI	RW	32	0x0000 00A4	0x4806 48A4
INSNREG05_ULPI	RW	32	0x0000 00A4	0x4806 48A4

EHCI register descriptions conform to the EHCI USB standard: *Enhanced Host Controller Interface (EHCI) Specification for USB*, Release 1.0.

For more information about these registers or for new specification releases, search EHCI on www.usb.org.

24.2.6.4 Register Description

24.2.6.4.1 USBTLL Registers

Table 24-56. USBTLL_REVISION

Address Offset	0x0000 0000		
Physical Address	0x4806 2000	**Instance**	USBTLL
Description	Standard revision number, BCD encoded		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																								MAJOR				MINOR			

Bits	Field Name	Description	Type	Reset
31:8	RESERVED	Reserved	R	0x000000
7:4	MAJOR	Major revision number	R	0x0
3:0	MINOR	Minor revision number	R	0x1

Table 24-57. Register Call Summary for Register USBTLL_REVISION

High-Speed USB Host Subsystem

- High-Speed USB Host Subsystem Register Mapping Summary: [0]

Table 24-58. USBTLL_SYSCONFIG

Address Offset	0x0000 0010		
Physical Address	0x4806 2010	**Instance**	USBTLL
Description	Standard system configuration register		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																								CLOCKACTIVITY	RESERVED			SIDLEMODE		ENAWAKEUP	SOFTRESET	AUTOIDLE

Bits	Field Name	Description	Type	Reset
31:9	RESERVED	Reserved	R	0x000000
8	CLOCKACTIVITY	Enable autogating of L3 interconnect-derived internal clocks while module is idle. 0x0: L3 interconnect-derived internal clocks OFF during idle 0x1: L3 interconnect-derived internal clocks ON during idle	RW	0x0
7:5	RESERVED	Reserved	R	0x0
4:3	SIDLEMODE	Slave interface power management control. Idle Req/ack control 0x0: Force-Idle mode. Sidleack asserted after Idlereq assertion 0x1: No-idle mode. Sidleack never asserted. 0x2: Smart-idle mode. Sidleack asserted after Idlereq assertion when no more activity on the USB.	RW	0x0
2	ENAWAKEUP	Asynchronous wakeup generation control (Swakeup) 0x0: Wakeup generation disabled 0x1: Wakeup generation enabled	RW	0x0
1	SOFTRESET	Module software reset 0x0: No effect 0x1: Starts softreset sequence.	W	0x0
0	AUTOIDLE	Internal autogating control 0x0: Clock always running 0x1: When no activity on L3 interconnect, clock is cut off.	RW	0x1

Table 24-59. Register Call Summary for Register USBTLL_SYSCONFIG

High-Speed USB Host Subsystem

- Reset, Clocking, and Power-Management Scheme: [0] [1] [2] [3]
- USBTLL Module Functionality: [4]
- High-Speed USB Host Subsystem Register Mapping Summary: [5]

Table 24-60. USBTLL_SYSSTATUS

Address Offset	0x0000 0014		
Physical Address	0x4806 2014	**Instance**	USBTLL
Description	Standard system status register		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																															RESETDONE

Bits	Field Name	Description	Type	Reset
31:1	RESERVED	Reserved	R	0x00000000
0	RESETDONE	Indicates when the module has entirely come out of reset 0x0: Reset is ongoing 0x1: Reset is done	R	0x0

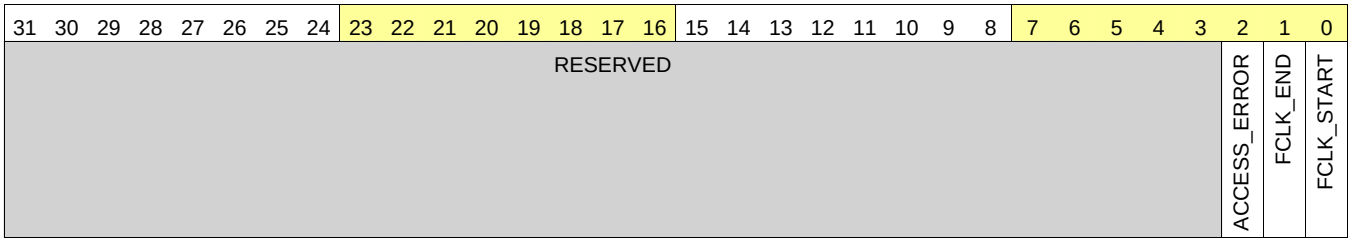
Table 24-61. Register Call Summary for Register USBTLL_SYSSTATUS

High-Speed USB Host Subsystem

- High-Speed USB Host Subsystem Register Mapping Summary: [0]

Table 24-62. USBTLL_IRQSTATUS

Address Offset	0x0000 0018		
Physical Address	0x4806 2018	**Instance**	USBTLL
Description	Standard IRQ status vector. Write 1 to clear a bit.		
Type	RW		



Bits	Field Name	Description	Type	Reset
31:3	RESERVED	Reserved	R	0x00000000
2	ACCESS_ERROR	Access error to ULPI register over L3 interconnect: USB clock must run for that type of access to succeed. 0x0: No event pending 0x1: Event pending	RW	0x0
1	FCLK_END	Functional clock is no longer requested for USB clocking When TLL_SHARED_CONF[1] FCLK_REQ=0 and TLL_SHARED_CONF[0] FCLK_IS_ON=1, IRQ is generated to request the clock to be switched OFF and FCLK_END is set to 1. 0x0: No event pending 0x1: Event pending	RW	0x0
0	FCLK_START	Functional clock is requested for USB clocking When TLL_SHARED_CONF[1] FCLK_REQ=1 and TLL_SHARED_CONF[0] FCLK_IS_ON=0, IRQ is generated to request the clock to be switched ON and FCLK_START is set to 1. 0x0: No event pending 0x1: Event pending	RW	0x0

Table 24-63. Register Call Summary for Register USBTLL_IRQSTATUS

High-Speed USB Host Subsystem

- High-Speed USB Host Subsystem Register Mapping Summary: [0]
- Register Description: [1] [2]

Table 24-64. USBTLL_IRQENABLE

Address Offset	0x0000 001C		
Physical Address	0x4806 201C	**Instance**	USBTLL
Description	Standard IRQ enable vector		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																													ACCESS_ERROR_EN	FCLK_END_EN	FCLK_START_EN

Bits	Field Name	Description	Type	Reset
31:3	RESERVED	Reserved	R	0x00000000
2	ACCESS_ERROR_EN	Enable IRQ generation upon access error to ULPI register over L3 interconnect 0x0: IRQ event is masked 0x1: IRQ event is enabled	RW	0x0
1	FCLK_END_EN	IRQ event mask for FCLK_END interrupt (see USBTLL_IRQSTATUS[1]) 0x0: IRQ event is masked 0x1: IRQ event is enabled	RW	0x0
0	FCLK_START_EN	IRQ event mask for FCLK_START interrupt (see USBTLL_IRQSTATUS[0]) 0x0: IRQ event is masked 0x1: IRQ event is enabled	RW	0x0

Table 24-65. Register Call Summary for Register USBTLL_IRQENABLE

High-Speed USB Host Subsystem

- USBTLL Module Functionality: [0]
- High-Speed USB Host Subsystem Register Mapping Summary: [1]

Table 24-66. TLL_SHARED_CONF

Address Offset	0x0000 0030		
Physical Address	0x4806 2030	**Instance**	USBTLL
Description	Common control register for all TLL channels		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																									USB_90D_DDR_EN	USB_180D_SDR_EN	USB_DIVRATIO			FCLK_REQ	FCLK_IS_ON

Bits	Field Name	Description	Type	Reset
31:7	RESERVED	Reserved	R	0x0000000
6	USB_90D_DDR_EN	Software enable/disable of the 90-degree phase shift scheme on output DDR data, when implemented. Read-only, always-0 when HDL generic ULPI_90DEG4DDR = 0. 0x0: ULPI DDR output DATA aligned with CLK 0x1: ULPI DDR output DATA delayed by 90 degree wrt CLK	RW	0x1

Bits	Field Name	Description	Type	Reset
5	USB_180D_SDR_EN	Software enable/disable of the 180-degree phase shift scheme on output SDR data, when implemented. Read-only, always-0 when HDL generic ULPI_180DEG4SDR = 0 0x0: ULPI SDR output DATA aligned with CLK 0x1: ULPI SDR output DATA delayed by 180 degree wrt CLK	RW	0x1
4:2	USB_DIVRATIO	(Log2 of) division ratio from functional clock to USB (UTMI/ULPI) clock 0x0: Div ratio is 2**0 = 1 : Bypass 0x1: Div ratio is 2**1 = 2 0x2: Div ratio is 2**2 = 4 0x3: Div ratio is 2**3 = 8 0x4: Div ratio is 2**4 = 16 0x5: Div ratio is 2**5 = 32 0x6: Div ratio is 2**6 = 64 0x7: Div ratio is 2**7 = 128	RW	0x0
1	FCLK_REQ	Functional clock request, ORed from all channels depending on their respective USB bus state. Combined with the Fclk_is_on status to generate fclk_start/end IRQs. 0x0: Func clock input is not requested by TLL 0x1: Func clock input is requested by TLL	R	0x0
0	FCLK_IS_ON	Status of the functional clock input, provided by the system to the TLL module. The TLL module will only use that clock if the current status indicated that it is ready. Combined with the Fclk_request to generate fclk_start/end IRQs. 0x0: Functional clock input is not guaranteed ON (can actually be ON, OFF, or unstable) 0x1: Functional clock input is guaranteed ON and stable	RW	0x0

Table 24-67. Register Call Summary for Register TLL_SHARED_CONF

High-Speed USB Host Subsystem

- USBTLL Module Functionality: [0]
- High-Speed USB Host Subsystem Register Mapping Summary: [1]
- Register Description: [2] [3] [4] [5]

Table 24-68. TLL_CHANNEL_CONF_i

Address Offset	0x0000 0040 + (0x04 * i)	**Index**	i = 0 to 2
Physical Address	0x4806 2040 + (0x04 * i)	**Instance**	USBTLL
Description	Control and Status register for channel I.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED		FSLSLINESTATE		FSLSMODE				RESERVED			TESTTXSE0	TESTTXDAT	TESTTXEN	TESTEN	DRVVBUS	CHRGVBUS	RESERVED			ULPINOBITSTUFF	ULPIAUTOIDLE	UTMIAUTOIDLE	ULPIDDRMODE	ULPIOUTCLKMODE	TLLFULLSPEED	TLLCONNECT	TLLATTACH	UTMIISADEV	CHANMODE		CHANEN

Bits	Field Name	Description	Type	Reset
31:30	RESERVED	Reserved	R	0x0
29:28	FSLSLINESTATE	Line state for Full/Low speed serial modes Bit1 = D-/ Bit0 = D+ 0x0: Single-ended 0 0x1: Full-Speed J = differential 1 0x2: Full-Speed K = differential 0 0x3: Single-ended 1 (illegal in USB)	R	0x0
27:24	FSLSMODE	Multiple-mode serial interface's mode select. Only when main channel mode is serial. No effect in other main modes. 0x0: "6pin" unidirectional PHY i/f mode. TX encoding is Dat/Se0 (default) 0x1: "6-pin" unidirectional PHY i/f mode. TX encoding is Dp/Dm 0x2: "3-pin" bidirectional PHY i/f mode. 0x3: "4-pin" bidirectional PHY i/f mode. 0x4: "6pin" unidirectional TLL mode. TX encoding is Dat/Se0 0x5: "6pin" unidirectional TLL mode. TX encoding is Dp/Dm 0x6: "3-pin" bidirectional TLL mode. 0x7: "4-pin" bidirectional TLL mode. 0xA: "2-pin" bidirectional TLL mode. Encoding is Dat/Se0 0xB: "2-pin" bidirectional TLL mode. Encoding is Dp/Dm	RW	0x0
23:21	RESERVED	Reserved	R	0x0
20	TESTTXSE0	Force-Se0 transmit override value for serial mode test Don't care if TestEn = 0 (functional mode) or = TestTxen = 1 (tx = hiz) 0x0: Drive differential value on TX according to TestTXDat 0x1: Drive SE0 on TX	RW	0x0
19	TESTTXDAT	Differential data transmit override value for serial mode test Don't care if TestEn = 0 (functional mode) or = TestTxen = 1 (tx = hiz) or TestSe0 = 1 (tx = se0) 0x0: Drive full-speed K = differential 0 0x1: Drive full-speed J = differential 1	RW	0x0
18	TESTTXEN	Differential data transmit override value for serial mode test Don't care if TestEn = 0 (functional mode) 0x0: Drive TX according to TestTXDat/Se0 0x1: Drive TX Hiz (no drive: Pullups determine line state)	RW	0x0
17	TESTEN	Enable manual test override for serial mode TX path (from local controller's UTMI port) 0x0: No override. TX is from local link controller 0x1: Override enabled	RW	0x0
16	DRVVBUS	VBUS-drive for ChanMode = serial * In TLL config, write 1 to emulate serial-side VBUS drive * In PHY config, write 1 to report "VBUS valid" status (of actual VBUS) to UTMI controller 0x0: VBUS not driven 0x1: VBUS driven to 5V	RW	0x0
15	CHRGVBUS	VBUS-drive for ChanMode = serial * In TLL config, write 1 to emulate serial-side VBUS charge/pullup (OTG) * In PHY config, write 1 to reports "session valid" status (of actual VBUS) to UTMI controller 0x0: VBUS not charged, session not valid 0x1: VBUS charged, session valid	RW	0x0
14:12	RESERVED	Reserved	R	0x0

Bits	Field Name	Description	Type	Reset
11	ULPINOBITSTUFF	Disable bitstuff emulation in ULPI TLL for ULPI ChanMode	RW	0x0
		0x0: Bitstuff enabled, following USB standard		
		0x1: No bitstuff or associated delays (non-standard)		
10	ULPIAUTOIDLE	For ChanMode = ULPI TLL only. Allow the ULPI output clock to be stopped when ULPI goes into asynchronous mode (low-power, 3-pin serial, 6-pin serial). No effect in ULPI input clock mode.	RW	0x1
		0x0: ULPI output clock always-on		
		0x1: ULPI output clock stops during asynchronous ULPI modes		
9	UTMIAUTOIDLE	For ChanMode = ULPI TLL only. Allow the UTMI clock (output) to be stopped when UTMII goes to suspended mode (suspendm = 0)	RW	0x1
		0x0: UTMI clock output always on		
		0x1: UTMI clock output gated upon suspend		
8	ULPIDDRMODE	Select single/double data rate (SDR/DDR) mode for ULPI TLL Reset value depends on hardware generics ULPI_SDR/DDR_MODE.	RW	0x0
		0x0: SDR mode (8 data bit/12 pin)		
		0x1: DDR mode (4 data bit/8 pin)		
7	ULPIOUTCLKMODE	ULPI clocking mode select for ULPI TLL ChanMode	RW	0x1
		0x0: ULPI clock provided by LINK (that is, off-chip). ULPI clock is input		
		0x1: ULPI clock provided by PHY side (that is, TLL, from functional clock). ULPI clock is output		
6	TLLFULLSPEED	Sets PHY speed emulation in TLL (full/slow), which determines the line to pull up upon connect. The two connect source controls are: Input m(N)_tllpuen, register field TllConnect.	RW	0x1
		0x0: Connect is Low-speed: D- pullup		
		0x1: Connect is Full-Speed: D+ pullup		
5	TLLCONNECT	Emulation of Full/Low-Speed connect (that is, D+ resp D- pullup) for serial TLL modes. Speed is determined by field TllSpeed.	RW	0x0
		0x0: Unconnected		
		0x1: Connected		
4	TLLATTACH	Emulates cable attach/detach for all serial TLL modes: * ChanMode = serial, in TLL mode (FsLsMode) * ChanMode = ULPI, in serial mode (6pin/3pin TLL)	RW	0x1
		0x0: Cable detach emulated on serial TLL		
		0x1: Cable attach emulated on serial TLL		
3	UTMIISADEV	Select the cable end "seen" by UTMI side of TLL, that is, the emulated USB cable's orientation. Note that host must always be on A side, Peripheral on B side. Reset value depends on generic DEFUTMIISHOST.	RW	0x1
		0x0: UTMI side is peripheral, ULPI side is host		
		0x1: UTMI side is host, ULPI side is peripheral		
2:1	CHANMODE	Main channel mode selection	RW	0x0
		0x0: UTMI-to-ULPI TLL mode (HS capable): To ULPI controller		
		0x1: UTMI-to-serial (FS/LS) mode: To serial controller (TLL) or serial PHY		
		0x2: Transparent UTMI mode: To UTMI PHY		
		0x3: No mode selected		
0	CHANEN	Active-high channel enable. A disabled channel is unclocked and kept under reset.	RW	0x0
		0x0: Channel #N disabled		
		0x1: Channel #N enabled		

Table 24-69. Register Call Summary for Register TLL_CHANNEL_CONF_i

High-Speed USB Host Subsystem

- USBTLL Module Functionality: [0] [1] [2] [3] [4] [5] [6] [7] [8] [9] [10] [11] [12] [13] [14] [15]
- Selecting and Configuring USB Connectivity: [16] [17] [18] [19] [20]
- High-Speed USB Host Subsystem Register Mapping Summary: [21]

Table 24-70. ULPI_VENDOR_ID_LO_i

Address Offset	0x0000 0000 + (0x100 * i)	**Index**	i = 0 to 2
Physical Address	0x4806 2800 + (0x100 * i)	**Instance**	ULPI
Description	Lower byte of USB-IF-supplied vendor ID Value is set for all channels by HDL generic ULPI_VENDORID Default is Texas-Instruments Vendor ID = 0x0451		
Type	R		

7	6	5	4	3	2	1	0
VENDOR_ID_LO							

Bits	Field Name	Description	Type	Reset
7:0	VENDOR_ID_LO		R	0x51

Table 24-71. Register Call Summary for Register ULPI_VENDOR_ID_LO_i

High-Speed USB Host Subsystem

- High-Speed USB Host Subsystem Register Mapping Summary: [0]

Table 24-72. ULPI_VENDOR_ID_HI_i

Address Offset	0x0000 0001 + (0x100 * i)	**Index**	i = 0 to 2
Physical Address	0x4806 2801 + (0x100 * i)	**Instance**	ULPI
Description	Upper byte of USB-IF-supplied 16-bit vendor ID Value is set for all channels by HDL generic ULPI_VENDORID Default is Texas-Instruments Vendor ID = 0x0451		
Type	R		

7	6	5	4	3	2	1	0
VENDOR_ID_HI							

Bits	Field Name	Description	Type	Reset
7:0	VENDOR_ID_HI		R	0x04

Table 24-73. Register Call Summary for Register ULPI_VENDOR_ID_HI_i

High-Speed USB Host Subsystem

- High-Speed USB Host Subsystem Register Mapping Summary: [0]

Table 24-74. ULPI_PRODUCT_ID_LO_i

Address Offset	0x0000 0002 + (0x100 * i)	**Index**	i = 0 to 2
Physical Address	0x4806 2802 + (0x100 * i)	**Instance**	ULPI
Description	Lower byte of vendor-chosen 16-bit product ID Value is set for all channels by HDL generic ULPI_PRODUCTID		
Type	R		

7	6	5	4	3	2	1	0
PRODUCT_ID_LO							

Bits	Field Name	Description	Type	Reset
7:0	PRODUCT_ID_LO		R	0x00

Table 24-75. Register Call Summary for Register ULPI_PRODUCT_ID_LO_i

High-Speed USB Host Subsystem

- High-Speed USB Host Subsystem Register Mapping Summary: [0]

Table 24-76. ULPI_PRODUCT_ID_HI_i

Address Offset	0x0000 0003 + (0x100 * i)	**Index**	i = 0 to 2
Physical Address	0x4806 2803 + (0x100 * i)	**Instance**	ULPI
Description	Upper byte of vendor-chosen 16-bit product ID Value is set for all channels by HDL generic ULPI_PRODUCTID		
Type	R		

7	6	5	4	3	2	1	0
PRODUCT_ID_HI							

Bits	Field Name	Description	Type	Reset
7:0	PRODUCT_ID_HI		R	0x00

Table 24-77. Register Call Summary for Register ULPI_PRODUCT_ID_HI_i

High-Speed USB Host Subsystem

- High-Speed USB Host Subsystem Register Mapping Summary: [0]

Table 24-78. ULPI_FUNCTION_CTRL_i

Address Offset	0x0000 0004 + (0x100 * i)	**Index**	i = 0 to 2
Physical Address	0x4806 2804 + (0x100 * i)	**Instance**	ULPI
Description	Controls UTMI function settings of the PHY. Read/Write address.		
Type	RW		

7	6	5	4	3	2	1	0
RESERVED	SUSPENDM	RESET	OPMODE		TERMSELECT	XCVRSELECT	

Bits	Field Name	Description	Type	Reset
7	RESERVED	Reserved	R	0x0
6	SUSPENDM	Active low PHY suspend: Puts the ULPI bus in Low Power Mode. Automatically set back to 1 upon Low Power Mode exit. 0x0: PHY is in low-power mode 0x1: PHY is not in low-power mode	RW	0x1
5	RESET	Active high UTMI transceiver reset. Autocleared. Does not reset the ULPI interface or ULPI register set. 0x0: No ongoing reset/ no action 0x1: Ongoing reset/apply reset	RW	0x0
4:3	OPMODE	Select the required bit encoding style during transmit 0x0: Normal operation 0x1: Non-driving 0x2: Disable bit-stuff and NRZI encoding 0x3: Reserved	RW	0x0

Bits	Field Name	Description	Type	Reset
2	TERMSELECT	Controls the internal 1.5Kohms pull-up resistor and 45ohms HS terminations. Control over bus resistors changes depending on XcvrSelect, OpMode, DpPulldown and DmPulldown. 0x0: HS termination enabled (other conditions) 0x1: FS termination enabled (other conditions)	RW	0x0
1:0	XCVRSELECT	Select the required transceiver speed. 0x0: Enable HS transceiver 0x1: Enable FS transceiver 0x2: Enable LS transceiver 0x3: Enable FS transceiver for LS packets (automatic FS preamble pre-pending)	RW	0x1

Table 24-79. Register Call Summary for Register ULPI_FUNCTION_CTRL_i

High-Speed USB Host Subsystem

- USBTLL Module Functionality: [0]
- High-Speed USB Host Subsystem Register Mapping Summary: [1]

Table 24-80. ULPI_FUNCTION_CTRL_SET_i

Address Offset	0x0000 0005 + (0x100 * i)	**Index**	i = 0 to 2
Physical Address	0x4806 2805 + (0x100 * i)	**Instance**	ULPI
Description	Controls UTMI function settings of the PHY. Read/set address (write 1 to a bit to set it to 1; writing 0 has no effect on bit value). See fields description at the read/write address of the same register.		
Type	RW		

7	6	5	4	3	2	1	0
RESERVED	SUSPENDM	RESET	OPMODE		TERMSELECT	XCVRSELECT	

Bits	Field Name	Description	Type	Reset
7	RESERVED	Reserved	R	0x0
6	SUSPENDM	Active low PHY suspend: Puts the ULPI bus in Low Power Mode. Automatically set back to 1 upon Low Power Mode exit. Write 0x0: No effect on bit value Write 0x1: Set the bit to 1.	RW	0x0
5	RESET	Active high UTMI transceiver reset. Autocleared. Does not reset the ULPI interface or ULPI register set. Write 0x0: No effect on bit value Write 0x1: Set the bit to 1.	RW	0x0
4:3	OPMODE	Select the required bit encoding style during transmit Write 0x0: No effect on bit value Write 0x1: Set the bit to 1.	RW	0x0
2	TERMSELECT	Controls the internal 1.5Kohms pull-up resistor and 45ohms HS terminations. Control over bus resistors changes depending on XcvrSelect, OpMode, DpPulldown and DmPulldown. Write 0x0: No effect on bit value Write 0x1: Set the bit to 1.	RW	0x0
1:0	XCVRSELECT	Select the required transceiver speed. Write 0x0: No effect on bit value Write 0x1: Set the bit to 1.	RW	0x0

Table 24-81. Register Call Summary for Register ULPI_FUNCTION_CTRL_SET_i

High-Speed USB Host Subsystem

- High-Speed USB Host Subsystem Register Mapping Summary: [0]

Table 24-82. ULPI_FUNCTION_CTRL_CLR_i

Address Offset	0x0000 0006 + (0x100 * i)	**Index**	i = 0 to 2
Physical Address	0x4806 2806 + (0x100 * i)	**Instance**	ULPI
Description	Controls UTMI function settings of the PHY. Read/clear address (write 1 to a bit to clear it to 0, writing 0 has no effect on bit value). See fields description at the read/write address of the same register.		
Type	RW		

7	6	5	4	3	2	1	0
RESERVED	SUSPENDM	RESET	OPMODE		TERMSELECT	XCVRSELECT	

Bits	Field Name	Description	Type	Reset
7	RESERVED	Reserved	R	0x0
6	SUSPENDM	Active low PHY suspend: Puts the ULPI bus in Low Power Mode. Automatically set back to 1 upon Low Power Mode exit. Write 0x0: No effect on bit value Write 0x1: Clear the bit to 0	RW	0x0
5	RESET	Active high UTMI transceiver reset. Autocleared. Does not reset the ULPI interface or ULPI register set. Write 0x0: No effect on bit value Write 0x1: Clear the bit to 0	RW	0x0
4:3	OPMODE	Select the required bit encoding style during transmit Write 0x0: No effect on bit value Write 0x1: Clear the bit to 0	RW	0x0
2	TERMSELECT	Controls the internal 1.5Kohms pull-up resistor and 45ohms HS terminations. Control over bus resistors changes depending on XcvrSelect, OpMode, DpPulldown and DmPulldown. Write 0x0: No effect on bit value Write 0x1: Clear the bit to 0	RW	0x0
1:0	XCVRSELECT	Select the required transceiver speed. Write 0x0: No effect on bit value Write 0x1: Clear the bit to 0	RW	0x0

Table 24-83. Register Call Summary for Register ULPI_FUNCTION_CTRL_CLR_i

High-Speed USB Host Subsystem

- High-Speed USB Host Subsystem Register Mapping Summary: [0]

Table 24-84. ULPI_INTERFACE_CTRL_i

Address Offset	0x0000 0007 + (0x100 * i)	**Index**	i = 0 to 2
Physical Address	0x4806 2807 + (0x100 * i)	**Instance**	ULPI
Description	Enables alternative interfaces and PHY features. Read/Write address.		
Type	RW		

7	6	5	4	3	2	1	0
INTERFACE_PROTECT_DISABLE	RESERVED		AUTORESUME	CLOCKSUSPENDM	RESERVED	FSLSSERIALMODE_3PIN	FSLSSERIALMODE_6PIN

Bits	Field Name	Description	Type	Reset
7	INTERFACE_PROTECT_DISABLE	Controls circuitry built into the PHY for protecting the ULPI interface when the link 3-states stp and data. 0x0: Enables the interface protect circuit 0x1: Disables the interface protect circuit	RW	0x0
6:5	RESERVED	Reserved	R	0x0
4	AUTORESUME	Enables the PHY to automatically drive resume signaling. On by default. 0x0: AutoResume disabled 0x1: AutoResume enabled	RW	0x1
3	CLOCKSUSPENDM	Active low clock suspend for serial modes (6pin/3-pin). 0x0: ULPI clock will stop during serial modes. 0x1: ULPI clock will run during serial modes.	RW	0x0
2	RESERVED	Reserved	R	0x0
1	FSLSSERIALMODE_3PIN	Sets the ULPI interface to 3-pin (FS/LS only) Serial Mode. Autocleared when serial mode is exited. 0x0: ULPI is not in 3-pin mode 0x1: ULPI in 3-pin serial mode	RW	0x0
0	FSLSSERIALMODE_6PIN	Sets the ULPI interface to 6-pin (FS/LS only) Serial Mode. Autocleared when serial mode is exited. 0x0: ULPI is not in 6-pin mode 0x1: ULPI in 6-pin serial mode	RW	0x0

Table 24-85. Register Call Summary for Register ULPI_INTERFACE_CTRL_i

High-Speed USB Host Subsystem

- USBTLL Module Functionality: [0]
- High-Speed USB Host Subsystem Register Mapping Summary: [1]

Table 24-86. ULPI_INTERFACE_CTRL_SET_i

Address Offset	0x0000 0008 + (0x100 * i)	**Index**	i = 0 to 2
Physical Address	0x4806 2808 + (0x100 * i)	**Instance**	ULPI
Description	Enables alternative interfaces and PHY features. Read/set address (write 1 to a bit to set it to 1; writing 0 has no effect on bit value). See fields description at the read/write address of the same register.		
Type	RW		

7	6	5	4	3	2	1	0
INTERFACE_PROTECT_DISABLE	RESERVED		AUTORESUME	CLOCKSUSPENDM	RESERVED	FSLSSERIALMODE_3PIN	FSLSSERIALMODE_6PIN

Bits	Field Name	Description	Type	Reset
7	INTERFACE_PROTECT_DISABLE	Controls circuitry built into the PHY for protecting the ULPI interface when the link 3-states stp and data. Write 0x0: No effect on bit value Write 0x1: Set the bit to 1.	RW	0x0
6:5	RESERVED	Reserved	R	0x0
4	AUTORESUME	Enables the PHY to automatically drive resume signaling. On by default. Write 0x0: No effect on bit value Write 0x1: Set the bit to 1.	RW	0x0
3	CLOCKSUSPENDM	Active low clock suspend for serial modes (6pin/3-pin). Write 0x0: No effect on bit value Write 0x1: Set the bit to 1.	RW	0x0
2	RESERVED	Reserved	R	0x0
1	FSLSSERIALMODE_3PIN	Sets the ULPI interface to 3-pin (FS/LS only) Serial Mode. Autocleared when serial mode is exited. Write 0x0: No effect on bit value Write 0x1: Set the bit to 1.	RW	0x0
0	FSLSSERIALMODE_6PIN	Sets the ULPI interface to 6-pin (FS/LS only) Serial Mode. Autocleared when serial mode is exited. Write 0x0: No effect on bit value Write 0x1: Set the bit to 1.	RW	0x0

Table 24-87. Register Call Summary for Register ULPI_INTERFACE_CTRL_SET_i

High-Speed USB Host Subsystem

- High-Speed USB Host Subsystem Register Mapping Summary: [0]

Table 24-88. ULPI_INTERFACE_CTRL_CLR_i

Address Offset	0x0000 0009 + (0x100 * i)	**Index**	i = 0 to 2
Physical Address	0x4806 2809 + (0x100 * i)	**Instance**	ULPI
Description	Enables alternative interfaces and PHY features. Read/clear address (write 1 to a bit to clear it to 0, writing 0 has no effect on bit value). See fields description at the read/write address of the same register.		
Type	RW		

7	6	5	4	3	2	1	0
INTERFACE_PROTECT_DISABLE	RESERVED		AUTORESUME	CLOCKSUSPENDM	RESERVED	FSLSSERIALMODE_3PIN	FSLSSERIALMODE_6PIN

Bits	Field Name	Description	Type	Reset
7	INTERFACE_PROTECT_DISABLE	Controls circuitry built into the PHY for protecting the ULPI interface when the link 3-states stp and data. Write 0x0: No effect on bit value Write 0x1: Clear the bit to 0	RW	0x0
6:5	RESERVED	Reserved	R	0x0
4	AUTORESUME	Enables the PHY to automatically drive resume signaling. On by default. Write 0x0: No effect on bit value Write 0x1: Clear the bit to 0	RW	0x0
3	CLOCKSUSPENDM	Active low clock suspend for serial modes (6pin/3-pin). Write 0x0: No effect on bit value Write 0x1: Clear the bit to 0	RW	0x0
2	RESERVED	Reserved	R	0x0
1	FSLSSERIALMODE_3PIN	Sets the ULPI interface to 3-pin (FS/LS only) Serial Mode. Autocleared when serial mode is exited. Write 0x0: No effect on bit value Write 0x1: Clear the bit to 0	RW	0x0
0	FSLSSERIALMODE_6PIN	Sets the ULPI interface to 6-pin (FS/LS only) Serial Mode. Autocleared when serial mode is exited. Write 0x0: No effect on bit value Write 0x1: Clear the bit to 0	RW	0x0

Table 24-89. Register Call Summary for Register ULPI_INTERFACE_CTRL_CLR_i

High-Speed USB Host Subsystem

- High-Speed USB Host Subsystem Register Mapping Summary: [0]

Table 24-90. ULPI_OTG_CTRL_i

Address Offset	0x0000 000A + (0x100 * i)	**Index**	i = 0 to 2
Physical Address	0x4806 280A + (0x100 * i)	**Instance**	ULPI
Description	Controls UTMI+ OTG functions of the PHY. Read/Write address.		
Type	RW		

7	6	5	4	3	2	1	0
RESERVED		DRVVBUS	CHRGVBUS	DISCHRGVBUS	DMPULLDOWN	DPPULLDOWN	IDPULLUP

Bits	Field Name	Description	Type	Reset
7:6	RESERVED	Reserved	R	0x0
5	DRVVBUS	Drive 5 V on VBUS 0x0: No action 0x1: Drive VBUS	RW	0x0
4	CHRGVBUS	Charge VBUS through a resistor for VBUS-pulsing SRP. 0x0: No action 0x1: Charge VBUS	RW	0x0
3	DISCHRGVBUS	Discharge VBUS through a resistor, until the session-end VBUS state is reached. 0x0: No action 0x1: Discharge VBUS	RW	0x0
2	DMPULLDOWN	Enables the 15kΩ pull-down resistor on D- 0x0: Pull-down resistor not connected to D- 0x1: Pull-down resistor connected to D-	RW	0x1
1	DPPULLDOWN	Enables the 15kΩ pull-down resistor on D+ 0x0: Pull-down resistor not connected to D+ 0x1: Pull-down resistor connected to D+	RW	0x1
0	IDPULLUP	Pull-up to the (OTG) ID line to allow its sampling 0x0: Disable sampling of ID line. 0x1: Enable sampling of ID line.	RW	0x0

Table 24-91. Register Call Summary for Register ULPI_OTG_CTRL_i

High-Speed USB Host Subsystem

- USBTLL Module Functionality: [0]
- High-Speed USB Host Subsystem Register Mapping Summary: [1]

Table 24-92. ULPI_OTG_CTRL_SET_i

Address Offset	0x0000 000B + (0x100 * i)	**Index**	i = 0 to 2
Physical Address	0x4806 280B + (0x100 * i)	**Instance**	ULPI
Description	Controls UTMI+ OTG functions of the PHY. Read/set address (write 1 to a bit to set it to 1; writing 0 has no effect on bit value). See fields description at the read/write address of the same register.		
Type	RW		

7	6	5	4	3	2	1	0
RESERVED		DRVVBUS	CHRGVBUS	DISCHRGVBUS	DMPULLDOWN	DPPULLDOWN	IDPULLUP

Bits	Field Name	Description	Type	Reset
7:6	RESERVED	Reserved	R	0x0
5	DRVVBUS	Drive 5 V on VBUS Write 0x0: No effect on bit value Write 0x1: Set the bit to 1.	RW	0x0
4	CHRGVBUS	Charge VBUS through a resistor for VBUS-pulsing SRP. Write 0x0: No effect on bit value 0x1: Set the bit to 1.	RW	0x0

Bits	Field Name	Description	Type	Reset
3	DISCHRGVBUS	Discharge VBUS through a resistor, until the session-end VBUS state is reached. Write 0x0: No effect on bit value Write 0x1: Set the bit to 1.	RW	0x0
2	DMPULLDOWN	Enables the 15kΩ pull-down resistor on D- Write 0x0: No effect on bit value Write 0x1: Set the bit to 1.	RW	0x0
1	DPPULLDOWN	Enables the 15kΩ pull-down resistor on D+ Write 0x0: No effect on bit value Write 0x1: Set the bit to 1.	RW	0x0
0	IDPULLUP	Pull-up to the (OTG) ID line to allow its sampling Write 0x0: No effect on bit value Write 0x1: Set the bit to 1.	RW	0x0

Table 24-93. Register Call Summary for Register ULPI_OTG_CTRL_SET_i

High-Speed USB Host Subsystem

- High-Speed USB Host Subsystem Register Mapping Summary: [0]

Table 24-94. ULPI_OTG_CTRL_CLR_i

Address Offset	0x0000 000C + (0x100 * i)	**Index**	i = 0 to 2
Physical Address	0x4806 280C + (0x100 * i)	**Instance**	ULPI
Description	Controls UTMI+ OTG functions of the PHY. Read/clear address (write 1 to a bit to clear it to 0, writing 0 has no effect on bit value). See fields description at the read/write address of the same register.		
Type	RW		

7	6	5	4	3	2	1	0
RESERVED		DRVVBUS	CHRGVBUS	DISCHRGVBUS	DMPULLDOWN	DPPULLDOWN	IDPULLUP

Bits	Field Name	Description	Type	Reset
7:6	RESERVED	Reserved	R	0x0
5	DRVVBUS	Drive 5 V on VBUS Write 0x0: No effect on bit value Write 0x1: Clear the bit to 0	RW	0x0
4	CHRGVBUS	Charge VBUS through a resistor for VBUS-pulsing SRP. Write 0x0: No effect on bit value Write 0x1: Clear the bit to 0	RW	0x0
3	DISCHRGVBUS	Discharge VBUS through a resistor, until the session-end VBUS state is reached. Write 0x0: No effect on bit value Write 0x1: Clear the bit to 0	RW	0x0
2	DMPULLDOWN	Enables the 15kΩ pull-down resistor on D- Write 0x0: No effect on bit value Write 0x1: Clear the bit to 0	RW	0x0
1	DPPULLDOWN	Enables the 15kΩ pull-down resistor on D+ Write 0x0: No effect on bit value Write 0x1: Clear the bit to 0	RW	0x0

Bits	Field Name	Description	Type	Reset
0	IDPULLUP	Pull-up to the (OTG) ID line to allow its sampling Write 0x0: No effect on bit value Write 0x1: Clear the bit to 0	RW	0x0

Table 24-95. Register Call Summary for Register ULPI_OTG_CTRL_CLR_i

High-Speed USB Host Subsystem

- High-Speed USB Host Subsystem Register Mapping Summary: [0]

Table 24-96. ULPI_USB_INT_EN_RISE_i

Address Offset	0x0000 000D + (0x100 * i)	**Index**	i = 0 to 2
Physical Address	0x4806 280D + (0x100 * i)	**Instance**	ULPI
Description	Enables an interrupt event notification when the corresponding status bit changes from low to high. By default, all transitions are enabled. Read/Write address.		
Type	RW		

7	6	5	4	3	2	1	0
RESERVED			IDGND_RISE	SESSEND_RISE	SESSVALID_RISE	VBUSVALID_RISE	HOSTDISCONNECT_RISE

Bits	Field Name	Description	Type	Reset
7:5	RESERVED	Reserved	R	0x0
4	IDGND_RISE	Generate an interrupt event notification when IdGnd changes from low to high. Event is automatically masked if IdPullup bit is cleared to 0 and for 50 ms after IdPullup is set to 1.	RW	0x1
3	SESSEND_RISE	Generate an interrupt event notification when SessEnd changes from low to high.	RW	0x1
2	SESSVALID_RISE	Generate an interrupt event notification when SessValid changes from low to high. SessValid is the same as UTMI+ AValid.	RW	0x1
1	VBUSVALID_RISE	Generate an interrupt event notification when VbusValid changes from low to high.	RW	0x1
0	HOSTDISCONNECT_RISE	Generate an interrupt event notification when Hostdisconnect changes from low to high. Applicable only in host mode (DpPulldown and DmPulldown both set to 1b).	RW	0x1

Table 24-97. Register Call Summary for Register ULPI_USB_INT_EN_RISE_i

High-Speed USB Host Subsystem

- USBTLL Module Functionality: [0]
- High-Speed USB Host Subsystem Register Mapping Summary: [1]

Table 24-98. ULPI_USB_INT_EN_RISE_SET_i

Address Offset	0x0000 000E + (0x100 * i)	**Index**	i = 0 to 2
Physical Address	0x4806 280E + (0x100 * i)	**Instance**	ULPI
Description	Enables an interrupt event notification when the corresponding status bit changes from low to high. Read/set address (write 1 to a bit to set it to 1; writing 0 has no effect on bit value). See fields description at the read/write address of the same register.		
Type	RW		

7	6	5	4	3	2	1	0
RESERVED			IDGND_RISE	SESSEND_RISE	SESSVALID_RISE	VBUSVALID_RISE	HOSTDISCONNECT_RISE

Bits	Field Name	Description	Type	Reset
7:5	RESERVED	Reserved	R	0x0
4	IDGND_RISE	Generate an interrupt event notification when IdGnd changes from low to high. Event is automatically masked if the IdPullup bit is cleared to 0 and for 50 ms after IdPullup is set to 1. Write 0x0: No effect on bit value Write 0x1: Set the bit to 1.	RW	0x0
3	SESSEND_RISE	Generate an interrupt event notification when SessEnd changes from low to high. Write 0x0: No effect on bit value Write 0x1: Set the bit to 1.	RW	0x0
2	SESSVALID_RISE	Generate an interrupt event notification when SessValid changes from low to high. SessValid is the same as UTMI+ AValid. Write 0x0: No effect on bit value Write 0x1: Set the bit to 1.	RW	0x0
1	VBUSVALID_RISE	Generate an interrupt event notification when VbusValid changes from low to high. Write 0x0: No effect on bit value Write 0x1: Set the bit to 1.	RW	0x0
0	HOSTDISCONNECT_RISE	Generate an interrupt event notification when Hostdisconnect changes from low to high. Applicable only in host mode (DpPulldown and DmPulldown both set to 1b). Write 0x0: No effect on bit value Write 0x1: Set the bit to 1.	RW	0x0

Table 24-99. Register Call Summary for Register ULPI_USB_INT_EN_RISE_SET_i

High-Speed USB Host Subsystem

- High-Speed USB Host Subsystem Register Mapping Summary: [0]

Table 24-100. ULPI_USB_INT_EN_RISE_CLR_i

Address Offset	0x0000 000F + (0x100 * i)	**Index**	i = 0 to 2
Physical Address	0x4806 280F + (0x100 * i)	**Instance**	ULPI
Description	Enables an interrupt event notification when the corresponding status bit changes from low to high. Read/clear address (write 1 to a bit to clear it to 0, writing 0 has no effect on bit value). See fields description at the read/write address of the same register.		
Type	RW		

7	6	5	4	3	2	1	0
RESERVED			IDGND_RISE	SESSEND_RISE	SESSVALID_RISE	VBUSVALID_RISE	HOSTDISCONNECT_RISE

Bits	Field Name	Description	Type	Reset
7:5	RESERVED	Reserved	R	0x0
4	IDGND_RISE	Generate an interrupt event notification when IdGnd changes from low to high. Event is automatically masked if the IdPullup bit is cleared to 0 and for 50 ms after IdPullup is set to 1. Write 0x0: No effect on bit value Write 0x1: Clear the bit to 0	RW	0x0
3	SESSEND_RISE	Generate an interrupt event notification when SessEnd changes from low to high. Write 0x0: No effect on bit value Write 0x1: Clear the bit to 0	RW	0x0
2	SESSVALID_RISE	Generate an interrupt event notification when SessValid changes from low to high. SessValid is the same as UTMI+ AValid. Write 0x0: No effect on bit value Write 0x1: Clear the bit to 0	RW	0x0
1	VBUSVALID_RISE	Generate an interrupt event notification when VbusValid changes from low to high. Write 0x0: No effect on bit value Write 0x1: Clear the bit to 0	RW	0x0
0	HOSTDISCONNECT_RISE	Generate an interrupt event notification when Hostdisconnect changes from low to high. Applicable only in host mode (DpPulldown and DmPulldown both set to 1b). Write 0x0: No effect on bit value Write 0x1: Clear the bit to 0	RW	0x0

Table 24-101. Register Call Summary for Register ULPI_USB_INT_EN_RISE_CLR_i

High-Speed USB Host Subsystem

- High-Speed USB Host Subsystem Register Mapping Summary: [0]

Table 24-102. ULPI_USB_INT_EN_FALL_i

Address Offset	0x0000 0010 + (0x100 * i)	**Index**	i = 0 to 2
Physical Address	0x4806 2810 + (0x100 * i)	**Instance**	ULPI
Description	Enables an interrupt event notification when the corresponding status bit changes from high to low. By default, all transitions are enabled. Read/Write address.		
Type	RW		

7	6	5	4	3	2	1	0
RESERVED			IDGND_FALL	SESSEND_FALL	SESSVALID_FALL	VBUSVALID_FALL	HOSTDISCONNECT_FALL

Bits	Field Name	Description	Type	Reset
7:5	RESERVED	Reserved	R	0x0
4	IDGND_FALL	Generate an interrupt event notification when IdGnd changes from high to low. Event is automatically masked if IdPullup bit is cleared to 0 and for 50 ms after IdPullup is set to 1.	RW	0x1
3	SESSEND_FALL	Generate an interrupt event notification when SessEnd changes from high to low.	RW	0x1
2	SESSVALID_FALL	Generate an interrupt event notification when SessValid changes from high to low. SessValid is the same as UTMI+ AValid.	RW	0x1
1	VBUSVALID_FALL	Generate an interrupt event notification when VbusValid changes from high to low.	RW	0x1
0	HOSTDISCONNECT_FALL	Generate an interrupt event notification when Hostdisconnect changes from high to low. Applicable only in host mode (DpPulldown and DmPulldown both set to 1b).	RW	0x1

Table 24-103. Register Call Summary for Register ULPI_USB_INT_EN_FALL_i

High-Speed USB Host Subsystem

- USBTLL Module Functionality: [0]
- High-Speed USB Host Subsystem Register Mapping Summary: [1]

Table 24-104. ULPI_USB_INT_EN_FALL_SET_i

Address Offset	0x0000 0011 + (0x100 * i)	**Index**	i = 0 to 2
Physical Address	0x4806 2811 + (0x100 * i)	**Instance**	ULPI
Description	Enables an interrupt event notification when the corresponding status bit changes from high to low. Read/set address (write 1 to a bit to set it to 1; writing 0 has no effect on bit value). See fields description at the read/write address of the same register.		
Type	RW		

7	6	5	4	3	2	1	0
RESERVED			IDGND_FALL	SESSEND_FALL	SESSVALID_FALL	VBUSVALID_FALL	HOSTDISCONNECT_FALL

Bits	Field Name	Description	Type	Reset
7:5	RESERVED	Reserved	R	0x0
4	IDGND_FALL	Generate an interrupt event notification when IdGnd changes from high to low. Event is automatically masked if the IdPullup bit is cleared to 0 and for 50 ms after IdPullup is set to 1. Write 0x0: No effect on bit value Write 0x1: Set the bit to 1.	RW	0x0
3	SESSEND_FALL	Generate an interrupt event notification when SessEnd changes from high to low. Write 0x0: No effect on bit value Write 0x1: Set the bit to 1.	RW	0x0
2	SESSVALID_FALL	Generate an interrupt event notification when SessValid changes from high to low. SessValid is the same as UTMI+ AValid. Write 0x0: No effect on bit value Write 0x1: Set the bit to 1.	RW	0x0
1	VBUSVALID_FALL	Generate an interrupt event notification when VbusValid changes from high to low. Write 0x0: No effect on bit value Write 0x1: Set the bit to 1.	RW	0x0
0	HOSTDISCONNECT_FALL	Generate an interrupt event notification when Hostdisconnect changes from high to low. Applicable only in host mode (DpPulldown and DmPulldown both set to 1b). Write 0x0: No effect on bit value Write 0x1: Set the bit to 1.	RW	0x0

Table 24-105. Register Call Summary for Register ULPI_USB_INT_EN_FALL_SET_i

High-Speed USB Host Subsystem

- High-Speed USB Host Subsystem Register Mapping Summary: [0]

Table 24-106. ULPI_USB_INT_EN_FALL_CLR_i

Address Offset	0x0000 0012 + (0x100 * i)	**Index**	i = 0 to 2
Physical Address	0x4806 2812 + (0x100 * i)	**Instance**	ULPI
Description	Enables an interrupt event notification when the corresponding status bit changes from high to low. Read/clear address (write 1 to a bit to clear it to 0, writing 0 has no effect on bit value). See fields description at the read/write address of the same register.		
Type	RW		

7	6	5	4	3	2	1	0
RESERVED			IDGND_FALL	SESSEND_FALL	SESSVALID_FALL	VBUSVALID_FALL	HOSTDISCONNECT_FALL

Bits	Field Name	Description	Type	Reset
7:5	RESERVED	Reserved	R	0x0
4	IDGND_FALL	Generate an interrupt event notification when IdGnd changes from high to low. Event is automatically masked if the IdPullup bit is cleared to 0 and for 50 ms after IdPullup is set to 1. Write 0x0: No effect on bit value Write 0x1: Clear the bit to 0	RW	0x0
3	SESSEND_FALL	Generate an interrupt event notification when SessEnd changes from high to low. Write 0x0: No effect on bit value Write 0x1: Clear the bit to 0	RW	0x0
2	SESSVALID_FALL	Generate an interrupt event notification when SessValid changes from high to low. SessValid is the same as UTMI+ AValid. Write 0x0: No effect on bit value Write 0x1: Clear the bit to 0	RW	0x0
1	VBUSVALID_FALL	Generate an interrupt event notification when VbusValid changes from high to low. Write 0x0: No effect on bit value Write 0x1: Clear the bit to 0	RW	0x0
0	HOSTDISCONNECT_FALL	Generate an interrupt event notification when Hostdisconnect changes from high to low. Applicable only in host mode (DpPulldown and DmPulldown both set to 1b). Write 0x0: No effect on bit value Write 0x1: Clear the bit to 0	RW	0x0

Table 24-107. Register Call Summary for Register ULPI_USB_INT_EN_FALL_CLR_i

High-Speed USB Host Subsystem

- High-Speed USB Host Subsystem Register Mapping Summary: [0]

Table 24-108. ULPI_USB_INT_STATUS_i

Address Offset	0x0000 0013 + (0x100 * i)	**Index**	i = 0 to 2
Physical Address	0x4806 2813 + (0x100 * i)	**Instance**	ULPI
Description	Indicates the current value of the interrupt source signal.		
Type	R		

7	6	5	4	3	2	1	0
RESERVED			IDGND	SESSEND	SESSVALID	VBUSVALID	HOSTDISCONNECT

Bits	Field Name	Description	Type	Reset
7:5	RESERVED	Reserved	R	0x0
4	IDGND	Value of UTMI+ IdDig output. Undefined unless IdPullup = 1 0x0: ID pin is grounded = OTG A = default Host 0x1: ID pin is floating = OTG B = default Peripheral	R	0x0
3	SESSEND	Current value of UTMI+ SessEnd output. 0x0: VBUS is above Session-End threshold 0x1: VBUS is below Session-End threshold	R	0x0
2	SESSVALID	Current value of UTMI+ SessValid output. SessValid is the same as UTMI+ AValid. 0x0: VBUS is below Session-Valid threshold 0x1: VBUS is above Session-Valid threshold	R	0x0
1	VBUSVALID	Current value of UTMI+ VbusValid output. 0x0: VBUS is below Vbus-Valid threshold 0x1: VBUS is above Vbus-Valid threshold	R	0x0
0	HOSTDISCONNECT	Current value of UTMI+ Hostdisconnect output. Applicable only in host mode. Automatically reset to 0 when Low Power Mode is entered. 0x0: Peripheral not disconnected or non-host mode 0x1: Peripheral disconnected	R	0x0

Table 24-109. Register Call Summary for Register ULPI_USB_INT_STATUS_i

High-Speed USB Host Subsystem

- USBTLL Module Functionality: [0]
- High-Speed USB Host Subsystem Register Mapping Summary: [1]

Table 24-110. ULPI_USB_INT_LATCH_i

Address Offset	0x0000 0014 + (0x100 * i)	**Index**	i = 0 to 2
Physical Address	0x4806 2814 + (0x100 * i)	**Instance**	ULPI
Description	Set by unmasked changes on the corresponding status bits to generate the ULPI interrupt. Cleared upon read, and when Low Power Mode, Serial Mode or Carkit Mode are entered.		
Type	R		

7	6	5	4	3	2	1	0
RESERVED			IDGND_LATCH	SESSEND_LATCH	SESSVALID_LATCH	VBUSVALID_LATCH	HOSTDISCONNECT_LATCH

Bits	Field Name	Description	Type	Reset
7:5	RESERVED	Reserved	R	0x0
4	IDGND_LATCH	Set to 1 by the PHY when an unmasked event occurs on IdGnd. Cleared when this register is read.	R	0x0
3	SESSEND_LATCH	Set to 1 by the PHY when an unmasked event occurs on SessEnd. Cleared when this register is read.	R	0x0
2	SESSVALID_LATCH	Set to 1 by the PHY when an unmasked event occurs on SessValid. Cleared when this register is read. SessValid is the same as UTMI+ AValid.	R	0x0

Bits	Field Name	Description	Type	Reset
1	VBUSVALID_LATCH	Set to 1 by the PHY when an unmasked event occurs on VbusValid. Cleared when this register is read.	R	0x0
0	HOSTDISCONNECT_LATCH	Set to 1 by the PHY when an unmasked event occurs on Hostdisconnect. Cleared when this register is read. Applicable only in host mode.	R	0x0

Table 24-111. Register Call Summary for Register ULPI_USB_INT_LATCH_i

High-Speed USB Host Subsystem

- High-Speed USB Host Subsystem Register Mapping Summary: [0]

Table 24-112. ULPI_DEBUG_i

Address Offset	0x0000 0015 + (0x100 * i)	**Index**	i = 0 to 2
Physical Address	0x4806 2815 + (0x100 * i)	**Instance**	ULPI
Description	Indicates the current value of various signals useful for debugging.		
Type	R		

7	6	5	4	3	2	1	0
RESERVED						LINESTATE	

Bits	Field Name	Description	Type	Reset
7:2	RESERVED	Reserved	R	0x00
1:0	LINESTATE	Current state of the USB line: D+ (bit 0) and D- (bit 1). 0x0: SE0 (LS/FS), Squelch (HS/Chirp) 0x1: LS: 'K' State, FS: 'J' State, HS: !Squelch, Chirp: !Squelch & HS_Differential_Receiver_Output 0x2: LS: 'J' State, FS: 'K' State, HS: Invalid, Chirp: !Squelch & !HS_Differential_Receiver_Output 0x3: SE1 (LS/FS), Invalid (HS/Chirp)	R	0x0

Table 24-113. Register Call Summary for Register ULPI_DEBUG_i

High-Speed USB Host Subsystem

- High-Speed USB Host Subsystem Register Mapping Summary: [0]

Table 24-114. ULPI_SCRATCH_REGISTER_i

Address Offset	0x0000 0016 + (0x100 * i)	**Index**	i = 0 to 2
Physical Address	0x4806 2816 + (0x100 * i)	**Instance**	ULPI
Description	Register byte for register access testing purposes. Value has no functional effect on PHY. Read/Write address.		
Type	RW		

7	6	5	4	3	2	1	0
SCRATCH							

Bits	Field Name	Description	Type	Reset
7:0	SCRATCH	Scratch data.	RW	0x00

Table 24-115. Register Call Summary for Register ULPI_SCRATCH_REGISTER_i

High-Speed USB Host Subsystem

- High-Speed USB Host Subsystem Register Mapping Summary: [0]

Table 24-116. ULPI_SCRATCH_REGISTER_SET_i

Address Offset	0x0000 0017 + (0x100 * i)	**Index**	i = 0 to 2
Physical Address	0x4806 2817 + (0x100 * i)	**Instance**	ULPI
Description	Register byte for register access testing purposes. Value has no functional effect on PHY. Read/set address (write 1 to a bit to set it to 1; writing 0 has no effect on bit value). See fields description at the read/write address of the same register.		
Type	RW		

7	6	5	4	3	2	1	0
SCRATCH							

Bits	Field Name	Description	Type	Reset
7:0	SCRATCH	Scratch data write 1 to a bit to set it to 1 writing 0 has no effect on bit value	RW	0x00

Table 24-117. Register Call Summary for Register ULPI_SCRATCH_REGISTER_SET_i

High-Speed USB Host Subsystem

- High-Speed USB Host Subsystem Register Mapping Summary: [0]

Table 24-118. ULPI_SCRATCH_REGISTER_CLR_i

Address Offset	0x0000 0018 + (0x100 * i)	**Index**	i = 0 to 2
Physical Address	0x4806 2818 + (0x100 * i)	**Instance**	ULPI
Description	Register byte for register access testing purposes. Value has no functional effect on PHY. Read/clear address (write 1 to a bit to clear it to 0, writing 0 has no effect on bit value). See fields description at the read/write address of the same register.		
Type	RW		

7	6	5	4	3	2	1	0
SCRATCH							

Bits	Field Name	Description	Type	Reset
7:0	SCRATCH	Scratch data write 1 to a bit to clear it to 0 writing 0 has no effect on bit value	RW	0x00

Table 24-119. Register Call Summary for Register ULPI_SCRATCH_REGISTER_CLR_i

High-Speed USB Host Subsystem

- High-Speed USB Host Subsystem Register Mapping Summary: [0]

Table 24-120. ULPI_EXTENDED_SET_ACCESS_i

Address Offset	0x0000 002F + (0x100 * i)	**Index**	i = 0 to 2
Physical Address	0x4806 282F + (0x100 * i)	**Instance**	ULPI
Description	This address is used to access the extended register set, that is, addresses above 0x40.		
Type	UNDEFINED_TYPE_STRING		

7	6	5	4	3	2	1	0
SET_ACCESS							

Bits	Field Name	Description	Type	Reset
7:0	SET_ACCESS	This bitfield is used to access the extended register set, that is, addresses above 0x40.	RW	0x00

Table 24-121. Register Call Summary for Register ULPI_EXTENDED_SET_ACCESS_i

High-Speed USB Host Subsystem

- High-Speed USB Host Subsystem Register Mapping Summary: [0]

Table 24-122. ULPI_UTMI_VCONTROL_EN_i

Address Offset	0x0000 0030 + (0x100 * i)	**Index**	i = 0 to 2
Physical Address	0x4806 2830 + (0x100 * i)	**Instance**	ULPI
Description	Part of non-standard UTMI-to-ULPI mailbox system, implemented if HDL generic VCS_MAILBOX bit is 1 Enables an interrupt notification when the corresponding vcontrol_status bit changes. Read/Write address. Lowest VCS_CTRL_WIDTH (HDL generic) bits are implemented, others are always-0, read-only. (UTMI standard is 4-bit).		
Type	RW		

7	6	5	4	3	2	1	0
VC7_EN	VC6_EN	VC5_EN	VC4_EN	VC3_EN	VC2_EN	VC1_EN	VC0_EN

Bits	Field Name	Description	Type	Reset
7	VC7_EN	Enable alt_int assertion upon vcontrol_status bit change	RW	0x0
6	VC6_EN	Enable alt_int assertion upon vcontrol_status bit change	RW	0x0
5	VC5_EN	Enable alt_int assertion upon vcontrol_status bit change	RW	0x0
4	VC4_EN	Enable alt_int assertion upon vcontrol_status bit change	RW	0x0
3	VC3_EN	Enable alt_int assertion upon vcontrol_status bit change	RW	0x0
2	VC2_EN	Enable alt_int assertion upon vcontrol_status bit change	RW	0x0
1	VC1_EN	Enable alt_int assertion upon vcontrol_status bit change	RW	0x0
0	VC0_EN	Enable alt_int assertion upon vcontrol_status bit change	RW	0x0

Table 24-123. Register Call Summary for Register ULPI_UTMI_VCONTROL_EN_i

High-Speed USB Host Subsystem

- High-Speed USB Host Subsystem Register Mapping Summary: [0]

Table 24-124. ULPI_UTMI_VCONTROL_EN_SET_i

Address Offset	0x0000 0031 + (0x100 * i)	**Index**	i = 0 to 2
Physical Address	0x4806 2831 + (0x100 * i)	**Instance**	ULPI
Description	Part of non-standard UTMI-to-ULPI mailbox system, implemented if HDL generic VCS_MAILBOX bit is 1 Enables an interrupt notification when the corresponding vcontrol_status bit changes. Read/set address (write 1 to a bit to set it to 1; writing 0 has no effect on bit value). See fields description at the read/write address of the same register.		
Type	RW		

7	6	5	4	3	2	1	0
VC7_EN	VC6_EN	VC5_EN	VC4_EN	VC3_EN	VC2_EN	VC1_EN	VC0_EN

Bits	Field Name	Description	Type	Reset
7	VC7_EN	Enable alt_int assertion upon vcontrol_status bit change Write 0x0: No effect on bit value Write 0x1: Set the bit to 1.	RW	0x0
6	VC6_EN	Enable alt_int assertion upon vcontrol_status bit change Write 0x0: No effect on bit value Write 0x1: Set the bit to 1.	RW	0x0

Bits	Field Name	Description	Type	Reset
5	VC5_EN	Enable alt_int assertion upon vcontrol_status bit change Write 0x0: No effect on bit value Write 0x1: Set the bit to 1.	RW	0x0
4	VC4_EN	Enable alt_int assertion upon vcontrol_status bit change Write 0x0: No effect on bit value Write 0x1: Set the bit to 1.	RW	0x0
3	VC3_EN	Enable alt_int assertion upon vcontrol_status bit change Write 0x0: No effect on bit value Write 0x1: Set the bit to 1.	RW	0x0
2	VC2_EN	Enable alt_int assertion upon vcontrol_status bit change Write 0x0: No effect on bit value Write 0x1: Set the bit to 1.	RW	0x0
1	VC1_EN	Enable alt_int assertion upon vcontrol_status bit change Write 0x0: No effect on bit value Write 0x1: Set the bit to 1.	RW	0x0
0	VC0_EN	Enable alt_int assertion upon vcontrol_status bit change Write 0x0: No effect on bit value Write 0x1: Set the bit to 1.	RW	0x0

Table 24-125. Register Call Summary for Register ULPI_UTMI_VCONTROL_EN_SET_i

High-Speed USB Host Subsystem

- High-Speed USB Host Subsystem Register Mapping Summary: [0]

Table 24-126. ULPI_UTMI_VCONTROL_EN_CLR_i

Address Offset	0x0000 0032 + (0x100 * i)	**Index**	i = 0 to 2
Physical Address	0x4806 2832 + (0x100 * i)	**Instance**	ULPI
Description	Part of non-standard UTMI-to-ULPI mailbox system, implemented if HDL generic VCS_MAILBOX bit is 1 Enables an interrupt notification when the corresponding vcontrol_status bit changes. Read/clear address (write 1 to a bit to clear it to 0, writing 0 has no effect on bit value). See fields description at the read/write address of the same register.		
Type	RW		

7	6	5	4	3	2	1	0
VC7_EN	VC6_EN	VC5_EN	VC4_EN	VC3_EN	VC2_EN	VC1_EN	VC0_EN

Bits	Field Name	Description	Type	Reset
7	VC7_EN	Enable alt_int assertion upon vcontrol_status bit change Write 0x0: No effect on bit value Write 0x1: Clear the bit to 0	RW	0x0
6	VC6_EN	Enable alt_int assertion upon vcontrol_status bit change Write 0x0: No effect on bit value Write 0x1: Clear the bit to 0	RW	0x0
5	VC5_EN	Enable alt_int assertion upon vcontrol_status bit change Write 0x0: No effect on bit value Write 0x1: Clear the bit to 0	RW	0x0
4	VC4_EN	Enable alt_int assertion upon vcontrol_status bit change Write 0x0: No effect on bit value Write 0x1: Clear the bit to 0	RW	0x0
3	VC3_EN	Enable alt_int assertion upon vcontrol_status bit change Write 0x0: No effect on bit value Write 0x1: Clear the bit to 0	RW	0x0
2	VC2_EN	Enable alt_int assertion upon vcontrol_status bit change Write 0x0: No effect on bit value Write 0x1: Clear the bit to 0	RW	0x0
1	VC1_EN	Enable alt_int assertion upon vcontrol_status bit change Write 0x0: No effect on bit value Write 0x1: Clear the bit to 0	RW	0x0

Bits	Field Name	Description	Type	Reset
0	VC0_EN	Enable alt_int assertion upon vcontrol_status bit change Write 0x0: No effect on bit value Write 0x1: Clear the bit to 0	RW	0x0

Table 24-127. Register Call Summary for Register ULPI_UTMI_VCONTROL_EN_CLR_i

High-Speed USB Host Subsystem

- High-Speed USB Host Subsystem Register Mapping Summary: [0]

Table 24-128. ULPI_UTMI_VCONTROL_STATUS_i

Address Offset	0x0000 0033 + (0x100 * i)	**Index**	i = 0 to 2
Physical Address	0x4806 2833 + (0x100 * i)	**Instance**	ULPI
Description	Part of non-standard UTMI-to-ULPI mailbox system, implemented if HDL generic VCS_MAILBOX bit is 1 UTMI-standard Vcontrol vector byte is sent by the UTMI controller (other side of TLL) to its PHY (emulated here by the TLL). Alternatively, data can be also written directly into the register. Can contain any user-defined data. Vcontrol bit changes can be used to assert the ULPI ALT interrupt. Lowest VCS_CTRL_WIDTH (HDL generic) bits are implemented, others are always-0, read-only. (UTMI standard is 4-bit).		
Type	RW		

7	6	5	4	3	2	1	0
VC							

Bits	Field Name	Description	Type	Reset
7:0	VC	User-defined UTMI Control data byte	RW	0x00

Table 24-129. Register Call Summary for Register ULPI_UTMI_VCONTROL_STATUS_i

High-Speed USB Host Subsystem

- High-Speed USB Host Subsystem Register Mapping Summary: [0]

Table 24-130. ULPI_UTMI_VCONTROL_LATCH_i

Address Offset	0x0000 0034 + (0x100 * i)	**Index**	i = 0 to 2
Physical Address	0x4806 2834 + (0x100 * i)	**Instance**	ULPI
Description	Part of non-standard UTMI-to-ULPI mailbox system, implemented if HDL generic VCS_MAILBOX bit is 1 Set by unmasked changes on the corresponding vcontrol_status bits to generate the ULPI ALT interrupt. Cleared upon read, and when Low Power Mode, Serial Mode or Carkit Mode are entered. Lowest VCS_CTRL_WIDTH (HDL generic) bits are implemented, others are always-0, read-only. (UTMI standard is 4-bit).		
Type	R		

7	6	5	4	3	2	1	0
VC7_CHANGE	VC6_CHANGE	VC5_CHANGE	VC4_CHANGE	VC3_CHANGE	VC2_CHANGE	VC1_CHANGE	VC0_CHANGE

Bits	Field Name	Description	Type	Reset
7	VC7_CHANGE	Unmasked change on vcontrol_status bit	R	0x0
6	VC6_CHANGE	Unmasked change on vcontrol_status bit	R	0x0
5	VC5_CHANGE	Unmasked change on vcontrol_status bit	R	0x0
4	VC4_CHANGE	Unmasked change on vcontrol_status bit	R	0x0
3	VC3_CHANGE	Unmasked change on vcontrol_status bit	R	0x0
2	VC2_CHANGE	Unmasked change on vcontrol_status bit	R	0x0
1	VC1_CHANGE	Unmasked change on vcontrol_status bit	R	0x0
0	VC0_CHANGE	Unmasked change on vcontrol_status bit	R	0x0

Table 24-131. Register Call Summary for Register ULPI_UTMI_VCONTROL_LATCH_i

High-Speed USB Host Subsystem

- High-Speed USB Host Subsystem Register Mapping Summary: [0]

Table 24-132. ULPI_UTMI_VSTATUS_i

Address Offset	0x0000 0035 + (0x100 * i)	**Index**	i = 0 to 2
Physical Address	0x4806 2835 + (0x100 * i)	**Instance**	ULPI
Description	Part of non-standard UTMI-to-ULPI mailbox system, implemented if HDL generic VCS_MAILBOX bit is 1 UTMI-standard Vstatus vector byte is sent by the PHY (emulated here by the TLL) to the UTMI controller (other side of TLL): Information written into this register will go directly to the UTMI controller, and can contain any user-defined data. Read/Write address. Lowest VCS_STAT_WIDTH (HDL generic) bits are implemented, others are always-0, read-only. (UTMI standard is 8-bit).		
Type	RW		

7	6	5	4	3	2	1	0
VS							

Bits	Field Name	Description	Type	Reset
7:0	VS	User-defined UTMI Status data byte	RW	0x00

Table 24-133. Register Call Summary for Register ULPI_UTMI_VSTATUS_i

High-Speed USB Host Subsystem

- High-Speed USB Host Subsystem Register Mapping Summary: [0]

Table 24-134. ULPI_UTMI_VSTATUS_SET_i

Address Offset	0x0000 0036 + (0x100* i)	**Index**	i = 0 to 2
Physical Address	0x4806 2836 + (0x100 * i)	**Instance**	ULPI
Description	Part of non-standard UTMI-to-ULPI mailbox system, implemented if HDL generic VCS_MAILBOX bit is 1 UTMI-standard Vstatus vector byte is sent by the PHY (emulated here by the TLL) to the UTMI controller (other side of TLL): Information written into this register will go directly to the UTMI controller, and can contain any user-defined data. Read/set address (write 1 to a bit to set it to 1; writing 0 has no effect on bit value). See fields description at the read/write address of the same register.		
Type	RW		

7	6	5	4	3	2	1	0
VS							

Bits	Field Name	Description	Type	Reset
7:0	VS	User-defined UTMI status data byte Write 0x0: No effect on bit value Write 0x1: Set the bit to 1.	RW	0x00

Table 24-135. Register Call Summary for Register ULPI_UTMI_VSTATUS_SET_i

High-Speed USB Host Subsystem

- High-Speed USB Host Subsystem Register Mapping Summary: [0]

Table 24-136. ULPI_UTMI_VSTATUS_CLR_i

Address Offset	0x0000 0037 + (0x100 * i)	**Index**	i = 0 to 2
Physical Address	0x4806 2837 + (0x100 * i)	**Instance**	ULPI
Description	Part of non-standard UTMI-to-ULPI mailbox system, implemented if HDL generic VCS_MAILBOX bit is 1 UTMI-standard Vstatus vector byte is sent by the PHY (emulated here by the TLL) to the UTMI controller (other side of TLL): Information written into this register will go directly to the UTMI controller, and can contain any user-defined data. Read/clear address (write 1 to a bit to clear it to 0, writing 0 has no effect on bit value). See fields description at the read/write address of the same register.		
Type	RW		

7	6	5	4	3	2	1	0
VS							

Bits	Field Name	Description	Type	Reset
7:0	VS	User-defined UTMI status data byte Write 0x0: No effect on bit value Write 0x1: Clear the bit to 0	RW	0x00

Table 24-137. Register Call Summary for Register ULPI_UTMI_VSTATUS_CLR_i

High-Speed USB Host Subsystem

- High-Speed USB Host Subsystem Register Mapping Summary: [0]

Table 24-138. ULPI_USB_INT_LATCH_NOCLR_i

Address Offset	0x0000 0038 + (0x100 * i)	**Index**	i = 0 to 2
Physical Address	0x4806 2838 + (0x100 * i)	**Instance**	ULPI
Description	Set by unmasked changes on the corresponding status bits to generate the ULPI interrupt. Debug, non-standard address to the standard register: Register is not cleared on read. See fields description at the "clear-on-read" address of the same register.		
Type	UNDEFINED_TYPE_STRING		

7	6	5	4	3	2	1	0
RESERVED			IDGND_LATCH	SESSEND_LATCH	SESSVALID_LATCH	VBUSVALID_LATCH	HOSTDISCONNECT_LATCH

Bits	Field Name	Description	Type	Reset
7:5	RESERVED	Reserved	R	0x0
4	IDGND_LATCH	Set to 1 by the PHY when an unmasked event occurs on IdGnd.	R	0x0
3	SESSEND_LATCH	Set to 1 by the PHY when an unmasked event occurs on SessEnd.	R	0x0
2	SESSVALID_LATCH	Set to 1 by the PHY when an unmasked event occurs on SessValid. SessValid is the same as UTMI+ AValid.	R	0x0
1	VBUSVALID_LATCH	Set to 1 by the PHY when an unmasked event occurs on VbusValid.	R	0x0
0	HOSTDISCONNECT_LATCH	Set to 1 by the PHY when an unmasked event occurs on Hostdisconnect. Applicable only in host mode.	R	0x0

Table 24-139. Register Call Summary for Register ULPI_USB_INT_LATCH_NOCLR_i

High-Speed USB Host Subsystem

- High-Speed USB Host Subsystem Register Mapping Summary: [0]

Table 24-140. ULPI_VENDOR_INT_EN_i

Address Offset	0x0000 003B + (0x100 * i)	**Index**	i = 0 to 2
Physical Address	0x4806 283B + (0x100 * i)	**Instance**	ULPI
Description	Vendor-specific interrupt enables (mask) for miscellaneous ULPI alt_int events. Read/Write address.		
Type	RW		

7	6	5	4	3	2	1	0
RESERVED							P2P_EN

Bits	Field Name	Description	Type	Reset
7:1	RESERVED	Reserved	R	0x00
0	P2P_EN	Enable PHY-to-PHY ULPI wakeup upon inactive UTMI suspendm. 0x0: PHY-to-PHY wakeup enabled 0x1: PHY-to-PHY wakeup enabled	RW	0x0

Table 24-141. Register Call Summary for Register ULPI_VENDOR_INT_EN_i

High-Speed USB Host Subsystem

- USBTLL Module Functionality: [0]
- High-Speed USB Host Subsystem Register Mapping Summary: [1]

Table 24-142. ULPI_VENDOR_INT_EN_SET_i

Address Offset	0x0000 003C + (0x100 * i)	**Index**	i = 0 to 2
Physical Address	0x4806 283C + (0x100 * i)	**Instance**	ULPI
Description	Vendor-specific interrupt enable bit (mask) for miscellaneous ULPI alt_int events. Read/set address (write 1 to a bit to set it to 1; writing 0 has no effect on bit value). See fields description at the read/write address of the same register.		
Type	RW		

7	6	5	4	3	2	1	0
RESERVED							P2P_EN

Bits	Field Name	Description	Type	Reset
7:1	RESERVED	Reserved	R	0x00
0	P2P_EN	Enable PHY-to-PHY ULPI wakeup upon inactive UTMI suspendm. Write 0x0: No effect on bit value Write 0x1: Set the bit to 1.	RW	0x0

Table 24-143. Register Call Summary for Register ULPI_VENDOR_INT_EN_SET_i

High-Speed USB Host Subsystem

- High-Speed USB Host Subsystem Register Mapping Summary: [0]

Table 24-144. ULPI_VENDOR_INT_EN_CLR_i

Address Offset	0x0000 003D + (0x100 * i)	**Index**	i = 0 to 2
Physical Address	0x4806 283D + (0x100 * i)	**Instance**	ULPI
Description	Vendor-specific interrupt enables (mask) for miscellaneous ULPI alt_int events. Read/clear address (write 1 to a bit to clear it to 0, writing 0 has no effect on bit value). See fields description at the read/write address of the same register.		
Type	RW		

7	6	5	4	3	2	1	0
RESERVED							P2P_EN

Bits	Field Name	Description	Type	Reset
7:1	RESERVED	Reserved	R	0x00
0	P2P_EN	Enable PHY-to-PHY ULPI wakeup upon inactive UTMI suspendm. Write 0x0: No effect on bit value Write 0x1: Clear the bit to 0	RW	0x0

Table 24-145. Register Call Summary for Register ULPI_VENDOR_INT_EN_CLR_i

High-Speed USB Host Subsystem

- High-Speed USB Host Subsystem Register Mapping Summary: [0]

Table 24-146. ULPI_VENDOR_INT_STATUS_i

Address Offset	0x0000 003E + (0x100 * i)	**Index**	i = 0 to 2
Physical Address	0x4806 283E + (0x100 * i)	**Instance**	ULPI
Description	Vendor-specific interrupt sources for miscellaneous ULPI alt_int events.		
Type	R		

7	6	5	4	3	2	1	0
RESERVED							UTMI_SUSPENDM

Bits	Field Name	Description	Type	Reset
7:1	RESERVED	Reserved	R	0x00
0	UTMI_SUSPENDM	UTMI suspendm status (active-low), source of TLL PHY-to-PHY wakeup interrupt. 0x0: UTMI interface is suspended 0x1: UTMI interface is active (not suspended)	R	0x1

Table 24-147. Register Call Summary for Register ULPI_VENDOR_INT_STATUS_i

High-Speed USB Host Subsystem

- USBTLL Module Functionality: [0]
- High-Speed USB Host Subsystem Register Mapping Summary: [1]

Table 24-148. ULPI_VENDOR_INT_LATCH_i

Address Offset	0x0000 003F + (0x100 * i)	**Index**	i = 0 to 2
Physical Address	0x4806 283F + (0x100 * i)	**Instance**	ULPI
Description	Vendor-specific interrupt latches for miscellaneous ULPI alt_int events. Cleared upon read, and when Low Power Mode, Serial Mode or Carkit Mode are entered.		
Type	R		

7	6	5	4	3	2	1	0
RESERVED							P2P_LATCH

Bits	Field Name	Description	Type	Reset
7:1	RESERVED	Reserved	R	0x00
0	P2P_LATCH	PHY-to-PHY ULPI wakeup event latch. Set when ULPI is in low-power mode (suspendm = 0) and UTMI is active (suspendm = 1). 0x0: No PHY-to-PHY wakeup event latched 0x1: PHY-to-PHY wakeup event was latched, ALT interrupt active	R	0x0

Table 24-149. Register Call Summary for Register ULPI_VENDOR_INT_LATCH_i

High-Speed USB Host Subsystem

- High-Speed USB Host Subsystem Register Mapping Summary: [0]

24.2.6.4.2 UHH_config Registers

Table 24-150. UHH_REVISION

Address Offset	0x0000 0000		
Physical Address	0x4806 4000	**Instance**	UHH_config
Description	Standard revision number, BCD encoded Revision = <maj>.<min>		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																								MAJ_REV				MIN_REV			

Bits	Field Name	Description	Type	Reset
31:8	RESERVED	Reserved	R	0x000000
7:4	MAJ_REV	Major revision number 0..9	R	0x1
3:0	MIN_REV	Minor revision number 0..9	R	0x0

Table 24-151. Register Call Summary for Register UHH_REVISION

High-Speed USB Host Subsystem

- High-Speed USB Host Subsystem Register Mapping Summary: [0]

Table 24-152. UHH_SYSCONFIG

Address Offset	0x0000 0010		
Physical Address	0x4806 4010	**Instance**	UHH_config
Description	Standard system configuration register		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																		MIDLEMODE		RESERVED				CLOCKACTIVITY	RESERVED				SIDLEMODE		ENAWAKEUP	SOFTRESET	AUTOIDLE

Bits	Field Name	Description	Type	Reset
31:14	RESERVED	Reserved	R	0x00000
13:12	MIDLEMODE	Master interface power management control. Standby/wait control 0x0: Force-standby mode. Mstandby asserted unconditionally 0x1: No-standby mode. Mstandby never asserted. 0x2: Smart-standby mode. Mstandby asserted when initiator activity stops	RW	0x0
11:9	RESERVED	Reserved	R	0x0
8	CLOCKACTIVITY	Control of clock internal gating while module is idle. One bit per clock, actual register width depends on the number of functional clocks controlled. Lower bit: Interface clock Upper bits (if any): Functional clocks 1: Clock is kept on during idle 0: Clock is switched off during idle	RW	0x0
7:5	RESERVED	Reserved	R	0x0
4:3	SIDLEMODE	Slave interface power management control. Idle Req/ack control 0x0: Force-Idle mode. Sidleack asserted after Idlereq assertion 0x1: No-idle mode. Sidleack never asserted. 0x2: Smart-idle mode. Sidleack asserted upon Idlereq assertion, after target activity is over	RW	0x0
2	ENAWAKEUP	Asynchronous wakeup generation control (Swakeup) 0x0: Wakeup generation disabled 0x1: Wakeup generation enabled	RW	0x0
1	SOFTRESET	Module software reset 0x0: No effect 0x1: Starts softreset sequence.	W	0x0
0	AUTOIDLE	Internal autogating control 0x0: Clock always running 0x1: When no activity on L3 interconnect, clock is cut off.	RW	0x1

Table 24-153. Register Call Summary for Register UHH_SYSCONFIG

High-Speed USB Host Subsystem

- Reset, Clocking, and Power-Management Scheme: [0] [1] [2] [3] [4]
- High-Speed USB Host Subsystem Register Mapping Summary: [5]

Table 24-154. UHH_SYSSTATUS

Address Offset	0x0000 0014		
Physical Address	0x4806 4014	**Instance**	UHH_config
Description	Standard system status register		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																													EHCI_RESETDONE	OHCI_RESETDONE	RESETDONE

Bits	Field Name	Description	Type	Reset
31:3	RESERVED	Reserved	R	0x00000000
2	EHCI_RESETDONE	Indicated when the EHCI HS host is out of reset	R	0x0
1	OHCI_RESETDONE	Indicates when the OHCI FS/LS host is out of reset	R	0x0
0	RESETDONE	Indicates when the USB Host has come out of reset 0x0: Reset is ongoing 0x1: Reset is done	R	0x0

Table 24-155. Register Call Summary for Register UHH_SYSSTATUS

High-Speed USB Host Subsystem

- High-Speed USB Host Subsystem Register Mapping Summary: [0]

Table 24-156. UHH_HOSTCONFIG

Address Offset	0x0000 0040		
Physical Address	0x4806 4040	**Instance**	UHH_config
Description	Static configuration of the OTG controller host		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																			P3_ULPI_BYPASS	P2_ULPI_BYPASS	P3_CONNECT_STATUS	P2_CONNECT_STATUS	P1_CONNECT_STATUS	RESERVED		ENA_INCR_ALIGN	ENA_INCR16	ENA_INCR8	ENA_INCR4	AUTOPPD_ON_OVERCUR_EN	P1_ULPI_BYPASS

Bits	Field Name	Description	Type	Reset
31:13	RESERVED	Reserved	R	0x00000000
12	P3_ULPI_BYPASS	Host controller (root hub) port 3 control. 0x0: ULPI port 3 is active (and UTMI port 3 is inactive) 0x1: UTMI port 3 is active (and ULPI port 3 is inactive)	RW	0

Bits	Field Name	Description	Type	Reset
11	P2_ULPI_BYPASS	Host controller (root hub) port 2 control. 0x0: ULPI port 2 is active (and UTMI port 2 is inactive) 0x1: UTMI port 2 is active (and ULPI port 2 is inactive)	RW	0
10	P3_CONNECT_STATUS	Connection status for port 3. 0x0: USB port 3 is disconnected 0x1: USB port 3 is connected and in use (also the default state)	RW	0x1
9	P2_CONNECT_STATUS	Connection status for port 2. 0x0: USB port 2 is disconnected 0x1: USB port 2 is connected and in use (also the default state)	RW	0x1
8	P1_CONNECT_STATUS	Connection status for port 1. 0x0: USB port 1 is disconnected 0x1: USB port 1 is connected and in use (also the default state)	RW	0x1
7:6	RESERVED	Reserved	RW	0x0
5	ENA_INCR_ALIGN	Force alignment of bursts to the respective burst-size boundaries 0x0: Disable burst type 0x1: Enable burst type	RW	0x0
4	ENA_INCR16	Control the use of INCR16-type bursts (in AHB sense) 0x0: Disable burst type 0x1: Enable burst type	RW	0x0
3	ENA_INCR8	Control the use of INCR8-type bursts (in AHB sense) 0x0: Disable burst type 0x1: Enable burst type	RW	0x0
2	ENA_INCR4	Control the use of INCR4-type bursts (in AHB sense) 0x0: Disable burst type 0x1: Enable burst type	RW	0x0
1	AUTOPPD_ON_OVERCUR_EN	Configure reaction upon port overcurrent condition:(1) 0x0: Port remains on upon overcurrent. 0x1: Port is powered down automatically upon overcurrent.	RW	0x0
0	P1_ULPI_BYPASS	Host controller (root hub) port 1 control. 0x0: ULPI port 1 is active (and UTMI port 1 is inactive) 0x1: UTMI port 1 is active (and ULPI port 1 is inactive)	RW	0

(1) This bit must not be used and must always be written to with its default (reset) value, because the hardware overcurrent inputs are tied off at the device level.

Table 24-157. Register Call Summary for Register UHH_HOSTCONFIG

High-Speed USB Host Subsystem

- High-Speed USB Host Controller Functionality: [0] [1] [2] [3] [4] [5] [6] [7] [8] [9] [10] [11] [12]
- Selecting and Configuring USB Connectivity: [13] [14] [15] [16] [17] [18] [19] [20] [21] [22] [23] [24] [25] [26] [27]
- High-Speed USB Host Subsystem Register Mapping Summary: [28]

Table 24-158. UHH_DEBUG_CSR

Address Offset	0x0000 0044		
Physical Address	0x4806 4044	**Instance**	UHH_config
Description	Debug control and status for the EHCI, OHCI hosts.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED												OHCI_CCS_3	OHCI_CCS_2	OHCI_CCS_1	OHCI_GLOBALSUSPEND	RESERVED								OCHI_CNTSEL	EHCI_SIMULATION_MODE	EHCI_FLADJ					

Bits	Field Name	Description	Type	Reset
31:20	RESERVED	Reserved	R	0x000
19	OHCI_CCS_3	Current Connect Status of port 3 0x0: No periph connected 0x1: Periph connected	R	0x0
18	OHCI_CCS_2	Current Connect Status of port 2 0x0: No periph connected 0x1: Periph connected	R	0x0
17	OHCI_CCS_1	Current Connect Status of port 1 0x0: No periph connected 0x1: Periph connected	R	0x0
16	OHCI_GLOBALSUSPEND	OHCI global suspend status, asserted 5ms after the suspend order. 0x0: Host is not suspended 0x1: Host is suspended	R	0x0
15:8	RESERVED	Reserved	R	0x00
7	OCHI_CNTSEL	Selection of a shorter "1 ms" counter in OHCI host, to speed up long USB phases like reset, resume, etc (Used only for simulation.) 0x0: Functional mode, 1ms = 12,000 x 12 MHz cycles 0x1: Simulation mode, 1ms = 7 x 12 MHz cycles = 583 ns	RW	0x0
6	EHCI_SIMULATION_MODE	Sets the PHY to non-driving mode. (Used only for simulation.) 0x0: Functional mode 0x1: PHY set to non-driving	RW	0x0
5:0	EHCI_FLADJ	EHCI host frame length adjust. Modify only when EHCI bitfield USBSTS.HCHalted = 1 Field value + 59,488 = 60,000 by default = number of 60 MHz UTMI/ULPI clock cycles per 1 ms USB frame = number of 480 MHz HS bits per 125 us HS USB microframe	RW	0x20

Table 24-159. Register Call Summary for Register UHH_DEBUG_CSR

High-Speed USB Host Subsystem

- High-Speed USB Host Subsystem Register Mapping Summary: [0]

24.2.6.4.3 OHCI Registers

Table 24-160. HCREVISION

Address Offset	0x0000 0000		
Physical Address	0x4806 4400	**Instance**	OHCI
Description	OHCI revision number		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																								REV							

Bits	Field Name	Description	Type	Reset
31:8	RESERVED	Reserved	R	0x000000
7:0	REV	OHCI specification revision the OHCI revision number upon which the USB host controller is based. Examples: 0x10 for 1.0 0x21 for 2.1	R	0x10

Table 24-161. Register Call Summary for Register HCREVISION

High-Speed USB Host Subsystem

- High-Speed USB Host Subsystem Register Mapping Summary: [0]

Table 24-162. HCCONTROL

Address Offset	0x0000 0004		
Physical Address	0x4806 4404	**Instance**	OHCI
Description	HC Operating Mode Register		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																					RWE	RWC	IR	HCFS		BLE	CLE	IE	PLE	CBSR	

Bits	Field Name	Description	Type	Reset
31:11	RESERVED	Reserved	R	0x000000
10	RWE	Remote wake-up enable This bit is used to enable or disable the remote wakeup feature upon detection of upstream resume signaling.	RW	0
9	RWC	Remote wake-up connected. This bit indicates whether the host controller supports remote wakeup signaling.	RW	0
8	IR	Interrupt routing. This bit determines the routing of interrupts generated by events registered in USBHOST.HCINTERRUPTSTATUS. 0x0: All interrupts are routed to the normal host bus interrupt mechanism. 0x1: interrupts are routed to the system management Interrupt.	RW	0
7:6	HCFS	Host Controller Functional State 0x0: HCFS: USB Reset 0x1: HCFS: USB Resume 0x2: HCFS: USB Operational 0x3: HCFS: USB Suspend	RW	0x0

Bits	Field Name	Description	Type	Reset
5	BLE	Bulk list processing enable. 0x0: Bulk ED list is not processed after the next SOF. 0x1: Enables processing of bulk ED list in the next frame.	RW	0
4	CLE	Control list processing enable. 0x0: Control ED list is not processed after the next SOF. 0x1: Enables processing of control ED list in the next frame.	RW	0
3	IE	Isochronous ED processing enabled by Host Controller Driver. 0x0: Isochronous EDs are not processed 0x1: Enables processing of isochronous EDs.	RW	0
2	PLE	Periodic list enable 0x0: Periodic ED lists are not processed after the next frame. 0x1: Enables processing of periodic ED lists in the next frame.	RW	0
1:0	CBSR	Control/bulk service ratio. Specifies the ratio between control and bulk EDs processed in a frame. 0x0: One control ED per bulk ED 0x1: Two control ED per bulk ED 0x2: Three control ED per bulk ED 0x3: Four control ED per bulk ED	RW	0x0

Table 24-163. Register Call Summary for Register HCCONTROL

High-Speed USB Host Subsystem

- High-Speed USB Host Subsystem Register Mapping Summary: [0]

Table 24-164. HCCOMMANDSTATUS

Address Offset	0x0000 0008		
Physical Address	0x4806 4408	**Instance**	OHCI
Description	HC Command and Status		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED														SOC		RESERVED												OCR	BLF	CLF	HCR

Bits	Field Name	Description	Type	Reset
31:18	RESERVED	Reserved	R	0x0000
17:16	SOC	Scheduling overrun count. This is used to monitor any persistent scheduling problems. These bits are incremented on each scheduling overrun error. It is initialized to 0x0 and wraps around at 0x3.	R	0x0
15:4	RESERVED	Reserved	R	0x000
3	OCR	Ownership change request. This bit is set to request a change of control of the host controller.	RW	0
2	BLF	Bulk list filled. This bit is used to indicate whether there are any TDs on the bulk list. It is set whenever it adds a TD to an ED in the bulk list.	RW	0
1	CLF	Control list filled. This bit is used to indicate whether there are any TDs on the control list. It is set whenever it adds a TD to an ED in the control list.	RW	0

Bits	Field Name	Description	Type	Reset
0	HCR	Host controller reset. (software reset) Set this bit to initiate a USB host controller reset. This resets most USB host controller OHCI registers. OHCI register accesses must not be attempted until a read of this register returns a 0. 0x0: No effect 0x1: USB host controller is reset.	RW	0

Table 24-165. Register Call Summary for Register HCCOMMANDSTATUS

High-Speed USB Host Subsystem

- High-Speed USB Host Subsystem Register Mapping Summary: [0]
- Register Description: [1]

Table 24-166. HCINTERRUPTSTATUS

Address Offset	0x0000 000C		
Physical Address	0x4806 440C	**Instance**	OHCI
Description	HC Interrupt Status		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED	OC	RESERVED																							RHSC	FNO	UE	RD	SF	WDH	SO

Bits	Field Name	Description	Type	Reset
31	RESERVED	Reserved	R	0
30	OC	Ownership change. This bit is set when the USBHOST.HCCOMMANDSTATUS[3] OCR bit is set. Read 0x1: An ownership change has occurred. Write 0x0: No effect Write 0x1: Clears this bit	R	0
29:7	RESERVED	Reserved	R	0x000000
6	RHSC	Root hub status change. When 0x1: A root hub status change has occurred. Write 0x0: No effect. Write 0x1: Clears this bit.	RW	0
5	FNO	Frame number overflow. When 0x1: A frame number overflow has occurred. Write 0x0: No effect. Write 0x1: Clears this bit.	RW	0
4	UE	Unrecoverable error. When 0x1: An unrecoverable error has occurred. Write 0x0: No effect. Write 0x1: Clears this bit.	RW	0
3	RD	Resume detected. When 0x1: A downstream device has issued a resume request. Write 0x0: No effect. Write 0x1: Clears this bit.	RW	0
2	SF	Start of frame. When 0x1: A SOF has been issued. Write 0x0: No effect. Write 0x1: Clears this bit.	RW	0

Bits	Field Name	Description	Type	Reset
1	WDH	Write done head. When 0x1: The USB host controller has updated the HCDONEHEAD register. Write 0x0: No effect. Write 0x1: Clears this bit.	RW	0
0	SO	Scheduling overrun. When 0x1: A scheduling overrun has occurred. Write 0x0: No effect. Write 0x1: Clears this bit	RW	0

Table 24-167. Register Call Summary for Register HCINTERRUPTSTATUS

High-Speed USB Host Subsystem

- High-Speed USB Host Subsystem Register Mapping Summary: [0]
- Register Description: [1]

Table 24-168. HCINTERRUPTENABLE

Address Offset	0x0000 0010		
Physical Address	0x4806 4410	**Instance**	OHCI
Description	HC Interrupt Enable		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MIE	OC	RESERVED																						RHSC	FNO	UE	RD	SF	WDH	SO	

Bits	Field Name	Description	Type	Reset
31	MIE	Master interrupt enable. When 0x1: Allows other enabled OHCI interrupt sources to propagate to the device interrupt controller. When 0x0: OHCI interrupt sources are ignored. Write 0x0: No effect. Write 0x1: Sets this bit.	RW	0
30	OC	Ownership change. Write 0x0: No effect. Write 0x1: Enable interrupt generation due to ownership change.	RW	0
29:7	RESERVED	Reserved	R	0x000000
6	RHSC	Root hub status change. When 0x1 and MIE is 0x1: Allows root hub status change interrupts to propagate to the device interrupt controller. When 0x0 or MIE is 0x0: Root hub status change interrupts do not propagate. Write 0x0: No effect. Write 0x1: Sets this bit.	RW	0
5	FNO	Frame number overflow. When 0x1 and MIE is 0x1: Allows FNO interrupts to propagate to the device interrupt controller. When 0x0 or MIE is 0x0: FNO interrupts do not propagate. Write 0x0: No effect. Write 0x1: Sets this bit.	RW	0
4	UE	Unrecoverable error. When 0x1 and MIE is 0x1: Allows UE interrupts to propagate to the device interrupt controller. When 0x0 or MIE is 0x0: UE interrupts do not propagate. Write 0x0: No effect. Write 0x1: Sets this bit.	RW	0

Bits	Field Name	Description	Type	Reset
3	RD	Resume detected. When 0x1 and MIE is 0x1: Allows RD interrupts to propagate to the device interrupt controller. When 0x0 or MIE is 0x0: RD interrupts do not propagate. Write 0x0: No effect. Write 0x1: Sets this bit.	RW	0
2	SF	Start of frame. When 0x1 and MIE is 0x1: Allows SF interrupts to propagate to the device interrupt controller. When 0x0 or MIE is 0x0: SF interrupts do not propagate. Write 0x0: No effect. Write 0x1: Sets this bit.	RW	0
1	WDH	Write done head. When 0x1 and MIE is 0x1: Allows WDH interrupts to propagate to the device interrupt controller. When 0x0 or MIE is 0x0: WDH interrupts do not propagate. Write 0x0: No effect. Write 0x1: Sets this bit.	RW	0
0	SO	Scheduling overrun. When 0x1 and MIE is 0x1: Allows SO interrupts to propagate to the device interrupt controller. When 0x0 or MIE is 0x0: SO interrupts do not propagate. Write 0x0: No effect. Write 0x1: Sets this bit.	RW	0

Table 24-169. Register Call Summary for Register HCINTERRUPTENABLE

High-Speed USB Host Subsystem

- High-Speed USB Host Subsystem Register Mapping Summary: [0]
- Register Description: [1] [2] [3] [4] [5] [6] [7] [8]

Table 24-170. HCINTERRUPTDISABLE

Address Offset	0x0000 0014		
Physical Address	0x4806 4414	**Instance**	OHCI
Description	HC Interrupt Disable		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MIE	OC	RESERVED																						RHSC	FNO	UE	RD	SF	WDH	SO	

Bits	Field Name	Description	Type	Reset
31	MIE	Master interrupt enable. Always reads 0x0. Write 0x0: No effect. Write 0x1: Clears the HCINTERRUPTENABLE MIE bit.	RW	0
30	OC	Ownership change. Write 0x0: No effect. Write 0x1: Disable interrupt generation due to ownership change.	RW	0
29:7	RESERVED	Reserved	R	0x000000
6	RHSC	Root hub status change. Always reads 0x0. Write 0x0: No effect. Write 0x1: Clears the HCINTERRUPTENABLE RHSC bit.	RW	0
5	FNO	Frame number overflow. Always reads 0x0. Write 0x0: No effect. Write 0x1: Clears the HCINTERRUPTENABLE FNO bit.	RW	0

Bits	Field Name	Description	Type	Reset
4	UE	Unrecoverable error. Always reads 0x0. Write 0x0: No effect. Write 0x1: Clears the HCINTERRUPTENABLE UE bit.	RW	0
3	RD	Resume detected. Always reads 0x0. Write 0x0: No effect. Write 0x1: Clears the HCINTERRUPTENABLE RD bit.	RW	0
2	SF	Start of frame. Always reads 0x0. Write 0x0: No effect. Write 0x1: Clears the HCINTERRUPTENABLE SF bit.	RW	0
1	WDH	Write done head. Always reads 0x0. Write 0x0: No effect. Write 0x1: Clears the HCINTERRUPTENABLE WDH bit.	RW	0
0	SO	Scheduling overrun. Always reads 0x0. Write 0x0: No effect. Write 0x1: Clears the HCINTERRUPTENABLE SO bit.	RW	0

Table 24-171. Register Call Summary for Register HCINTERRUPTDISABLE

High-Speed USB Host Subsystem

- High-Speed USB Host Subsystem Register Mapping Summary: [0]

Table 24-172. HCHCCA

Address Offset	0x0000 0018		
Physical Address	0x4806 4418	**Instance**	OHCI
Description	HC HCCA Address Register		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
HCCA																								RESERVED							

Bits	Field Name	Description	Type	Reset
31:8	HCCA	Physical address of the beginning of the HCCA.	RW	0x000000
7:0	RESERVED	Reserved	R	0x00

Table 24-173. Register Call Summary for Register HCHCCA

High-Speed USB Host Subsystem

- High-Speed USB Host Subsystem Register Mapping Summary: [0]

Table 24-174. HCPERIODCURRENTED

Address Offset	0x0000 001C		
Physical Address	0x4806 441C	**Instance**	OHCI
Description	HC Current Periodic Register		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PCED																												RESERVED			

Bits	Field Name	Description	Type	Reset
31:4	PCED	Physical address of current ED on the periodic ED list.	R	0x0000000
3:0	RESERVED	Reserved	R	0x0

Table 24-175. Register Call Summary for Register HCPERIODCURRENTED

High-Speed USB Host Subsystem

- High-Speed USB Host Subsystem Register Mapping Summary: [0]

Table 24-176. HCCONTROLHEADED

Address Offset	0x0000 0020		
Physical Address	0x4806 4420	**Instance**	OHCI
Description	HC Head Control Register		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CHED																												RESERVED			

Bits	Field Name	Description	Type	Reset
31:4	CHED	Physical address of head ED on the control ED list.	RW	0x0000000
3:0	RESERVED	Reserved	R	0x0

Table 24-177. Register Call Summary for Register HCCONTROLHEADED

High-Speed USB Host Subsystem

- High-Speed USB Host Subsystem Register Mapping Summary: [0]

Table 24-178. HCCONTROLCURRENTED

Address Offset	0x0000 0024		
Physical Address	0x4806 4424	**Instance**	OHCI
Description	HC Current Control Register		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CCED																												RESERVED			

Bits	Field Name	Description	Type	Reset
31:4	CCED	Physical address of current ED on the control ED list.	RW	0x0000000
3:0	RESERVED	Reserved	R	0x0

Table 24-179. Register Call Summary for Register HCCONTROLCURRENTED

High-Speed USB Host Subsystem

- High-Speed USB Host Subsystem Register Mapping Summary: [0]

Table 24-180. HCBULKHEADED

Address Offset	0x0000 0028		
Physical Address	0x4806 4428	**Instance**	OHCI
Description	HC Head Bulk Register		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BHED																												RESERVED			

Bits	Field Name	Description	Type	Reset
31:4	BHED	Physical address of head ED on the bulk ED list.	RW	0x0000000
3:0	RESERVED	Reserved	R	0x0

Table 24-181. Register Call Summary for Register HCBULKHEADED

High-Speed USB Host Subsystem

- High-Speed USB Host Subsystem Register Mapping Summary: [0]

Table 24-182. HCBULKCURRENTED

Address Offset	0x0000 002C		
Physical Address	0x4806 442C	**Instance**	OHCI
Description	HC Current Bulk Register		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BCED																												RESERVED			

Bits	Field Name	Description	Type	Reset
31:4	BCED	Physical address of current ED on the bulk ED list.	RW	0x0000000
3:0	RESERVED	Reserved	R	0x0

Table 24-183. Register Call Summary for Register HCBULKCURRENTED

High-Speed USB Host Subsystem

- High-Speed USB Host Subsystem Register Mapping Summary: [0]

Table 24-184. HCDONEHEAD

Address Offset	0x0000 0030		
Physical Address	0x4806 4430	**Instance**	OHCI
Description	HC Head Done Register		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DH																												RESERVED			

Bits	Field Name	Description	Type	Reset
31:4	DH	Physical address of last TD that was added to the Done queue.	R	0x0000000
3:0	RESERVED	Reserved	R	0x0

Table 24-185. Register Call Summary for Register HCDONEHEAD

High-Speed USB Host Subsystem

- High-Speed USB Host Subsystem Register Mapping Summary: [0]
- Register Description: [1]

Table 24-186. HCFMINTERVAL

Address Offset	0x0000 0034		
Physical Address	0x4806 4434	**Instance**	OHCI
Description	HC Frame Interval Register		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIT	FSMPS															RESERVED		FI													

Bits	Field Name	Description	Type	Reset
31	FIT	Frame interval toggle. This bit is toggled whenever it loads a new value to FI.	RW	0
30:16	FSMPS	Largest data packet size for full-speed packets, bit times. This field specifies a value which is loaded into the largest data packet counter at the beginning of each frame.	RW	0x0000
15:14	RESERVED	Reserved	R	0x0
13:0	FI	Frame interval. Number of 12-MHz clocks in the USB frame. The nominal value is set to 11,999, to give a 1-ms frame.	RW	0x2EDF

Table 24-187. Register Call Summary for Register HCFMINTERVAL

High-Speed USB Host Subsystem

- High-Speed USB Host Controller Functionality: [0]
- High-Speed USB Host Subsystem Register Mapping Summary: [1]
- Register Description: [2] [3] [4]

Table 24-188. HCFMREMAINING

Address Offset	0x0000 0038		
Physical Address	0x4806 4438	**Instance**	OHCI
Description	HC Frame Remaining Register		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FRT	RESERVED																	FR													

Bits	Field Name	Description	Type	Reset
31	FRT	Frame remaining toggle. This bit is used for the synchronization between USBHOST.HCFMINTERVAL[13:0] FI and FR. This bit is loaded from the USBHOST.HCFMINTERVAL[31] FIT field whenever FR reaches 0.	R	0
30:14	RESERVED	Reserved	R	0x00000
13:0	FR	Frame remaining.	R	0x0000

Bits	Field Name	Description	Type	Reset
		This counter is decremented at each bit time. When it reaches 0, it is reset by loading the USBHOST.HCFMINTERVAL[13:0] FI value at the next bit time boundary.		

Table 24-189. Register Call Summary for Register HCFMREMAINING

High-Speed USB Host Subsystem

- High-Speed USB Host Subsystem Register Mapping Summary: [0]
- Register Description: [1]

Table 24-190. HCFMNUMBER

Address Offset	0x0000 003C		
Physical Address	0x4806 443C	**Instance**	OHCI
Description	HC Frame Number Register		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																FN															

Bits	Field Name	Description	Type	Reset
31:16	RESERVED	Reserved	R	0x0000
15:0	FN	Frame Number. This is incremented when USBHOST.HCFMREMAINING is reloaded. It is rolled over to 0x0000 after 0xFFFF.	R	0x0000

Table 24-191. Register Call Summary for Register HCFMNUMBER

High-Speed USB Host Subsystem

- High-Speed USB Host Subsystem Register Mapping Summary: [0]

Table 24-192. HCPERIODICSTART

Address Offset	0x0000 0040		
Physical Address	0x4806 4440	**Instance**	OHCI
Description	HC Periodic Start Register		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																PS															

Bits	Field Name	Description	Type	Reset
31:14	RESERVED	Reserved	R	0x00000
13:0	PS	Periodic start. The host controller driver must program this value to be about 10% less than the frame interval field value so that control and bulk EDs have priority for the first 10% of the frame; then periodic EDs have priority for the remaining 90% of the frame.	RW	0x0000

Table 24-193. Register Call Summary for Register HCPERIODICSTART

High-Speed USB Host Subsystem

- High-Speed USB Host Subsystem Register Mapping Summary: [0]

Table 24-194. HCLSTHRESHOLD

Address Offset	0x0000 0044		
Physical Address	0x4806 4444	**Instance**	OHCI
Description	HC Low-Speed Threshold Register		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																				LST											

Bits	Field Name	Description	Type	Reset
31:12	RESERVED	Reserved	R	0x00000
11:0	LST	Low-speed threshold.	RW	0x628

Table 24-195. Register Call Summary for Register HCLSTHRESHOLD

High-Speed USB Host Subsystem

- High-Speed USB Host Subsystem Register Mapping Summary: [0]

Table 24-196. HCRHDESCRIPTORA

Address Offset	0x0000 0048		
Physical Address	0x4806 4448	**Instance**	OHCI
Description	HC Root Hub A Register		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
POTPG								RESERVED											NOCP	OCPM	DT	NPS	PSM	NDP							

Bits	Field Name	Description	Type	Reset
31:24	POTPG	Power-on to power-good time. Defines the minimum amount of time (2 ms * POTPG) between the USB host controller turning on power to a downstream port and when the USB host can access the downstream device.	RW	0x0A
23:13	RESERVED	Reserved	R	0x000
12	NOCP	No overcurrent protection[(1)] 0x0: Overcurrent status is reported collectively for all downstream ports. 0x1: The USB host controller does not implement overcurrent protection inputs.	RW	0x0
11	OCPM	Overcurrent protection mode[(1)]	RW	0x1
10	DT	Device type. Always reads 0x0: Indicates that the USB host controller implemented is not a compound device.	R	0
9	NPS	No power switching 0x0: VBUS power switching is supported, either per-port or all-port switched per the power 0x1: VBUS power switching is not supported, power is available to all downstream ports.	RW	0
8	PSM	Power switching mode. 0x0: Indicates that all ports are powered at the same time 0x1: Individual port power switching is supported	RW	1
7:0	NDP	Number of downstream ports.	R	0x03

(1) This bit must not be used and must always be written to with its default (reset) value, because the hardware overcurrent inputs are tied off at the device level.

Bits	Field Name	Description	Type	Reset
		These bits specify the number of downstream ports supported by the root hub. It is implementation-specific. The minimum number of ports is 1. The maximum number of ports supported by OHCI is 15.		

Table 24-197. Register Call Summary for Register HCRHDESCRIPTORA

High-Speed USB Host Subsystem

- High-Speed USB Host Controller Functionality: [0] [1] [2] [3]
- High-Speed USB Host Subsystem Register Mapping Summary: [4]

Table 24-198. HCRHDESCRIPTORB

Address Offset	0x0000 004C		
Physical Address	0x4806 444C	**Instance**	OHCI
Description	HC Root Hub B Register		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PPCM																DR															

Bits	Field Name	Description	Type	Reset
31:16	PPCM	Port power control mask. Each bit defines whether a corresponding downstream port has port power controlled by the global power control. When set the port's power state is only affected by per-port power control. When cleared the port is controlled by the global power switch. If the device is configured to global switch mode this field is not valid. bit 0: Reserved, bit 1: Ganged-power mask on port 1, ..., bit 15: Ganged-power mask on port 15	RW	0x0000
15:0	DR	Device removable. Each bit defines whether a corresponding downstream port has a removable device. When cleared the attached device is removable. When set the attached device is not removable. Bit 0: Reserved, bit 1 : Device attached to port 1, &, bit 15: Device attached to port 15	RW	0x0000

Table 24-199. Register Call Summary for Register HCRHDESCRIPTORB

High-Speed USB Host Subsystem

- High-Speed USB Host Controller Functionality: [0] [1]
- High-Speed USB Host Subsystem Register Mapping Summary: [2]
- Register Description: [3] [4] [5] [6] [7] [8]

Table 24-200. HCRHSTATUS

Address Offset	0x0000 0050		
Physical Address	0x4806 4450	**Instance**	OHCI
Description	HC Root Hub Status Register		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CRWE	RESERVED														LPSC	DRWE	RESERVED														LPS

Bits	Field Name	Description	Type	Reset
31	CRWE	Clear remote wake-up enable. Write 0x0: No effect. Write 0x1: Clears the device remote wake-up enable bit.	W	0
30:17	RESERVED	Reserved	R	0x0000

Bits	Field Name	Description	Type	Reset
16	LPSC	Local power status change. Always reads 0x0: The Root Hub does not support the local power status feature Write 0x0: No effect. Write 0x1: Sets port power status bits for all ports, if power switching mode is 0. Sets port power status bits for ports with their corresponding port power control mask bits cleared if power switching mode is 1.	RW	0
15	DRWE	Device remote wake-up enable. Enables a connect status change event as a resume event, causing a USB suspend to USB resume state transition and sets the resume detected interrupt status bit. Read 0x1: Connect status change is a remote wake-up event. Read 0x0: Connect status change is not a remote wake-up event. Write 0x0: No effect. Write 0x1: Sets the device remote wake-up enable bit.	RW	0
14:1	RESERVED	Reserved	R	0x0000
0	LPS	Local power status. Always reads 0x0. Write 0x0: No effect. Write 0x1: When in global power mode (power switching mode = 0), turns off power to all ports. If in per-port power mode (power switching mode = 1), turns of power to those ports whose corresponding port power control mask bit is 0.	RW	0

Table 24-201. Register Call Summary for Register HCRHSTATUS

High-Speed USB Host Subsystem

- High-Speed USB Host Subsystem Register Mapping Summary: [0]

Table 24-202. HCRHPORTSTATUS_1

Address Offset	0x0000 0054		
Physical Address	0x4806 4454	**Instance**	OHCI
Description	HC Port 1 Status and Control Register		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED											PRSC	RESERVED	PSSC	PESC	CSC	RESERVED							LSDA_CPP	PPS_SPP	RESERVED			PRS_SPR	RESERVED	PSS_SPS	PES_SPE	CCS_CPE

Bits	Field Name	Description	Type	Reset
31:21	RESERVED	Reserved	R	0x000
20	PRSC	Port 1 reset status change. This bit is set when the Port 1 port reset status bit has changed. Write 0x0: No effect. Write 0x1: Clears this bit.	RW	0
19	RESERVED	Reserved	RW	0
18	PSSC	Port 1 suspend status change. This bit is set when the Port1 port suspend status has changed. Write 0x0: No effect. Write 0x1: Clears this bit.	RW	0
17	PESC	Port 1 enable status change. This bit is set when the Port1 port enable status has changed. Write 0x0: No effect. Write 0x1: Clears this bit.	RW	0

Bits	Field Name	Description	Type	Reset
16	CSC	Port 1 connect status change. This bit is set when the Port1 port current connect status has changed due to a connect or disconnect event. If current connect status is 0 when a set port reset, set port enable, or set port suspend write occurs, this bit is set. Write 0x0: No effect. Write 0x1: Clears this bit. Note: If the DR bit HCRHDESCRIPTORB[1] is set, this bit is set only after a root hub reset to inform the system that the device is attached.	RW	0
15:10	RESERVED	Reserved	R	0x00
9	LSDA_CPP	Port 1 low-speed device attached/clear port power. This bit is valid only when port 1 current connect status is 1. Read 0x0: A full-speed device is attached to port 1. Read 0x1: A low-speed device is attached to port 1. Write 0x0: No effect. Write 0x1: Clears the port 1 port power status.	RW	0
8	PPS_SPP	Port 1 port power status/set port power. Read 0x0: Port 1 power is enabled. Read 0x1: Port 1 power is not enabled. Write 0x0: No effect. Write 0x1: Sets the port 1 port power status bit.	RW	0
7:5	RESERVED	Reserved	R	0x0
4	PRS_SPR	Port 1 port reset status/set port reset. Read 0x0: USB reset is not being sent to port 1. Read 0x1: Port 1 is signaling the USB reset. Write 0x0: No effect. Write 0x1: Sets the port 1 port reset status bit and causes the USB host controller to begin signaling USB reset to port 1.	RW	0
3	RESERVED	Reserved	RW	0
2	PSS_SPS	Port 1 port suspend status/set port suspend. This bit is cleared automatically at the end of the USB resume sequence and also at the end of the USB reset sequence. Write 0x0: No effect. Read 0x0: Port 1 is not in the USB suspend state. Read 0x1: Port 1 is in the USB suspend state or is in the resume sequence. Write 0x1: If port 1 current connect status is 1, sets the port 1 port suspend status bit and places port 1 in USB suspend state. If current connect status is 0, sets instead connect status change to inform the USB host controller driver of an attempt to suspend a disconnected port.	RW	0
1	PES_SPE	Port 1 port enable status/set port enable. This bit is automatically set at completion of port 1 USB reset if it was not already set before the USB reset completed, and is automatically set at the end of a USB suspend if the port was not enabled when the USB resume completed. Read 0x0: Port 1 is not enabled. Read 0x1: Port 1 is enabled. Write 0x0: No effect. Write 0x1: When port 1 current connect status is 1 sets the port 1 port enable status bit. When port 1 current status is 0 has no effect.	RW	0
0	CCS_CPE	Port 1 current connection status/clear port enable. Read 0x0: No USB device is attached to port 1. Read 0x1: Port 1 currently has a USB device attached. Write 0x0: No effect. Write 0x1: Clears the port 1 port enable bit. Note: This bit is set to 1 if the DR bit HCRHDESCRIPTORB[1] is set to indicate a non-removable device on port 1.	RW	0

Table 24-203. Register Call Summary for Register HCRHPORTSTATUS_1

High-Speed USB Host Subsystem

- High-Speed USB Host Controller Functionality: [0]
- High-Speed USB Host Subsystem Register Mapping Summary: [1]

Table 24-204. HCRHPORTSTATUS_2

Address Offset	0x0000 0058		
Physical Address	0x4806 4458	**Instance**	OHCI
Description	HC Port 2 Status and Control Register		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED											PRSC	RESERVED	PSSC	PESC	CSC	RESERVED							LSDA_CPP	PPS_SPP	RESERVED				PRS_SPR	RESERVED	PSS_SPS	PES_SPE	CCS_CPE

Bits	Field Name	Description	Type	Reset
31:21	RESERVED	Reserved	R	0x000
20	PRSC	Port 2 reset status change. This bit is set when the Port 2 port reset status bit has changed. Write 0x0: No effect. Write 0x1: Clears this bit.	RW	0
19	RESERVED	Reserved	RW	0
18	PSSC	Port 2 suspend status changed. This bit is set when the Port 2 port suspend status has changed. Write 0x0: No effect. Write 0x1: Clears this bit.	RW	0
17	PESC	Port 2 enable status change. This bit is set when the Port 2 port enable status has changed. Write 0x0: No effect. Write 0x1: Clears this bit.	RW	0
16	CSC	Port 2 connect status change. This bit is set when the Port 2 port current connect status has changed due to a connect or disconnect event. If current connect status is 0 when a set port reset, set port enable, or set port suspend write occurs, this bit is set. Write 0x0: No effect. Write 0x1: Clears this bit. Note: If the DR bit HCRHDESCRIPTORB[1] is set, this bit is set only after a root hub reset to inform the system that the device is attached.	RW	0
15:10	RESERVED	Reserved	R	0x00
9	LSDA_CPP	Port 2 low-speed device attached/clear port power. This bit is valid only when port 2 current connect status is 1. Read 0x0: A full-speed device is attached to port 2. Read 0x1: A low-speed device is attached to port 2. Write 0x0: No effect. Write 0x1: Clears the port 2 port power status.	RW	0
8	PPS_SPP	Port 2 port power status/set port power. Read 0x0: Port 2 power is enabled. Read 0x1: Port 2 power is not enabled. Write 0x0: No effect. Write 0x1: Sets the port 2 port power status bit.	RW	0
7:5	RESERVED	Reserved	R	0x0
4	PRS_SPR	Port 2 port reset status/set port reset. Read 0x0: USB reset is not being sent to port 2. Read 0x1: Port 2 is signaling the USB reset. Write 0x0: No effect. Write 0x1: Sets the port 2 port reset status bit and causes the USB host controller to begin signaling USB reset to port 2.	RW	0
3	RESERVED	Reserved	RW	0

Bits	Field Name	Description	Type	Reset
2	PSS_SPS	Port 2 port suspend status/set port suspend. This bit is cleared automatically at the end of the USB resume sequence and also at the end of the USB reset sequence. Write 0x0: No effect. Read 0x0: Port 2 is not in the USB suspend state. Read 0x1: Port 2 is in the USB suspend state or is in the resume sequence. Write 0x1: If port 2 current connect status is 1, sets the port 2 port suspend status bit and places port 2 in USB suspend state. If current connect status is 0, sets instead connect status change to inform the USB host controller driver of an attempt to suspend a disconnected port.	RW	0
1	PES_SPE	Port 2 port enable status/set port enable. This bit is automatically set at completion of port 2 USB reset if it was not already set before the USB reset completed, and is automatically set at the end of a USB suspend if the port was not enabled when the USB resume completed. Read 0x0: Port 2 is not enabled. Read 0x1: Port 2 is enabled. Write 0x0: No effect. Write 0x1: When port 2 current connect status is 1 sets the port 2 port enable status bit. When port 2 current status is 0 has no effect.	RW	0
0	CCS_CPE	Port 2 current connection status/clear port enable. Read 0x0: No USB device is attached to port 2. Read 0x1: Port 2 currently has a USB device attached. Write 0x0: No effect. Write 0x1: Clears the port 2 port enable bit. Note: This bit is set to 1 if the DR bit HCRHDESCRIPTORB[1] is set to indicate a non-removable device on port 2.	RW	0

Table 24-205. Register Call Summary for Register HCRHPORTSTATUS_2

High-Speed USB Host Subsystem

- High-Speed USB Host Subsystem Register Mapping Summary: [0]

Table 24-206. HCRHPORTSTATUS_3

Address Offset	0x0000 005C		
Physical Address	0x4806 445C	**Instance**	OHCI
Description	HC Port 3 Status and Control Register		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED											PRSC	RESERVED	PSSC	PESC	CSC	RESERVED							LSDA_CPP	PPS_SPP	RESERVED			PRS_SPR	RESERVED	PSS_SPS	PES_SPE	CCS_CPE

Bits	Field Name	Description	Type	Reset
31:21	RESERVED	Reserved	R	0x000
20	PRSC	Port 3 reset status change. This bit is set when the Port 3 port reset status bit has changed. Write 0x0: No effect. Write 0x1: Clears this bit.	RW	0
19	RESERVED	Reserved	RW	0
18	PSSC	Port 3 suspend status change. This bit is set when the Port 3 port suspend status has changed. Write 0x0: No effect. Write 0x1: Clears this bit.	RW	0
17	PESC	Port 3 enable status change. This bit is set when the Port 3 port enable status has changed. Write 0x0: No effect. Write 0x1: Clears this bit.	RW	0

Bits	Field Name	Description	Type	Reset
16	CSC	Port 3 connect status change. This bit is set when the Port 3 port current connect status has changed due to a connect or disconnect event. If current connect status is 0 when a set port reset, set port enable, or set port suspend write occurs, this bit is set. Write 0x0: No effect. Write 0x1: Clears this bit. Note: If the DR bit HCRHDESCRIPTORB[1] is set, this bit is set only after a root hub reset to inform the system that the device is attached.	RW	0
15:10	RESERVED	Reserved	R	0x00
9	LSDA_CPP	Port 3 low-speed device attached/clear port power. This bit is valid only when port 3 current connect status is 1. Read 0x0: A full-speed device is attached to port 3. Read 0x1: A low-speed device is attached to port 3. Write 0x0: No effect. Write 0x1: Clears the port 3 port power status.	RW	0
8	PPS_SPP	Port 3 power status/set port power. Read 0x0: Port 3 power is enabled. Read 0x1: Port 3 power is not enabled. Write 0x0: No effect. Write 0x1: Sets the port 3 port power status bit.	RW	0
7:5	RESERVED	Reserved	R	0x0
4	PRS_SPR	Port 3 reset status/set port reset. Read 0x0: USB reset is not being sent to port 3. Read 0x1: Port 3 is signaling the USB reset. Write 0x0: No effect. Write 0x1: Sets the port 3 port reset status bit and causes the USB host controller to begin signaling USB reset to port 3.	RW	0
3	RESERVED	Reserved	RW	0
2	PSS_SPS	Port 3 port suspend status/set port suspend. This bit is cleared automatically at the end of the USB resume sequence and also at the end of the USB reset sequence. Write 0x0: No effect. Read 0x0: Port 3 is not in the USB suspend state. Read 0x1: Port 3 is in the USB suspend state or is in the resume sequence. Write 0x1: If port 3 current connect status is 1, sets the port 3 port suspend status bit and places port 3 in USB suspend state. If current connect status is 0, sets instead connect status change to inform the USB host controller driver of an attempt to suspend a disconnected port.	RW	0
1	PES_SPE	Port 3 enable status/set port enable. This bit is automatically set at completion of port 3 USB reset if it was not already set before the USB reset completed, and is automatically set at the end of a USB suspend if the port was not enabled when the USB resume completed. Read 0x0: Port 3 is not enabled. Read 0x1: Port 3 is enabled. Write 0x0: No effect. Write 0x1: When port 3 current connect status is 1 sets the port 3 port enable status bit. When port 3 current status is 0 has no effect.	RW	0
0	CCS_CPE	Port 3 current connection status/clear port enable. Read 0x0: No USB device is attached to port 3. Read 0x1: Port 3 currently has a USB device attached. Write 0x0: No effect. Write 0x1: Clears the port 3 port enable bit. Note: This bit is set to 1 if the DR bit HCRHDESCRIPTORB[1] is set to indicate a non-removable device on port 3.	RW	0

Table 24-207. Register Call Summary for Register HCRHPORTSTATUS_3

High-Speed USB Host Subsystem

- High-Speed USB Host Subsystem Register Mapping Summary: [0]

24.2.6.4.4 EHCI Registers

Table 24-208. HCCAPBASE

Address Offset	0x0000 0000		
Physical Address	0x4806 4800	**Instance**	EHCI
Description	Host Controller Capability register		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
HCIVERSION																RESERVED								CAPLENGTH							

Bits	Field Name	Description	Type	Reset
31:16	HCIVERSION	Interface version number It contains a BCD encoding of the EHCI revision number supported by this host controller. [7:4] Major revision [3:0] Minor revision	R	0x0100
15:8	RESERVED	Reserved	R	0x00
7:0	CAPLENGTH	Capability register length	R	0x10

Table 24-209. Register Call Summary for Register HCCAPBASE

High-Speed USB Host Subsystem

- High-Speed USB Host Subsystem Register Mapping Summary: [0]

Table 24-210. HCSPARAMS

Address Offset	0x0000 0004		
Physical Address	0x4806 4804	**Instance**	EHCI
Description	Host Controller Structural Parameters		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED												RESERVED			P_INDICATOR	N_CC				N_PCC			PRR	RESERVED		PPC	N_PORTS			

Bits	Field Name	Description	Type	Reset
31:20	RESERVED	Reserved	R	0x000
19:17	RESERVED	Reserved	R	0x0
16	P_INDICATOR	Port indicator support indication This bit indicates whether the ports support port indicator control. 0x1: The port status and control registers include a read/write field for controlling the state of the port indicator.	R	0
15:12	N_CC	Number of Companion Controllers This field indicates the number of companion controllers associated with this USB 2.0 host controller. 0x0: There are no companion host controllers. Port-ownership hand-off is not supported. Only high-speed devices are supported on the host controller root ports. Others: there are companion USB 1.1 host controller(s). Port-ownership hand-off is supported. High-, full-, and low-speed devices are supported on the host controller root ports.	R	0x1
11:8	N_PCC	Number of Ports per Companion Controller	R	0x3

Bits	Field Name	Description	Type	Reset
		This field indicates the number of ports supported per companion host controller. It is used to indicate the port routing configuration to system software. For example, if N_PORTS has a value of 6 and N_CC has a value of 2, then N_PCC can have a value of 3. The convention is that the first N_PCC ports are assumed to be routed to companion controller 1, the next N_PCC ports to companion controller 2, etc. The number in this field must be consistent with N_PORTS and N_CC.		
7	PRR	Port Routing Rules The first N_PCC ports are routed to the lowest-numbered function companion host controller, the next N_PCC ports are routed to the next lowest-function companion controller, and so on.	R	0
6:5	RESERVED	Reserved	R	0x0
4	PPC	Port Power control This field indicates whether the host controller implementation includes port power control. 0x0: The ports do not have port power switches. 0x1: The ports have port power switches.	R	1
3:0	N_PORTS	Number of downstream ports This field specifies the number of physical downstream ports implemented on this host controller.	R	0x3

Table 24-211. Register Call Summary for Register HCSPARAMS

High-Speed USB Host Subsystem

- High-Speed USB Host Subsystem Register Mapping Summary: [0]
- Register Description: [1] [2] [3]

Table 24-212. HCCPARAMS

Address Offset	0x0000 0008		
Physical Address	0x4806 4808	**Instance**	EHCI
Description	Host Controller Capability Parameters		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																EECP								IST				RESERVED	ASPC	PFLF	BIT64AC

Bits	Field Name	Description	Type	Reset
31:16	RESERVED	Reserved	R	0x0000
15:8	EECP	EHCI Extended Capabilities Pointer This field indicates the existence of a capabilities list. 0x0: No extended capabilities are implemented. Others: The offset in PCI configuration space of the first EHCI extended capability.	R	0x00
7:4	IST	Isochronous Scheduling Threshold This field indicateswhere software can reliably update the isochronous schedule in relation to the current position of the executing host controller. The host controller can hold 1 microframe of isochronous data structures before flushing the state.	R	0x1

Bits	Field Name	Description	Type	Reset
3	RESERVED	Reserved	R	0
2	ASPC	Asynchronous Schedule Park Capability 0x1: The host controller supports the park feature for high-speed queue heads in the asynchronous schedule. The feature can be disabled or enabled and set to a specific level by using the USBHOST.USBCMD[11]ASPME and the USBHOST.USBCMD[9:8] ASPMC fields.	R	1
1	PFLF	Programmable Frame List Flag 0x0: System software must use a frame list length of 1024 elements with this host controller. 0x1: System software can specify and use a smaller frame list and configure the host controller via the USBHOST.USBCMD[3:2] FLS field. The frame list must always be aligned on a 4K-page boundary.	R	1
0	BIT64AC	64-bit addressing capability This field documents the addressing range capability of this implementation. 0x0: Data structures using 32-bit address memory pointers 0x1: Data structures using 64-bit address memory pointers	R	0

Table 24-213. Register Call Summary for Register HCCPARAMS

High-Speed USB Host Subsystem

- High-Speed USB Host Subsystem Register Mapping Summary: [0]
- Register Description: [1]

Table 24-214. USBCMD

Address Offset	0x0000 0010		
Physical Address	0x4806 4810	**Instance**	EHCI
Description	USB Command		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								ITC								RESERVED				ASPME	RESERVED	ASPMC		LHCR	IAAD	ASE	PSE	FLS		HCR	RS

Bits	Field Name	Description	Type	Reset
31:24	RESERVED	Reserved	R	0x00
23:16	ITC	Interrupt Threshold Control This field is used by the system software to select the maximum rate at which the host controller issues interrupts. The only valid values are defined below. If software writes an invalid value to this register, the results are undefined. 0x00: Reserved 0x01: 1 microframe 0x02: 2 microframes 0x04: 4 microframes 0x08: 8 microframes (default, equates to 1 ms) 0x10: 16 microframes (2 ms) 0x20: 32 microframes (4 ms) 0x40: 64 microframes (8 ms) Others: Undefined	RW	0x08

Bits	Field Name	Description	Type	Reset
15:12	RESERVED	Reserved	R	0x0
11	ASPME	Asynchronous Schedule Park Mode Enable 0x0: Park mode is disabled. 0x1: Park mode is enabled.	RW	1
10	RESERVED	Reserved	R	0
9:8	ASPMC	Asynchronous Schedule Park Mode Count It contains a count of the number of successive transactions the host controller is allowed to execute from a high-speed queue head on the asynchronous schedule before continuing traversal of the asynchronous schedule. Valid values are 0x1 to 0x3. The software must not write a 0 to this bit when Park Mode Enable is a 1 as this may result in undefined behavior.	RW	0x3
7	LHCR	Light Host Controller Reset It allows the driver to reset the EHCI controller without affecting the state of the ports or the relationship to the companion host controllers. Read 0x0: Light host controller reset is complete and it is safe for host software to reinitialize the host controller. Read 0x1: Light host controller reset is still ongoing.	RW	0
6	IAAD	Interrupt on Async Advance Doorbell This bit is used as a doorbell by software to tell the host controller to issue an interrupt the next time it advances asynchronous schedule. Write 0x1: Ring the doorbell. Software should not write a 1 to this bit when the asynchronous schedule is disabled. Doing so may yield undefined results.	RW	0
5	ASE	Asynchronous Schedule Enable This bit controls whether the host controller skips processing the asynchronous schedule. 0x0: Do not process the asynchronous schedule 0x1: Use the USBHOST.ASYNCLISTADDR register to access the asynchronous schedule.	RW	0
4	PSE	Periodic Schedule Enable This bit controls whether the host controller skips processing the periodic schedule. 0x0: Do not process the periodic schedule 0x1: Use the USBHOST.PERIODICLISTBASE register to access the periodic schedule.	RW	0
3:2	FLS	Frame List Size This field specifies the size of the frame list. The size of the frame list controls which bits in the frame index register should be used for the frame list vurrent index. 0x0: 1024 elements (4096 bytes) 0x1: 512 elements (2048 bytes) 0x2: 256 elements (1024 bytes), for resource-constrained environments 0x3: Reserved	RW	0
1	HCR	Host Controller Reset This control bit is used by software to reset the host controller. Write 0x1: Reset the host controller, the PCI configuration registers are not affected by this reset and all operational registers are set to their initial values. This bit is set to 0 by the host controller when the reset process is complete.	W	0
0	RS	Run/stop	RW	0

Bits	Field Name	Description	Type	Reset
		0x1: Run, the host controller proceeds with execution of the schedule. The host controller continues execution as long as this bit is set to 1. 0x0: Stop, the host controller completes the current and any actively pipelined transactions on the USB and then halts.		

Table 24-215. Register Call Summary for Register USBCMD

High-Speed USB Host Subsystem

- High-Speed USB Host Subsystem Register Mapping Summary: [0]
- Register Description: [1] [2] [3] [4] [5]

Table 24-216. USBSTS

Address Offset	0x0000 0014		
Physical Address	0x4806 4814	**Instance**	EHCI
Description	USB status		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																ASS	PSS	REC	HCH	RESERVED				IAA	HSE	FLR	PCD	USBEI	USBI		

Bits	Field Name	Description	Type	Reset
31:16	RESERVED	Reserved	R	0x0000
15	ASS	Asynchronous Schedule Status The bit reports the current real status of the asynchronous schedule. 0x0: The status of the asynchronous schedule is disabled. 0x1: The status of the asynchronous schedule is enabled.	R	0
14	PSS	Periodic Schedule Status The bit reports the current real status of the periodic schedule. 0x0: The status of the periodic schedule is disabled. 0x1: The status of the periodic schedule is enabled.	R	0
13	REC	Reclamation It is used to detect an empty asynchronous schedule.	R	0
12	HCH	Host Controller Halted This bit is a 0 whenever the USBHOST.USBCMD[0] RS bit is a 1. The host controller sets this bit to 1 after it has stopped executing as a result of the RS bit being set to 0, either by software or by the host controller hardware.	R	1
11:6	RESERVED	Reserved	R	0x00
5	IAA	Interrupt on Async Advance System software can force the host controller to issue an interrupt the next time the host controller advances the asynchronous schedule by writing a 1 in the USBHOST.USBCMD[6] IAAD bit. This status bit indicates the assertion of that interrupt source.	RW	0
4	HSE	Host System Error The host controller sets this bit to 1 when a serious error occurs during a host system access involving the host controller module.	RW	0
3	FLR	Frame List Rollover The host controller sets this bit to 1 when the USBHOST.FRINDEX rolls over from its maximum value to 0. The exact value at which the rollover occurs depends on the frame list size.	RW	0
2	PCD	Port Change Detect	RW	0

Bits	Field Name	Description	Type	Reset
		The host controller sets this bit to 1 when any port for which the USBHOST.PORTSC_i[13] PO bit is set to 0 has a change bit transition from a 0 to a 1 or a USBHOST.PORTSC_i[6] FPR bit transition from a 0 to a 1. This bit is also set as a result of the USBHOST.PORTSC_i[1] CSC bit being set to 1 after system software has relinquished ownership of a connected port by writing a 1 to a USBHOST.PORTSC_i[13] PO bit.		
1	USBEI	USB Error Interrupt The host controller sets this bit to 1 when completion of a USB transaction results in an error condition.	RW	0
0	USBI	USB Interrupt The host controller sets this bit to 1 on completion of a USB transaction, which results in the retirement of a transfer descriptor that had its IOC bit set. The host controller also sets this bit to 1 when a short packet is detected (actual number of bytes received was less than the expected number of bytes).	RW	0

Table 24-217. Register Call Summary for Register USBSTS

High-Speed USB Host Subsystem

- High-Speed USB Host Subsystem Register Mapping Summary: [0]
- Register Description: [1] [2] [3] [4] [5] [6] [7] [8] [9] [10] [11] [12] [13]

Table 24-218. USBINTR

Address Offset	0x0000 0018		
Physical Address	0x4806 4818	**Instance**	EHCI
Description	USB interrupt enable		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																								IAAE	HSEE	FLRE	PCIE	USBEIE	USBIE		

Bits	Field Name	Description	Type	Reset
31:6	RESERVED	Reserved	R	0x0000000
5	IAAE	Interrupt on Async Advance Enable 0x1: When the USBHOST.USBSTS[5] IAA bit is 1, the host controller issues an interrupt at the next interrupt threshold. The interrupt is acknowledged by software clearing the USBHOST.USBSTS[5] IAA bit.	RW	0
4	HSEE	Host System Error Enable 0x1: When the USBHOST.USBSTS[4] HSE bit is 1, the host controller issues an interrupt. The interrupt is acknowledged by software clearing the USBHOST.USBSTS[4] HSE bit.	RW	0
3	FLRE	Frame List Rollover Enable 0x1: When the USBHOST.USBSTS[3] FLR bit is 1, the host controller issues an interrupt. The interrupt is acknowledged by software clearing the USBHOST.USBSTS[3] FLR bit.	RW	0
2	PCIE	Port Change Interrupt Enable 0x1: When the USBHOST.USBSTS[2] PCD bit is 1, the host controller issues an interrupt. The interrupt is acknowledged by software clearing the USBHOST.USBSTS[3] FLR bit.	RW	0
1	USBEIE	USB Error Interrupt Enable	RW	0

Bits	Field Name	Description	Type	Reset
		0x1: When the USBHOST.USBSTS[1] USBEI bit is 1, the host controller issues an interrupt at the next interrupt threshold. The interrupt is acknowledged by software clearing the USBHOST.USBSTS[1] USBEI bit.		
0	USBIE	USB Interrupt Enable 0x1: When the USBHOST.USBSTS[0] USBI bit is 1, the host controller issues an interrupt at the next interrupt threshold. The interrupt is acknowledged by software clearing the USBHOST.USBSTS[0] USBI bit.	RW	0

Table 24-219. Register Call Summary for Register USBINTR

High-Speed USB Host Subsystem

- High-Speed USB Host Subsystem Register Mapping Summary: [0]

Table 24-220. FRINDEX

Address Offset	0x0000 001C		
Physical Address	0x4806 481C	**Instance**	EHCI
Description	USB frame index		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																								FI							

Bits	Field Name	Description	Type	Reset
31:14	RESERVED	Reserved	R	0x00000
13:0	FI	Frame index The value in this register is incremented at the end of each time frame.	RW	0x0000

Table 24-221. Register Call Summary for Register FRINDEX

High-Speed USB Host Subsystem

- High-Speed USB Host Subsystem Register Mapping Summary: [0]
- Register Description: [1]

Table 24-222. CTRLDSSEGMENT

Address Offset	0x0000 0020		
Physical Address	0x4806 4820	**Instance**	EHCI
Description	4G segment selector		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CDSS																															

Bits	Field Name	Description	Type	Reset
31:0	CDSS	This 32-bit register corresponds to the most significant address bits [63:32] for all EHCI data structures.	R	0x00000000

Table 24-223. Register Call Summary for Register CTRLDSSEGMENT

High-Speed USB Host Subsystem

- High-Speed USB Host Subsystem Register Mapping Summary: [0]

Table 24-224. PERIODICLISTBASE

Address Offset	0x0000 0024		
Physical Address	0x4806 4824	**Instance**	EHCI
Description	Frame list base address		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BAL																				RESERVED											

Bits	Field Name	Description	Type	Reset
31:12	BAL	Base address (low) These bits correspond to memory address signals.	RW	0x00000
11:0	RESERVED	Reserved	R	0x000

Table 24-225. Register Call Summary for Register PERIODICLISTBASE

High-Speed USB Host Subsystem

- High-Speed USB Host Subsystem Register Mapping Summary: [0]
- Register Description: [1]

Table 24-226. ASYNCLISTADDR

Address Offset	0x0000 0028		
Physical Address	0x4806 4828	**Instance**	EHCI
Description	Next asynchronous list address		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
LPL																											RESERVED			

Bits	Field Name	Description	Type	Reset
31:5	LPL	Link Pointer Low It contains the address of the next asynchronous queue head to be executed.	RW	0x0000000
4:0	RESERVED	Reserved	R	0x00

Table 24-227. Register Call Summary for Register ASYNCLISTADDR

High-Speed USB Host Subsystem

- High-Speed USB Host Subsystem Register Mapping Summary: [0]
- Register Description: [1]

Table 24-228. CONFIGFLAG

Address Offset	0x0000 0050		
Physical Address	0x4806 4850	**Instance**	EHCI
Description	Configured flag register		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																															CF

Bits	Field Name	Description	Type	Reset
31:1	RESERVED	Reserved	R	0x00000000
0	CF	Configure Flag This bit controls the default port-routing control logic. 0x0: Port routing control logic default-routes each port to an implementation dependent classic host controller. 0x1: Port routing control logic default-routes all ports to this host controller.	RW	0

Table 24-229. Register Call Summary for Register CONFIGFLAG

High-Speed USB Host Subsystem

- High-Speed USB Host Subsystem Register Mapping Summary: [0]
- Register Description: [1] [2]

Table 24-230. PORTSC_i

Address Offset	0x0000 0054 + (0x04 * i)	**Index**	i = 0 to 2
Physical Address	0x4806 4854 + (0x04 * i)	**Instance**	EHCI
Description	Port Status/Control		
Type	RW		

31 30 29 28 27 26 25 24	23 22	21	20	19 18 17 16	15 14	13	12	11 10	9	8	7	6	5 4	3	2	1	0
RESERVED		WDE	WCE	PTC	PIC	PO	PP	LS	RESERVED	PR	SUS	FPR	RESERVED	PEDC	PED	CSC	CCS

Bits	Field Name	Description	Type	Reset
31:22	RESERVED	Reserved	R	0x000
21	WDE	Wake on Disconnect Enable This field is 0 if the PP bit is 0. Write 0x1: Enables the port to be sensitive to device disconnects as wake-up events.	RW	0
20	WCE	Wake on Connect Enable This field is 0 if the PP bit is 0. Write 0x1: Enables the port to be sensitive to device connects as wake-up events.	RW	0
19:16	PTC	Port Test Control The port is operating in specific test modes as indicated by the specific value. The encoding of the test mode bits are: 0x0: Test mode not enabled 0x1: Test J_STATE 0x2: Test K_STATE 0x3: Test SE0_NAK 0x4: Test Packet 0x5: Test FORCE_ENABLE Others: Reserved	RW	0x0
15:14	PIC	Port Indicator Control (not implemented)	R	0x0
13	PO	Port Owner This bit unconditionally goes to a 0x0 when the USBHOST.CONFIGFLAG[0] CF bit makes a 0 to 1 transition. This bit unconditionally goes to 0 whenever the USBHOST.CONFIGFLAG[0] CF bit is 0. 0x1: A companion host controller owns and controls the port.	RW	1

Bits	Field Name	Description	Type	Reset
12	PP	Port Power The function of this bit depends on the value of the USBHOST.HCSPARAMS[4] PPC bit. The behavior is as follows: **PPC** / **PP** / **Operation** 0x0 / 0x1 / Host controller does not have port power. control switches. Each port is hard-wired to power. 0x1 / N/A / Host controller has port power control switches. This bit represents the current setting of the switch (0 = Off, 1 = On). When an overcurrent condition is detected on a powered port and the USBHOST.HCSPARAMS[4] PPC bit is a 1, the PP bit in each affected port may be transitioned by the host controller from 1 to 0.	RW	0
11:10	LS	Line Status These bits reflect the current logical levels of the D+ (bit 11) and D- (bit 10) signal lines. This field is valid only when the port enable bit is 0 and the current connect status bit is set to 1. The encoding of the bits is: **Bits[11:10]** / **USB State** / **Interpretation** 0x0 / SE0 / Not low-speed device, perform EHCI reset. 0x2 / J-state / Not low-speed device, perform EHCI reset. 0x1 / K-state / Low-speed device, release ownership of port. 0x3 / Undefined / Not low-speed device, perform EHCI reset.	R	0x0
9	RESERVED	Reserved	R	0
8	PR	Port Reset This field is 0 if the PP bit is 0. 0x0: Port is not in reset. 0x1: Port is in reset. Write 0x0: Terminate the bus reset sequence. Write 0x1 when at 0x0: The bus reset sequence is started.	RW	0
7	SUS	Suspend This field is 0 if the PP bit is 0. 0x0 when PED = 0x1: Port enabled 0x1 when PED = 0x1: Port in suspend state When PED = 0x0: Port disabled	RW	0
6	FPR	Force Port Resume This field is 0 if the PP bit is 0. 0x0: No resume (K-state) detected/driven on port 0x1: Resume detected/driven on port	RW	0
5:4	RESERVED	Reserved	RW	0
3	PEDC	Port Enabled/Disabled Change This field is 0 if the PP bit is 0. Read 0x0: No change. Read 0x1: Port enabled/disabled status has changed. Write 0x1: Clears this bit to 0.	RW	0
2	PED	Port Enabled/Disabled Software cannot enable a port by writing a 1 to this field. The host controller only sets this to 1 when the reset sequence determines that the attached device is a high-speed device. Ports can be disabled by either a fault condition (disconnect event or other fault condition) or by host software. This field is 0 if the PP bit is 0.	RW	0

Bits	Field Name	Description	Type	Reset
		0x0: Disable		
		0x1: Enable		
1	CSC	Connect Status Change	RW	0
		Indicates a change has occurred in the port CCS bit.		
		This field is 0 if the PP bit is 0.		
		Read 0x0: No change		
		Read 0x1: Change in current connect status		
		Write 0x1: Clears this bit to 0		
0	CCS	Current Connect Status	R	0
		This value reflects the current state of the port, and may not correspond directly to the event that caused the CSC bit to be set.		
		This field is 0 if the PP bit is 0.		
		0x0: No device is present.		
		0x1: Device is present on port.		

Table 24-231. Register Call Summary for Register PORTSC_i

High-Speed USB Host Subsystem

- High-Speed USB Host Subsystem Register Mapping Summary: [0]
- Register Description: [1] [2] [3] [4]

Table 24-232. INSNREG00

Address Offset	0x0000 0090		
Physical Address	0x4806 4890	**Instance**	EHCI
Description	Implementation-specific register #0		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																UFRAME_CNT															EN

Bits	Field Name	Description	Type	Reset
31:14	RESERVED	Reserved	R	0x00000
13:1	UFRAME_CNT	1-microframe length value, to reduce simulation time SIMULATIONS ONLY, NOT AN ACTUAL REGISTER	RW	0x0000
0	EN	Enable of this register	RW	0

Table 24-233. Register Call Summary for Register INSNREG00

High-Speed USB Host Subsystem

- High-Speed USB Host Subsystem Register Mapping Summary: [0]

Table 24-234. INSNREG01

Address Offset	0x0000 0094		
Physical Address	0x4806 4894	**Instance**	EHCI
Description	Implementation-specific register #1		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
OUT_THRESHOLD																IN_THRESHOLD															

Bits	Field Name	Description	Type	Reset
31:16	OUT_THRESHOLD	Programmable output packet buffer threshold, in 32-bit words	RW	0x0020
15:0	IN_THRESHOLD	Programmable input packet buffer threshold, in 32-bit words	RW	0x0020

Table 24-235. Register Call Summary for Register INSNREG01

High-Speed USB Host Subsystem

- High-Speed USB Host Subsystem Register Mapping Summary: [0]
- Register Description: [1]

Table 24-236. INSNREG02

Address Offset	0x0000 0098		
Physical Address	0x4806 4898	**Instance**	EHCI
Description	Implementation-specific register #2		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																				BUF_DEPTH											

Bits	Field Name	Description	Type	Reset
31:12	RESERVED	Reserved	R	0x00000
11:0	BUF_DEPTH	Programmable packet buffer depth, in 32-bit words	RW	0x080

Table 24-237. Register Call Summary for Register INSNREG02

High-Speed USB Host Subsystem

- High-Speed USB Host Subsystem Register Mapping Summary: [0]

Table 24-238. INSNREG03

Address Offset	0x0000 009C		
Physical Address	0x4806 489C	**Instance**	EHCI
Description	Implementation-specific register #3		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																															BRK_MEM_TRSF

Bits	Field Name	Description	Type	Reset
31:1	RESERVED	Reserved	R	0x00000000
0	BRK_MEM_TRSF	Break Memory Transfer, with INSNREG01 0x0: Disabled 0x1: Enabled	RW	0x1

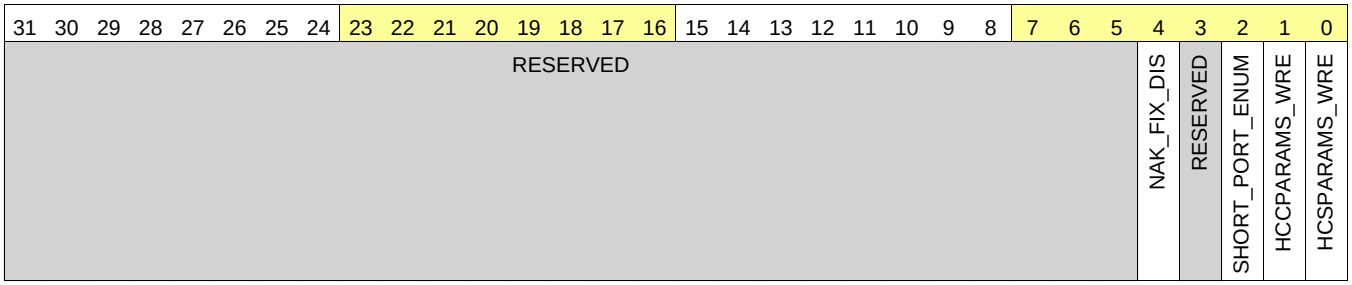
Table 24-239. Register Call Summary for Register INSNREG03

High-Speed USB Host Subsystem

- High-Speed USB Host Subsystem Register Mapping Summary: [0]

Table 24-240. INSNREG04

Address Offset	0x0000 00A0		
Physical Address	0x4806 48A0	**Instance**	EHCI
Description	Implementation-specific register #4		
Type	RW		



Bits	Field Name	Description	Type	Reset
31:5	RESERVED	Reserved	R	0x0000000
4	NAK_FIX_DIS	Disable NAK fix (don't touch)	RW	0
3	RESERVED	Reserved	R	0
2	SHORT_PORT_ENUM	Scale down Port enumeration time (debug)	RW	0
1	HCCPARAMS_WRE	Make read-only HCCPARAMS register writable (debug)	RW	0
0	HCSPARAMS_WRE	Make read-only HCSPARAMS register writable (debug)	RW	0

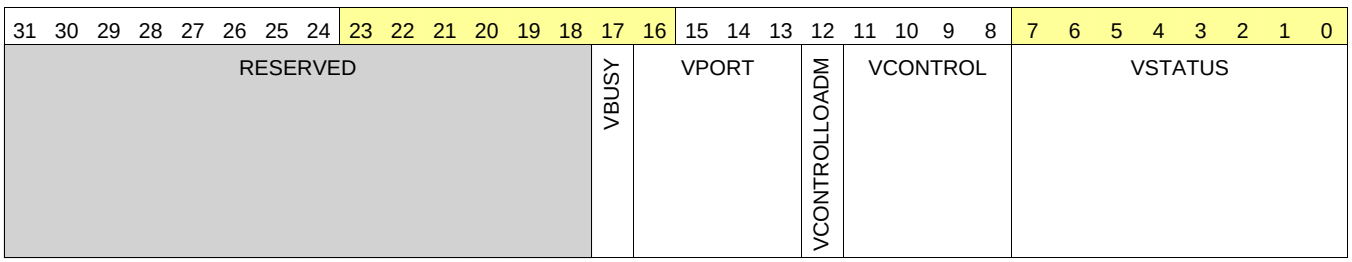
Table 24-241. Register Call Summary for Register INSNREG04

High-Speed USB Host Subsystem

- High-Speed USB Host Subsystem Register Mapping Summary: [0]

Table 24-242. INSNREG05_UTMI

Address Offset	0x0000 00A4		
Physical Address	0x4806 48A4	**Instance**	EHCI
Description	Implementation-specific register #5 Register functionality for UTMI mode		
Type	RW		



Bits	Field Name	Description	Type	Reset
31:18	RESERVED	Reserved	R	0x0000
17	VBUSY	0x0: Vendor interface is done/inactive 0x1: Vendor interface is busy	R	0
16:13	VPORT	Vendor interface port selection 0x1: Port 1 vendor interface selected 0x2: Port 2 vendor interface selected 0x3: Port 3 vendor interface selected	RW	0x0
12	VCONTROLLOADM	UTMI VcontrolLoadM output (active-low) 0x0: Load Vcontrol value into PHY 0x1: No Action	RW	0
11:8	VCONTROL	UTMI Vcontrol output, to be loaded into the PHY	RW	0x0
7:0	VSTATUS	UTMI Vstatus input image, from PHY	R	0x00

Table 24-243. Register Call Summary for Register INSNREG05_UTMI

High-Speed USB Host Subsystem

- High-Speed USB Host Subsystem Register Mapping Summary: [0]

Table 24-244. INSNREG05_ULPI

Address Offset	0x0000 00A4		
Physical Address	0x4806 48A4	**Instance**	EHCI
Description	Implementation-specific register #5 Register functionality for ULPI mode		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CONTROL	RESERVED			PORTSEL				OPSEL	REGADD							EXTREGADD								WRDATA							

Bits	Field Name	Description	Type	Reset
31	CONTROL	Control/status of the ULPI register access 0x0: ULPI access done 0x1: Start ULPI access	RW	0
30:28	RESERVED	Reserved	R	0x0
27:24	PORTSEL	0x1: Port 1 selected for register access 0x2: Port 2 selected for register access 0x3: Port 3 selected for register access	RW	0x0
23:22	OPSEL	0x2: Register access is Write 0x3: Register access is Read	RW	0x0
21:16	REGADD	ULPI direct register address, for any value different than 0x2F. 0x2F: Triggers an extended address	RW	0x00
15:8	EXTREGADD	Address for extended register accesses. Don't care for direct accesses.	RW	0x00
7:0	WRDATA	Read/Write data of register access	RW	0x00

Table 24-245. Register Call Summary for Register INSNREG05_ULPI

High-Speed USB Host Subsystem

- High-Speed USB Host Subsystem Register Mapping Summary: [0]

General-Purpose Interface

This chapter describes the general-purpose interface for the OMAP34xx multimedia device.

Topic **Page**

25.1 General-Purpose Interface Overview 3326
25.2 General-Purpose Interface Environment 3328
25.3 General-Purpose Interface Integration 3331
25.4 General-Purpose Interface Functional Description 3338
25.5 General-Purpose Interface Basic Programming Model 3342
25.6 General-Purpose Interface Register Manual 3348

25.1 General-Purpose Interface Overview

The general-purpose interface combines six general-purpose input/output (GPIO) banks.

Each GPIO module provides 32 dedicated general-purpose pins with input and output capabilities; thus, the general-purpose interface supports up to 192 (6 x 32) pins.

These pins can be configured for the following applications:

- Data input (capture)/output (drive)
- Keyboard interface with a debounce cell
- Interrupt generation in active mode upon the detection of external events. Detected events are processed by two parallel independent interrupt-generation submodules to support biprocessor operations.
- Wake-up request generation in idle mode upon the detection of external events

These modules do not include pad control (pull up/down control, open-drain feature). For more information, see Chapter 7, *System Control Module*.

25.1.1 Global Features

The GPIO modules include the following global features:

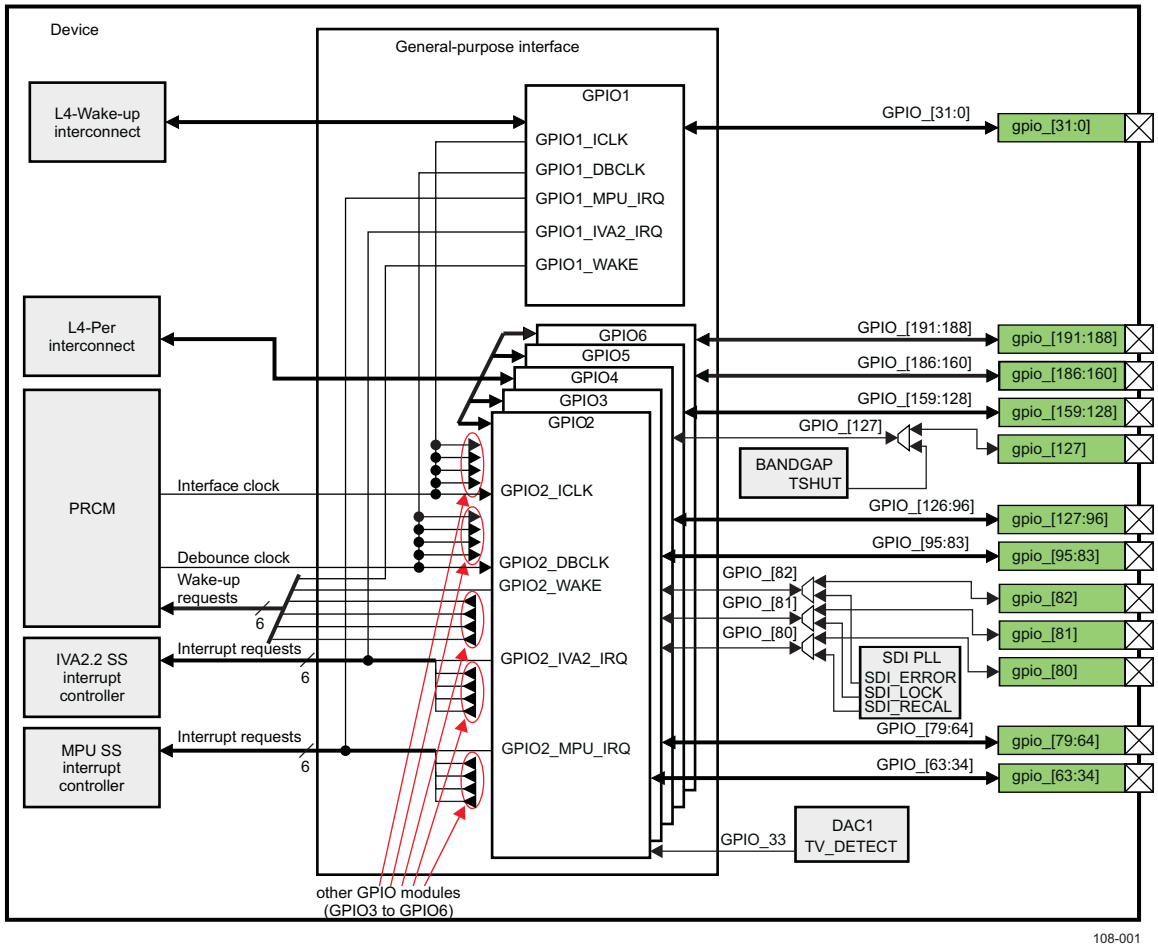
- Synchronous interrupt requests in active mode from each channel are processed by two identical interrupt generation submodules used independently by the imaging video and audio accelerator (IVA2.2) and the microprocessor unit (MPU) subsystems. One of these interrupts is mapped on the IVA2.2 subsystem interrupt controller and the other on the MPU subsystem interrupt controller.
- Asynchronous wake-up requests in idle mode from input channels are merged together to issue one wake-up signal per GPIO module.
- Data input (capture)/output (drive)
- Power management support

The general-purpose interface has 12 interrupt lines (two interrupt lines per GPIO module instance).

Each GPIO module produces a wake-up request signal to the power, reset, and clock management (PRCM) module.

Figure 25-1 shows an overview of the general-purpose interface.

Figure 25-1. General-Purpose Interface Overview



Each channel in the GPIO modules has the following features:

- The GPIOi.GPIO_OE register controls the output capability for each pin.
- The output line level reflects the value written in the GPIOi.GPIO_DATAOUT register through the L4 interconnect.
- The input line can be fed to the GPIO module through an optional and configurable debounce cell. (The debouncing time value is global for all ports of one GPIO module, so up to five different debouncing time values are possible.)
- The input line value is sampled into the GPIOi.GPIO_DATAIN register and can be read through the L4 interconnect.
- In active mode, the input line can be used through level and edge detectors to trigger synchronous interrupts. The edge (rising, falling, or both) or the level (logical 0, logical 1, or both) used can be configured.
- In idle mode, the input line can be used to activate the asynchronous wake-up request (on edge detection: rising edge, falling edge, or both).

The module provides an alternative to the atomic test and set operations for the following registers:

- GPIOi.GPIO_DATAOUT
- GPIOi.GPIO_IRQENABLE1
- GPIOi.GPIO_IRQENABLE2
- GPIOi.GPIO_WAKEUPENABLE

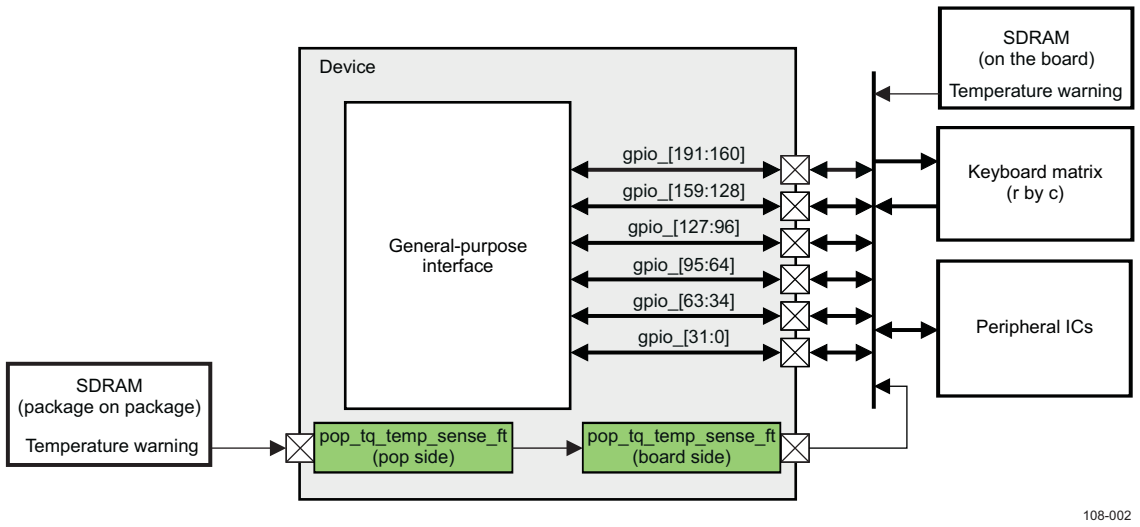
For these registers, the modules implement the set-and-clear protocol register update (see Section 25.5.2, *Set-and-Clear Instructions*).

25.2 General-Purpose Interface Environment

The general-purpose interface combines six GPIO modules for a flexible, user-programmable, general-purpose input/output (I/O) controller. The general-purpose interface implements functions that are not implemented with the dedicated controllers in the device and require simple input and/or output software-controlled signals. The general-purpose interface allows a variety of custom connections and expands the I/O capabilities of the system to the real world.

Figure 25-2 shows a typical application using the general-purpose interface.

Figure 25-2. General-Purpose Interface Typical Application System Overview



Note: Temperature Sensing

Most memories provide a temperature sensor to control the auto-refresh duty cycle. The device monitors the temperature of the external memory using the pop_tq_temp_sense_ft ball and a GPIO input. To do this, pop_tq_temp_sense_ft is connected to a GPIO through the customer board. This feature is application-dependent.

CAUTION

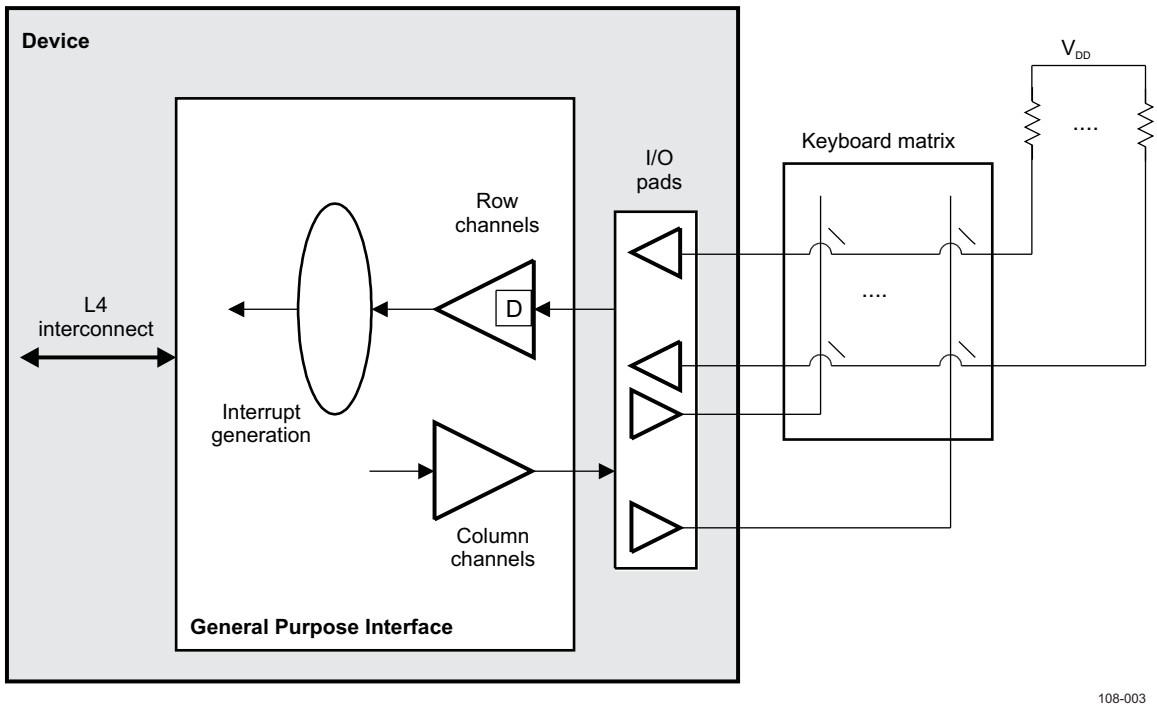
Due to buffer strength, an external serial resistor must be connected to the balls where gpio_120 through gpio_129 are muxed with MMC signals.

The general-purpose interface can physically connect the device to a keyboard matrix and peripheral integrated circuits (ICs).

25.2.1 GPIO as a Keyboard Interface

The general-purpose interface can be used as a keyboard interface. You can dedicate channels based on the keyboard matrix (r * c). Figure 25-3 shows row channels configured as inputs with the input debounce feature enabled. The row channels are driven high with an external pullup. Column channels are configured as outputs and drive a low level.

Figure 25-3. General-Purpose Interface Used as a Keyboard Interface



When a keyboard matrix key is pressed, the corresponding row and column lines are shorted together and a low level is driven on the corresponding row channel. This generates an interrupt based on the proper configuration (see Section 25.5.3, *Interrupt and Wakeup*).

When the keyboard interrupt is received, the processor (MPU and/or IVA2.2 subsystem) can disable the keyboard interrupt and scan the column channels for the key coordinates.

- The scanning sequence has as many states as column channels: For each step in the sequence, the processor drives one column channel low and the others high.
- The processor reads the values of the row channels and thus detects which keys in the column are pressed.

At the end of the scanning sequence, the processor establishes which keys are pressed. The keyboard interface can then be reconfigured in the interrupt waiting state.

25.2.2 General-Purpose Interface Functional Interfaces

25.2.2.1 Basic General-Purpose Interface Pins

Table 25-1 lists the interface pins of the general-purpose interface.

Table 25-1. General-Purpose Interface Functional Pins Description

Signal Name	I/O (1)	Description (2)	Module Reset Value
gpio_[31:0]	I/O	GPIO in configuration mode 4.	Input until software configuration
gpio_[186:34]	I/O	GPIO in configuration mode 4.	Input until software configuration
gpio_[191:188]	I/O	GPIO in configuration mode 4.	Input until software configuration

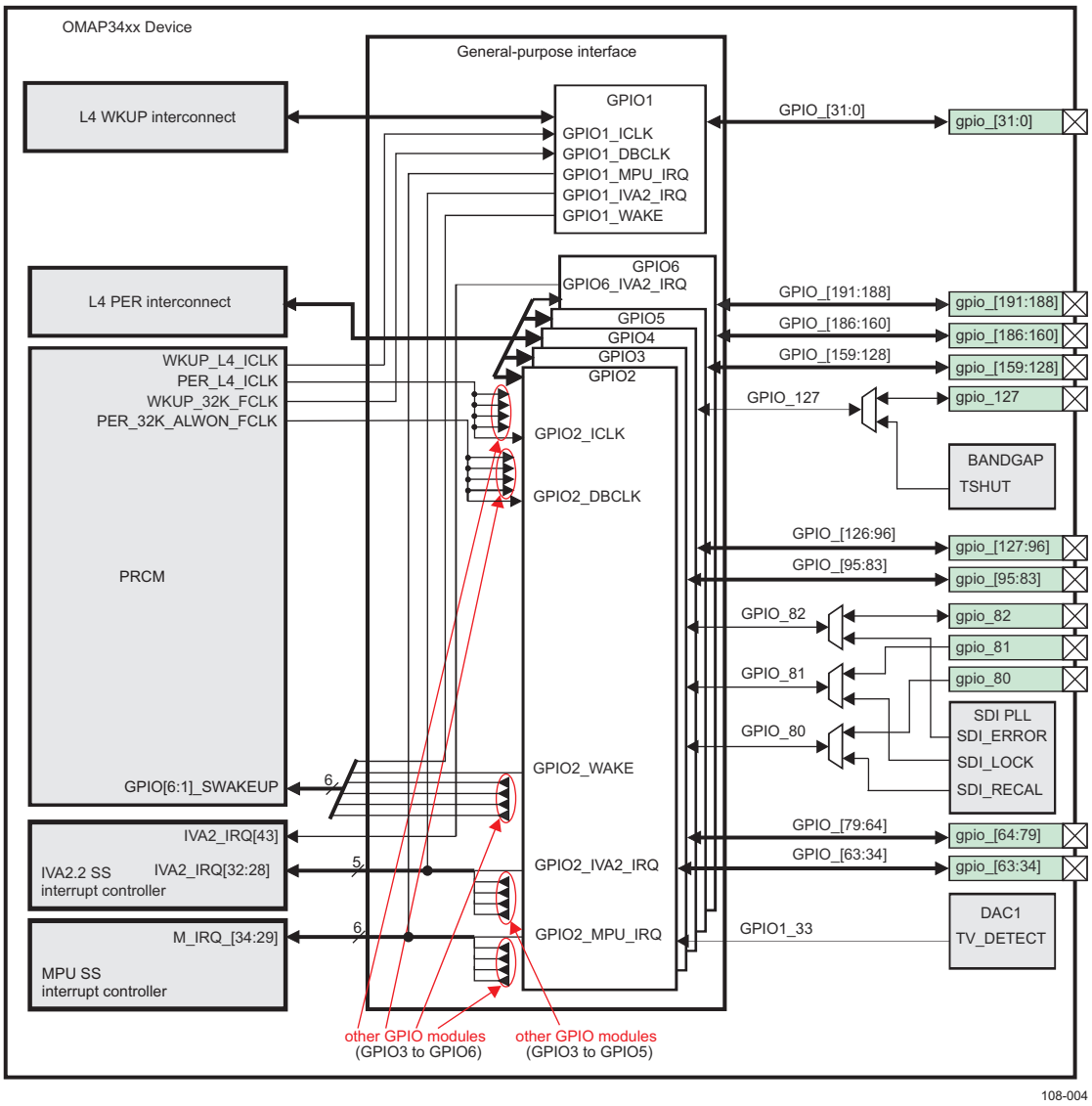
(1) I = Input, O = Output
(2) See Chapter 7, *System Control Module*, for more information about pin configuration modes.

25.3 General-Purpose Interface Integration

25.3.1 Description

Figure 25-4 highlights the general-purpose interface integration in the device.

Figure 25-4. General-Purpose Interface Integration Overview



25.3.1.1 Clocking, Reset, and Power-Management Scheme

25.3.1.1.1 Clocking

Each GPIO module uses two clocks:

- Debounce clock: The 32-KHz debounce clock, GPIOi_DBCLK, (where i = 1, 2, 3, 4, 5, and 6, with one debounce clock per module), comes from the PRCM module and is used for the debounce cell logic (without the corresponding configuration registers). This cell can sample the input line and filters the input level using a programmed delay.

  For GPIO2 to GPIO6, this clock is controlled by the EN_GPIOi (where i = 2 to 6) bit PRCM.CM_FCLKEN_PER (0: disabled, 1: enabled the clock). For GPIO1, this clock is controlled by EN_GPIO1 bit PRCM.CM_FCLKEN_WKUP[3] (0: disabled, 1: enabled the clock) for GPIO1.

- Interface clock: The interface clock, GPIOi_ICLK (where i = 1, 2, 3, 4, 5, and 6), comes from the PRCM module and is used throughout the GPIO module (except within the debounce cell logic). The interface clock clocks the data exchanges between the L4 interconnect and the internal logic. The clock-gating features allow module power consumption to be adapted to the activity.

  For GPIO1, this clock is controlled by the EN_GPIO1 bit PRCM.CM_ICLKEN_WKUP[3] (0: disabled, 1:enabled the clock) and AUTO_GPIO1 bit PRCM.CM_AUTOIDLE_WKUP[3] (enables/disables automatic control of the interface clock). For GPIO2 to GPIO6, this clock is controlled by the EN_GPIOi (where i = 2 to 6) bit PRCM.CM_ICLKEN_PER (0: disabled, 1: enabled the clock) and AUTO_GPIOi (where i = 2 to 6) bit PRCM.CM_AUTOIDLE_PER (enables/disables automatic control of the interface clock). Table 25-2 describes the GPIO module clocks.

Table 25-2. Clocks

Attribute	Frequency	Name	Mapping	Comments
Debounce clock	32 KHz	GPIOi_DBCLK, where I = 2 to 6	PER_32K_ALWON_FCLK	Source is PRCM module.
		GPIO1_DBCLK	WKUP_32K_CLK	
Interface clock	Depends on PRCM registers settings	GPIOi_ICLK, where I = 2 to 6	PER_L4_ICLK	
		GPIO1_ICLK	WKUP_L4_ICLK	

25.3.1.1.2 Reset

The general-purpose interface can be reset by using the domain reset (hardware reset) or by setting a dedicated configuration bit (software reset) in each GPIO module.

- Hardware reset: The GPIO2 to GPIO6 modules are attached to the PER_RST reset domain. The GPIO1 module is attached to the WKUP_RST reset domain.

  The hardware reset has a global reset action on the GPIO modules of the general-purpose interface. All configuration registers and internal logic are reset when it is active (low level). In each GPIO module, the RESETDONE bit GPIOi.GPIO_SYSSTATUS[0] monitors the internal reset status; it is set when the reset completes. For more information, see Chapter 4, *Power, Reset, and Clock Management*.
- Software reset: Each GPIO module has its own software reset using the GPIOi.GPIO_SYSCONFIG[1] SOFTRESET bit (where i = 1, 2, 3, 4, 5, or 6). The software reset has the same effect as the hardware reset signal, but this reset can be applied on one module or more.

  Writing 1 to SOFTRESET bit GPIOi.GPIO_SYSCONFIG[1](where i = 1, 2, 3, 4, 5, or 6) resets the module. Bit value 1 remains until the reset is complete. When the software reset is complete, the GPIOi.GPIO_SYSCONFIG[1] SOFTRESET bit is automatically reset to 0 and has the same effect as the hardware reset. The GPIOi.GPIO_SYSSTATUS[0] RESETDONE is cleared during a software reset. This bit is set to 1 when the software reset is complete.

25.3.1.1.3 Power Domain

The GPIO1 module is attached to the WKUP power domain (see Chapter 4, *Power, Reset, and Clock Management*). This domain is composed of the logic permanently supplied to manage domain power state transitions and detect wake-up events. The WKUP power domain is continuously active. The GPIO2 to GPIO6 modules are attached to the PER power domain (see Chapter 4, *Power, Reset, and Clock Management*). The PER power domain is not active continuously.

25.3.1.1.4 Power Management

25.3.1.1.4.1 Idle Scheme

To save dynamic consumption, an efficient idle scheme is based on the following:

- An efficient local autoclock gating for each module
- The implementation of control sideband signals between the PRCM module and each module

This enhanced idle control allows clocks to be activated and deactivated safely without requiring a complex software management.

The idle mode request, idle acknowledge, and wake-up request (GPIOi_SWAKEUP, where i = 1, 2, 3, 4, 5, and 6) are sideband signals between the PRCM module and the general-purpose interface (see Section 25.3.1.2, *Hardware Requests*).

25.3.1.1.4.2 Operating Modes

The following four operating modes are defined for the modules:

- Active mode: The module runs synchronously on the interface clock; interrupts can be generated based on the configuration and external signals.
- Idle mode: Power-saving mode with the module in a waiting state. The interface clock can be stopped, an interrupt cannot be generated, and a wake-up signal can be generated based on the configuration and external signals.

  If the debounce clock provided by the PRCM module is active, the debounce cell can sample and filter the input to generate a wake-up event. If the debounce clock is inactive, the debounce cell gates all input signals and thus cannot be used.
- Inactive mode: The module has no activity. The interface clock can be stopped, an interrupt cannot be generated, and the wake-up feature is inhibited.
- Disabled mode: The module is not used. The internal clock paths are gated, and an interrupt or wake-up request cannot be generated.

The idle and inactive modes are configured within the module and activated on request by the PRCM module (see Chapter 4, *Power, Reset, and Clock Management*) through sideband signals (see Section 25.3.1.1.4.3, *System Power Management and Wake-up*).

The disabled mode is set by software through a dedicated configuration bit, the GPIOi.GPIO_CTRL[0] DISABLEMODULE bit (0: the module is enabled and clocks are not gated; 1: the module is disabled and clocks are gated). It unconditionally gates the internal clock paths that are not used for the L4 interconnect.

25.3.1.1.4.3 System Power Management and Wake-Up

The PRCM module can require the GPIO modules to be idled for power saving purposes.

The general-purpose interface has six identical idle mode request/acknowledge (handshake) mechanisms with the PRCM module (see Figure 25-4 and Section 25.3.1.2, *Hardware Requests*): one per GPIO module. The general-purpose interface allows the GPIO modules to enter idle mode based on the GPIOi.GPIO_SYSCONFIG[4:3] IDLEMODE field.

The idle acknowledge depends on the configuration and activity of each GPIO module:

- Smart-idle mode (recommended)

  When the GPIO module is configured in smart-idle mode (GPIOi.GPIO_SYSCONFIG[4:3] IDLEMODE field [10]) and receives an idle request from the PRCM module (for GPIO2 to GPIO6: the corresponding bits in the PRCM.CM_FCLKEN_PER and PRCM.CM_ICLKEN_PER registers cleared to 0 or the corresponding bit in the PRCM.CM_AUTOIDLE_PER bit set to 1 and L4 interface clock idle transitions; for GPIO1: PRCM.CM_FCLKEN_ WKUP[3] EN_GPIO1 bit cleared to 0, PRCM.CM_ICLKEN_WKUP[3] EN_GPIO1 bit cleared to 0, or PRCM.CM_AUTOIDLE_WKUP[3] AUTO_ GPIO1 bit set to 1 and L4 interface clock idle transitions), the GPIO module checks for more activity (capture of the input GPIO pins in the GPIOi.GPIO_DATAIN register is complete with no pending interrupt; all interrupt status bits are cleared) ; and there is no access to GPIO.GPIO_DEBOUNCINGTIME register pending to be synchronized.

  Idle acknowledge is then asserted and the module enters in idle-mode. It waits for active system clock gating by the PRCM module (when all peripherals supplied by the same L4 interface clock domain are also ready for idle).

Idle mode (that is, when the PRCM module gates the interface clock), no interrupt occurs and the module is ready to issue a wake-up request.

When the expected transition occurs on an enabled GPIO input pin, the GPIO module exits from idle mode, if the GPIOi.GPIO_SYSCONFIG[2] ENAWAKEUP bit is set to 1 (wake-up capability enabled), and the corresponding bit in the PRCM.PM_WKEN_PER register is also set to 1 for the GPIO2 to GPIO6 modules, and/or the PRCM.PM_WKEN_WKUP[3] EN_GPIO1 bit is also set to 1 for GPIO1.

- Force-idle mode

  When the GPIO module is configured in force-idle mode (GPIOi.GPIO_SYSCONFIG[4:3] IDLEMODE field [00]) and receives an idle request from the PRCM module (for the GPIO2 to GPIO6: the corresponding bits in the PRCM.CM_FCLKEN_PER and PRCM.CM_ICLKEN_PER registers cleared to 0 or the corresponding bit in the PRCM.CM_AUTOIDLE_PER bit set to 1 and L4 interface clock idle transitions; for the GPIO1: PRCM.CM_FCLKEN_ WKUP[3] EN_GPIO1 bit cleared to 0, PRCM.CM_ICLKEN_WKUP[3] EN_GPIO1 bit cleared to 0, or PRCM.CM_AUTOIDLE_WKUP[3] AUTO_ GPIO1 bit set to 1 and L4 interface clock idle transitions), the GPIO module waits unconditionally for active system clock gating by the PRCM module. (This occurs only when all peripherals supplied by the same L4 interface clock domain are also ready for idle.)

  When in idle mode (that is, when the PRCM module gates the interface clock), the module (in inactive mode) has no activity, the interface clock paths are gated, an interrupt cannot be generated, and the wake-up feature is totally inhibited.

- No-idle mode

  When the GPIO module is configured in no-idle mode (GPIOi.GPIO_SYSCONFIG[4:3] IDLEMODE field [01]) and receives an idle request from the PRCM module (for the GPIO2 to GPIO6: the corresponding bits in the PRCM.CM_FCLKEN_PER and PRCM.CM_ICLKEN_PER registers cleared to 0 or the corresponding bit in the PRCM.CM_AUTOIDLE_PER bit set to 1 and L4 interface clock idle transitions; for the GPIO1: PRCM.CM_FCLKEN_ WKUP[3] EN_GPIO1 bit cleared to 0, PRCM.CM_ICLKEN_WKUP[3] EN_GPIO1 bit cleared to 0 or PRCM.CM_AUTOIDLE_WKUP[3] AUTO_ GPIO1 bit set to 1 and L4 interface clock idle transitions), the GPIO module does not go to the idle mode and the idle acknowledge is never sent.

Note: The GPIO2 to GPIO6 idle state can be checked by reading the corresponding status bits in the PRCM.CM_IDLEST_PER register (0: active, 1: idle) and is idle only when the GPIO2 to GPIO6 modules are configured in smart-idle mode and have asserted their idle acknowledge.

The GPIO1 idle state can be checked by the PRCM.CM_IDLEST_WKUP[3] ST_GPIO1 bit (0: idle, 1: active) and is idle only when the GPIO1 module is configured in smart-idle mode and has asserted its idle acknowledge.

The GPIO2 to GPIO6 wake-up status can be checked by accessing the corresponding bits in the PRCM.PM_WKST_PER register (read 0: no wakeup occurred; read 1: wakeup occurred; write 1: status bit reset).

The GPIO1 wake-up status can also be checked by the PRCM.PM_WKST_WKUP[3] ST_GPIO1 bit (read 0: no wakeup occurred; read 1: wakeup occurred; write 1: status bit reset).

25.3.1.1.4.4 Module Power Saving

The GPIO module has local power management by internal clock-gating features:

- Internal interface clock gating: The clock for the L4 interconnect logic can be gated when the module is not accessed, if the GPIOi.GPIO_SYSCONFIG[0] AUTOIDLE bit is set. Otherwise, this logic is free-running on the interface clock.
- Clock gating for the input data sample logic: Clock for the input data sample logic can be gated when the data in register is not accessed.
- Clock gating for the event detection logic: EachGPIO module implements four clock groups used for the logic in the synchronous events detection. Each group of eight input GPIO pins has a separate enable signal depending on the edge/level detection register setting. If a group requires no detection, the corresponding clock is gated off (see Section 25.5.1, *Power Saving by Grouping the Edge/Level Detection*). All channels are also gated using a one-out-of-N scheme. N is the GATINGRATIO field

GPIOi.GPIO_CTRL[2:1] and can take the values 1 (b00), 2 (0b01), 4 (b10), or 8 (0b11). The interface clock is enabled for this logic one cycle every N cycles. When N is equal to 1, there is no gating and this logic is free-running on the interface clock. When N is 2, 4, or 8, this logic is running at the equivalent frequency of interface clock frequency divided by N.

- Inactive mode: In inactive mode, all internal clock paths are gated.
- Disabled mode: All internal clock paths not used for the L4 interconnect are gated. The GPIOi.GPIO_CTRL[0] DISABLEMODULE bit controls a clock-gating feature at the module level. When set to 1, this bit forces clock gating for all internal clock paths. Module internal activity is suspended. The L4 interconnect is not affected by this bit.

  The interface clock gating is controlled with the GPIOi.GPIO_SYSCONFIG[0] AUTOIDLE bit, which is used to save power when the module is not used because of the multiplexing configuration selected at the chip level. This bit has precedence over all other internal configuration bits.

25.3.1.2 Hardware Requests

25.3.1.2.1 Interrupt Requests

All interrupt sources (the 32 input GPIO channels) are merged to issue two synchronous interrupt requests in each GPIO module. Thus, the general-purpose interface has 12 interrupt lines (two interrupt lines per GPIO module instance).

Synchronous interrupt request lines 1 and 2 are active depending on their respective interrupt enable 1 and 2 registers (GPIOi.GPIO_IRQENABLE1 and GPIOi.GPIO_IRQENABLE2).

- Synchronous interrupt request line 1 is mapped on the MPU interrupt controller.
- Synchronous interrupt request line 2 is mapped on the IVA2.2 interrupt controller.

Table 25-3 lists the interrupt lines that are driven out from the general-purpose interface to the MPU subsystem and IVA2.2 subsystem interrupt cont

Table 25-3. Interrupts

Name	Mapping	Comments
GPIO1 Module		
GPIO1_MPU_IRQ	M_IRQ_29	Destination is the MPU subsystem interrupt controller.
GPIO1_IVA2_IRQ	IVA2_IRQ[28]	Destination is the IVA2.2 subsystem interrupt controller.
GPIO2 Module		
GPIO2_MPU_IRQ	M_IRQ_30	Destination is the MPU subsystem interrupt controller.
GPIO2_IVA2_IRQ	IVA2_IRQ[29]	Destination is the IVA2.2 subsystem interrupt controller.
GPIO3 Module		
GPIO3_MPU_IRQ	M_IRQ_31	Destination is the MPU subsystem interrupt controller.
GPIO3_IVA2_IRQ	IVA2_IRQ[30]	Destination is the IVA2.2 subsystem interrupt controller.
GPIO4 Module		
GPIO4_MPU_IRQ	M_IRQ_32	Destination is the MPU subsystem interrupt controller.
GPIO4_IVA2_IRQ	IVA2_IRQ[31]	Destination is the IVA2.2 subsystem interrupt controller.
GPIO5 Module		
GPIO5_MPU_IRQ	M_IRQ_33	Destination is the MPU subsystem interrupt controller.
GPIO5_IVA2_IRQ	IVA2_IRQ[32]	Destination is the IVA2.2 subsystem interrupt controller.
GPIO6 Module		
GPIO6_MPU_IRQ	M_IRQ_34	Destination is the MPU subsystem interrupt controller.
GPIO6_IVA2_IRQ	IVA2_IRQ[43]	Destination is the IVA2.2 subsystem interrupt controller.

25.3.1.2.1.1 Wake-Up Generation

The GPIO1 module of the general-purpose interface is attached to the WKUP power domain (see Chapter 4, *Power, Reset, and Clock Management)* and can wake up the system.

Note: The GPIO2 to GPIO6 modules belong to the PER power domain and thus have wake-up system capability only when the PER power domain is active.

All wake-up sources (the 32 input GPIO channels) are merged together to issue a single asynchronous wake-up request in each GPIO module following the expected transition(s) (based on register programming). Each GPIO module generates a wake-up signal to the PRCM module.

Note: Only gpio_1, gpio_9, gpio_10, gpio_11, gpio_30, and gpio_31 can be used to generate a direct wake-up event. The other GPIO1 pins cannot be used to generate a direct wake-up event because they are connected to the device I/O pad logic in the CORE power domain (VDD2). When the CORE power domain is off, the I/O pins of the GPIO1 module, which are supplied by VDD2, cannot generate a wake-up event.

The asynchronous wake-up request line is active based on the GPIOi.GPIO_WAKEUPENABLE register (where i = 1, 2, 3, 4, 5, and 6).

CAUTION

The wake-up capabilities of the GPIO2 to GPIO6 modules are operational only when the PER power domain is active.

Table 25-4 shows the wake-up signals mapping.

Table 25-4. Wake-Up Signals

Name	Mapping	Comments
GPIOi_WAKE	GPIOi_SWAKEUP	Where i = 1, 2, 3, 4, 5, and 6. Destination is the PRCM module.

Table 25-5 describes the GPIO channels.

Table 25-5. GPIO Channel Description

Channel Number	Type (1)	Mapping	Wake-Up Feature	Comments
GPIO1 Module				
[31:0]	I/O	gpio_[31:0]	Yes	GPIO (2)
GPIO2 Module				
[0]	I	-	No	Not available on external balls. Read value is always 0.
[1]	I	TV_DETECT	Yes (3)	Internal TV detection signal from the 10-bit composite/luma video DAC1
[31:2]	I/O	gpio_[63:34]	Yes (3)	GPIO (2)
GPIO3 Module				
[15:0]	I/O	gpio_[79:64]	Yes (3)	GPIO (2)
[16]	I/O	gpio_80	Yes (3)	GPIO (2)
	I	SDI_RECAL	Yes (3)	Internal SDI_RECAL signal from the SDI PLL module (4)
[17]	I/O	gpio_81	Yes (3)	GPIO (2)
	I	SDI_LOCK	Yes (3)	Internal SDI_LOCK signal from the SDI PLL module (4)

(1) I = Input, O = Output
(2) Configuration mode 4. See Chapter 7, *System Control Module*.
(3) Only when the PER power domain is active
(4) All configuration modes except configuration mode 4. See Chapter 7, *System Control Module*.

Table 25-5. GPIO Channel Description (continued)

Channel Number	Type [1]	Mapping	Wake-Up Feature	Comments
[18]	I/O	gpio_82	Yes [3]	GPIO [2]
	I	SDI_ERROR	Yes [3]	Internal SDI_ERROR signal from the SDI PLL module [4]
[31:19]	I/O	gpio_[95:83]	Yes [3]	GPIO [2]
GPIO4 Module				
[30:0]	I/O	gpio_[126:96] [5]	Yes [3]	GPIO [2]
[31]	I/O	gpio_127	Yes [3]	GPIO [2]
	I	TSHUT	Yes [3]	Internal TSHUT signal from the BANDGAP module for the SRAMs LDOs [4]
GPIO5 Module				
[31:0]	I/O	gpio_[159:128]	Yes [3]	GPIO [2]
GPIO6 Module				
[26:0]	I/O	gpio_[186:160]	Yes [3]	GPIO [2]
[27]	I	-	No	Not available on external balls. Read value is always 0.
[31:28]	I/O	gpio_[191:188]]	Yes [3]	GPIO [2]

(5) All signals are I/O type, except gpio_99, gpio_100, gpio_112, gpio_113, gpio_114, and gpio_115, which are input-only signals.

Note: The thermal shutdown comparator output signal (TSHUT) is an output from the BANDGAP module. This signal is low during normal operation and goes high during a thermal shutdown event. When channel 31 of the GPIO4 is not connected to a ball of the device (the corresponding pin is configured in a mode different from the configuration mode 4; see Chapter 7, *System Control Module*, for more information about pin configuration), TSHUT is connected to channel 31 of the GPIO4, and an interrupt can be generated when a low-to-high transition occurs on TSHUT whether or not the interrupt generation for channel 31 of the GPIO4 is correctly configured.

25.4 General-Purpose Interface Functional Description

Figure 25-5 shows the general-purpose interface description.

Figure 25-5. General-Purpose Interface Description

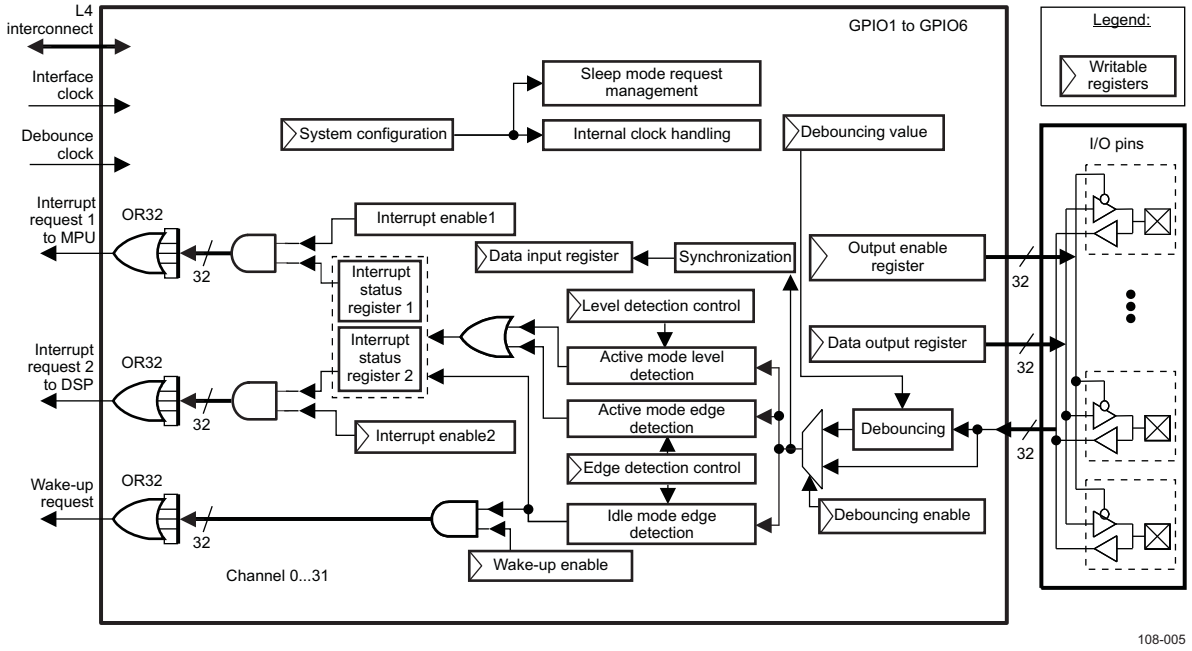
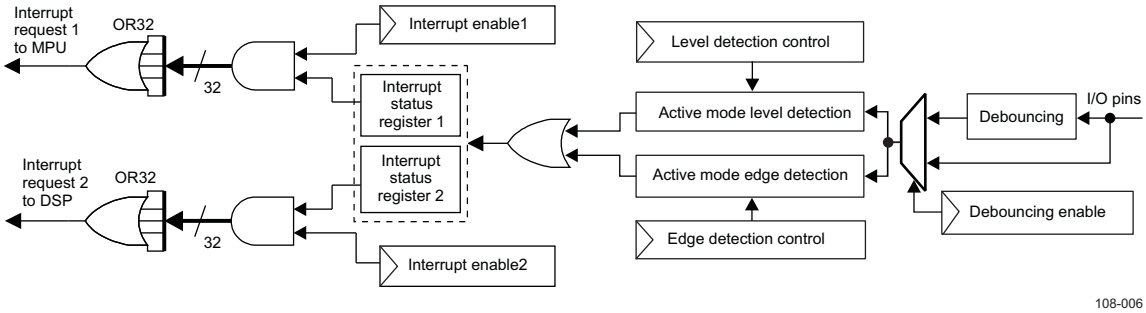


Figure 25-5 details the GPIO modules in the general-purpose interface block diagram with their configuration registers and their main functional paths:

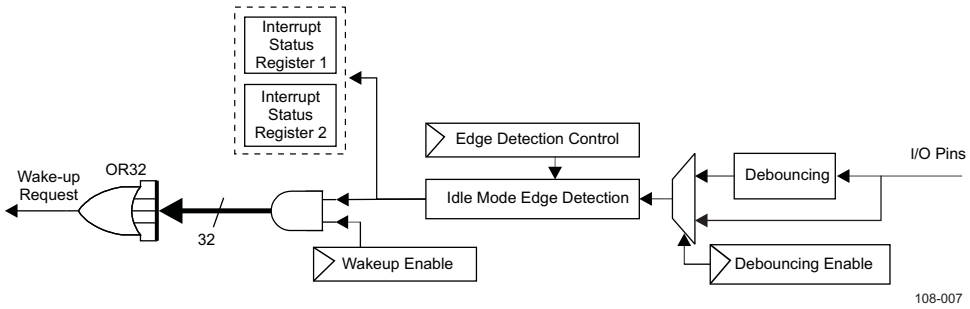
- The synchronous path (for active mode operation) used to generate a synchronous interrupt request on expected event detection on any input GPIO; the synchronous interrupt request lines 1 and 2 are active based on their respective interrupt enable 1 and 2 registers (GPIOi.GPIO_IRQENABLE1 and GPIOi.GPIO_IRQENABLE2). See Figure 25-6.

Figure 25-6. Synchronous Path



- The asynchronous path (for idle mode operation) used to generate an asynchronous wake-up request on the expected edge detection on any input GPIO; the asynchronous wake-up request line is active based on the wake-up enable register. See Figure 25-7.

Figure 25-7. Asynchronous Path



- The blocks handling the internal clock (clock gating) and managing the sleep mode request/acknowledge protocol (enabling the synchronous path in active mode and the asynchronous path in idle mode).

25.4.1 Interrupt and Wake-Up Features

25.4.1.1 Synchronous Path: Interrupt Request Generation

The general-purpose interface has 12 interrupt lines (two interrupt lines per GPIO module instance). The 12 interrupt signals are GPIOi_MPU_IRQ (used by the MPU subsystem) and GPIOi_IVA2_IRQ (used by the IVA2.2 subsystem), where i = 1, 2, 3, 4, 5, and 6.

Synchronous interrupt requests from each channel are processed by two identical interrupt generation submodules used independently by the IVA2.2 subsystem and the MPU subsystem. Each submodule controls its own synchronous interrupt request line and has its own interrupt enable (GPIOi.GPIO_IRQENABLE1 or GPIOi.GPIO_IRQENABLE2) and interrupt status (GPIOi.GPIO_IRQSTATUS1 or GPIOi.GPIO_IRQSTATUS2) registers. The interrupt enable register selects the channel(s) considered for the interrupt request generation, and the interrupt status register determines which channel(s) activate the interrupt request. Event detection on GPIO channels is reflected in the interrupt status registers independent of the content of the interrupt enable registers.

In active mode, when the GPIO configuration registers are set to enable the interrupt generation (see Section 25.5.3, *Interrupt and Wakeup*), a synchronous path samples the transitions and levels on the input GPIO with the internally gated interface clock (see Section 25.3.1.1.4.4, *Module Power Saving*). When an event matches the programmed settings (see Section 25.5.3, *Interrupt and Wakeup*), the corresponding bit in the interrupt status register is set to 1 and, on the following interface clock cycle, the interrupt lines 1 and/or 2 are activated (depending on the interrupt enable registers).

Because of the sampling operation, the minimum pulse width on the input GPIO to trigger a synchronous interrupt request is two times the internally gated interface clock period (the internally gated interface clock period equals *N* times the interface clock period; see Section 25.3.1.1.4.4, *Module Power Saving*). This minimum pulse width must be met before and after any expected level transition detection. Level detection requires the selected level to be stable for at least two times the internally gated interface clock period to trigger a synchronous interrupt.

Because the module is synchronous, latency is minimal between the expected event occurrence and the activation of the interrupt line(s). This latency must not exceed four internally gated interface clock cycles + one interface clock cycle when the debounce feature is not used.

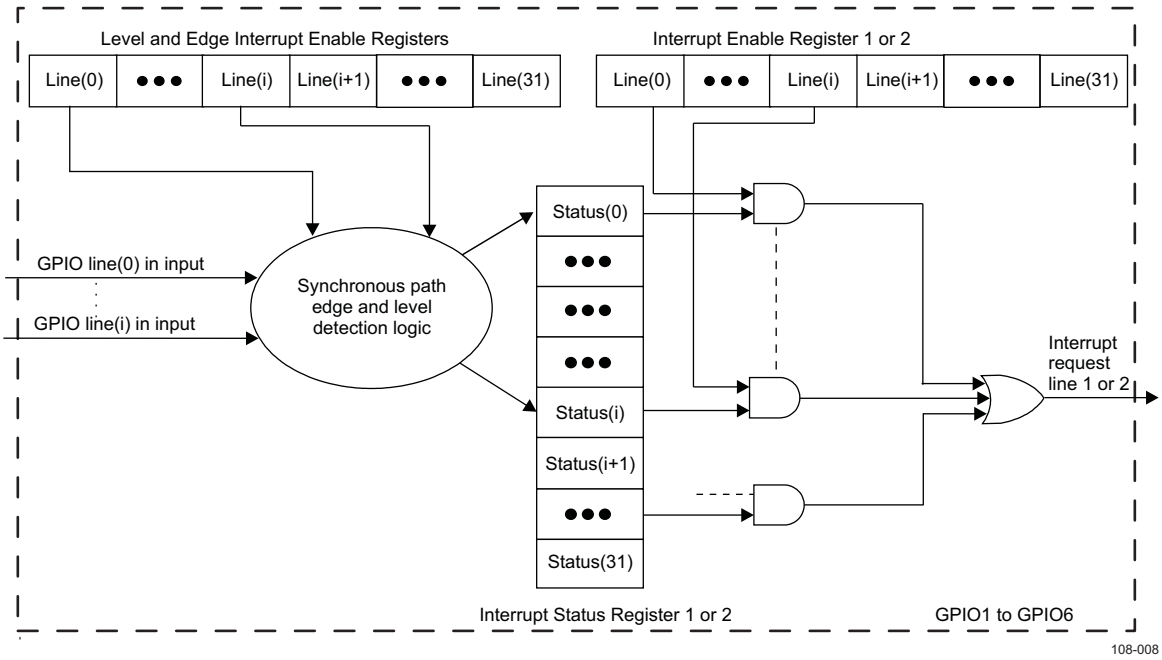
When the debounce feature is active, the latency depends on the debouncing time register (GPIOi.GPIO_DEBOUNCINGTIME) value (see Section 25.5.5, *Debouncing Time*) and is less than three internally gated interface clock cycles + two interface clock cycle + GPIOi.GPIO_DEBOUNCINGTIME register value debounce clock cycles + three debounce clock cycles.

Synchronous interrupt request line 1 is mapped on the MPU interrupt controller.

Synchronous interrupt request line 2 is mapped on the IVA2.2 interrupt controller.

Figure 25-8 shows an overview of the interrupt request generation.

Figure 25-8. Interrupt Request Generation



25.4.1.2 Asynchronous Path: Wake-Up Request Generation

The general-purpose interface has six wake-up lines (one wake-up line per GPIO module instance) connected to the PRCM module.

Asynchronous wake-up requests from input channels are merged to issue one wake-up signal to the system per GPIO module. The wake-up enable register (GPIOi.GPIO_WAKEUPENABLE) selects the channel(s) considered for the wake-up request generation. The asynchronous wake-up request is reflected into the synchronous interrupt status registers (GPIOi.GPIO_IRQSTATUS1 and GPIOi.GPIO_IRQSTATUS2).

In idle mode (the interface clock is shut down and the GPIO configuration registers are programmed; see Section 25.5.3, *Interrupt and Wakeup*), an asynchronous path detects the expected transition(s) on a GPIO input (based on register programming) and activates an asynchronous wake-up request by the sideband signal (GPIOi_SWAKEUP, where i = 1, 2, 3, 4, 5, and 6), if the wake-up enable register is set.

When the system is awakened, the interface clock is restarted and synchronously set to 1 based on the input GPIO pin triggering the wake-up request and the corresponding bit in the interrupt status registers (GPIOi.GPIO_IRQSTATUS1 and GPIOi.GPIO_IRQSTATUS2). On the following internal clock cycle, the interrupt lines 1 and/or 2 are active (active low) when the corresponding bits are set in the interrupt enable registers (GPIOi.GPIO_IRQENABLE1 and GPIOi.GPIO_IRQENABLE2).

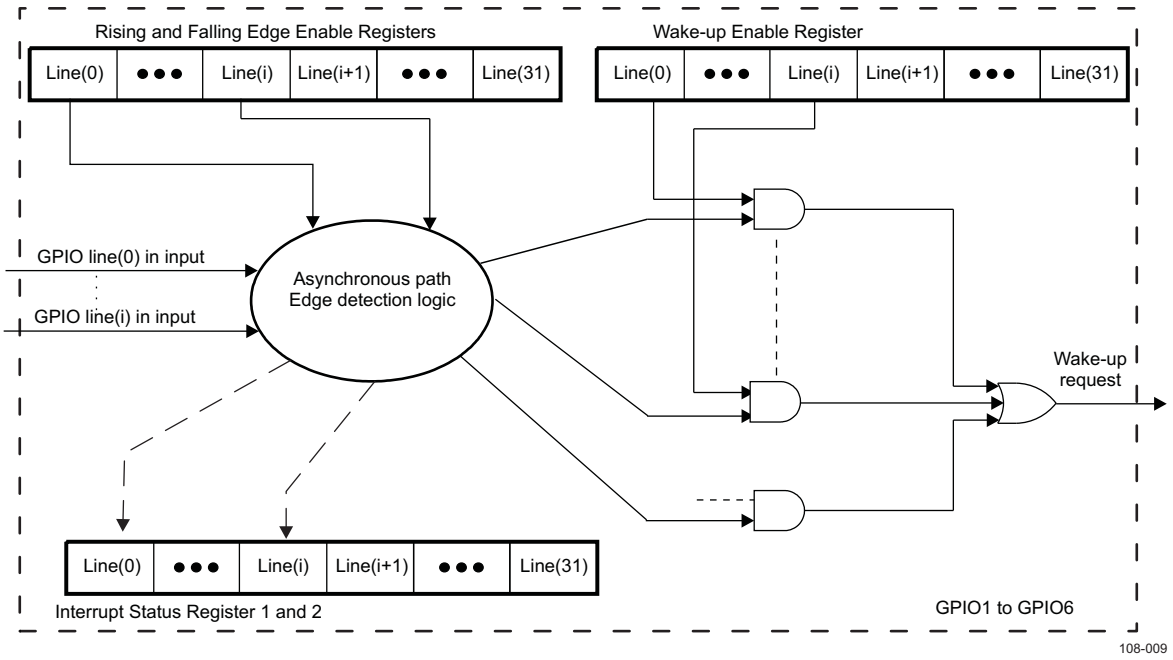
Note: When debouncing is not enabled, a minimum input pulse width does not trigger the wake-up request because there is no sampling operation.

When debouncing is enabled, the minimum pulse width is set by the specified debouncing time.

The GPIOi.GPIO_SYSCONFIG[2] ENAWAKEUP bit enables or disables the GPIO wake-up feature globally. If the bit is 0, the wake-up enable register (GPIOi.GPIO_WAKEUPENABLE) has no effect.

Figure 25-9 shows an overview of the wake-up request generation.

Figure 25-9. Wake-Up Request Generation



25.4.1.3 Interrupt (or Wake-Up) Line Release

When the host processor (the MPU and/or IVA2.2 subsystem in the device) receives an interrupt request issued by the GPIO module, it reads the corresponding interrupt status register (GPIOi.GPIO_IRQSTATUS1 or GPIOi.GPIO_IRQSTATUS2) to determine which GPIO input triggered the interrupt (or the wake-up request).

After servicing the interrupt (or acknowledging the wake-up request), the processor resets the status bit and releases the interrupt line by writing 1 in the corresponding bit of the interrupt status register. If there is still a pending interrupt request to serve (all bits in the interrupt status register that are not masked by the interrupt enable register are not cleared), the interrupt line is reasserted.

Note: The status bit must be reset to re-enter idle mode.

25.5 General-Purpose Interface Basic Programming Model

25.5.1 Power Saving by Grouping the Edge/Level Detection

Each GPIO module implements four gated clocks used by the edge/level detection logic to save power. Each group of eight input GPIO pins generates a separate enable signal depending on the edge/level detection register setting (because the input is 32 bits, four groups of eight inputs are defined for each GPIO module). If a group requires no edge/level detection, then the corresponding clock is gated (cut off). Grouping the edge/level enable can save the power consumption of the module as described in the following example.

If any of the registers:

GPIOi.GPIO_LEVELDETECT0
GPIOi.GPIO_LEVELDETECT1
GPIOi.GPIO_RISINGDETECT
GPIOi.GPIO_FALLINGDETECT

are set to 0x01 01 01 01, then all clocks are active (power consumption is high).

are set to 0x00 00 00 FF, then a single clock is active (power saving).

Note: When the clocks are enabled by writing to the GPIOi.GPIO_LEVELDETECT0, GPIOi.GPIO_LEVELDETECT1, GPIOi.GPIO_RISINGDETECT, and GPIOi.GPIO_FALLINGDETECT registers, the detection starts after five clock cycles. This period is required to clean the synchronization edge/level detection pipeline.

The mechanism is independent of each clock group. If the clock has been started before and a new setting is performed, the following is recommended: First, set the new detection required; second, disable the previous setting (if necessary). In this way, the corresponding clock is not gated and the detection starts immediately.

25.5.2 Set-and-Clear Instructions

25.5.2.1 Description

The GPIO module implements the set-and-clear protocol register update for the GPIOi.GPIO_DATAOUT, GPIOi.GPIO_IRQENABLE1, GPIOi.GPIO_IRQENABLE2, and GPIOi.GPIO_WAKEUPENABLE registers. This protocol is an alternative to the atomic test and set operations and consists of writing operations at dedicated addresses (one address for setting bit[s] and one address for clearing bit[s]). The data to write is 1 at bit position(s) to clear (or to set) and 0 at unaffected bit(s). Registers can be accessed in two ways:

- Standard: Full register read and write operations at the primary register address
- Set and clear (recommended): Separate addresses are provided to set (and clear) bits in registers. Writing 1 at these addresses sets (or clears) the corresponding bit into the equivalent register; writing a 0 has no effect.

Therefore, for these registers, three addresses are defined for one unique physical register. Reading these addresses has the same effect and returns the register value.

25.5.2.2 Clear Instruction

25.5.2.2.1 Clear Registers Addresses

Clear interrupt enable registers (GPIOi.GPIO_CLEARIRQENABLE1 and GPIOi.GPIO_CLEARIRQENABLE2).

A write operation in the clear interrupt enable1 (or enable2) register clears the corresponding bit in the interrupt enable1 (or enable2) register when the written bit is 1; a written bit at 0 has no effect.

A read of the clear interrupt enable1 (or enable2) register returns the value of the interrupt enable1 (or enable2) register

Clear wake-up enable register (GPIOi.GPIO_CLEARWKUENA).

A write operation in the clear wake-up enable register clears the corresponding bit in the wake-up enable register when the written bit is 1; a written bit at 0 has no effect.

A read of the clear wake-up enable register returns the value of the wake-up enable register.

Clear data output register (GPIOi.GPIO_CLEARDATAOUT).

A write operation in the clear data output register clears the corresponding bit in the data output register when the written bit is 1; a written bit at 0 has no effect.

A read of the clear data output register returns the value of the data output register.

25.5.2.2.2 Clear Instruction Example

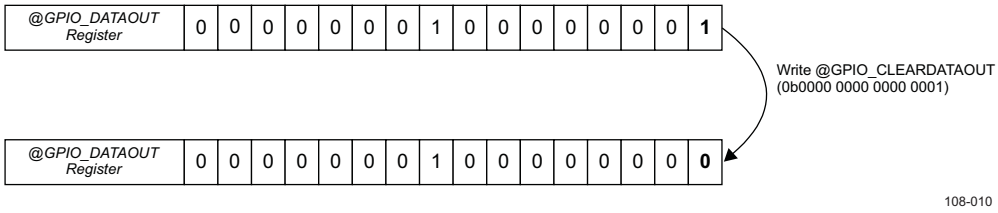
Assume the data output register (or one of the interrupt/wake-up enable register) contains the binary value, 0b0000 0001 0000 0001, and you want to clear the bit 0.

With the clear instruction feature, write 0b0000 0000 0000 0001 at the address of the clear data output register (or at the address of the clear interrupt/wake-up enable register). After this write operation, a reading of the data output register (or the interrupt/wake-up enable register) returns 0b0000 0001 0000 0000; the bit 0 is cleared.

Note: Although the general-purpose interface registers are 32 bits wide, only the less-significant 16 bits are represented in this example.

Figure 25-10 shows an example of a clear instruction.

Figure 25-10. Write @GPIO_CLEARDATAOUT Register Example



25.5.2.3 Set Instruction

25.5.2.3.1 Set Registers Addresses

Set interrupt enable registers (GPIOi.GPIO_SETIRQENABLE1 and GPIOi.GPIO_SETIRQENABLE2).

A write operation in the set interrupt enable1 (or enable2) register sets the corresponding bit in the interrupt enable1 (or enable2) register when the written bit is 1; a written bit at 0 has no effect.

A read of the set interrupt enable1 (or enable2) register returns the value of the interrupt enable1 (or enable2) register.

Set wake-up enable register (GPIOi.GPIO_SETWKUENA).

A write operation in the set wake-up enable register sets the corresponding bit in the wake-up enable register when the written bit is 1; a written bit at 0 has no effect.

A read of the set wake-up enable register returns the value of the wake-up enable register.

Set data output register (GPIOi.GPIO_SETDATAOUT).

A write operation in the set data output register sets the corresponding bit in the data output register when the written bit is 1; a written bit at 0 has no effect.

A read of the set data output register returns the value of the data output register.

25.5.2.3.2 Set Instruction Example

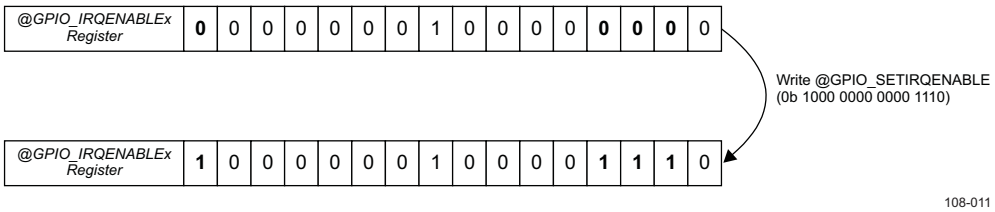
Assume the interrupt enable1 (or enable2) register (or the data output register) contains the binary value, 0b0000 0001 0000 0000, and you want to set the bits 15, 3, 2, and 1.

With the set instruction feature, write 0b1000 0000 0000 1110 at the address of the set interrupt enable1 (or enable2) register (or at the address of the set data output register). After this write operation, a reading of the interrupt enable1 (or enable2) register (or the data output register) returns 0b1000 0001 0000 1110; the bits 15, 3, 2, and 1 are set.

Note: Although the general-purpose interface registers are 32 bits wide, only the less-significant 16 bits are represented in this example.

Figure 25-11 shows an example of a set instruction.

Figure 25-11. Write @GPIO_SETIRQENABLEx Register Example



The set wake-up enable register offers the same feature with the wake-up enable register.

25.5.3 Interrupt and Wakeup

25.5.3.1 Involved Configuration Registers

- Interrupt enable registers (GPIOi.GPIO_IRQENABLE1 and GPIOi.GPIO_IRQENABLE2)

  The interrupt enable1 (or interrupt enable2) register allows masking of the expected transition on input GPIO to prevent the generation of an interrupt request on line1 (or line2). The interrupt enable registers are programmed synchronously with the interface clock.

  These registers can be accessed with direct read/write operations or using the alternate set and clear protocol register update feature. This feature enables to set or clear specific bits of these registers with a single write access to the corresponding set interrupt enable1 (or interrupt enable2) registers (or to the clear interrupt enable1 [or interrupt enable2] registers) address (see Section 25.5.2, *Set-and-Clear Instructions*).

- Wake-up enable register (GPIOi.GPIO_WAKEUPENABLE)

  The wake-up enable register allows masking of the expected transition on input GPIO to prevent the generation of a wake-up request. The wake-up enable register is programmed synchronously with the interface clock before any idle mode request coming from the host processor.

  This register can be accessed with direct read/write operations or by using the alternate set and clear protocol register update feature. This feature allows setting or clearing specific bits of this register with a single write access to the set wake-up enable register (or to the clear wake-up enable register) address (see Section 25.5.2, *Set-and-Clear Instructions*).

  Note: It must be a correlation between Wake-up enable and interrupt enable registers. If a GPIO pin has a Wake-up configured on it, it should also have the corresponding interrupt enabled (on one of the 2 interrupt lines). Otherwise, it is possible to have a Wake-up event, but after exiting the Idle state, no interrupt will be generated, thus the corresponding bit from the interrupt status register will not be cleared, and the module will not acknowledge a future Idle Request.

- Interrupt status registers (GPIOi.GPIO_IRQSTATUS1 and GPIOi.GPIO_IRQSTATUS2)

  The interrupt status1 (or interrupt status2) register determines which of the input GPIO pins triggered

the interrupt line1 (or interrupt line2) request (or the wake-up line).

When a bit in this register is set to 1, it indicates that the corresponding GPIO pin is requesting the interrupt (or the wake-up). To reset a bit in this register, write 1 to the appropriate bit. However, an interrupt cannot be generated by writing 1 to the interrupt status1 (or interrupt status2) register.

If 0 is written to a bit in this register, the value remains unchanged. The interrupt status1 (or interrupt status2) register is synchronous with the interface clock. In idle mode, the event is detected via an asynchronous path, and the corresponding bit in the interrupt status1 and interrupt status2 registers are set when the GPIO module is awake.

Note: The wake-up capabilities of the GPIO2 to GPIO6 modules are operational only when the PER power domain is active.

25.5.3.2 Description

To generate interrupt request to a host processor (the MPU and/or DSP subsystem in the device) at a defined event (level or edge logic transition) occurring on a GPIO pin (interrupt source), the GPIO configuration registers must be programmed as follows:

1. The GPIO channel must be configured as input by the output enable register (write 1 to the corresponding bit of the GPIOi.GPIO_OE register).
2. The expected event(s) on the GPIO input to trigger the interrupt request must be selected in the low-level interrupt enable register (write 1or 0 to the corresponding bit of GPIOi.GPIO_LEVELDETECT0), and/or high-level interrupt enable register (write 1 or 0 to the corresponding bit of GPIOi.GPIO_LEVELDETECT1), and/or rising-edge interrupt/wake-up enable register (write 1or 0 to the corresponding bit of GPIOi.GPIO_RISINGDETECT), and/or falling edge interrupt/wake-up enable register (write 1or 0 to the corresponding bit of GPIOi.GPIO_FALLINGDETECT).

   Note: Interrupt generation on both edges on one input is configured by setting the corresponding bit to 1 in the rinsing detect enabling register (GPIOi.GPIO_RISINGDETECT) and falling detect enabling register (GPIOi.GPIO_FALLINGDETECT) along with the interrupt enable by setting the corresponding bit to 1 in on one or both interrupt enable registers (GPIOi.GPIO_IRQENABLE1 and GPIOi.GPIO_IRQENABLE2).

   Enabling at the same time high level detection and low level detection for one given pin makes a constant interrupt generator.
3. Interrupts from the GPIO channel must be enabled in the interrupt 1 enable register (write 1 to the corresponding bit of GPIOi.GPIO_IRQENABLE1 register) and/or the interrupt 2 enable register (write 1 to the corresponding bit of GPIOi.GPIO_IRQENABLE2 register).

To configure a GPIO module to sent a wake-up request to the PRCM at a defined event (logic transition) occurring on a GPIO pin (wake-up source), the GPIO configuration registers must be programmed as follows:

1. The GPIO pin must be configured as input by the output enable register (write 1 to the corresponding bit of the GPIOi.GPIO_OE register).
2. The expected event(s) on the GPIO input to trigger the wake-up request must be selected in the rising-edge interrupt/wake-up enable register (write 1 or 0 to the corresponding bit of GPIOi.GPIO_RISINGDETECT) and/or falling-edge interrupt/wake-up enable register (write 1 or 0 to the corresponding bit of GPIOi.GPIO_FALLINGDETECT). The wake-up request can only be generated on edge transitions.
3. The GPIO channel must be enabled in the wake-up enable register (write 1 to the corresponding bit of the GPIOi.GPIO_WAKEUPENABLE).
4. The wake-up request generation on the expected transition occurring on the GPIO input pins must enable for the module (write 1 to the corresponding bit of the GPIOi.GPIO_SYSCONFIG[2] ENAWAKEUP) .

CAUTION

For each GPIO channel used, do not forget to configure the corresponding pad configuration registers in Chapter 7 , *System Control Module*.

After servicing the interrupt, the status bit in the interrupt status register (GPIOi.GPIO_IRQSTATUS1 or GPIOi.GPIO_IRQSTATUS2) must be reset and the interrupt line released (by writing 1 in the corresponding bit of the interrupt status register) before enabling an interrupt for the GPIO channel in the interrupt enable register (GPIOi.GPIO_IRQENABLE1 or GPIOi.GPIO_IRQENABLE2) to prevent the occurrence of unexpected interrupts when enabling an interrupt for the GPIO channel.

25.5.4 Data Input (Capture)/Output (Drive)

The output enable register (GPIOi.GPIO_OE) controls the output/input capability for each pin. At reset, all the GPIO-related pins are configured as input and output capabilities are disabled. This register is not used within the module. Its only function is to carry the pads configuration.

When configured as an output (the desired bit reset in the GPIOi.GPIO_OE register), the value of the corresponding bit in the GPIOi.GPIO_DATAOUT register is driven on the corresponding GPIO pin. Data is written to the data output register synchronously with the interface clock. This register can be accessed with read/write operations or by using the alternate set and clear protocol register update feature. This feature lets you set or clear specific bits of this register with a single write access to the set output data register (GPIOi.GPIO_SETDATAOUT) or to the clear output data register (GPIOi.GPIO_CLEARDATAOUT) address (see Section 25.5.2, *Set-and-Clear Instructions*). If the application uses a pin as an output and does not want interrupt/wake-up generation from this pin, the application must properly configure the wake-up enable (GPIOi.GPIO_ WAKEUPENABLE) and the interrupt enable (GPIOi.GPIO_IRQENABLE1 and GPIOi.GPIO_IRQENABLE2) registers.

When configured as an input (the desired bit set to 1 in the GPIOi.GPIO_OE register), the state of the input can be read from the corresponding bit in the GPIOi.GPIO_DATAIN register. The input data is sampled synchronously with the interface clock and then captured in the data input register synchronously with the interface clock (see Section 25.5.2, *Set-and-Clear Instructions*). When the GPIO pin levels change, they are captured into this register after two interface clock cycles (the required cycles to synchronize and to write data). If the application uses a pin as an input, the application must properly configure the wake-up enable (GPIOi.GPIO_WAKEUPENABLE) and the interrupt enable (GPIOi.GPIO_IRQENABLE1 and GPIOi.GPIO_IRQENABLE2) registers to the interrupt and wake-up feature as needed.

25.5.5 Debouncing Time

To enable the debounce feature for a pin, the GPIO configuration registers must be programmed as follows:

1. The GPIO pin must be configured as input in the output enable register (write 1 to the corresponding bit of the GPIOi.GPIO_OE register)
2. The deboucing time must be set in the deboucing time register (GPIOi.GPIO_DEBOUNCINGTIME)

   The debouncing value register (GPIOi.GPIO_DEBOUNCINGTIME) is used to set the debouncing time for all input lines in the GPIO module. The value is global for all the ports of one GPIO module, so up to six different debouncing values are possible. The debounce cell is running with the debounce clock (32 kHz). This register represents the number of the clock cycle(s) (one cycle is 31 microseconds long) to be used.

   The following formula describes the required input stable time to be propagated to the debounced output:

   Required input line stable = (GPIOi.GPIO_DEBOUNCINGTIME[7:0] DEBOUNCVAL field value + 1) x 31 µs.

   where GPIOi.GPIO_DEBOUNCINGTIME[7:0] field DEBOUNCVAL value is from 0 to 255.
3. The debouncing feature must be enabled in the debouncing enable register (write 1 to the corresponding bit of the GPIOi.GPIO_DEBOUNCENABLE register)

25.6 General-Purpose Interface Register Manual

This section summarizes the hardware interface for the GPIO product. Each module instance within the design is shown, together with the module register map and bit definitions for each bit field.

Table 25-6 shows the base address and address space for the GPIO module instances.

Table 25-6. Instance Summary

Module Name	Base Address	Size
GPIO1	0x4831 0000	4K bytes
GPIO2	0x4905 0000	4K bytes
GPIO3	0x4905 2000	4K bytes
GPIO4	0x4905 4000	4K bytes
GPIO5	0x4905 6000	4K bytes
GPIO6	0x4905 8000	4K bytes

25.6.1 General-Purpose Interface Register Mapping Summary

All module registers are 8-, 16-, or 32-bit accessible through the L4 interconnect (little endian encoding). Access to registers is direct; no shadow registers are implemented.

Table 25-7 through Table 25-8 describe the GPIO register offset addresses.

Table 25-7. GPIO1 to GPIO3 Register Summary

Register Name	Type	Register Width (Bits)	Address Offset	Physical Address (GPIO1)	Physical Address (GPIO2)	Physical Address (GPIO3)
GPIO_REVISION	R	32	0x000	0x4831 0000	0x4905 0000	0x4905 2000
GPIO_SYSCONFIG	RW	32	0x010	0x4831 0010	0x4905 0010	0x4905 2010
GPIO_SYSSTATUS	R	32	0x014	0x4831 0014	0x4905 0014	0x4905 2014
GPIO_IRQSTATUS1	RW	32	0x018	0x4831 0018	0x4905 0018	0x4905 2018
GPIO_IRQENABLE1	RW	32	0x01C	0x4831 001C	0x4905 001C	0x4905 201C
GPIO_WAKEUPENABLE	RW	32	0x020	0x4831 0020	0x4905 0020	0x4905 2020
GPIO_IRQSTATUS2	RW	32	0x028	0x4831 0028	0x4905 0028	0x4905 2028
GPIO_IRQENABLE2	RW	32	0x02C	0x4831 002C	0x4905 002C	0x4905 202C
GPIO_CTRL	RW	32	0x030	0x4831 0030	0x4905 0030	0x4905 2030
GPIO_OE	RW	32	0x034	0x4831 0034	0x4905 0034	0x4905 2034
GPIO_DATAIN	R	32	0x038	0x4831 0038	0x4905 0038	0x4905 2038
GPIO_DATAOUT	RW	32	0x03C	0x4831 003C	0x4905 003C	0x4905 203C
GPIO_LEVELDETECT0	RW	32	0x040	0x4831 0040	0x4905 0040	0x4905 2040
GPIO_LEVELDETECT1	RW	32	0x044	0x4831 0044	0x4905 0044	0x4905 2044
GPIO_RISINGDETECT	RW	32	0x048	0x4831 0048	0x4905 0048	0x4905 2048
GPIO_FALLINGDETECT	RW	32	0x04C	0x4831 004C	0x4905 004C	0x4905 204C
GPIO_DEBOUNCENABLE	RW	32	0x050	0x4831 0050	0x4905 0050	0x4905 2050
GPIO_DEBOUNCINGTIME	RW	32	0x054	0x4831 0054	0x4905 0054	0x4905 2054
GPIO_CLEARIRQENABLE 1	RW	32	0x060	0x4831 0060	0x4905 0060	0x4905 2060
GPIO_SETIRQENABLE1	RW	32	0x064	0x4831 0064	0x4905 0064	0x4905 2064
GPIO_CLEARIRQENABLE 2	RW	32	0x070	0x4831 0070	0x4905 0070	0x4905 2070
GPIO_SETIRQENABLE2	RW	32	0x074	0x4831 0074	0x4905 0074	0x4905 2074
GPIO_CLEARWKUENA	RW	32	0x080	0x4831 0080	0x4905 0080	0x4905 2080
GPIO_SETWKUENA	RW	32	0x084	0x4831 0084	0x4905 0084	0x4905 2084
GPIO_CLEARDATAOUT	RW	32	0x090	0x4831 0090	0x4905 0090	0x4905 2090

Table 25-7. GPIO1 to GPIO3 Register Summary (continued)

Register Name	Type	Register Width (Bits)	Address Offset	Physical Address (GPIO1)	Physical Address (GPIO2)	Physical Address (GPIO3)
GPIO_SETDATAOUT	RW	32	0x094	0x4831 0094	0x4905 0094	0x4905 2094

Table 25-8. GPIO4 to GPIO6 Register Summary

Register Name	Type	Register Width (Bits)	Address Offset	Physical Address (GPIO4)	Physical Address (GPIO5)	Physical Address (GPIO6)
GPIO_REVISION	R	32	0x000	0x4905 4000	0x4905 6000	0x4905 8000
GPIO_SYSCONFIG	RW	32	0x010	0x4905 4010	0x4905 6010	0x4905 8010
GPIO_SYSSTATUS	R	32	0x014	0x4905 4014	0x4905 6014	0x4905 8014
GPIO_IRQSTATUS1	RW	32	0x018	0x4905 4018	0x4905 6018	0x4905 8018
GPIO_IRQENABLE1	RW	32	0x01C	0x4905 401C	0x4905 601C	0x4905 801C
GPIO_WAKEUPENABLE	RW	32	0x020	0x4905 4020	0x4905 6020	0x4905 8020
GPIO_IRQSTATUS2	RW	32	0x028	0x4905 4028	0x4905 6028	0x4905 8028
GPIO_IRQENABLE2	RW	32	0x02C	0x4905 402C	0x4905 602C	0x4905 802C
GPIO_CTRL	RW	32	0x030	0x4905 4030	0x4905 6030	0x4905 8030
GPIO_OE	RW	32	0x034	0x4905 4034	0x4905 6034	0x4905 8034
GPIO_DATAIN	R	32	0x038	0x4905 4038	0x4905 6038	0x4905 8038
GPIO_DATAOUT	RW	32	0x03C	0x4905 403C	0x4905 603C	0x4905 803C
GPIO_LEVELDETECT0	RW	32	0x040	0x4905 4040	0x4905 6040	0x4905 8040
GPIO_LEVELDETECT1	RW	32	0x044	0x4905 4044	0x4905 6044	0x4905 8044
GPIO_RISINGDETECT	RW	32	0x048	0x4905 4048	0x4905 6048	0x4905 8048
GPIO_FALLINGDETECT	RW	32	0x04C	0x4905 404C	0x4905 604C	0x4905 804C
GPIO_DEBOUNCENABLE	RW	32	0x050	0x4905 4050	0x4905 6050	0x4905 8050
GPIO_DEBOUNCINGTIME	RW	32	0x054	0x4905 4054	0x4905 6054	0x4905 8054
GPIO_CLEARIRQENABLE 1	RW	32	0x060	0x4905 4060	0x4905 6060	0x4905 8060
GPIO_SETIRQENABLE1	RW	32	0x064	0x4905 4064	0x4905 6064	0x4905 8064
GPIO_CLEARIRQENABLE 2	RW	32	0x070	0x4905 4070	0x4905 6070	0x4905 8070
GPIO_SETIRQENABLE2	RW	32	0x074	0x4905 4074	0x4905 6074	0x4905 8074
GPIO_CLEARWKUENA	RW	32	0x080	0x4905 4080	0x4905 6080	0x4905 8080
GPIO_SETWKUENA	RW	32	0x084	0x4905 4084	0x4905 6084	0x4905 8084
GPIO_CLEARDATAOUT	RW	32	0x090	0x4905 4090	0x4905 6090	0x4905 8090
GPIO_SETDATAOUT	RW	32	0x094	0x4905 4094	0x4905 6094	0x4905 8094

The write latency for all the R/W registers is immediate (with respect to the interface clock)

Note: When two write accesses in the GPIOi.GPIO_DEBOUNCINGTIME register are performed in less than two debounce clock cycles (32 kHz) + four interface clock cycles, the first write access latency is immediate, but the second write access is acknowledged only after this interval ends.

In the register descriptions in this section, when one single register carries an individual configuration or setting for all the channels of the module, one bit in the register is dedicated to each channel. The bit and the corresponding channel are identified with the same number: bit 0 refers to channel 0, bit 1 refers to channel 1, and so on, up to 31.

25.6.2 Register Descriptions

Table 25-9 through Table 25-59 describe the register bits.

Table 25-9. GPIO_REVISION

Address Offset	0x000		
Physical Address	0x4831 0000	**Instance**	GPIO1
	0x4905 0000		GPIO2
	0x4905 2000		GPIO3
	0x4905 4000		GPIO4
	0x4905 6000		GPIO5
	0x4905 8000		GPIO6
Description	This register contains the IP revision code.		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																								GPIOREVISION							

Bits	Field Name	Description	Type	Reset
31:8	RESERVED	Read returns 0	R	0x000000
7:0	GPIOREVISION	IP revision [7:4] Major revision [3:0] Minor revision Examples: 0x10 for 1.0, 0x21 for 2.1	R	See (1)

(1) TI internal data

Table 25-10. Register Call Summary for Register GPIO_REVISION

General-Purpose Interface Register Manual

- General-Purpose Interface Register Mapping Summary: [0] [1]

Table 25-11. GPIO_SYSCONFIG

Address Offset	0x010		
Physical Address	0x4831 0010	**Instance**	GPIO1
	0x4905 0010		GPIO2
	0x4905 2010		GPIO3
	0x4905 4010		GPIO4
	0x4905 6010		GPIO5
	0x4905 8010		GPIO6
Description	This register controls the various parameters of the L4 interconnect.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																											IDLEMODE		ENAWAKEUP	SOFTRESET	AUTOIDLE

Bits	Field Name	Description	Type	Reset
31:5	RESERVED	Write 0s for future compatibility. Read returns 0	RW	0x0000000
4:3	IDLEMODE	Power Management, Req/Ack control 0x0: Force-idle. An idle request is acknowledged unconditionally 0x1: No-idle. An idle request is never acknowledged 0x2: Smart-idle. Acknowledgment to an idle request is given based on the internal activity of the module 0x3: reserved do not use	RW	0x0
2	ENAWAKEUP	Wakeup capability enabled/disabled 0x0: Wakeup disable 0x1: Wakeup enable	RW	0x0
1	SOFTRESET	Software reset. This bit is automatically reset by the hardware. During reads, it always returns 0. 0x0: Normal mode 0x1: The module is reset	RW	0x0
0	AUTOIDLE	Internal interface clock gating strategy 0x0: interface clock is free-running 0x1: Automatic interface clock gating strategy is applied, based on the L4 interconnect activity	RW	0x0

Table 25-12. Register Call Summary for Register GPIO_SYSCONFIG

General-Purpose Interface Integration • Clocking, Reset, and Power-Management Scheme: [0] [1] [2] [3] [4] [5] [6] [7] [8] [9]
General-Purpose Interface Functional Description • Asynchronous Path: Wake-Up Request Generation: [10]
General-Purpose Interface Basic Programming Model • Description: [11]
General-Purpose Interface Register Manual • General-Purpose Interface Register Mapping Summary: [12] [13]

Table 25-13. GPIO_SYSSTATUS

Address Offset	0x014		
Physical Address	0x4831 0014	**Instance**	GPIO1
	0x4905 0014		GPIO2
	0x4905 2014		GPIO3
	0x4905 4014		GPIO4
	0x4905 6014		GPIO5
	0x4905 8014		GPIO6
Description	This register provides status information about the module, excluding the interrupt status information.		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																								RESERVED							RESETDONE

Bits	Field Name	Description	Type	Reset
31:8	RESERVED	Read returns 0	R	0x000000
7:1	RESERVED	Read returns 0	R	0x00
0	RESETDONE	Internal reset monitoring	R	
		0x0: Internal module reset in on-going		
		0x1: Reset completed		

Table 25-14. Register Call Summary for Register GPIO_SYSSTATUS

General-Purpose Interface Integration
• Clocking, Reset, and Power-Management Scheme: [0] [1]
General-Purpose Interface Register Manual
• General-Purpose Interface Register Mapping Summary: [2] [3]

Table 25-15. GPIO_IRQSTATUS1

Address Offset	0x018		
Physical Address	0x4831 0018	**Instance**	GPIO1
	0x4905 0018		GPIO2
	0x4905 2018		GPIO3
	0x4905 4018		GPIO4
	0x4905 6018		GPIO5
	0x4905 8018		GPIO6
Description	This register provides IRQ 1 status information.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IRQSTATUS1																															

Bits	Field Name	Description	Type	Reset
31:0	IRQSTATUS1	Interrupt 1 Status Register. Write a 1 in the corresponding bit to clear it to 0. Write 0 in the corresponding bit does not affect its value.	RW	0x00000000
		0x0: IRQ channel N not triggered		
		0x1: IRQ channel N triggered		

Table 25-16. Register Call Summary for Register GPIO_IRQSTATUS1

General-Purpose Interface Functional Description
• Synchronous Path: Interrupt Request Generation: [0]
• Asynchronous Path: Wake-Up Request Generation: [1] [2]
• Interrupt (or Wake-Up) Line Release: [3]
General-Purpose Interface Basic Programming Model
• Involved Configuration Registers: [4]
• Description: [5]
General-Purpose Interface Register Manual
• General-Purpose Interface Register Mapping Summary: [6] [7]

Table 25-17. GPIO_IRQENABLE1

Address Offset	0x01C		
Physical Address	0x4831 001C	**Instance**	GPIO1
	0x4905 001C		GPIO2
	0x4905 201C		GPIO3
	0x4905 401C		GPIO4
	0x4905 601C		GPIO5
	0x4905 801C		GPIO6
Description	This register provides IRQ 1 enable information.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IRQENABLE1																															

Bits	Field Name	Description	Type	Reset
31:0	IRQENABLE1	Interrupt 1 Enable Register	RW	0x00000000
		0x0: disable IRQ generation for channel N		
		0x1: enable IRQ generation for channel N		

Table 25-18. Register Call Summary for Register GPIO_IRQENABLE1

General-Purpose Interface Overview
- Global Features: [0]

General-Purpose Interface Integration
- Hardware Requests: [1]

General-Purpose Interface Functional Description
- General-Purpose Interface Functional Description: [2]
- Synchronous Path: Interrupt Request Generation: [3]
- Asynchronous Path: Wake-Up Request Generation: [4]

General-Purpose Interface Basic Programming Model
- Description: [5]
- Involved Configuration Registers: [6]
- Description: [7] [8] [9]
- Data Input (Capture)/Output (Drive): [10] [11]

General-Purpose Interface Register Manual
- General-Purpose Interface Register Mapping Summary: [12] [13]
- Register Descriptions: [14] [15] [16] [17]

Table 25-19. GPIO_WAKEUPENABLE

Address Offset	0x020		
Physical Address	0x4831 0020	**Instance**	GPIO1
	0x4905 0020		GPIO2
	0x4905 2020		GPIO3
	0x4905 4020		GPIO4
	0x4905 6020		GPIO5
	0x4905 8020		GPIO6
Description	This register provides wake-up enable information.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
WAKEUPEN																															

Bits	Field Name	Description	Type	Reset
31:0	WAKEUPEN	Wake Up Enable Register	RW	0x00000000
		0x0: disable wakeup generation for channel N		
		0x1: enable wakeup generation for channel N		

Table 25-20. Register Call Summary for Register GPIO_WAKEUPENABLE

General-Purpose Interface Overview
- Global Features: [0]

General-Purpose Interface Integration
- Hardware Requests: [1]

General-Purpose Interface Functional Description
- Asynchronous Path: Wake-Up Request Generation: [2] [3]

General-Purpose Interface Basic Programming Model
- Description: [4]
- Involved Configuration Registers: [5]
- Description: [6]
- Data Input (Capture)/Output (Drive): [7]

General-Purpose Interface Register Manual
- General-Purpose Interface Register Mapping Summary: [8] [9]
- Register Descriptions: [10] [11] [12] [13]

Table 25-21. GPIO_IRQSTATUS2

Address Offset	0x028		
Physical Address	0x4831 0028	**Instance**	GPIO1
	0x4905 0028		GPIO2
	0x4905 2028		GPIO3
	0x4905 4028		GPIO4
	0x4905 6028		GPIO5
	0x4905 8028		GPIO6
Description	This register provides IRQ 2 status information.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IRQSTATUS2																															

Bits	Field Name	Description	Type	Reset
31:0	IRQSTATUS2	Interrupt 2 Status Register. Write a 1 in the corresponding bit to clear it to 0. Write 0 in the corresponding bit does not affect its value. 0x0: IRQ channel N not triggered 0x1: IRQ channel N triggered	RW	0x00000000

Table 25-22. Register Call Summary for Register GPIO_IRQSTATUS2

General-Purpose Interface Functional Description
- Synchronous Path: Interrupt Request Generation: [0]
- Asynchronous Path: Wake-Up Request Generation: [1] [2]
- Interrupt (or Wake-Up) Line Release: [3]

General-Purpose Interface Basic Programming Model
- Involved Configuration Registers: [4]
- Description: [5]

General-Purpose Interface Register Manual
- General-Purpose Interface Register Mapping Summary: [6] [7]

Table 25-23. GPIO_IRQENABLE2

Address Offset	0x02C		
Physical Address	0x4831 002C	**Instance**	GPIO1
	0x4905 002C		GPIO2
	0x4905 202C		GPIO3
	0x4905 402C		GPIO4
	0x4905 602C		GPIO5
	0x4905 802C		GPIO6
Description	This register provides IRQ 2 enable information.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IRQENABLE2																															

Bits	Field Name	Description	Type	Reset
31:0	IRQENABLE2	Interrupt 2 Enable Register 0x0: disable IRQ generation for channel N 0x1: enable IRQ generation for channel N	RW	0x00000000

Table 25-24. Register Call Summary for Register GPIO_IRQENABLE2

General-Purpose Interface Overview
- Global Features: [0]

General-Purpose Interface Integration
- Hardware Requests: [1]

General-Purpose Interface Functional Description
- General-Purpose Interface Functional Description: [2]
- Synchronous Path: Interrupt Request Generation: [3]
- Asynchronous Path: Wake-Up Request Generation: [4]

General-Purpose Interface Basic Programming Model
- Description: [5]
- Involved Configuration Registers: [6]
- Description: [7] [8] [9]
- Data Input (Capture)/Output (Drive): [10] [11]

Table 25-24. Register Call Summary for Register GPIO_IRQENABLE2 (continued)

General-Purpose Interface Register Manual

- General-Purpose Interface Register Mapping Summary: [12] [13]
- Register Descriptions: [14] [15] [16] [17]

Table 25-25. GPIO_CTRL

Address Offset	0x030		
Physical Address	0x4831 0030	**Instance**	GPIO1
	0x4905 0030		GPIO2
	0x4905 2030		GPIO3
	0x4905 4030		GPIO4
	0x4905 6030		GPIO5
	0x4905 8030		GPIO6
Description	This register controls the clock gating functionality.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																													GATINGRATIO		DISABLEMODULE

Bits	Field Name	Description	Type	Reset
31:3	RESERVED	Read returns 0	RW	0x00000000
2:1	GATINGRATIO	Gating Ratio	RW	0x1
		0x0: Functional clock is interface clock.		
		0x1: Functional clock is interface clock divided by 2.		
		0x2: Functional clock is interface clock divided by 4.		
		0x3: Functional clock is interface clock divided by 8.		
0	DISABLEMODULE	Module Disable	RW	0x0
		0x0: Module is enabled, clocks are not gated		
		0x1: Module is disabled, clocks are gated		

Table 25-26. Register Call Summary for Register GPIO_CTRL

General-Purpose Interface Integration

- Clocking, Reset, and Power-Management Scheme: [0] [1] [2]

General-Purpose Interface Register Manual

- General-Purpose Interface Register Mapping Summary: [3] [4]

Table 25-27. GPIO_OE

Address Offset	0x034		
Physical Address	0x4831 0034	**Instance**	GPIO1
	0x4905 0034		GPIO2
	0x4905 2034		GPIO3
	0x4905 4034		GPIO4
	0x4905 6034		GPIO5
	0x4905 8034		GPIO6
Description	This register is used to enable the pins output capabilities. Its only function is to carry the pads configuration.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
OUTPUTEN																															

Bits	Field Name	Description	Type	Reset
31:0	OUTPUTEN	Output Data Enable	RW	0xFFFFFFFF
		0x0: The corresponding GPIO port is configured as output		
		0x1: The corresponding GPIO port is configured as input		

Table 25-28. Register Call Summary for Register GPIO_OE

General-Purpose Interface Overview • Global Features: [0]
General-Purpose Interface Basic Programming Model • Description: [1] [2] • Data Input (Capture)/Output (Drive): [3] [4] [5] • Debouncing Time: [6]
General-Purpose Interface Register Manual • General-Purpose Interface Register Mapping Summary: [7] [8]

Table 25-29. GPIO_DATAIN

Address Offset	0x038		
Physical Address	0x4831 0038	**Instance**	GPIO1
	0x4905 0038		GPIO2
	0x4905 2038		GPIO3
	0x4905 4038		GPIO4
	0x4905 6038		GPIO5
	0x4905 8038		GPIO6
Description	This register is used to register the data that is read from the GPIO pins.		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DATAINPUT																															

Bits	Field Name	Description	Type	Reset
31:0	DATAINPUT	Sampled Input Data	R	0x00000000

Table 25-30. Register Call Summary for Register GPIO_DATAIN

General-Purpose Interface Overview • Global Features: [0]

Table 25-30. Register Call Summary for Register GPIO_DATAIN (continued)

General-Purpose Interface Integration • Clocking, Reset, and Power-Management Scheme: [1]
General-Purpose Interface Basic Programming Model • Data Input (Capture)/Output (Drive): [2]
General-Purpose Interface Register Manual • General-Purpose Interface Register Mapping Summary: [3] [4]

Table 25-31. GPIO_DATAOUT

Address Offset	0x03C		
Physical Address	0x4831 003C	**Instance**	GPIO1
	0x4905 003C		GPIO2
	0x4905 203C		GPIO3
	0x4905 403C		GPIO4
	0x4905 603C		GPIO5
	0x4905 803C		GPIO6
Description	This register is used for setting the value of the GPIO output pins		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DATAOUTPUT																															

Bits	Field Name	Description	Type	Reset
31:0	DATAOUTPUT	Output Data	RW	0x00000000

Table 25-32. Register Call Summary for Register GPIO_DATAOUT

General-Purpose Interface Overview • Global Features: [0] [1]
General-Purpose Interface Basic Programming Model • Description: [2] • Data Input (Capture)/Output (Drive): [3]
General-Purpose Interface Register Manual • General-Purpose Interface Register Mapping Summary: [4] [5] • Register Descriptions: [6] [7] [8] [9]

Table 25-33. GPIO_LEVELDETECT0

Address Offset	0x040		
Physical Address	0x4831 0040	**Instance**	GPIO1
	0x4905 0040		GPIO2
	0x4905 2040		GPIO3
	0x4905 4040		GPIO4
	0x4905 6040		GPIO5
	0x4905 8040		GPIO6
Description	This register is used to enable/disable for each input lines the low-level (0) detection to be used for the interrupt request generation.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
LOWLEVEL																															

Bits	Field Name	Description	Type	Reset
31:0	LOWLEVEL	Low Level Interrupt Enable 0x0: disable the IRQ assertion on low level detect 0x1: enable the IRQ assertion on low level detect	RW	0x00000000

Table 25-34. Register Call Summary for Register GPIO_LEVELDETECT0

General-Purpose Interface Basic Programming Model

- Power Saving by Grouping the Edge/Level Detection: [0] [1]
- Description: [2]

General-Purpose Interface Register Manual

- General-Purpose Interface Register Mapping Summary: [3] [4]

Table 25-35. GPIO_LEVELDETECT1

Address Offset	0x044		
Physical Address	0x4831 0044	**Instance**	GPIO1
	0x4905 0044		GPIO2
	0x4905 2044		GPIO3
	0x4905 4044		GPIO4
	0x4905 6044		GPIO5
	0x4905 8044		GPIO6
Description	This register is used to enable/disable for each input lines the high-level (1) detection to be used for the interrupt request generation.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
HIGHLEVEL																															

Bits	Field Name	Description	Type	Reset
31:0	HIGHLEVEL	High Level Interrupt Enable 0x0: disable the IRQ assertion on high level detect 0x1: enable the IRQ assertion on high level detect	RW	0x00000000

Table 25-36. Register Call Summary for Register GPIO_LEVELDETECT1

General-Purpose Interface Basic Programming Model

- Power Saving by Grouping the Edge/Level Detection: [0] [1]
- Description: [2]

Table 25-36. Register Call Summary for Register GPIO_LEVELDETECT1 (continued)

General-Purpose Interface Register Manual • General-Purpose Interface Register Mapping Summary: [3] [4]

Table 25-37. GPIO_RISINGDETECT

Address Offset	0x048		
Physical Address	0x4831 0048	**Instance**	GPIO1
	0x4905 0048		GPIO2
	0x4905 2048		GPIO3
	0x4905 4048		GPIO4
	0x4905 6048		GPIO5
	0x4905 8048		GPIO6
Description	This register is used to enable/disable for each input lines the rising-edge (transition 0=>1) detection to be used for the interrupt request and the wake-up generation.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RISINGEDGE																															

Bits	Field Name	Description	Type	Reset
31:0	RISINGEDGE	Rising Edge Interrupt/Wakeup Enable 0x0: disable IRQ/Wakeup on rising edge detect 0x1: enable IRQ/Wakeup on rising edge detect	RW	0x00000000

Table 25-38. Register Call Summary for Register GPIO_RISINGDETECT

General-Purpose Interface Basic Programming Model • Power Saving by Grouping the Edge/Level Detection: [0] [1] • Description: [2] [3] [4]
General-Purpose Interface Register Manual • General-Purpose Interface Register Mapping Summary: [5] [6]

Table 25-39. GPIO_FALLINGDETECT

Address Offset	0x04C		
Physical Address	0x4831 004C	**Instance**	GPIO1
	0x4905 004C		GPIO2
	0x4905 204C		GPIO3
	0x4905 404C		GPIO4
	0x4905 604C		GPIO5
	0x4905 804C		GPIO6
Description	This register is used to enable/disable for each input lines the falling-edge (transition 1=>0) detection to be used for the interrupt request and the wake-up generation.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FALLINGEDGE																															

Bits	Field Name	Description	Type	Reset
31:0	FALLINGEDGE	Falling Edge Interrupt/Wakeup Enable 0x0: disable IRQ/Wakeup on falling edge detect 0x1: enable IRQ/Wakeup on falling edge detect	RW	0x00000000

Table 25-40. Register Call Summary for Register GPIO_FALLINGDETECT

General-Purpose Interface Basic Programming Model

- Power Saving by Grouping the Edge/Level Detection: [0] [1]
- Description: [2] [3] [4]

General-Purpose Interface Register Manual

- General-Purpose Interface Register Mapping Summary: [5] [6]

Table 25-41. GPIO_DEBOUNCENABLE

Address Offset	0x050		
Physical Address	0x4831 0050	**Instance**	GPIO1
	0x4905 0050		GPIO2
	0x4905 2050		GPIO3
	0x4905 4050		GPIO4
	0x4905 6050		GPIO5
	0x4905 8050		GPIO6
Description	This register is used to enable/disable the debouncing feature for each input line.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DEBOUNCEEN																															

Bits	Field Name	Description	Type	Reset
31:0	DEBOUNCEEN	Input Debounce Enable 0x0: disable debouncing feature on the corresponding input port 0x1: enable debouncing feature on the corresponding input port	RW	0x00000000

Table 25-42. Register Call Summary for Register GPIO_DEBOUNCENABLE

General-Purpose Interface Basic Programming Model

- Debouncing Time: [0]

General-Purpose Interface Register Manual

- General-Purpose Interface Register Mapping Summary: [1] [2]

Table 25-43. GPIO_DEBOUNCINGTIME

Address Offset	0x054		
Physical Address	0x4831 0054	**Instance**	GPIO1
	0x4905 0054		GPIO2
	0x4905 2054		GPIO3
	0x4905 4054		GPIO4
	0x4905 6054		GPIO5
	0x4905 8054		GPIO6
Description	This register controls debouncing time (the value is global for all ports).		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																								DEBOUNCEVAL							

Bits	Field Name	Description	Type	Reset
31:8	RESERVED	Read returns 0	RW	0x000000
7:0	DEBOUNCEVAL	Input Debouncing Value in 31 microsecond steps. debouncing time = (DEBOUNCEVAL+1) x 31 s	RW	0x00

Table 25-44. Register Call Summary for Register GPIO_DEBOUNCINGTIME

General-Purpose Interface Integration • Clocking, Reset, and Power-Management Scheme: [0]
General-Purpose Interface Functional Description • Synchronous Path: Interrupt Request Generation: [1] [2]
General-Purpose Interface Basic Programming Model • Debouncing Time: [3] [4] [5] [6]
General-Purpose Interface Register Manual • General-Purpose Interface Register Mapping Summary: [7] [8] [9]

Table 25-45. GPIO_CLEARIRQENABLE1

Address Offset	0x060		
Physical Address	0x4831 0060	**Instance**	GPIO1
	0x4905 0060		GPIO2
	0x4905 2060		GPIO3
	0x4905 4060		GPIO4
	0x4905 6060		GPIO5
	0x4905 8060		GPIO6
Description	Clear to 0 the corresponding bits in the GPIO_IRQENABLE1 register		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CLEARIRQEN1																															

Bits	Field Name	Description	Type	Reset
31:0	CLEARIRQEN1	Clear Interrupt Enable 1 0x0: no effect 0x1: Clear the corresponding bit in the GPIO_IRQENABLE1 register	RW	0x00000000

Table 25-46. Register Call Summary for Register GPIO_CLEARIRQENABLE1

General-Purpose Interface Basic Programming Model • Clear Instruction: [0]
General-Purpose Interface Register Manual • General-Purpose Interface Register Mapping Summary: [1] [2]

Table 25-47. GPIO_SETIRQENABLE1

Address Offset	0x064		
Physical Address	0x4831 0064	**Instance**	GPIO1
	0x4905 0064		GPIO2
	0x4905 2064		GPIO3
	0x4905 4064		GPIO4
	0x4905 6064		GPIO5
	0x4905 8064		GPIO6
Description	Set to 1 the corresponding bits in the GPIO_IRQENABLE1 register		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SETIRQEN1																															

Bits	Field Name	Description	Type	Reset
31:0	SETIRQEN1	Set Interrupt Enable 1	RW	0x00000000
		0x0: no effect		
		0x1: Set the corresponding bit in the GPIO_IRQENABLE1 register		

Table 25-48. Register Call Summary for Register GPIO_SETIRQENABLE1

General-Purpose Interface Basic Programming Model
- Set Instruction: [0]

General-Purpose Interface Register Manual
- General-Purpose Interface Register Mapping Summary: [1] [2]

Table 25-49. GPIO_CLEARIRQENABLE2

Address Offset	0x070		
Physical Address	0x4831 0070	**Instance**	GPIO1
	0x4905 0070		GPIO2
	0x4905 2070		GPIO3
	0x4905 4070		GPIO4
	0x4905 6070		GPIO5
	0x4905 8070		GPIO6
Description	Clear to 0 the corresponding bits in the GPIO_IRQENABLE2 register		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CLEARIRQEN2																															

Bits	Field Name	Description	Type	Reset
31:0	CLEARIRQEN2	Clear Interrupt Enable 2	RW	0x00000000
		0x0: no effect		
		0x1: Clear the corresponding bit in the GPIO_IRQENABLE2 register		

Table 25-50. Register Call Summary for Register GPIO_CLEARIRQENABLE2

General-Purpose Interface Basic Programming Model
- Clear Instruction: [0]

General-Purpose Interface Register Manual
- General-Purpose Interface Register Mapping Summary: [1] [2]

Table 25-51. GPIO_SETIRQENABLE2

Address Offset	0x074		
Physical Address	0x4831 0074	**Instance**	GPIO1
	0x4905 0074		GPIO2
	0x4905 2074		GPIO3
	0x4905 4074		GPIO4
	0x4905 6074		GPIO5
	0x4905 8074		GPIO6
Description	Set to 1 the corresponding bits in the GPIO_IRQENABLE2 register		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SETIRQEN2																															

Bits	Field Name	Description	Type	Reset
31:0	SETIRQEN2	Set Interrupt Enable 2 0x0: no effect 0x1: Set the corresponding bit in the GPIO_IRQENABLE2 register	RW	0x00000000

Table 25-52. Register Call Summary for Register GPIO_SETIRQENABLE2

General-Purpose Interface Basic Programming Model
- Set Instruction: [0]

General-Purpose Interface Register Manual
- General-Purpose Interface Register Mapping Summary: [1] [2]

Table 25-53. GPIO_CLEARWKUENA

Address Offset	0x080		
Physical Address	0x4831 0080	**Instance**	GPIO1
	0x4905 0080		GPIO2
	0x4905 2080		GPIO3
	0x4905 4080		GPIO4
	0x4905 6080		GPIO5
	0x4905 8080		GPIO6
Description	Clear to 0 the corresponding bits in the GPIO_WAKEUPENABLE register		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CLEARWAKEUPEN																															

Bits	Field Name	Description	Type	Reset
31:0	CLEARWAKEUPEN	Clear Wakeup Enable 0x0: no effect 0x1: Clear the corresponding bit in the GPIO_WAKEUPENABLE register	RW	0x00000000

Table 25-54. Register Call Summary for Register GPIO_CLEARWKUENA

General-Purpose Interface Basic Programming Model
- Clear Instruction: [0]

Table 25-54. Register Call Summary for Register GPIO_CLEARWKUENA (continued)

General-Purpose Interface Register Manual • General-Purpose Interface Register Mapping Summary: [1] [2]

Table 25-55. GPIO_SETWKUENA

Address Offset	0x084		
Physical Address	0x4831 0084	**Instance**	GPIO1
	0x4905 0084		GPIO2
	0x4905 2084		GPIO3
	0x4905 4084		GPIO4
	0x4905 6084		GPIO5
	0x4905 8084		GPIO6
Description	Set to 1 the corresponding bits in the GPIO_WAKEUPENABLE register		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SETWAKEUPEN																															

Bits	Field Name	Description	Type	Reset
31:0	SETWAKEUPEN	Set Wakeup Enable 0x0: no effect 0x1: Set the corresponding bit in the GPIO_WAKEUPENABLE register	RW	0x00000000

Table 25-56. Register Call Summary for Register GPIO_SETWKUENA

General-Purpose Interface Basic Programming Model • Set Instruction: [0]
General-Purpose Interface Register Manual • General-Purpose Interface Register Mapping Summary: [1] [2]

Table 25-57. GPIO_CLEARDATAOUT

Address Offset	0x090		
Physical Address	0x4831 0090	**Instance**	GPIO1
	0x4905 0090		GPIO2
	0x4905 2090		GPIO3
	0x4905 4090		GPIO4
	0x4905 6090		GPIO5
	0x4905 8090		GPIO6
Description	Clear to 0 the corresponding bits in the GPIO_DATAOUT register		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CLEARDATAOUT																															

Bits	Field Name	Description	Type	Reset
31:0	CLEARDATAOUT	Clear Data Output Register 0x0: no effect 0x1: Clear the corresponding bit in the GPIO_DATAOUT register	RW	0x00000000

Table 25-58. Register Call Summary for Register GPIO_CLEARDATAOUT

General-Purpose Interface Basic Programming Model

- Clear Instruction: [0]
- Data Input (Capture)/Output (Drive): [1]

General-Purpose Interface Register Manual

- General-Purpose Interface Register Mapping Summary: [2] [3]

Table 25-59. GPIO_SETDATAOUT

Address Offset	0x094		
Physical Address	0x4831 0094	**Instance**	GPIO1
	0x4905 0094		GPIO2
	0x4905 2094		GPIO3
	0x4905 4094		GPIO4
	0x4905 6094		GPIO5
	0x4905 8094		GPIO6
Description	Set to 1 the corresponding bits in the GPIO_DATAOUT register		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SETDATAOUT																															

Bits	Field Name	Description	Type	Reset
31:0	SETDATAOUT	Set Data Output Register	RW	0x00000000
		0x0: no effect		
		0x1: Set the corresponding bit in the GPIO_DATAOUT register		

Table 25-60. Register Call Summary for Register GPIO_SETDATAOUT

General-Purpose Interface Basic Programming Model

- Set Instruction: [0]
- Data Input (Capture)/Output (Drive): [1]

General-Purpose Interface Register Manual

- General-Purpose Interface Register Mapping Summary: [2] [3]

Initialization

This chapter introduces the OMAP34xx multimedia general-purpose device initialization.

Topic **Page**

26.1 Initialization Overview 3368
26.2 Preinitialization 3369
26.3 Power, Clocks, and Reset Power-Up Sequence 3378
26.4 Device Initialization by ROM Code 3378
26.5 Wake-Up Booting by ROM Code 3432
26.6 Debug Configuration 3436

26.1 Initialization Overview

This chapter provides an overview of the requirements for initializing the OMAP34xx device from power on to firmware execution. An overview of the overall initialization process is given, including hardware- and software-related steps, a general overview of the boot ROM code operational requirements, and behavior expectations.

26.1.1 Terminology

- Bootstrap: Initial software (SW) launched by the ROM code during the memory booting phase
- Downloaded software: Initial software downloaded into the internal static RAM (SRAM) by the ROM code during the peripheral booting phase
- eFuse: A one-time programmable memory location usually set at the factory
- Flash loader: Downloaded software launched by the ROM code in preflashing. It also programs an image in external memories.
- GP device: General-purpose device
- Initial software: Software executed by any of the ROM code mechanisms (memory booting or peripheral booting). Initial software is a generic term for bootstrap and downloaded software.
- Memory booting: ROM code mechanism that consists of executing initial software from external memory
- Peripheral booting: ROM code mechanism that consists of polling selected interfaces, downloading, and executing initial software (in this case, downloaded software) in the internal RAM.
- Permanent booting device: Memory device containing, by default, the image to be executed during the booting sequence. It is the default memory booting device. The permanent booting device is used after warm reset if no software booting configuration is programmed.
- Preflashing: A specific case of peripheral booting where the ROM code mechanism is used to program the external flash memory
- ROM code: The on-chip software in OMAP™ ROM that implements booting

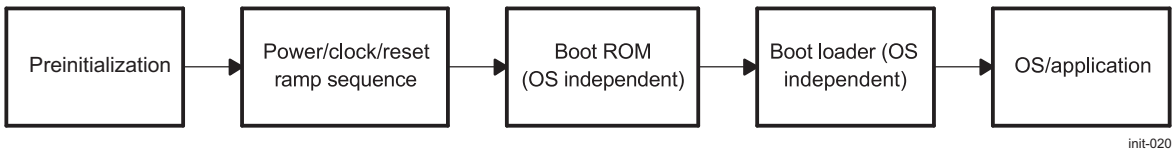
26.1.2 Initialization Process

Figure 26-1 is an overview of the initialization process and its steps:

- Preinitialization
- Power/clock/reset ramp sequence
- Boot ROM
- Boot loader
- OS/application

Each step, up to OS/applications running, is explained in the following sections.

Figure 26-1. Initialization Process



The first two steps in the initialization process are hardware-oriented; however, they require understanding of the process of configuring those system interface pins (balls on the device) that have software-configurable functionality. This configuration is an essential part of chip configuration and is application-dependent. This chapter refers to those pins and the associated configuration registers that are vital for correct device initialization.

26.2 Preinitialization

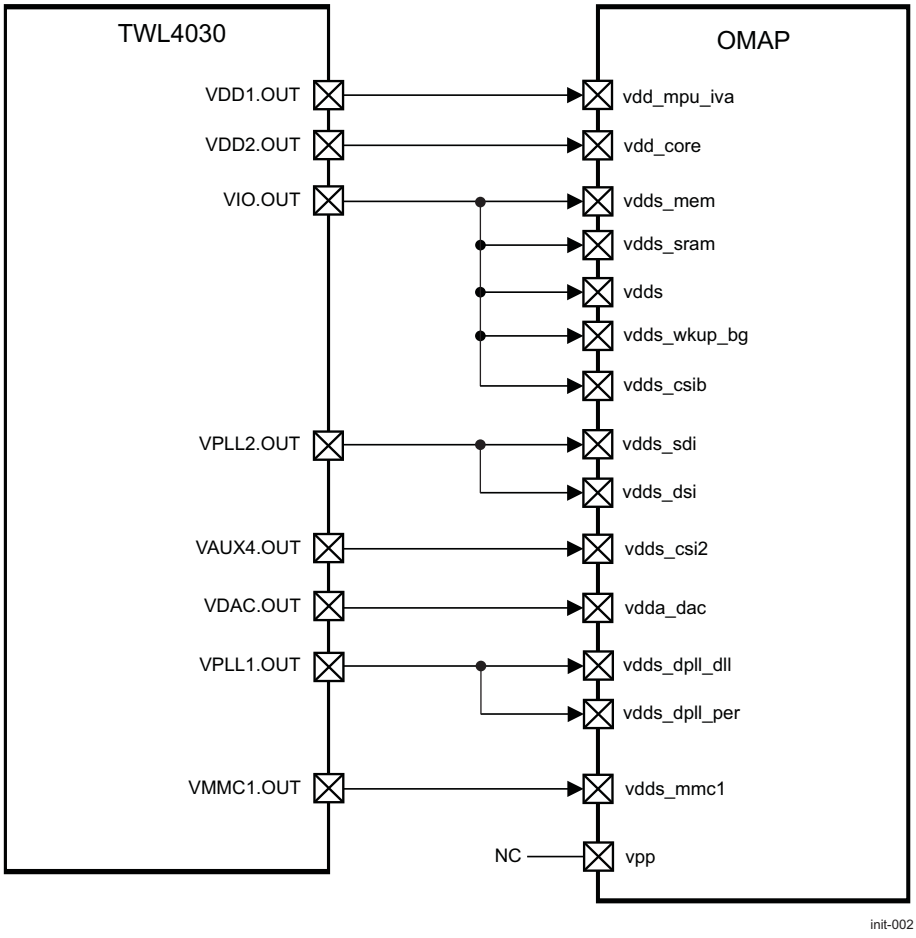
To accomplish a successful boot-up operation with a general-purpose (GP) device, certain hardware configuration settings must be in place. Clock, reset, and power connections, as well as pins involved in setting the boot memory space for the microprocessor unit (MPU), must be connected and driven correctly for successful device initialization. The following sections describe the specific requirements for the preinitialization stage.

26.2.1 Power Connections

The device can be supplied by an external power integrated circuit (IC). TI provides a global solution with the OMAP34xx connected to the TWL4030 power IC.

Figure 26-2 shows the power connections between the OMAP34xx and the TWL4030 power IC.

Figure 26-2. OMAP34xx and TWL4030 Power Connections



Note: Figure 26-2 is an example of power connections between the OMAP3430 device and the TWL4030. These connections depend on the application: For example, if a serial display or camera interface is used, it must be supplied by the low-dropout (LDO) regulator (low noise).

Notes:

- The sys_clkout output clock cannot be provided to peripherals when the core is off.
- The sys_clkreq output signal switches the system clock on or off.
- After power-on reset (POR), the hardware configuration enables the internal oscillator (assuming that the input clock comes from a crystal). The decision whether to bypass the internal oscillator is controlled by the polarity of the sys_boot[6] pin.

Table 26-1 describes the power pins.

Table 26-1. Power Pins(1)

OMAP34xx Input/Output (I/O) Voltage Name	Description
vdd_mpu_iva	MPU and image and video accelerator (IVA) subsystem power supply
vdd_core	Core power supply
vdds	I/O power supply
vdds_mem	Memory I/O power supply
vdds_wkup_bg	Wake-up LDO and VDDA (2LDO SRAM and ball grid [BG]) power supply
vdds_sram	SRAM LDO power supply
vdds_dpll_dll	Power supply for three phase-locked loop (3PLL) (1.8 V)
vdds_dpll_per	Peripheral digital phase-locked loop (DPLL) power supply
vdds_sdi	Dedicated power supply for serial display interface (SDI) I/O cell
vdds_csi2	Dedicated power supply for CSI2 Complex I/O
vdds_csib	Dedicated power supply for CSIb Complex I/O
vdds_dsi	1.8-V analog power supply
vdda_dac	Video digital-to-analog converter (DAC) power supply
vdds_mmc1	Power supply for multimedia card (MMC) and secure digital (SD) interfaces
vpp	eFuse programming

(1) The OMAP34xx package-on-package (POP) device provides feedthroughs from the bottom of the package to the POP interface. Among these feedthroughs (FEEDTHROUGH balls), several provide power to the top memory device. The correct power supply to the feedthroughs must be provided based on memory requirements.

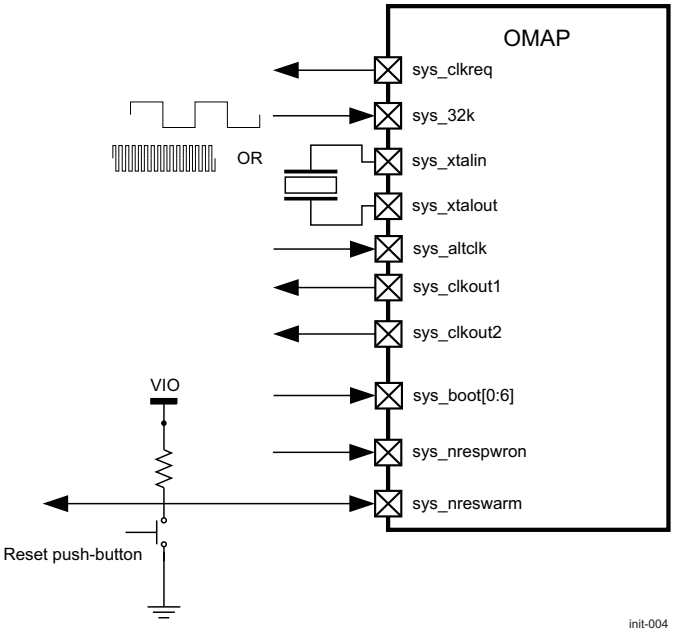
For more information about power management, see Chapter 4, *Power, Reset, and Clock Management.*

26.2.2 Clock and Reset

26.2.2.1 Clock and Reset Overview

Figure 26-3 shows the clock and reset environment that gathers the clocks and reset signals related at the system level, as well as system expansion signals.

Figure 26-3. Clock and Reset Environment



The main features of the system interface are:

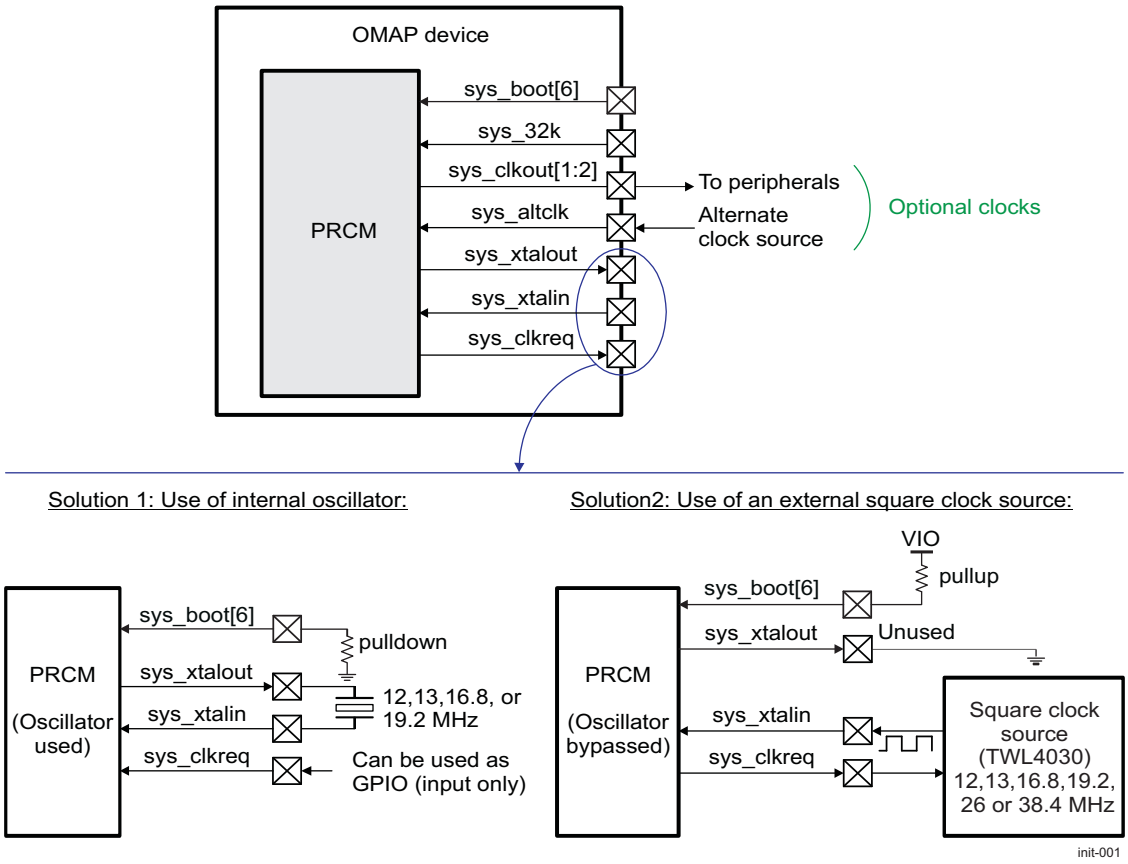
- A clock request output to an external square clock source
- 12, 13, 16.8, 19.2, 26, or 38.4-MHz reference clock input from the external crystal oscillator (only 12, 13, 16.8, or 19.2 MHz) or the digital clock input (all frequencies)
- 32-kHz CMOS clock input
- An additional clock input up to 54 MHz
- Two configurable output clocks
- Seven input signals to define the boot mode
- Two reset sources
  - POR (cold reset)
  - Bidirectional warm reset

26.2.2.2 Clock Configuration

26.2.2.2.1 Required System Input Clocks

Figure 26-4 shows the clock interface.

Figure 26-4. Clock Interface



Notes:

- The sys_clkout output clock cannot be provided to peripherals when the core is off.
- The sys_clkreq output signal switches the system clock on or off.
- After POR, the hardware configuration enables the internal oscillator (assuming that the input clock comes from a crystal). The decision whether to bypass the internal oscillator is controlled by the polarity of the sys_boot[6] pin.

The device operation requires two external input clocks, as follows:

- sys_32k: The 32-kHz frequency is used for low-frequency operation.
- sys_xtal (in/out): The system clock is the main clock source of the device. The system clock input can be connected in either of two ways:
  - Crystal quartz through the sys_xtalin and sys_xtalout pins, using an internal oscillator up to 19.2 MHz
  - Complementary metal oxide semiconductor (CMOS) digital clock (square clock) through the sys_xtalin pin. The oscillator is bypassed.

Table 26-2 lists the mapping for input sources.

Table 26-2. Mapping for Input Sources

Input Source	Mapping	Frequencies	Comment
Crystal quartz	sys_xtalin and sys_xtalout	12, 13, 16.8, or 19.2 MHz	Requires use of internal oscillator
Square clock (1.8 CMOS signal)	sys_xtalin (sys_xtalout unconnected)	12, 13, 16.8, 19.2, 26, or 38.4 MHz	Internal oscillator is bypassed.

Because sys_32k, sys_xtalin, and sys_xtalout have permanently assigned pin locations with no pad configuration for these pins through a system control module (SCM) register, only one source input at a time can be used.

26.2.2.2.2 Optional System Input Clock: sys_altclk

An additional clock can be provided through the sys_altclk input pin to supply internal peripherals, PLLs, a precise clock for National Television System Committee (NTSC) standard (54 MHz), and a universal serial bus (USB) full-speed (FS) controller (48 MHz). If not used as a clock input, this pin can be configured as a general-purpose input/output (GPIO), using the SCM CONTROL.CONTROL_PADCONF_I2C3_SDA register. As an example, to use the sys_altclk pin, the CONTROL.CONTROL_PADCONF_I2C3_SDA[18:16] MUXMODE1 bit field must be set to 0x01.

26.2.2.2.3 Optional System Output Clock: sys_clkout1 and sys_clkout2

Two output clocks (sys_clkout1 and sys_clkout2 pins) are available:

- sys_clkout1 can output the oscillator clock. Its OFF state polarity is programmable.
- sys_clkout2 can output the system clock (12, 13, 16.8, 19.2, 26, or 38.4 MHz), the core clock (CORE DPLL output), 96 MHz, or 54 MHz. sys_clkout2 can be divided by 2, 4, 8, or 16, and its OFF state polarity is programmable.

CAUTION

Clock configurations depend on core voltage, and maximum clock frequencies may not be applicable to production.

The clocks can be managed by software with the appropriate register in the power, reset, clock, management (PRCM) module:

- sys_clkout1 is managed using PRCM.PRM_CLKOUT_CTRL and PRCM.PRM_POLCTRL[2].
- sys_clkout2 is managed using PRCM.CM_CLKOUT_CTRL and PRCM.CM_CLKSEL1_PLL[28:27].

26.2.2.3 Reset Configuration

The sys_nrespwron reset pin resets the entire chip during POR.

The sys_nreswarm reset pin resets the entire chip when the device is supplied and operating, except for the following:

- Part of synchronous dynamic RAM (SDRAM) controller (SDRC)
- Part of IVA
- Part of PRCM
- Control module
- Watchdog
- 32-kHz synchronization timer
- DPLLs

CAUTION

sys_nrespwron must be driven low during a power-up sequence.

sys_nreswarm is a bidirectional reset. When an internal reset occurs, sys_nreswarm goes low and resets all the peripherals. The sys_nreswarm output is open-drain; consequently, an external pullup resistor is required.

sys_nrespwron and sys_nreswarm have permanently assigned pin locations.

At reset, the cause of reset is stored in the PRCM.RM_RSTST_MPU register and used by the boot ROM code.

26.2.3 Boot Configuration

Six external pins (sys_boot[5:0]) are used to select interfaces or devices for booting. The sys_boot[6] pin is used to select whether the internal oscillator is bypassed.

These seven pins are sampled and latched onto the CONTROL.CONTROL_STATUS register after POR. After booting, these pins can be used for other functions and the associated CONTROL.CONTROL_STATUS bits are not updated by the new functionality. For more information about pad multiplexing configuration, see Chapter 7, *System Control Module*.

Note: If used as GPIOs, these pins must be used in output mode. To ensure that the sys_boot[6:0] input values are selected by the pullup and pulldown on sys_boot[6:0] pins at POR, these GPIOs must be used only in output mode. Because these balls have no pullup or pulldown capacity, care must be taken when choosing to use these GPIOs.

Table 26-3, Table 26-4, and Table 26-5 are decoding tables for sys_boot pins. Depending on sys_boot pin configuration during the reset (leftmost column), the ROM code tries to boot on the first device listed. If the boot fails on this first device, the ROM code tries the second device, then the third, then the fourth.

The following names are used in the tables:

- Memory types:
  - Execute in place (XIP): XIP memory without wait monitoring enabled (NOR flash memories)
  - XIP wait: XIP memory with wait monitoring
  - DOC: DiskOnChip™ memory (H3 device types)
  - NAND: NAND flash memories (non-XIP)
  - OneNAND: OneNAND flash memories and Flex-OneNAND™ flash memories
  - MMC1: MMC or SD flash cards with active primary partition of type file allocation table (FAT)12/16/32 connected to the first MMC/SD/secure digital input/output (SDIO)1 card interface
  - MMC2: MMC or SD flash cards with active primary partition of type FAT12/16/32 connected to the second MMC/SD/SDIO2 card interface
- Peripheral interfaces:
  - USB: High-speed (HS) USB
  - UART3: UART interface
- Permanent booting devices are in *italics*.
  Table 26-3 and Table 26-4 list the sys_boot pin configuration after a POR.

CAUTION

– • UART boot: UART3 is the only possible UART from which boot can be performed. Additionally, only UART3 pads that provide UART3 functionality in their MUXMODE 0 can be used. No other UART configuration allows booting from the UART.
– • HS USB boot: If TWL4030 or TWL5030 processor is used as the USB transceiver, it must be connected to the OMAP3430 through I2C1. No other I^2C interface allows the necessary TWL processor configuration.

Table 26-3. Memory Booting Configuration Pins After POR

sys_boot [4:0]	Booting Sequence When SYS.BOOT[5] = 0				
	Memory Booting Preferred Order				
	First	Second	Third	Fourth	Fifth
0b00000			Reserved[1]		
0b00001					
0b00010					
0b00011					
0b00100	*OneNAND*	USB			
0b00101	*MMC2*	USB			
0b00110	*MMC1*	USB			
0b00111			Reserved[1]		
0b01000					
0b01001					
0b01010					
0b01011					
0b01100					
0b01101	*XIP*	USB	UART3	MMC1	
0b01110	*XIPwait*	*DOC*	USB	UART3	MMC1
0b01111	*NAND*	USB	UART3	MMC1	
0b10000	*OneNAND*	USB	UART3	MMC1	
0b10001	*MMC2*	USB	UART3	MMC1	
0b10010	*MMC1*	USB	UART3		
0b10011	*XIP*	UART3			
0b10100	*XIPwait*	*DOC*	UART3		
0b10101	*NAND*	UART3			
0b10110	*OneNAND*	UART3			
0b10111	*MMC2*	UART3			
0b11000	*MMC1*	UART3			
0b11001	*XIP*	USB			
0b11010	*XIPwait*	*DOC*	USB		
0b11011	*NAND*	USB			
0b11100			Reserved[1]		
0b11101					
0b11110					
0b11111	Fast XIP booting. Wait monitoring OFF (only for GP devices)	USB (only on GP devices)	UART3 (only on GP devices)		

(1) Must not be selected

Table 26-4. Peripheral Booting Configuration Pins After POR

sys_boot [4:0]	Booting Sequence When SYS.BOOT[5] = 1 Peripheral Booting Preferred Order				
	First	Second	Third	Fourth	Fifth
0b00000	Reserved[1]				
0b00001					
0b00010					
0b00011					
0b00100	USB	*OneNAND*			
0b00101	USB	*MMC2*			
0b00110	USB	*MMC1*			
0b00111	Reserved[1]				
0b01000					
0b01001					
0b01010					
0b01011					
0b01100					
0b01101	USB	UART3	MMC1	*XIP*	
0b01110	USB	UART3	MMC1	*XIPwait*	*DOC*
0b01111	USB	UART3	MMC1	*NAND*	
0b10000	USB	UART3	MMC1	*OneNAND*	
0b10001	USB	UART3	MMC1	*MMC2*	
0b10010	USB	UART3	MMC1		
0b10011	UART3	*XIP*			
0b10100	UART3	*XIPwait*	*DOC*		
0b10101	UART3	*NAND*			
0b10110	UART3	*OneNAND*			
0b10111	UART3	*MMC2*			
0b11000	UART3	*MMC1*			
0b11001	USB	*XIP*			
0b11010	USB	*XIPwait*	*DOC*		
0b11011	USB	*NAND*			
0b11100	Reserved[1]				
0b11101					
0b11110					
0b11111	Fast XIP booting. Wait monitoring ON (only for GP devices)	USB (only on GP devices)	UART3 (only on GP devices)		

(1) Must not be selected

Table 26-5 shows the sys_boot pin configuration after a warm reset.

Table 26-5. Booting Configuration Pins After a Warm Reset

sys_boot[4:0]	Booting Sequence When SYS.BOOT[5] = 0		Booting Sequence When SYS.BOOT[5] = 1	
	Memory Booting Preferred Order		Peripheral Booting Preferred Order	
	First	Second	First	Second
0b00000	*OneNAND*		OneNAND	
0b00001	*NAND*		NAND	
0b00010	*OneNAND*		OneNAND	
0b00011	*MMC2*		MMC2	
0b00100	*OneNAND*		*OneNAND*	
0b00101	*MMC2*		*MMC2*	
0b00110	*MMC1*		*MMC1*	
0b00111	*XIP*		XIP	
0b01000	*XIPwait*	*DOC*	XIPwait	*DOC*
0b01001	*MMC2*		MMC2	
0b01010	*XIP*		XIP	
0b01011	*XIPwait*	*DOC*	XIPwait	*DOC*
0b01100	*NAND*		NAND	
0b01101	*XIP*		*XIP*	
0b01110	*XIPwait*	*DOC*	*XIPwait*	*DOC*
0b01111	*NAND*		*NAND*	
0b10000	*OneNAND*		*OneNAND*	
0b10001	*MMC2*		*MMC2*	
0b10010	*MMC1*		*MMC1*	
0b10011	*XIP*		*XIP*	
0b10100	*XIPwait*	*DOC*	*XIPwait*	*DOC*
0b10101	*NAND*		*NAND*	
0b10110	*OneNAND*		*OneNAND*	
0b10111	*MMC2*		*MMC2*	
0b11000	*MMC1*		*MMC1*	
0b11001	*XIP*		*XIP*	
0b11010	*XIPwait*	*DOC*	*XIPwait*	*DOC*
0b11011	*NAND*		*NAND*	
0b11100	Reserved[1]			
0b11101				
0b11110				
0b11111	ROM code fast XIP booting (only for GP devices)		ROM code fast XIP booting (only for GP devices)	

(1) Must not be selected

26.3 Power, Clocks, and Reset Power-Up Sequence

See Chapter 4, *Power, Reset, and Clock Management*, section *Power-Up Sequence*, for the power-on sequence, including power supplies, clocks, and reset signals.

26.4 Device Initialization by ROM Code

This section describes high-level booting concepts and provides basic knowledge for booting on the device.

26.4.1 Booting Overview

CAUTION

To use the level 2 (L2) cache with the OMAP34xx device, the ROM code provides three primitive services. These services are implemented in monitor mode and do not use any resources outside the MPU subsystem. The services are described below. To call a service, a register r12 must be set to service ID and the SMI instruction must be executed.

- r12=1: To use the L2 cache, all L2 line data must be invalidated through the CP15 registers. This service invalidates the entire L2 cache and must be performed after a POR or a loss of L2 cache after reset. This register can also be read.
- r12=2: This service writes the value of the central processing unit (CPU) register R0 in the L2 cache auxiliary control register. This register can also be read.
- r12=3: This service writes the value of the CPU register R0 in the auxiliary control. This register can also be read. For more information about ARM L2 cache and registers, see the *Cortex-A8 Technical Reference Manual*. For more information about ARM CP15 registers, see the *ARM Architecture Reference Manual*.

26.4.1.1 Booting Types

Booting is the process of starting a bootstrap from one of the booting memories.

The ROM code has two functionalities for booting: peripheral booting and memory booting.

- In peripheral booting, the ROM code polls a selected communication interface such as UART or USB, downloads the executable code over the interface, and executes it in internal RAM. Downloaded software from an external host can be used to program flash memories connected to the device. This special case of peripheral booting is called preflashing; software downloaded for preflashing is called the flash loader. The flash loader burns a new client application image in external flash memory. Initial software is a generic term for bootstrap, downloaded software, and flash loader. After the image is burned, a software reset can be performed.
- In memory booting, the ROM code finds the bootstrap in permanent memories such as flash memory or memory cards and executes it. This process is normally performed after cold or warm device reset.

The ROM code detects whether the device should download software from a peripheral interface (HS USB or UART 3) by using the sys_boot pin configuration. This mechanism encompasses initial flashing in production (external memory is empty) and reflashing in service (external memory is already programmed).

Note: After reset release, the ROM code tries to boot from several interfaces, testing one after the other until it finds a bootable interface. These interfaces (GPMC, MMC1, MMC2, USB, and UART) are selected in a specific order defined by the sys_boot settings.

As a consequence, if the first bootable interface found by the ROM code is not first in the defined order, some activity occurs on the interfaces that are tested before the bootable interface is found. This activity must be considered in case these intermediary interfaces are connected to other peripherals for other purposes (for example, an LED connected to a GPMC pad muxed internally to a GPIO).

Table 26-6 lists the pin multiplexing according to boot peripheral.

Table 26-6. Pin Multiplexing According to Boot Peripheral

Boot Device	Pins
NAND/OneNAND	General-purpose memory controller (GPMC) pins in mode 0
XIP memory	GPMC pins in mode 0
DiskOnChip	GPMC pins in mode 0
MMC/SD1	MMC1 pins in mode 0
MMC/SD2	MMC2 pins in mode 0
UART3	UART3 pins in mode 0
HS USB	HSUSB0 pins in mode 0

CAUTION

- • UART boot: UART3 is the only possible UART from which boot can be performed. Additionally, only UART3 pads that provide UART3 functionality in their MUXMODE 0 can be used. No other UART configuration allows booting from the UART.
- • HS USB boot: If TWL4030 or TWL5030 processor is used as the USB transceiver, it must be connected to the OMAP3430 through I2C1. No other I^2C interface allows the necessary TWL processor configuration.

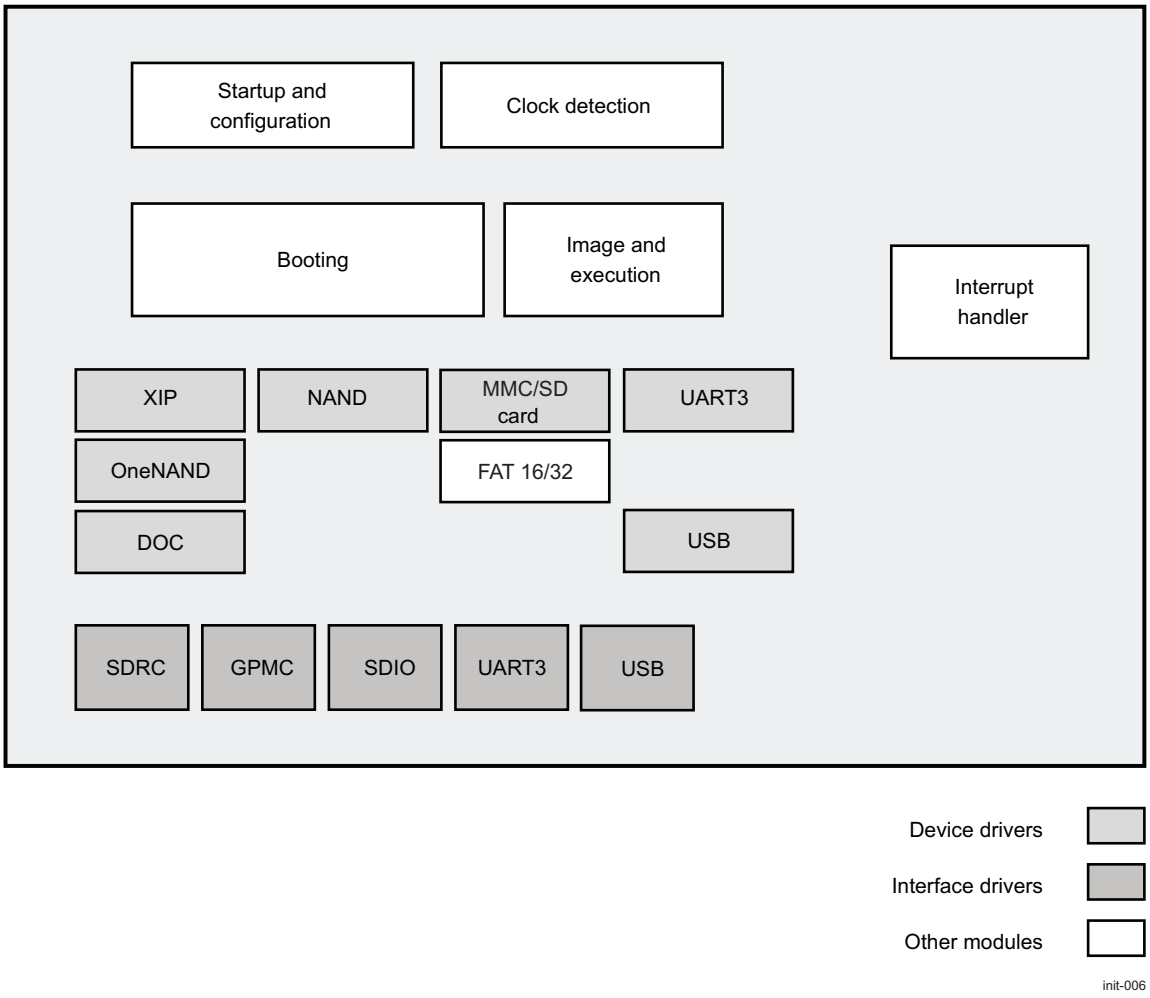
26.4.1.2 Main Features

The ROM code architecture is shown in Figure 26-5. The ROM code is made up of several modules:

- The start-up module takes care of basic system configuration and dispatches control to the booting module.
- The clock detection module supports startup by detecting the system clock frequency.
- The booting module uses several device drivers to boot from an external device.
- The device drivers implement device protocols and perform logical operation on a device. They are abstracted from interface hardware by several interface drivers, which are responsible for interacting with hardware interface facilities.
- The interrupt handler module provides services used by all modules. It allows registering interrupt service routines (ISRs) and calls them when an interrupt occurs.

Figure 26-5 shows each module.

Figure 26-5. ROM Code Architecture

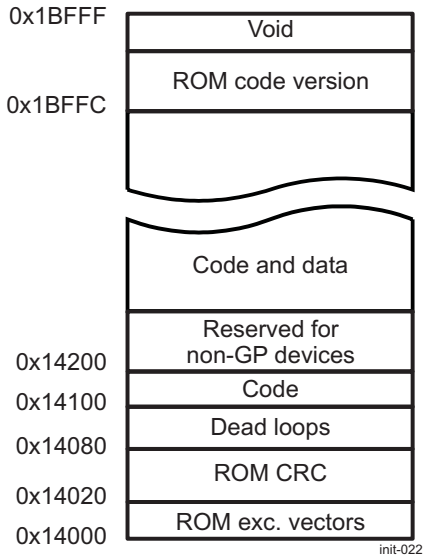


26.4.2 Memory Map

26.4.2.1 ROM Memory Map

Figure 26-6 shows the ROM memory map.

Figure 26-6. 32KB ROM Memory Map



- ROM exception vectors
  Exceptions are redirected to ROM exception vectors (see Table 26-7). The reset exception is redirected to the public ROM code startup. Other exceptions are redirected to RAM handlers by loading appropriate addresses in the PC register.

Table 26-7. ROM Exception Vectors

Address	Exception	Content
14000h	Reset	Branch to the public ROM code startup
14004h	Undefined	PC = 4020FFC8h
14008h	Software interrupt (SWI)	PC = 4020FFCCh
1400Ch	Prefetch abort	PC = 4020FFD0h
14010h	Data abort	PC = 4020FFD4h
14014h	Unused	PC = 4020FFD8h
14018h	IRQ	PC = 4020FFDCh
1401Ch	FIQ	PC = 4020FFE0h

- ROM code cyclic redundancy check (CRC)
  The ROM code CRC is calculated as 32-bit CRC code (CRC-32-IEEE 802.3) for the address range 14000h–1BFFFh. The 4-byte CRC code is stored at location 14020h, which is filled with FFFFFFFFh before the CRC is calculated.
- Dead loops
  Dead loops are branch instructions coded in ARM mode. They have multiple purposes (see Table 26-8).

Table 26-8. Dead Loops

Address	Purpose
14080h	Undefined exception default handler
14084h	SWI exception default handler
14088h	Prefetch abort exception default handler
1408Ch	Data abort exception default handler
14090h	Unused exception default handler
14094h	IRQ exception default handler
14098h	FIQ exception default handler
1409Ch	Validation tests Pass
140A0h	Validation tests Fail
140A4h	Booting failed: No more devices
140A8h	Image not executed or returned
140ACh	Reserved
140B0h	Reserved
140B4h	Reserved
140B8h	Reserved
140BCh	Reserved

The fixed location of these dead loops facilitates debugging and testing. The first seven dead loops are default exception handlers linked with RAM exception vectors.

Dead loops can be called directly from code, but there is also a special function called from ROM code to execute a dead loop. This function is at address 140C0h. The function is assembly code in ARM mode, which takes the dead loop address from the R0 register. The main purpose of the function is to issue a global software reset before going to a dead loop. In addition, the function clears the global cold reset status before issuing the global software reset.

- Code

  This space is used to keep code.
- Code and data

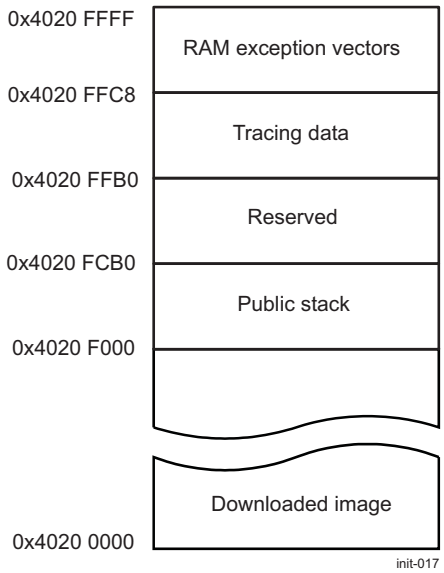
  This space is used to keep code and other data.
- ROM code version

  The ROM code version consists of two decimal numbers: major and minor. The major number is always 14. The minor number identifies the ROM code version. The minor number is not aligned with the ROM code release number, but it can identify it. The ROM code version is coded as hexadecimal readable values (for example, ROM version 14.04 is coded as a 3-bit word: 00001404h).

26.4.2.2 RAM Memory Map

Figure 26-7 shows the RAM memory map. The partitioning of the on-chip SRAM shown in Figure 26-7 is used only during the booting process.

Figure 26-7. 64KB RAM Memory Map of GP Devices



- Downloaded image

  This space is used by the public ROM code to store a downloaded booting image.
- Public stack

  This space is reserved for stacks.
- Tracing data

  The public ROM code tracing data is described in Table 26-9. More information about ROM code tracing can be found in Section 26.4.9, *Tracing*.

Table 26-9. Tracing Data

Address	Size (Bytes)	Description
0x4020FFB0	4	Current tracing vector, word 1
0x4020FFB4	4	Current tracing vector, word 2
0x4020FFB8	4	Current copy of the PRM_RSTST register (reset reasons)
0x4020FFBC	4	Cold reset run tracing vector, word 1
0x4020FFC0	4	Cold reset run tracing vector, word 2
0x4020FFC4	4	Reserved

- RAM exception vectors

  The RAM exception vectors provide an easy way to redirect exceptions to the custom handler. Table 26-10 shows the contents of the RAM space reserved for RAM vectors. The first seven addresses are ARM instructions that load into the PC the value in the next seven addresses. These instructions are executed when an exception occurs, because they are called from ROM exception vectors. By default, all exceptions are redirected to the exception dead loops. Users can redirect an exception to another handler by writing its address to the appropriate position from 0x4020FFE4 to 0x4020FFFC, or by overriding instructions between addresses from 0x4020FFC8 to 0x4020FFE0.

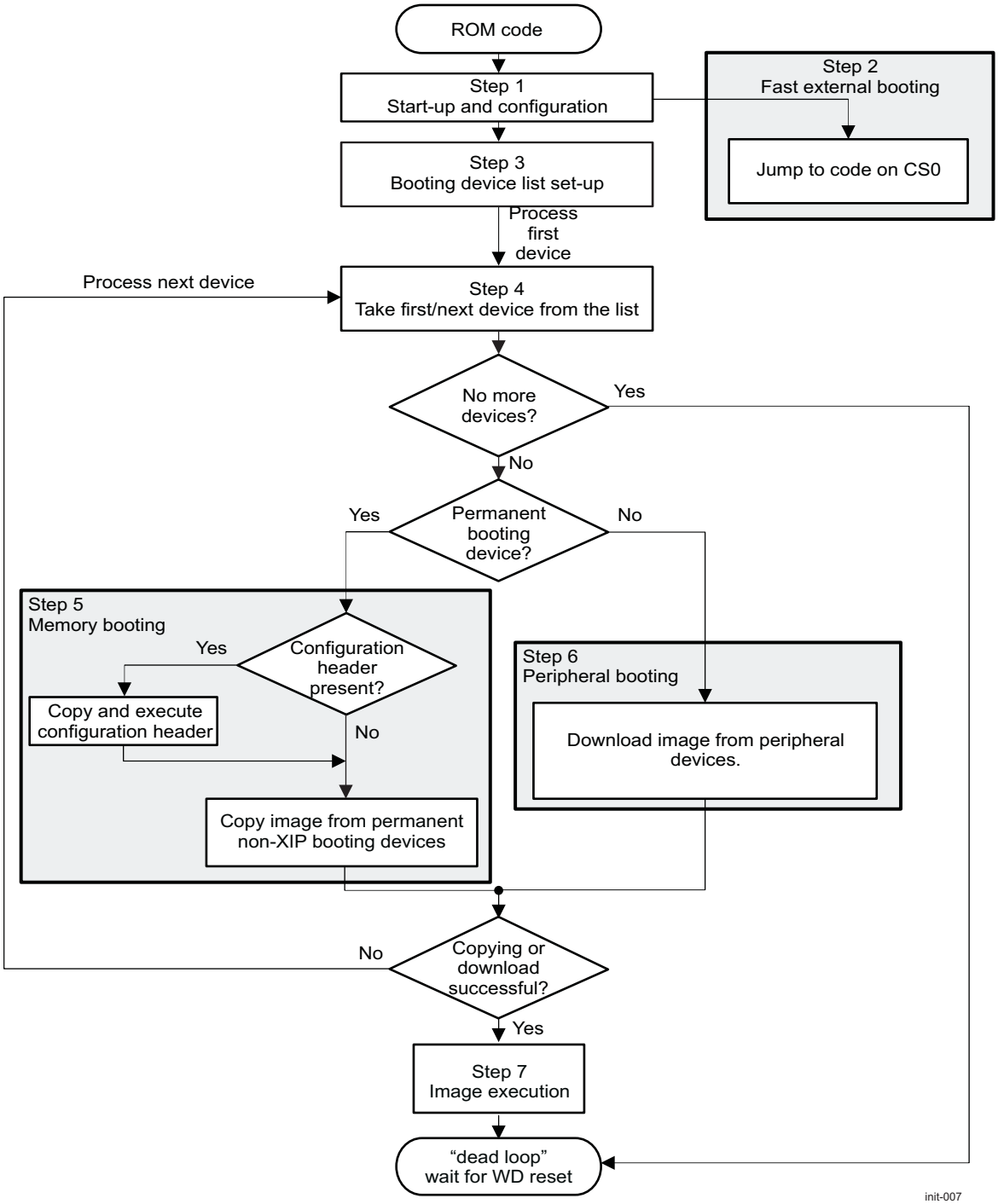
Table 26-10. RAM Exception Vectors

Address	Exception	Content
0x4020FFC8	Undefined	PC = [0x4020FFE4]
0x4020FFCC	SWI	PC = [0x4020FFE8]
0x4020FFD0	Prefetch abort	PC = [0x4020FFEC]
0x4020FFD4	Data abort	PC = [0x4020FFF0]
0x4020FFD8	Unused	PC = [0x4020FFF4]
0x4020FFDC	IRQ	PC = [0x4020FFF8]
0x4020FFE0	FIQ	PC = [0x4020FFFC]
0x4020FFE4	Undefined	0x14080
0x4020FFE8	SWI	0x14084
0x4020FFEC	Prefetch abort	0x14088
0x4020FFF0	Data abort	0x1408C
0x4020FFF4	Unused	0x14090
0x4020FFF8	Interrupt request (IRQ)	0x14094
0x4020FFFC	Fast interrupt request (FIQ)	0x14098

26.4.3 Overall Booting Sequence

Figure 26-8 is the ROM code flow chart.

Figure 26-8. Overall Booting Sequence



The main loop of the booting module goes through the booting device list and tries to get an image from the currently selected booting device. The ROM code performs the following steps:

Step 1. Basic configurations and initializations.

Step 2. The path named fast external boot is a special low-latency boot mode. It consists of a blind jump to the external addressable memory as soon as possible.

Step 3. A booting device list is created (see Section 26.4.4.3, *Booting Device List Setup*). The list consists of all devices to be searched for a booting image. The list is created based on the sys_boot pins and the software booting configuration described in Section 26.4.4.4, *Software Booting Configuration*. The software booting configuration structure is in the scratchpad memory and can be written by software before executing a software reset. The scratchpad memory is an internal RAM memory that keeps its contents after software reset or wakeup, but not after POR. After a software reset, the software booting configuration has priority over the sys_boot pin configuration.

Step 4. After the booting device list is set, the booting procedure examines the devices on the list serially and executes memory booting or peripheral booting, depending on current booting device type:

- Memory booting is executed when the booting device is permanent: XIP memory, NAND, DiskOnChip, OneNAND/Flex-OneNAND, or MMC/SD cards.
- Peripheral booting is executed when device is temporary: UART or USB.

Step 5. Memory booting reads data from memory type devices. Memory booting is described in detail in Section 26.4.7, *Memory Booting*.

Step 6. Peripheral booting downloads data from communication interfaces. The ROM code uses a simple logical protocol with peripheral booting. First, the OMAP device sends an ASIC ID structure to inform the host about the peripheral booting start. The host responds by sending a booting message that can have one of three meanings:

- Skip peripheral booting.
- Continue peripheral booting.
- Change the booting device.

If the message is to continue, the host sends the entire image preceded by its size. Peripheral booting is described in Section 26.4.5, *Peripheral Booting*.

Step 7. For this device, the image is automatically started.

An additional feature of the booting module is the execution of the configuration header (CH). The CH configures the system for faster and more flexible booting from the selected permanent device. The CH, which is optional, is described in *Section 26.4.8.2, CH*.

26.4.4 Startup and Configuration

26.4.4.1 Startup

The ROM code starts at address 0x0001 4000.

26.4.4.2 Clocking Configuration

The ROM code detects the system input clock frequency. The supported frequencies are:

- 12 MHz
- 13 Mhz
- 16.8 MHz
- 19.2 MHz
- 26 MHz
- 38.4 MHz

After detecting the input clock, the ROM code configures the clocks and DPLLs required for ROM code execution.

The configured DPLLs are:

- Peripheral DPLL, set to provide 96 MHz and 48 MHz for peripheral blocks
- MPU DPLL, set to provide 96 MHz for the Cortex™-A8 MPU
- Core DPLL, set to provide 192, 96, 48, or 24 MHz for various blocks, such as interconnect, clocked by this DPLL output

The multipliers and dividers of the DPLLs are set to values which depend on the input clock detected.

Table 26-11 summarizes the ROM code default settings for key clocks.

Table 26-11. ROM Code Default Clock Settings

Clock	Frequency (MHz)	Source
CORE.CLK	192	CORE DPLL output
L3x2.CLK	192	CORE.CLK
L3.ICLK	96	CORE.CLK/2
L4.ICLK	48	L3.ICLK/2
RM clock	24	L4.ICLK/2
MPU	96	CORE.CLK/2 (MPU DPLL in bypass)

The DPLLs and other settings are configured by default after each type of reset to give the ROM code the same working conditions. However, it is possible to override the default clock settings by means of the software booting configuration; for details, see Section 26.4.4.4, *Software Booting Configuration*.

There are three ways to change DPLLs and all related clock divider, gating, and multiplexer configurations during the boot:

- ROM code default settings, described in this paragraph
- Software booting configuration after a software reset, described in Section 26.4.4.4, *Software Booting Configuration*
- CH, described in Section 26.4.8.2, *CH*. This configuration can be blocked by the software booting configuration; this is possible during the memory booting. The CH lets the user have a known configuration (about GPMC, SDRC, and clock registers) after memory booting.

26.4.4.3 Booting Device List Setup

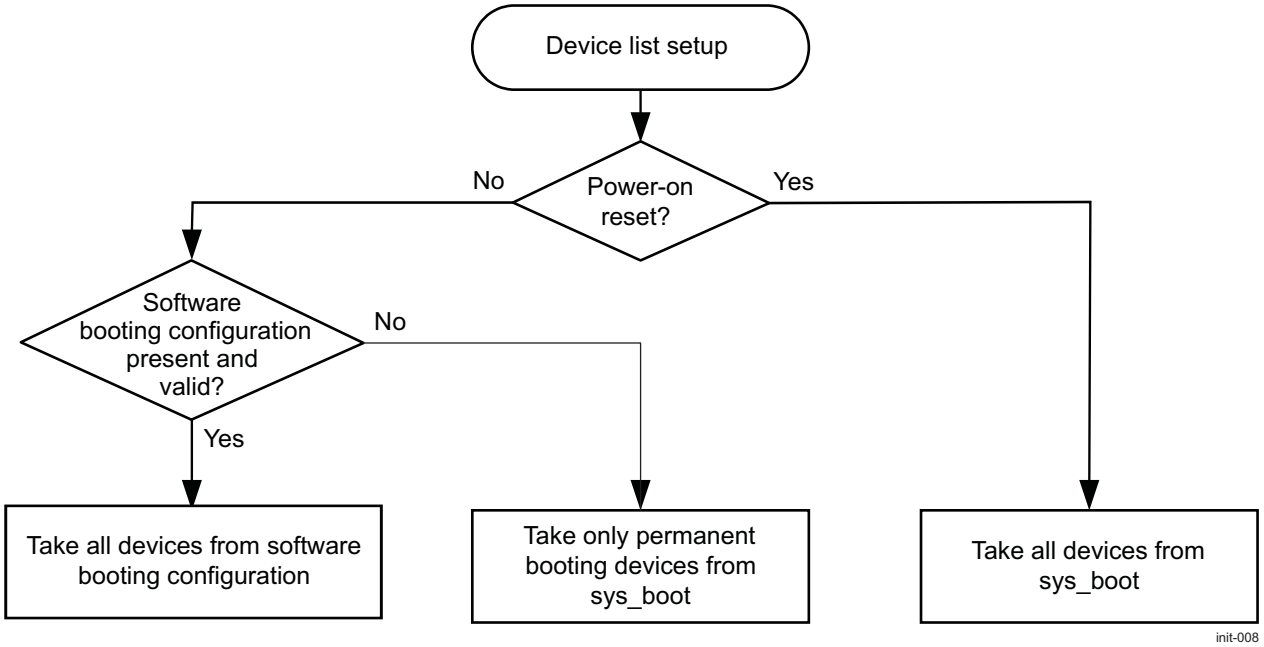
The ROM code creates a device list based on two sources:

- The software booting configuration is stored in nonvolatile RAM memory called scratchpad memory.
- The sys_boot[5:0] pins are used to index the device table from which the list of devices is extracted.

Figure 26-9 shows how the ROM code sets up the device list depending on the reset source.

Note: Only permanent booting devices are put on the list when reset is not power on and devices are taken from the sys_boot pins. Users can force peripheral booting after software reset using software booting configuration.

Figure 26-9. Device List Setup



26.4.4.4 Software Booting Configuration

The software booting configuration is stored in the special scratchpad memory, which is not cleared after software resets or wakeups. The software booting configuration is a simple structure at the address stored at the first location of available scratchpad memory: 0x48002910. Table 26-12 shows the software booting configuration. There are two sections in this structure:

- The first section provides devices for the booting device list.
- The second section provides clock settings, which are applied before booting.

The sections are not mandatory and their order is not important. The ROM code searches for the next section at the location based on the size filled in the previous section. The clock configuration from software booting configuration overwrites the CH settings.

Table 26-12. Software Booting Configuration Structure

Field	Size [bytes]	Description
Booting Configuration		
Section 1 key	4	Synchronization key for section 1: 0xCF00AA01
Section 1 size	4	Size of section 1: 0x0000000C (12)
Flags	2	Bits [4:1]: Mask the CH, when one of these 4 bits is set to 1, the CH section is not analyzed: [1]: SETTINGS section [2]: RAM [3]: FLASH (GPMC) [4]: MMCSD
First device	2	Devices to be put on the device list 0x00: Void, no device 0x01: XIP memory 0x02: NAND 0x03: OneNAND 0x04: DOC 0x05: MMC/SD2
Second device	2	

Table 26-12. Software Booting Configuration Structure (continued)

Field	Size [bytes]	Description
		0x06: MMC/SD1 0x07: XIP memory with wait monitoring
Third device	2	0x08 to 0x0F: Reserved 0x10: UART 0x11: HS USB
Fourth device	2	Others: Reserved
Padding	2	Reserved
Clock Settings		
Section 2 key	4	Synchronization key for section 2: 0xCF00AA02
Section 2 size	4	Size of section 2: 0x00000048 (72)
Flags	4	Bit mask of various switches, active when set to 1: Bit [0]: If 1, the clock configuration defined in this structure is applied. Bit [1]: Reserved Bit [2]: Perform clock configuration settings. Bit [3]: Set and lock DPLL4 PER. Bit [4]: Set and lock DPLL1 (MPU). Bit [5]: Set and lock DPLL3 (CORE). Bit [6]: Bypass DPLL4 before setting clocks. Bit [7]: Bypass DPLL1 before setting clocks. Bit [8]: Bypass DPLL3 before setting clocks. Bits [24..31]: System clock ID Must be set accordingly to the SYS.CLK: 0x01: 12 MHz 0x02: 13 MHz 0x03: 16.8 MHz 0x04: 19.2 MHz 0x05: 26 MHz 0x06: 38.4 MHz Others: Reserved, must not be set
General Clock Settings		
PRM_CLKSRC_CTRL	4	Register value
PRM_CLKSEL	4	Register value
CM_CLKSEL1_EMU	4	Register value
Clock Configuration		
CM_CLKSEL_CORE	4	Register value
CM_CLKSEL_WKUP	4	Register value
DPLL3 (Core) Settings		
CM_CLKEN_PLL	4	Register value
CM_AUTOIDLE_PLL	4	Register value
CM_CLKSEL1_PLL	4	Register value
DPLL4 (Peripheral) Settings		
CM_CLKEN_PLL	4	Register value
CM_AUTOIDLE_PLL	4	Register value
CM_CLKSEL2_PLL	4	Register value
CM_CLKSEL3_PLL	4	Register value
DPLL1 (MPU) Settings		
CM_CLKEN_PLL_MPU	4	Register value
CM_AUTOIDLE_PLL_MPU	4	Register value

Table 26-12. Software Booting Configuration Structure (continued)

Field	Size [bytes]	Description
CM_CLKSEL1_PLL_MPU	4	Register value
CM_CLKSEL2_PLL_MPU	4	Register value
CM_CLKSTCTRL_MPU	4	Register value

26.4.5 Peripheral Booting

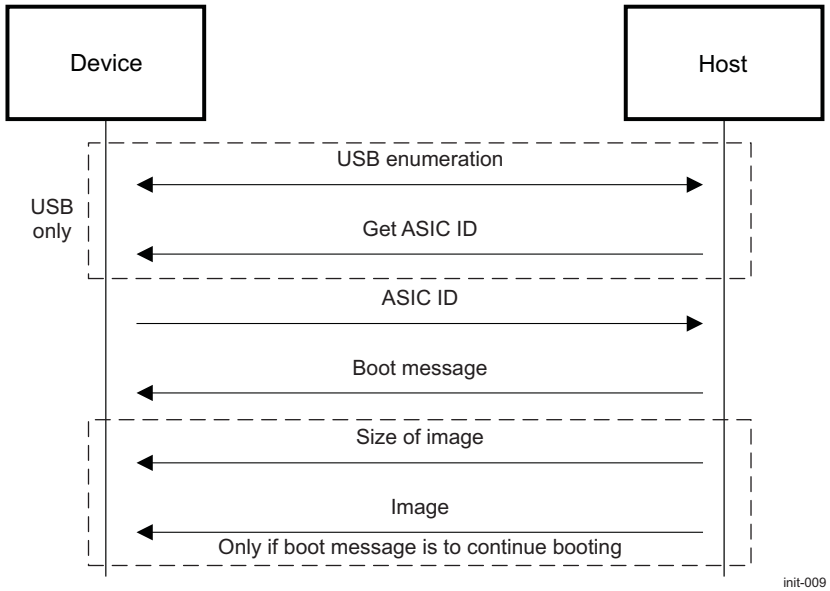
26.4.5.1 Overview

The ROM code can boot from different peripherals:

- HS USB: HS USB interface
- UART 3: Baud rate 115.2 Kbps, 8 bits, even parity, 1 stop-bit

The purpose of booting from a peripheral is to boot from an external host, such as a PC. This booting method is used primarily for programming flash memories connected to the OMAP device. The protocol is common to all peripherals. Some minor exceptions are described in the following sections. The common peripheral booting protocol is shown in Figure 26-10.

Figure 26-10. Common Peripheral Booting Protocol



The ROM code first initializes the interface and sends a message called ASIC ID to a host. The content of this message is summarized in Table 26-13. The host uses this message to send only appropriate data to the OMAP device according to the identification codes sent.

The ROM code waits 300 ms for an answer from the host. If a time-out occurs, the peripheral booting returns to the main booting procedure with TIMEOUT status.

Table 26-13. ASIC ID Structure

ASIC ID Item	Size [Bytes]	Description
Items	1	Number of subblocks
ID subblock	7	Device identification information
Reserved for non-GP devices	4	Reserved
ID subblock	23	Identification data

Table 26-13. ASIC ID Structure (continued)

ASIC ID Item	Size [Bytes]	Description
Reserved for non-GP devices	23	Reserved
Checksum subblock	11	CRC (4 bytes)

The host can send different messages, as described in Table 26-14. If the second or third message is not received, the ROM code stops the current peripheral booting procedure and returns to the main booting, which determines the next booting device according to the boot message received.

If the first message is received without a time-out, the image size (a 32-bit word) and the image itself are expected to be received. The image is downloaded directly at address 0x40200000 in the internal RAM.

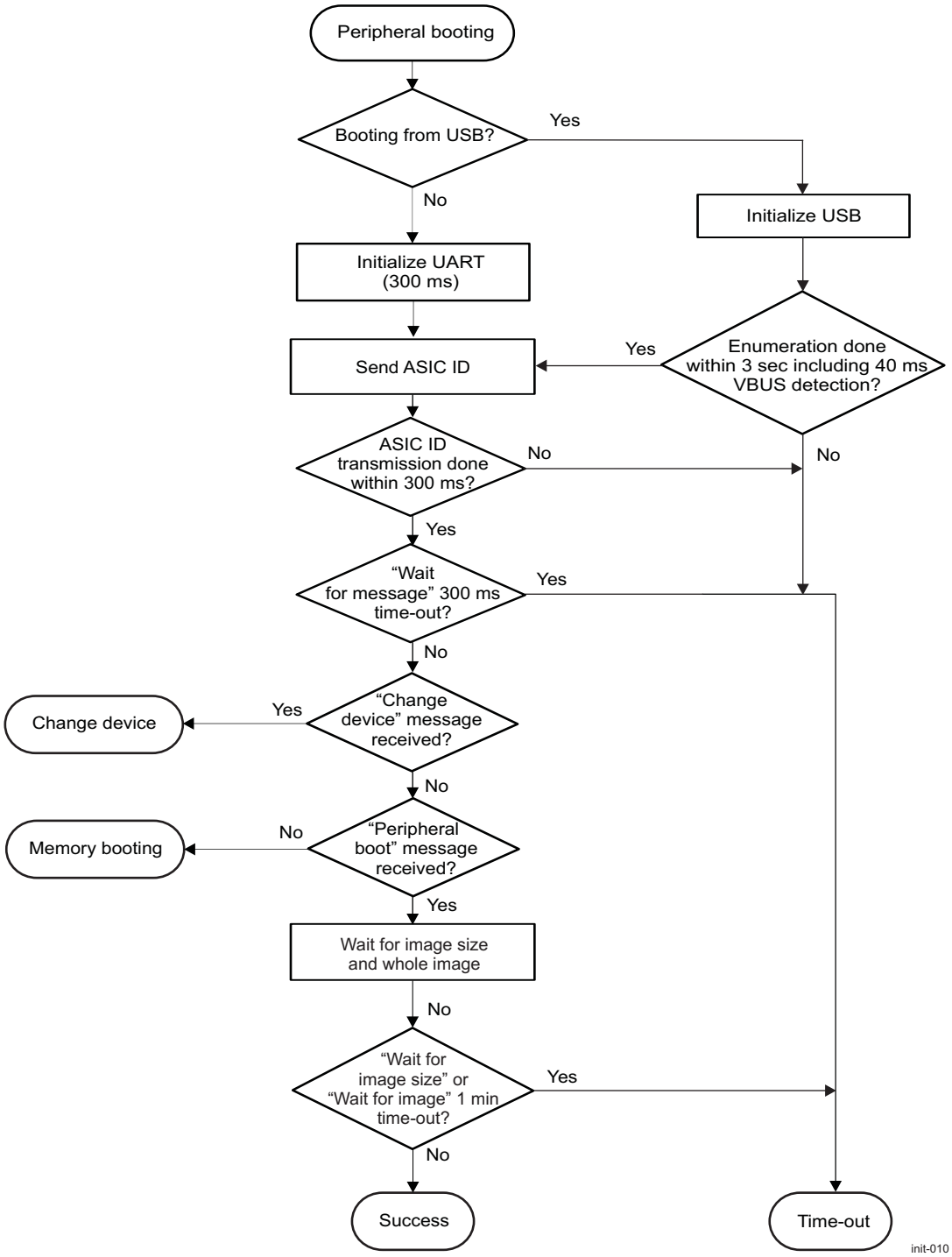
The ROM code waits up to 1 minute for completion. If the downloading procedure does not complete before this period, the peripheral booting fails. If the download passes, the peripheral booting succeeds and the image can be executed.

Table 26-14. Boot Messages

Message Name	Value	Description
Peripheral boot	0xF0030002	Continue peripheral booting.
Change device	0xF003XX06	Skip current peripheral booting and continue booting from device type indicated by XX: 0x00: Void, no device 0x01: XIP memory 0x02: NAND 0x03: OneNAND 0x04: DOC 0x05: MMC/SD2 0x06: MMC/SD1 0x07: XIP memory with wait monitoring 0x08..0x0F: Reserved 0x10: UART 0x11: HS USB Others: Reserved
Next device	0xFFFFFFFF	Skip current device and move to the next device on the device list.
Memory booting	Others	Skip current peripheral booting and move to the first device for memory booting.

Figure 26-11 shows the peripheral booting procedure.

Figure 26-11. Peripheral Booting Procedure



26.4.5.2 UART

The ROM code supports booting from a UART interface with the following characteristics:

- UART interface 3
- Communication parameters set to 115.2 Kbps, 8 bits, even parity, 1 stop-bit
- Two-pin interface: RX/TX with software flow control (XON/XOFF). The other UART pins are left at their default configuration.
- The UART time-out is 300 ms.

26.4.5.3 USB

The ROM code supports booting from a USB interface with the following characteristics:

- HS USB interface
- USB 2.0 transceiver macrocell interface (UTMI)+ low pin interface (ULPI) 8-bit data transceiver support (single data rate)
- TWL4030 device detection and automatic configuration of its USB transceiver using I2C1
- The enumeration time-out is 3 seconds.

The ROM code USB driver conforms with the USB 2.0 specification and the USB on-the-go (OTG) supplement. It supports transactions at HS (that is, 480 Mbps) and FS (that is, 12 Mbps). During peripheral booting, only the USB device functionality is used. The driver resides in the on-chip memory (OCM) ROM, which is small. The driver therefore contains the minimum functionality needed as a USB device and is not a full-fledged driver. It does not contain the functionality needed for a USB host. The device functionality of the USB OTG controller is used for the peripheral booting process in the ROM code.

HS USB peripheral booting with a cellular systems software tool (CSST) is possible with a software development platform (SDP). The user can download CSST on www.ti.com (Wireless Handset Solution > OMAP Platform > Development Tools).

26.4.5.3.1 USB Driver Descriptors

USB devices report their attributes using descriptors. A descriptor is a data structure with a defined format. Each descriptor begins with a byte-wide field that contains the total number of bytes in the descriptor followed by a byte-wide field that identifies the descriptor type. Using descriptors allows concise storage of the attributes of individual configurations so that each configuration can reuse descriptors or portions of descriptors from other configurations that have the same characteristics. Where appropriate, descriptors contain references to string descriptors. String descriptors contain displayable, human-readable information that describes a descriptor. These descriptor details can be used for tool development or debugging:

- Device descriptor

  A device descriptor contains general information about a USB device, including information that applies globally to the device and all device configurations. A USB device has only one device descriptor. Because the ROM code uses the HS feature of the USB core, a device-qualifier descriptor is required. Table 26-15 describes the device descriptors.

Table 26-15. Device Descriptor

Field	Value	Description
bLength	0x12	Size of this descriptor in bytes
bDescriptorType	0x01	Device descriptor type
bcdUSB	0x0210	USB specification release number in binary coded decimal (BCD) format
bDeviceClass	Vendor-specific (0xFF)	Class code
bDeviceSubClass	Vendor-specific (0xFF)	Subclass code
bDeviceProtocol	Vendor-specific (0xFF)	Protocol code
bMaxPacketSize0	0x40	Maximum packet size for endpoint 0

Table 26-15. Device Descriptor (continued)

Field	Value	Description
idVendor	0x0451	Vendor ID
idProduct	0xD009	Product ID
bcdDevice	0x0000	Device release number
iManufacturer	See values in Section 26.4.5.3.2.	Index of string descriptor describing manufacturer
iProduct	See values in Section 26.4.5.3.2.	Index of string descriptor describing product
iSerialNumber	See values in Section 26.4.5.3.2.	Index of string descriptor describing device serial number
bNumConfigurations	0x01	Number of possible configurations

- Device-qualifier descriptor

  The device-qualifier descriptor contains information about a HS-capable device that changes if the device operates at its other speed. This descriptor is retrieved by the host using the GetDescriptor() request (standard device request). Table 26-16 describes a device-qualifier descriptor.

Table 26-16. Device-Qualifier Descriptor

Field	Value	Description
bLength	0x0a	Size of this descriptor in bytes
bDescriptorType	0x06	Device-qualifier descriptor type
bcdUSB	0x0210	USB specification release number in BCD
bDeviceClass	0xFF	Class code
bDeviceSubClass	0xFF	Subclass code
bDeviceProtocol	0xFF	Protocol code
bMaxPacketSize0	0x40	Maximum packet size for endpoint 0
bNumConfigurations	0x01	Number of possible configurations
bReserved	0x00	Reserved for future use

- Configuration descriptor

  This descriptor gives information about a specific device configuration. The descriptor describes the number of interfaces supported by the configuration. See Table 26-17 for details.

Table 26-17. Configuration Descriptor

Field	Value	Description
bLength	0x09	Size of this descriptor in bytes
bDescriptorType	0x02	Configuration descriptor type
wTotalLength	–	Combined length of all descriptors
bNumInterfaces	0x01	Number of interfaces supported
bConfigurationValue	0x01	Value to use as an argument for the SetConfiguration() request
iConfiguration	Index	Index of string descriptor describing this configuration
bmAttributes	0xc0	Power setting and remote wakeup
bMaxPower	0x32	Maximum power consumption of the USB device

- Other speed configuration descriptor

  This descriptor describes the configuration of a HS-capable device if it operates at its other possible speed. See Table 26-18 for details.

Table 26-18. Other Speed Configuration Descriptor

Field	Value	Description
bLength	0x09	Size of this descriptor in bytes
bDescriptorType	0x07	Other speed configuration descriptor type
wTotalLength	–	Combined length of all descriptors
bNumInterfaces	0x01	Number of interfaces supported
bConfigurationValue	0x01	Value to use as an argument for the SetConfiguration() request
iConfiguration	Index	Index of string descriptor describing this configuration
bmAttributes	0xc0	Power setting and remote wakeup
bMaxPower	0x32	Maximum power consumption of the USB device

- Interface descriptor

  This descriptor describes a specific interface in a configuration. See Table 26-19 for details.

Table 26-19. Interface Descriptor

Field	Value	Description
bLength	0x09	Size of this descriptor in bytes
bDescriptorType	0x04	Interface descriptor type
bInterfaceNumber	0x00	Number of this descriptor
bAlternateSetting	0x00	Value to select the alternate setting
bNumEndpoints	0x02	Number of endpoints used for this interface
bInterfaceClass	0xFF	Class code
bInterfaceSubClass	0xFF	Subclass code
bInterfaceProtocol	0xFF	Protocol code
iInterface	Index	Index of string descriptor describing this interface

- Endpoint descriptor

  Each endpoint used for an interface has its own descriptor. This descriptor contains information required by the host to determine the bandwidth requirements of each endpoint. This descriptor is returned as part of the GetDescriptor(Configuration) request. See Table 26-20 and Table 26-21 for details.

Table 26-20. BULK IN Endpoint Descriptor

Field	Value	Description
bLength	0x07	Size of this descriptor in bytes
bDescriptorType	0x05	Endpoint descriptor type
bEndpointAddress	0x81 (1 IN)	Address of the endpoint on the USB device
bmAttributes	0x02 (Bulk)	Type of transfer
wMaxPacketSize	See (1).	Number of endpoints used for this interface
bInterval	0x00	Maximum NAK rate

(1) The maximum size is 0x0200 (512 bytes) for HS bulk endpoint and 0x0040 (64 bytes) for FS bulk endpoint.

Table 26-21. BULK OUT Endpoint Descriptor

Field	Value	Description
bLength	0x07	Size of this descriptor in bytes
bDescriptorType	0x05	Endpoint descriptor type
bEndpointAddress	0x01 (1 OUT)	Address of the endpoint on the USB device
bmAttributes	0x02 (Bulk)	Type of transfer
wMaxPacketSize	See (1).	Number of endpoints used for this interface
bInterval	0x00	Maximum NAK rate

(1) The maximum size is 0x0200 (512 bytes) for HS bulk endpoint and 0x0040 (64 bytes) for FS bulk endpoint.

- String descriptors

  String descriptors use UNICODE encoding. The strings in a USB device can support multiple languages. When requesting a string descriptor, the requester specifies the desired language using a 16-bit language ID (LANGID) defined by the USB interface. String index 0 for all languages returns a string descriptor that contains an array of 2-byte LANGID codes supported by the device.

  See the tables describing string descriptors:

  – The language ID string descriptor (Table 26-22)
  – The manufacturer ID string descriptor (Table 26-23)
  – The product ID string descriptor (Table 26-24)
  – The configuration string descriptor (Table 26-25)
  – The interface string descriptor (Table 26-26)

Table 26-22. Language ID String Descriptor

Field	Value	Description
bLength	0x04	Size of this descriptor in bytes
bDescriptorType	0x03	String descriptor type
wLangId	0x0409 (US English)	Language ID code

Table 26-23. Manufacturer ID String Descriptor

Field	Value	Description
bLength	0x24	Size of this descriptor in bytes
bDescriptorType	0x03	String descriptor type
bString	Texas Instruments	Manufacturer string

Table 26-24. Product ID String Descriptor

Field	Value	Description
bLength	0x12	Size of this descriptor in bytes
bDescriptorType	0x03	String descriptor type
bString	OMAP3430 or specific vendor string	Product string

Table 26-25. Configuration String Descriptor

Field	Value	Description
bLength	0x08	Size of this descriptor in bytes
bDescriptorType	0x03	String descriptor type
bString	pbc	Configuration string

Table 26-26. Interface String Descriptor

Field	Value	Description
bLength	0x08	Size of this descriptor in bytes
bDescriptorType	0x03	String descriptor type
bString	pbi	Interface string

26.4.5.3.2 USB Customized Descriptors

There are two parameters in USB descriptors that customers can define after the chip is created: vendor ID (VID) and product ID (PID). The ROM code uses dedicated eFuses that hold VID and PID values. Other parameters can also be changed based on VID value. The ROM code has an encoded set of parameters for customers who have defined their requirements before the ROM code has been done. Table 26-27 lists the parameters that depend on VID value.

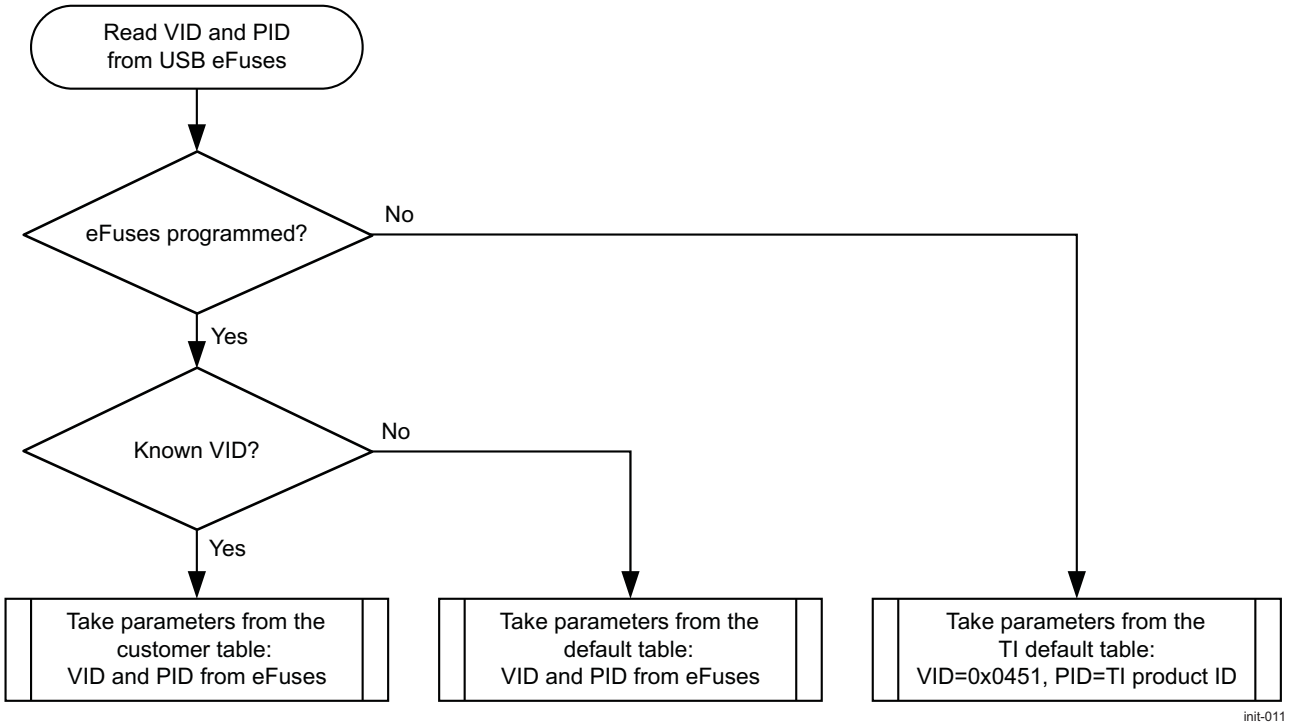
Table 26-27. Customized Descriptor Parameters

Parameter	Size [Bytes]	Default Values	TI Default Values
Device ID code	2	0x0000	0x0000
Device class	1	0xFF	0xFF
Device subclass	1	0xFF	0xFF
Device protocol	1	0xFF	0xFF
Manufacturer	String	N/A	Texas Instruments
Product	String	OMAP3430 or specific vendor string	OMAP3430 or specific vendor string
Serial number	String	See(1)	See (1)

(1) The standard device descriptor indicates that the device has no serial number.

Figure 26-12 describes an additional customer parameter selection method. It is based on the VID burned in the USB eFuses.

Figure 26-12. Customer USB Descriptor Selection



26.4.5.3.3 USB Driver Functionality

- Transactions supported
  The following transactions are supported:
  - Control transactions: Used for standard device requests
  - Bulk transactions: Used for data transfer in the image downloading stage. The ASIC ID is sent on

the Bulk IN endpoint and the image is transferred over the Bulk OUT endpoint from the host.

The OMAP USB device first attaches to the host as an FS device. In the reset mechanism, the USB core requests HS operation. If the HS negotiation in the reset phase is successful, further transactions are at HS; otherwise, they are at FS. After reset, the USB driver checks for the speed of the device, whether it is FS or HS. Depending on the speed configured by the host, the standard USB device requests are responded to with the corresponding descriptors.

- Standard device request restrictions

  Because the USB driver is used only by the ROM code for peripheral booting, some standard device requests are not supported by the driver. Table 26-28 lists the standard device requests supported by the driver.

Table 26-28. Standard Device Requests Supported

Request	Description	Support
CLEAR_FEATURE	Sets/clears a specific feature	Supported only for ENDPOINT_HALT feature
GET_CONFIGURATION	Returns the current device configuration value	Yes
GET_DESCRIPTOR	Returns the specified descriptor	Yes
GET_INTERFACE	Returns the selected alternate setting for the specified interface	Yes
GET_STATUS	Returns the status for the specified recipient	Yes
SET_ADDRESS	Sets the device address	Yes
SET_CONFIGURATION	Sets the device configuration	Yes
SET_DESCRIPTOR	Updates existing descriptors or adds new descriptors	No. Runtime updating of descriptors is not supported.
SET_FEATURE	Sets or enables a specific feature	Supported only for ENDPOINT_HALT feature
SET_INTERFACE	Selects an alternate setting in an interface	No. Runtime setting of alternate features is not supported.
SYNCH_FRAME	Sets and reports an endpoint synchronization frame	No, because isochronous transfers are not used

26.4.6 Fast External Booting

26.4.6.1 Overview

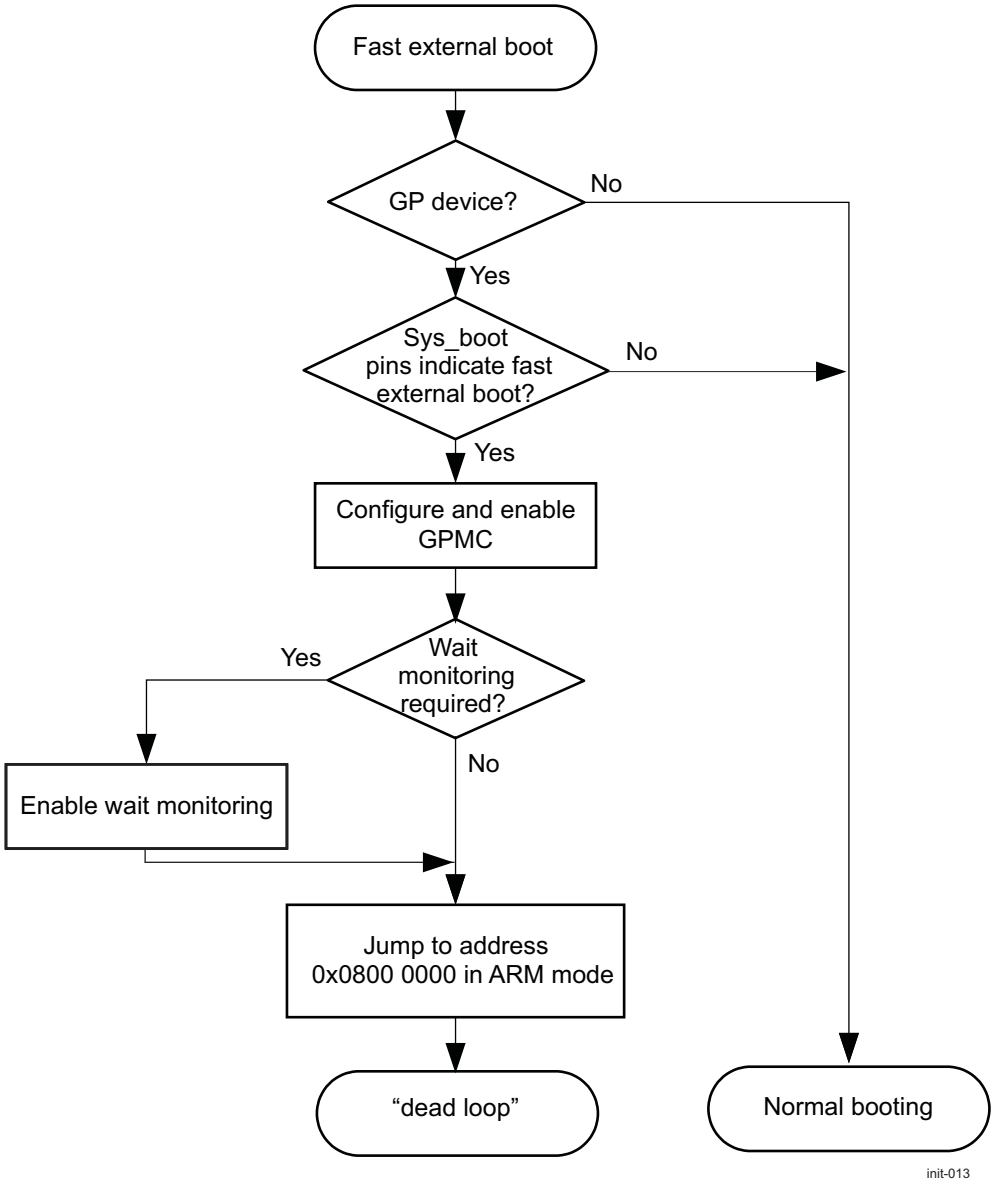
The fast external boot is a special memory booting mode. It is a blind jump to a code in an external XIP device connected to chip-select 0 (CS0). Fast external booting lets customers create their own booting code.

The jump is performed with minimum on-chip ROM code execution.

26.4.6.2 External Booting

Figure 26-13 shows the fast external boot procedure. The code is at the beginning and is written in assembly. The code does not use any RAM.

Figure 26-13. Fast External Boot Procedure



26.4.7 Memory Booting

26.4.7.1 Overview

The memory booting process starts an external code in memory type devices. Because the OMAP34xx device always uses these devices for booting, they are called permanent booting devices. The supported permanent booting devices are:

- All NOR devices up to 1 Gb (128M bytes)
- NAND devices from 64Mb
- OneNAND/Flex-OneNAND devices from 512Mb

- SD/MMC flash cards with active primary partition of type FAT12/16/32
- DiskOnChip H3 devices

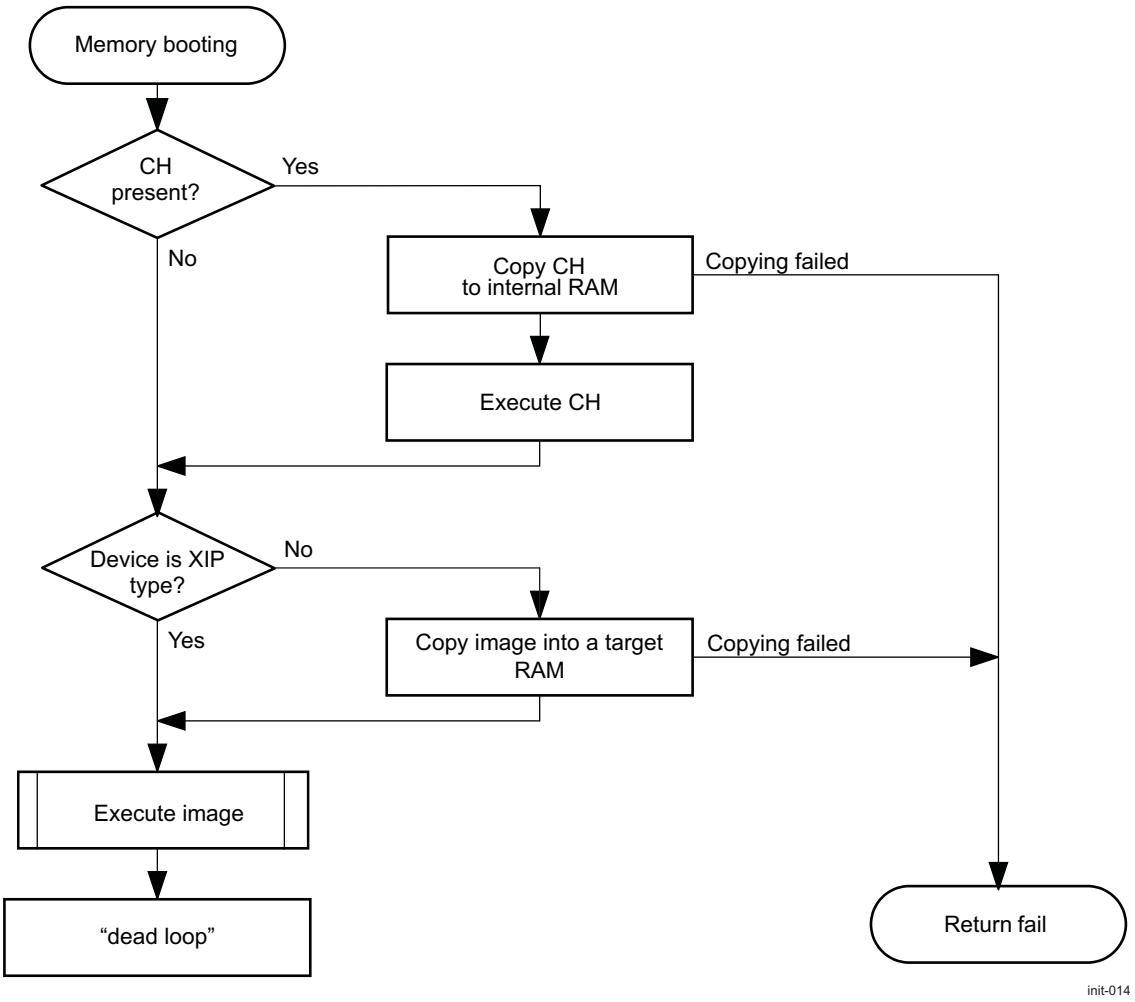
Two main groups of permanent booting devices are distinguished by code shadowing. Code shadowing means copying a code from a nondirectly addressable device (non-XIP) to RAM, where the code can be executed. Directly addressable devices are XIP devices.

Figure 26-14 is the general memory booting procedure common to all types of devices. First, the CH is copied to internal RAM. It is copied even for XIP devices, because the OMAP34xx device can temporarily lose a connection with XIP memory during CH execution. The second step is to shadow the image, if the device is not XIP. The last step is image authentication and execution.

Unsuccessful authentication or return from image results in a dead loop.

If CH copying or shadowing fails, memory booting returns to the main booting procedure, which selects the next device for booting.

Figure 26-14. Memory Booting Procedure



26.4.7.2 Non-XIP Memory

Figure 26-15 shows the procedure used when memory booting runs with non-XIP devices. The grayed procedures are specific to each device. NAND and OneNAND/Flex-OneNAND devices use up to four copies of the image in the first four physical blocks. Therefore, the ROM code searches for the image in the first four physical blocks of these devices. Other devices use only one copy of the image and the block loop runs only once.

During image shadowing on a GP device, the CH is expected to be in a separate sector before the initial software.

Figure 26-15. Detailed Memory Booting for Non-XIP Devices

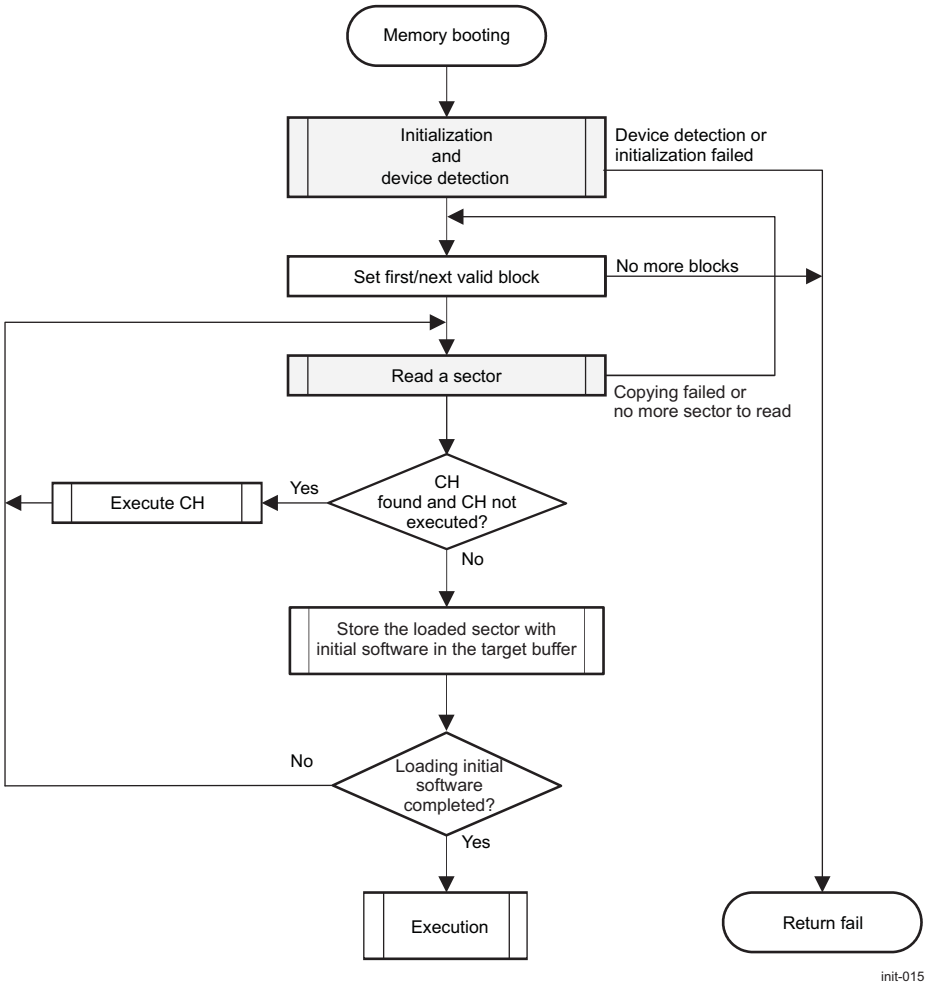


Table 26-29 summarizes numbers of blocks and sectors that are searched during memory booting from devices requiring image shadowing. NANDs and OneNAND/Flex-OneNAND are organized with blocks, which are erasable units. DOC memory reserves only one block for booting, which overlaps the XIP part. MMC/SD card booting consists of reading a file. Because there is only one file read, it can be considered one block trial.

Table 26-29. Blocks and Sectors Searched on Non-XIP Memories

Memory	Maximum Number of Checked Blocks	Number of Sectors Searched
NAND	First 4	Number of sectors in a block(1)
OneNAND/Flex-OneNAND	First 4	8
DOC(2)	1	4
MMC/SD + FAT system	1 file	
MMC/SD raw	2	1

(1) Depends on NAND type

(2) Because XIP booting always precedes DOC booting and the same data is used for XIP areas, the DOC memory image must contain the first sector filled with 0xFF or 0x00. Therefore, a void sector at the beginning makes XIP booting fail and moves to DOC booting.

The following sections describe the supported device types.

For more information about the GPMC module, see Chapter 11, *Memory Subsystem*.

26.4.7.3 XIP Memory

The ROM code can boot directly from XIP devices, such as NOR flash memories, that have the following characteristics:

- The GPMC is the communication interface.
- Memories up to 1GB (128MB) can be connected.
- x16 data bus width only
- Asynchronous protocol and address/data multiplexed mode
- The GPMC clock is 48 MHz.
- The device is connected to CS0 mapped to address 0x0800 0000.
- The wait pin signal gpmc_wait0 is monitored according to the sys_boot configuration pins.

Depending on the sys_boot option, the GPMC can be configured to use the wait signal connected to the gpmc_wait0 pin. Wait pin polarity is set to stall accessing memory when gpmc_wait0 is low. Wait monitoring is used with memories that require a long time for initialization after reset, or that must pause while reading data. An example of such memory is DiskOnChip.

For an XIP memory booting, no user intervention is needed; the following steps are described for debugging. Only the CH, which is not mandatory, lets the user change clock settings and GPMC parameters. Failure in CH copying causes a return to the main booting procedure, which selects the next device.

Booting from an XIP device consists of the following steps:

1. Configure the GPMC for XIP device access.
2. Verify that the CH is present at address 0x0800 0000. If it is, copy the entire sector (512 B) to internal RAM and execute the CH.
3. Set the image location:
   - 0x0800 0000 if the CH is not found
   - 0x0800 0200 if the CH (512 bits) is found
4. Verify that a bootable image is at the image location.
5. Execute the image if it is found.
6. If the image is not found, return from XIP booting to the main booting loop.

26.4.7.3.1 GPMC Initialization

Table 26-30 describes the timing settings of GPMC set for XIP and other address-data accessible devices, like DiskOnChip or OneNAND/Flex-OneNAND. Table 26-30 is included for debug information.

Table 26-30. XIP Timing Parameters

Parameter	Value [Clock Cycles]	Register Initialization (i = 0–7)	Reset Value
Write cycle time	17	The GPMC_CONFIG5_i[12:8] WRCYCLETIME bit field is set to 0x11.	0x11
Read cycle time	17	The GPMC_CONFIG5_i[4:0] RDCYCLETIME bit field is set to 0x11.	0x11
CS low time	1	The GPMC_CONFIG2_i[3:0] CSONTIME bit field is set to 0x1.	0x1
CS high time	16	The GPMC_CONFIG2_i[12:8] CSRDOFFTIME bit field is set to 0x10.	0x10
ADV low time	1	The GPMC_CONFIG3_i[3:0] ADVONTIME bit field is set to 0x1.	0x1
ADV high time	2	The GPMC_CONFIG3_i[12:8] ADVRDOFFTIME bit field is set to 0x2.	0x2

Table 26-30. XIP Timing Parameters (continued)

Parameter	Value [Clock Cycles]	Register Initialization (i = 0–7)	Reset Value
OE low time	3	The GPMC_CONFIG4_i[3:0] OEONTIME bit field is set to 0x3.	0x3
OE high time	16	The GPMC_CONFIG4_i[12:8] OEOFFTIME bit field is set to 0x10.	0x10
WE low time	3	The GPMC_CONFIG4_i[19:16] WEONTIME bit field is set to 0x3.	0x03
WE high time	15	The GPMC_CONFIG4_i[28:24] WEOFFTIME bit field is set to 0xF.	0x10
Data latch time	15	The GPMC_CONFIG5_i[20:16] RDACCESSTIME bit field is set to 0xF.	0x0F

26.4.7.4 NAND

NAND flash memory is not an XIP device; it requires shadowing before the code can be executed. ROM code support for the NAND flash devices has the following characteristics:

- The GPMC is the communication interface.
- Device from 64 Mb (8MB)
- x8 and x16 bus width
- Small page size (512 bytes + 16 bytes) and large page size (2048 bytes + 64 bytes)
- Chip enable (CE) don't care devices only
- Single level cell (SLC) and multilevel cell (MLC) devices
- Device identification is based on standard identification data or ID2 protocol.
- One-bit error checking and correction (ECC) is used to protect a 512-byte sector.
- GPMC timings are adjusted for NAND access.
- The GPMC clock is 48 MHz.
- The device is connected to CS0.
- The wait pin signal gpmc_wait0 is connected to the NAND BUSY output.
- Four physical blocks are searched for image. Block size depends on the device.

For NAND memory booting, no user intervention is needed; the information in the following subsections is included for debugging. Only the CH, which is not mandatory, lets the user change clock settings and GPMC parameters. Failure in CH copying causes a return to the main booting procedure, which selects the next device for booting.

26.4.7.4.1 Initialization and NAND Detection

The initialization routine for NAND consists of three parts: GPMC initialization, device detection with parameter determination, and bad block detection/verification.

- GPMC initialization

  The GPMC interface is configured so that it can access NANDs. Because NANDs do not need the address bus, it is released. The data bus width is initially set to 8 bits. If necessary, it is changed to 16 bits after the device parameters are determined. Table 26-31 shows the GPMC configuration used during NAND boot. Table 26-31 is included for debug information.

Table 26-31. NAND Timing Parameters

Parameter	Value [Clock Cycles]	Register Initialization (i = 0–7)	Reset Value
Write cycle time	20	The GPMC_CONFIG5_i[12:8] WRCYCLETIME bit field is set to 0x14.	0x11
Read cycle time	20	The GPMC_CONFIG5_i[4:0] RDCYCLETIME bit field is set to 0x14.	0x11

Table 26-31. NAND Timing Parameters (continued)

Parameter	Value [Clock Cycles]	Register Initialization (i = 0–7)	Reset Value
CS low time	0	The GPMC_CONFIG2_i[3:0] CSONTIME bit field is set to 0x0.	0x1
OE low time	5	The GPMC_CONFIG4_i[3:0] OEONTIME bit field is set to 0x5.	0x3
OE high time	16	The GPMC_CONFIG4_i[12:8] OEOFFTIME bit field is set to 0x10.	0x10
WE low time	3	The GPMC_CONFIG4_i[19:16] WEONTIME bit field is set to 0x3.	0x3
WE high time	15	The GPMC_CONFIG4_i[28:24] WEOFFTIME bit field is set to 0xF.	0x10
Data latch time	14	The GPMC_CONFIG5_i[20:16] RDACCESSTIME bit field is set to 0xE.	0xF

- Device detection and parameters

  The ROM code must first identify the NAND type connected on the GPMC interface. The GPMC is initialized using 8 bits, asynchronous mode. The NAND device is reset and its status is polled until it is ready for operation, then the Read ID command is issued. If the Read Device ID is recognized as a supported device, the device parameters are extracted from an internal ROM code table. Table 26-32 lists the supported devices.

Table 26-32. Supported NAND Devices

Capacity	Device ID	Bus Width	Page Size in KB
64Mb	E6h	8	512
128Mb	33h	8	512
128Mb	73h	8	512
128Mb	43h	16	512
128Mb	53h	16	512
256Mb	35h	8	512
256Mb	75h	8	512
256Mb	45h	16	512
256Mb	55h	16	512
512Mb	36h	8	512
512Mb	76h	8	512
512Mb	46h	16	512
512Mb	56h	16	512
512Mb	A2h	8	2048
512Mb	F2h	8	2048
512Mb	B2h	16	2048
512Mb	C2h	16	2048
1Gb	39h	8	512
1Gb	79h	8	512
1Gb	49h	16	512
1Gb	59h	16	512
1Gb	78h	8	512
1Gb	72h	16	512
1Gb	74h	16	512
1Gb	A1h	8	2048
1Gb	F1h	8	2048
1Gb	B1h	16	2048
1Gb	C1h	16	2048

Table 26-32. Supported NAND Devices (continued)

Capacity	Device ID	Bus Width	Page Size in KB
2Gb	AAh	8	2048
2Gb	DAh	8	2048
2Gb	BAh	16	2048
2Gb	CAh	16	2048
2Gb	71h	8	512
2Gb	51h	16	512
2Gb	31h	8	512
2Gb	41h	16	512
4Gb	ACh	8	2048
4Gb	DCh	8	2048
4Gb	BCh	16	2048
4Gb	CCh	16	2048
8Gb	A3h	8	2048
8Gb	D3h	8	2048
8Gb	B3h	16	2048
8Gb	C3h	16	2048
16Gb	A5h	8	2048
16Gb	D5h	8	2048
16Gb	B5h	16	2048
16Gb	C5h	16	2048
32Gb	A7h	8	2048
32Gb	B7h	16	2048
64Gb	AEh	8	2048
64Gb	BEh	16	2048

After retrieving parameters from the table, page size and block size are updated based on the fourth byte of the NAND ID data. Because of inconsistency among manufacturers, only devices recognized to be at least 2Gb have these parameters updated. Therefore, the ROM code supports 4-KB page devices, but only if their size, according to the table, is at least 2Gb. Devices smaller than 2Gb have the block size parameter set to 32KB when the page size is 512KB and to 128KB when the page size is 2048KB. Table 26-33 shows the fourth ID data byte encoding used in the ROM code.

Table 26-33. Fourth NAND ID Data Byte

Item	Description	I/O Number							
		7	6	5	4	3	2	1	0
Page Size	1KB							0	0
	2KB							0	1
	4KB							1	0
	8KB							1	1
Block Size	64KB			0	0				
	128KB			0	1				
	256KB			1	0				
	512KB			1	1				

The detection procedure is described in Figure 26-16. If the device ID is not recognized, the ROM code tries to read ID2 from the device; the sequence is described in Figure 26-17. The description of the ID2 data content is summarized in Table 26-34. If the ROM code fails to identify device ID or ID2, it returns with FAIL. When the device is successfully detected, the ROM code changes the GPMC to 16-bit bus width if necessary.

Figure 26-16. NAND Device Detection

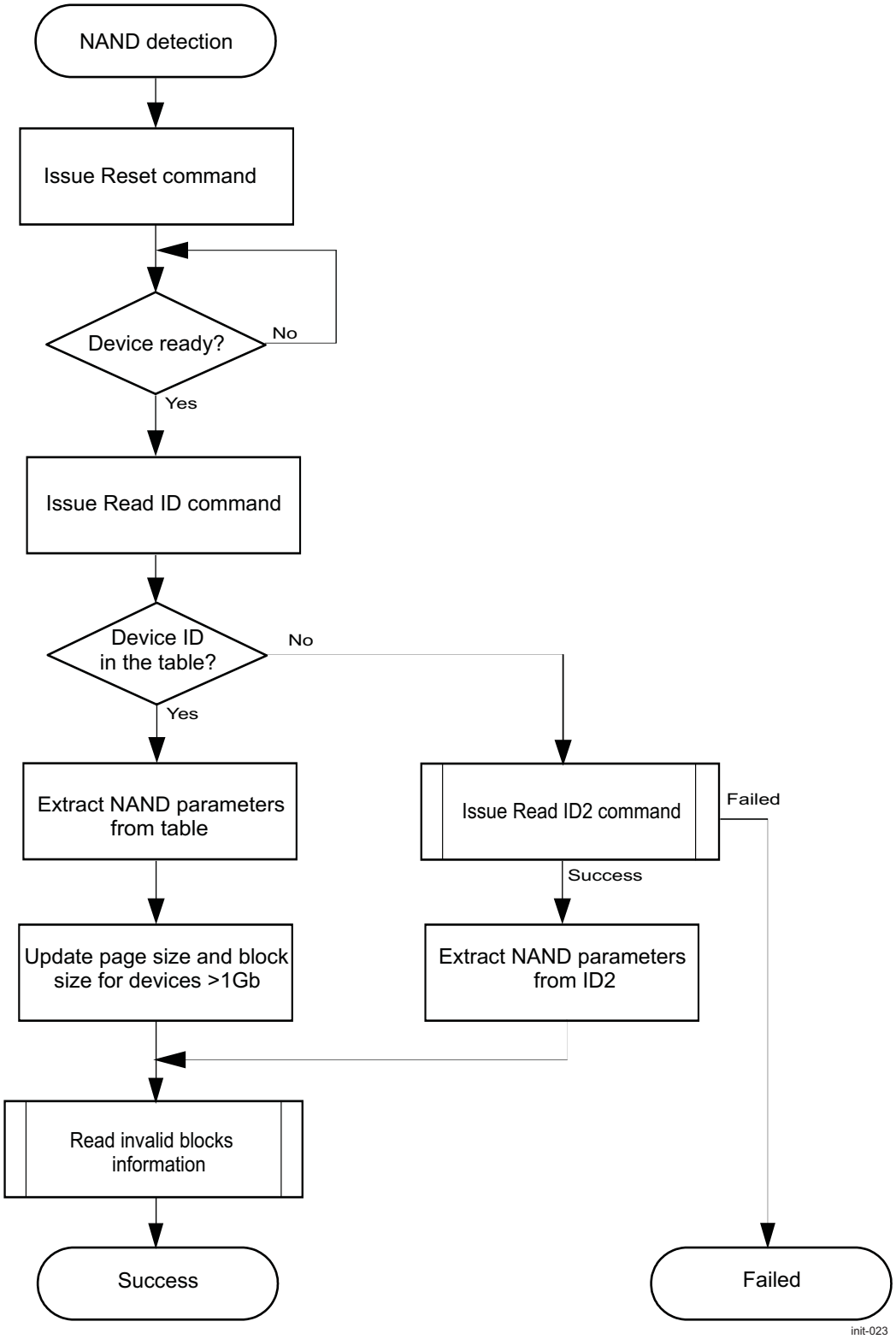


Figure 26-17. NAND ID2 Detection

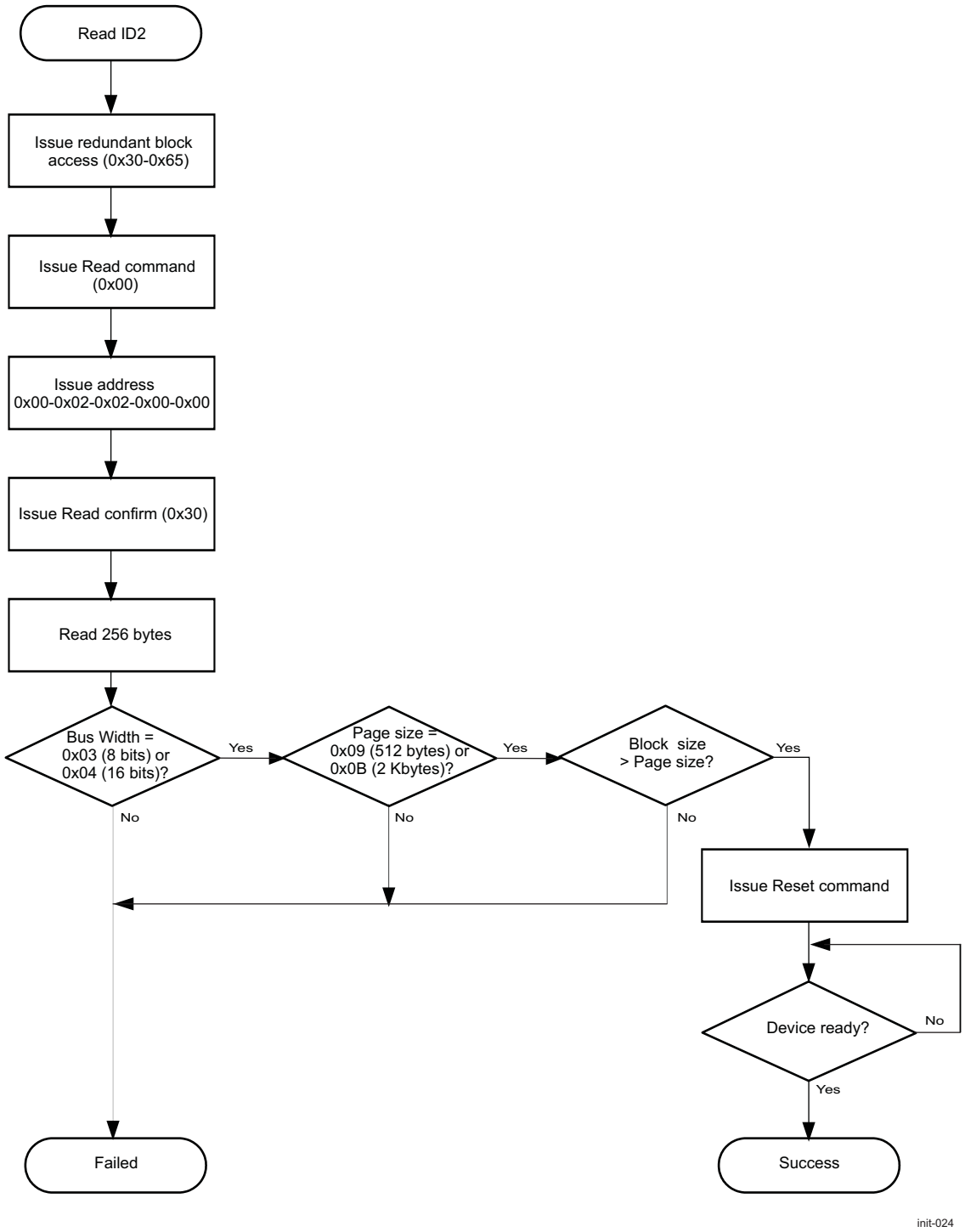


Table 26-34. ID2 Byte Description

Byte Number	Name	Value	Unit	Notes
1	Page size	2X	Bytes	00H = 1B 09H = 512B 0BH = 2KB
2	Block size	2X	Bytes	00H = 1B 0EH = 16KB 11H = 128KB
3	Block count	2X	Pcs	00H = 1 pc 0BH = 2048 pcs 0CH = 4096 pcs
4	Spare size	2X	Bytes	00H = 1B 04H = 16B 06H = 64B
5	Column address	X	Pcs	Higher nibble 1H = 1 column address sequence 2H = 2 column address sequence
	Row address	X	Pcs	Lower nibble 1H = 1 row address sequence 2H = 2 row address sequence
6	ECC type	X	Bit ECC	Higher nibble 0H = No ECC needed 1H = 1-bit ECC 4H = 4-bit ECC
	Bus width	2x	Width	Lower nibble 3H = 8-bit NAND interface 4H = 16-bit NAND interface
7	Number of CEs	X	Pcs	Higher nibble 1H = 1x CE# 2H = 2x CE#
	Cell type	X	Bit/cell	Lower nibble 1H = 1 bit per cell 2H = 2 bits per cell
8	Boot block	X	kB	0H = No boot block 1H = 1KB boot block 2H = 2KB boot block
9	Multiple page prg	X	Pcs	Higher nibble 1H = 1 plane 4H = 4 planes
				For future use
10	Partial prg count	X	Per page	1H = No partial prg allowed 2H = 2 per page
11	Read time maximum			
12	Prg time maximum			
13	Erase time maximum			
252nd 255th	Identification number	X		B2184D7Bh
256th	Register/spec version	XvX		Higher nibble: Major digit Lower nibble: Decimal digit Registers according to spec: 2v0: 20h

- Bad block detection/verification

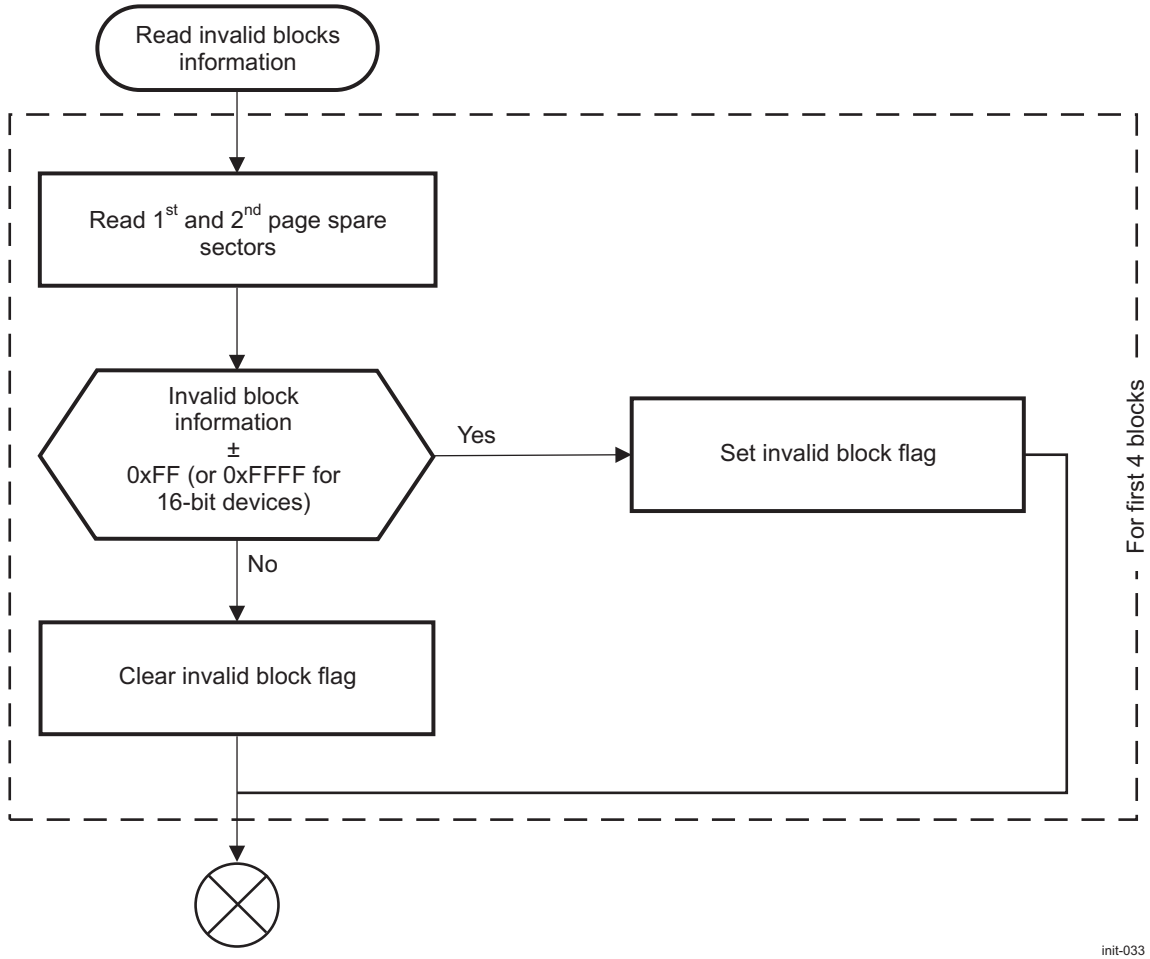
  Invalid blocks contain invalid bits whose reliability cannot be ensured by the manufacturer. These bits are identified in the factory or during the programming and reported in the initial invalid block information in the spare area on the first and second page of each block. Because the ROM code looks for an image in the first four blocks, it detects the validity status of these blocks. Blocks detected as invalid are not accessed later. Block validity status is coded in the spare areas of the first two pages of a block. Table 26-35 lists the validity status coding for the four NAND families.

Table 26-35. Bad Block Mark Locations in NAND Spare Areas

	Small Page NAND	Large Page NAND
8-bit device Block invalid when any byte does not equal FFh	6th byte in 1st page 6th byte in 2nd page	1st byte in 1st page 1st byte in 2nd page
16-bit device Block invalid when any word does not equal FFFFh	1st word in 1st page 6th word in 1st page 1st word in 2nd page 6th word in 2nd page	1st word in 1st page 1st word in 2nd page

Figure 26-18 shows the invalid block detection routine. The routine consists in reading spare areas and checking data according to the conditions listed in Table 26-35. The flags are used internally to give information about the validity of each block.

Figure 26-18. Bad NAND Invalid Block Detection



26.4.7.4.2 Read Sector Procedure

During the booting procedure, the ROM code reads 512-byte sectors from the NAND device. The reading fails in two cases:

- The accessed sector is in a block marked as invalid.

- The accessed sector contains an error that cannot be corrected with ECC.

Pages can contain errors caused by memory alteration. To correct these errors, the ROM code uses ECC, based on Hamming codes for SLC NAND and BCH (Bose, Ray-Chaudhuri, Hocquenghem) code for multilevel cell (MLC) devices. The computed ECC is compared to ECC stored in the spare area of the corresponding page. If there are uncorrectable errors, the ROM code returns with FAIL.

26.4.7.5 OneNAND/Flex-OneNAND

ROM code support for OneNAND/Flex-OneNAND devices has the following characteristics:

- Devices from 512Mb
- The GPMC is the communication interface.
- x16 data bus width only
- Asynchronous protocol and address/data multiplexed mode
- GPMC reset default timings are used.
- The GPMC clock is 48 MHz.
- The device is connected to CS0 mapped to address 0x0800 0000.
- The wait pin signal gpmc_wait0 is not monitored.
- Four physical blocks are searched for image. The block size is 128KB.

The OneNAND/Flex-OneNAND device is a NAND matrix coupled with RAM buffers and a NOR-type interface. ECC correction handling is done automatically by the internal state-machine. The page to be accessed is first loaded in the RAM buffer using memory-mapped registers. Then, the page is read directly from the buffer using a NOR-type interface.

For OneNAND/Flex-OneNAND memory booting, no user intervention is needed. The information in the following subsections is included for debugging. Only the CH, which is not mandatory, lets the user change clock settings and GPMC parameters. Failure in CH copying causes a return to the main booting procedure, which selects the next device for booting.

26.4.7.5.1 Initialization and OneNAND/Flex-OneNAND Detection

The initialization routine for OneNAND/Flex-OneNAND consists of two parts: GPMC initialization and device detection with parameter determination:

- GPMC initialization

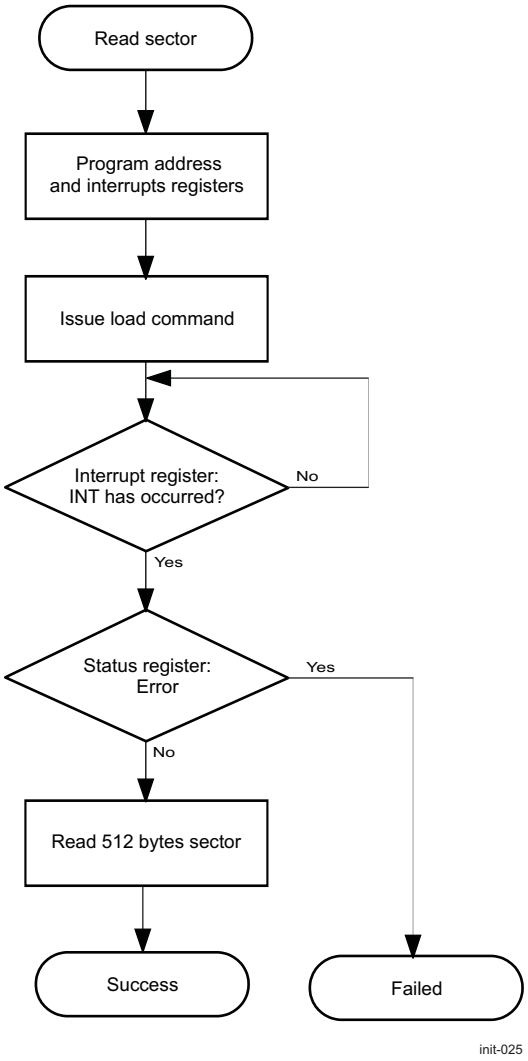
  The ROM code first initializes the GPMC interface the same way as for an XIP memory (that is, asynchronous 16-bit multiplexed mode). Wait signal monitoring is disabled. See Table 26-30.
- Device detection and parameters

  The ROM code identifies a OneNAND/Flex-OneNAND device by reading the device identification data. There are two ways to read identification data: using serial commands and reading from fixed memory mapped registers. The ROM code reads identification data using both methods and compares the result. When the comparison passes, the ROM code assumes that the OneNAND/Flex-OneNAND device is connected. If the device is successfully recognized, the ROM code reads the device configuration (amount and size of data buffers) and configures it for asynchronous mode (default).

26.4.7.5.2 OneNAND/Flex-OneNAND Read Sector Procedure

When booting requests a sector from the OneNAND/Flex-OneNAND device, the ROM code issues the load operation, which transfers the content of the requested sector to the data buffer RAM. The ROM code waits until the operation completes, polling the OneNAND/Flex-OneNAND interrupt register. The status register is then checked and the ROM code returns FAIL if the operation completes with an error. Otherwise, the data buffer RAM is copied to the destination buffer. Figure 26-19 shows this procedure.

Figure 26-19. OneNAND/Flex-OneNAND Read Sector



26.4.7.5.3 Flex-OneNAND Support Limitations

As described in Section 26.4.7.5.1, *Initialization and OneNAND/Flex-OneNAND Detection*, the ROM code checks only the coherency of the device ID obtained by two different methods. There is no table of supported OneNANDs/Flex-OneNANDs in the ROM code. This removes any dependency of the ROM code on device IDs from different manufacturers. However, the driver works with the following assumptions:

- Flash memory can be OneNAND. Only a 1-bit error is tolerated. Any error greater than 1 bit generates a FAIL.
- Page size is assumed to be 2048 bytes, divided into four sectors of 512 bytes. In case of Flex-OneNAND, the pages are 4096 bytes, divided into eight sectors. As a consequence, users must program only the first sectors (sectors 0, 1, 2, 3) of each page. Sectors 4, 5, 6, and 7 of each page are not considered by the driver.

26.4.7.6 MMC/SD Cards

The ROM code supports MMC/SD cards, with some limitations:

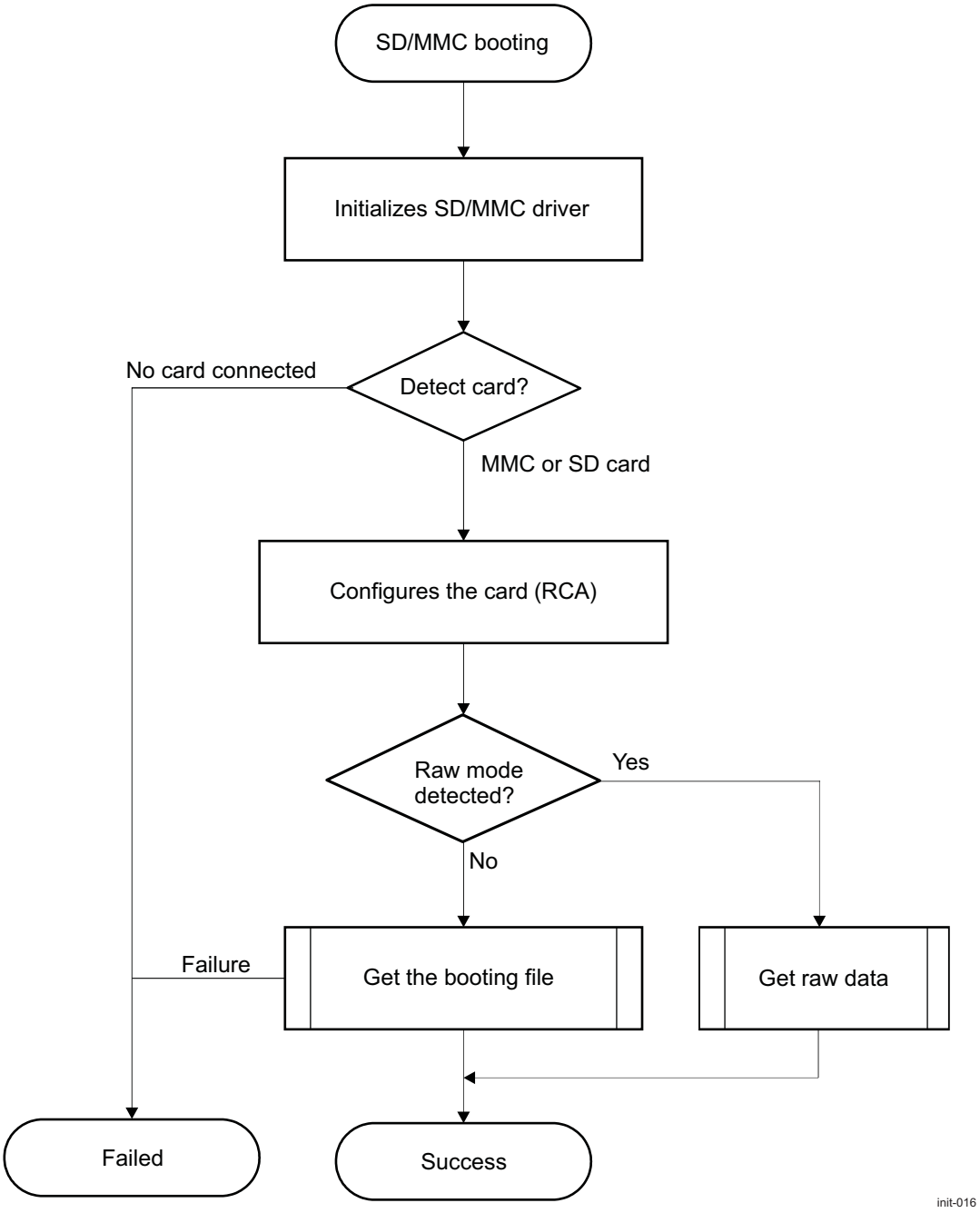
- Supports MMC/SD cards compliant with the *Multimedia Card System Specification v4.2* from the MMCA Technical Committee and the *SD I/O Card Specification v2.0* from the SD Association. Includes high-capacity (size >2GB) cards: HC-SD and HC MMC
- 3-V power supply, 3-V I/O and 1.8-V I/O voltages on POR 1
- Supports eMMC/eSD (1.8-V I/O voltage and 3.0-V Core voltage) on port 2. The external transceiver mode on port 2 is not supported.
- Initial 1-bit MMC mode, 4-bit SD mode
- Clock frequency:
  - Identification mode: 400 kHz
  - Data transfer mode: 20 MHz
- Only one card connected to the bus
- Raw mode, image data read directly from card sectors
- FAT12/16/32 support, with or without a master boot record (MBR).
- For a FAT (12/16/32)-formatted memory card, the booting file must not exceed 128 KB.
- For a raw-mode memory card, the booting image must not exceed 128 KB.

For MMC memory booting, no user intervention is needed. The information in the following subsections is included for debugging. Failure in MMC/SD card detection causes a return to the main booting procedure, which selects the next device for booting.

The HS MMC/SD/SDIO host controllers (MMCHS) handle the physical layer, while the ROM code handles the simplified logical protocol layer (read-only protocol). A limited range of commands is implemented in the ROM code. The MMC/SD specification defines two operating voltages for standard or HS cards. The ROM code supports only standard operating voltage range (3 V) (both modes supported).

The ROM code reads a booting file from the card file system and boots from it. Figure 26-20 shows the complete procedure.

Figure 26-20. MMC/SD Booting



26.4.7.6.1 Connectivity Constraints

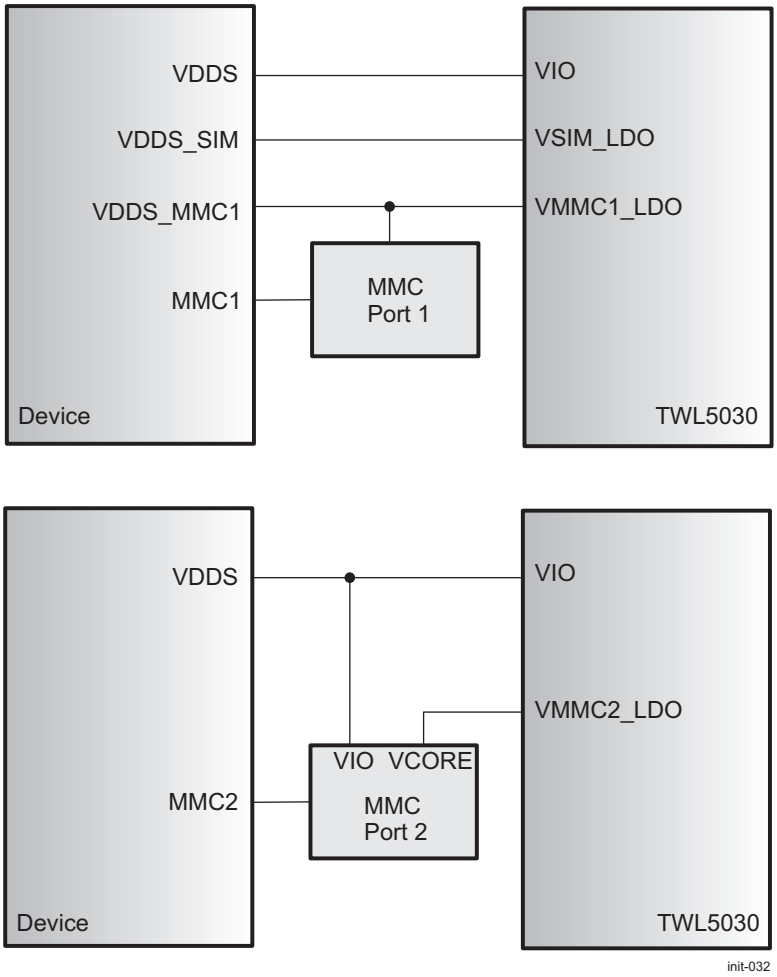
MMC port 1 can support I/O in two modes, 1.8 V or 3.0 V, provided the correct voltage supply is connected to the VDDS_MMC1 pin. Hardware provides the mechanism to detect the presence of 3.0 V on VDDS_MMC1. If the hardware does not detect 3.0 V, the ROM code assumes 1.8-V voltage is present and configures the I/O pads accordingly.

To enable the hardware to generate the correct levels on the MMC interface, the following connectivity constraints must be respected.

26.4.7.6.1.1 Booting From MMC/SD Memory in Case TWL5030 is Implemented

Before performing a memory access, the ROM code tries to detect the presence of a TWL5030 power management device on the I2C bus. If one is detected, the ROM code assumes the connections shown in Figure 26-21 are present.

Figure 26-21. TWL5030 Connectivity Constraints to Support MMC/SD/eMMC/eSD Booting on MMC Port 1 and MMC Port 2



The ROM code configures the TWL5030 through I2C to set the following:

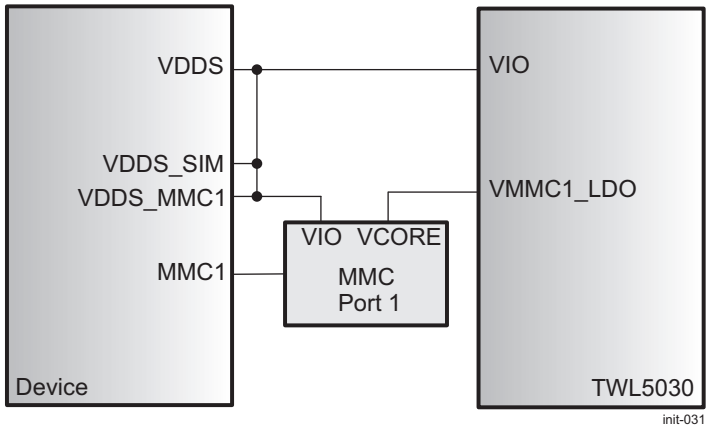
- VMMC1_LDO is set to 3.0 V when booting from MMC port 1 is requested from the sys-boot pins.
- VSIM_LDO is set to 3.0 V only when a configuration header configuring an 8-bit bus is embedded in the booting image. If an 8-bit bus is not required at boot time, it is not mandatory to connect VSIM_LDO to VDDS_SIM.
- VMMC2_LDO is set to 3.0 V when a booting from MMC port 2 is requested from the sys-boot pins. Users must furnish VIO = 1.8 V to the MMC memory, respecting the timing sequence imposed by the memory.

26.4.7.6.1.2 Booting From eMMC/eSD Memory From MMC Port 1 When TWL5030 is Implemented

ROM code can boot from the embedded multimedia card/embedded SD card (eMMC/eSD) on port 1. Most of these memories require two power supplies. One power supply is required for the I/O interface (CMD, CLK, DAT[7:0]), and the other is required for the memory core. The VIO voltage is typically 1.8 V, and the VCORE voltage is 3.0 V.

For eMMC/eSD booting with two power supplies, the implementation shown in Figure 26-22 is required.

Figure 26-22. TWL5030 Connectivity Constraints to Support MMC/SD/eMMC/eSD Booting on MMC Port 1



26.4.7.6.1.3 Booting When TWL5030 is Not Implemented

When TWL5030 is not implemented, the ROM code assumes the following:

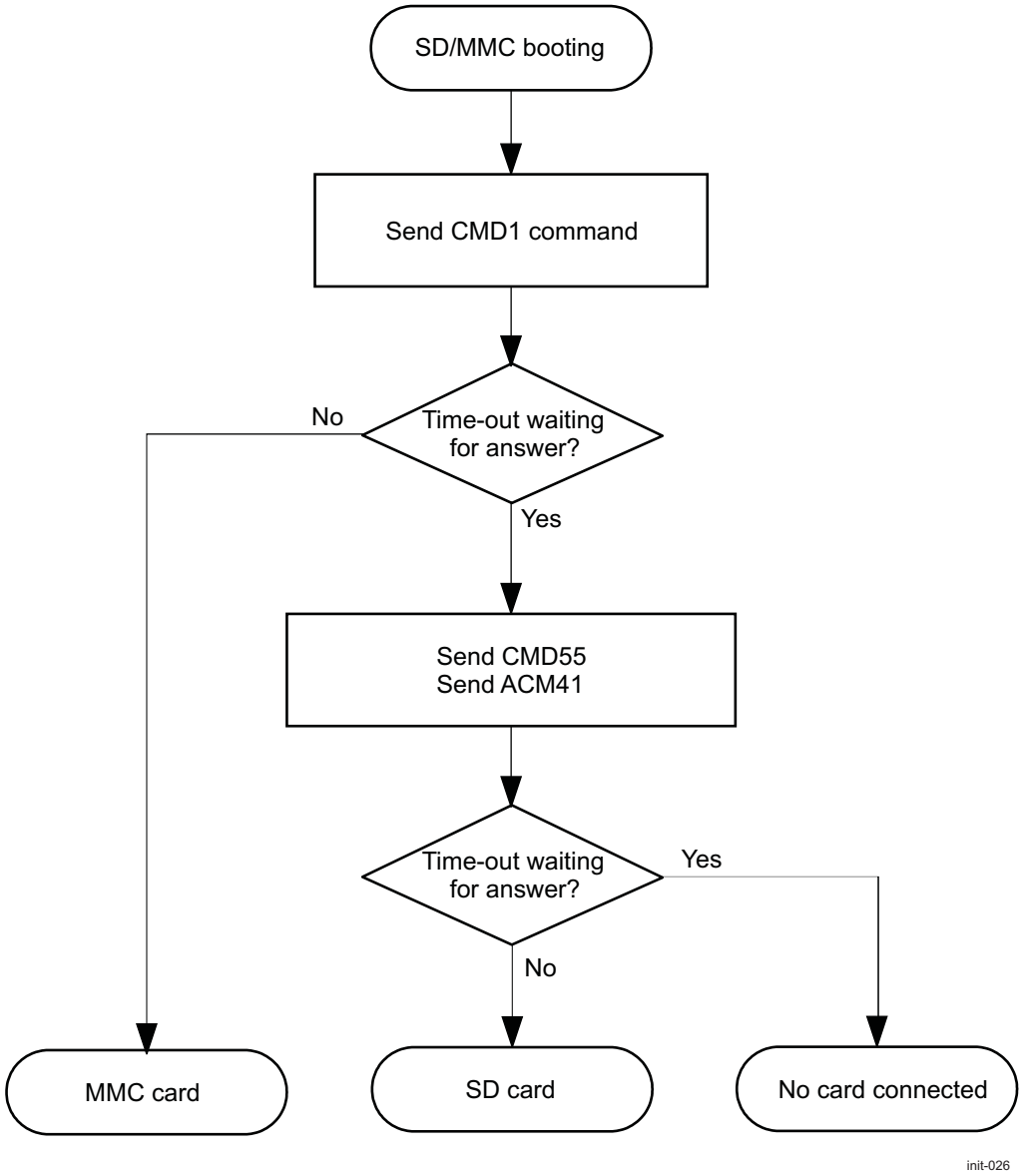
- The MMC/SD card memory connected to MMC port 1 is powered by a 3.0-V power supply after any POR or software or warm reset.
- The VDDS_MMC1 pin of the device is connected to a 3.0-V power supply that is the same as the one that powers the MMC/SD memory on MMC port 1.
- The VDDS_VSIM pin of the device is connected to a power supply that is the same as the one that powers the MMC/SD memory, when a configuration header setting the bus width to 8 bits is implemented in the booting image.
- The eMMC/eSD memory connected to MMC port 2 has its VIO power domain connected to a 1.8-V power supply after any POR or software or warm reset.
- The eMMC/eSD memory connected to MMC port 2 has its VCORE power domain connected to a 3.0-V power supply after any POR or software or warm reset.

26.4.7.6.2 Initialization and MMC/SD Card Detection

The ROM code initializes the card connected on interface 1 using the standard high-voltage range (3.0 V) if a card is plugged in. If no card is present, the ROM code goes to the next booting device. The standard identification process and relative card address (RCA) assignment are used. However, the ROM code searches for only one card connected on the bus. This is done using the CMD line common to the SD and MMC cards. The MMC and SD standards describe this phase as the initialization phase. They differ in the first commands, CMD1 and ACMD41. The ROM code uses this command difference to differentiate between MMC and SD cards; that is, CMD1 is supported only by MMC and ACMD41 is supported only by SD. The ROM code first sends a CMD1 to the card and gets an answer only if an MMC card is connected. If no answer is received, ACMD41 (a combination of CMD55 and ACMD41) is sent, and an answer is expected from an SD card. If no answer is received, no cards are connected and the ROM code exits MMC/SD booting with FAIL.

Figure 26-23 shows the MMC/SD detection procedure.

Figure 26-23. MMC/SD Detection Procedure



26.4.7.6.3 Read Sector Procedure

- Raw mode

  In raw mode, an image can be at offset 0 or 128KB and must not be bigger than 128KB. Raw mode is detected by reading sector 0 and sector 256. The content of these sectors is verified for presence of a TOC structure. In the case of a GP device, a configuration header (CH) must be located in the first sector followed by a GP header. The CH can be void (contains only a CHSETTINGS item for which the Valid field is 0), as described in Section 26.4.8.2, *CH*. Image data is read directly from continuous sectors of a card. If raw mode is not detected, file system mode is checked.

- File system handling

  The sector read procedure uses the standard MMC/SD read data procedure. The sector address is generated based on the booting memory file map collected during initialization. Thus, the ROM code can address sectors freely in the booting file space.

26.4.7.6.4 File System Handling

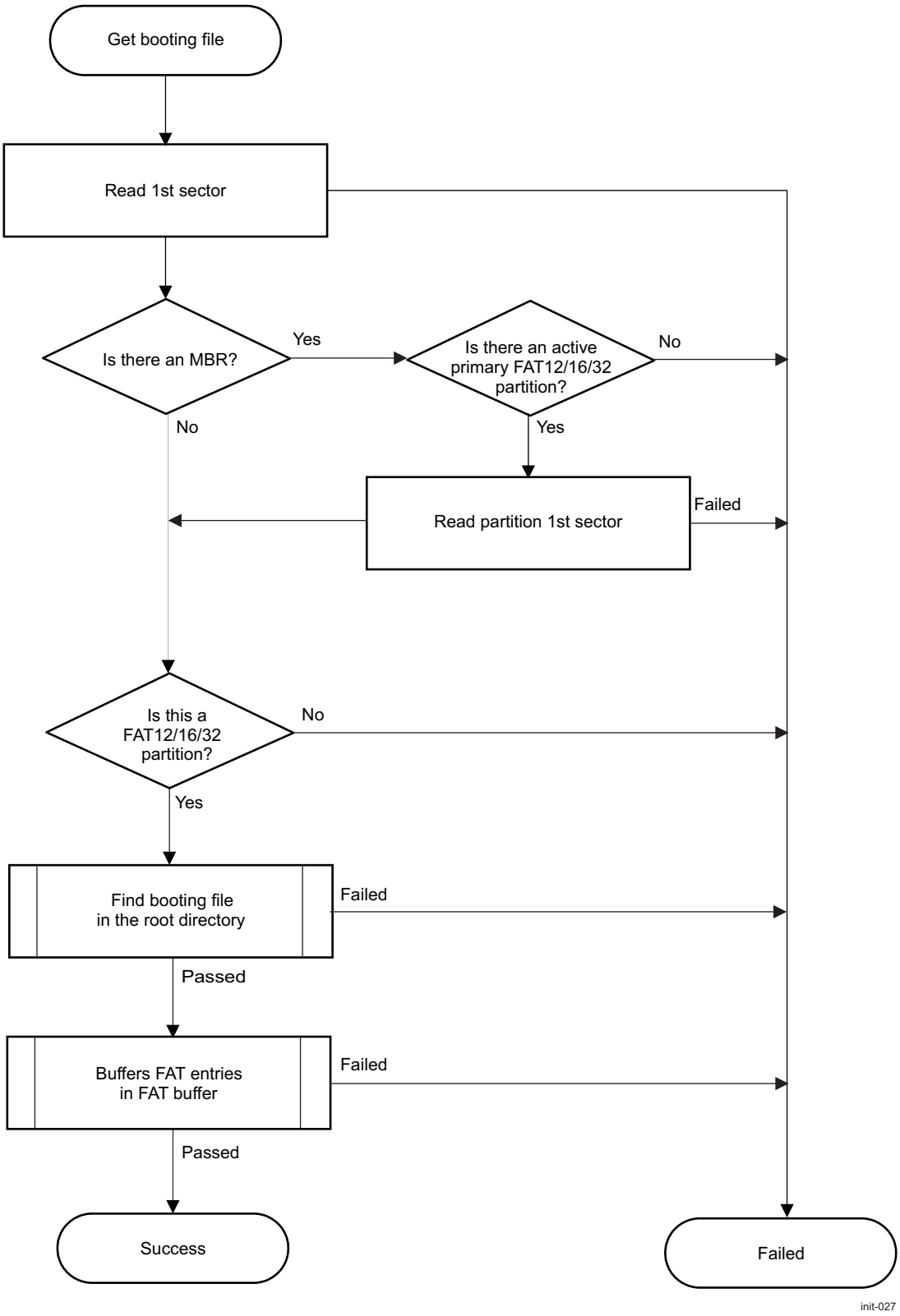
The MMC/SD cards can hold a file system that ROM code reads. The image used by the booting procedure is taken from a booting file named MLO. This file must be in the root directory on an active primary partition of type FAT12/16 or FAT32.

An MMC/SD card can be configured as floppy-like or hard-drive-like:

- When acting like a floppy, the content of the card is a single FAT12/16/32 file system without an MBR holding a partition table.
- When acting like a hard drive, an MBR is present in the first sector of the card. This MBR holds a table of partitions, one of which must be FAT12/16/32, primary, and active.

According to the *MultiMediaCard FAT16 File System Specification* from the MMCA Technical Committee, the card should always hold an MBR, except for MMC cards using a floppy-like file system. However, depending on the operating system used, the MMC/SD card is formatted with or without partition(s) (using an MBR). The ROM code supports both types: floppy-like or hard-drive-like. The ROM code retrieves a map of the booting file from the FAT table. The booting file map is a collection of all FAT table entries related to the booting file (a FAT entry points to a cluster holding part of the file). The booting procedure uses this map to access any 512-byte sector in the booting file without involving the ROM code FAT module. Figure 26-24 shows the complete process.

Figure 26-24. SD/MMC Booting



26.4.7.6.4.1 MBR and FAT File System

This paragraph describes functionalities used by the ROM code to recognize whether an MBR with a FAT is used. It is not intended to fully describe the MBR and the FAT file system detection and reading procedure. The ROM code can detect FAT12/16/32 allocation table types. It cannot boot on devices with NTFS or Linux FS partitions. Some memory devices that support file systems can be formatted with or without MBR; therefore, the first task of the ROM code is to detect whether the device is holding an MBR in the first sector.

The MBR is the first sector of a memory device. It consists of executable code, four partition entries, and one signature. The aim of such a structure is to divide the hard disk in partitions used primarily to boot different systems (for instance, Microsoft Windows™). This structure is described in Table 26-36; partition table entry is described in Table 26-37.

Table 26-36. Master Boot Record Structure

Offset	Length (Bytes)	Entry Description
0000h	446	Optional code
01BEh	16	Partition table entry
01CEh	16	Partition table entry
01DEh	16	Partition table entry
01EEh	16	Partition table entry
01FEh	2	Signature (= 0xAA55)

Table 26-37. Partition Table Entry

Offset	Length (Bytes)	Entry Description	Value
0000h	1	Partition state	00h: Inactive 80h: Active
0001h	1	Partition start head	Hs
0002h	2	Partition start cylinder and sector	Cs[7:0]–Cs[9:8]–Ss[5:0]
0004h	1	Partition type	01h: FAT12 04h, 06h, 0Eh: FAT16 0Bh, 0Ch, 0Fh: FAT32
0005h	16	Partition end head	He
0006h	2	Partition end cylinder and sector	Ce[7:0]–Ce[9:8]–Se[5:0]
0008h	4	First sector position relative to the beginning of media	LBAs = Cs.H.S+ Hs.S+ Ss–1
000Ch	4	Number of sectors in partition	LBAe = Ce.H.S+ He.S+ Se–1 Nb s= LBAe–LBAs + 1

SD/MMC booting consists of the following steps:

Step 1. MBR detection

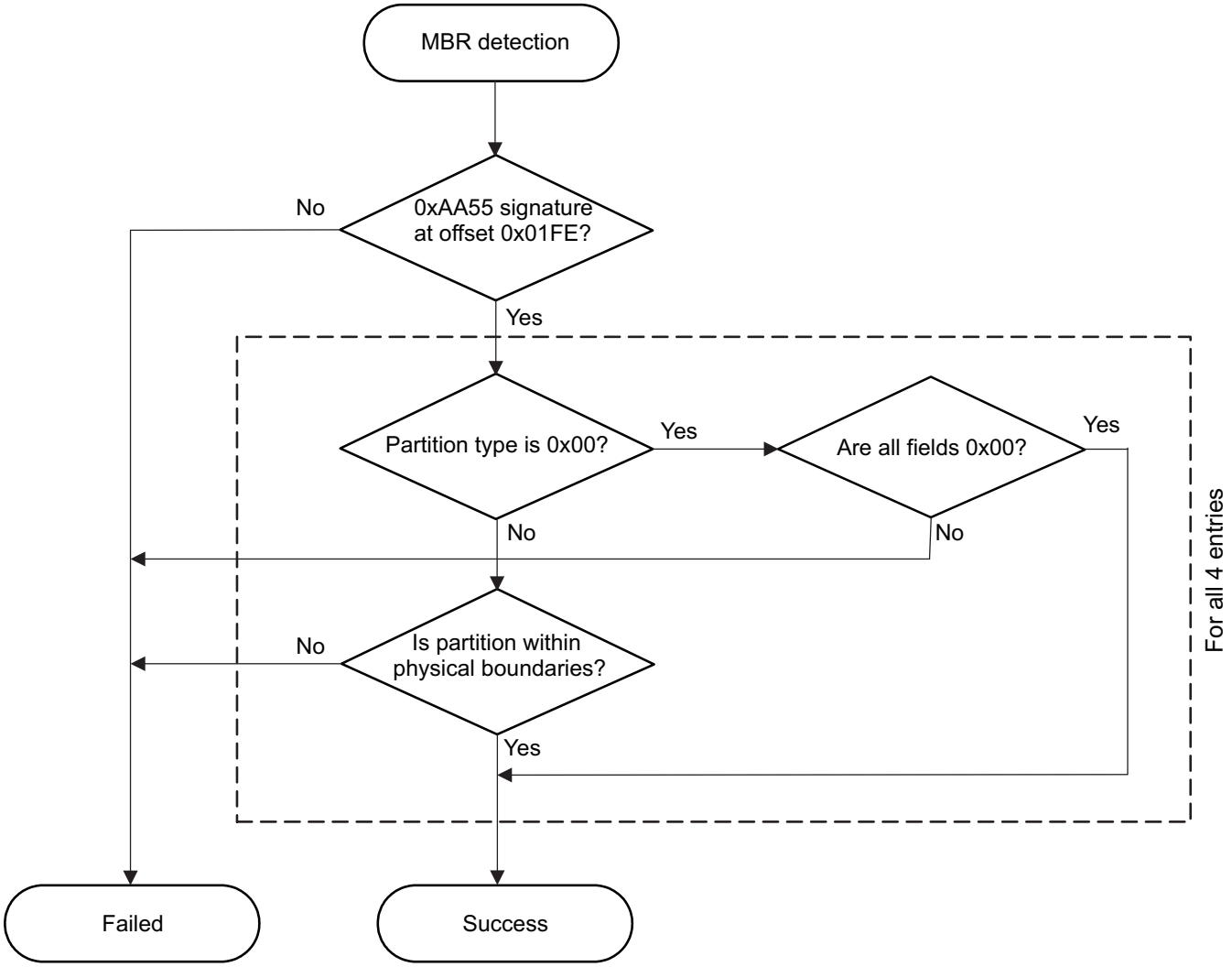
The ROM code first checks whether the MBR signature is present and then it searches an active FAT12/16/32 partition in all four MBR partition entries, based on the Type field. If the MBR entries are not valid or if no usable partition is found, the ROM code returns to the booting procedure with FAIL. The extended partitions are not checked; the booting file must reside in a primary partition. Each partition entry is checked:

a. If its type is set to 00h, all fields in the entry must be 00h.

b. The partition is checked to be within physical boundaries (that is, the partition is inside and it fits the total physical sectors).

See Figure 26-25 for more information about MBR detection.

Figure 26-25. MBR Detection Procedure



Step 2. Get the MBR partition.

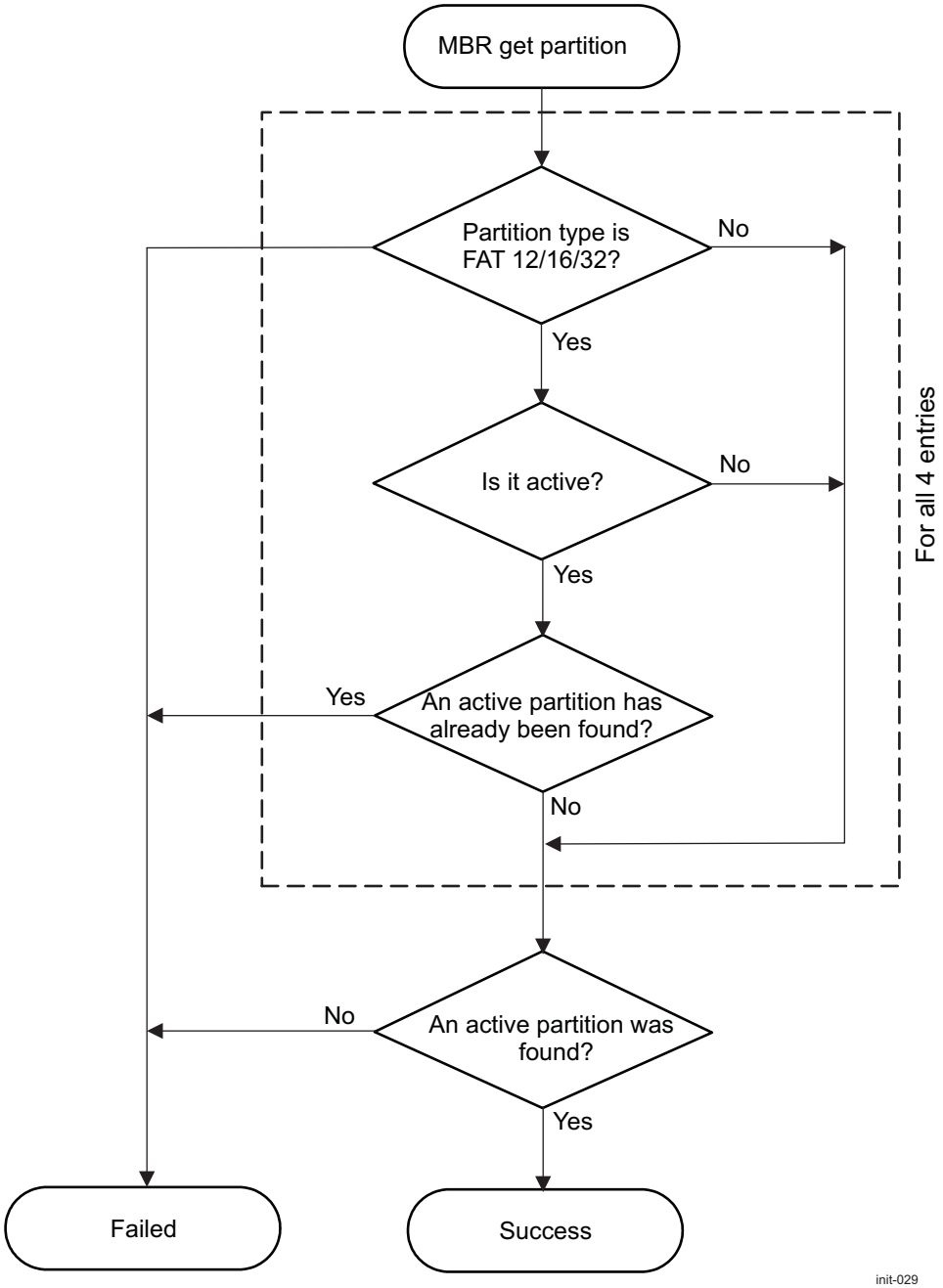
Once identified, the ROM code gets the partition using the procedure described in Figure 26-25. The partition type is checked to be FAT12/16 or FAT32. Its state must be 00h (inactive) or 80h (active.) The ROM code returns with FAIL if no active primary FAT12/16/32 is found, or if there is more than one active partition, the test fails. If an active partition is found, its first sector is read and used later. If no MBR is present (in case of a floppy-like system), the first sector of the device is read and used later. The read sector is checked to be a valid FAT12/16 or FAT32 partition. If this fails, if another partition type is used (for instance, Linux® FS) or if the partition is not valid, the ROM code returns with FAIL.

The FAT file system consists of several parts:

- Boot sector, which holds the BIOS parameter block (BPB). Not all are used by the ROM code.
- FAT, which describes the use of each cluster of the partition
- Data area, which holds the files, directories, and root directory (for FAT12/16, the root directory has a specific fixed location).

To check whether a sector holds a valid FAT12/16/32 partition, many fields of the boot sector (used by all FAT types) that must have specific values are checked. Figure 26-26 shows more information about getting the MBR partition.

Figure 26-26. Get MBR Partition



Step 3. Find the booting file.

When a partition is found, the root directory entries are searched for a booting file named MLO in the root directory of the FAT12/16/32 file system. The file is not searched in any other location. For a FAT12/16 file system, the root directory has a fixed location, which is cluster 0. For a FAT32 file system, its cluster location is given by BPB_RootClus. The formula to find the sector number (relative to device sector 0, not partition sector 0) of a cluster is given by:

$$Cluster_{sector} = BPB_HiddSec + BPB_RsvdSecCnt + BPB_NumFATs \cdot BPB_FATSz + Cluster \cdot BPB_SecPerCLus$$

init-E001

Note: BPB_FatSz is BPB_FatSz16 for FAT12/16 or BPB_FatSz32 for FAT32.

Note: The BPB_HiddSec field can contain 0 even though the FAT file system is somewhere other than on sector 0 (floppy-like). The ROM code uses the partition offset taken from the MBR instead of this field, which can be wrong. If no MBR is found (floppy-like), the value 0 is used.

Each entry in the root directory is 32 bytes and holds information about the file (the filename, date of creation, rights, cluster location, etc.). (See Table 26-38.)

Table 26-38. FAT Directory Entry

Offset	Length (Bytes)	Name	Description
0000h	11	DIR_Name	Short Name (8 + 3)
000Bh	1	DIR_Attr	File Attributes: ATTR_READ_ONLY 01h ATTR_HIDDEN 02h ATTR_SYSTEM 04h ATTR_VOLUME_ID 08h ATTR_DIRECTORY 10h ATTR_ARCHIVE 20h ATTR_LONG_NAME ATTR_READ_ONLY | ATTR_HIDDEN ATTR_SYSTEM | ATTR_VOLUME_ID
000Ch	1	DIR_NTRes	Reserved. Set to 00h.
000Dh	1	DIR_CrtTimeTenth	Millisecond stamp at file creation
000Eh	2	DIR_CrtTime	Time file was created.
0010h	2	DIR_CrtDate	Date file was created.
0012h	2	DIR_LstAccDate	Last access date
0014h	2	DIR_FstClusHi	High word of the first cluster number of this entry
0016h	2	DIR_WrtTime	Time of last write
0018h	2	DIR_WrtDate	Date of last write
001Ah	2	DIR_FstClusLo	Low word of the first cluster number of this entry
001Ch	4	DIR_FileSize	File size in bytes

The ROM code checks each entry in the root directory until either the booting file is found or the entry is empty (first byte is 00h), or when the end of the root directory is reached. Entries with the ATTR_LONG_NAME attribute (LFN) and first byte at E5h (erased file) are ignored. When found, the first cluster offset of the file is read from the DIR_FstClusHi/DIR_FstClusLo fields. There is a slight difference between FAT12/16 and FAT32 when handling the root directory. On FAT12/16, this directory has a fixed location (see above) and length fixed by BPB_RootEntCnt, which is the total of 32-byte entries. Handling this directory is therefore straightforward. On FAT32, the root directory is like a standard file. The FAT must be used to retrieve each sector of the directory. The way in which the FAT is handled is described in the following paragraph.

Step 4. Buffer FAT entries in the FAT buffer

When the booting file is found, the ROM code reads the FAT and buffers the singly-linked chain of clusters in a FAT buffer used by booting to access the file directly, sector by sector. For FAT12/16 and for FAT32, multiple copies of the FAT exist (ROM code supports only two copies), after the boot sector.

$$FATn_{sector} = BPB_HiddSec + BPB_RsvdSecCnt + BPB_FatSz \cdot n$$

init-E002

The size of the FAT buffer is given by BPB_FATSz16 or BPB_FATSz32. The ROM code checks each copy of the FAT if they are identical. If the values are different, the ROM code uses the value from the last FAT copy. With the FAT32 file system, the copy system can be disabled according to a flag in BPB_ExtFlags[7]. If this flag is set, the FAT BPB_ExtFlags[3:0] bit field is used. In this case no verification is made by the ROM code with other copies of FAT.

The FAT is a simple array of values, each referring to a cluster located in the data area. One entry of the array is 12, 16, or 32 bits, depending on the file system in use. The value in an entry defines whether the cluster is being used or not, and if another cluster must be considered. This creates a singly-linked chain of clusters defining the file. The meaning of an entry is described in Table 26-39.

Note: For compatibility, cluster 0 and cluster 1 are not used for files, and those entries must contain FF8h and FFFh (for FAT12), FFF8h and FFFFh (for FAT16), and ?FFFFFF8h and ?FFFFFFFh (for FAT32).

Table 26-39. FAT Entry Description

FAT12	FAT16	FAT32	Description
000h	0000h	?0000000h	Free cluster
001h	0001h	?0000001h	Reserved cluster
002h-FEFh	0002h-FFEFh	00000002h-?FFFFFEFh	Used cluster; value points to next cluster
FF0h-FF6h	FFF0h-FFF6h	?FFFFFF0h-?FFFFFF6h	Reserved values
FF7h	FFF7h	?FFFFFF7h	Bad cluster
FF8h-FFFh	FFF8h-FFFFh	?FFFFFF8h-?FFFFFFFh	Last cluster in file

Note: FAT32 uses only bits [27:0]; the upper 4 bits are usually 0 and must be left untouched.

When accessing the root directory for FAT32, the ROM code starts from the root directory cluster entry and follows the linked chain to retrieve the clusters.

When the booting file has been found, the ROM code buffers each FAT entry corresponding to the file in a sector way. This means each cluster is translated to one or several sectors, depending on how many sectors are in a cluster (BPB_SecPerClus). This buffer is used later by the booting procedure to access the file.

26.4.7.7 DiskOnChip

The ROM code support for DiskOnChip devices has the following characteristics:

- DiskOnChip H3
- 1.8-V power supply
- Density from 256 Mb

The DiskOnChip contains an XIP part that can hold a boot image. In this case, the DiskOnChip is connected just like a regular NOR device, and uses the same procedure as for NOR, described in Section 26.4.7.3, *XIP Memory*.

If booting determines that it must boot from DOC, the ROM code initializes the GPMC using standard asynchronous, 16-bit, multiplexed mode. The DOC device is then detected using the M-System driver. This proprietary driver handles all DiskOnChip protocols and identification. If the device cannot be identified, the ROM code returns a failed code. Otherwise, booting directly requests sectors from the M-System driver.

26.4.8 Image Format

26.4.8.1 Overview

An image has two major parts:

- An optional CH
- Software to execute

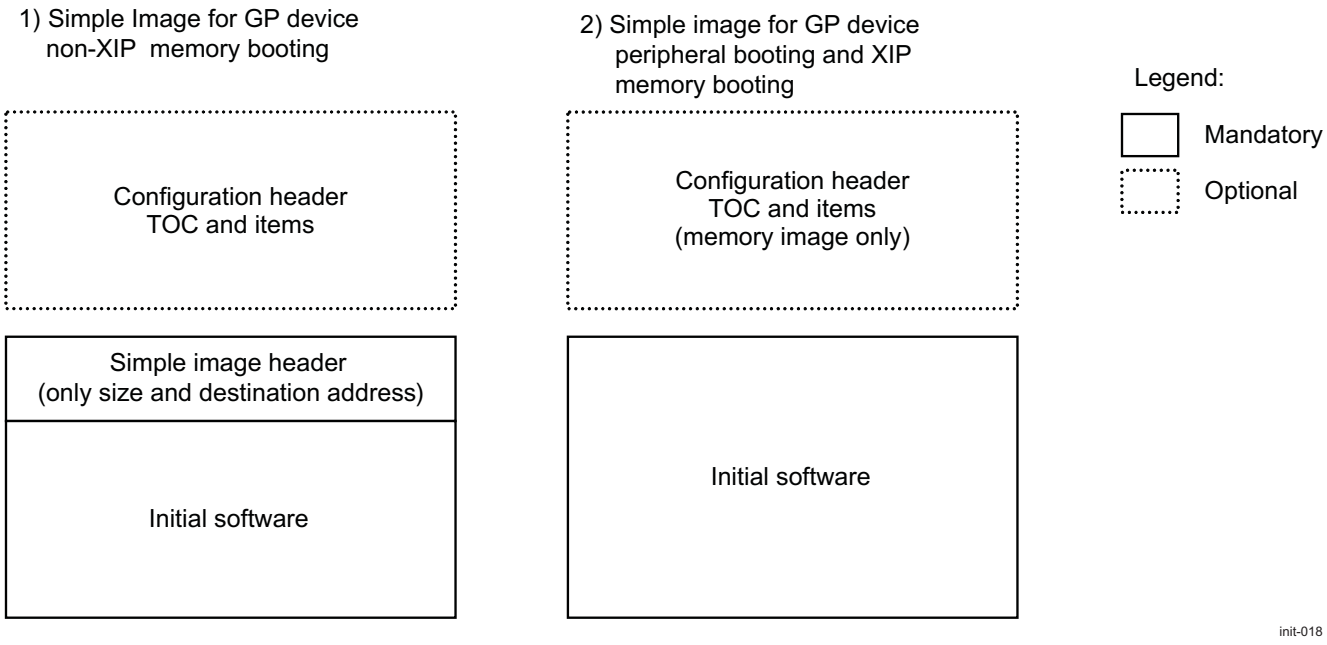
The CH, which is optional, can contain several parameters set by users to speed up booting. It is further described in the next section.

The mandatory part contains the software that is loaded into the memory and executed.

An overview of the image formats is shown in Figure 26-27. There are two image types:

- GP non-XIP memory booting: This image type is used for memories that require shadowing. The image must begin with a simple header that contains information about the size to copy and the destination. This format is described in Section 26.4.8.3, *Image Format for GP Devices*.
- GP XIP memory booting and peripheral booting: A GP image on XIP memory contains only code. The first sector can contain CH. The GP peripheral booting image contains only code.

Figure 26-27. Image Format



26.4.8.2 CH

The CH is optional and is required only if the customer wants to use settings different from the ROM code defaults (for example, clock frequencies, SDRAM/double-data rate [DDR] SDRAM settings, GPMC settings). The CH can be present only when booting from a memory-type device (for example, CH is not supported when booting from UART or USB). Therefore, the CH contains only settings for memory booting.

The CH can contain up to four parts:

- Settings: Clock settings (mandatory)
- RAM: SDRAM/DDR SDRAM interface settings
- FLASH: Flash interface (GPMC) settings
- MMC/SD: MMC/SDIO interface settings

The beginning of the CH is a table of contents (TOC) pointing to each item. This is described in Figure 26-28. Each TOC item is a simple structure described in Table 26-40. The complete CH (CH TOC and items) should fit in a 512-byte sector.

The ROM code identifies the presence of a CH by reading the first TOC item if it contains a known string (CHSETTINGS, CHRAM, etc.). Then, the TOC is identified and searched until a 0xFFFFFFFF offset is found. The CH is read and parameters are executed sequentially.

Figure 26-28. CH Format

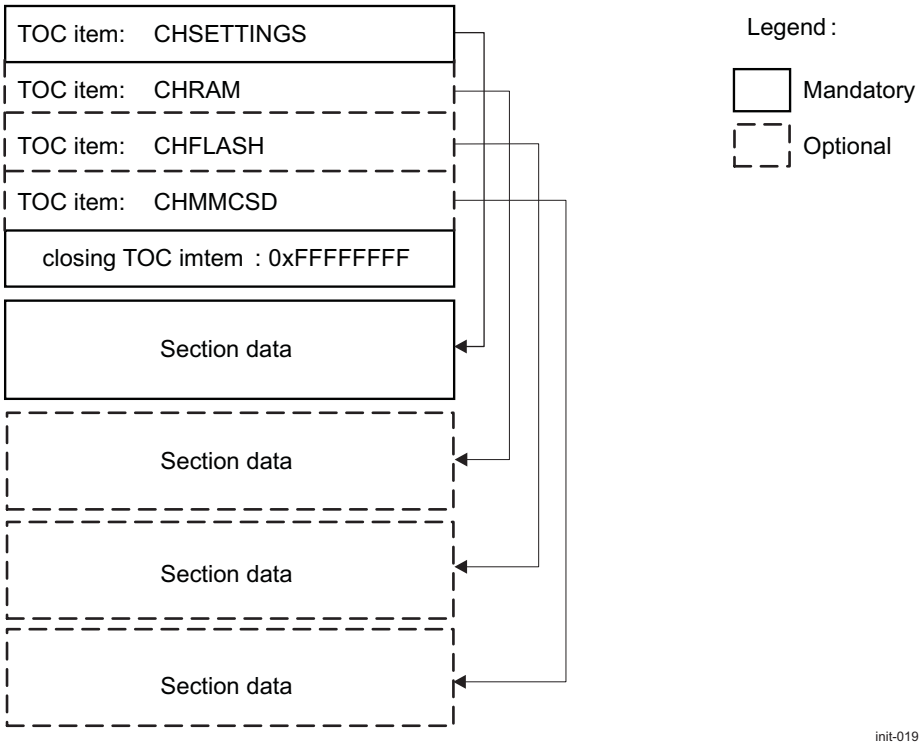


Table 26-40. CH TOC Item

Offset	Field	Size (Bytes)	Description
0x0000	Start	4	Offset from the start address of TOC to the actual address of a section
0x0004	Size	4	Size of a section
0x0008	Reserved	4	Unused
0x000C	Reserved	4	Unused
0x0010	Reserved	4	Unused
0x0014	Filename	12	12-character name of a section, including the zero (\0) terminator

26.4.8.2.1 CHSETTINGS

The CHSETTINGS configuration header contains settings specific to the clock system. The ROM code configures the OMAP clocking to some default settings as described in Section 26.4.4.2, *Clocking Configuration*. The CH CHSETTINGS section contains a method to override the ROM code default clocking settings.

The fields are described in Table 26-41. The clocking procedure, as well as the clocking setting structure are described in Section 26.4.4.4, *Software Booting Configuration*.

The peripherals used during booting, UART and USB, are fed with 96 MHz and 48 MHz. This cannot be changed in the configuration header as it is only available for memory booting.

Table 26-41. CHSETTINGS

Offset	Field	Description
0000h	Section key	Key used for section verification: C0C0C0C1h

Table 26-41. CHSETTINGS (continued)

Offset	Field	Description
0004h	Valid	Enables/disables the section 00h – Disable Other – Enable
0005h	Version	Configuration header version 0001h Others – For future use
0006h	Reserved	
0008h	Clocking settings	Described in Table 26-12, *Software Booting Configuration Structure*, starting from the FLAGS field.

26.4.8.2.2 CHRAM

The CHRAM configuration header contains settings specific to SDRAM memory controller (SDRC). The fields are described in Table 26-42. The ROM code does not configure the SDRC by default, as it cannot assume any external SDR/DDR SDRAM type. Refer to the SDRC section of Chapter 11 for more information.

Table 26-42. CHRAM

Offset	Field	Description CH – used in CH process CR – used in context restore	
0000h	Section key	Key used for section verification: C0C0C0C2h.	
0004h	Valid	Enables/disables the section: 00h – Disable Others – Enable	
0005h	Reserved		
0008h	SDRC_SYSCONFIG (LSB)		CR
000Ah	SDRC_CS_CFG (LSB)	CH	CR
000Ch	SDRC_SHARING (LSB)	CH	CR
000Eh	SDRC_ERR_TYPE (LSB)		CR
0010h	SDRC_DLLA_CTRL (LSB)	CH	CR
0012h	SDRC_DLLA_CTRL (MSB)	CH	CR
0014h	Reserved, write 0s for future compatibility.		CR
0016h	Reserved, write 0s for future compatibility.		CR
0018h	SDRC_POWER (LSB)		CR
001Ah	SDRC_POWER (MSB)		CR
001Ch	Memory type (LSB)	CH 0 = SDRAM 1 = Low power SDR 2 = DDR 3 = Mobile DDR 4 = Unknown	CR
001Eh	Must be 0	CH	CR
0020h	SDRC_MCFG_0 (LSB)	CH	CR
0022h	SDRC_MCFG_0 (MSB)	CH	CR
0024h	SDRC_MR_0 (LSB)	CH	CR
0026h	SDRC_EMR1_0 (LSB)	CH	CR
0028h	SDRC_EMR2_0 (LSB)	CH	CR
002Ah	SDRC_EMR3_0 (LSB)		CR

Table 26-42. CHRAM (continued)

Offset	Field	Description CH – used in CH process CR – used in context restore
002Ch	SDRC_ACTIM_CTRLA_0 (LSB)	CH CR
002Eh	SDRC_ACTIM_CTRLA_0 (MSB)	CH CR
0030h	SDRC_ACTIM_CTRLB_0 (LSB)	CH CR
0032h	SDRC_ACTIM_CTRLB_0 (MSB)	CH CR
0034h	SDRC_RFRCTRL_0 (LSB)	CH CR
0036h	SDRC_RFRCTRL_0 (MSB)	CH CR
0038h	Memory type (LSB)	CH CR 0 = SDRAM 1 = Low power SDR 2 = DDR 3 = Mobile DDR 4 = Unknown
003Ah	Must be 0	CH CR
003Ch	SDRC_MCFG_1 (LSB)	CH CR
003Eh	SDRC_MCFG_1 (MSB)	CH CR
0040h	SDRC_MR_1 (LSB)	CH CR
0042h	SDRC_EMR1_1 (LSB)	CH CR
0044h	SDRC_EMR2_1 (LSB)	CH CR
0046h	SDRC_EMR3_1 (LSB)	CR
0048h	SDRC_ACTIM_CTRLA_1 (LSB)	CH CR
004Ah	SDRC_ACTIM_CTRLA_1 (MSB)	CH CR
004Ch	SDRC_ACTIM_CTRLB_1 (LSB)	CH CR
004Eh	SDRC_ACTIM_CTRLB_1 (MSB)	CH CR
0050h	SDRC_RFRCTRL_1 (LSB)	CH CR
0052h	SDRC_RFRCTRL_1 (MSB)	CH CR
0054h	Reserved, write 0s for future compatibility	CR
0056h	Reserved, write 0s for future compatibility	CR
0058h	Flags	CH CR [0]: 0= CS0 not configured 1= CS0 configured [1]: 0= CS1 not configured 1= CS1 configured
005A	Must be 0	CH CR

26.4.8.2.3 CHFLASH

The CHFLASH configuration header contains settings specific to the general-purpose memory controller (GPMC). Refer to the *GPMC* section of Chapter 11, *Memory Subsystem*, for more information. The fields are described in Table 26-43. The ROM code configures the GPMC by default to these settings:

- Chip select 0
- Asynchronous mode
- Wait input enabled
- Base address 08000000h

Table 26-43. CHFLASH

Offset	Field	Description
0000h	Section Key	Key used for section verification: C0C0C0C3h.
0004h	Valid	Enables/disables the section: 00h – Disable Others – Enable
0005h	Reserved	
0008h	GPMC_SYSCONFIG (LSB)	Register value
000Ah	GPMC_IRQENABLE (LSB)	
000Ch	GPMC_TIMEOUT_CONTROL (LSB)	
000Eh	GPMC_CONFIG (LSB)	
0010h	GPMC_CONFIG1_0	
0014h	GPMC_CONFIG2_0	
0018h	GPMC_CONFIG3_0	
001Ch	GPMC_CONFIG4_0	
0020h	GPMC_CONFIG5_0	
0024h	GPMC_CONFIG6_0	
0028h	GPMC_CONFIG7_0	
002Ch	GPMC_PREFETCH_CONFIG1	
0030h	GPMC_PREFETCH_CONFIG2 (LSB)	
0032h	GPMC_PREFETCH_CONTROL (LSB)	
0034h	GPMC_ECC_CONFIG	
0036h	GPMC_ECC_CONTROL	
0038h	GPMC_ECC_SIZE_CONFIG (LSB)	
003Ch	Enable_A1_A10	0 – Do not change A1 – A10 Others - Enable pins A1 – A10

26.4.8.2.4 CHMMCSD

The CHMMCSD configuration header contains settings specific to the high-speed MMC/SD/SDIO host controller (MMCHS). Refer to Chapter 22, *MMC/SD/SDIO Card Interface*, for more information. The fields are described in Table 26-44. The ROM code configures the MMCHS by default to these settings:

- 400-kHz clock during identification mode
- 19.2-MHz clock during data transfer mode

Table 26-44. CHMMCSD CH

Offset	Register Modified	Description
0000h	Section key	Key used for section verification C0C0C0C4h
0004h	Valid	Enables/disables the section 00h – Disable Other – Enable
0005h	Reserved	
0008h	MMCHS_SYSCTRL(MSB)	Register value
000Ah	MMCHS_SYSCTRL(LSB)	0xFFFFFFFF -> Do not update register.
000Ch	Bus width	1 -> 1 bit 2 -> 4 bits 4 -> 8 bits 0xFFFFFFFF -> Do not update register.

Note: The ROM code transmits a booting parameters structure to the initial software (see Section 26.4.8.4, *Image Execution*). This structure contains a field indicating whether the configuration header sections have been correctly taken into account. For the CHMMCSD section, if all the section fields are set to 0xFFFFFFFF, regardless of the value of the VALID bit field, the booting parameters shoud indicate that the CHMMCSD section has not been executed.

26.4.8.3 Image Format for GP Devices

For a GP device, the image is simple and must contain a small header having the size of the software to load and the destination address of where to store it when a device is other than XIP. The XIP device image is even simpler and starts with executable code. The image format is described in Table 26-45. The header is used only for memory booting. The peripheral booting image does not have any header and starts directly with executable code.

Table 26-45. GP Device Software Image

Field	Non-XIP Device Offset	XIP Device Offset	Size (Bytes)	Description
Size	0x0000	–	4	Size of the image
Destination	0x0004	–	4	Address where to store the image
Image	0x0008	0x0000	x	Image

26.4.8.4 Image Execution

The image is executed when the ROM code performs a branch instruction to the first executable instruction in the initial software. The execution address is the first word after the image header. After the branch, the ARM runs in public ARM supervisor mode. The R0 register points to the booting parameter structure that contains information about booting. Table 26-46 shows the booting parameters structure.

Table 26-46. Booting Parameter Structure

Offset	Field	Size (Bytes)	Description
0x00	Booting message	4	Last received booting message
0x04	Current booting device	1	Code of device used for booting: 0x01: XIP memory 0x02: NAND 0x03: OneNAND 0x04: DOC 0x05: MMC/SD2 0x06: MMC/SD1 0x07: XIP memory with wait monitoring 0x10: UART 0x11: HS USB Other: Reserved
0x05	Reserved	1	Reserved

Table 26-46. Booting Parameter Structure (continued)

Offset	Field	Size (Bytes)	Description
0x06	Reset reason	1	Current reset reason bit mask (bit = 1, event present): [0]: POR [1]: Global software reset [3]: Violation reset [4]: MPU watchdog reset [5]: Watchdog reset [6]: External warm reset Other bits: Reserved
0x07	CH flags	1	Configuration header sections flag. Each section is described by 1 bit. A set bit indicates that the section was executed: [0]: CHSETTINGS [1]: CHRAM [2]: CHFLASH [3]: CHMMC Other bits: Reserved
0x08	Device descriptor	4	Pointer to the device descriptor structure. This pointer is required when current booting device driver functions are called.

26.4.9 Tracing

Tracing in the public ROM code consists of a 64-bit vector in which each bit corresponds to a certain point of the ROM code execution flow. The tracing vector is divided into two 32-bit words. Table 26-47 lists the location and organization of the tracing data in RAM. Tracing data is initialized at the beginning of startup.

There are two sets of tracing data: current trace information and tracing collected during the first ROM code run after a cold reset. This cold reset tracing is copied to its location during the tracing initialization of the second ROM code run after the cold reset run. These data can be used for debugging.

Table 26-47. Tracing Vector

Bit No.	Group	Meaning
0	General	Reset
1	General	ROM code C main
2	General	ROM code runs after the cold reset
3	Boot	Booting started
4	Memory boot	Memory booting started
5	Boot	No more device to check
6	Peripheral boot	Peripheral booting started
7	Boot	Booting message change device
8	Boot	Booting message skip per. booting
9	Reserved	Reserved
10	Memory boot	CH found
11	Memory boot	Image header correct
12	Peripheral boot	Device initialized
13	Peripheral boot	ASIC ID sent
14	Peripheral boot	Booting message received
15	Peripheral boot	Image received
16	Peripheral boot	Peripheral booting failed

Table 26-47. Tracing Vector (continued)

Bit No.	Group	Meaning
17	Peripheral boot	UART
18	Peripheral boot	USB
19	Reserved	Reserved
20	Reserved	Reserved
21	Peripheral boot	NULL device
22	Execute	Image executed
23	Reserved	Reserved for non-GP devices
24	Reserved	Reserved for non-GP devices
25	Reserved	Reserved for non-GP devices
26	Reserved	Reserved for non-GP devices
27	Reserved	Reserved for non-GP devices
28	Reserved	Reserved for non-GP devices
29	Boot	Software booting configuration section 1 found
30	General	Software booting configuration clocking section found
31	Reserved	Reserved
32	Memory boot	Null device
33	Memory boot	XIP
34	Memory boot	NAND
35	Memory boot	OneNAND
36	Memory boot	DOC
37	Memory boot	MMC/SD2
38	Memory boot	MMC/SD1
39	Memory boot	XIP memory with wait monitoring
40	Reserved	Reserved
41	Reserved	Reserved
42	Reserved	Reserved
43	Reserved	Reserved
44	Reserved	Reserved
45	Reserved	Reserved
46	Reserved	Reserved
47	Memory boot	No known NAND was detected.
48	Memory boot	MLC NAND image was detected.
49	Memory boot	NAND boot was attempted from block 0.
50	Memory boot	NAND boot was attempted from block 1.
51	Memory boot	NAND boot was attempted from block 2.
52	Memory boot	NAND boot was attempted from block 3.

26.5 Wake-Up Booting by ROM Code

In CORE OFF power state, all configurations outside the WKUP power domain are reset and must be restored to restart the system in the condition it was in before being stopped. After an MPU and CORE power domain wake-up reset, the ROM code restores the SDRC and clock settings to allow a jump to a public restore function. This public restore function is specifically implemented by users to restore their own hardware and software environments. Users must set the public restore function address in the public restore pointer field of the CONTROL_SAVE_RESTORE_MEM structure.

The CONTROL_SAVE_RESTORE_MEM memory of the SCM saves and restores the mandatory information to automatically reconfigure the SDRC and PRCM registers. The SDRC and clock registers must be saved by users in the respective structures (see Table 26-49 and Table 26-50) in the SCM (scratchpad memory) before going to off mode to be automatically restored on wake-up boot by the ROM code.

The size of CONTROL_SAVE_RESTORE_MEM is 240 bytes starting at the hardware physical address 0x48002910 to 0x480029FF. Figure 26-29 shows the memory format that must be obeyed to handle the context restoration.

Figure 26-29. CONTROL_SAVE_RESTORE_MEM Format

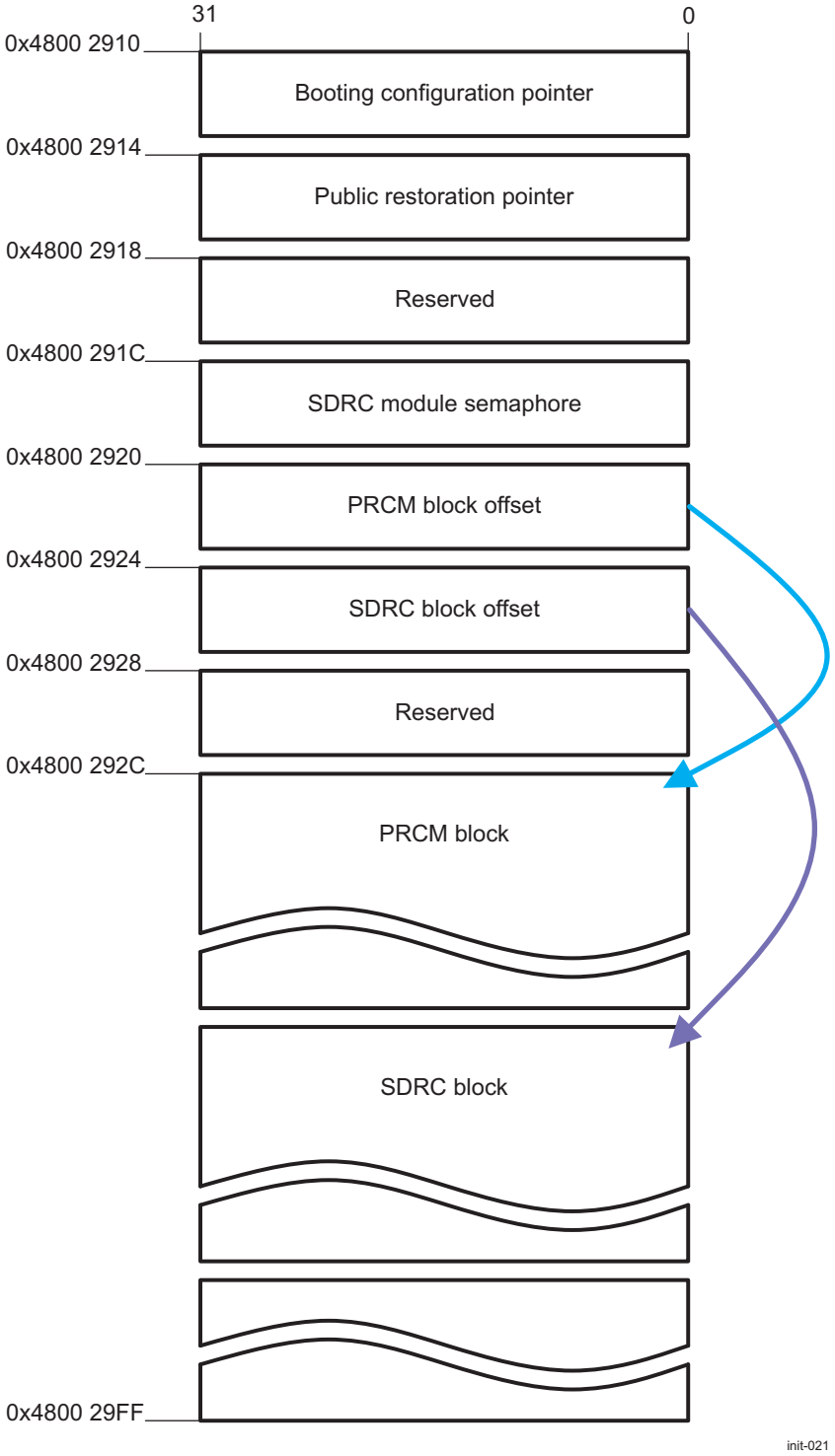


Table 26-48. CONTROL_SAVE_RESTORE_MEM Field Definitions

Field Name	Size	Description
Booting Configuration Pointer	32 bits	Address of public booting information used after a software reset

Table 26-48. CONTROL_SAVE_RESTORE_MEM Field Definitions (continued)

Field Name	Size	Description
Public Restore Pointer(1)	32 bits	Address of the public restore function to branch to when exiting a wake-up reset boot
SDRC Module Semaphore(2)	32 bits	Semaphore used for accessing the SDRC module
PRCM Block Offset(1)	32 bits	Offset of the PRCM configuration in the control save restore memory. Set 0x0 to not use this block.
SDRC Block Offset(1)	32 bits	Offset of the SDRC configuration in the control save restore memory. Set 0x0 to not use this block.

(1) Field to be set by users
(2) ROM code use only

Table 26-49 describes the PRCM context restore block.

Table 26-49. PRCM Register Organization in the SCM Block

Offset	PRCM Register Byte Organization in the SCM Block			
	31:24	23:16	15:8	7:0
0x0000	PRM_CLKSRC_CTRL			
0x0004	PRM_CLKSEL			
0x0008	CM_CLKSEL_CORE			
0x000C	CM_CLKSEL_WKUP			
0x0010	CM_CLKEN_PLL			
0x0014	CM_AUTOIDLE_PLL			
0x0018	CM_CLKSEL1_PLL			
0x001C	CM_CLKSEL2_PLL			
0x0020	CM_CLKSEL3_PLL			
0x0024	CM_CLKEN_PLL_MPU			
0x0028	CM_AUTOIDLE_PLL_MPU			
0x002C	CM_CLKSEL1_PLL_MPU			
0x0030	CM_CLKSEL2_PLL_MPU			
0x0034	PRCM Block Size			

Table 26-50 describes the SDRC context restore block.

Table 26-50. SDRC Register Organization in the SCM Block

Offset	SDRC Register Byte Organization in the SCM Block			
	31:24	23:16	15:8	7:0
0x0000	CS_CFG		SYSCONFIG	
0x0004	ERR_TYPE		SHARING	
0x0008	DLLA_CTRL			
0x000C	Reserved			
0x0010	POWER			
0x0014	Reserved			
0x0018	MCFG_0			
0x001C	Reserved		MR_0	
0x0020	Reserved		EMR2_0	
0x0024	ACTIM_CTRLA_0			
0x0028	ACTIM_CTRLB_0			
0x002C	RFR_CTRL_0			
0x0030	Reserved			
0x0034	MCFG_1			

Table 26-50. SDRC Register Organization in the SCM Block (continued)

Offset	SDRC Register Byte Organization in the SCM Block			
	31:24	23:16	15:8	7:0
0x0038	Reserved		MR_1	
0x003C	Reserved		EMR2_1	
0x0040	ACTIM_CTRLA_1			
0x0044	ACTIM_CTRLB_1			
0x0048	RFR_CTRL_1			
0x004C	Reserved		Reserved	
0x0050	Reserved			
0x0054	SDRC block size			

26.6 Debug Configuration

26.6.1 Overview

This section provides information for using a debugger on the public ARM Cortex™-A8 in the OMAP34xx device. The debug capability is accessible through the JTAG interface with a limited number of pins.

26.6.2 JTAG Port Signal Description

The OMAP3430 target debug interface uses the five standard IEEE 1149.1 (JTAG) signals (nTRST, TCK, TMS, TDI, and TDO), a return clock (RTCK) to meet the clocking requirements of the ARM968 processor, and the two instrumentations pins (EMU0, EMU1). For more information, see Table 26-51.

Table 26-51. Debug POR Signals

Pin	Type[(1)]	Name	Description
nTRST	I	Test logic reset	When asserted (active low), causes all test and debug logic in the device to be reset with the IEEE 1149.1 interface
TCK	I	Test clock	This is the test clock used to drive an IEEE 1149.1 test access port (TAP) state-machine and logic. Depending on the emulator attached to OMAP34xx, this is a free-running clock or a gated clock, depending on RTCK monitoring.
RTCK	O	Returned test clock	Synchronized TCK. Depending on the emulator attached to OMAP3430, the JTAG signals are clocked from RTCK or RTCK is monitored by the emulator to gate TCK.
TMS	I	Test mode select	Directs the next state of the IEEE 1149.1 TAP state-machine
TDI	I	Test data input	Scan data input to the device.
TDO	O	Test data output	Scan data output of the device.
EMU0	I/O	Emulation 0	Channel 0 trigger: Boot mode
EMU1	I/O	Emulation 1	Channel 1 trigger: Boot mode

(1) I = Input, O = Output

The power domain containing the debug logic can be switched OFF in normal operating mode to reduce power consumption.

CAUTION

Before starting the debugger, the DEBUG power domain must be activated by applying a minimum of 10 TCK pulses to the OMAP3430 device after nTRST is pulled high. To prevent the JTAG_TMS pin from floating when the EMU power domain is off, add an external pullup (10 kΩ) on this pin.

26.6.3 Initial Scan Chain Configuration

The general-purpose ports that can provide access to many test support functions built into the device (including the test logic defined by the IEEE 1149.1 standard) are the TAP.

The first level of the debug interface that sees the scan controller is the TAP router module.

The debugger can configure the TAP router to serially link to 16 TAP controllers or to individually scan one TAP controller without disrupting the instruction register (IR) state of the other TAPs. The initial scan chain configuration of the OMAP34xx device is determined from the level of the EMU0 and EMU1 pins on the release of POR. At POR, EMU0 and EMU1 are automatically configured as inputs. The EMU0 and EMU1 pins should be pulled high at POR to configure the initial scan chain of the OMAP34xx device to TAP router-only mode. In the TAP router-only configuration, no secondary TAPs are selected. The TAP router is the only TAP between the device-level TDI and TDO.

The router TAP has an IR length of 6 bits. This is the recommended boot mode. The third-party debugger must assume that the TAP router is the only TAP between TDI and TDO at boot.

26.6.4 Adding TAPs to the Scan Chain

The TAP router must be programmed to add TAPs to the scan chain (used in Design For Test). The following JTAG scans must be completed before the CoreSight debug access port (DAP) is added to the scan chain. CoreSight is the on-chip debug and trace ARM technology. IR and DR are the instruction register and the data register, respectively.

```
## Function  : Update the JTAG preamble and post-amble counts.
Parameter : The IR pre-amble count is '0'.
Parameter : The IR post-amble count is '0'.
Parameter : The DR pre-amble count is '0'.
Parameter : The DR post-amble count is '0'.
Parameter : The IR main count is '6'.
Parameter : The DR main count is '1'.

## Function  : Do a send-only JTAG IR/DR scan.
Parameter : The route to JTAG shift state is 'shortest transition'.
Parameter : The JTAG shift state is 'shift-ir'.
Parameter : The JTAG destination state is 'pause-ir'.
Parameter : The bit length of the command is '6'.
Parameter : The send data value is '0x00000007'.
Parameter : The actual receive data is 'discarded'.

## Function  : Do a send-only JTAG IR/DR scan.
Parameter : The route to JTAG shift state is 'shortest transition'.
Parameter : The JTAG shift state is 'shift-dr'.
Parameter : The JTAG destination state is 'pause-dr'.
Parameter : The bit length of the command is '8'.
Parameter : The send data value is '0x00000089'.
Parameter : The actual receive data is 'discarded'.

## Function  : Do a send-only JTAG IR/DR scan.
Parameter : The route to JTAG shift state is 'shortest transition'.
Parameter : The JTAG shift state is 'shift-ir'.
Parameter : The JTAG destination state is 'pause-ir'.
Parameter : The bit length of the command is '6'.
Parameter : The send data value is '0x00000002'.
Parameter : The actual receive data is 'discarded'.

## Function  : Embed the POR address in next command.
Parameter : The POR address field is '0x0f000000'.
Parameter : The POR address value is '3'.

## Function  : Do a send-only JTAG IR/DR scan.
Parameter : The route to JTAG shift state is 'shortest transition'.
Parameter : The JTAG shift state is 'shift-dr'.
Parameter : The JTAG destination state is 'pause-dr'.
Parameter : The bit length of the command is '32'.
Parameter : The send data value is '0xa3002108'.
Parameter : The actual receive data is 'discarded'.
```

```
## Function  : Do a send-only all-ones JTAG IR/DR scan.
Parameter : The JTAG shift state is 'shift-ir'.
Parameter : The JTAG destination state is 'run-test/idle'.
Parameter : The bit length of the command is '6'.
Parameter : The send data value is 'all-ones'.
Parameter : The actual receive data is 'discarded'.

## Function  : Wait for a minimum number of TCLK pulses.
Parameter : The count of TCLK pulses is '10'.

## Function  : Update the JTAG preamble and post-amble counts.
Parameter : The IR pre-amble count is '0'.
Parameter : The IR post-amble count is '6'.
Parameter : The DR pre-amble count is '0'.
Parameter : The DR post-amble count is '1'.
Parameter : The IR main count is '4'.
Parameter : The DR main count is '1'.
```

After the DAP is added to the scan chain, the debugger can communicate with the Cortex-A8 processor.

26.6.5 Debugger Address Space

The CoreSight components are interfaced with the TAP router through the DAP. As recommended by the CoreSight architecture, the DAP is directly interfaced to the OMAP34xx bus. The debugger can directly access the entire memory space without requiring the processor to enter debug state and be programmed with a load or store instruction. Table 26-52 lists the modules that are mapped to the DAP address, and the address space detail.

Table 26-52. Debugger Address Space

Start Addr (Hex)	End Addr (Hex)	Size	Description
0x5401 0000	0x5401 0FFF	4KB	ARM embedded trace macrocell (ETM) module
0x5401 1000	0x5401 1FFF	4KB	Cortex-A8 module
0x5401 9000	0x5401 9FFF	4KB	Trace port interface unit (TPIU)
0x5401 B000	0x5401 BFFF	4KB	ARM embedded trace buffer (ETB) module

The DBGEM signal on the Cortex-A8 is driven by setting bit 13 at address 0x5401 D030 in the DAP-APB address space.

Glossary

Numerical

2D— Two dimensional

3D— Three dimensional

3GPP— 3rd Generation Partnership Project

A

ABS— Absolute value

ADC— Analog-to-digital converter/conversion

AEW— Adaptive multirate

AHB— Advanced high-performance bus

AMR— Adaptive multirate; also Audio modem riser: An Intel specification that defines a new architecture for the design of motherboards.

APB— Advanced peripheral bus

APE— Applicative processor engine

API— Application programming interface; also ARM port interface

ARAU— Auxiliary register arithmetic unit

ARGB— Alpha, red, green, blue

ASIC— Application-specific integrated circuit: A chip built for a specific application. In the context of this document, ASIC refers to the FPGA that resides on the EVM board.

ASCII— American standard code for information

ATR— Answer to reset

AVC— Advanced video coding (MPEG4 - Part 10 also known as H264)

AXI— Advanced extensible interface

B

BB— Busy bus

BCD— Binary-coded decimal: A representation of decimal digits (0–9) using a nibble that uses a certain number of bits. For example, using 4 bits, two BCDs can be packed into 1 byte.

BGA— Ball grid array

BGAPTS— Band gap voltage and temperature sensor

BGT— Block guard time

BIOS— Built-in operating system

BIST— Built-in self-test

Bluetooth®— A short-range radio technology designed to simplify communications among network devices and between devices and the Internet. It also simplifies data synchronization between network devices and other computers.

BPP— Bits per pixel

BTA— Bus turn around

C

CALU— Central arithmetic logic unit

CAVLC— Context-adaptive variable length coder

CAVLD— Context-adaptive variable length decoder

CBUFF— Circular buffer

CDL— Controlled delay line

CE— Chip enable

CH— Configuration header. To use settings other than ROM code defaults, (that is, clock frequencies, SDRAM/DDRAM settings, or GPMC settings).

CID— Card identification number

CIF— Common intermediate format. A video format used in video conferencing systems that easily supports both NTSC and PAL signals. CIF is part of the ITU H.261 video conferencing standard. It specifies a data rate of 30 fps, with each frame containing 288 lines and 352 pixels per line.

CLE— Command latch enable

CLK— Clock

CLUT— Color look-up table

CMOS— Complementary metal oxide semiconductor

CMT— Cellular mobile telephone

Codec— Coder/decoder or compression/decompression: A device that codes in one direction of transmission and decodes in another direction of transmission.

ConnID— Connection identifier: An initiator module identifier. A ConnID transmitted in-band with the request and is used for protection and error logging mechanism.

CP15— Coprocessor 15: This coprocessor controls the operation and configuration of the TI925T.

CPSR— Current program status register

CPU— Central processing unit: The CPU is the portion of the processor involved in arithmetic, shifting, and Boolean logic operations, as well as the generation of data and program memory addresses. The CPU includes the CALU, the multiplier, and the ARAU.

CRC— Cyclic redundancy check

CS— Chip-select

CTRL— Control

CVBS— Composite video broadcast signal

CWT— Command wire tracer

D

D2D— Die-to-die

DAC— Digital-to-analog converter

DBB— Digital baseband

DBI— Display date interface

DCS— Display command set

DCO— Digitally controlled oscillator

DCT— Discrete cosine transform: A fast Fourier transform used to manipulate compressed still and moving picture data.

DDR— Dual data rate

DE— Date enable

DFF— Digital flip-flop

DFT— Design for test

DI— Data identifier

DISPC— Display controller

DLL— Delay-locked loop

DMA— Direct memory access: A mechanism whereby a device other than the host processor contends for and receives mastery of the memory bus so that data transfers can occur independent of the host.

DMC— Data memory controller

DPD, DPDM— Deep power-down mode

DPF— Dynamic power framework

DPI— Display parallel interface. Digital implementation of PLL

DPLL— Digital phase-locked loop

DRD— Dual role device

DRDY— Data ready

DRM— Digital rights management

DSI— Display serial interface

DSP— Digital signal processor: A semiconductor that manipulates discrete or discontinuous electrical impulses in a manner that implements a desired algorithm.

DSS— Display subsystem

DT— Data type

DVFS— Dynamic voltage and frequency scaling

E

EAV— End of active video

ECC— Error checking and correction; also error correction code.

ED— Endpoint descriptor

EDC— Error detection code

eFuse— Electrical fuse: A one-time programmable memory location usually set at the factory

EHCI— Enhanced host controller interface

EMC— External memory controller

EMI— Electromagnetic interference

EMV— Initial letters of Europay, MasterCard, and VISA, the three companies which originally cooperated to develop the EMV standard

EOF— End of frame

EOT— End of transfer

ES— Erase status

ETB— Embedded trace buffer

ETSI— European Telecommunications Standard Institute

ETM— Embedded trace macrocell

F

FC— Frame counter

FE— Framing error: An error that occurs when the asynchronous serial port receives a data character that does not have a valid stop-bit.

FIFO— First in first out: A queue; a data structure or hardware buffer from which items are removed in the same order they were put in. A FIFO is useful for buffering a stream of data between a sender and receiver which are not synchronized; that is, the sender and receiver are not sending and receiving at exactly the same rate. If the rates differ by too much in one direction for too long, the FIFO becomes either full (blocking the sender) or empty (blocking the receiver).

FIQ— Fast interrupt request

FlatLink™ 3G—A Texas Instruments display interface protocol

FS— Frame synchronization

FSM— Finite state-machine: A model of computation consisting of a set of states, a start state, an input alphabet, and a transition function that maps input symbols and current states to a next state.

FSR— Fault status register

FT— Feed through

G

GDD— Generic distributed DMA

GP device—General-purpose device

GPIO— General-purpose input/output: Pins that can accept input signals and/or send output signals but are not linked to specific uses.

GPMC— General-purpose memory controller

GSM— Global system for mobile communications

H

HC— Host controller

HDTV— High-definition television

HFP— Horizontal Front Porch

HNP— Host Negotiation Protocol

HPI— Host port interface

HS— High speed

HSEC— Horizontal sync end code

HSSC— Horizontal sync start code

HSYNC— Horizontal synchronization: A bidirectional horizontal timing signal occurring once per line with a pulse width defined as an integral number of FCLK periods. Synchronization signals can be used to enable retrace of the electron beam of a display screen. Also HS.

HSW— Horizontal Synchronization Pulse Width

HVGA— Half-size video graphics array. One-half the resolution of VGA

HW, H/W—Hardware

HWA— Hardware accelerators

I

I^2C— Inter-integrated circuit: A multimaster bus where multiple chips can be connected. Each chip can act as a master by initiating a data transfer.

I2S— Inter-IC sound: A digital audio interface standard.

IA— Initiator agent, also Identifier address

IC— Integrated circuit

ICR— Intersystem communication registers

IF— Interface

INT— Interrupt: A signal sent by hardware or software to a processor requesting attention. An interrupt tells the processor to suspend its current operation, save the current task status, and perform a particular set of instructions. Interrupts communicate with the operating system and prioritize tasks to be performed.

INTC— Interrupt controller

I/O— Input/output

IP— Intellectual property

IPC— Interprocessor communication. Also referred to as *mailbox.*

IrDA— Infrared Data Association: Represents the group of device manufacturers that developed a standard for transmitting data via infrared light waves.

IRQ— Interrupt request: IRQs are hardware lines over which devices can send interrupt signals to the microprocessor.

ISO— Isochronous: This refers to processes where data must be delivered within certain time constraints. For example, multimedia streams require an isochronous transport mechanism to ensure that data is delivered as fast as it is displayed and to ensure that the audio is synchronized with the video. Also, *International Standards Organization.*

ISP— Image sensing product; also image signal processor

ISR— Interrupt service routine: A function or set of functions that are called when an interrupt is encountered.

ITU— International Telecommunications Union

IVA— Image and video accelerator

J

JPEG— Joint Photographics Experts Group

JTAG— Joint Test Action Group: The JTAG was formed in 1985 to develop economical test methodologies for systems designed around complex integrated circuits and assembled with surface-mount technologies. The group drafted a standard that was subsequently adopted by IEEE as IEEE Standard 1149.1-1990, IEEE Standard Test Access Port, and Boundary-Scan Architecture.

K

Kb— Kilobits

KB— Kilobyte, 1024 B

Kbps— Kilobits per second

L

L1— Level 1 cache/memory

L2— Level 2 cache/memory

L3— First level of interconnect in OMAP platform

L4— Second level of interconnect in OMAP platform

LCD— Liquid crystal display: A display that uses two sheets of polarizing material with a liquid crystal solution between them.

LCh— Logical DMA channel

LDM— Load multiple

LDO— Low dropout

LH— Local host

LINK— Link layer device

LLP— Low-level protocol

LP— Low power, operation mode for PHY

LPDDR— Low-power, double-data rate (SDRAM)

LPM— Low-power mode

LPP— Lines per panel

LPSDR— Low-power, single-data rate (SDRAM)

LRC— Longitudinal redundancy check

LRU— Least recently used

LS— Level shifter; also low speed

LSB— Least-significant bit

LUT— Look-up table

LVDS— Low-voltage differential signaling

M

Mailbox— See *IPC*

Mb— Megabit

Mbps— Megabits per second

McBSP— Multichannel buffered serial port: An enhanced buffered serial port that includes the following standard features: buffered data registers, full duplex communication, and independent clocking and framing for receive and transmit. In addition, the McBSP includes the following enhanced features: internal programmable clock and frame generation, multichannel mode, and general-purpose I/O.

MCSPI— Multichannel serial port interface

MCP— Multi-chip package

MCU— Microcontroller unit (refers to the MPU)

MDDR— Mobile double-data-rate SDRAM, dedicated to mobile applications

MIPI®— Mobile industry processor interface

MMC— Multimedia card

MMC/SD— Multimedia card/secure data

MMU— Memory management unit: The MMU performs virtual-to-physical address translations, performs access permission checks for access to the system memory, and provides the flexibility and protection required for the OS to manage a shared physical memory space between the two processors.

MPEG— Motion Pictures Expert Group: A compression scheme for full-motion video.

MPEG1— The first MPEG compression scheme specification

MPEG4— The most current MPEG compression scheme specification, intended for very narrow bandwidths.

MPU— Microprocessor unit

MS— Mobile station

MSB— Most-significant bit: The highest order bit in a word. The plural form (MSBs) refers to a specified number of high-order bits, beginning with the highest order bit and counting to the right. For example, the 8 MSBs of a 16-bit value are bits 15 through 8.

MUX— Multiplex/multiplexer

Muxed pin— A pin is muxed when its pin control register field can be reconfigured by software to change the function associated with the pin.

MuxMode— A 3-bit field of the pin control register field which enables to change the mode. Mode programming is assumed by software and selects a function on the device external interface.

N

N/A— Not applicable

NAK— Not acknowledged

NAND— NAND flash memory that is a high-capacity, low-cost, embedded, permanent data storage solution

NF— Noise filter

NMI— Nonmaskable interrupt: An interrupt that cannot be masked or disabled.

NOP— No operation (DSP/CPU instruction)

NR— Noise reduction

NSC— National Semiconductor Corporation

NTSC— National Television System Committee (television broadcast system)

O

OCM— On-chip memory

OCO— One change only

OCP— Open-core protocol

OCPI— Open-core protocol interface

OEM— Original equipment manufacturer

OGL— Open GL (programming API) enable

OHCI— Open host controller interface: This is an industry standard USB host controller interface.

OMAP— An open software and hardware platform targeted at second- and third-generation cellular phones with multimedia capabilities.

OneNand™— A memory chip based on NAND architecture integrating SRAM buffers and NOR logic interface. It combines the advanced data storage function of NAND flash with fast read speed function of NOR flash.

OTG— On-the-go (USB 2.0 specification)

P

PBIAS— PMOS bias transistor to provide the bias voltage to extended drain I/Os

PCB— Printed circuit board

PCLK— Pixel clock

PCM— Pulse code modulation: A technique for digitizing speech by sampling the sound waves and converting each sample into a binary number.

PDA— Personal digital assistant

PDE— Personal down enable

PE— Packet end

PF— Packet footer

PH— Packet header

PG— Protection group

PGA— Pin grid array

PHY— Physical layer device

PI— Packet identifier

PID— Protocol identifier: The PID register is used in Windows CE mode only.

PLD— Programmable logic device

PLL— Phase-locked loop: A closed loop frequency control system whose function is based on the phase-sensitive detection of the phase difference between the input signal and the output signal of the controlled oscillator (CO).

PMC— Program memory controller

POP— Package-on-package technology

PMP— Power management port

POR— Power-on reset

PPI— Physical layer protocol interface

PPS— Protocol and parameter selection

PRCM— Power, reset, and clock management

PRM— Power and reset manager

PS— Packet start

PSA— Parallel signature analyzer

PT— Packet type

PVT— Process, voltage, and temperature (that is, PVT dispersion)

PWB— Printed wiring board

PWM— Pulse width modulation

PWR— Power

Q

QCIF— Quarter common intermediate format: A video conferencing format that specifies data rates of 30 fps, with each frame containing 144 lines and 176 pixels per line. This is one-fourth the resolution of CIF. QCIF support is required by the ITU H.261 video conferencing standard.

QVGA— Quarter video graphics array (one-fourth the resolution of VGA).

R

R— Read only

RBL— Read buffer logic

RFBI— Remote frame buffer interface

RFU— Reserved for future use

RGB— Red, green, blue

RGBA— Red, green, blue, alpha

ROM— Read only memory: A semiconductor storage element containing permanent data that cannot be changed.

RST— Reset

RT— Real-time

RVLC— Reversible variable length coder

RVLD— Reversible variable length decoder

R/W— Read/write

RX— Receive/receiver

S

S3220— Abbreviation of Sonics3220. It refers to the L4 interconnect.

SAM— Shared access mode: The mode that allows both the DSP and the host to access host port interface (HPI) memory. In this mode, asynchronous host accesses are synchronized internally, and, in case of conflict, the host has access priority and the DSP waits one cycle.

SAR— Save and restore: Hardware context saving for power saving

SAV— Start of active video

SCCB— Serial camera control interface: 3-wire and 2-wire serial bus defined and deployed by Omnivision Technologies, Inc.

SCL— Serial clock: Programmable serial clock used in the I^2C interface. Also SCLK.

SCM— Scan combiner module; also statistic collection module

SCP— Serial configuration port

SD— Starting delimiter

SDA— Serial data: Serial data bus in the I^2C interface.

SDI— Serial display interface

SDIO— Secure digital input/output

sDMA— System direct memory access

SDR— Single data rate

SDRC— SDRAM controller

SDRAM— Synchronous dynamic random access memory

SDTI— System debug trace interface

SDTV— Standard digital television

SGX— Abbreviation for graphic accelerator (GFX in ES1.0)

SIM— Subscriber identity module

SIMD— Single instruction multiple data

SILVS— Scalable low-voltage signaling

SMS— SDRAM memory scheduler

SMX— Abbreviation of SonicsMX. It refers to the L3 interconnect.

SNR— Signal-to-noise ratio

SOF— Start of frame

SOT— Start of transmission

SRAM— Static random access memory. Fast memory that does not require refreshing, as DRAM does. It is more expensive than DRAM, though, and is not available in as high a density as DRAM.

SRG— Sample rate generator

SRP— Session request protocol

SRRP— Session RAM restoration pointer

SS— Subsystem

SSR— Serial synchronous receiver

SST— Serial synchronous transmitter

ST— Start timer

STM— Synchronous transfer mode; also store multiple

STN— Super-Twist Nematic: A technique for improving LCD display screens by twisting light rays.

SVGA— Super video graphic adapter

SW— Software

SWI— Software interrupt

T

TC— Traffic controller. Allows asynchronous operation among the external memory interface, the MPU, and the DSP.

TCK— Test clock

TD— Transfer descriptor

TDM— Time division multiplex/multiplexing: The process by which a single serial bus is shared by multiple devices with each device taking turns to communicate on the bus. The total number of time slots (channels) depends on the number of devices connected. During a time slot, a given device may talk to any combination of devices on the bus.

TFT— Thin film transistor. A type of LCD flat panel display screen in which each pixel is controlled by one to four transistors.

TLB— Translation lookaside buffer: A cache that contains entries for virtual-to-physical address translation and access permission checking.

TLL— Transceiverless link: This is logic that lets the user connect two USB transceiver interfaces together directly without the use of differential transceivers.

TM— Target module: A target module cannot generate read/write requests to the chip interconnects, but it responds to these requests. However, it may generate interrupts or a DMA request to the system (typically: peripherals, memory controllers).

TOC— Table of contents

TRM— Technical reference manual

TRX— USB transceiver: The USB analog driver/receiver.

TTB— Translation table base: It points to the base of a table in physical memory that contains section and page table descriptors.

TWL— Table walking logic

TX— Transmit/transmitter

U

UART— Universal asynchronous receiver/transmitter: Another name for the asynchronous serial port.

UICC— Universal integrated circuit card

ULPM— Ultra-low power mode

ULPI— UTMI+ low pincount interface

ULPS— Ultra-low power state

UMC— United memory controller

UNP— Unpredictable

USAR— Universal synchronous/asynchronous receiver

USART— Universal synchronous/asynchronous receiver/transmitter

USB— Universal serial bus: An external bus standard that supports data transfer rates of 12M bps (12 million bits per second). A single USB port can be used to connect up to 127 peripheral devices.

UTMI— USB 2.0 transceiver macrocell interface

UTMI+— UTMI extension supporting USB host and on-the-go

V

VA— Volt-amps: A form of power management. A VA rating is the volts rating multiplied by the amps (current) rating, used to indicate the output capacity of an uninterruptible power supply (UPS) or other power source.

VC— Virtual channel

VENC— Video encoder

VESA— Video Electronics Standards Association

VFP— Vertical front porch

VGA— Video graphics array: An industry standard for video cards.

VLC— Variable length decoder

VLD— Variable length coder

VMODE— Bi-level voltage control interface

VPBE— Video processing back end

VSEC— Vertical synchronization end code

VSSC— Vertical synchronization start code

VSYNC, VS— Vertical synchronization: A bidirectional vertical timing signal occurring once per frame with a pulse-width defined as an integral number of lines (half-lines for interlaced mode).

W

WBL— Write buffer logic

WC— Word count

WD— Watchdog: A timer that requires the user program or OS periodically write to the count register before the counter underflows.

WDT— Watchdog timer

WLAN— Wireless local area network

WMV— Windows media video

WMA— Windows media audio

Word16— 16-bit word

Word32— 32-bit word

WP— Write protect

WSS— Wide-screen signaling

WWT— Work waiting time

X

XGA, XVGA—Extended graphics array

XIP— Execution in place

Y

YUV— Luminance-Bandwidth-Chrominance

IMPORTANT NOTICE

Texas Instruments Incorporated and its subsidiaries (TI) reserve the right to make corrections, modifications, enhancements, improvements, and other changes to its products and services at any time and to discontinue any product or service without notice. Customers should obtain the latest relevant information before placing orders and should verify that such information is current and complete. All products are sold subject to TI's terms and conditions of sale supplied at the time of order acknowledgment.

TI warrants performance of its hardware products to the specifications applicable at the time of sale in accordance with TI's standard warranty. Testing and other quality control techniques are used to the extent TI deems necessary to support this warranty. Except where mandated by government requirements, testing of all parameters of each product is not necessarily performed.

TI assumes no liability for applications assistance or customer product design. Customers are responsible for their products and applications using TI components. To minimize the risks associated with customer products and applications, customers should provide adequate design and operating safeguards.

TI does not warrant or represent that any license, either express or implied, is granted under any TI patent right, copyright, mask work right, or other TI intellectual property right relating to any combination, machine, or process in which TI products or services are used. Information published by TI regarding third-party products or services does not constitute a license from TI to use such products or services or a warranty or endorsement thereof. Use of such information may require a license from a third party under the patents or other intellectual property of the third party, or a license from TI under the patents or other intellectual property of TI.

Reproduction of TI information in TI data books or data sheets is permissible only if reproduction is without alteration and is accompanied by all associated warranties, conditions, limitations, and notices. Reproduction of this information with alteration is an unfair and deceptive business practice. TI is not responsible or liable for such altered documentation. Information of third parties may be subject to additional restrictions.

Resale of TI products or services with statements different from or beyond the parameters stated by TI for that product or service voids all express and any implied warranties for the associated TI product or service and is an unfair and deceptive business practice. TI is not responsible or liable for any such statements.

TI products are not authorized for use in safety-critical applications (such as life support) where a failure of the TI product would reasonably be expected to cause severe personal injury or death, unless officers of the parties have executed an agreement specifically governing such use. Buyers represent that they have all necessary expertise in the safety and regulatory ramifications of their applications, and acknowledge and agree that they are solely responsible for all legal, regulatory and safety-related requirements concerning their products and any use of TI products in such safety-critical applications, notwithstanding any applications-related information or support that may be provided by TI. Further, Buyers must fully indemnify TI and its representatives against any damages arising out of the use of TI products in such safety-critical applications.

TI products are neither designed nor intended for use in military/aerospace applications or environments unless the TI products are specifically designated by TI as military-grade or "enhanced plastic." Only products designated by TI as military-grade meet military specifications. Buyers acknowledge and agree that any such use of TI products which TI has not designated as military-grade is solely at the Buyer's risk, and that they are solely responsible for compliance with all legal and regulatory requirements in connection with such use.

TI products are neither designed nor intended for use in automotive applications or environments unless the specific TI products are designated by TI as compliant with ISO/TS 16949 requirements. Buyers acknowledge and agree that, if they use any non-designated products in automotive applications, TI will not be responsible for any failure to meet such requirements.

Following are URLs where you can obtain information on other Texas Instruments products and application solutions:

Products		**Applications**	
Amplifiers	amplifier.ti.com	Audio	www.ti.com/audio
Data Converters	dataconverter.ti.com	Automotive	www.ti.com/automotive
DLP® Products	www.dlp.com	Communications and Telecom	www.ti.com/communications
DSP	dsp.ti.com	Computers and Peripherals	www.ti.com/computers
Clocks and Timers	www.ti.com/clocks	Consumer Electronics	www.ti.com/consumer-apps
Interface	interface.ti.com	Energy	www.ti.com/energy
Logic	logic.ti.com	Industrial	www.ti.com/industrial
Power Mgmt	power.ti.com	Medical	www.ti.com/medical
Microcontrollers	microcontroller.ti.com	Security	www.ti.com/security
RFID	www.ti-rfid.com	Space, Avionics & Defense	www.ti.com/space-avionics-defense
RF/IF and ZigBee® Solutions	www.ti.com/lprf	Video and Imaging	www.ti.com/video
		Wireless	www.ti.com/wireless-apps

Mailing Address: Texas Instruments, Post Office Box 655303, Dallas, Texas 75265
Copyright © 2010, Texas Instruments Incorporated