TMS320DA804/DA803/DA802 DSP

Technical Reference Manual

Literature Number: SPRUH97B
March 2013–Revised September 2016

Contents

Preface **38**

1 Overview **39**

1.1 Introduction 40

2 DSP Subsystem **41**

2.1 Introduction 42

2.2 TMS320C674x Megamodule 43

2.2.1 Internal Memory Controllers 43

2.2.2 Internal Peripherals 43

2.3 Memory Map 47

2.3.1 DSP Internal Memory 47

2.3.2 External Memory 47

2.4 Advanced Event Triggering (AET) 47

3 System Interconnect **48**

3.1 Introduction 49

3.2 System Interconnect Block Diagram 50

4 System Memory **51**

4.1 Introduction 52

4.2 DSP Memories 52

4.3 Peripherals 52

5 Memory Protection Unit (MPU) **53**

5.1 Introduction 54

5.1.1 Purpose of the MPU 54

5.1.2 Features 54

5.1.3 Block Diagram 54

5.1.4 MPU Default Configuration 55

5.2 Architecture 55

5.2.1 Privilege Levels 55

5.2.2 Memory Protection Ranges 56

5.2.3 Permission Structures 56

5.2.4 Protection Check 57

5.2.5 DSP L1/L2 Cache Controller Accesses 57

5.2.6 MPU Register Protection 58

5.2.7 Invalid Accesses and Exceptions 58

5.2.8 Reset Considerations 58

5.2.9 Interrupt Support 58

5.2.10 Emulation Considerations 59

5.3 MPU Registers 59

5.3.1 Revision Identification Register (REVID) 60

5.3.2 Configuration Register (CONFIG) 60

5.3.3 Interrupt Raw Status/Set Register (IRAWSTAT) 61

5.3.4 Interrupt Enable Status/Clear Register (IENSTAT) 62

5.3.5 Interrupt Enable Set Register (IENSET) 63

5.3.6 Interrupt Enable Clear Register (IENCLR) 63

5.3.7 Programmable Range *n* Start Address Registers (PROG*n*_MPSAR) 64

Copyright © 2013–2016, Texas Instruments Incorporated

5.3.8 Programmable Range *n* End Address Registers (PROG*n*_MPEAR) 65
5.3.9 Programmable Range *n* Memory Protection Page Attributes Register (PROG*n*_MPPA) 66
5.3.10 Fault Address Register (FLTADDRR) 67
5.3.11 Fault Status Register (FLTSTAT) 68
5.3.12 Fault Clear Register (FLTCLR) 69
6 Device Clocking 70
6.1 Overview 71
6.2 Frequency Flexibility 72
6.3 I/O Domains 74
7 Phase-Locked Loop Controller (PLLC) 75
7.1 Introduction 76
7.2 PLL0 Control 76
7.2.1 Device Clock Generation 77
7.2.2 Steps for Changing PLL0 Domain Frequency 78
7.3 Locking/Unlocking PLL Register Access 79
7.4 PLLC Registers 80
7.4.1 Revision Identification Register (REVID) 81
7.4.2 Reset Type Status Register (RSTYPE) 81
7.4.3 PLL Control Register (PLLCTL) 82
7.4.4 PLL Multiplier Control Register (PLLM) 83
7.4.5 PLL Pre-Divider Control Register (PREDIV) 83
7.4.6 PLL Controller Divider 1 Register (PLLDIV1) 84
7.4.7 PLL Controller Divider 2 Register (PLLDIV2) 84
7.4.8 PLL Controller Divider 3 Register (PLLDIV3) 85
7.4.9 PLL Controller Divider 4 Register (PLLDIV4) 85
7.4.10 PLL Controller Divider 5 Register (PLLDIV5) 86
7.4.11 PLL Controller Divider 6 Register (PLLDIV6) 86
7.4.12 PLL Controller Divider 7 Register (PLLDIV7) 87
7.4.13 PLL Post-Divider Control Register (POSTDIV) 88
7.4.14 PLL Controller Command Register (PLLCMD) 88
7.4.15 PLL Controller Status Register (PLLSTAT) 89
7.4.16 PLL Controller Clock Align Control Register (ALNCTL) 90
7.4.17 PLLDIV Ratio Change Status Register (DCHANGE) 91
7.4.18 Clock Enable Control Register (CKEN) 92
7.4.19 Clock Status Register (CKSTAT) 93
7.4.20 SYSCLK Status Register (SYSTAT) 94
7.4.21 Emulation Performance Counter 0 Register (EMUCNT0) 95
7.4.22 Emulation Performance Counter 1 Register (EMUCNT1) 95
8 Power and Sleep Controller (PSC) 96
8.1 Introduction 97
8.2 Power Domain and Module Topology 97
8.2.1 Power Domain States 99
8.2.2 Module States 99
8.3 Executing State Transitions 101
8.3.1 Power Domain State Transitions 101
8.3.2 Module State Transitions 101
8.4 IcePick Emulation Support in the PSC 102
8.5 PSC Interrupts 102
8.5.1 Interrupt Events 102
8.5.2 Interrupt Registers 103
8.5.3 Interrupt Handling 104
8.6 PSC Registers 105
8.6.1 Revision Identification Register (REVID) 106

Copyright © 2013–2016, Texas Instruments Incorporated

8.6.2 Interrupt Evaluation Register (INTEVAL) 106
8.6.3 PSC0 Module Error Pending Register 0 (modules 0-15) (MERRPR0) 107
8.6.4 PSC1 Module Error Pending Register 0 (modules 0-31) (MERRPR0) 107
8.6.5 PSC0 Module Error Clear Register 0 (modules 0-15) (MERRCR0) 108
8.6.6 PSC1 Module Error Clear Register 0 (modules 0-31) (MERRCR0) 108
8.6.7 Power Error Pending Register (PERRPR) 109
8.6.8 Power Error Clear Register (PERRCR) 109
8.6.9 Power Domain Transition Command Register (PTCMD) 110
8.6.10 Power Domain Transition Status Register (PTSTAT) 111
8.6.11 Power Domain 0 Status Register (PDSTAT0) 112
8.6.12 Power Domain 1 Status Register (PDSTAT1) 113
8.6.13 Power Domain 0 Control Register (PDCTL0) 114
8.6.14 Power Domain 1 Control Register (PDCTL1) 115
8.6.15 Power Domain 0 Configuration Register (PDCFG0) 116
8.6.16 Power Domain 1 Configuration Register (PDCFG1) 117
8.6.17 Module Status *n* Register (MDSTAT*n*) 118
8.6.18 PSC0 Module Control *n* Register (modules 0-15) (MDCTL*n*) 119
8.6.19 PSC1 Module Control *n* Register (modules 0-31) (MDCTL*n*) 120

9 Power Management 121
9.1 Introduction 122
9.2 Power Consumption Overview 122
9.3 PSC and PLLC Overview 122
9.4 Features 123
9.5 Clock Management 124
9.5.1 Module Clock ON/OFF 124
9.5.2 Module Clock Frequency Scaling 124
9.5.3 PLL Bypass and Power Down 124
9.6 DSP Sleep Mode Management 125
9.6.1 C674x DSP CPU Sleep Mode 125
9.6.2 C674x Megamodule Sleep Mode 125

10 System Configuration (SYSCFG) Module 126
10.1 Introduction 127
10.2 Protection 128
10.2.1 Requirements to Access SYSCFG Registers 128
10.3 Master Priority Control 129
10.4 Interrupt Support 130
10.4.1 Interrupt Events and Requests 130
10.4.2 Interrupt Multiplexing 130
10.4.3 Host-DSP Communication Interrupts 130
10.5 SYSCFG Registers 130
10.5.1 Revision Identification Register (REVID) 132
10.5.2 Device Identification Register 0 (DEVIDR0) 132
10.5.3 Boot Configuration Register (BOOTCFG) 133
10.5.4 Silicon Revision Identification Register (CHIPREVID) 133
10.5.5 Kick Registers (KICK0R-KICK1R) 134
10.5.6 Host 1 Configuration Register (HOST1CFG) 135
10.5.7 Interrupt Registers 136
10.5.8 Fault Registers 139
10.5.9 Master Priority Registers (MSTPRI0-MSTPRI2) 141
10.5.10 Pin Multiplexing Control Registers (PINMUX0-PINMUX19) 144
10.5.11 Suspend Source Register (SUSPSRC) 171
10.5.12 Chip Signal Register (CHIPSIG) 173
10.5.13 Chip Signal Clear Register (CHIPSIG_CLR) 174

Copyright © 2013–2016, Texas Instruments Incorporated

10.5.14 Chip Configuration 0 Register (CFGCHIP0) ... 175
10.5.15 Chip Configuration 1 Register (CFGCHIP1) ... 176
10.5.16 Chip Configuration 4 Register (CFGCHIP4) ... 178
11 Boot Considerations ... 179
11.1 Introduction ... 180
12 Dual Data Movement Accelerator (dMAX) ... 181
13 Enhanced Capture (eCAP) Module ... 183
13.1 Introduction ... 184
13.1.1 Purpose of the Peripheral ... 184
13.1.2 Features ... 184
13.2 Architecture ... 185
13.2.1 Capture and APWM Operating Mode ... 186
13.2.2 Capture Mode Description ... 187
13.3 Applications ... 194
13.3.1 Absolute Time-Stamp Operation Rising Edge Trigger Example ... 195
13.3.2 Absolute Time-Stamp Operation Rising and Falling Edge Trigger Example ... 197
13.3.3 Time Difference (Delta) Operation Rising Edge Trigger Example ... 199
13.3.4 Time Difference (Delta) Operation Rising and Falling Edge Trigger Example ... 201
13.3.5 Application of the APWM Mode ... 203
13.4 Registers ... 210
13.4.1 Time-Stamp Counter Register (TSCTR) ... 210
13.4.2 Counter Phase Control Register (CTRPHS) ... 211
13.4.3 Capture 1 Register (CAP1) ... 211
13.4.4 Capture 2 Register (CAP2) ... 212
13.4.5 Capture 3 Register (CAP3) ... 212
13.4.6 Capture 4 Register (CAP4) ... 213
13.4.7 ECAP Control Register 1 (ECCTL1) ... 213
13.4.8 ECAP Control Register 2 (ECCTL2) ... 215
13.4.9 ECAP Interrupt Enable Register (ECEINT) ... 216
13.4.10 ECAP Interrupt Flag Register (ECFLG) ... 218
13.4.11 ECAP Interrupt Clear Register (ECCLR) ... 219
13.4.12 ECAP Interrupt Forcing Register (ECFRC) ... 220
13.4.13 Revision ID Register (REVID) ... 221
14 Enhanced High-Resolution Pulse-Width Modulator (eHRPWM) ... 222
14.1 Introduction ... 223
14.1.1 Introduction ... 223
14.1.2 Submodule Overview ... 223
14.1.3 Register Mapping ... 227
14.2 Architecture ... 228
14.2.1 Overview ... 228
14.2.2 Proper Interrupt Initialization Procedure ... 231
14.2.3 Time-Base (TB) Submodule ... 232
14.2.4 Counter-Compare (CC) Submodule ... 241
14.2.5 Action-Qualifier (AQ) Submodule ... 246
14.2.6 Dead-Band Generator (DB) Submodule ... 264
14.2.7 PWM-Chopper (PC) Submodule ... 268
14.2.8 Trip-Zone (TZ) Submodule ... 272
14.2.9 Event-Trigger (ET) Submodule ... 276
14.2.10 High-Resolution PWM (HRPWM) Submodule ... 280
14.3 Applications to Power Topologies ... 287
14.3.1 Overview of Multiple Modules ... 287
14.3.2 Key Configuration Capabilities ... 288

Copyright © 2013–2016, Texas Instruments Incorporated

14.3.3 Controlling Multiple Buck Converters With Independent Frequencies 289
14.3.4 Controlling Multiple Buck Converters With Same Frequencies 292
14.3.5 Controlling Multiple Half H-Bridge (HHB) Converters 295
14.3.6 Controlling Dual 3-Phase Inverters for Motors (ACI and PMSM) 298
14.3.7 Practical Applications Using Phase Control Between PWM Modules 302
14.3.8 Controlling a 3-Phase Interleaved DC/DC Converter 303
14.3.9 Controlling Zero Voltage Switched Full Bridge (ZVSFB) Converter 308
14.4 Registers 311
14.4.1 Time-Base Submodule Registers 311
14.4.2 Counter-Compare Submodule Registers 315
14.4.3 Action-Qualifier Submodule Registers 318
14.4.4 Dead-Band Generator Submodule Registers 322
14.4.5 PWM-Chopper Submodule Register 325
14.4.6 Trip-Zone Submodule Registers 326
14.4.7 Event-Trigger Submodule Registers 330
14.4.8 High-Resolution PWM Submodule Registers 333
15 Enhanced Quadrature Encoder Pulse (eQEP) Module 336
15.1 Introduction 337
15.2 Architecture 340
15.2.1 EQEP Inputs 340
15.2.2 Functional Description 340
15.2.3 Quadrature Decoder Unit (QDU) 342
15.2.4 Position Counter and Control Unit (PCCU) 345
15.2.5 eQEP Edge Capture Unit 353
15.2.6 eQEP Watchdog 356
15.2.7 Unit Timer Base 357
15.2.8 eQEP Interrupt Structure 357
15.3 eQEP Registers 358
15.3.1 eQEP Position Counter Register (QPOSCNT) 359
15.3.2 eQEP Position Counter Initialization Register (QPOSINIT) 359
15.3.3 eQEP Maximum Position Count Register (QPOSMAX) 359
15.3.4 eQEP Position-Compare Register (QPOSCMP) 360
15.3.5 eQEP Index Position Latch Register (QPOSILAT) 360
15.3.6 eQEP Strobe Position Latch Register (QPOSSLAT) 360
15.3.7 eQEP Position Counter Latch Register (QPOSLAT) 361
15.3.8 eQEP Unit Timer Register (QUTMR) 361
15.3.9 eQEP Unit Period Register (QUPRD) 361
15.3.10 eQEP Watchdog Timer Register (QWDTMR) 362
15.3.11 eQEP Watchdog Period Register (QWDPRD) 362
15.3.12 QEP Decoder Control Register (QDECCTL) 363
15.3.13 eQEP Control Register (QEPCTL) 363
15.3.14 eQEP Capture Control Register (QCAPCTL) 366
15.3.15 eQEP Position-Compare Control Register (QPOSCTL) 367
15.3.16 eQEP Interrupt Enable Register (QEINT) 368
15.3.17 eQEP Interrupt Flag Register (QFLG) 369
15.3.18 eQEP Interrupt Clear Register (QCLR) 370
15.3.19 eQEP Interrupt Force Register (QFRC) 372
15.3.20 eQEP Status Register (QEPSTS) 373
15.3.21 eQEP Capture Timer Register (QCTMR) 374
15.3.22 eQEP Capture Period Register (QCPRD) 374
15.3.23 eQEP Capture Timer Latch Register (QCTMRLAT) 374
15.3.24 eQEP Capture Period Latch Register (QCPRDLAT) 375
15.3.25 eQEP Revision ID Register (REVID) 375

Copyright © 2013–2016, Texas Instruments Incorporated

16 Enhanced Direct Memory Access (EDMA3) Controller 376
16.1 Introduction 377
16.1.1 Overview 377
16.1.2 Features 377
16.1.3 Functional Block Diagram 379
16.1.4 Terminology Used in This Document 379
16.2 Architecture 381
16.2.1 Functional Overview 381
16.2.2 Types of EDMA3 Transfers 384
16.2.3 Parameter RAM (PaRAM) 387
16.2.4 Initiating a DMA Transfer 397
16.2.5 Completion of a DMA Transfer 400
16.2.6 Event, Channel, and PaRAM Mapping 401
16.2.7 EDMA3 Channel Controller Regions 404
16.2.8 Chaining EDMA3 Channels 406
16.2.9 EDMA3 Interrupts 406
16.2.10 Event Queue(s) 413
16.2.11 EDMA3 Transfer Controller (EDMA3TC) 415
16.2.12 Event Dataflow 418
16.2.13 EDMA3 Prioritization 419
16.2.14 EDMA3CC and EDMA3TC Performance and System Considerations 421
16.2.15 EDMA3 Operating Frequency (Clock Control) 422
16.2.16 Reset Considerations 422
16.2.17 Power Management 422
16.2.18 Emulation Considerations 423
16.3 Transfer Examples 423
16.3.1 Block Move Example 423
16.3.2 Subframe Extraction Example 425
16.3.3 Data Sorting Example 426
16.3.4 Peripheral Servicing Example 428
16.4 Registers 440
16.4.1 Parameter RAM (PaRAM) Entries 440
16.4.2 EDMA3 Channel Controller (EDMA3CC) Registers 447
16.4.3 EDMA3 Transfer Controller (EDMA3TC) Registers 486
16.5 Tips 507
16.5.1 Debug Checklist 507
16.5.2 Miscellaneous Programming/Debug Tips 508
16.6 Setting Up a Transfer 509
17 General-Purpose Input/Output (GPIO) 510
17.1 Introduction 511
17.1.1 Purpose of the Peripheral 511
17.1.2 Features 511
17.1.3 Functional Block Diagram 511
17.1.4 Industry Standard(s) Compliance Statement 511
17.2 Architecture 512
17.2.1 Clock Control 512
17.2.2 Signal Descriptions 512
17.2.3 Pin Multiplexing 512
17.2.4 Endianness Considerations 512
17.2.5 GPIO Register Structure 513
17.2.6 Using a GPIO Signal as an Output 516
17.2.7 Using a GPIO Signal as an Input 517
17.2.8 Reset Considerations 517

Copyright © 2013–2016, Texas Instruments Incorporated

17.2.9 Initialization 518
17.2.10 Interrupt Support 518
17.2.11 EDMA Event Support 519
17.2.12 Power Management 519
17.2.13 Emulation Considerations 519
17.3 Registers 520
17.3.1 Revision ID Register (REVID) 521
17.3.2 GPIO Interrupt Per-Bank Enable Register (BINTEN) 522
17.3.3 GPIO Direction Registers (DIR*n*) 523
17.3.4 GPIO Output Data Registers (OUT_DATA*n*) 525
17.3.5 GPIO Set Data Registers (SET_DATA*n*) 527
17.3.6 GPIO Clear Data Registers (CLR_DATA*n*) 529
17.3.7 GPIO Input Data Registers (IN_DATA*n*) 531
17.3.8 GPIO Set Rising Edge Interrupt Registers (SET_RIS_TRIG*n*) 533
17.3.9 GPIO Clear Rising Edge Interrupt Registers (CLR_RIS_TRIG*n*) 535
17.3.10 GPIO Set Falling Edge Interrupt Registers (SET_FAL_TRIG*n*) 537
17.3.11 GPIO Clear Falling Edge Interrupt Registers (CLR_FAL_TRIG*n*) 539
17.3.12 GPIO Interrupt Status Registers (INTSTAT*n*) 541

18 Inter-Integrated Circuit (I2C) Module 543
18.1 Introduction 544
18.1.1 Purpose of the Peripheral 544
18.1.2 Features 544
18.1.3 Functional Block Diagram 545
18.1.4 Industry Standard(s) Compliance Statement 545
18.2 Architecture 546
18.2.1 Bus Structure 546
18.2.2 Clock Generation 547
18.2.3 Clock Synchronization 548
18.2.4 Signal Descriptions 548
18.2.5 START and STOP Conditions 549
18.2.6 Serial Data Formats 550
18.2.7 Operating Modes 552
18.2.8 NACK Bit Generation 553
18.2.9 Arbitration 554
18.2.10 Reset Considerations 555
18.2.11 Initialization 555
18.2.12 Interrupt Support 556
18.2.13 DMA Events Generated by the I2C Peripheral 557
18.2.14 Power Management 557
18.2.15 Emulation Considerations 557
18.3 Registers 558
18.3.1 I2C Own Address Register (ICOAR) 559
18.3.2 I2C Interrupt Mask Register (ICIMR) 560
18.3.3 I2C Interrupt Status Register (ICSTR) 561
18.3.4 I2C Clock Divider Registers (ICCLKL and ICCLKH) 564
18.3.5 I2C Data Count Register (ICCNT) 565
18.3.6 I2C Data Receive Register (ICDRR) 566
18.3.7 I2C Slave Address Register (ICSAR) 567
18.3.8 I2C Data Transmit Register (ICDXR) 568
18.3.9 I2C Mode Register (ICMDR) 569
18.3.10 I2C Interrupt Vector Register (ICIVR) 573
18.3.11 I2C Extended Mode Register (ICEMDR) 574
18.3.12 I2C Prescaler Register (ICPSC) 575

Copyright © 2013–2016, Texas Instruments Incorporated

18.3.13 I2C Revision Identification Register (REVID1) 576
18.3.14 I2C Revision Identification Register (REVID2) 576
18.3.15 I2C DMA Control Register (ICDMAC) 577
18.3.16 I2C Pin Function Register (ICPFUNC) 578
18.3.17 I2C Pin Direction Register (ICPDIR) 579
18.3.18 I2C Pin Data In Register (ICPDIN) 580
18.3.19 I2C Pin Data Out Register (ICPDOUT) 581
18.3.20 I2C Pin Data Set Register (ICPDSET) 582
18.3.21 I2C Pin Data Clear Register (ICPDCLR) 583

19 Multichannel Audio Serial Port (McASP) 584
19.0.22 Features 585
19.0.23 Protocols Supported 586
19.0.24 Functional Block Diagram 587
19.0.25 Definition of Terms 595
19.0.26 Overview 598
19.0.27 Clock and Frame Sync Generators 598
19.0.28 Reset Considerations 638
19.0.29 EDMA Event Support 638
19.0.30 Power Management 638
19.1 Registers 639
19.1.1 Register Bit Restrictions 642
19.1.2 Revision Identification Register (REV) 643
19.1.3 Pin Function Register (PFUNC) 644
19.1.4 Pin Direction Register (PDIR) 646
19.1.5 Pin Data Output Register (PDOUT) 648
19.1.6 Pin Data Input Register (PDIN) 650
19.1.7 Pin Data Set Register (PDSET) 652
19.1.8 Pin Data Clear Register (PDCLR) 654
19.1.9 Global Control Register (GBLCTL) 656
19.1.10 Audio Mute Control Register (AMUTE) 658
19.1.11 Digital Loopback Control Register (DLBCTL) 660
19.1.12 Digital Mode Control Register (DITCTL) 661
19.1.13 Receiver Global Control Register (RGBLCTL) 662
19.1.14 Receive Format Unit Bit Mask Register (RMASK) 663
19.1.15 Receive Bit Stream Format Register (RFMT) 664
19.1.16 Receive Frame Sync Control Register (AFSRCTL) 666
19.1.17 Receive Clock Control Register (ACLKRCTL) 667
19.1.18 Receive High-Frequency Clock Control Register (AHCLKRCTL) 668
19.1.19 Receive TDM Time Slot Register (RTDM) 669
19.1.20 Receiver Interrupt Control Register (RINTCTL) 670
19.1.21 Receiver Status Register (RSTAT) 671
19.1.22 Current Receive TDM Time Slot Registers (RSLOT) 672
19.1.23 Receive Clock Check Control Register (RCLKCHK) 673
19.1.24 Receiver DMA Event Control Register (REVTCTL) 674
19.1.25 Transmitter Global Control Register (XGBLCTL) 675
19.1.26 Transmit Format Unit Bit Mask Register (XMASK) 676
19.1.27 Transmit Bit Stream Format Register (XFMT) 677
19.1.28 Transmit Frame Sync Control Register (AFSXCTL) 679
19.1.29 Transmit Clock Control Register (ACLKXCTL) 680
19.1.30 Transmit High-Frequency Clock Control Register (AHCLKXCTL) 681
19.1.31 Transmit TDM Time Slot Register (XTDM) 682
19.1.32 Transmitter Interrupt Control Register (XINTCTL) 683
19.1.33 Transmitter Status Register (XSTAT) 684

Copyright © 2013–2016, Texas Instruments Incorporated

19.1.34 Current Transmit TDM Time Slot Register (XSLOT) 685
19.1.35 Transmit Clock Check Control Register (XCLKCHK) 686
19.1.36 Transmitter DMA Event Control Register (XEVTCTL) 687
19.1.37 Serializer Control Registers (SRCTL*n*) 688
19.1.38 DIT Left Channel Status Registers (DITCSRA0-DITCSRA5) 689
19.1.39 DIT Right Channel Status Registers (DITCSRB0-DITCSRB5) 689
19.1.40 DIT Left Channel User Data Registers (DITUDRA0-DITUDRA5) 690
19.1.41 DIT Right Channel User Data Registers (DITUDRB0-DITUDRB5) 690
19.1.42 Transmit Buffer Registers (XBUF*n*) 691
19.1.43 Receive Buffer Registers (RBUF*n*) 691
19.1.44 AFIFO Revision Identification Register (AFIFOREV) 692
19.1.45 Write FIFO Control Register (WFIFOCTL) 693
19.1.46 Write FIFO Status Register (WFIFOSTS) 694
19.1.47 Read FIFO Control Register (RFIFOCTL) 695
19.1.48 Read FIFO Status Register (RFIFOSTS) 696

20 Serial Peripheral Interface (SPI) 697

20.1 Introduction 698
20.1.1 Purpose of the Peripheral 698
20.1.2 Features 698
20.1.3 Functional Block Diagram 699
20.1.4 Industry Standard(s) Compliance Statement 699
20.2 Architecture 700
20.2.1 Clock 700
20.2.2 Signal Descriptions 700
20.2.3 Operation Modes 700
20.2.4 Programmable Registers 701
20.2.5 Master Mode Settings 702
20.2.6 Slave Mode Settings 704
20.2.7 SPI Operation: 3-Pin Mode 705
20.2.8 SPI Operation: 4-Pin with Chip Select Mode 706
20.2.9 SPI Operation: 4-Pin with Enable Mode 708
20.2.10 SPI Operation: 5-Pin Mode 710
20.2.11 Data Formats 712
20.2.12 Interrupt Support 715
20.2.13 DMA Events Support 716
20.2.14 Robustness Features 716
20.2.15 Reset Considerations 718
20.2.16 Power Management 718
20.2.17 General-Purpose I/O Pin 719
20.2.18 Emulation Considerations 719
20.2.19 Initialization 719
20.2.20 Timing Diagrams 720
20.3 Registers 726
20.3.1 SPI Global Control Register 0 (SPIGCR0) 726
20.3.2 SPI Global Control Register 1 (SPIGCR1) 727
20.3.3 SPI Interrupt Register (SPIINT0) 729
20.3.4 SPI Interrupt Level Register (SPILVL) 731
20.3.5 SPI Flag Register (SPIFLG) 732
20.3.6 SPI Pin Control Register 0 (SPIPC0) 734
20.3.7 SPI Pin Control Register 1 (SPIPC1) 735
20.3.8 SPI Pin Control Register 2 (SPIPC2) 736
20.3.9 SPI Pin Control Register 3 (SPIPC3) 737
20.3.10 SPI Pin Control Register 4 (SPIPC4) 738

Copyright © 2013–2016, Texas Instruments Incorporated

20.3.11 SPI Pin Control Register 5 (SPIPC5) 739
20.3.12 SPI Transmit Data Register 0 (SPIDAT0) 740
20.3.13 SPI Transmit Data Register 1 (SPIDAT1) 741
20.3.14 SPI Receive Buffer Register (SPIBUF) 742
20.3.15 SPI Emulation Register (SPIEMU) 744
20.3.16 SPI Delay Register (SPIDELAY) 745
20.3.17 SPI Default Chip Select Register (SPIDEF) 748
20.3.18 SPI Data Format Registers (SPIFMT*n*) 749
20.3.19 SPI Interrupt Vector Register 1 (INTVEC1) 751

21 64-Bit Timer Plus 752
21.1 Introduction 753
21.1.1 Purpose of the Peripheral 753
21.1.2 Features 753
21.1.3 Block Diagram 754
21.1.4 Industry Standard Compatibility Statement 754
21.1.5 Architecture – General-Purpose Timer Mode 754
21.1.6 Architecture – Watchdog Timer Mode 766
21.1.7 Reset Considerations 768
21.1.8 Interrupt Support 768
21.1.9 DMA Event Support 768
21.1.10 TM64P_OUT Event Support 769
21.1.11 Interrupt/DMA Event Generation Control and Status 770
21.1.12 Power Management 770
21.1.13 Emulation Considerations 770
21.2 Registers 771
21.2.1 Revision ID Register (REVID) 773
21.2.2 Emulation Management Register (EMUMGT) 773
21.2.3 GPIO Interrupt Control and Enable Register (GPINTGPEN) 774
21.2.4 GPIO Data and Direction Register (GPDATGPDIR) 775
21.2.5 Timer Counter Registers (TIM12 and TIM34) 776
21.2.6 Timer Period Registers (PRD12 and PRD34) 777
21.2.7 Timer Control Register (TCR) 778
21.2.8 Timer Global Control Register (TGCR) 780
21.2.9 Watchdog Timer Control Register (WDTCR) 781
21.2.10 Timer Reload Register 12 (REL12) 782
21.2.11 Timer Reload Register 34 (REL34) 782
21.2.12 Timer Capture Register 12 (CAP12) 783
21.2.13 Timer Capture Register 34 (CAP34) 783
21.2.14 Timer Interrupt Control and Status Register (INTCTLSTAT) 784

22 Universal Asynchronous Receiver/Transmitter (UART) 786
22.1 Introduction 787
22.1.1 Purpose of the Peripheral 787
22.1.2 Features 787
22.1.3 Functional Block Diagram 787
22.1.4 Industry Standard(s) Compliance Statement 787
22.2 Peripheral Architecture 789
22.2.1 Clock Generation and Control 789
22.2.2 Signal Descriptions 791
22.2.3 Pin Multiplexing 791
22.2.4 Protocol Description 791
22.2.5 Operation 793
22.2.6 Reset Considerations 797
22.2.7 Initialization 797

Copyright © 2013–2016, Texas Instruments Incorporated

22.2.8 Interrupt Support 797
22.2.9 DMA Event Support 799
22.2.10 Power Management 799
22.2.11 Emulation Considerations 799
22.2.12 Exception Processing 799
22.3 Registers 800
22.3.1 Receiver Buffer Register (RBR) 801
22.3.2 Transmitter Holding Register (THR) 802
22.3.3 Interrupt Enable Register (IER) 803
22.3.4 Interrupt Identification Register (IIR) 804
22.3.5 FIFO Control Register (FCR) 805
22.3.6 Line Control Register (LCR) 807
22.3.7 Modem Control Register (MCR) 809
22.3.8 Line Status Register (LSR) 810
22.3.9 Modem Status Register (MSR) 813
22.3.10 Scratch Pad Register (SCR) 814
22.3.11 Divisor Latches (DLL and DLH) 814
22.3.12 Revision Identification Registers (REVID1 and REVID2) 816
22.3.13 Power and Emulation Management Register (PWREMU_MGMT) 817
22.3.14 Mode Definition Register (MDR) 818
Revision History 819

Copyright © 2013–2016, Texas Instruments Incorporated

List of Figures

1-1. TMS320DA804/DA803/DA802 DSP Block Diagram 40
2-1. TMS320C674x Megamodule Block Diagram 42
3-1. System Interconnect Block Diagram 50
5-1. MPU Block Diagram 54
5-2. Permission Fields 56
5-3. Revision ID Register (REVID) 60
5-4. Configuration Register (CONFIG) 60
5-5. Interrupt Raw Status/Set Register (IRAWSTAT) 61
5-6. Interrupt Enable Status/Clear Register (IENSTAT) 62
5-7. Interrupt Enable Set Register (IENSET) 63
5-8. Interrupt Enable Clear Register (IENCLR) 63
5-9. MPU1 Programmable Range *n* Start Address Register (PROG*n*_MPSAR) 64
5-10. MPU1 Programmable Range *n* End Address Register (PROG*n*_MPEAR) 65
5-11. Programmable Range Memory Protection Page Attributes Register (PROG*n*_MPPA) 66
5-12. Fault Address Register (FLTADDRR) 67
5-13. Fault Status Register (FLTSTAT) 68
5-14. Fault Clear Register (FLTCLR) 69
6-1. Overall Clocking Diagram 72
7-1. PLL0 Structure 77
7-2. Revision Identification Register (REVID) 81
7-3. Reset Type Status Register (RSTYPE) 81
7-4. PLL Control Register (PLLCTL) 82
7-5. PLL Multiplier Control Register (PLLM) 83
7-6. PLL Pre-Divider Control Register (PREDIV) 83
7-7. PLL Controller Divider 1 Register (PLLDIV1) 84
7-8. PLL Controller Divider 2 Register (PLLDIV2) 84
7-9. PLL Controller Divider 3 Register (PLLDIV3) 85
7-10. PLL Controller Divider 4 Register (PLLDIV4) 85
7-11. PLL Controller Divider 5 Register (PLLDIV5) 86
7-12. PLL Controller Divider 6 Register (PLLDIV6) 86
7-13. PLL Controller Divider 7 Register (PLLDIV7) 87
7-14. PLL Post-Divider Control Register (POSTDIV) 88
7-15. PLL Controller Command Register (PLLCMD) 88
7-16. PLL Controller Status Register (PLLSTAT) 89
7-17. PLL Controller Clock Align Control Register (ALNCTL) 90
7-18. PLLDIV Ratio Change Status Register (DCHANGE) 91
7-19. Clock Enable Control Register (CKEN) 92
7-20. Clock Status Register (CKSTAT) 93
7-21. SYSCLK Status Register (SYSTAT) 94
7-22. Emulation Performance Counter 0 Register (EMUCNT0) 95
7-23. Emulation Performance Counter 1 Register (EMUCNT1) 95
8-1. Revision Identification Register (REVID) 106
8-2. Interrupt Evaluation Register (INTEVAL) 106
8-3. PSC0 Module Error Pending Register 0 (MERRPR0) 107
8-4. PSC1 Module Error Pending Register 0 (MERRPR0) 107
8-5. PSC0 Module Error Clear Register 0 (MERRCR0) 108
8-6. PSC1 Module Error Clear Register 0 (MERRCR0) 108

Copyright © 2013–2016, Texas Instruments Incorporated

8-7. Power Error Pending Register (PERRPR) 109
8-8. Power Error Clear Register (PERRCR) 109
8-9. Power Domain Transition Command Register (PTCMD) 110
8-10. Power Domain Transition Status Register (PTSTAT) 111
8-11. Power Domain 0 Status Register (PDSTAT0) 112
8-12. Power Domain 1 Status Register (PDSTAT1) 113
8-13. Power Domain 0 Control Register (PDCTL0) 114
8-14. Power Domain 1 Control Register (PDCTL1) 115
8-15. Power Domain 0 Configuration Register (PDCFG0) 116
8-16. Power Domain 1 Configuration Register (PDCFG1) 117
8-17. Module Status *n* Register (MDSTAT*n*) 118
8-18. PSC0 Module Control *n* Register (MDCTL*n*) 119
8-19. PSC1 Module Control *n* Register (MDCTL*n*) 120
10-1. Revision Identification Register (REVID) 132
10-2. Device Identification Register 0 (DEVIDR0) 132
10-3. Boot Configuration Register (BOOTCFG) 133
10-4. Silicon Revision Identification Register (CHIPREVID) 133
10-5. Kick 0 Register (KICK0R) 134
10-6. Kick 1 Register (KICK1R) 134
10-7. Host 1 Configuration Register (HOST1CFG) 135
10-8. Interrupt Raw Status/Set Register (IRAWSTAT) 136
10-9. Interrupt Enable Status/Clear Register (IENSTAT) 137
10-10. Interrupt Enable Register (IENSET) 138
10-11. Interrupt Enable Clear Register (IENCLR) 138
10-12. End of Interrupt Register (EOI) 139
10-13. Fault Address Register (FLTADDRR) 139
10-14. Fault Status Register (FLTSTAT) 140
10-15. Master Priority 0 Register (MSTPRI0) 141
10-16. Master Priority 1 Register (MSTPRI1) 142
10-17. Master Priority 2 Register (MSTPRI2) 143
10-18. Pin Multiplexing Control 0 Register (PINMUX0) 144
10-19. Pin Multiplexing Control 1 Register (PINMUX1) 145
10-20. Pin Multiplexing Control 2 Register (PINMUX2) 146
10-21. Pin Multiplexing Control 3 Register (PINMUX3) 147
10-22. Pin Multiplexing Control 4 Register (PINMUX4) 148
10-23. Pin Multiplexing Control 5 Register (PINMUX5) 149
10-24. Pin Multiplexing Control 6 Register (PINMUX6) 150
10-25. Pin Multiplexing Control 7 Register (PINMUX7) 152
10-26. Pin Multiplexing Control 8 Register (PINMUX8) 154
10-27. Pin Multiplexing Control 9 Register (PINMUX9) 156
10-28. Pin Multiplexing Control 10 Register (PINMUX10) 158
10-29. Pin Multiplexing Control 11 Register (PINMUX11) 160
10-30. Pin Multiplexing Control 12 Register (PINMUX12) 162
10-31. Pin Multiplexing Control 13 Register (PINMUX13) 163
10-32. Pin Multiplexing Control 14 Register (PINMUX14) 164
10-33. Pin Multiplexing Control 15 Register (PINMUX15) 166
10-34. Pin Multiplexing Control 16 Register (PINMUX16) 167
10-35. Pin Multiplexing Control 17 Register (PINMUX17) 168
10-36. Pin Multiplexing Control 18 Register (PINMUX18) 169

Copyright © 2013–2016, Texas Instruments Incorporated

10-37. Pin Multiplexing Control 19 Register (PINMUX19) 170
10-38. Suspend Source Register (SUSPSRC) 171
10-39. Chip Signal Register (CHIPSIG) 173
10-40. Chip Signal Clear Register (CHIPSIG_CLR) 174
10-41. Chip Configuration 0 Register (CFGCHIP0) 175
10-42. Chip Configuration 1 Register (CFGCHIP1) 176
10-43. Chip Configuration 4 Register (CFGCHIP4) 178
13-1. Multiple eCAP Modules 185
13-2. Capture and APWM Modes of Operation 186
13-3. Capture Function Diagram 187
13-4. Event Prescale Control 188
13-5. Prescale Function Waveforms 188
13-6. Continuous/One-shot Block Diagram 189
13-7. Counter and Synchronization Block Diagram 190
13-8. Interrupts in eCAP Module 192
13-9. PWM Waveform Details Of APWM Mode Operation 193
13-10. Capture Sequence for Absolute Time-Stamp, Rising Edge Detect 195
13-11. Capture Sequence for Absolute Time-Stamp, Rising and Falling Edge Detect 197
13-12. Capture Sequence for Delta Mode Time-Stamp, Rising Edge Detect 199
13-13. Capture Sequence for Delta Mode Time-Stamp, Rising and Falling Edge Detect 201
13-14. PWM Waveform Details of APWM Mode Operation 203
13-15. Multichannel PWM Example Using 4 eCAP Modules 205
13-16. Multiphase (channel) Interleaved PWM Example Using 3 eCAP Modules 208
13-17. Time-Stamp Counter Register (TSCTR) 210
13-18. Counter Phase Control Register (CTRPHS) 211
13-19. Capture 1 Register (CAP1) 211
13-20. Capture 2 Register (CAP2) 212
13-21. Capture 3 Register (CAP3) 212
13-22. Capture 4 Register (CAP4) 213
13-23. ECAP Control Register 1 (ECCTL1) 213
13-24. ECAP Control Register 2 (ECCTL2) 215
13-25. ECAP Interrupt Enable Register (ECEINT) 217
13-26. ECAP Interrupt Flag Register (ECFLG) 218
13-27. ECAP Interrupt Clear Register (ECCLR) 219
13-28. ECAP Interrupt Forcing Register (ECFRC) 220
13-29. Revision ID Register (REVID) 221
14-1. Multiple ePWM Modules 224
14-2. Submodules and Signal Connections for an ePWM Module 225
14-3. ePWM Submodules and Critical Internal Signal Interconnects 226
14-4. Time-Base Submodule Block Diagram 232
14-5. Time-Base Submodule Signals and Registers 233
14-6. Time-Base Frequency and Period 235
14-7. Time-Base Counter Synchronization Scheme 1 236
14-8. Time-Base Up-Count Mode Waveforms 238
14-9. Time-Base Down-Count Mode Waveforms 239
14-10. Time-Base Up-Down-Count Waveforms, TBCTL[PHSDIR = 0] Count Down on Synchronization Event 239
14-11. Time-Base Up-Down Count Waveforms, TBCTL[PHSDIR = 1] Count Up on Synchronization Event 240
14-12. Counter-Compare Submodule 241
14-13. Counter-Compare Submodule Signals and Registers 241

Copyright © 2013–2016, Texas Instruments Incorporated

14-14. Counter-Compare Event Waveforms in Up-Count Mode 244
14-15. Counter-Compare Events in Down-Count Mode 244
14-16. Counter-Compare Events in Up-Down-Count Mode, TBCTL[PHSDIR = 0] Count Down on Synchronization Event 245
14-17. Counter-Compare Events in Up-Down-Count Mode, TBCTL[PHSDIR = 1] Count Up on Synchronization Event 245
14-18. Action-Qualifier Submodule 246
14-19. Action-Qualifier Submodule Inputs and Outputs 247
14-20. Possible Action-Qualifier Actions for EPWMxA and EPWMxB Outputs 248
14-21. Up-Down-Count Mode Symmetrical Waveform 251
14-22. Up, Single Edge Asymmetric Waveform, With Independent Modulation on EPWMxA and EPWMxB—Active High 252
14-23. Up, Single Edge Asymmetric Waveform With Independent Modulation on EPWMxA and EPWMxB—Active Low 254
14-24. Up-Count, Pulse Placement Asymmetric Waveform With Independent Modulation on EPWMxA 256
14-25. Up-Down-Count, Dual Edge Symmetric Waveform, With Independent Modulation on EPWMxA and EPWMxB — Active Low 258
14-26. Up-Down-Count, Dual Edge Symmetric Waveform, With Independent Modulation on EPWMxA and EPWMxB — Complementary 260
14-27. Up-Down-Count, Dual Edge Asymmetric Waveform, With Independent Modulation on EPWMxA—Active Low 262
14-28. Dead-Band Generator Submodule 264
14-29. Configuration Options for the Dead-Band Generator Submodule 265
14-30. Dead-Band Waveforms for Typical Cases (0% < Duty < 100%) 267
14-31. PWM-Chopper Submodule 268
14-32. PWM-Chopper Submodule Signals and Registers 269
14-33. Simple PWM-Chopper Submodule Waveforms Showing Chopping Action Only 270
14-34. PWM-Chopper Submodule Waveforms Showing the First Pulse and Subsequent Sustaining Pulses 270
14-35. PWM-Chopper Submodule Waveforms Showing the Pulse Width (Duty Cycle) Control of Sustaining Pulses 271
14-36. Trip-Zone Submodule 272
14-37. Trip-Zone Submodule Mode Control Logic 275
14-38. Trip-Zone Submodule Interrupt Logic 275
14-39. Event-Trigger Submodule 276
14-40. Event-Trigger Submodule Inter-Connectivity to Interrupt Controller 277
14-41. Event-Trigger Submodule Showing Event Inputs and Prescaled Outputs 277
14-42. Event-Trigger Interrupt Generator 279
14-43. HRPWM System Interface 280
14-44. Resolution Calculations for Conventionally Generated PWM 281
14-45. Operating Logic Using MEP 282
14-46. Required PWM Waveform for a Requested Duty = 40.5% 284
14-47. Low % Duty Cycle Range Limitation Example When PWM Frequency = 1 MHz 286
14-48. High % Duty Cycle Range Limitation Example when PWM Frequency = 1 MHz 286
14-49. Simplified ePWM Module 287
14-50. EPWM1 Configured as a Typical Master, EPWM2 Configured as a Slave 288
14-51. Control of Four Buck Stages. (Note: $F_{PWM1} \neq F_{PWM2} \neq F_{PWM3} \neq F_{PWM4}$) 289
14-52. Buck Waveforms for (Note: Only three bucks shown here) 290
14-53. Control of Four Buck Stages. (Note: $F_{PWM2} = N \times F_{PWM1}$) 292
14-54. Buck Waveforms for (Note: $F_{PWM2} = F_{PWM1}$) 293
14-55. Control of Two Half-H Bridge Stages ($F_{PWM2} = N \times F_{PWM1}$) 295
14-56. Half-H Bridge Waveforms for (Note: $F_{PWM2} = F_{PWM1}$) 296

Copyright © 2013–2016, Texas Instruments Incorporated

14-57. Control of Dual 3-Phase Inverter Stages as Is Commonly Used in Motor Control 298
14-58. 3-Phase Inverter Waveforms for (Only One Inverter Shown) 299
14-59. Configuring Two PWM Modules for Phase Control 302
14-60. Timing Waveforms Associated With Phase Control Between 2 Modules 303
14-61. Control of a 3-Phase Interleaved DC/DC Converter 304
14-62. 3-Phase Interleaved DC/DC Converter Waveforms for 305
14-63. Controlling a Full-H Bridge Stage (F_{PWM2} = F_{PWM1}) 308
14-64. ZVS Full-H Bridge Waveforms 309
14-65. Time-Base Control Register (TBCTL) 311
14-66. Time-Base Status Register (TBSTS) 313
14-67. Time-Base Phase Register (TBPHS) 314
14-68. Time-Base Counter Register (TBCNT) 314
14-69. Time-Base Period Register (TBPRD) 315
14-70. Counter-Compare Control Register (CMPCTL) 316
14-71. Counter-Compare A Register (CMPA) 317
14-72. Counter-Compare B Register (CMPB) 318
14-73. Action-Qualifier Output A Control Register (AQCTLA) 319
14-74. Action-Qualifier Output B Control Register (AQCTLB) 320
14-75. Action-Qualifier Software Force Register (AQSFRC) 321
14-76. Action-Qualifier Continuous Software Force Register (AQCSFRC) 322
14-77. Dead-Band Generator Control Register (DBCTL) 323
14-78. Dead-Band Generator Rising Edge Delay Register (DBRED) 324
14-79. Dead-Band Generator Falling Edge Delay Register (DBFED) 324
14-80. PWM-Chopper Control Register (PCCTL) 325
14-81. Trip-Zone Select Register (TZSEL) 326
14-82. Trip-Zone Control Register (TZCTL) 327
14-83. Trip-Zone Enable Interrupt Register (TZEINT) 327
14-84. Trip-Zone Flag Register (TZFLG) 328
14-85. Trip-Zone Clear Register (TZCLR) 329
14-86. Trip-Zone Force Register (TZFRC) 329
14-87. Event-Trigger Selection Register (ETSEL) 330
14-88. Event-Trigger Prescale Register (ETPS) 331
14-89. Event-Trigger Flag Register (ETFLG) 332
14-90. Event-Trigger Clear Register (ETCLR) 332
14-91. Event-Trigger Force Register (ETFRC) 333
14-92. Time-Base Phase High-Resolution Register (TBPHSHR) 334
14-93. Counter-Compare A High-Resolution Register (CMPAHR) 334
14-94. HRPWM Configuration Register (HRCNFG) 335
15-1. Optical Encoder Disk 337
15-2. QEP Encoder Output Signal for Forward/Reverse Movement 338
15-3. Index Pulse Example 338
15-4. Functional Block Diagram of the eQEP Peripheral 341
15-5. Functional Block Diagram of Decoder Unit 342
15-6. Quadrature Decoder State Machine 344
15-7. Quadrature-clock and Direction Decoding 344
15-8. Position Counter Reset by Index Pulse for 1000 Line Encoder (QPOSMAX = 3999 or F9Fh) 346
15-9. Position Counter Underflow/Overflow (QPOSMAX = 4) 347
15-10. Software Index Marker for 1000-line Encoder (QEPCTL[IEL] = 1) 349
15-11. Strobe Event Latch (QEPCTL[SEL] = 1) 350

Copyright © 2013–2016, Texas Instruments Incorporated

15-12. eQEP Position-compare Unit 351
15-13. eQEP Position-compare Event Generation Points 352
15-14. eQEP Position-compare Sync Output Pulse Stretcher 352
15-15. eQEP Edge Capture Unit 354
15-16. Unit Position Event for Low Speed Measurement (QCAPCTL[UPPS] = 0010) 354
15-17. eQEP Edge Capture Unit - Timing Details 355
15-18. eQEP Watchdog Timer 356
15-19. eQEP Unit Time Base 357
15-20. EQEP Interrupt Generation 357
15-21. eQEP Position Counter Register (QPOSCNT) 359
15-22. eQEP Position Counter Initialization Register (QPOSINIT) 359
15-23. eQEP Maximum Position Count Register (QPOSMAX) 359
15-24. eQEP Position-Compare Register (QPOSCMP) 360
15-25. eQEP Index Position Latch Register (QPOSILAT) 360
15-26. eQEP Strobe Position Latch Register (QPOSSLAT) 360
15-27. eQEP Position Counter Latch Register (QPOSLAT) 361
15-28. eQEP Unit Timer Register (QUTMR) 361
15-29. eQEP Unit Period Register (QUPRD) 361
15-30. eQEP Watchdog Timer Register (QWDTMR) 362
15-31. eQEP Watchdog Period Register (QWDPRD) 362
15-32. QEP Decoder Control Register (QDECCTL) 363
15-33. eQEP Control Register (QEPCTL) 364
15-34. eQEP Capture Control Register (QCAPCTL) 366
15-35. eQEP Position-Compare Control Register (QPOSCTL) 367
15-36. eQEP Interrupt Enable Register (QEINT) 368
15-37. eQEP Interrupt Flag Register (QFLG) 369
15-38. eQEP Interrupt Clear Register (QCLR) 370
15-39. eQEP Interrupt Force Register (QFRC) 372
15-40. eQEP Status Register (QEPSTS) 373
15-41. eQEP Capture Timer Register (QCTMR) 374
15-42. eQEP Capture Period Register (QCPRD) 374
15-43. eQEP Capture Timer Latch Register (QCTMRLAT) 374
15-44. eQEP Capture Period Latch Register (QCPRDLAT) 375
15-45. eQEP Revision ID Register (REVID) 375
16-1. EDMA3 Controller Block Diagram 379
16-2. EDMA3 Channel Controller (EDMA3CC) Block Diagram 382
16-3. EDMA3 Transfer Controller (EDMA3TC) Block Diagram 383
16-4. Definition of ACNT, BCNT, and CCNT 384
16-5. A-Synchronized Transfers (ACNT = n, BCNT = 4, CCNT = 3) 385
16-6. AB-Synchronized Transfers (ACNT = n, BCNT = 4, CCNT = 3) 386
16-7. PaRAM Set 387
16-8. Linked Transfer Example 395
16-9. Link-to-Self Transfer Example 396
16-10. QDMA Channel to PaRAM Mapping 403
16-11. Shadow Region Registers 405
16-12. Interrupt Diagram 409
16-13. Error Interrupt Operation 412
16-14. EDMA3 Prioritization 419
16-15. Block Move Example 423

Copyright © 2013–2016, Texas Instruments Incorporated

16-16. Block Move Example PaRAM Configuration 424
16-17. Subframe Extraction Example 425
16-18. Subframe Extraction Example PaRAM Configuration 425
16-19. Data Sorting Example 426
16-20. Data Sorting Example PaRAM Configuration 427
16-21. Servicing Incoming McBSP Data Example 428
16-22. Servicing Incoming McBSP Data Example PaRAM 429
16-23. Servicing Peripheral Burst Example 430
16-24. Servicing Peripheral Burst Example PaRAM 430
16-25. Servicing Continuous McBSP Data Example 431
16-26. Servicing Continuous McBSP Data Example PaRAM 432
16-27. Servicing Continuous McBSP Data Example Reload PaRAM 432
16-28. Ping-Pong Buffering for McBSP Data Example 435
16-29. Ping-Pong Buffering for McBSP Example PaRAM 436
16-30. Ping-Pong Buffering for McBSP Example Pong PaRAM 436
16-31. Ping-Pong Buffering for McBSP Example Ping PaRAM 437
16-32. Intermediate Transfer Completion Chaining Example 439
16-33. Single Large Block Transfer Example 439
16-34. Smaller Packet Data Transfers Example 440
16-35. Channel Options Parameter (OPT) 441
16-36. Channel Source Address Parameter (SRC) 443
16-37. A Count/B Count Parameter (A_B_CNT) 443
16-38. Channel Destination Address Parameter (DST) 444
16-39. Source B Index/Destination B Index Parameter (SRC_DST_BIDX) 444
16-40. Link Address/B Count Reload Parameter (LINK_BCNTRLD) 445
16-41. Source C Index/Destination C Index Parameter (SRC_DST_CIDX) 446
16-42. C Count Parameter (CCNT) 446
16-43. Revision ID Register (REVID) 450
16-44. EDMA3CC Configuration Register (CCCFG) 450
16-45. QDMA Channel *n* Mapping Register (QCHMAP*n*) 452
16-46. DMA Channel Queue Number Register *n* (DMAQNUM*n*) 453
16-47. QDMA Channel Queue Number Register (QDMAQNUM) 454
16-48. Event Missed Register (EMR) 455
16-49. Event Missed Clear Register (EMCR) 456
16-50. QDMA Event Missed Register (QEMR) 457
16-51. QDMA Event Missed Clear Register (QEMCR) 458
16-52. EDMA3CC Error Register (CCERR) 459
16-53. EDMA3CC Error Clear Register (CCERRCLR) 460
16-54. Error Evaluate Register (EEVAL) 461
16-55. DMA Region Access Enable Register for Region *m* (DRAE*m*) 462
16-56. QDMA Region Access Enable for Region *m* (QRAE*m*) 463
16-57. Event Queue Entry Registers (Q*x*E*y*) 464
16-58. Queue *n* Status Register (QSTAT*n*) 465
16-59. Queue Watermark Threshold A Register (QWMTHRA) 466
16-60. EDMA3CC Status Register (CCSTAT) 467
16-61. Event Register (ER) 469
16-62. Event Clear Register (ECR) 470
16-63. Event Set Register (ESR) 471
16-64. Chained Event Register (CER) 472

Copyright © 2013–2016, Texas Instruments Incorporated

16-65. Event Enable Register (EER) 473
16-66. Event Enable Clear Register (EECR) 474
16-67. Event Enable Set Register (EESR) 474
16-68. Secondary Event Register (SER) 475
16-69. Secondary Event Clear Register (SECR) 475
16-70. Interrupt Enable Register (IER) 476
16-71. Interrupt Enable Clear Register (IECR) 477
16-72. Interrupt Enable Set Register (IESR) 477
16-73. Interrupt Pending Register (IPR) 478
16-74. Interrupt Clear Register (ICR) 479
16-75. Interrupt Evaluate Register (IEVAL) 480
16-76. QDMA Event Register (QER) 481
16-77. QDMA Event Enable Register (QEER) 482
16-78. QDMA Event Enable Clear Register (QEECR) 483
16-79. QDMA Event Enable Set Register (QEESR) 483
16-80. QDMA Secondary Event Register (QSER) 484
16-81. QDMA Secondary Event Clear Register (QSECR) 485
16-82. Revision ID Register (REVID) 487
16-83. EDMA3TC Configuration Register (TCCFG) 488
16-84. EDMA3TC Channel Status Register (TCSTAT) 489
16-85. Error Status Register (ERRSTAT) 490
16-86. Error Enable Register (ERREN) 491
16-87. Error Clear Register (ERRCLR) 492
16-88. Error Details Register (ERRDET) 493
16-89. Error Interrupt Command Register (ERRCMD) 494
16-90. Read Command Rate Register (RDRATE) 495
16-91. Source Active Options Register (SAOPT) 496
16-92. Source Active Source Address Register (SASRC) 497
16-93. Source Active Count Register (SACNT) 497
16-94. Source Active Destination Address Register (SADST) 498
16-95. Source Active B-Index Register (SABIDX) 498
16-96. Source Active Memory Protection Proxy Register (SAMPPRXY) 499
16-97. Source Active Count Reload Register (SACNTRLD) 500
16-98. Source Active Source Address B-Reference Register (SASRCBREF) 500
16-99. Source Active Destination Address B-Reference Register (SADSTBREF) 501
16-100. Destination FIFO Set Count Reload Register (DFCNTRLD) 501
16-101. Destination FIFO Set Source Address B-Reference Register (DFSRCBREF) 502
16-102. Destination FIFO Set Destination Address B-Reference Register (DFDSTBREF) 502
16-103. Destination FIFO Options Register *n* (DFOPT*n*) 503
16-104. Destination FIFO Source Address Register *n* (DFSRC*n*) 504
16-105. Destination FIFO Count Register *n* (DFCNT*n*) 504
16-106. Destination FIFO Destination Address Register *n* (DFDST*n*) 505
16-107. Destination FIFO B-Index Register *n* (DFBIDX*n*) 505
16-108. Destination FIFO Memory Protection Proxy Register *n* (DFMPPRXY*n*) 506
17-1. GPIO Block Diagram 512
17-2. Revision ID Register (REVID) 521
17-3. GPIO Interrupt Per-Bank Enable Register (BINTEN) 522
17-4. GPIO Banks 0 and 1 Direction Register (DIR01) 523
17-5. GPIO Banks 2 and 3 Direction Register (DIR23) 523

Copyright © 2013–2016, Texas Instruments Incorporated

17-6. GPIO Banks 4 and 5 Direction Register (DIR45) 523
17-7. GPIO Banks 6 and 7 Direction Register (DIR67) 523
17-8. GPIO Bank 8 Direction Register (DIR8) 524
17-9. GPIO Banks 0 and 1 Output Data Register (OUT_DATA01) 525
17-10. GPIO Banks 2 and 3 Output Data Register (OUT_DATA23) 525
17-11. GPIO Banks 4 and 5 Output Data Register (OUT_DATA45) 525
17-12. GPIO Banks 6 and 7 Output Data Register (OUT_DATA67) 525
17-13. GPIO Bank 8 Output Data Register (OUT_DATA8) 526
17-14. GPIO Banks 0 and 1 Set Data Register (SET_DATA01) 527
17-15. GPIO Banks 2 and 3 Set Data Register (SET_DATA23) 527
17-16. GPIO Banks 4 and 5 Set Data Register (SET_DATA45) 527
17-17. GPIO Banks 6 and 7 Set Data Register (SET_DATA67) 527
17-18. GPIO Bank 8 Set Data Register (SET_DATA8) 528
17-19. GPIO Banks 0 and 1 Clear Data Register (CLR_DATA01) 529
17-20. GPIO Banks 2 and 3 Clear Data Register (CLR_DATA23) 529
17-21. GPIO Banks 4 and 5 Clear Data Register (CLR_DATA45) 529
17-22. GPIO Banks 6 and 7 Clear Data Register (CLR_DATA67) 529
17-23. GPIO Bank 8 Clear Data Register (CLR_DATA8) 530
17-24. GPIO Banks 0 and 1 Input Data Register (IN_DATA01) 531
17-25. GPIO Banks 2 and 3 Input Data Register (IN_DATA23) 531
17-26. GPIO Banks 4 and 5 Input Data Register (IN_DATA45) 531
17-27. GPIO Banks 6 and 7 Input Data Register (IN_DATA67) 531
17-28. GPIO Bank 8 Input Data Register (IN_DATA8) 532
17-29. GPIO Banks 0 and 1 Set Rise Trigger Register (SET_RIS_TRIG01) 533
17-30. GPIO Banks 2 and 3 Set Rise Trigger Register (SET_RIS_TRIG23) 533
17-31. GPIO Banks 4 and 5 Set Rise Trigger Register (SET_RIS_TRIG45) 533
17-32. GPIO Banks 6 and 7 Set Rise Trigger Register (SET_RIS_TRIG67) 533
17-33. GPIO Bank 8 Set Rise Trigger Register (SET_RIS_TRIG8) 534
17-34. GPIO Banks 0 and 1 Clear Rise Trigger Register (CLR_RIS_TRIG01) 535
17-35. GPIO Banks 2 and 3 Clear Rise Trigger Register (CLR_RIS_TRIG23) 535
17-36. GPIO Banks 4 and 5 Clear Rise Trigger Register (CLR_RIS_TRIG45) 535
17-37. GPIO Banks 6 and 7 Clear Rise Trigger Register (CLR_RIS_TRIG67) 535
17-38. GPIO Bank 8 Clear Rise Trigger Register (CLR_RIS_TRIG8) 536
17-39. GPIO Banks 0 and 1 Set Rise Trigger Register (SET_FAL_TRIG01) 537
17-40. GPIO Banks 2 and 3 Set Rise Trigger Register (SET_FAL_TRIG23) 537
17-41. GPIO Banks 4 and 5 Set Rise Trigger Register (SET_FAL_TRIG45) 537
17-42. GPIO Banks 6 and 7 Set Rise Trigger Register (SET_FAL_TRIG67) 537
17-43. GPIO Bank 8 Set Rise Trigger Register (SET_FAL_TRIG8) 538
17-44. GPIO Banks 0 and 1 Clear Rise Trigger Register (CLR_FAL_TRIG01) 539
17-45. GPIO Banks 2 and 3 Clear Rise Trigger Register (CLR_FAL_TRIG23) 539
17-46. GPIO Banks 4 and 5 Clear Rise Trigger Register (CLR_FAL_TRIG45) 539
17-47. GPIO Banks 6 and 7 Clear Rise Trigger Register (CLR_FAL_TRIG67) 539
17-48. GPIO Bank 8 Clear Rise Trigger Register (CLR_FAL_TRIG8) 540
17-49. GPIO Banks 0 and 1 Interrupt Status Register (INTSTAT01) 541
17-50. GPIO Banks 2 and 3 Interrupt Status Register (INTSTAT23) 541
17-51. GPIO Banks 4 and 5 Interrupt Status Register (INTSTAT45) 541
17-52. GPIO Banks 6 and 7 Interrupt Status Register (INTSTAT67) 541
17-53. GPIO Bank 8 Interrupt Status Register (INTSTAT8) 542
18-1. I2C Peripheral Block Diagram 545

Copyright © 2013–2016, Texas Instruments Incorporated

18-2. Multiple I2C Modules Connected 546
18-3. Clocking Diagram for the I2C Peripheral 547
18-4. Synchronization of Two I2C Clock Generators During Arbitration 548
18-5. Bit Transfer on the I2C-Bus 549
18-6. I2C Peripheral START and STOP Conditions 549
18-7. I2C Peripheral Data Transfer 550
18-8. I2C Peripheral 7-Bit Addressing Format (FDF = 0, XA = 0 in ICMDR) 550
18-9. I2C Peripheral 10-Bit Addressing Format With Master-Transmitter Writing to Slave-Receiver (FDF = 0, XA = 1 in ICMDR) 551
18-10. I2C Peripheral Free Data Format (FDF = 1 in ICMDR) 551
18-11. I2C Peripheral 7-Bit Addressing Format With Repeated START Condition (FDF = 0, XA = 0 in ICMDR) 551
18-12. Arbitration Procedure Between Two Master-Transmitters 554
18-13. I2C Own Address Register (ICOAR) 559
18-14. I2C Interrupt Mask Register (ICIMR) 560
18-15. I2C Interrupt Status Register (ICSTR) 561
18-16. I2C Clock Low-Time Divider Register (ICCLKL) 564
18-17. I2C Clock High-Time Divider Register (ICCLKH) 564
18-18. I2C Data Count Register (ICCNT) 565
18-19. I2C Data Receive Register (ICDRR) 566
18-20. I2C Slave Address Register (ICSAR) 567
18-21. I2C Data Transmit Register (ICDXR) 568
18-22. I2C Mode Register (ICMDR) 569
18-23. Block Diagram Showing the Effects of the Digital Loopback Mode (DLB) Bit 572
18-24. I2C Interrupt Vector Register (ICIVR) 573
18-25. I2C Extended Mode Register (ICEMDR) 574
18-26. I2C Prescaler Register (ICPSC) 575
18-27. I2C Revision Identification Register 1 (REVID1) 576
18-28. I2C Revision Identification Register 2 (REVID2) 576
18-29. I2C DMA Control Register (ICDMAC) 577
18-30. I2C Pin Function Register (ICPFUNC) 578
18-31. I2C Pin Direction Register (ICPDIR) 579
18-32. I2C Pin Data In Register (ICPDIN) 580
18-33. I2C Pin Data Out Register (ICPDOUT) 581
18-34. I2C Pin Data Set Register (ICPDSET) 582
18-35. I2C Pin Data Clear Register (ICPDCLR) 583
19-1. McASP Block Diagram 587
19-2. McASP to Parallel 2-Channel DACs 588
19-3. McASP to 6-Channel DAC and 2-Channel DAC 588
19-4. McASP to Digital Amplifier 589
19-5. McASP as Digital Audio Encoder 589
19-6. TDM Format–6 Channel TDM Example 590
19-7. TDM Format Bit Delays from Frame Sync 591
19-8. Inter-IC Sound (I2S) Format 591
19-9. Biphase-Mark Code (BMC) 592
19-10. S/PDIF Subframe Format 593
19-11. S/PDIF Frame Format 594
19-12. Definition of Bit, Word, and Slot 595
19-13. Bit Order and Word Alignment Within a Slot Examples 596
19-14. Definition of Frame and Frame Sync Width 597

Copyright © 2013–2016, Texas Instruments Incorporated

19-15. Transmit Clock Generator Block Diagram 599
19-16. Receive Clock Generator Block Diagram 600
19-17. Frame Sync Generator Block Diagram 601
19-18. Individual Serializer and Connections Within McASP 602
19-19. Receive Format Unit 603
19-20. Transmit Format Unit 604
19-21. McASP I/O Pin Control Block Diagram 606
19-22. McASP I/O Pin to Control Register Mapping 607
19-23. Burst Frame Sync Mode 612
19-24. Transmit DMA Event (AXEVT) Generation in TDM Time Slots 615
19-25. DSP Service Time Upon Transmit DMA Event (AXEVT) 620
19-26. DSP Service Time Upon Receive DMA Event (AREVT) 621
19-27. DMA Events in an Audio Example–Two Events 623
19-28. McASP Audio FIFO (AFIFO) Block Diagram 624
19-29. Data Flow Through Transmit Format Unit 627
19-30. Data Flow Through Receive Format Unit 629
19-31. Audio Mute (AMUTE) Block Diagram 631
19-32. Transmit Clock Failure Detection Circuit Block Diagram 635
19-33. Receive Clock Failure Detection Circuit Block Diagram 636
19-34. Serializers in Loopback Mode 637
19-35. Revision Identification Register (REV) 643
19-36. Pin Function Register (PFUNC) 644
19-37. Pin Direction Register (PDIR) 646
19-38. Pin Data Output Register (PDOUT) 648
19-39. Pin Data Input Register (PDIN) 650
19-40. Pin Data Set Register (PDSET) 652
19-41. Pin Data Clear Register (PDCLR) 654
19-42. Global Control Register (GBLCTL) 656
19-43. Audio Mute Control Register (AMUTE) 658
19-44. Digital Loopback Control Register (DLBCTL) 660
19-45. Digital Mode Control Register (DITCTL) 661
19-46. Receiver Global Control Register (RGBLCTL) 662
19-47. Receive Format Unit Bit Mask Register (RMASK) 663
19-48. Receive Bit Stream Format Register (RFMT) 664
19-49. Receive Frame Sync Control Register (AFSRCTL) 666
19-50. Receive Clock Control Register (ACLKRCTL) 667
19-51. Receive High-Frequency Clock Control Register (AHCLKRCTL) 668
19-52. Receive TDM Time Slot Register (RTDM) 669
19-53. Receiver Interrupt Control Register (RINTCTL) 670
19-54. Receiver Status Register (RSTAT) 671
19-55. Current Receive TDM Time Slot Registers (RSLOT) 672
19-56. Receive Clock Check Control Register (RCLKCHK) 673
19-57. Receiver DMA Event Control Register (REVTCTL) 674
19-58. Transmitter Global Control Register (XGBLCTL) 675
19-59. Transmit Format Unit Bit Mask Register (XMASK) 676
19-60. Transmit Bit Stream Format Register (XFMT) 677
19-61. Transmit Frame Sync Control Register (AFSXCTL) 679
19-62. Transmit Clock Control Register (ACLKXCTL) 680
19-63. Transmit High-Frequency Clock Control Register (AHCLKXCTL) 681

Copyright © 2013–2016, Texas Instruments Incorporated

19-64. Transmit TDM Time Slot Register (XTDM) 682
19-65. Transmitter Interrupt Control Register (XINTCTL) 683
19-66. Transmitter Status Register (XSTAT) 684
19-67. Current Transmit TDM Time Slot Register (XSLOT) 685
19-68. Transmit Clock Check Control Register (XCLKCHK) 686
19-69. Transmitter DMA Event Control Register (XEVTCTL) 687
19-70. Serializer Control Registers (SRCTL*n*) 688
19-71. DIT Left Channel Status Registers (DITCSRA0-DITCSRA5) 689
19-72. DIT Right Channel Status Registers (DITCSRB0-DITCSRB5) 689
19-73. DIT Left Channel User Data Registers (DITUDRA0-DITUDRA5) 690
19-74. DIT Right Channel User Data Registers (DITUDRB0-DITUDRB5) 690
19-75. Transmit Buffer Registers (XBUF*n*) 691
19-76. Receive Buffer Registers (RBUF*n*) 691
19-77. AFIFO Revision Identification Register (AFIFOREV) 692
19-78. Write FIFO Control Register (WFIFOCTL) 693
19-79. Write FIFO Status Register (WFIFOSTS) 694
19-80. Read FIFO Control Register (RFIFOCTL) 695
19-81. Read FIFO Status Register (RFIFOSTS) 696
20-1. SPI Block Diagram 699
20-2. SPI 3-Pin Option 705
20-3. SPI 4-Pin Option with $\overline{\text{SPIx_SCS[n]}}$ 707
20-4. SPI 4-Pin Option with $\overline{\text{SPIx_ENA}}$ 709
20-5. SPI 5-Pin Option with $\overline{\text{SPIx_ENA}}$ and $\overline{\text{SPIx_SCS[n]}}$ 711
20-6. Format for Transmitting 12-Bit Word 712
20-7. Format for 10-Bit Received Word 712
20-8. Clock Mode with POLARITY = 0 and PHASE = 0 713
20-9. Clock Mode with POLARITY = 0 and PHASE = 1 714
20-10. Clock Mode with POLARITY = 1 and PHASE = 0 714
20-11. Clock Mode with POLARITY = 1 and PHASE = 1 714
20-12. Five Bits per Character (5-Pin Option) 715
20-13. SPI 3-Pin Master Mode with WDELAY 720
20-14. SPI 4-Pin with $\overline{\text{SPIx_SCS[n]}}$ Mode with T2CDELAY, WDELAY, and C2TDELAY 721
20-15. SPI 4-Pin with $\overline{\text{SPIx_ENA}}$ Mode Demonstrating T2EDELAY and WDELAY 722
20-16. SPI 5-Pin Mode Demonstrating T2CDELAY, T2EDELAY, and WDELAY 724
20-17. SPI 5-Pin Mode Demonstrating C2TDELAY and C2EDELAY 725
20-18. SPI Global Control Register 0 (SPIGCR0) 726
20-19. SPI Global Control Register 1 (SPIGCR1) 727
20-20. SPI Interrupt Register (SPIINT0) 729
20-21. SPI Interrupt Level Register (SPILVL) 731
20-22. SPI Flag Register (SPIFLG) 732
20-23. SPI Pin Control Register 0 (SPIPC0) 734
20-24. SPI Pin Control Register 1 (SPIPC1) 735
20-25. SPI Pin Control Register 2 (SPIPC2) 736
20-26. SPI Pin Control Register 3 (SPIPC3) 737
20-27. SPI Pin Control Register 4 (SPIPC4) 738
20-28. SPI Pin Control Register 5 (SPIPC5) 739
20-29. SPI Data Register 0 (SPIDAT0) 740
20-30. SPI Data Register 1 (SPIDAT1) 741
20-31. SPI Buffer Register (SPIBUF) 742

Copyright © 2013–2016, Texas Instruments Incorporated

20-32. SPI Emulation Register (SPIEMU) 744
20-33. SPI Delay Register (SPIDELAY) 745
20-34. Example: $t_{C2TDELAY}$ = 8 SPI Module Clock Cycles 746
20-35. Example: $t_{T2CDELAY}$ = 4 SPI Module Clock Cycles 747
20-36. Transmit-Data-Finished-to-$\overline{\text{SPIx_ENA}}$-Inactive-Timeout 747
20-37. Chip-Select-Active-to-$\overline{\text{SPIx_ENA}}$-Signal-Active-Timeout 747
20-38. SPI Default Chip Select Register (SPIDEF) 748
20-39. SPI Data Format Register (SPIFMT*n*) 749
20-40. SPI Interrupt Vector Register 1 (INTVEC1) 751
21-1. Timer Block Diagram 754
21-2. Timer Clock Source Block Diagram 755
21-3. 64-Bit Timer Mode Block Diagram 756
21-4. Dual 32-Bit Timers Chained Mode Block Diagram 759
21-5. Dual 32-Bit Timers Chained Mode Example 759
21-6. Dual 32-Bit Timers Unchained Mode Block Diagram 761
21-7. Dual 32-Bit Timers Unchained Mode Example 762
21-8. 32-Bit Timer Counter Overflow Example 765
21-9. Watchdog Timer Mode Block Diagram 767
21-10. Watchdog Timer Operation State Diagram 767
21-11. Timer Operation in Pulse Mode (CP*n* = 0) 769
21-12. Timer Operation in Clock Mode (CP*n* = 1) 769
21-13. Revision ID Register (REVID) 773
21-14. Emulation Management Register (EMUMGT) 773
21-15. GPIO Interrupt Control and Enable Register (GPINTGPEN) 774
21-16. GPIO Data and Direction Register (GPDATGPDIR) 775
21-17. Timer Counter Register 12 (TIM12) 776
21-18. Timer Counter Register 34 (TIM34) 776
21-19. Timer Period Register 12 (PRD12) 777
21-20. Timer Period Register 34 (PRD34) 777
21-21. Timer Control Register (TCR) 778
21-22. Timer Global Control Register (TGCR) 780
21-23. Watchdog Timer Control Register (WDTCR) 781
21-24. Timer Reload Register 12 (REL12) 782
21-25. Timer Reload Register 34 (REL34) 782
21-26. Timer Capture Register 12 (CAP12) 783
21-27. Timer Capture Register 34 (CAP34) 783
21-28. Timer Interrupt Control and Status Register (INTCTLSTAT) 784
21-29. Timer Compare Register (CMP*n*) 785
22-1. UART Block Diagram 788
22-2. UART Clock Generation Diagram 789
22-3. Relationships Between Data Bit, BCLK, and UART Input Clock 790
22-4. UART Protocol Formats 792
22-5. UART Interface Using Autoflow Diagram 795
22-6. Autoflow Functional Timing Waveforms for $\overline{\text{UARTn_RTS}}$ 796
22-7. Autoflow Functional Timing Waveforms for $\overline{\text{UARTn_CTS}}$ 796
22-8. UART Interrupt Request Enable Paths 798
22-9. Receiver Buffer Register (RBR) 801
22-10. Transmitter Holding Register (THR) 802
22-11. Interrupt Enable Register (IER) 803

Copyright © 2013–2016, Texas Instruments Incorporated

22-12. Interrupt Identification Register (IIR) 804
22-13. FIFO Control Register (FCR) 806
22-14. Line Control Register (LCR) 807
22-15. Modem Control Register (MCR) 809
22-16. Line Status Register (LSR) 810
22-17. Modem Status Register (MSR) 813
22-18. Scratch Pad Register (SCR) 814
22-19. Divisor LSB Latch (DLL) 815
22-20. Divisor MSB Latch (DLH) 815
22-21. Revision Identification Register 1 (REVID1) 816
22-22. Revision Identification Register 2 (REVID2) 816
22-23. Power and Emulation Management Register (PWREMU_MGMT) 817
22-24. Mode Definition Register (MDR) 818

Copyright © 2013–2016, Texas Instruments Incorporated

List of Tables

2-1. DSP Interrupt Map 43
3-1. TMS320DA804/DA803/DA802 DSP System Interconnect Matrix 49
5-1. MPU Memory Regions 55
5-2. MPU1 Default Configuration 55
5-3. Device Master Settings 55
5-4. Request Type Access Controls 57
5-5. MPU_BOOTCFG_ERR Interrupt Sources 59
5-6. Memory Protection Unit 1 (MPU1) Registers 59
5-7. Revision ID Register (REVID) Field Descriptions 60
5-8. Configuration Register (CONFIG) Field Descriptions 60
5-9. Interrupt Raw Status/Set Register (IRAWSTAT) Field Descriptions 61
5-10. Interrupt Enable Status/Clear Register (IENSTAT) Field Descriptions 62
5-11. Interrupt Enable Set Register (IENSET) Field Descriptions 63
5-12. Interrupt Enable Clear Register (IENCLR) Field Descriptions 63
5-13. MPU1 Programmable Range *n* Start Address Register (PROG*n*_MPSAR) Field Descriptions 64
5-14. MPU1 Programmable Range *n* End Address Register (PROG*n*_MPEAR) Field Descriptions 65
5-15. Programmable Range Memory Protection Page Attributes Register (PROG*n*_MPPA) Field Descriptions 66
5-16. Fault Address Register (FLTADDRR) Field Descriptions 67
5-17. Fault Status Register (FLTSTAT) Field Descriptions 68
5-18. Fault Clear Register (FLTCLR) Field Descriptions 69
6-1. Device Clock Inputs 71
6-2. System Clock Domains 71
6-3. Example PLL Frequencies 73
6-4. Peripherals 74
7-1. System PLLC0 Output Clocks 77
7-2. PLL Controller (PLLC) Registers 80
7-3. Revision Identification Register (REVID) Field Descriptions 81
7-4. Reset Type Status Register (RSTYPE) Field Descriptions 81
7-5. PLL Control Register (PLLCTL) Field Descriptions 82
7-6. PLL Multiplier Control Register (PLLM) Field Descriptions 83
7-7. PLL Pre-Divider Control Register (PREDIV) Field Descriptions 83
7-8. PLL Controller Divider 1 Register (PLLDIV1) Field Descriptions 84
7-9. PLL Controller Divider 2 Register (PLLDIV2) Field Descriptions 84
7-10. PLL Controller Divider 3 Register (PLLDIV3) Field Descriptions 85
7-11. PLL Controller Divider 4 Register (PLLDIV4) Field Descriptions 85
7-12. PLL Controller Divider 5 Register (PLLDIV5) Field Descriptions 86
7-13. PLL Controller Divider 6 Register (PLLDIV6) Field Descriptions 86
7-14. PLL Controller Divider 7 Register (PLLDIV7) Field Descriptions 87
7-15. PLL Post-Divider Control Register (POSTDIV) Field Descriptions 88
7-16. PLL Controller Command Register (PLLCMD) Field Descriptions 88
7-17. PLL Controller Status Register (PLLSTAT) Field Descriptions 89
7-18. PLL Controller Clock Align Control Register (ALNCTL) Field Descriptions 90
7-19. PLLDIV Ratio Change Status Register (DCHANGE) Field Descriptions 91
7-20. Clock Enable Control Register (CKEN) Field Descriptions 92
7-21. Clock Status Register (CKSTAT) Field Descriptions 93
7-22. SYSCLK Status Register (SYSTAT) Field Descriptions 94
7-23. Emulation Performance Counter 0 Register (EMUCNT0) Field Descriptions 95

Copyright © 2013–2016, Texas Instruments Incorporated

7-24. Emulation Performance Counter 1 Register (EMUCNT1) Field Descriptions 95
8-1. PSC0 Default Module Configuration 97
8-2. PSC1 Default Module Configuration 98
8-3. Module States 100
8-4. IcePick Emulation Commands 102
8-5. PSC Interrupt Events 102
8-6. Power and Sleep Controller 0 (PSC0) Registers 105
8-7. Power and Sleep Controller 1 (PSC1) Registers 105
8-8. Revision Identification Register (REVID) Field Descriptions 106
8-9. Interrupt Evaluation Register (INTEVAL) Field Descriptions 106
8-10. PSC0 Module Error Pending Register 0 (MERRPR0) Field Descriptions 107
8-11. PSC0 Module Error Clear Register 0 (MERRCR0) Field Descriptions 108
8-12. Power Error Pending Register (PERRPR) Field Descriptions 109
8-13. Power Error Clear Register (PERRCR) Field Descriptions 109
8-14. Power Domain Transition Command Register (PTCMD) Field Descriptions 110
8-15. Power Domain Transition Status Register (PTSTAT) Field Descriptions 111
8-16. Power Domain 0 Status Register (PDSTAT0) Field Descriptions 112
8-17. Power Domain 1 Status Register (PDSTAT1) Field Descriptions 113
8-18. Power Domain 0 Control Register (PDCTL0) Field Descriptions 114
8-19. Power Domain 1 Control Register (PDCTL1) Field Descriptions 115
8-20. Power Domain 0 Configuration Register (PDCFG0) Field Descriptions 116
8-21. Power Domain 1 Configuration Register (PDCFG1) Field Descriptions 117
8-22. Module Status *n* Register (MDSTAT*n*) Field Descriptions 118
8-23. PSC0 Module Control *n* Register (MDCTL*n*) Field Descriptions 119
8-24. PSC1 Module Control *n* Register (MDCTL*n*) Field Descriptions 120
9-1. Power Management Features 123
10-1. System Configuration (SYSCFG) Module Register Access 128
10-2. Master IDs 129
10-3. Default Master Priority 130
10-4. System Configuration Module (SYSCFG) Registers 130
10-5. Revision Identification Register (REVID) Field Descriptions 132
10-6. Device Identification Register 0 (DEVIDR0) Field Descriptions 132
10-7. Boot Configuration Register (BOOTCFG) Field Descriptions 133
10-8. Silicon Revision Identification Register (CHIPREVID) Field Descriptions 133
10-9. Kick 0 Register (KICK0R) Field Descriptions 134
10-10. Kick 1 Register (KICK1R) Field Descriptions 134
10-11. Host 1 Configuration Register (HOST1CFG) Field Descriptions 135
10-12. Interrupt Raw Status/Set Register (IRAWSTAT) Field Descriptions 136
10-13. Interrupt Enable Status/Clear Register (IENSTAT) Field Descriptions 137
10-14. Interrupt Enable Register (IENSET) Field Descriptions 138
10-15. Interrupt Enable Clear Register (IENCLR) Field Descriptions 138
10-16. End of Interrupt Register (EOI) Field Descriptions 139
10-17. Fault Address Register (FLTADDRR) Field Descriptions 139
10-18. Fault Status Register (FLTSTAT) Field Descriptions 140
10-19. Master Priority 0 Register (MSTPRI0) Field Descriptions 141
10-20. Master Priority 1 Register (MSTPRI1) Field Descriptions 142
10-21. Master Priority 2 Register (MSTPRI2) Field Descriptions 143
10-22. Pin Multiplexing Control 0 Register (PINMUX0) Field Descriptions 144
10-23. Pin Multiplexing Control 1 Register (PINMUX1) Field Descriptions 145

Copyright © 2013–2016, Texas Instruments Incorporated

10-24. Pin Multiplexing Control 2 Register (PINMUX2) Field Descriptions 146
10-25. Pin Multiplexing Control 3 Register (PINMUX3) Field Descriptions 147
10-26. Pin Multiplexing Control 4 Register (PINMUX4) Field Descriptions 148
10-27. Pin Multiplexing Control 5 Register (PINMUX5) Field Descriptions 149
10-28. Pin Multiplexing Control 6 Register (PINMUX6) Field Descriptions 150
10-29. Pin Multiplexing Control 7 Register (PINMUX7) Field Descriptions 152
10-30. Pin Multiplexing Control 8 Register (PINMUX8) Field Descriptions 154
10-31. Pin Multiplexing Control 9 Register (PINMUX9) Field Descriptions 156
10-32. Pin Multiplexing Control 10 Register (PINMUX10) Field Descriptions 158
10-33. Pin Multiplexing Control 11 Register (PINMUX11) Field Descriptions 160
10-34. Pin Multiplexing Control 12 Register (PINMUX12) Field Descriptions 162
10-35. Pin Multiplexing Control 13 Register (PINMUX13) Field Descriptions 163
10-36. Pin Multiplexing Control 14 Register (PINMUX14) Field Descriptions 164
10-37. Pin Multiplexing Control 15 Register (PINMUX15) Field Descriptions 166
10-38. Pin Multiplexing Control 16 Register (PINMUX16) Field Descriptions 167
10-39. Pin Multiplexing Control 17 Register (PINMUX17) Field Descriptions 168
10-40. Pin Multiplexing Control 18 Register (PINMUX18) Field Descriptions 169
10-41. Pin Multiplexing Control 19 Register (PINMUX19) Field Descriptions 170
10-42. Suspend Source Register (SUSPSRC) Field Descriptions 171
10-43. Chip Signal Register (CHIPSIG) Field Descriptions 173
10-44. Chip Signal Clear Register (CHIPSIG_CLR) Field Descriptions 174
10-45. Chip Configuration 0 Register (CFGCHIP0) Field Descriptions 175
10-46. Chip Configuration 1 Register (CFGCHIP1) Field Descriptions 176
10-47. Chip Configuration 4 Register (CFGCHIP4) Field Descriptions 178
13-1. ECAP Initialization for CAP Mode Absolute Time, Rising Edge Trigger 196
13-2. ECAP Initialization for CAP Mode Absolute Time, Rising and Falling Edge Trigger 198
13-3. ECAP Initialization for CAP Mode Delta Time, Rising Edge Trigger 200
13-4. ECAP Initialization for CAP Mode Delta Time, Rising and Falling Edge Triggers 202
13-5. ECAP Initialization for APWM Mode 204
13-6. ECAP1 Initialization for Multichannel PWM Generation with Synchronization 206
13-7. ECAP2 Initialization for Multichannel PWM Generation with Synchronization 206
13-8. ECAP3 Initialization for Multichannel PWM Generation with Synchronization 206
13-9. ECAP4 Initialization for Multichannel PWM Generation with Synchronization 206
13-10. ECAP1 Initialization for Multichannel PWM Generation with Phase Control 209
13-11. ECAP2 Initialization for Multichannel PWM Generation with Phase Control 209
13-12. ECAP3 Initialization for Multichannel PWM Generation with Phase Control 209
13-13. Control and Status Register Set 210
13-14. Time-Stamp Counter Register (TSCTR) Field Descriptions 210
13-15. Counter Phase Control Register (CTRPHS) Field Descriptions 211
13-16. Capture 1 Register (CAP1) Field Descriptions 211
13-17. Capture 2 Register (CAP2) Field Descriptions 212
13-18. Capture 3 Register (CAP3) Field Descriptions 212
13-19. Capture 4 Register (CAP4) Field Descriptions 213
13-20. ECAP Control Register 1 (ECCTL1) Field Descriptions 213
13-21. ECAP Control Register 2 (ECCTL2) Field Descriptions 215
13-22. ECAP Interrupt Enable Register (ECEINT) Field Descriptions 217
13-23. ECAP Interrupt Flag Register (ECFLG) Field Descriptions 218
13-24. ECAP Interrupt Clear Register (ECCLR) Field Descriptions 219
13-25. ECAP Interrupt Forcing Register (ECFRC) Field Descriptions 220

Copyright © 2013–2016, Texas Instruments Incorporated

13-26. Revision ID Register (REVID) Field Descriptions 221
14-1. ePWM Module Control and Status Registers Grouped by Submodule 227
14-2. Submodule Configuration Parameters 228
14-3. Time-Base Submodule Registers 233
14-4. Key Time-Base Signals 234
14-5. Counter-Compare Submodule Registers 242
14-6. Counter-Compare Submodule Key Signals 242
14-7. Action-Qualifier Submodule Registers 246
14-8. Action-Qualifier Submodule Possible Input Events 247
14-9. Action-Qualifier Event Priority for Up-Down-Count Mode 249
14-10. Action-Qualifier Event Priority for Up-Count Mode 249
14-11. Action-Qualifier Event Priority for Down-Count Mode 249
14-12. Behavior if CMPA/CMPB is Greater than the Period 250
14-13. EPWMx Initialization for 253
14-14. EPWMx Run Time Changes for 253
14-15. EPWMx Initialization for 255
14-16. EPWMx Run Time Changes for 255
14-17. EPWMx Initialization for 257
14-18. EPWMx Run Time Changes for 257
14-19. EPWMx Initialization for 259
14-20. EPWMx Run Time Changes for 259
14-21. EPWMx Initialization for 261
14-22. EPWMx Run Time Changes for 261
14-23. EPWMx Initialization for 263
14-24. EPWMx Run Time Changes for 263
14-25. Dead-Band Generator Submodule Registers 264
14-26. Classical Dead-Band Operating Modes 266
14-27. PWM-Chopper Submodule Registers 268
14-28. Trip-Zone Submodule Registers 273
14-29. Possible Actions On a Trip Event 274
14-30. Event-Trigger Submodule Registers 276
14-31. Resolution for PWM and HRPWM 281
14-32. HRPWM Submodule Registers 282
14-33. Relationship Between MEP Steps, PWM Frequency and Resolution 283
14-34. CMPA vs Duty (left), and [CMPA:CMPAHR] vs Duty (right) 284
14-35. EPWM1 Initialization for 291
14-36. EPWM2 Initialization for 291
14-37. EPWM3 Initialization for 291
14-38. EPWM1 Initialization for 294
14-39. EPWM2 Initialization for 294
14-40. EPWM1 Initialization for 297
14-41. EPWM2 Initialization for 297
14-42. EPWM1 Initialization for 300
14-43. EPWM2 Initialization for 300
14-44. EPWM3 Initialization for 301
14-45. EPWM1 Initialization for 306
14-46. EPWM2 Initialization for 306
14-47. EPWM3 Initialization for 307
14-48. EPWM1 Initialization for 310

Copyright © 2013–2016, Texas Instruments Incorporated

14-49. EPWM2 Initialization for 310
14-50. Submodule Registers 311
14-51. Time-Base Submodule Registers 311
14-52. Time-Base Control Register (TBCTL) Field Descriptions 312
14-53. Time-Base Status Register (TBSTS) Field Descriptions 313
14-54. Time-Base Phase Register (TBPHS) Field Descriptions 314
14-55. Time-Base Counter Register (TBCNT) Field Descriptions 314
14-56. Time-Base Period Register (TBPRD) Field Descriptions 315
14-57. Counter-Compare Submodule Registers 315
14-58. Counter-Compare Control Register (CMPCTL) Field Descriptions 316
14-59. Counter-Compare A Register (CMPA) Field Descriptions 317
14-60. Counter-Compare B Register (CMPB) Field Descriptions 318
14-61. Action-Qualifier Submodule Registers 318
14-62. Action-Qualifier Output A Control Register (AQCTLA) Field Descriptions 319
14-63. Action-Qualifier Output B Control Register (AQCTLB) Field Descriptions 320
14-64. Action-Qualifier Software Force Register (AQSFRC) Field Descriptions 321
14-65. Action-Qualifier Continuous Software Force Register (AQCSFRC) Field Descriptions 322
14-66. Dead-Band Generator Submodule Registers 322
14-67. Dead-Band Generator Control Register (DBCTL) Field Descriptions 323
14-68. Dead-Band Generator Rising Edge Delay Register (DBRED) Field Descriptions 324
14-69. Dead-Band Generator Falling Edge Delay Register (DBFED) Field Descriptions 324
14-70. PWM-Chopper Control Register (PCCTL) Bit Descriptions 325
14-71. Trip-Zone Submodule Registers 326
14-72. Trip-Zone Submodule Select Register (TZSEL) Field Descriptions 326
14-73. Trip-Zone Control Register (TZCTL) Field Descriptions 327
14-74. Trip-Zone Enable Interrupt Register (TZEINT) Field Descriptions 327
14-75. Trip-Zone Flag Register (TZFLG) Field Descriptions 328
14-76. Trip-Zone Clear Register (TZCLR) Field Descriptions 329
14-77. Trip-Zone Force Register (TZFRC) Field Descriptions 329
14-78. Event-Trigger Submodule Registers 330
14-79. Event-Trigger Selection Register (ETSEL) Field Descriptions 330
14-80. Event-Trigger Prescale Register (ETPS) Field Descriptions 331
14-81. Event-Trigger Flag Register (ETFLG) Field Descriptions 332
14-82. Event-Trigger Clear Register (ETCLR) Field Descriptions 332
14-83. Event-Trigger Force Register (ETFRC) Field Descriptions 333
14-84. High-Resolution PWM Submodule Registers 333
14-85. Time-Base Phase High-Resolution Register (TBPHSHR) Field Descriptions 334
14-86. Counter-Compare A High-Resolution Register (CMPAHR) Field Descriptions 334
14-87. HRPWM Configuration Register (HRCNFG) Field Descriptions 335
15-1. Quadrature Decoder Truth Table 343
15-2. eQEP Registers 358
15-3. eQEP Position Counter Register (QPOSCNT) Field Descriptions 359
15-4. eQEP Position Counter Initialization Register (QPOSINIT) Field Descriptions 359
15-5. eQEP Maximum Position Count Register (QPOSMAX) Field Descriptions 359
15-6. eQEP Position-Compare Register (QPOSCMP) Field Descriptions 360
15-7. eQEP Index Position Latch Register (QPOSILAT) Field Descriptions 360
15-8. eQEP Strobe Position Latch Register (QPOSSLAT) Field Descriptions 360
15-9. eQEP Position Counter Latch Register (QPOSLAT) Field Descriptions 361
15-10. eQEP Unit Timer Register (QUTMR) Field Descriptions 361

Copyright © 2013–2016, Texas Instruments Incorporated

15-11. eQEP Unit Period Register (QUPRD) Field Descriptions 361
15-12. eQEP Watchdog Timer Register (QWDTMR) Field Descriptions 362
15-13. eQEP Watchdog Period Register (QWDPRD) Field Description 362
15-14. eQEP Decoder Control Register (QDECCTL) Field Descriptions 363
15-15. eQEP Control Register (QEPCTL) Field Descriptions 364
15-16. eQEP Capture Control Register (QCAPCTL) Field Descriptions 366
15-17. eQEP Position-Compare Control Register (QPOSCTL) Field Descriptions 367
15-18. eQEP Interrupt Enable Register (QEINT) Field Descriptions 368
15-19. eQEP Interrupt Flag Register (QFLG) Field Descriptions 369
15-20. eQEP Interrupt Clear Register (QCLR) Field Descriptions 370
15-21. eQEP Interrupt Force Register (QFRC) Field Descriptions 372
15-22. eQEP Status Register (QEPSTS) Field Descriptions 373
15-23. eQEP Capture Time Register (QCTMR) Field Descriptions 374
15-24. eQEP Capture Period Register (QCPRD) Field Descriptions 374
15-25. eQEP Capture Timer Latch Register (QCTMRLAT) Field Descriptions 374
15-26. eQEP Capture Period Latch Register (QCPRDLAT) Field Descriptions 375
15-27. eQEP Revision ID Register (REVID) Field Descriptions 375
16-1. EDMA3 Channel Parameter Description 388
16-2. Dummy and Null Transfer Request 391
16-3. Parameter Updates in EDMA3CC (for Non-Null, Non-Dummy PaRAM Set) 392
16-4. Expected Number of Transfers for Non-Null Transfer 400
16-5. EDMA3 DMA Channel to PaRAM Mapping 402
16-6. Shadow Region Registers 404
16-7. Chain Event Triggers 406
16-8. Transfer Complete Code (TCC) to EDMA3CC Interrupt Mapping 407
16-9. Number of Interrupts 408
16-10. Read/Write Command Optimization Rules 421
16-11. EDMA3 Channel Controller (EDMA3CC) Parameter RAM (PaRAM) Entries 440
16-12. Channel Options Parameters (OPT) Field Descriptions 441
16-13. Channel Source Address Parameter (SRC) Field Descriptions 443
16-14. A Count/B Count Parameter (A_B_CNT) Field Descriptions 443
16-15. Channel Destination Address Parameter (DST) Field Descriptions 444
16-16. Source B Index/Destination B Index Parameter (SRC_DST_BIDX) Field Descriptions 444
16-17. Link Address/B Count Reload Parameter (LINK_BCNTRLD) Field Descriptions 445
16-18. Source C Index/Destination C Index Parameter (SRC_DST_CIDX) Field Descriptions 446
16-19. C Count Parameter (CCNT) Field Descriptions 446
16-20. EDMA3 Channel Controller (EDMA3CC) Registers 447
16-21. Revision ID Register (REVID) Field Descriptions 450
16-22. EDMA3CC Configuration Register (CCCFG) Field Descriptions 451
16-23. QDMA Channel *n* Mapping Register (QCHMAP*n*) Field Descriptions 452
16-24. DMA Channel Queue Number Register *n* (DMAQNUM*n*) Field Descriptions 453
16-25. Bits in DMAQNUM*n* 453
16-26. QDMA Channel Queue Number Register (QDMAQNUM) Field Descriptions 454
16-27. Event Missed Register (EMR) Field Descriptions 455
16-28. Event Missed Clear Register (EMCR) Field Descriptions 456
16-29. QDMA Event Missed Register (QEMR) Field Descriptions 457
16-30. QDMA Event Missed Clear Register (QEMCR) Field Descriptions 458
16-31. EDMA3CC Error Register (CCERR) Field Descriptions 459
16-32. EDMA3CC Error Clear Register (CCERRCLR) Field Descriptions 460

Copyright © 2013–2016, Texas Instruments Incorporated

16-33. Error Evaluate Register (EEVAL) Field Descriptions 461
16-34. DMA Region Access Enable Register for Region *m* (DRAE*m*) Field Descriptions 462
16-35. QDMA Region Access Enable for Region *m* (QRAE*m*) Field Descriptions 463
16-36. Event Queue Entry Registers (Q*x*E*y*) Field Descriptions 464
16-37. Queue *n* Status Register (QSTAT*n*) Field Descriptions 465
16-38. Queue Watermark Threshold A Register (QWMTHRA) Field Descriptions 466
16-39. EDMA3CC Status Register (CCSTAT) Field Descriptions 467
16-40. Event Register (ER) Field Descriptions 469
16-41. Event Clear Register (ECR) Field Descriptions 470
16-42. Event Set Register (ESR) Field Descriptions 471
16-43. Chained Event Register (CER) Field Descriptions 472
16-44. Event Enable Register (EER) Field Descriptions 473
16-45. Event Enable Clear Register (EECR) Field Descriptions 474
16-46. Event Enable Set Register (EESR) Field Descriptions 474
16-47. Secondary Event Register (SER) Field Descriptions 475
16-48. Secondary Event Clear Register (SECR) Field Descriptions 475
16-49. Interrupt Enable Register (IER) Field Descriptions 476
16-50. Interrupt Enable Clear Register (IECR) Field Descriptions 477
16-51. Interrupt Enable Set Register (IESR) Field Descriptions 477
16-52. Interrupt Pending Register (IPR) Field Descriptions 478
16-53. Interrupt Clear Register (ICR) Field Descriptions 479
16-54. Interrupt Evaluate Register (IEVAL) Field Descriptions 480
16-55. QDMA Event Register (QER) Field Descriptions 481
16-56. QDMA Event Enable Register (QEER) Field Descriptions 482
16-57. QDMA Event Enable Clear Register (QEECR) Field Descriptions 483
16-58. QDMA Event Enable Set Register (QEESR) Field Descriptions 483
16-59. QDMA Secondary Event Register (QSER) Field Descriptions 484
16-60. QDMA Secondary Event Clear Register (QSECR) Field Descriptions 485
16-61. EDMA3 Transfer Controller (EDMA3TC) Registers 486
16-62. Revision ID Register (REVID) Field Descriptions 487
16-63. EDMA3TC Configuration Register (TCCFG) Field Descriptions 488
16-64. EDMA3TC Channel Status Register (TCSTAT) Field Descriptions 489
16-65. Error Status Register (ERRSTAT) Field Descriptions 490
16-66. Error Enable Register (ERREN) Field Descriptions 491
16-67. Error Clear Register (ERRCLR) Field Descriptions 492
16-68. Error Details Register (ERRDET) Field Descriptions 493
16-69. Error Interrupt Command Register (ERRCMD) Field Descriptions 494
16-70. Read Command Rate Register (RDRATE) Field Descriptions 495
16-71. Source Active Options Register (SAOPT) Field Descriptions 496
16-72. Source Active Source Address Register (SASRC) Field Descriptions 497
16-73. Source Active Count Register (SACNT) Field Descriptions 497
16-74. Source Active Destination Address Register (SADST) Field Descriptions 498
16-75. Source Active B-Index Register (SABIDX) Field Descriptions 498
16-76. Source Active Memory Protection Proxy Register (SAMPPRXY) Field Descriptions 499
16-77. Source Active Count Reload Register (SACNTRLD) Field Descriptions 500
16-78. Source Active Source Address B-Reference Register (SASRCBREF) Field Descriptions 500
16-79. Source Active Destination Address B-Reference Register (SADSTBREF) Field Descriptions 501
16-80. Destination FIFO Set Count Reload Register (DFCNTRLD) Field Descriptions 501
16-81. Destination FIFO Set Source Address B-Reference Register (DFSRCBREF) Field Descriptions 502

Copyright © 2013–2016, Texas Instruments Incorporated

16-82. Destination FIFO Set Destination Address B-Reference Register (DFDSTBREF) Field Descriptions 502
16-83. Destination FIFO Options Register *n* (DFOPT*n*) Field Descriptions 503
16-84. Destination FIFO Source Address Register *n* (DFSRC*n*) Field Descriptions 504
16-85. Destination FIFO Count Register *n* (DFCNT*n*) Field Descriptions 504
16-86. Destination FIFO Destination Address Register *n* (DFDST*n*) Field Descriptions 505
16-87. Destination FIFO B-Index Register *n* (DFBIDX*n*) Field Descriptions 505
16-88. Destination FIFO Memory Protection Proxy Register *n* (DFMPPRXY*n*) Field Descriptions 506
16-89. Debug List 507
17-1. GPIO Register Bits and Banks Associated With GPIO Signals 513
17-2. GPIO Registers 520
17-3. Revision ID Register (REVID) Field Descriptions 521
17-4. GPIO Interrupt Per-Bank Enable Register (BINTEN) Field Descriptions 522
17-5. GPIO Direction Register (DIR*n*) Field Descriptions 524
17-6. GPIO Output Data Register (OUT_DATA*n*) Field Descriptions 526
17-7. GPIO Set Data Register (SET_DATA*n*) Field Descriptions 528
17-8. GPIO Clear Data Register (CLR_DATA*n*) Field Descriptions 530
17-9. GPIO Input Data Register (IN_DATA*n*) Field Descriptions 532
17-10. GPIO Set Rising Edge Trigger Interrupt Register (SET_RIS_TRIG*n*) Field Descriptions 534
17-11. GPIO Clear Rising Edge Interrupt Register (CLR_RIS_TRIG*n*) Field Descriptions 536
17-12. GPIO Set Falling Edge Trigger Interrupt Register (SET_FAL_TRIG*n*) Field Descriptions 538
17-13. GPIO Clear Falling Edge Interrupt Register (CLR_FAL_TRIG*n*) Field Descriptions 540
17-14. GPIO Interrupt Status Register (INTSTAT*n*) Field Descriptions 542
18-1. Operating Modes of the I2C Peripheral 552
18-2. Ways to Generate a NACK Bit 553
18-3. Descriptions of the I2C Interrupt Events 557
18-4. Inter-Integrated Circuit (I2C) Registers 558
18-5. I2C Own Address Register (ICOAR) Field Descriptions 559
18-6. I2C Interrupt Mask Register (ICIMR) Field Descriptions 560
18-7. I2C Interrupt Status Register (ICSTR) Field Descriptions 561
18-8. I2C Clock Low-Time Divider Register (ICCLKL) Field Descriptions 564
18-9. I2C Clock High-Time Divider Register (ICCLKH) Field Descriptions 564
18-10. I2C Data Count Register (ICCNT) Field Descriptions 565
18-11. I2C Data Receive Register (ICDRR) Field Descriptions 566
18-12. I2C Slave Address Register (ICSAR) Field Descriptions 567
18-13. I2C Data Transmit Register (ICDXR) Field Descriptions 568
18-14. I2C Mode Register (ICMDR) Field Descriptions 569
18-15. Master-Transmitter/Receiver Bus Activity Defined by RM, STT, and STP Bits 571
18-16. How the MST and FDF Bits Affect the Role of TRX Bit 571
18-17. I2C Interrupt Vector Register (ICIVR) Field Descriptions 573
18-18. I2C Extended Mode Register (ICEMDR) Field Descriptions 574
18-19. I2C Prescaler Register (ICPSC) Field Descriptions 575
18-20. I2C Revision Identification Register 1 (REVID1) Field Descriptions 576
18-21. I2C Revision Identification Register 2 (REVID2) Field Descriptions 576
18-22. I2C DMA Control Register (ICDMAC) Field Descriptions 577
18-23. I2C Pin Function Register (ICPFUNC) Field Descriptions 578
18-24. I2C Pin Direction Register (ICPDIR) Field Descriptions 579
18-25. I2C Pin Data In Register (ICPDIN) Field Descriptions 580
18-26. I2C Pin Data Out Register (ICPDOUT) Field Descriptions 581
18-27. I2C Pin Data Set Register (ICPDSET) Field Descriptions 582

Copyright © 2013–2016, Texas Instruments Incorporated

18-28. I2C Pin Data Clear Register (ICPDCLR) Field Descriptions 583
19-1. Biphase-Mark Encoder 592
19-2. Preamble Codes 593
19-3. Channel Status and User Data for Each DIT Block 619
19-4. Transmit Bitstream Data Alignment 626
19-5. Receive Bitstream Data Alignment 628
19-6. EDMA Events - McASP 638
19-7. McASP Registers Accessed by CPU/EDMA Through Peripheral Configuration Port 639
19-8. McASP Registers Accessed by CPU/EDMA Through DMA Port 642
19-9. McASP AFIFO Registers Accessed Through Peripheral Configuration Port 642
19-10. Revision Identification Register (REV) Field Descriptions 643
19-11. Pin Function Register (PFUNC) Field Descriptions 645
19-12. Pin Direction Register (PDIR) Field Descriptions 647
19-13. Pin Data Output Register (PDOUT) Field Descriptions 649
19-14. Pin Data Input Register (PDIN) Field Descriptions 651
19-15. Pin Data Set Register (PDSET) Field Descriptions 653
19-16. Pin Data Clear Register (PDCLR) Field Descriptions 655
19-17. Global Control Register (GBLCTL) Field Descriptions 656
19-18. Audio Mute Control Register (AMUTE) Field Descriptions 658
19-19. Digital Loopback Control Register (DLBCTL) Field Descriptions 660
19-20. Digital Mode Control Register (DITCTL) Field Descriptions 661
19-21. Receiver Global Control Register (RGBLCTL) Field Descriptions 662
19-22. Receive Format Unit Bit Mask Register (RMASK) Field Descriptions 663
19-23. Receive Bit Stream Format Register (RFMT) Field Descriptions 664
19-24. Receive Frame Sync Control Register (AFSRCTL) Field Descriptions 666
19-25. Receive Clock Control Register (ACLKRCTL) Field Descriptions 667
19-26. Receive High-Frequency Clock Control Register (AHCLKRCTL) Field Descriptions 668
19-27. Receive TDM Time Slot Register (RTDM) Field Descriptions 669
19-28. Receiver Interrupt Control Register (RINTCTL) Field Descriptions 670
19-29. Receiver Status Register (RSTAT) Field Descriptions 671
19-30. Current Receive TDM Time Slot Registers (RSLOT) Field Descriptions 672
19-31. Receive Clock Check Control Register (RCLKCHK) Field Descriptions 673
19-32. Receiver DMA Event Control Register (REVTCTL) Field Descriptions 674
19-33. Transmitter Global Control Register (XGBLCTL) Field Descriptions 675
19-34. Transmit Format Unit Bit Mask Register (XMASK) Field Descriptions 676
19-35. Transmit Bit Stream Format Register (XFMT) Field Descriptions 677
19-36. Transmit Frame Sync Control Register (AFSXCTL) Field Descriptions 679
19-37. Transmit Clock Control Register (ACLKXCTL) Field Descriptions 680
19-38. Transmit High-Frequency Clock Control Register (AHCLKXCTL) Field Descriptions 681
19-39. Transmit TDM Time Slot Register (XTDM) Field Descriptions 682
19-40. Transmitter Interrupt Control Register (XINTCTL) Field Descriptions 683
19-41. Transmitter Status Register (XSTAT) Field Descriptions 684
19-42. Current Transmit TDM Time Slot Register (XSLOT) Field Descriptions 685
19-43. Transmit Clock Check Control Register (XCLKCHK) Field Descriptions 686
19-44. Transmitter DMA Event Control Register (XEVTCTL) Field Descriptions 687
19-45. Serializer Control Registers (SRCTL*n*) Field Descriptions 688
19-46. AFIFO Revision Identification Register (AFIFOREV) Field Descriptions 692
19-47. Write FIFO Control Register (WFIFOCTL) Field Descriptions 693
19-48. Write FIFO Status Register (WFIFOSTS) Field Descriptions 694

Copyright © 2013–2016, Texas Instruments Incorporated

19-49. Read FIFO Control Register (RFIFOCTL) Field Descriptions 695
19-50. Read FIFO Status Register (RFIFOSTS) Field Descriptions 696
20-1. SPI Pins 700
20-2. SPI Registers 701
20-3. SPI Register Settings Defining Master Modes 702
20-4. Allowed SPI Register Settings in Master Modes 702
20-5. SPI Register Settings Defining Slave Modes 704
20-6. Allowed SPI Register Settings in Slave Modes 704
20-7. Clocking Modes 713
20-8. SPI Registers 726
20-9. SPI Global Control Register 0 (SPIGCR0) Field Descriptions 726
20-10. SPI Global Control Register 1 (SPIGCR1) Field Descriptions 727
20-11. SPI Interrupt Register (SPIINT0) Field Descriptions 729
20-12. SPI Interrupt Level Register (SPILVL) Field Descriptions 731
20-13. SPI Flag Register (SPIFLG) Field Descriptions 732
20-14. SPI Pin Control Register 0 (SPIPC0) Field Descriptions 734
20-15. SPI Pin Control Register 1 (SPIPC1) Field Descriptions 735
20-16. SPI Pin Control Register 2 (SPIPC2) Field Descriptions 736
20-17. SPI Pin Control Register 3 (SPIPC3) Field Descriptions 737
20-18. SPI Pin Control Register 4 (SPIPC4) Field Descriptions 738
20-19. SPI Pin Control Register 5 (SPIPC5) Field Descriptions 739
20-20. SPI Data Register 0 (SPIDAT0) Field Descriptions 740
20-21. SPI Data Register 1 (SPIDAT1) Field Descriptions 741
20-22. SPI Buffer Register (SPIBUF) Field Descriptions 742
20-23. SPI Emulation Register (SPIEMU) Field Descriptions 744
20-24. SPI Delay Register (SPIDELAY) Field Descriptions 745
20-25. SPI Default Chip Select Register (SPIDEF) Field Descriptions 748
20-26. SPI Data Format Register (SPIFMT*n*) Field Descriptions 749
20-27. SPI Interrupt Vector Register 1 (INTVEC1) Field Descriptions 751
21-1. Timer Clock Source Selection 755
21-2. 64-Bit Timer Configurations 757
21-3. 32-Bit Timer Chained Mode Configurations 760
21-4. 32-Bit Timer Unchained Mode Configurations 763
21-5. Counter and Period Registers Used in GP Timer Modes 765
21-6. TSTAT Parameters in Pulse and Clock Modes 769
21-7. Timer Emulation Modes Selection 771
21-8. Timer Registers 771
21-9. Revision ID Register (REVID) Field Descriptions 773
21-10. Emulation Management Register (EMUMGT) Field Descriptions 773
21-11. GPIO Interrupt Control and Enable Register (GPINTGPEN) Field Descriptions 774
21-12. GPIO Data and Direction Register (GPDATGPDIR) Field Descriptions 775
21-13. Timer Counter Register 12 (TIM12) Field Descriptions 776
21-14. Timer Counter Register 34 (TIM34) Field Descriptions 776
21-15. Timer Period Register (PRD12) Field Descriptions 777
21-16. Timer Period Register (PRD34) Field Descriptions 777
21-17. Timer Control Register (TCR) Field Descriptions 778
21-18. Timer Global Control Register (TGCR) Field Descriptions 780
21-19. Watchdog Timer Control Register (WDTCR) Field Descriptions 781
21-20. Timer Reload Register 12 (REL12) Field Descriptions 782

Copyright © 2013–2016, Texas Instruments Incorporated

21-21. Timer Reload Register 34 (REL34) Field Descriptions 782
21-22. Timer Capture Register 12 (CAP12) Field Descriptions 783
21-23. Timer Capture Register 34 (CAP34) Field Descriptions 783
21-24. Timer Interrupt Control and Status Register (INTCTLSTAT) Field Descriptions 784
21-25. Timer Compare Register (CMP*n*) Field Descriptions 785
22-1. Baud Rate Examples for 150-MHZ UART Input Clock and 16× Over-sampling Mode 790
22-2. Baud Rate Examples for 150-MHZ UART Input Clock and 13× Over-sampling Mode 790
22-3. UART Signal Descriptions 791
22-4. Character Time for Word Lengths 794
22-5. UART Interrupt Requests Descriptions 798
22-6. UART Registers 800
22-7. Receiver Buffer Register (RBR) Field Descriptions 801
22-8. Transmitter Holding Register (THR) Field Descriptions 802
22-9. Interrupt Enable Register (IER) Field Descriptions 803
22-10. Interrupt Identification Register (IIR) Field Descriptions 804
22-11. Interrupt Identification and Interrupt Clearing Information 805
22-12. FIFO Control Register (FCR) Field Descriptions 806
22-13. Line Control Register (LCR) Field Descriptions 807
22-14. Relationship Between ST, EPS, and PEN Bits in LCR 808
22-15. Number of STOP Bits Generated 808
22-16. Modem Control Register (MCR) Field Descriptions 809
22-17. Line Status Register (LSR) Field Descriptions 810
22-18. Modem Status Register (MSR) Field Descriptions 813
22-19. Scratch Pad Register (MSR) Field Descriptions 814
22-20. Divisor LSB Latch (DLL) Field Descriptions 815
22-21. Divisor MSB Latch (DLH) Field Descriptions 815
22-22. Revision Identification Register 1 (REVID1) Field Descriptions 816
22-23. Revision Identification Register 2 (REVID2) Field Descriptions 816
22-24. Power and Emulation Management Register (PWREMU_MGMT) Field Descriptions 817
22-25. Mode Definition Register (MDR) Field Descriptions 818

Copyright © 2013–2016, Texas Instruments Incorporated

Read This First

About This Manual

This Technical Reference Manual (TRM) describes the System-on-Chip (SoC) and each peripheral in the device. The SoC consists of the following primary components

- DSP subsystem and associated memories
- A set of I/O peripherals

Notational Conventions

This document uses the following conventions.

- Hexadecimal numbers are shown with the suffix h. For example, the following number is 40 hexadecimal (decimal 64): 40h.
- Registers in this document are shown in figures and described in tables.
  - Each register figure shows a rectangle divided into fields that represent the fields of the register. Each field is labeled with its bit name, its beginning and ending bit numbers above, and its read/write properties below. A legend explains the notation used for the properties.
  - Reserved bits in a register figure designate a bit that is used for future device expansion.

Related Documentation From Texas Instruments

Copies of these documents are available on the Internet at www.ti.com. *Tip:* Enter the literature number in the search box provided at www.ti.com.

The current documentation that describes related peripherals and other technical collateral, is available in the C6000 DSP product folder at: www.ti.com/c6000.

SPRUFK5— ***TMS320C674x DSP Megamodule Reference Guide.*** Describes the TMS320C674x digital signal processor (DSP) megamodule. Included is a discussion on the internal direct memory access (IDMA) controller, the interrupt controller, the power-down controller, memory protection, bandwidth management, and the memory and cache.

SPRUFE8— ***TMS320C674x DSP CPU and Instruction Set Reference Guide.*** Describes the CPU architecture, pipeline, instruction set, and interrupts for the TMS320C674x digital signal processors (DSPs). The C674x DSP is an enhancement of the C64x+ and C67x+ DSPs with added functionality and an expanded instruction set.

SPRUG82— ***TMS320C674x DSP Cache User's Guide.*** Explains the fundamentals of memory caches and describes how the two-level cache-based internal memory architecture in the TMS320C674x digital signal processor (DSP) can be efficiently used in DSP applications. Shows how to maintain coherence with external memory, how to use DMA to reduce memory latencies, and how to optimize your code to improve cache efficiency. The internal memory architecture in the C674x DSP is organized in a two-level hierarchy consisting of a dedicated program cache (L1P) and a dedicated data cache (L1D) on the first level. Accesses by the CPU to the these first level caches can complete without CPU pipeline stalls. If the data requested by the CPU is not contained in cache, it is fetched from the next lower memory level, L2 or external memory.

Code Composer Studio is a trademark of Texas Instruments.

Copyright © 2013–2016, Texas Instruments Incorporated

Overview

Topic Page

1.1 Introduction 40

Copyright © 2013–2016, Texas Instruments Incorporated

1.1 Introduction

The TMS320DA804/DA803/DA802 DSP contains a powerful DSP to efficiently handle communication and audio processing tasks. The DA804/DA803/DA802 DSP consists of the following primary components:

- DSP subsystem and associated memories
- A set of I/O peripherals
- A powerful DMA subsystem

Block Diagram

A block diagram for the TMS320DA804/DA803/DA802 DSP is shown in Figure 1-1.

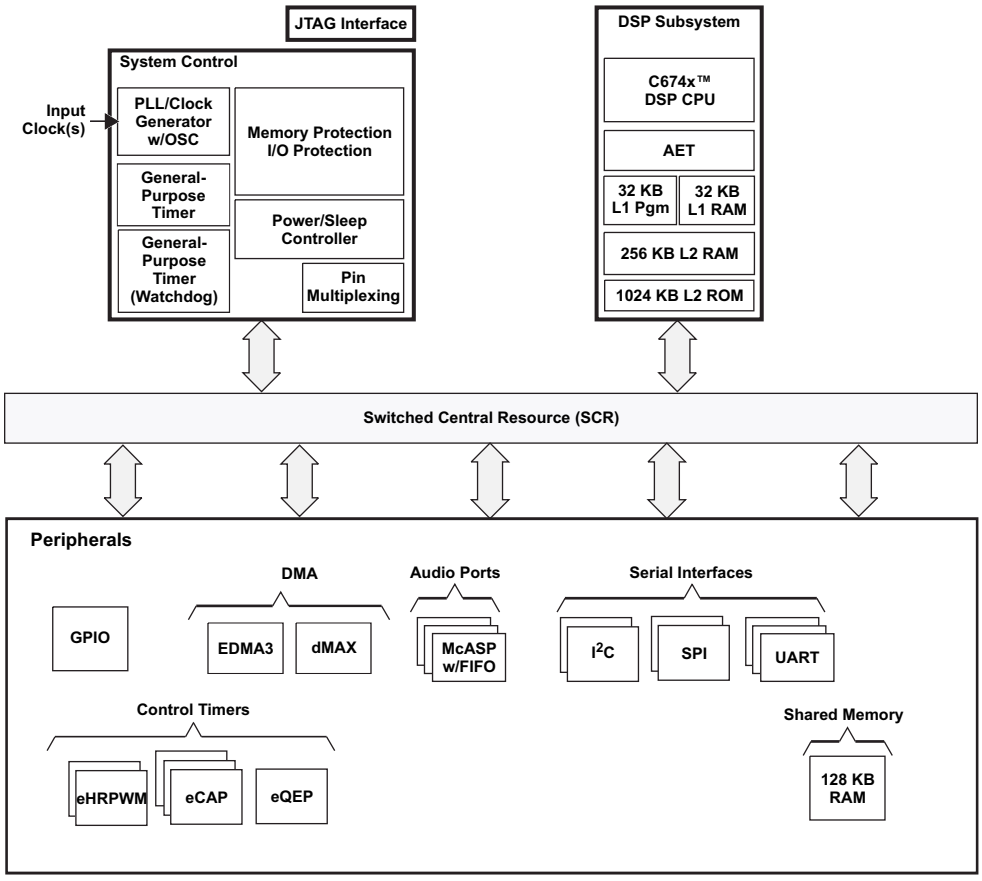
DSP Subsystem

The DSP subsystem (DSPSS) includes TI's standard TMS320C674x megamodule and several blocks of internal memory (L1P, L1D, and L2). The *DSP Subsystem* chapter describes the DSPSS components.

DMA Subsystem

The DMA subsystem includes two instances of the enhanced DMA controller (EDMA3). For more information, see the *Enhanced Direct Memory Access (EDMA3) Controller* chapter.

Figure 1-1. TMS320DA804/DA803/DA802 DSP Block Diagram



Notes: Not all peripherals are available at the same time due to multiplexing.

Copyright © 2013–2016, Texas Instruments Incorporated

DSP Subsystem

Topic **Page**

2.1 Introduction 42
2.2 TMS320C674x Megamodule 43
2.3 Memory Map 47
2.4 Advanced Event Triggering (AET) 47

Copyright © 2013–2016, Texas Instruments Incorporated

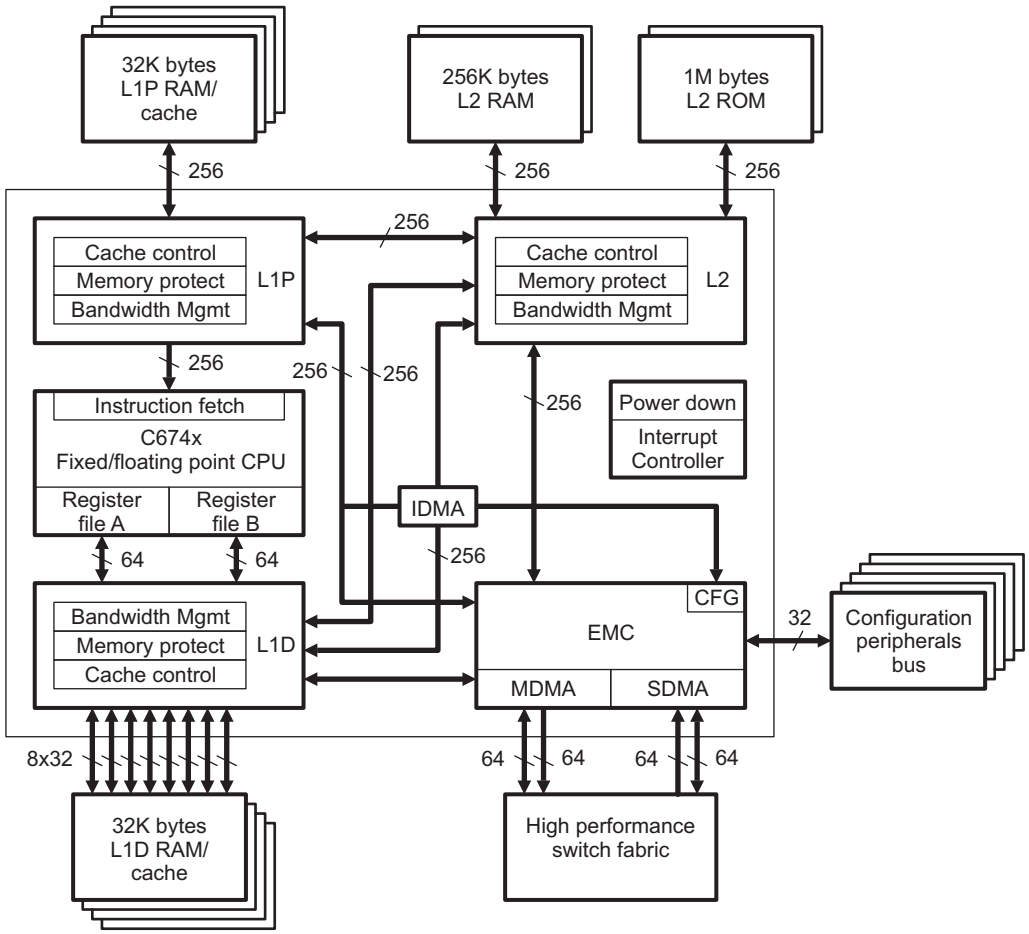
2.1 Introduction

The DSP subsystem (Figure 2-1) includes TI's standard TMS320C674x megamodule and several blocks of internal memory (L1P, L1D, and L2). This document provides an overview of the DSP subsystem and the following considerations associated with it:

- Memory mapping
- Interrupts
- Power management

For more information, see the *TMS320C674x DSP Megamodule Reference Guide* (SPRUFK5), the *TMS320C674x DSP CPU and Instruction Set Reference Guide* (SPRUFE8), and the *TMS320C674x DSP Cache User's Guide* (SPRUG82).

Figure 2-1. TMS320C674x Megamodule Block Diagram



Copyright © 2013–2016, Texas Instruments Incorporated

2.2 TMS320C674x Megamodule

The C674x megamodule (Figure 2-1) consists of the following components:

- TMS320C674x CPU
- Internal memory controllers:
  - Level 1 program memory controller (PMC)
  - Level 1 data memory controller (DMC)
  - Level 2 unified memory controller (UMC)
  - Extended memory controller (EMC)
  - Internal direct memory access (IDMA) controller
- Internal peripherals:
  - Interrupt controller (INTC)
  - Power-down controller (PDC)
  - Bandwidth manager (BWM)
- Advanced event triggering (AET)

2.2.1 Internal Memory Controllers

The C674x megamodule implements a two-level internal cache-based memory architecture with external memory support. Level 1 memory (L1) is split into separate program memory (L1P memory) and data memory (L1D memory). L1 memory is accessible to the CPU without stalls. Level 2 memory (L2) can also be split into L2 RAM (normal addressable on-chip memory) and L2 cache for caching external memory locations. The internal direct memory access controller (IDMA) manages DMA among the L1P, L1D, and L2 memories.

For more information about each of these controllers, see the *TMS320C674x DSP Megamodule Reference Guide* (SPRUFK5).

2.2.2 Internal Peripherals

The C674x megamodule includes the following internal peripherals:

- DSP interrupt controller (INTC)
- DSP power-down controller (PDC)
- Bandwidth manager (BWM)
- Internal DMA (IDMA) controller

This section briefly describes the INTC, PDC, BWM, and IDMA controller. For more information on these internal peripherals, see the *TMS320C674x DSP Megamodule Reference Guide* (SPRUFK5).

2.2.2.1 Interrupt Controller (INTC)

The C674x megamodule includes an interrupt controller (INTC) to manage CPU interrupts. The INTC maps DSP device events to 12 CPU interrupts. All DSP device events are listed in Table 2-1. The INTC is fully described in the *TMS320C674x DSP Megamodule Reference Guide* (SPRUFK5).

Table 2-1. DSP Interrupt Map

Event	Interrupt Name	Source
0	EVT0	C674x Interrupt Control 0
1	EVT1	C674x Interrupt Control 1
2	EVT2	C674x Interrupt Control 2
3	EVT3	C674x Interrupt Control 3
4	T64P0_TINT12	Timer64P0 - TINT12

Copyright © 2013–2016, Texas Instruments Incorporated

Table 2-1. DSP Interrupt Map (continued)

Event	Interrupt Name	Source
5	SYSCFG_CHIPINT2	SYSCFG CHIPSIG Register
6	DMAX_EVTOUT0	dMAX Interrupt
7	EHRPWM0	HiResTimer/PWM0 Interrupt
8	TPCC0_INT1	TPCC0 Region 1 Interrupt
9	EMU-DTDMA	C674x-ECM
10	EHRPWM0TZ	HiResTimer/PWM0 Trip Zone Interrupt
11	EMU-RTDXRX	C674x-RTDX
12	EMU-RTDXTX	C674x-RTDX
13	IDMAINT0	C674x-EMC
14	IDMAINT1	C674x-EMC
15-16	—	Reserved
17	DMAX_EVTOUT1	dMAX Interrupt
18-21	—	Reserved
22	DMAX_EVTOUT2	dMAX Interrupt
23-34	—	Reserved
35	DMAX_EVTOUT3	dMAX Interrupt
36	IIC0_INT	I2C0
37	SPI0_INT	SPI0
38	UART0_INT	UART0
39	DMAX_EVTOUT5	dMAX Interrupt
40	—	Reserved
41	GPIO_B1INT	GPIO Bank 1 Interrupt
42	IIC1_INT	I2C1
43	SPI1_INT	SPI1
44	DMAX_EVTOUT6	dMAX Interrupt
45	ECAP0	ECAP0
46	UART_INT1	UART1
47	ECAP1	ECAP1
48	—	Reserved
49	GPIO_B2INT	GPIO Bank 2 Interrupt
50	DMAX_EVTOUT7	dMAX Interrupt
51	ECAP2	ECAP2
52	GPIO_B3INT	GPIO Bank 3 Interrupt
53	—	Reserved
54	GPIO_B4INT	GPIO Bank 4 Interrupt
55	—	Reserved
56	EDMA3_CC0_ERRINT	EDMA3 Channel Controller 0
57	EDMA3_TC0_ERRINT	EDMA3 Transfer Controller 0
58	EDMA3_TC1_ERRINT	EDMA3 Transfer Controller 1
59	GPIO_B5INT	GPIO Bank 5 Interrupt
60	—	Reserved
61	MCASP_INT	McASP0,1,2 Combined RX/TX Interrupts
62	GPIO_B6INT	GPIO Bank 6 Interrupt
63	—	Reserved
64	T64P0_TINT34	Timer64P0 Interrupt 34
65	GPIO_B0INT	GPIO Bank 0 Interrupt
66	DMAX_EVTOUT4	dMAX Interrupt

Copyright © 2013–2016, Texas Instruments Incorporated

Table 2-1. DSP Interrupt Map (continued)

Event	Interrupt Name	Source
67	SYSCFG_CHIPINT3	SYSCFG CHIPSIG Register
68	EQEP0	EQEP0
69	UART2_INT	UART2
70	PSC0_ALLINT	PSC0
71	PSC1_ALLINT	PSC1
72	GPIO_B7INT	GPIO Bank 7 Interrupt
73	—	Reserved
74	MPU_BOOTCFG_ERR	MPU Shared Interrupt
75-77	—	Reserved
78	T64P0_CMPINT0	Timer64P0 - Compare 0
79	T64P0_CMPINT1	Timer64P0 - Compare 1
80	T64P0_CMPINT2	Timer64P0 - Compare 2
81	T64P0_CMPINT3	Timer64P0 - Compare 3
82	T64P0_CMPINT4	Timer64P0 - Compare 4
83	T64P0_CMPINT5	Timer64P0 - Compare 5
84	T64P0_CMPINT6	Timer64P0 - Compare 6
85	T64P0_CMPINT7	Timer64P0 - Compare 7
86-95	—	Reserved
96	INTERR	C674x-Interrupt Control
97	EMC_IDMAERR	C674x-EMC
98-112	—	Reserved
113	PMC_ED	C674x-PMC
114-115	—	Reserved
116	UMC_ED1	C674x-UMC
117	UMC_ED2	C674x-UMC
118	PDC_INT	C674x-PDC
119	SYS_CMPA	C674x-SYS
120	PMC_CMPA	C674x-PMC
121	PMC_CMPA	C674x-PMC
122	DMC_CMPA	C674x-DMC
123	DMC_CMPA	C674x-DMC
124	UMC_CMPA	C674x-UMC
125	UMC_CMPA	C674x-UMC
126	EMC_CMPA	C674x-EMC
127	EMC_BUSERR	C674x-EMC

2.2.2.1.1 Interrupt Controller Registers

For more information on the DSP interrupt controller (INTC) registers, see the *TMS320C674x DSP Megamodule Reference Guide* (SPRUFK5).

2.2.2.1.2 NMI Interrupt

In addition to the interrupts listed in Table 2-1, the DSP also supports a special interrupt that behaves more like an exception, non-maskable interrupt (NMI). The NMI interrupt is controlled by two registers in the System Configuration Module, the chip signal register (CHIPSIG) and the chip signal clear register (CHIPSIG_CLR).

Copyright © 2013–2016, Texas Instruments Incorporated

The NMI interrupt is asserted by writing a 1 to the CHIPSIG4 bit in CHIPSIG. The NMI interrupt is cleared by writing a 1 to the CHIPSIG4 bit in CHIPSIG_CLR. For more information on the System Configuration Module, CHIPSIG, and CHIPSIG_CLR, see the *System Configuration (SYSCFG) Module* chapter.

2.2.2.2 Power-Down Controller (PDC)

The C674x megamodule includes a power-down controller (PDC). The PDC can power-down all of the following components of the C674x megamodule and internal memories of the DSP subsystem:

- C674x CPU
- Level 1 program memory controller (PMC)
- Level 1 data memory controller (DMC)
- Level 2 unified memory controller (UMC)
- Extended memory controller (EMC)
- Internal Direct Memory Access controller (IDMA)
- L1P memory
- L1D memory
- L2 memory

This device supports the static power-down feature from the C674x megamodule. The *TMS320C674x DSP Megamodule Reference Guide* (SPRUFK5) describes the power-down control in more detail.

- Static power-down: The PDC initiates power-down (clock gating) of the entire C674x megamodule and all internal memories immediately upon command from software.

Static power-down (clock gating) affects all components of the C674x megamodule and all internal memories. Software can initiate static power-down by way of a register bit in the power-down controller command register (PDCCMD) of the PDC. For more information on the PDC, see the *TMS320C674x DSP Megamodule Reference Guide* (SPRUFK5).

2.2.2.3 Bandwidth Manager (BWM)

The bandwidth manager (BWM) provides a programmable interface for optimizing bandwidth among the requesters for resources, which include the following:

- EDMA3-initiated DMA transfers (and resulting coherency operations)
- DSP subsystem IDMA-initiated transfers (and resulting coherency operations)
- Programmable cache coherency operations
  - Block based coherency operations
  - Global coherency operations
- CPU direct-initiated transfers
  - Data access (load/store)
  - Program access

The resources include the following:

- L1P memory
- L1D memory
- L2 memory
- Resources outside of the C674x megamodule: external memory, on-chip peripherals, registers

Since any given requestor could potentially block a resource for extended periods of time, the bandwidth manager is implemented to assure fairness for all requesters.

Copyright © 2013–2016, Texas Instruments Incorporated

The bandwidth manager implements a weighted-priority-driven bandwidth allocation. Each requestor (EDMA, IDMA, CPU, etc.) is assigned a priority level on a per-transfer basis. The programmable priority level has a single meaning throughout the system. There are a total of nine priority levels, where priority zero is the highest priority and priority eight is the lowest priority. When requests for a single resource contend, access is granted to the highest-priority requestor. When the contention occurs for multiple successive cycles, a contention counter assures that the lower-priority requestor gets access to the resource every 1 out of *n* arbitration cycles, where *n* is programmable. A priority level of -1 represents a transfer whose priority has been increased due to expiration of the contention counter or a transfer that is fixed as the highest-priority transfer to a given resource.

2.2.2.4 Internal DMA (IDMA) Controller

The IDMA controller performs fast block transfers between any two memory locations local to the C674x megamodule. Local memory locations are defined as those in Level 1 program (L1P), Level 1 data (L1D), and Level 2 (L2) memories, or in the external peripheral configuration (CFG) memory. The IDMA cannot transfer data to or from the internal DSP memory-mapped register space. The IDMA is fully described in the *TMS320C674x DSP Megamodule Reference Guide* (SPRUFK5).

2.3 Memory Map

Refer to your device-specific data manual for memory-map information.

2.3.1 DSP Internal Memory

See the *System Memory* chapter for a description of the DSP internal memory.

2.3.2 External Memory

See the *System Interconnect* chapter and the *System Memory* chapter for a description of the additional system memory and peripherals that the DSP has access to.

2.4 Advanced Event Triggering (AET)

The C674x megamodule supports advanced event triggering (AET). This capability can be used to debug complex problems as well as understand performance characteristics of user applications. AET provides the following capabilities:

- Hardware Program Breakpoints: specify addresses or address ranges that can generate events such as halting the processor or triggering the trace capture.
- Data Watchpoints: specify data variable addresses, address ranges, or data values that can generate events such as halting the processor or triggering the trace capture.
- Counters: count the occurrence of an event or cycles for performance monitoring.
- State Sequencing: allows combinations of hardware program breakpoints and data watchpoints to precisely generate events for complex sequences.

Copyright © 2013–2016, Texas Instruments Incorporated

System Interconnect

Topic **Page**

3.1 Introduction **49**
3.2 System Interconnect Block Diagram **50**

Copyright © 2013–2016, Texas Instruments Incorporated

3.1 Introduction

The DSP, the EDMA3 transfer controllers, and the device peripherals are interconnected through a switch fabric architecture (see Section 3.2). The switch fabric is composed of multiple switched central resources (SCRs) and multiple bridges. The SCRs establish low-latency connectivity between master peripherals and slave peripherals.

Additionally, the SCRs provide priority-based arbitration and facilitate concurrent data movement between master and slave peripherals. Bridges are mainly used to perform bus-width conversion as well as bus operating frequency conversion.

The DSP, the EDMA3 transfer controllers, and the various device peripherals can be classified into two categories: master peripherals and slave peripherals.

Master peripherals are typically capable of initiating read and write transfers in the system and do not rely on the EDMA3 or on a CPU to perform transfers to and from them. The system master peripherals include the DSP and the EDMA3 transfer controllers. Not all master peripherals may connect to all slave peripherals. The supported connections are designated by an X in Table 3-1.

Table 3-1. TMS320DA804/DA803/DA802 DSP System Interconnect Matrix

Masters		Slaves			
Master	**Default Priority**	**DSP SDMA**	**128 kB RAM**	**EDMA3TC Group(1)**	**Peripheral Group(2)**
EDMA3CC0	0			X	
EDMA3TC0	0	X	X	X	X
EDMA3TC1	0	X	X	X	X
dMAX0	0	X	X	X	X
dMAX1	0	X	X	X	X
DSP CFG	2			X	X
DSP MDMA	2		X		

(1) EDMA3TC group: EDMA3TC0, EDMA3TC1

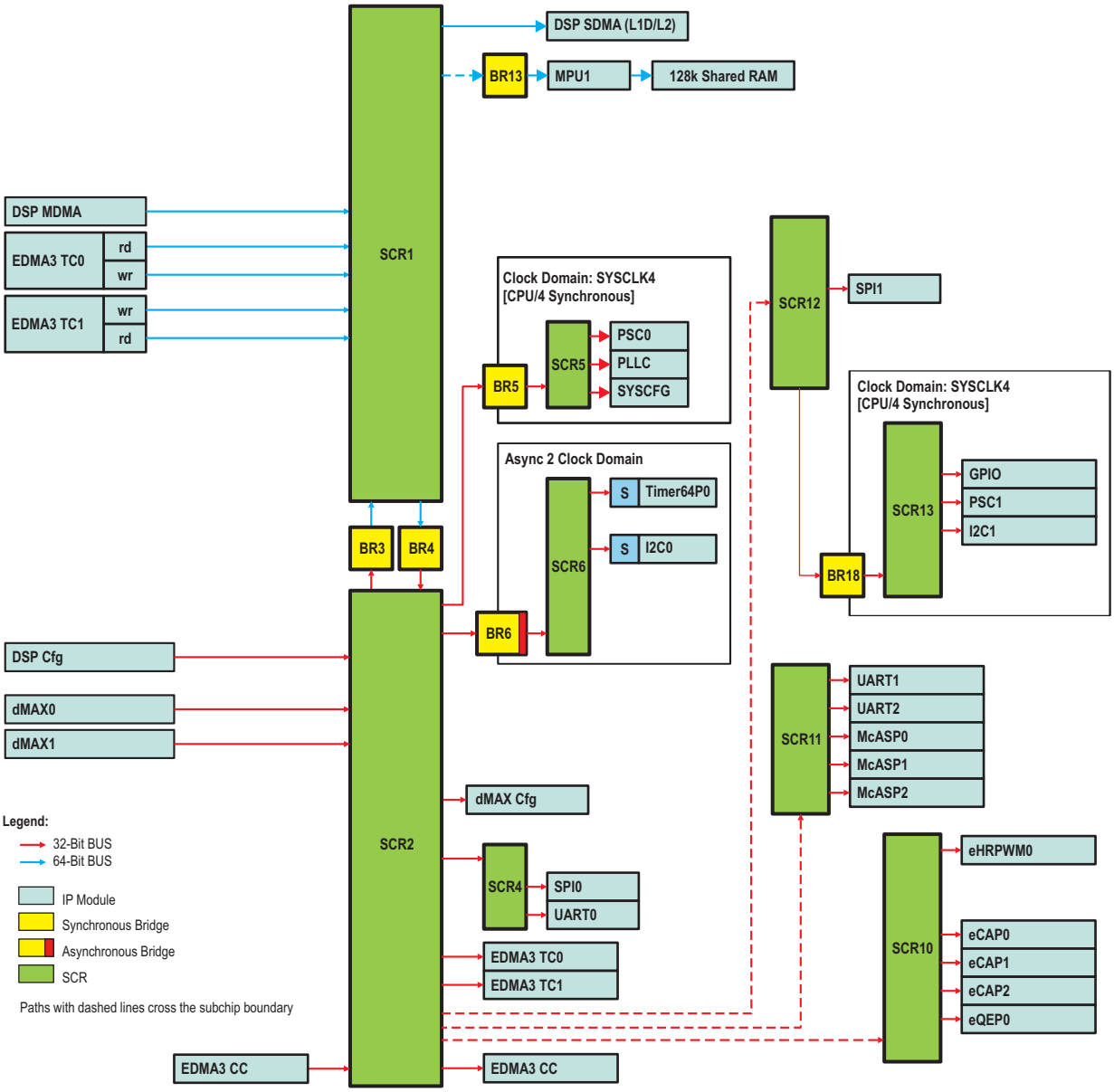
(2) Peripheral group: SYSCFG, eCAP0, eCAP1, eCAP2, eHRPWM0, GPIO, I2C0, I2C1, McASP0, McASP1, McASP2, PLLC, dMAX RAM0, dMAX RAM1, dMAX Config, PSC0, PSC1, SPI0, SPI1, TIMER64P0, EDMA3CC0, UART0, UART1, UART2.

Copyright © 2013–2016, Texas Instruments Incorporated

3.2 System Interconnect Block Diagram

Figure 3-1 shows a system interconnect block diagram.

Figure 3-1. System Interconnect Block Diagram



Copyright © 2013–2016, Texas Instruments Incorporated

System Memory

Topic **Page**

4.1 Introduction **52**
4.2 DSP Memories **52**
4.3 Peripherals **52**

Copyright © 2013–2016, Texas Instruments Incorporated

4.1 Introduction

This device has multiple on-chip/off-chip memories and several external device interfaces associated with the DSP and various subsystems. To help simplify software development, a unified memory-map is used wherever possible to maintain a consistent view of device resources across all masters (CPU and master peripherals).

For details on the memory addresses, actual memory supported and accessibility by various bus masters, see the detailed memory-map information in the device-specific data manual.

4.2 DSP Memories

The DSP internal memories are accessible by the and other master peripherals (as dictated by the connectivity matrix) via the system interconnect through the DSP SDMA port. The accesses by the DSP to its internal memory are internal to the DSP subsystem and do not go out on the system interconnect.

The DSP internal memory consists of L1P, L1D, and L2. The DSP internal memory configuration is:

- L1P memory includes 32 kB of RAM. The DSP program memory controller (PMC) allows you to configure part or all of the L1P RAM as normal program RAM or as cache. You can configure cache sizes of 0 kB, 4 kB, 8 kB, 16 kB, or 32 kB of the 32 kB of RAM. The default configuration is 32 kB cache.
- L1D memory includes 32 kB of RAM. The DSP data memory controller (DMC) allows you to configure part of the L1D RAM as normal data RAM or as cache. You can configure cache sizes of 0 kB, 4 kB, 8 kB, 16 kB, or 32 kB of the 32 kB of RAM. The default configuration is 32 kB cache.
- L2 memory includes 256 kB of RAM. The DSP unified memory controller (UMC) allows you to configure part or all of the L2 RAM as normal RAM or as cache. You can configure cache sizes of 0 kB, 4 kB, 8 kB, 16 kB, 32 kB, 64 kB, 128 kB, or 256 kB of the 256 kB of RAM. The default configuration is 256 kB normal RAM.
- L2 memory also includes 1024 kB of ROM.

Shared RAM

This device also offers an on-chip 128-kB shared RAM, apart from the DSP level 1 and level 2 internal memories. This shared RAM is accessible by the DSP and also is accessible by several master peripherals.

Internal Peripherals

The following peripherals are internal to the DSP subsystem and are only accessible to the DSP:

- DSP interrupt controller (INTC)
- DSP power down controller (PDC)
- Bandwidth manager (BWM)
- Internal DMA (IDMA)

For more information on these internal peripherals, see the *TMS320C674x DSP Megamodule Reference Guide* (SPRUFK5).

4.3 Peripherals

The DSP has access to all peripherals on the device. See the device-specific data manual for the complete list of peripherals supported on your device.

Copyright © 2013–2016, Texas Instruments Incorporated

Memory Protection Unit (MPU)

Topic **Page**

5.1 Introduction **54**
5.2 Architecture **55**
5.3 MPU Registers **59**

Copyright © 2013–2016, Texas Instruments Incorporated

5.1 Introduction

This device supports one memory protection unit (MPU1). MPU1 supports the 128 kB shared RAM.

5.1.1 Purpose of the MPU

The memory protection unit (MPU) is provided to manage access to memory. The MPU allows you to define multiple ranges and limit access to system masters based on their privilege ID. The MPU can record a detected fault, or invalid access, and notify the system through an interrupt.

5.1.2 Features

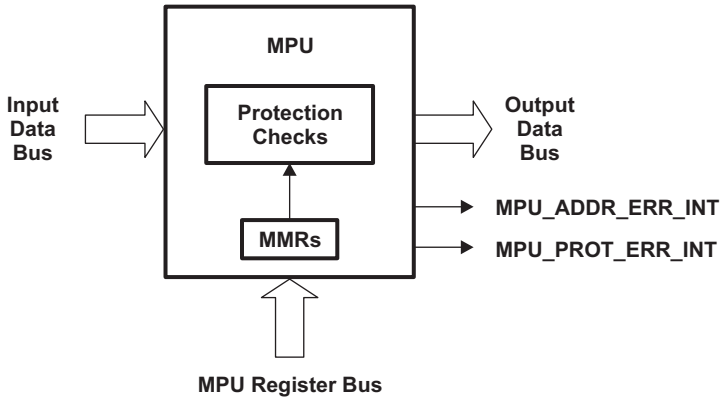
The MPU supports the following features:

- Supports multiple programmable address ranges
- Supports 0 or 1 fixed range
- Supports read, write, and execute access privileges
- Supports privilege ID associations with ranges
- Generates an interrupt when there is a protection violation, and saves violating transfer parameters
- Supports L1/L2 cache accesses
- Supports protection of its own registers

5.1.3 Block Diagram

Figure 5-1 shows a block diagram of the MPU. An access to a protected memory must pass through the MPU. During an access, the MPU checks the memory address on the input data bus against fixed and programmable ranges. If allowed, the transfer is passed unmodified to the output data bus. If the transfer fails the protection check then the MPU does not pass the transfer to the output bus but rather services the transfer internally back to the input bus (to prevent a hang) returning the fault status to the requestor as well as generating an interrupt about the fault. The MPU generates two interrupts: an address error interrupt (MPU_ADDR_ERR_INT) and a protection interrupt (MPU_PROT_ERR_INT).

Figure 5-1. MPU Block Diagram



Copyright © 2013–2016, Texas Instruments Incorporated

5.1.4 MPU Default Configuration

Table 5-1 shows the memory region protected by the MPU1. Table 5-2 shows the configuration of the MPU1.

Table 5-1. MPU Memory Regions

Unit	Memory Protection	Memory Region	
		Start Address	End Address
MPU1	128 kB Shared RAM	8000 0000h	8001 FFFFh

Table 5-2. MPU1 Default Configuration

Setting	MPU1
Default permission	Assume allowed
Number of allowed IDs supported	12
Number of fixed ranges supported	1
Number of programmable ranges supported	6
Compare width	1 kB granularity

5.2 Architecture

5.2.1 Privilege Levels

The privilege level of a memory access determines what level of permissions the originator of the memory access might have. Two privilege levels are supported: supervisor and user.

Supervisor level is generally granted access to peripheral registers and the memory protection configuration. User level is generally confined to the memory spaces that the OS specifically designates for its use.

DSP CPU instruction and data accesses have a privilege level associated with them. The privilege level is inherited from the code running on the CPU. See the *TMS320C674x DSP CPU and Instruction Set Reference Guide* (SPRUFE8) for more details on privilege levels of the DSP CPU.

Although master peripherals do not execute code, they still have a privilege level associated with them. Unlike the DSP CPU , the privilege level of this peripheral is fixed.

Table 5-3 shows the privilege ID of the CPU and every mastering peripheral. Table 5-3 also shows the privilege level (supervisor vs. user) and access type (instruction read vs. data/DMA read or write) of each master on the device. In some cases, a particular setting depends on software being executed at the time of the access or the configuration of the master peripheral.

Table 5-3. Device Master Settings

Master	Privilege ID	Privilege Level	Access Type
EDMA3CC	Inherited	Inherited	DMA
EDMA3TC0 and TC1	Inherited	Inherited	DMA
DSP	1	Software dependant	Software dependant
dMAX0/dMAX1	2	Supervisor	DMA

Copyright © 2013–2016, Texas Instruments Incorporated

5.2.2 Memory Protection Ranges

NOTE: In some cases the amount of physical memory in actual use may be less than the maximum amount of memory supported by the device. For example, the device may support a total of 512 Mbytes of SDRAM memory, but your design may only populate 128 Mbytes. In such cases, the “unpopulated” memory range must be protected in order to prevent unintended/disallowed “aliased” access to protected memory. One of the programmable address ranges could be used to detect accesses to this “unpopulated” memory.

The MPU divides its assigned memory into address ranges. Each MPU can support one fixed address range and multiple programmable address ranges. The fixed address range is configured to an exact address. The programmable address range allows software to program the start and end addresses.

Each address range has the following set of registers:

- Range start and end address registers (MPSAR and MPEAR): Specifies the starting and ending address of the address range.
- Memory protection page attribute register (MPPA): Use to program the permission settings of the address range.

It is allowed to configure ranges such that they overlap each other. In this case, all the overlapped ranges must allow the access, otherwise the access is not allowed. The final permissions given to the access are the lowest of each type of permission from any hit range.

Addresses not covered by a range are either allowed or disallowed based on the configuration of the MPU. The MPU can be configured for “assumed allowed” or “assumed disallowed” mode as dictated by the ASSUME_ALLOWED bit in the configuration register (CONFIG).

5.2.3 Permission Structures

The MPU defines a per-range permission structure with three permission fields in a 32-bit permission entry. Figure 5-2 shows the structure of a permission entry.

Figure 5-2. Permission Fields

31								22	21	20	19	18	17	16
Reserved									Allowed IDs					
									AID11	AID10	AID9	AID8	AID7	AID6
15	14	13	12	11	10	9	8	6	5	4	3	2	1	0
Allowed IDs							Reserved		Access Types					
AID5	AID4	AID3	AID2	AID1	AID0	AIX			SR	SW	SX	UR	UW	UX

5.2.3.1 Requestor-ID Based Access Controls

Each master on the device has an N-bit code associated with it that identifies it for privilege purposes. This privilege ID accompanies all memory accesses made on behalf of that master. That is, when a master triggers a memory access command, the privilege ID will be carried alongside the command.

Each memory protection range has an allowed ID (AID) field associated with it that indicates which requestors may access the given address range. The MPU maps the privilege IDs of all the possible requestors to bits in the allowed IDs field in the memory protection page attribute registers (MPPA).

- AID0 through AID11 are used to specify the allowed privilege IDs.
- An additional allowed ID bit, AIDX, captures access made by all privilege IDs not covered by AID0 through AID11.

When set to 1, the AID bit grants access to the corresponding ID. When cleared to 0, the AID bit denies access to the corresponding requestor.

Copyright © 2013–2016, Texas Instruments Incorporated

5.2.3.2 Request-Type Based Permissions

The memory protection model defines three fundamental functional access types: read, write, and execute. Read and write refer to data accesses -- accesses originating via the load/store units on the CPU or via a master peripheral. Execute refers to accesses associated with an instruction fetch.

The memory protection model allows controlling read, write, and execute permissions independently for both user and supervisor mode. This results in six permission bits, listed in Table 5-4. For each bit, a 1 permits the access type and a 0 denies access. For example, UX = 1 means that User Mode may execute from the given page. The memory protection unit allows you to specify all six of these bits separately; 64 different encodings are permitted altogether, although programs might not use all of them.

Table 5-4. Request Type Access Controls

Bit	Field	Description
5	SR	Supervisor may read
4	SW	Supervisor may write
3	SX	Supervisor may execute
2	UR	User may read
1	UW	User may write
0	UX	User may execute

5.2.4 Protection Check

During a memory access, the MPU checks if the address range of the input transfer overlaps one of the address ranges. When the input transfer address is within a range the transfer parameters are checked against the address range permissions.

The MPU first checks the transfer's privilege ID against the AID settings. If the AID bit is 0, then the range will not be checked; if the AID bit is 1, then the transfer parameters are checked against the memory protection page attribute register (MPPA) values to detect an allowed access.

For non-debug accesses, the read, write, and execute permissions are also checked. There is a set of permissions for supervisor mode and a set for user mode. For supervisor mode accesses, the SR, SW, and SX bits are checked. For user mode accesses, the UR, UW, and UX bits are checked.

If the transfer address range does not match any address range then the transfer is either allowed or disallowed based on the configuration of the MPU. The MPU can be configured for "assumed allowed" or "assumed disallowed" mode as dictated by the ASSUME_ALLOWED bit in the configuration register (CONFIG).

In the case that a transfer spans multiple address ranges, all the overlapped ranges must allow the access, otherwise the access is not allowed. The final permissions given to the access are the lowest of each type of permission from any hit range. Therefore, if a transfer matches 2 ranges, one that is RW and one that is RX, then the final permission is just R.

The MPU has a special mechanism for handling DSP L1/L2 cache controller read accesses, see Section 5.2.5 for more details.

5.2.5 DSP L1/L2 Cache Controller Accesses

A memory read access that originates from the DSP L1/L2 cache is treated differently to allow memory protection to be enforced by the DSP level. This is because a subsequent memory access that hits in the cache does not pass through the MPU. Instead the memory access is serviced directly by the L1/L2 memory controllers.

Copyright © 2013–2016, Texas Instruments Incorporated

During a cache memory read, the permission settings stored in the memory protection page attribute registers (MPPA) are passed to the L1/L2 memory controllers along with the read data. The permissions settings returned by the MPU are taken from MPPA that covers the address range of the original request—only the SR, SW, SX, UR, UW, and UX bits are passed. If the request address is covered by multiple address ranges, then the returned value is the logical-AND of all MPPA permissions. If the transfer address range is not covered by an address range then the transfer is either allowed or disallowed based on the configuration of the MPU.

5.2.6 MPU Register Protection

Access to the range start and end address registers (MPSAR and MPEAR) and memory protection page attribute registers (MPPA) is also protected. All non-debug writes must be by a supervisor entity. A protection fault can occur from a register write with invalid permissions and this triggers an interrupt just like a memory access.

Faults are not recorded (nor interrupts generated) for debug accesses.

5.2.7 Invalid Accesses and Exceptions

When a transfer fails the protection check, the MPU does not pass the transfer to the output bus. The MPU instead services the transfer locally to prevent a hang and returns a protection error to the requestor. The behavior of the MPU depends on whether the access was a read or a write:

- For a read: The MPU returns 0s, a permission value is 0 (no access allowed), a protection error status.
- For a write: The MPU receives all the write data and returns a protection error status.

The MPU captures system faults due to addressing or protection violations in its registers. The MPU can store the fault information for only one fault, so the first detected fault is recorded into the fault registers and an interrupt is generated. Software must use the fault clear register (FLTCLR) to clear the fault status so that another fault can be recorded. The MPU will not record another fault nor generate another interrupt until the existing fault has been cleared. Also, additional faults will be ignored. Faults are not recorded (no interrupts generated) for debug accesses.

5.2.8 Reset Considerations

After reset, the memory protection page attribute registers (MPPA) default to 0. This disables all protection features.

5.2.9 Interrupt Support

5.2.9.1 Interrupt Events and Requests

The MPU generates two interrupts: an address error interrupt (MPU_ADDR_ERR_INT) and a protection interrupt (MPU_PROT_ERR_INT). The MPU_ADDR_ERR_INT is generated when there is an addressing violation due to an access to a non-existent location in the MPU register space. The MPU_PROT_ERR_INT interrupt is generated when there is a protection violation of either in the defined ranges or to the MPU registers.

The transfer parameters that caused the violation are saved in the MPU registers.

5.2.9.2 Interrupt Multiplexing

The interrupt from MPU1 is combined with the boot configuration module into a single interrupt called MPU_BOOTCFG_ERR. The combined interrupt is routed to the DSP interrupt controller. Table 5-5 shows the interrupt sources that are combined to make MPU_BOOTCFG_ERR.

Copyright © 2013–2016, Texas Instruments Incorporated

Table 5-5. MPU_BOOTCFG_ERR Interrupt Sources

Interrupt	Source
MPU1_ADDR_ERR_INT	MPU1 address error interrupt
MPU1_PROT_ERR_INT	MPU1 protection interrupt
BOOTCFG_ADDR_ERR	Boot configuration address error
BOOTCFG_PROT_ERR	Boot configuration protection error

5.2.10 Emulation Considerations

Memory and MPU registers are not protected against emulation accesses.

5.3 MPU Registers

Table 5-6 lists the memory-mapped registers for the MPU1.

Table 5-6. Memory Protection Unit 1 (MPU1) Registers

Address	Acronym	Register Description	Section
01E1 4000h	REVID	Revision identification register	Section 5.3.1
01E1 4004h	CONFIG	Configuration register	Section 5.3.2
01E1 4010h	IRAWSTAT	Interrupt raw status/set register	Section 5.3.3
01E1 4014h	IENSTAT	Interrupt enable status/clear register	Section 5.3.4
01E1 4018h	IENSET	Interrupt enable set register	Section 5.3.5
01E1 401Ch	IENCLR	Interrupt enable clear register	Section 5.3.6
01E1 4200h	PROG1_MPSAR	Programmable range 1 start address register	Section 5.3.7
01E1 4204h	PROG1_MPEAR	Programmable range 1 end address register	Section 5.3.8
01E1 4208h	PROG1_MPPA	Programmable range 1 memory protection page attributes register	Section 5.3.9
01E1 4210h	PROG2_MPSAR	Programmable range 2 start address register	Section 5.3.7
01E1 4214h	PROG2_MPEAR	Programmable range 2 end address register	Section 5.3.8
01E1 4218h	PROG2_MPPA	Programmable range 2 memory protection page attributes register	Section 5.3.9
01E1 4220h	PROG3_MPSAR	Programmable range 3 start address register	Section 5.3.7
01E1 4224h	PROG3_MPEAR	Programmable range 3 end address register	Section 5.3.8
01E1 4228h	PROG3_MPPA	Programmable range 3 memory protection page attributes register	Section 5.3.9
01E1 4230h	PROG4_MPSAR	Programmable range 4 start address register	Section 5.3.7
01E1 4234h	PROG4_MPEAR	Programmable range 4 end address register	Section 5.3.8
01E1 4238h	PROG4_MPPA	Programmable range 4 memory protection page attributes register	Section 5.3.9
01E1 4240h	PROG5_MPSAR	Programmable range 5 start address register	Section 5.3.7
01E1 4244h	PROG5_MPEAR	Programmable range 5 end address register	Section 5.3.8
01E1 4248h	PROG5_MPPA	Programmable range 5 memory protection page attributes register	Section 5.3.9
01E1 4250h	PROG6_MPSAR	Programmable range 6 start address register	Section 5.3.7
01E1 4254h	PROG6_MPEAR	Programmable range 6 end address register	Section 5.3.8
01E1 4258h	PROG6_MPPA	Programmable range 6 memory protection page attributes register	Section 5.3.9
01E1 4300h	FLTADDRR	Fault address register	Section 5.3.10
01E1 4304h	FLTSTAT	Fault status register	Section 5.3.11
01E1 4308h	FLTCLR	Fault clear register	Section 5.3.12

Copyright © 2013–2016, Texas Instruments Incorporated

5.3.1 Revision Identification Register (REVID)

The revision ID register (REVID) contains the MPU revision. The REVID is shown in Figure 5-3 and described in Table 5-7.

Figure 5-3. Revision ID Register (REVID)

31	0
REV	
R-4E81 0101h	

LEGEND: R = Read only; -*n* = value after reset

Table 5-7. Revision ID Register (REVID) Field Descriptions

Bit	Field	Value	Description
31-0	REV	4E81 0101h	Revision ID of the MPU.

5.3.2 Configuration Register (CONFIG)

The configuration register (CONFIG) contains the configuration value of the MPU. The CONFIG is shown in Figure 5-4 and described in Table 5-8.

NOTE: Although the NUM_AIDS bit defaults to 12 (Ch), not all AIDs may be supported on your device. Unsupported AIDs should be cleared to 0 in the memory page protection attributes registers (MPPA). See Table 5-3 for a list of AIDs supported on your device.

Figure 5-4. Configuration Register (CONFIG)

31	24	23	20	19	16
ADDR_WIDTH		NUM_FIXED		NUM_PROG	
R-0		R-0		R-6h	

15	12	11	1	0
NUM_AIDS		Reserved		ASSUME_ALLOWED
R-Ch		R-0		R-1

LEGEND: R = Read only; -*n* = value after reset

Table 5-8. Configuration Register (CONFIG) Field Descriptions

Bit	Field	Value	Description
31-24	ADDR_WIDTH	0-FFh	Address alignment (2^n kByte alignment) for range checking.
23-20	NUM_FIXED	0-Fh	Number of fixed address ranges.
19-16	NUM_PROG	0-Fh	Number of programmable address ranges.
15-12	NUM_AIDS	0-Fh	Number of supported AIDs.
11-1	Reserved	0	Reserved
0	ASSUME_ALLOWED		Assume allowed. When an address is not covered by any MPU protection range, this bit determines whether the transfer is assumed to be allowed or not allowed.
		0	Assume is disallowed.
		1	Assume is allowed.

Copyright © 2013–2016, Texas Instruments Incorporated

5.3.3 Interrupt Raw Status/Set Register (IRAWSTAT)

Reading the interrupt raw status/set register (IRAWSTAT) returns the status of all interrupts. Software can write to IRAWSTAT to manually set an interrupt; however, an interrupt is generated only if the interrupt is enabled in the interrupt enable set register (IENSET). Writes of 0 have no effect. The IRAWSTAT is shown in Figure 5-5 and described in Table 5-9.

Figure 5-5. Interrupt Raw Status/Set Register (IRAWSTAT)

31	16
Reserved	
R-0	

15	2	1	0
Reserved		ADDRERR	PROTERR
R-0		R/W-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 5-9. Interrupt Raw Status/Set Register (IRAWSTAT) Field Descriptions

Bit	Field	Value	Description
31-2	Reserved	0	Reserved
1	ADDRERR		Address violation error. Reading this bit reflects the status of the interrupt. Writing 1 sets the status; writing 0 has no effect.
		0	Interrupt is not set.
		1	Interrupt is set.
0	PROTERR		Protection violation error. Reading this bit reflects the status of the interrupt. Writing 1 sets the status; writing 0 has no effect.
		0	Interrupt is not set.
		1	Interrupt is set.

Copyright © 2013–2016, Texas Instruments Incorporated

5.3.4 Interrupt Enable Status/Clear Register (IENSTAT)

Reading the interrupt enable status/clear register (IENSTAT) returns the status of only those interrupts that are enabled in the interrupt enable set register (IENSET). Software can write to IENSTAT to clear an interrupt; the interrupt is cleared from both IENSTAT and the interrupt raw status/set register (IRAWSTAT). Writes of 0 have no effect. The IENSTAT is shown in Figure 5-6 and described in Table 5-10.

Figure 5-6. Interrupt Enable Status/Clear Register (IENSTAT)

31	16
Reserved	
R-0	

15	2	1	0
Reserved		ADDRERR	PROTERR
R-0		R/W-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 5-10. Interrupt Enable Status/Clear Register (IENSTAT) Field Descriptions

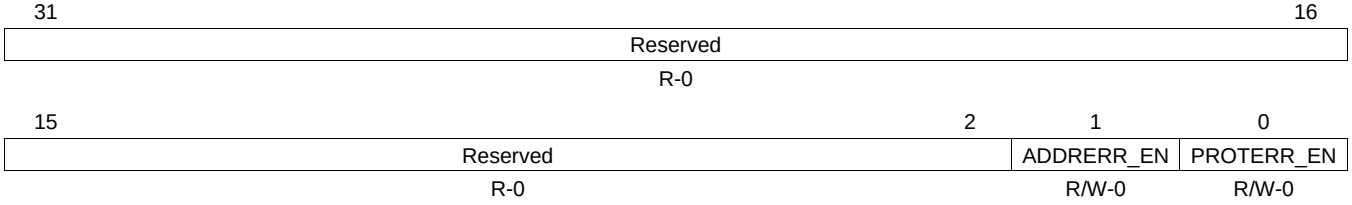
Bit	Field	Value	Description
31-2	Reserved	0	Reserved
1	ADDRERR		Address violation error. If the interrupt is enabled, reading this bit reflects the status of the interrupt. If the interrupt is disabled, reading this bit returns 0. Writing 1 sets the status; writing 0 has no effect.
		0	Interrupt is not set.
		1	Interrupt is set.
0	PROTERR		Protection violation error. If the interrupt is enabled, reading this bit reflects the status of the interrupt. If the interrupt is disabled, reading this bit returns 0. Writing 1 sets the status; writing 0 has no effect.
		0	Interrupt is not set.
		1	Interrupt is set.

Copyright © 2013–2016, Texas Instruments Incorporated

5.3.5 Interrupt Enable Set Register (IENSET)

Reading the interrupt enable set register (IENSET) returns the interrupts that are enabled. Software can write to IENSET to enable an interrupt. Writes of 0 have no effect. The IENSET is shown in Figure 5-7 and described in Table 5-11.

Figure 5-7. Interrupt Enable Set Register (IENSET)



LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

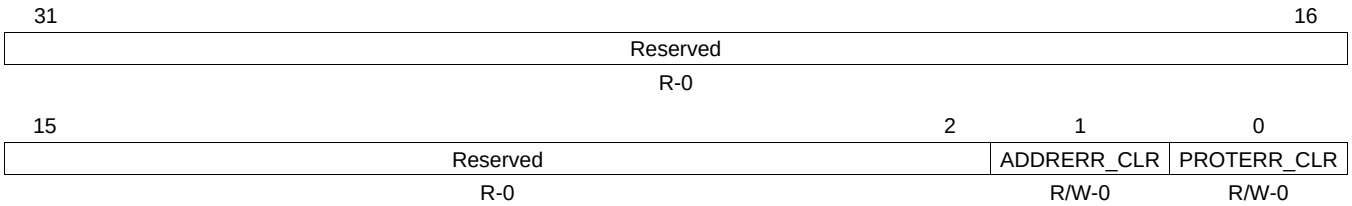
Table 5-11. Interrupt Enable Set Register (IENSET) Field Descriptions

Bit	Field	Value	Description
31-2	Reserved	0	Reserved
1	ADDRERR_EN		Address violation error enable.
		0	Writing 0 has no effect.
		1	Interrupt is enabled.
0	PROTERR_EN		Protection violation error enable.
		0	Writing 0 has no effect.
		1	Interrupt is enabled.

5.3.6 Interrupt Enable Clear Register (IENCLR)

Reading the interrupt enable clear register (IENCLR) returns the interrupts that are enabled. Software can write to IENCLR to clear/disable an interrupt. Writes of 0 have no effect. The IENCLR is shown in Figure 5-8 and described in Table 5-12.

Figure 5-8. Interrupt Enable Clear Register (IENCLR)



LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 5-12. Interrupt Enable Clear Register (IENCLR) Field Descriptions

Bit	Field	Value	Description
31-2	Reserved	0	Reserved
1	ADDRERR_CLR		Address violation error disable.
		0	Writing 0 has no effect.
		1	Interrupt is cleared/disabled.
0	PROTERR_CLR		Protection violation error disable.
		0	Writing 0 has no effect.
		1	Interrupt is cleared/disabled.

Copyright © 2013–2016, Texas Instruments Incorporated

5.3.7 Programmable Range *n* Start Address Registers (PROG*n*_MPSAR)

NOTE: In some cases the amount of physical memory in actual use may be less than the maximum amount of memory supported by the device. For example, the device may support a total of 512 Mbytes of SDRAM memory, but your design may only populate 128 Mbytes. In such cases, the unpopulated memory range must be protected in order to prevent unintended/disallowed aliased access to protected memory, especially memory. One of the programmable address ranges could be used to detect accesses to this unpopulated memory.

The programmable range *n* start address register (PROG*n*_MPSAR) holds the start address for the range *n*. The PROG*n*_MPSAR is writeable by a supervisor entity only.

The start address must be aligned on a page boundary. The page size for MPU1 is 1 kBbyte. The size of the page determines the width of the address field in PROG*n*_MPSAR and the programmable range *n* end address register (PROG*n*_MPEAR). For example, to protect a 64-kB page starting at byte address 8001 0000h, write 8001 0000h to PROG*n*_MPSAR and 8001 FFFFh to PROG*n*_MPEAR.

The PROG*n*_MPSAR for MPU1 is shown in Figure 5-9 and described in Table 5-13.

Figure 5-9. MPU1 Programmable Range *n* Start Address Register (PROG*n*_MPSAR)

31	10	9	0
START_ADDR		Reserved	
R/W-20 0000h		R-0	

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 5-13. MPU1 Programmable Range *n* Start Address Register (PROG*n*_MPSAR) Field Descriptions

Bit	Field	Value	Description
31-10	START_ADDR	20 0000h–20 007Fh	Start address for range N.
9-0	Reserved	0	Reserved

Copyright © 2013–2016, Texas Instruments Incorporated

5.3.8 Programmable Range *n* End Address Registers (PROG*n*_MPEAR)

The programmable range *n* end address register (PROG*n*_MPEAR) holds the end address for the range *n*. This register is writeable by a supervisor entity only.

The end address must be aligned on a page boundary. The page size for MPU1 is 1 kByte. The size of the page determines the width of the address field in the programmable range *n* start address register (PROG*n*_MPSAR) and PROG*n*_MPEAR. For example, to protect a 64-kB page starting at byte address 8001 0000h, write 8001 0000h to PROG*n*_MPSAR and 8001 FFFFh to PROG*n*_MPEAR.

The PROG*n*_MPEAR for MPU1 is shown in Figure 5-10 and described in Table 5-14.

Figure 5-10. MPU1 Programmable Range *n* End Address Register (PROG*n*_MPEAR)

31	10	9	0
END_ADDR		Reserved	
R/W-20 007Fh		R-3FFh	

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 5-14. MPU1 Programmable Range *n* End Address Register (PROG*n*_MPEAR) Field Descriptions

Bit	Field	Value	Description
31-10	END_ADDR	20 0000h–20 007Fh	End address for range N.
9-0	Reserved	3FFh	Reserved

Copyright © 2013–2016, Texas Instruments Incorporated

5.3.9 Programmable Range *n* Memory Protection Page Attributes Register (PROG*n*_MPPA)

The programmable range *n* memory protection page attributes register (PROG*n*_MPPA) holds the permissions for the region *n*. This register is writeable only by a supervisor entity. The PROG*n*_MPPA is shown in Figure 5-11 and described in Table 5-15.

Figure 5-11. Programmable Range Memory Protection Page Attributes Register (PROG*n*_MPPA)

31										26	25						22	21	20	19	18	17	16
Reserved											Reserved							AID11	AID10	AID9	AID8	AID7	AID6
R-0											R-Fh							R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0								
AID5	AID4	AID3	AID2	AID1	AID0	AIDX	Rsvd	Rsvd	Rsvd	SR	SW	SX	UR	UW	UX								
R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R-0	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1								

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 5-15. Programmable Range Memory Protection Page Attributes Register (PROG*n*_MPPA) Field Descriptions

Bit	Field	Value	Description
31-26	Reserved	0	Reserved
25-22	Reserved	Fh	Reserved
21-10	AID*n*		Controls access from ID = *n*.
		0	Access is denied.
		1	Access is granted.
9	AIDX		Controls access from ID > 11.
		0	Access is denied.
		1	Access is granted.
8	Reserved	0	Reserved
7	Reserved	1	Reserved. This bit must be written as 1.
6	Reserved	1	Reserved. This bit must be written as 1.
5	SR		Supervisor Read permission.
		0	Access is denied.
		1	Access is allowed.
4	SW		Supervisor Write permission.
		0	Access is denied.
		1	Access is allowed.
3	SX		Supervisor Execute permission.
		0	Access is denied.
		1	Access is allowed.
2	UR		User Read permission.
		0	Access is denied.
		1	Access is allowed.
1	UW		User Write permission.
		0	Access is denied.
		1	Access is allowed.
0	UX		User Execute permission.
		0	Access is denied.
		1	Access is allowed.

Copyright © 2013–2016, Texas Instruments Incorporated

5.3.10 Fault Address Register (FLTADDRR)

The fault address register (FLTADDRR) holds the address of the first protection fault transfer. The FLTADDRR is shown in Figure 5-12 and described in Table 5-16.

Figure 5-12. Fault Address Register (FLTADDRR)

31	0
FLTADDR	
R-0	

LEGEND: R = Read only; -*n* = value after reset

Table 5-16. Fault Address Register (FLTADDRR) Field Descriptions

Bit	Field	Value	Description
31-0	FLTADDR	0-FFFF FFFFh	Memory address of fault.

Copyright © 2013–2016, Texas Instruments Incorporated

5.3.11 Fault Status Register (FLTSTAT)

The fault status register (FLTSTAT) holds the status and attributes of the first protection fault transfer. The FLTSTAT is shown in Figure 5-13 and described in Table 5-17.

Figure 5-13. Fault Status Register (FLTSTAT)

31	24	23	16
Reserved		MSTID	
R-0		R-0	

15	13	12	9	8	6	5	0
Reserved		PRIVID		Reserved		TYPE	
R-0		R-0		R-0		R-0	

LEGEND: R = Read only; -*n* = value after reset

Table 5-17. Fault Status Register (FLTSTAT) Field Descriptions

Bit	Field	Value	Description
31-24	Reserved	0	Reserved
23-16	MSTID	0-FFh	Master ID of fault transfer.
15-13	Reserved	0	Reserved
12-9	PRIVID	0-Fh	Privilege ID of fault transfer.
8-6	Reserved	0	Reserved
5-0	TYPE	0-3Fh	Fault type. The TYPE bit field is cleared when a 1 is written to the CLEAR bit in the fault clear register (FLTCLR).
		0	No fault.
		1h	User execute fault.
		2h	User write fault.
		3h	Reserved
		4h	User read fault.
		5h-7h	Reserved
		8h	Supervisor execute fault.
		9h-Fh	Reserved
		10h	Supervisor write fault.
		11h	Reserved
		12h	Relaxed cache write back fault.
		13h-1Fh	Reserved
		20h	Supervisor read fault.
		21h-3Eh	Reserved
		3Fh	Relaxed cache line fill fault.

Copyright © 2013–2016, Texas Instruments Incorporated

5.3.12 Fault Clear Register (FLTCLR)

The fault clear register (FLTCLR) allows software to clear the current fault so that another can be captured in the fault status register (FLTSTAT) as well as produce an interrupt. Only the TYPE bit field in FLTSTAT is cleared when a 1 is written to the CLEAR bit. The FLTCLR is shown in Figure 5-14 and described in Table 5-18.

Figure 5-14. Fault Clear Register (FLTCLR)

31	16
Reserved	
R-0	

15	1	0
Reserved		CLEAR
R-0		W-0

LEGEND: R = Read only; W = Write only; -*n* = value after reset

Table 5-18. Fault Clear Register (FLTCLR) Field Descriptions

Bit	Field	Value	Description
31-1	Reserved	0	Reserved
0	CLEAR		Command to clear the current fault. Writing 0 has no effect.
		0	No effect.
		1	Clear the current fault.

Copyright © 2013–2016, Texas Instruments Incorporated

Device Clocking

Topic **Page**

6.1 Overview 71
6.2 Frequency Flexibility 72
6.3 I/O Domains 74

Copyright © 2013–2016, Texas Instruments Incorporated

6.1 Overview

This device requires one reference clock for the phase-locked loop controller (PLLC). This reference clock may be sourced from either a crystal input or by an external oscillator. For detailed specifications on clock frequency and voltage requirements, see the device-specific data manual.

All possible input clocks are described in Table 6-1. The CPU and the majority of the device peripherals operate at fixed ratios of the primary system/CPU clock frequency, as listed in Table 6-2. Figure 6-1 shows the clocking architecture.

Table 6-1. Device Clock Inputs

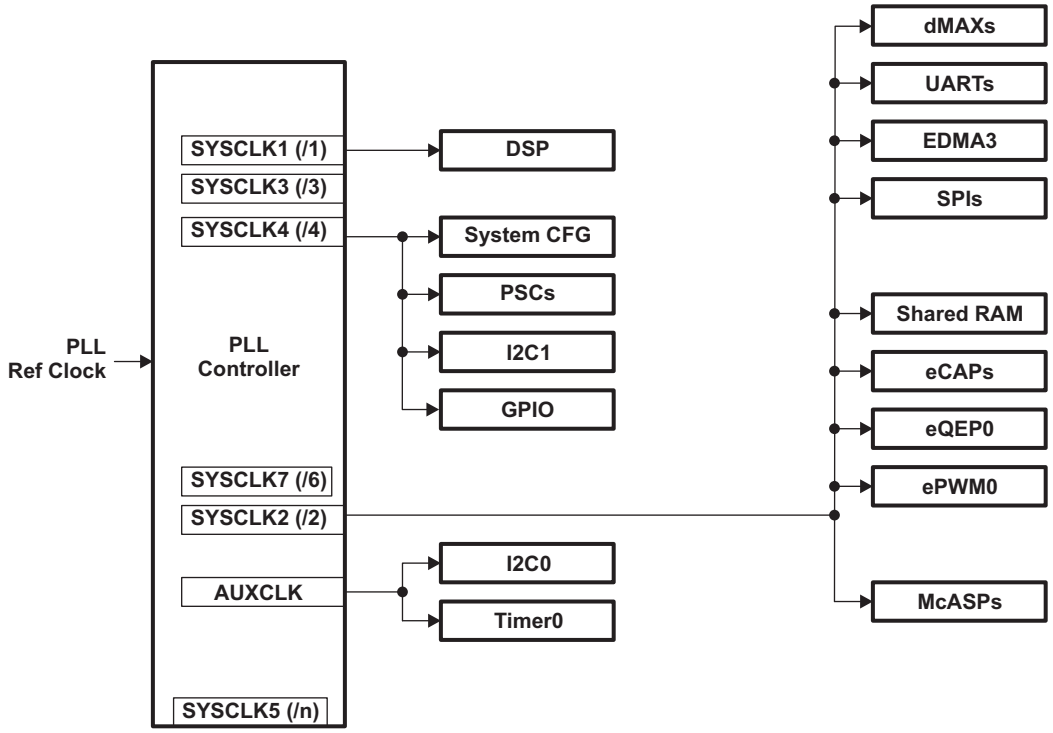
Peripheral	Input Clock Signal Name
Oscillator/PLL	OSCIN
JTAG	TCK
McASPs	ACLKRn, AHCLKRn, ACLKXn, AHCLKXn
I2Cs	I2Cn_SCL
SPIs	SPIn_CLK
Timer0	TM64P0_IN12

Table 6-2. System Clock Domains

CPU/Device Peripherals	System Clock Domain	Fixed Ratio to Clock Required?	Default Ratio to Clock
DSP	SYSCLK1	Yes	1:1
dMAX, UARTs, EDMA, SPIs, Shared RAM, eCAPs, eQEP0, eHRPWM0, McASPs,	SYSCLK2	Yes	1:2
SYSCFG, PSCs, I2C1, GPIO	SYSCLK4	Yes	1:4
I2C0, Timers, McASP serial clock	AUXCLK	Not Applicable	PLL Bypass Clock

Copyright © 2013–2016, Texas Instruments Incorporated

Figure 6-1. Overall Clocking Diagram



6.2 Frequency Flexibility

There are two clocking modes:

- PLL Bypass that can serve as a power savings mode
- PLL Active where the PLL is enabled and multiplies the input clock up to the desired operating frequency

When the PLL is in Bypass mode, the reference clock supplied on OSCIN serves as the clock source from which all of the system clocks (SYSCLK1-SYSCLK7) are derived. This means, when the PLL is in Bypass mode, the reference clock supplied on OSCIN passes directly to the system of PLLDIV blocks that creates each of the system clocks. When the PLL operates in Active mode, the PLL is enabled and the PLL multiplier setting is used to multiply the input clock frequency supplied on the OSCIN pin up to the desired frequency. It is this multiplied frequency that all system clocks are derived from in PLL Active mode.

The output of the PLL multiplier passes through a post divider (POSTDIV) block and then is applied to the system of PLLDIV blocks that creates each of the system clock domains (SYSCLK1-SYSCLK7). Each SYSCLK has a PLLDIV block associated with it. See the *Phase-Locked Loop Controller (PLLC)* chapter for more details on the PLL.

The combination of the PLL multiplier, POSTDIV, and PLLDIV blocks provides flexibility in the frequencies that the system clock domains support. This flexibility does have limitations, as follows:

- OSCIN input frequency is limited to a supported range.
- The output of the PLL Multiplier must be within the range specified in the device-specific data manual.
- The output of each PLLDIV block must be less than or equal to the maximum device frequency specified in the device-specific data manual.

NOTE: The above limitations are provided here as an example and are used to illustrate the recommended configuration of the PLL controller. These limitations may vary based on core voltage and between devices. See the device-specific data manual for more details.

Copyright © 2013–2016, Texas Instruments Incorporated

Table 6-3 shows examples of possible PLL multiplier settings, along with the available PLL post-divider modes. The PLL post-divider modes are defined by the value programmed in the RATIO field of the PLL post-divider control register (POSTDIV). For Div1, Div2, Div3, and Div4 modes, the RATIO field would be programmed to 0, 1, 2, and 3, respectively. The Div1, Div2, Div3, and Div4 modes are shown here as an example. Additional post-divider modes are supported and are documented in the *Phase-Locked Loop Controller (PLLC)* chapter.

NOTE: PLL power consumption increases as the output frequency of the PLL multiplier increases. To decrease PLL power consumption, the lowest PLL multiplier (PLLM) setting should be chosen that achieves the desired frequency. For example, if 200 MHz is the desired CPU operating frequency and the OSCIN frequency is 25 MHz; lower power consumption is achieved by choosing a PLLM setting of ×16 and a post-divider (POSTDIV) setting of /2 instead of a PLLM setting of ×24 and a POSTDIV setting of /3, even though both of these modes would result in a CPU frequency of 200 MHz.

Table 6-3. Example PLL Frequencies

OSCIN Frequency	PLL Multiplier	Multiplier Frequency (MHz)	Div1	Div2	Div3	Div4
20	30	600	600	300	200	150
24	25	600	600	300	200	150
25	24	600	600	300	200	150
30	20	600	600	300	200	150
20	25	500	500	250	167	125
24	20	480	480	240	160	120
25	18	450	450	225	150	112.5
30	14	420	420	210	140	105
25	16	400	400	200	133	100

Copyright © 2013–2016, Texas Instruments Incorporated

6.3 I/O Domains

The I/O domains refer to the frequencies of the peripherals that communicate through device pins. In many cases, there are frequency requirements for a peripheral pin interface that are set by an outside standard and must be met. It is not necessarily possible to obtain these frequencies from the on-chip clock generation circuitry, so the frequencies must be obtained from external sources and are asynchronous to the CPU frequency by definition.

Peripherals can be divided into 4 groups, depending upon their clock requirements, as shown in Table 6-4.

Table 6-4. Peripherals

Peripheral Group	Peripheral Group Definition	Peripherals Contained within Group	Source of Peripheral Clock
Fixed-Frequency Peripherals	As the name suggests, fixed-frequency peripherals have a fixed-frequency. They are fed the AUXCLK directly from the oscillator input.	Timer0	—
		I2C0	—
Synchronous Peripherals	Synchronous peripherals have their frequencies derived from the CPU clock frequency. The peripheral system clock frequency changes accordingly, if the PLL1 frequency changes. Most synchronous peripherals have internal dividers so they can generate their required clock frequencies.	eCAPs	—
		eQEP0	—
		eHRPWM0	—
		UARTs	—
		GPIO	—
Synchronous/Asynchronous Peripherals	Synchronous/asynchronous peripherals can be run with either internally generated synchronous clocks, or externally generated asynchronous clocks.	McASPs	AUXCLK or Peripheral Serial Clocks
		SPIs	SYSCLK2 or Peripheral Serial Clock
		I2C1	SYSCLK4 or Peripheral Serial Clock

Copyright © 2013–2016, Texas Instruments Incorporated

Phase-Locked Loop Controller (PLLC)

Topic **Page**

7.1 Introduction 76
7.2 PLL0 Control 76
7.3 Locking/Unlocking PLL Register Access 79
7.4 PLLC Registers 80

Copyright © 2013–2016, Texas Instruments Incorporated

7.1 Introduction

This device has one phase-locked loop (PLL) controller, PLL0, that provides a clock to different parts of the system. PLL0 provides clocks (through various dividers) to most of the components of the device.

The PLL0 provides the following:

- Glitch-Free Transitions (on changing clock settings)
- Domain Clocks Alignment
- Clock Gating
- PLL power-down

The various clock outputs given by the controller are as follows:

- Domain Clocks: SYSCLK [1:n]
- Auxiliary Clock from reference clock source: AUXCLK

Various dividers that can be used are as follows:

- Pre-PLL Divider: PREDIV
- Post-PLL Divider: POSTDIV
- SYSCLK Divider: D1, …, Dn

Various other controls supported are as follows:

- PLL Multiplier Control: PLLM
- Software programmable PLL Bypass: PLLEN

7.2 PLL0 Control

PLL0 supplies the primary system clock. Software controls the PLL0 operation through the system PLL controller 0 (PLLC0) registers. Figure 7-1 shows the PLL0 in the device.

AUXCLK is the clock provided to the fixed clock domain.

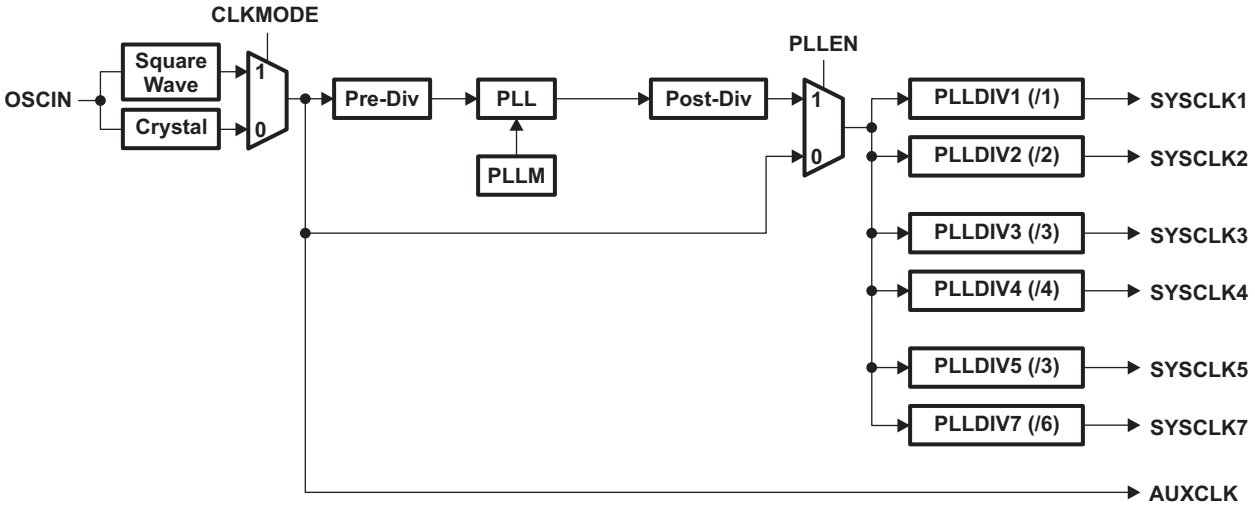
The PLL0 multiplier is controlled by the PLLM bits in the PLL multiplier control register (PLLM) and is set to a default value of 0000 0013h at power-up, resulting in a PLL multiplier of 20×. The PLL0 output clock may be divided-down for slower device operation using the PLLC0 post-divider. This divider defaults to a /2 value, but may be modified by software (RATIO bit in POSTDIV) to achieve lower power device operation. These default settings yield a 300-MHz PLL output clock when using a 30-MHz clock source. The PLL0 multiplier may be modified by software.

At power-up, PLL0 is powered-down/disabled and must be powered-up by software through the PLLPWRDN bit in the PLL control register (PLLCTL). The system operates in bypass mode by default and the system clock (OSCIN) is provided directly from an input reference clock (square wave or internal oscillator) selected by the CLKMODE bit in PLLCTL. Once the PLL is powered-up and locked, software can switch the device to PLL mode operation (set the PLLEN bit in PLLCTL to 1).

Registers used in PLLC0 are listed in Section 7.4.

Copyright © 2013–2016, Texas Instruments Incorporated

Figure 7-1. PLL0 Structure



7.2.1 Device Clock Generation

PLL0 is controlled by PLL controller 0. The PLLC0 manages the clock ratios, alignment, and gating for the system clocks to the chip. The PLLC is responsible for controlling all modes of the PLL through software, in terms of pre-division of the clock inputs, multiply factor within the PLL, and post-division for each of the chip-level clocks from the PLL output. The PLLC also controls reset propagation through the chip, clock alignment, and test points.

PLLC0 generates several clocks from the PLL0 output clock for use by the various processors and modules. These are summarized in Table 7-1. The output clock divider values SYSCLK1 to SYSCLK*n* are fixed. This maintains the clock ratios between the various device components no matter what reference clock (PLL or bypass) or PLL frequency is used.

Table 7-1. System PLLC0 Output Clocks

Output Clock	Used by	Default Ratio (relative to SYSCLK1)	Notes
SYSCLK1	DSP	/1	Fixed Ratio
SYSCLK2	EDMA, DSP ports, eCAPs, eHRPWM0, eQEP0, Shared RAM, McASPs, SPIs, UARTs, dMAX	/2	Fixed Ratio
SYSCLK4	System configuration (SYSCFG), PLLC0, PSCs, GPIO, I2C1	/4	Fixed Ratio
AUXCLK	McASP serial clock, Timers, I2C0	PLL Bypass Clock	Not Applicable

- The divide values in PLL controller 0 for SYSCLK1, SYSCLK2, and SYSCLK4 are not fixed so that you can change the divide values for power saving reasons. But you are responsible to assure that the divide ratios between these clock domains must be fixed to 1:2:4.
- PLL controller supports post-divider value n = 4.5. When 4.5 divide values are used, the duty cycle of the resulting clock will not be 50%. In this case, the duty cycle will be 44.4%.

Copyright © 2013–2016, Texas Instruments Incorporated

7.2.2 Steps for Changing PLL0 Domain Frequency

Refer to the appropriate subsection on how to program the PLL0/Core Domain clocks:

- If the PLL is powered down (PLLPWRDN bit in PLLCTL is set to 1), follow the full PLL initialization procedure in Section 7.2.2.1 to initialize the PLL.
- If the PLL is not powered down (PLLPWRDN bit in PLLCTL is cleared to 0), follow the sequence in Section 7.2.2.2 to change the PLL multiplier.
- If the PLL is already running at a desired multiplier and you only want to change the SYSCLK dividers, follow the sequence in Section 7.2.2.3.

Note that the PLL is powered down after a Power-on Reset (POR). The PLL is not powered down after a Warm Reset (RESET), but the PLLEN bit in PLLCTL is cleared to 0 (bypass mode) and the PLLDIVx registers are reset to default values.

7.2.2.1 Initializing PLL Mode from PLL Power Down

If the PLL is powered down (PLLPWRDN bit in PLLCTL is set to 1), perform the following procedure to initialize the PLL:

1. Clear the PLLEN bit in PLLCTL to 0 (select PLL Bypass mode) and reset the PLL by clearing PLLRST bit in PLLCTL. Wait for 4 OSCIN cycles to ensure PLLC switches to bypass mode properly.
2. Select the clock mode by programming the CLKMODE bit in PLLCTL.
   (a) Clear the PLLENSRC bit in PLLCTL to 0 to allow PLLCTL.PLLEN to take effect.
   (b) PLLCTL.EXTCLKSRC should be left to 0.
3. Clear the PLLRST bit in PLLCTL to 0 (reset PLL).
4. Clear the PLLPWRDN bit in PLLCTL to 0 to bring the PLL out of power-down mode.
5. Program the required multiplier value in PLLM. If desired to scale all the SYSCLK frequencies of a given PLLC, program the POSTDIV ratio.
6. If necessary, program PLLDIVn registers to change the SYSCLK0 to SYSCLKn divide values:
   (a) Check for GOSTAT bit in PLLSTAT to clear to 0 to indicate that no GO operation is currently in progress.
   (b) Program the RATIO field in PLLDIVx with the desired divide factors.
   (c) Set the GOSET bit in PLLCMD to 1 to initiate a new divider transition.
   (d) Wait for the GOSTAT bit in PLLSTAT to clear to 0 (completion of phase alignment).
7. Set the PLLRST bit in PLLCTL to 1 to bring the PLL out of reset.
8. Wait for PLL to lock. See the device-specific data manual for PLL lock time.
9. Set the PLLEN bit in PLLCTL to 1 to remove the PLL from bypass mode.

Copyright © 2013–2016, Texas Instruments Incorporated

7.2.2.2 Changing PLL Multiplier

If the PLL is not powered down (PLLPWRDN bit in PLLCTL is cleared to 0), perform the following procedure to change PLL0 multiplier.

1. Before changing the PLL frequency, switch to PLL bypass mode:
   (a) Clear the PLLENSRC bit in PLLCTL to 0 to allow PLLCTL.PLLEN to take effect.
   (b) Clear the PLLEN bit in PLLCTL to 0 (select PLL bypass mode).
   (c) Wait for 4 OSCIN cycles to ensure PLLC switches to bypass mode properly.
2. Clear the PLLRST bit in PLLCTL to 0 (reset PLL).
3. Program the required multiplier value in PLLM. If desired to scale all the SYSCLK frequencies of a given PLLC, program the POSTDIV ratio.
4. If necessary, program PLLDIVn registers to change the SYSCLKn divide values:
   (a) Program the RATIO field in PLLDIVn with the desired divide factors.
   (b) Set the GOSET bit in PLLCMD to 1 to initiate a new divider transition.
   (c) Wait for the GOSTAT bit in PLLSTAT to clear to 0 (completion of phase alignment).
5. Set the PLLRST bit in PLLCTL to 1 to bring the PLL out of reset.
6. Wait for PLL to lock. See the device-specific data manual for PLL lock time.
7. Set the PLLEN bit in PLLCTL to 1 to remove the PLL from bypass mode.

7.2.2.3 Changing SYSCLK Dividers

This section discusses the software sequence to change the SYSCLK dividers. The SYSCLK divider change sequence is also referred to as GO operation, as it involves hitting the GO bit (GOSET bit in PLLCMD) to initiate the divider change.

1. Check for the GOSTAT bit in PLLSTAT to clear to 0 to indicate that no GO operation is currently in progress.
2. Program the RATIO field in PLLDIVn with the desired divide factors.
3. Set the GOSET bit in PLLCMD to 1 to initiate a new divider transition.
4. Wait for the GOSTAT bit in PLLSTAT to clear to 0 (completion of divider change).

7.3 Locking/Unlocking PLL Register Access

A lock mechanism is present on the device that can prevent inadvertent reconfiguration of the PLLC registers. This primarily provides protection for the watchdog timer that runs on the AUXCLK output of PLL0. The PLL has a bit that is capable of disabling AUXCLK and therefore capable of stopping the watchdog timer.

To prevent this, when the PLL_MASTER_LOCK bit of the chip configuration 0 register (CFGCHIP0) in the System Configuration Module is set, writes to any PLLC registers are locked. The PLL_MASTER_LOCK bit is protected as type "Priv" and it is also protected by the Kick0 and Kick1 registers in the System Configuration Module. The master writing to the Kick0/Kick1/CFGCHIP0 registers needs to have appropriate privilege, and write the correct key values to the Kick0 and Kick 1 registers before writing to the PLLC registers. See the *System Configuration (SYSCFG) Module* chapter for information on privilege type and the Kick0 and Kick1 registers.

To modify the PLLC registers, use the following sequence:

1. Write the correct key values to Kick0 and Kick1 registers.
2. Clear the PLL_MASTER_LOCK bit in CFGCHIP0.
3. Configure the desired PLLC register values.
4. Write an incorrect key value to the Kick registers.

NOTE: The PLL_MASTER_LOCK bit in CFGCHIP0 defaults to unlocked after reset, so the above procedure is only required after the PLL_MASTER_LOCK bit has been locked (set to 1).

Copyright © 2013–2016, Texas Instruments Incorporated

7.4 PLLC Registers

Table 7-2 lists the memory-mapped registers for the PLLC.

Table 7-2. PLL Controller (PLLC) Registers

Address	Acronym	Register Description	Section
01C1 1000h	REVID	Revision Identification Register	Section 7.4.1
01C1 10E4h	RSTYPE	Reset Type Status Register	Section 7.4.2
01C1 1100h	PLLCTL	PLL Control Register	Section 7.4.3
01C1 1110h	PLLM	PLL Multiplier Control Register	Section 7.4.4
01C1 1114h	PREDIV	PLL Pre-Divider Control Register	Section 7.4.5
01C1 1118h	PLLDIV1	PLL Controller Divider 1 Register	Section 7.4.6
01C1 111Ch	PLLDIV2	PLL Controller Divider 2 Register	Section 7.4.7
01C1 1120h	PLLDIV3	PLL Controller Divider 3 Register	Section 7.4.8
01C1 1128h	POSTDIV	PLL Post-Divider Control Register	Section 7.4.13
01C1 1138h	PLLCMD	PLL Controller Command Register	Section 7.4.14
01C1 113Ch	PLLSTAT	PLL Controller Status Register	Section 7.4.15
01C1 1140h	ALNCTL	PLL Controller Clock Align Control Register	Section 7.4.16
01C1 1144h	DCHANGE	PLLDIV Ratio Change Status Register	Section 7.4.17
01C1 1148h	CKEN	Clock Enable Control Register	Section 7.4.18
01C1 114Ch	CKSTAT	Clock Status Register	Section 7.4.19
01C1 1150h	SYSTAT	SYSCLK Status Register	Section 7.4.20
01C1 1160h	PLLDIV4	PLL Controller Divider 4 Register	Section 7.4.9
01C1 1164h	PLLDIV5	PLL Controller Divider 5 Register	Section 7.4.10
01C1 1168h	PLLDIV6	PLL Controller Divider 6 Register	Section 7.4.11
01C1 116Ch	PLLDIV7	PLL Controller Divider 7 Register	Section 7.4.12
01C1 11F0h	EMUCNT0	Emulation Performance Counter 0 Register	Section 7.4.21
01C1 11F4h	EMUCNT1	Emulation Performance Counter 1 Register	Section 7.4.22

Copyright © 2013–2016, Texas Instruments Incorporated

7.4.1 Revision Identification Register (REVID)

The revision identification register (REVID) is shown in Figure 7-2 and described in Table 7-3.

Figure 7-2. Revision Identification Register (REVID)

31	0
REV	
R-4481 3C00h	

LEGEND: R = Read only; -*n* = value after reset

Table 7-3. Revision Identification Register (REVID) Field Descriptions

Bit	Field	Value	Description
31-0	REV	4481 3C00h	Peripheral revision ID.

7.4.2 Reset Type Status Register (RSTYPE)

The reset type status register (RSTYPE) is shown in Figure 7-3 and described in Table 7-4. RSTYPE latches the cause of the last reset. If multiple reset sources are asserted simultaneously, RSTYPE records the reset source that deasserts last. If multiple reset sources are asserted and deasserted simultaneously, RSTYPE latches the highest priority reset source.

Figure 7-3. Reset Type Status Register (RSTYPE)

31	16
Reserved	
R-0	

15	3	2	1	0
Reserved		PLLSWRST	XWRST	POR
R-0		R-0	R-0	R-0

LEGEND: R = Read only; -*n* = value after reset

Table 7-4. Reset Type Status Register (RSTYPE) Field Descriptions

Bit	Field	Value	Description
31-3	Reserved	0	Reserved
2	PLLSWRST		PLL software reset.
		0	PLL soft reset was not the last reset to occur.
		1	PLL soft was the last reset to occur.
1	XWRST		External warm reset.
		0	External warm reset was not the last reset to occur.
		1	External warm reset was the last reset to occur.
0	POR		Power on reset.
		0	Power On Reset (POR) was not the last reset to occur.
		1	Power On Reset (POR) was the last reset to occur.

Copyright © 2013–2016, Texas Instruments Incorporated

7.4.3 PLL Control Register (PLLCTL)

The PLL control register (PLLCTL) is shown in Figure 7-4 and described in Table 7-5.

Figure 7-4. PLL Control Register (PLLCTL)

31	16
Reserved	
R-0	

15	9	8	7	6	5	4	3	2	1	0
Reserved		CLKMODE	Reserved		PLLENSRC	Reserved	PLLRST	Rsvd	PLLPWRDN	PLLEN
R-0		R/W-0	R-1		R/W-1	R/W-1	R/W-0	R-0	R/W-1	R/W-0

LEGEND: R/W = Read/Write; R = Read only; *-n* = value after reset

Table 7-5. PLL Control Register (PLLCTL) Field Descriptions

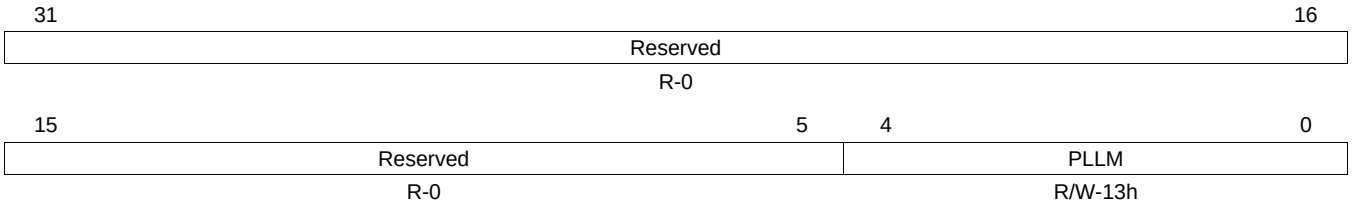
Bit	Field	Value	Description
31-9	Reserved	0	Reserved
8	CLKMODE		Reference Clock Selection
		0	Internal oscillator (crystal)
		1	Square wave
7-6	Reserved	1	Reserved
5	PLLENSRC	0	This bit must be cleared before PLLEN will have any effect.
4	Reserved	1	Reserved. Write the default value when modifying this register.
3	PLLRST		Asserts RESET to PLL if supported.
		0	PLL reset is asserted
		1	PLL reset is not asserted
2	Reserved	0	Reserved
1	PLLPWRDN		PLL power-down.
		0	PLL operation
		1	PLL power-down
0	PLLEN		PLL mode enables.
		0	Bypass mode
		1	PLL mode, not bypassed

Copyright © 2013–2016, Texas Instruments Incorporated

7.4.4 PLL Multiplier Control Register (PLLM)

The PLL multiplier control register (PLLM) is shown in Figure 7-5 and described in Table 7-6.

Figure 7-5. PLL Multiplier Control Register (PLLM)



LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

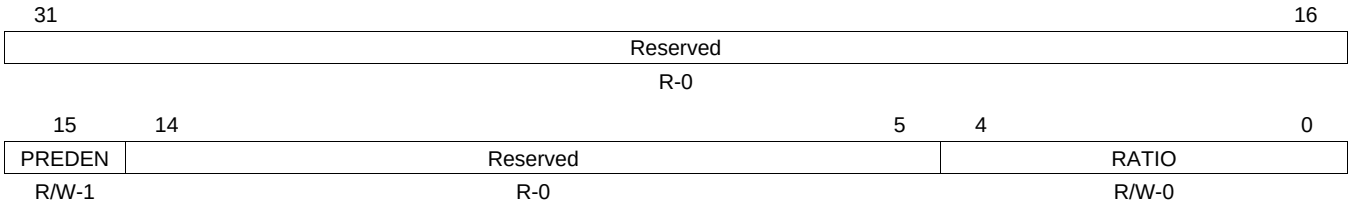
Table 7-6. PLL Multiplier Control Register (PLLM) Field Descriptions

Bit	Field	Value	Description
31-5	Reserved	0	Reserved
4-0	PLLM	0-1Fh	PLL Multiplier Select. Multiplier Value = PLLM + 1. The valid range of multiplier values for a given OSCIN is defined by the minimum and maximum frequency limits on the PLL VCO frequency. See the device-specific data manual for PLL VCO frequency specification limits.

7.4.5 PLL Pre-Divider Control Register (PREDIV)

The PLL pre-divider control register (PREDIV) is shown in Figure 7-6 and described in Table 7-7.

Figure 7-6. PLL Pre-Divider Control Register (PREDIV)



LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 7-7. PLL Pre-Divider Control Register (PREDIV) Field Descriptions

Bit	Field	Value	Description
31-14	Reserved	0	Reserved
15	PREDEN		Pre_Divider enable.
		0	Disable
		1	Enable
14-5	Reserved	0	Reserved
4-0	RATIO	0-1Fh	Divider ratio. Divider Value = RATIO + 1. RATIO defaults to 0 (PLL pre-divide by 1).

Copyright © 2013–2016, Texas Instruments Incorporated

7.4.6 PLL Controller Divider 1 Register (PLLDIV1)

The PLL controller divider 1 register (PLLDIV1) is shown in Figure 7-7 and described in Table 7-8. Divider 1 controls the divider for SYSCLK1.

Figure 7-7. PLL Controller Divider 1 Register (PLLDIV1)

31	16
Reserved	
R-0	

15	14	5	4	0
D1EN	Reserved		RATIO	
R/W-1	R-0		R/W-0	

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 7-8. PLL Controller Divider 1 Register (PLLDIV1) Field Descriptions

Bit	Field	Value	Description
31-16	Reserved	0	Reserved
15	D1EN		Divider Enable.
		0	Disable
		1	Enable
14-5	Reserved	0	Reserved
4-0	RATIO	0-1Fh	Divider ratio. Divider Value = RATIO + 1. RATIO defaults to 0 (PLL divide by 1).

7.4.7 PLL Controller Divider 2 Register (PLLDIV2)

The PLL controller divider 2 register (PLLDIV2) is shown in Figure 7-8 and described in Table 7-9. Divider 2 controls the divider for SYSCLK2.

Figure 7-8. PLL Controller Divider 2 Register (PLLDIV2)

31	16
Reserved	
R-0	

15	14	5	4	0
D2EN	Reserved		RATIO	
R/W-1	R-0		R/W-1	

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 7-9. PLL Controller Divider 2 Register (PLLDIV2) Field Descriptions

Bit	Field	Value	Description
31-16	Reserved	0	Reserved
15	D2EN		Divider Enable.
		0	Disable
		1	Enable
14-5	Reserved	0	Reserved
4-0	RATIO	0-1Fh	Divider ratio. Divider Value = RATIO + 1. RATIO defaults to 1 (PLL divide by 2).

Copyright © 2013–2016, Texas Instruments Incorporated

7.4.8 PLL Controller Divider 3 Register (PLLDIV3)

The PLL controller divider 3 register (PLLDIV3) is shown in Figure 7-9 and described in Table 7-10. Divider 3 controls the divider for SYSCLK3.

Figure 7-9. PLL Controller Divider 3 Register (PLLDIV3)

31	16
Reserved	
R-0	

15	14	5	4	0
D3EN	Reserved		RATIO	
R/W-1	R-0		R/W-2h	

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 7-10. PLL Controller Divider 3 Register (PLLDIV3) Field Descriptions

Bit	Field	Value	Description
31-16	Reserved	0	Reserved
15	D3EN		Divider Enable.
		0	Disable
		1	Enable
14-5	Reserved	0	Reserved
4-0	RATIO	0-1Fh	Divider ratio. Divider Value = RATIO + 1. RATIO defaults to 2h (PLL divide by 3).

7.4.9 PLL Controller Divider 4 Register (PLLDIV4)

The PLL controller divider 4 register (PLLDIV4) is shown inFigure 7-10 and described in Table 7-11. Divider 4 controls the divider for SYSCLK4.

Figure 7-10. PLL Controller Divider 4 Register (PLLDIV4)

31	16
Reserved	
R-0	

15	14	5	4	0
D4EN	Reserved		RATIO	
R/W-1	R-0		R/W-3h	

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 7-11. PLL Controller Divider 4 Register (PLLDIV4) Field Descriptions

Bit	Field	Value	Description
31-16	Reserved	0	Reserved
15	D4EN		Divider Enable.
		0	Disable
		1	Enable
14-5	Reserved	0	Reserved
4-0	RATIO	0-1Fh	Divider ratio. Divider Value = RATIO + 1. RATIO defaults 3 (PLL divide by 4).

Copyright © 2013–2016, Texas Instruments Incorporated

7.4.10 PLL Controller Divider 5 Register (PLLDIV5)

The PLL controller divider 5 register (PLLDIV5) is shown in Figure 7-11 and described in Table 7-12. Divider 5 controls the divider for SYSCLK5.

Figure 7-11. PLL Controller Divider 5 Register (PLLDIV5)

31	16
Reserved	
R-0	

15	14	5	4	0
D5EN	Reserved		RATIO	
R/W-1	R-0		R/W-2h	

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 7-12. PLL Controller Divider 5 Register (PLLDIV5) Field Descriptions

Bit	Field	Value	Description
31-16	Reserved	0	Reserved
15	D5EN		Divider Enable.
		0	Disable
		1	Enable
14-5	Reserved	0	Reserved
4-0	RATIO	0-1Fh	Divider ratio. Divider Value = RATIO + 1. RATIO defaults 2 (PLL divide by 3).

7.4.11 PLL Controller Divider 6 Register (PLLDIV6)

The PLL controller divider 6 register (PLLDIV6) is shown in Figure 7-12 and described in Table 7-13. Divider 6 controls the divider for SYSCLK6.

Figure 7-12. PLL Controller Divider 6 Register (PLLDIV6)

31	16
Reserved	
R-0	

15	14	5	4	0
D6EN	Reserved		RATIO	
R/W-1	R-0		R/W-0	

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 7-13. PLL Controller Divider 6 Register (PLLDIV6) Field Descriptions

Bit	Field	Value	Description
31-16	Reserved	0	Reserved
15	D6EN		Divider Enable.
		0	Disable
		1	Enable
14-5	Reserved	0	Reserved
4-0	RATIO	0-1Fh	Divider ratio. Divider Value = RATIO + 1. RATIO defaults to 0 (PLL divide by 1).

Copyright © 2013–2016, Texas Instruments Incorporated

7.4.12 PLL Controller Divider 7 Register (PLLDIV7)

The PLL controller divider 7 register (PLLDIV7) is shown in Figure 7-13 and described in Table 7-14. Divider 7 controls the divider for SYSCLK7.

Figure 7-13. PLL Controller Divider 7 Register (PLLDIV7)

31	16
Reserved	
R-0	

15	14	5	4	0
D7EN	Reserved		RATIO	
R/W-1	R-0		R/W-5h	

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 7-14. PLL Controller Divider 7 Register (PLLDIV7) Field Descriptions

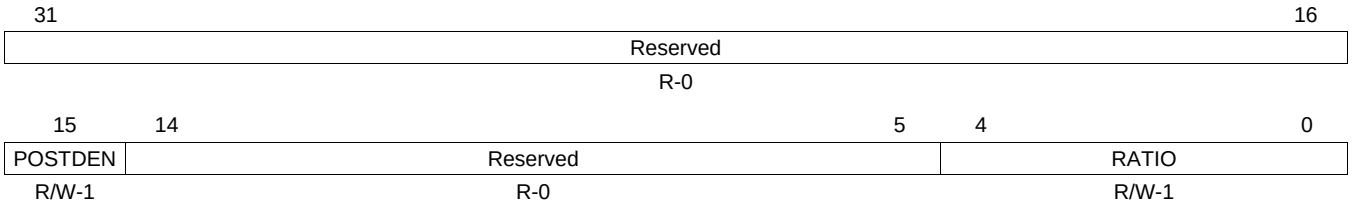
Bit	Field	Value	Description
31-16	Reserved	0	Reserved
15	D7EN		Divider Enable.
		0	Disable
		1	Enable
14-5	Reserved	0	Reserved
4-0	RATIO	0-1Fh	Divider ratio. Divider Value = RATIO + 1. RATIO defaults to 5 (PLL divide by 6).

Copyright © 2013–2016, Texas Instruments Incorporated

7.4.13 PLL Post-Divider Control Register (POSTDIV)

The PLL post-divider control register (POSTDIV) is shown in Figure 7-14 and described in Table 7-15.

Figure 7-14. PLL Post-Divider Control Register (POSTDIV)



LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

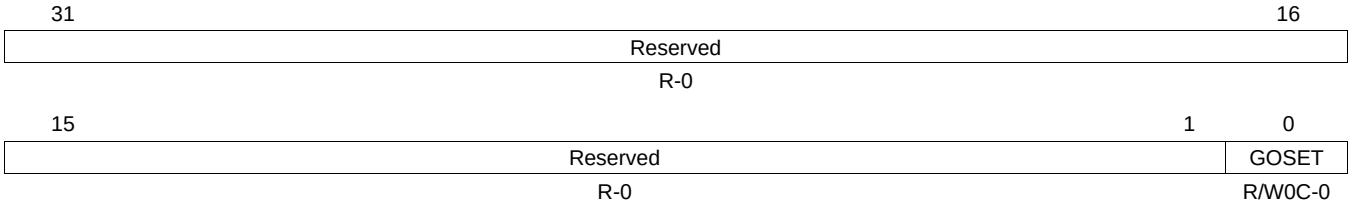
Table 7-15. PLL Post-Divider Control Register (POSTDIV) Field Descriptions

Bit	Field	Value	Description
31-16	Reserved	0	Reserved
15	POSTDEN		Post_Divider enable.
		0	Disable
		1	Enable
14-5	Reserved	0	Reserved
4-0	RATIO	0-1Fh	Divider ratio. Divider Value = RATIO + 1. RATIO defaults to 1 (PLL post-divide by 2).

7.4.14 PLL Controller Command Register (PLLCMD)

The PLL controller command register (PLLCMD) is shown in Figure 7-15 and described in Table 7-16. contains command bits for various operations. Writes of 1 initiate command; writes of 0 clear the bit, but have no effect.

Figure 7-15. PLL Controller Command Register (PLLCMD)



LEGEND: R/W = Read/Write; R = Read only; W0C = Write 0 to clear bit; -*n* = value after reset

Table 7-16. PLL Controller Command Register (PLLCMD) Field Descriptions

Bit	Field	Value	Description
31-1	Reserved	0	Reserved
0	GOSET		GO bit for SYSCLKx phase alignment.
		0	Clear bit (no effect)
		1	Phase alignment

Copyright © 2013–2016, Texas Instruments Incorporated

7.4.15 PLL Controller Status Register (PLLSTAT)

The PLL controller status register (PLLSTAT) is shown in Figure 7-16 and described in Table 7-17.

Figure 7-16. PLL Controller Status Register (PLLSTAT)

31	16
Reserved	
R-0	

15	3	2	1	0
Reserved		STABLE	Reserved	GOSTAT
R-0		R-0	R-0	R-0

LEGEND: R = Read only; -*n* = value after reset

Table 7-17. PLL Controller Status Register (PLLSTAT) Field Descriptions

Bit	Field	Value	Description
31-3	Reserved	0	Reserved
2	STABLE		OSC counter done, oscillator assumed to be stable. By the time the device comes out of reset, this bit should become 1.
		0	No
		1	Yes
1	Reserved	0	Reserved
0	GOSTAT		Status of GO operation. If 1, indicates GO operation is in progress.
		0	GO operation is not in progress.
		1	GO operation is in progress.

Copyright © 2013–2016, Texas Instruments Incorporated

7.4.16 PLL Controller Clock Align Control Register (ALNCTL)

The PLL controller clock align control register (ALNCTL) is shown in Figure 7-17 and described in Table 7-18. Indicates which SYSCLKs need to be aligned for proper device operation.

Figure 7-17. PLL Controller Clock Align Control Register (ALNCTL)

31	16
Reserved	
R-0	

15	7	6	5	4	3	2	1	0
Reserved		ALN7	ALN6	ALN5	ALN4	ALN3	ALN2	ALN1
R-0		R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 7-18. PLL Controller Clock Align Control Register (ALNCTL) Field Descriptions

Bit	Field	Value	Description
31-7	Reserved	0	Reserved
6	ALN7		SYSCLK7 needs to be aligned to others selected in this register.
		0	No
		1	Yes
5	ALN6		SYSCLK6 needs to be aligned to others selected in this register.
		0	No
		1	Yes
4	ALN5		SYSCLK5 needs to be aligned to others selected in this register.
		0	No
		1	Yes
3	ALN4		SYSCLK4 needs to be aligned to others selected in this register.
		0	No
		1	Yes
2	ALN3		SYSCLK3 needs to be aligned to others selected in this register.
		0	No
		1	Yes
1	ALN2		SYSCLK2 needs to be aligned to others selected in this register.
		0	No
		1	Yes
0	ALN1		SYSCLK1 needs to be aligned to others selected in this register.
		0	No
		1	Yes

Copyright © 2013–2016, Texas Instruments Incorporated

7.4.17 PLLDIV Ratio Change Status Register (DCHANGE)

The PLLDIV ratio change status register (DCHANGE) is shown in Figure 7-18 and described in Table 7-19. Indicates if SYSCLK divide ratio has been modified.

Figure 7-18. PLLDIV Ratio Change Status Register (DCHANGE)

31	16
Reserved	
R-0	

15	7	6	5	4	3	2	1	0
Reserved		SYS7	SYS6	SYS5	SYS4	SYS3	SYS2	SYS1
R-0		R-0	R-0	R-0	R-0	R-0	R-0	R-0

LEGEND: R = Read only; -*n* = value after reset

Table 7-19. PLLDIV Ratio Change Status Register (DCHANGE) Field Descriptions

Bit	Field	Value	Description
31-7	Reserved	0	Reserved
6	SYS7		SYSCLK7 divide ratio is modified.
		0	Ratio is not modified.
		1	Ratio is modified.
5	SYS6		SYSCLK6 divide ratio is modified.
		0	Ratio is not modified.
		1	Ratio is modified.
4	SYS5		SYSCLK5 divide ratio is modified.
		0	Ratio is not modified.
		1	Ratio is modified.
3	SYS4		SYSCLK4 divide ratio is modified.
		0	Ratio is not modified.
		1	Ratio is modified.
2	SYS3		SYSCLK3 divide ratio is modified.
		0	Ratio is not modified.
		1	Ratio is modified.
1	SYS2		SYSCLK2 divide ratio is modified.
		0	Ratio is not modified.
		1	Ratio is modified.
0	SYS1		SYSCLK1 divide ratio is modified.
		0	Ratio is not modified.
		1	Ratio is modified.

Copyright © 2013–2016, Texas Instruments Incorporated

7.4.18 Clock Enable Control Register (CKEN)

The clock enable control register (CKEN) is shown in Figure 7-19 and described in Table 7-20. Clock enable control for miscellaneous output clocks.

Figure 7-19. Clock Enable Control Register (CKEN)

31	16
Reserved	
R-0	

15	2	1	0
Reserved		Reserved	AUXEN
R-0		R/W-1	R/W-1

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 7-20. Clock Enable Control Register (CKEN) Field Descriptions

Bit	Field	Value	Description
31-2	Reserved	0	Reserved
1	Reserved	1	Reserved. Write the default value when modifying this register.
0	AUXEN		AUXCLK enable. Actual AUXCLK status is shown in the clock status register (CKSTAT).
		0	AUXCLK is disabled.
		1	AUXCLK is enabled.

Copyright © 2013–2016, Texas Instruments Incorporated

7.4.19 Clock Status Register (CKSTAT)

The clock status register (CKSTAT) is shown in Figure 7-20 and described in Table 7-21. Clock status for all clocks, except SYSCLK*n*.

Figure 7-20. Clock Status Register (CKSTAT)

31	16
Reserved	
R-0	

15	1	0
Reserved	Reserved	AUXEN
R-0	R-1	R-1

LEGEND: R = Read only; -*n* = value after reset

Table 7-21. Clock Status Register (CKSTAT) Field Descriptions

Bit	Field	Value	Description
31-2	Reserved	0	Reserved
1	Reserved	1	Reserved.
0	AUXEN		AUXCLK on status. AUXCLK is controlled by the AUXEN bit in the clock enable control register (CKEN).
		0	AUXCLK is off.
		1	AUXCLK is on.

Copyright © 2013–2016, Texas Instruments Incorporated

7.4.20 SYSCLK Status Register (SYSTAT)

The SYSCLK status register (SYSTAT) is shown in Figure 7-21 and described in Table 7-22. Indicates SYSCLK on/off status. Actual default is determined by actual clock on/off status, which depends on the D*n*EN bit in PLLDIV*n* default.

Figure 7-21. SYSCLK Status Register (SYSTAT)

31							8
Reserved							
R-0							
7	6	5	4	3	2	1	0
Reserved	SYS7ON	SYS6ON	SYS5ON	SYS4ON	SYS3ON	SYS2ON	SYS1ON
R-0	R-1	R-1	R-1	R-1	R-1	R-1	R-1

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 7-22. SYSCLK Status Register (SYSTAT) Field Descriptions

Bit	Field	Value	Description
31-7	Reserved	0	Reserved
6	SYS7ON		SYSCLK7 on status
		0	Off
		1	On
5	SYS6ON		SYSCLK6 on status
		0	Off
		1	On
4	SYS5ON		SYSCLK5 on status
		0	Off
		1	On
3	SYS4ON		SYSCLK4 on status
		0	Off
		1	On
2	SYS3ON		SYSCLK3 on status
		0	Off
		1	On
1	SYS2ON		SYSCLK2 on status
		0	Off
		1	On
0	SYS1ON		SYSCLK1 on status
		0	Off
		1	On

Copyright © 2013–2016, Texas Instruments Incorporated

7.4.21 Emulation Performance Counter 0 Register (EMUCNT0)

The emulation performance counter 0 register (EMUCNT0) is shown in Figure 7-22 and described in Table 7-23. EMUCNT0 is for emulation performance profiling. It counts in a divide-by-4 of the system clock. To start the counter, a write must be made to EMUCNT0. This register is not writable, but only used to start the register. After the register is started, it can not be stopped except for power on reset. When EMUCNT0 is read, it snapshots EMUCNT0 and EMUCNT1. The snapshot version is what is read. It is important to read the EMUCNT0 followed by EMUCNT1 or else the snapshot version may not get updated correctly.

Figure 7-22. Emulation Performance Counter 0 Register (EMUCNT0)

31	0
COUNT	
R-0	

LEGEND: R = Read only; -*n* = value after reset

Table 7-23. Emulation Performance Counter 0 Register (EMUCNT0) Field Descriptions

Bit	Field	Value	Description
31-0	COUNT	0-FFFF FFFFh	Counter value for lower 64-bits.

7.4.22 Emulation Performance Counter 1 Register (EMUCNT1)

The emulation performance counter 1 register (EMUCNT1) is shown in Figure 7-23 and described in Table 7-24. EMUCNT1 is for emulation performance profiling. To start the counter, a write must be made to EMUCNT0. This register is not writable, but only used to start the register. After the register is started, it can not be stopped except for power on reset. When EMUCNT0 is read, it snapshots EMUCNT0 and EMUCNT1. The snapshot version is what is read. It is important to read the EMUCNT0 followed by EMUCNT1 or else the snapshot version may not get updated correctly.

Figure 7-23. Emulation Performance Counter 1 Register (EMUCNT1)

31	0
COUNT	
R-0	

LEGEND: R = Read only; -*n* = value after reset

Table 7-24. Emulation Performance Counter 1 Register (EMUCNT1) Field Descriptions

Bit	Field	Value	Description
31-0	COUNT	0-FFFF FFFFh	Counter value for upper 64-bits.

Copyright © 2013–2016, Texas Instruments Incorporated

Power and Sleep Controller (PSC)

Topic Page

8.1 Introduction 97
8.2 Power Domain and Module Topology 97
8.3 Executing State Transitions 101
8.4 IcePick Emulation Support in the PSC 102
8.5 PSC Interrupts 102
8.6 PSC Registers 105

Copyright © 2013–2016, Texas Instruments Incorporated

8.1 Introduction

The Power and Sleep Controllers (PSC) are responsible for managing transitions of system power on/off, clock on/off, resets (device level and module level). It is used primarily to provide granular power control for on chip modules (peripherals and CPU). A PSC module consists of a Global PSC (GPSC) and a set of Local PSCs (LPSCs). The GPSC contains memory mapped registers, PSC interrupts, a state machine for each peripheral/module it controls. An LPSC is associated with every module that is controlled by the PSC and provides clock and reset control. Many of the operations of the PSC are transparent to user (software), such as power on and reset control. However, the PSC module(s) also provide you with interface to control several important power, clock and reset operations. The module level power, clock and reset operations managed and controlled by the PSC are the focus of this chapter.

The PSC includes the following features:

- Manages chip power-on/off
- Provides a software interface to:
  - Control module clock enable/disable
  - Control module reset
  - Control CPU local reset
- Manages on-chip RAM sleep modes (for DSP memories and L3 RAM)
- Supports IcePick emulation features: power, clock and reset

8.2 Power Domain and Module Topology

This device includes two PSC modules. Each PSC module consists of an Always On power domain and an additional pseudo/internal power domain that manages the sleep modes for the RAMs present in the DSP subsystem and the L3 RAM , respectively.

Each PSC module controls clock states for several on the on chip modules, controllers and interconnect components. Table 8-1 and Table 8-2 lists the set of peripherals/modules that are controlled by the PSC, the power domain they are associated with, the LPSC assignment and the default (power-on reset) module states. See the device-specific data manual for the peripherals available on a given device. The module states and terminology are defined in Section 8.2.2.

Even though there are 2 PSC modules with 2 power domains each on the device, both PSC modules and all the power domains are powered by the CVDD pins of the device. All power domains are on when the chip is powered on. There is no provision to remove power externally for the non Always On domains, that is, the pseudo/internal power domains.

There are a few modules/peripherals on the device that do not have a LPSC assigned to them. These modules do not have their module reset/clocks controlled by the PSC module. The decision to assign an LPSC to a module on a device is primarily based on whether or not disabling the clocks to a module will result in significant power savings. This typically depends on the size and the frequency of operation of the module.

Table 8-1. PSC0 Default Module Configuration

LPSC Number	Module Name	Power Domain	Default Module State	Auto Sleep/Wake Only
0	EDMA3 Channel Controller	AlwaysON (PD0)	SwRstDisable	—
1	EDMA3 Transfer Controller 0	AlwaysON (PD0)	SwRstDisable	—
2	EDMA3 Transfer Controller 1	AlwaysON (PD0)	SwRstDisable	—
3	Not Used	—	—	—
4	SPI0	AlwaysON (PD0)	SwRstDisable	—
5-8	Not Used	—	—	—
9	UART0	AlwaysON (PD0)	SwRstDisable	—
10	Not Used	—	—	—
11	SCR1 (BR4)	AlwaysON (PD0)	Enable	Yes
12	SCR2 (BR3, BR5, BR6)	AlwaysON (PD0)	Enable	Yes

Copyright © 2013–2016, Texas Instruments Incorporated

Table 8-1. PSC0 Default Module Configuration (continued)

LPSC Number	Module Name	Power Domain	Default Module State	Auto Sleep/Wake Only
13	dMAX	AlwaysON (PD0)	SwRstDisable	—
14	Not Used	—	—	—
15	DSP	PD_DSP (PD1)	Enable	—

Table 8-2. PSC1 Default Module Configuration

LPSC Number	Module Name	Power Domain	Default Module State	Auto Sleep/Wake Only
0-2	Not Used	—	—	—
3	GPIO	AlwaysON (PD0)	SwRstDisable	—
4-6	Not Used	—	—	—
7	McASP0 (+ McASP0 FIFO)	AlwaysON (PD0)	SwRstDisable	—
8	McASP1 (+ McASP1 FIFO)	AlwaysON (PD0)	SwRstDisable	—
9	McASP2 (+ McASP2 FIFO)	AlwaysON (PD0)	SwRstDisable	—
10	SPI1	AlwaysON (PD0)	SwRstDisable	—
11	I2C1	AlwaysON (PD0)	SwRstDisable	—
12	UART1	AlwaysON (PD0)	SwRstDisable	—
13	UART2	AlwaysON (PD0)	SwRstDisable	—
14-16	Not Used	—	—	—
17	eHRPWM0	AlwaysON (PD0)	SwRstDisable	—
18-19	Not Used	—	—	—
20	eCAP0/1/2	AlwaysON (PD0)	SwRstDisable	—
21	eQEP0	AlwaysON (PD0)	SwRstDisable	—
22-25	Not Used	—	—	—
26	SCR12 (BR18)	AlwaysON (PD0)	Enable	Yes
27-30	Not Used	—	—	—
31	Shared RAM (BR13)	PD_SHRAM	Enable	Yes

Copyright © 2013–2016, Texas Instruments Incorporated

8.2.1 Power Domain States

A power domain can only be in one of the two states: ON or OFF, defined as follows:

- ON: power to the domain is on
- OFF: power to the domain is off

In this device, for both PSC0 and PSC1, the Always ON domain (or PD0 power domain), is always in the ON state when the chip is powered-on. This domain is not programmable to OFF state (See details on PDCTL register).

Additionally, for both PSC0 and PSC1, the PD1 power domains, the internal/pseudo power domain can either be in the ON state or OFF state. Furthermore, for these power domains the transition from ON to OFF state is further qualified by the PSC0/1.PDCTL1.PDMODE settings. The PDCTL1.PDMODE settings determines the various sleep mode for the on-chip RAM associated with module in the PD1 domain.

- On PSC0 PD1/PD_DSP Domain: Controls the sleep state for DSP L1 and L2 Memories
- On PSC1 PD1/PD_SHRAM Domain: Controls the sleep state for the 128 kB Shared RAM

NOTE: Currently programming the PD1 power domain state to OFF is not supported. You should leave both the PDCTL1.NEXT and PDCTL1.PDMODE values at default/power on reset values.

Both PD0 and PD1 power domains in PSC0 and PSC1 are powered by the CVDD pins of the device. There is no capability to individually remove voltage/power from the DSP or Shared RAM power domains.

8.2.2 Module States

The PSC defines several possible states for a module. This various states are essentially a combination of the module reset asserted or de-asserted and module clock on/enabled or off/disabled. The various module states are defined in Table 8-3.

The key difference between the Auto Sleep and Auto Wake states is that once the module is configured in Auto Sleep mode, it will transition back to the clock disabled state (automatically sleep) after servicing the internal read/write access request where as in Auto Wake mode, on receiving the first internal read/write access request, the module will permanently transition from the clock disabled to clock enabled state (automatically wake).

When the module state is programmed to Disable, SwRstDisable, Auto Sleep or Auto Wake modes, where in the module clocks are off/disabled, an external event or I/O request cannot enable the clocks. For the module to appropriately respond to such external request, it would need to be reconfigured to the Enable state.

8.2.2.1 Auto Sleep/Wake Only Configurations and Limitation

NOTE: Currently no modules should be configured in Auto Sleep or Auto Wake modes. If the module clocks need to gated/disabled for power savings, you should program the module state to Disable. For Auto Sleep/Auto Wake Only modules, disabling the clock is not supported and they should be kept in their default “Enable” state.

Table 8-1 and Table 8-2 each have a column to indicate whether or not the LPSC configuration for a module is Auto Sleep/Wake Only. Modules that have a “Yes” marked for the Auto Sleep/Wake Only column can be programmed in software to be in Enable, Auto Sleep and Auto Wake states only; that is, if the software tries to program these modules to Disable, SyncReset, or SwRstDisable state the power sleep controller ignores these transition requests and transitions the module state to Enable.

Copyright © 2013–2016, Texas Instruments Incorporated

Table 8-3. Module States

Module State	Module Reset	Module Clock	Module State Definition
Enable	De-asserted	On	A module in the enable state has its module reset de-asserted and it has its clock on. This is the normal operational state for a given module
Disable	De-asserted	Off	A module in the disabled state has its module reset de-asserted and it has its module clock off. This state is typically used for disabling a module clock to save power. This device is designed in full static CMOS, so when you stop a module clock, it retains the module's state. When the clock is restarted, the module resumes operating from the stopping point.
SyncReset	Asserted	On	A module state in the SyncReset state has its module reset asserted and it has its clock on. Generally, software is not expected to initiate this state
SwRstDisable	Asserted	Off	A module in the SwResetDisable state has its module reset asserted and it has its clock disabled. After initial power-on, several modules come up in the SwRstDisable state. Generally, software is not expected to initiate this state
Auto Sleep	De-asserted	Off	A module in the Auto Sleep state also has its module reset de-asserted and its module clock disabled, similar to the Disable state. However this is a special state, once a module is configured in this state by software, it can "automatically" transition to "Enable" state whenever there is an internal read/write request made to it, and after servicing the request it will "automatically" transition into the sleep state (with module reset re de-asserted and module clock disabled), without any software intervention. The transition from sleep to enabled and back to sleep state has some cycle latency associated with it. It is not envisioned to use this mode when peripherals are fully operational and moving data. See Section 8.2.2.1 for additional considerations, constraints, limitations around this mode.
Auto Wake	De-asserted	Off	A module in the Auto Wake state also has its module reset de-asserted and its module clock disabled, similar to the Disable state. However this is a special state, once a module is configured in this state by software, it will "automatically" transition to "Enable" state whenever there is an internal read/write request made to it, and will remain in the "Enabled" state from then on (with module reset re de-asserted and module clock on), without any software intervention. The transition from sleep to enabled state has some cycle latency associated with it. It is not envisioned to use this mode when peripherals are fully operational and moving data. See Section 8.2.2.1 for additional considerations, constraints, limitations around this mode.

8.2.2.2 Local Reset

In addition to module reset, some modules can be reset using a special local reset that is also a part of the PSC module control for resets. The modules that support the local reset are:

- DSP: When the DSP local reset is asserted the DSP internal memories (L1P, L1D and L2) are still accessible. The local reset only resets the DSP CPU core, not the rest of DSP subsystem, as the DSP module reset would. Local Reset is useful in cases where the DSP is in enable or disable state; since when module is in SyncReset or SwRstDisable state the module reset is asserted, and the module reset takes precedence over the local reset.

The procedures for asserting and de-asserting the local reset are as follows (where *n* corresponds to the module that supports local reset):

1. Clear the LRST bit in the module control register (MDCTL*n*) to 0 to assert the module's local reset.
2. Set the LRST bit in the module control register (MDCTL*n*) to 1 to de-assert module's local reset.

If the CPU is in the enable state, it immediately executes program instructions after reset is de-asserted.

Copyright © 2013–2016, Texas Instruments Incorporated

8.3 Executing State Transitions

This section describes how to execute the state transitions modules.

8.3.1 Power Domain State Transitions

This device consists of 2 types of domain (in each PSC controller): the Always On Domain(s) and the pseudo/RAM power domain(s). The Always On power domains are always in the ON state when the chip is powered on. You are not allowed to change the power domain state to OFF.

The pseudo/RAM power domains allow internally powering down the state of the RAMs associated with these domains (L1/L2 for PD_DSP in PSC0 and Shared RAM for PD_SHRAM in PSC1) so that these RAMs can run in lower power sleep modes via the power sleep controller.

NOTE: Currently powering down the RAMs via the pseudo/RAM power domain is not supported; therefore, these domains and the RAM should be left in their default power on state.

As mentioned in Section 8.2, the pseudo/RAM power domains are powered down internally, and in this context powering down does not imply removing the core voltage from pins externally.

8.3.2 Module State Transitions

This section describes the procedure for transitioning the module state (clock and reset control). Note that some peripherals have special programming requirements and additional recommended steps you must take before you can invoke the PSC module state transition. See the individual peripheral user guides for more details. For example, the external memory controller requires that you first place the SDRAM memory in self-refresh mode before you invoke the PSC module state transitions, if you want to maintain the memory contents.

The following procedure is directly applicable for all modules that are controlled via the PSC (shown in Table 8-1 and Table 8-2), except for the core(s). To transition the DSP module state, there are additional system considerations and constraints that you should be aware of. These system considerations and the procedure for transitioning the DSP module state are described in details in the *Power Management* chapter.

NOTE: In the following procedure, x is 0 for modules in PD0 (Power Domain 0 or Always On domain) and x is 1 for modules in PD1 (Power Domain 1) . See Table 8-1 and Table 8-2 for power domain associations.

The procedure for module state transitions is:

1. Wait for the GOSTAT[x] bit in PTSTAT to clear to 0. You must wait for any previously initiated transitions to finish before initiating a new transition.
2. Set the NEXT bit in MDCTL*n* to SwRstDisable (0), SyncReset (1), Disable (2h), Enable (3h), Auto Sleep (4h) or Auto Wake (5h).

   NOTE: You may set transitions in multiple NEXT bits in MDCTL*n* in this step. Transitions do not actually take place until you set the GO[x] bit in PTCMD in a later step.
3. Set the GO[x] bit in PTCMD to 1 to initiate the transition(s).
4. Wait for the GOSTAT[x] bit in PTSTAT to clear to 0. The modules are safely in the new states only after the GOSTAT[x] bit in PTSTAT is cleared to 0.

Copyright © 2013–2016, Texas Instruments Incorporated

8.4 IcePick Emulation Support in the PSC

The PSC supports IcePick commands that allow IcePick emulation tools to have some control over the state of power domains and modules. This IcePick support only applies to the following modules:

- DSP [MDCTL15]

In particular, Table 8-4 shows IcePick emulation commands recognized by the PSC.

Table 8-4. IcePick Emulation Commands

Power On and Enable Features	Power On and Enable Descriptions	Reset Features	Reset Descriptions
Inhibit Sleep	Allows emulation to prevent software from transitioning the module out of the enable state.	Assert Reset	Allows emulation to assert the module's local reset.
Force Power	Allows emulation to force the power domain into an on state. Not applicable as AlwaysOn power domain is always on.	Wait Reset	Allows emulation to keep local reset asserted for an extended period of time after software initiates local reset de-assert.
Force Active	Allows emulation to force the module into the enable state.	Block Reset	Allows emulation to block software initiated local and module resets.

NOTE: When emulation tools remove the above commands, the PSC immediately executes a state transition based on the current values in the NEXT bit in PDCTL0 and the NEXT bit in MDCTL*n*, as set by software.

8.5 PSC Interrupts

The PSC has an interrupt that is tied to the core interrupt controller. This interrupt is named PSCINT in the interrupt map. The PSC interrupt is generated when certain IcePick emulation events occur.

8.5.1 Interrupt Events

The PSC interrupt is generated when any of the following events occur:

- Power Domain Emulation Event (applies to pseudo/RAM power domain only)
- Module State Emulation event
- Module Local Reset Emulation event

These interrupt events are summarized in Table 8-5 and described in more detail in this section.

Table 8-5. PSC Interrupt Events

Interrupt Enable Bits		
Control Register	Enable Bit	Interrupt Condition
PDCTL*n*	EMUIHBIE	Interrupt occurs when the emulation alters the power domain state
MDCTL*n*	EMUIHBIE	Interrupt occurs when the emulation alters the module state
MDCTL*n*	EMURSTIE	Interrupt occurs when the emulation tries to alter the module's local reset

The PSC interrupt events only apply when IcePick emulation alters the state of the module from the user-programmed state in the NEXT bit in the MDCTL/PDCTL registers. IcePick support only applies to the modules listed in Section 8.4; therefore, the PSC interrupt conditions only apply to those modules listed.

Copyright © 2013–2016, Texas Instruments Incorporated

8.5.1.1 Power Domain Emulation Events

A power domain emulation event occurs when emulation alters the state of a power domain (does not apply to the Always On domain). Status is reflected in the EMUIHB bit in PDSTAT*n*. In particular, a power domain emulation event occurs under the following conditions:

- When inhibit sleep is asserted by emulation and software attempts to transition the module out of the on state
- When force power is asserted by emulation and power domain is not already in the on state
- When force active is asserted by emulation and power domain is not already in the on state

NOTE: Putting the pseudo/RAM power domain associated with DSP (PD_DSP) to off state is currently **not** supported.

8.5.1.2 Module State Emulation Events

A module state emulation event occurs when emulation alters the state of a module. Status is reflected in the EMUIHB bit in the module status register (MDSTAT*n*). In particular, a module state emulation event occurs under the following conditions:

- When inhibit sleep is asserted by emulation and software attempts to transition the module out of the enable state
- When force active is asserted by emulation and module is not already in the enable state

8.5.1.3 Local Reset Emulation Events

A local reset emulation event occurs when emulation alters the local reset of a module. Status is reflected in the EMURST bit in the module status register (MDSTAT*n*). In particular, a module local reset emulation event occurs under the following conditions:

- When assert reset is asserted by emulation although software de-asserted the local reset
- When wait reset is asserted by emulation
- When block reset is asserted by emulation and software attempts to change the state of local reset

8.5.2 Interrupt Registers

The PSC interrupt enable bits are: the EMUIHBIE bit in PDCTL1 (PSC0), the EMUIHBIE and the EMURSTIE bits in MDCTL*n* (where n is the modules that have IcePick emulation support, as specified in Section 8.4).

NOTE: To interrupt the CPU, the power sleep controller interrupt (PSC0_ALLINT and PSC1_ALLINT) must also be enabled in the DSP interrupt controller. For details on the DSP interrupt controller, see the *DSP Subsystem* chapter.

The PSC interrupt status bits are:

- For DSP:
  - The M[15] bit in the module error pending register 0 (MERRPR0) in PSC0 module.
  - The EMUIHB and the EMURST bits in the module status register for DSP (MDSTAT15).
  - The P[1] bit in the power error pending register (PERRPR) for the pseudo/RAM power domain associated with DSP memories.

The status bit in MERRPR0 and PERRPR registers is read by software to determine which module or power domain has generated an emulation interrupt and then software can read the corresponding status bits in MDSTAT register or the PDSTATn (PDCTL1 for pseudo/RAM power domain in PSC0) to determine which event caused the interrupt.

The PSC interrupt can be cleared by writing to bit corresponding to the module number in the module error clear register (MERRCR0), or the bit corresponding to the power domain number in the power error clear register (PERRCR) in PSC0 module.

Copyright © 2013–2016, Texas Instruments Incorporated

The PSC interrupt evaluation bit is the ALLEV bit in the INTEVAL register. When set, this bit forces the PSC interrupt logic to re-evaluate event status. If any events are still active (if any status bits are set) when the ALLEV bit in the INTEVAL is set to 1, the PSC interrupt is re-asserted to the interrupt controller. Set the ALLEV bit in the INTEVAL before exiting your PSC interrupt service routine to ensure that you do not miss any PSC interrupts.

See Section 8.6 for a description of the PSC registers.

8.5.3 Interrupt Handling

Handle the PSC interrupts as described in the following procedure:

First, enable the interrupt:

1. Set the EMUIHBIE bit in PDCTL*n*, the EMUIHBIE and the EMURSTIE bits in MDCTL*n* to enable the interrupt events that you want.

NOTE: The PSC interrupt is sent to the device interrupt controller when at least one enabled event becomes active.

2. Enable the power sleep controller interrupt (PSC*n*_ALLINT) in the device interrupt controller. To interrupt the CPU, PSC*n*_ALLINT must be enabled in the device interrupt controller. See the *DSP Subsystem* chapter for more information on interrupts.

The CPU enters the interrupt service routine (ISR) when it receives the interrupt.

1. Read the P[n] bit in PERRPR, and/or the M[n] bit in MERRPR0, the M[n] bit in MERRPR1, to determine the source of the interrupt(s).
2. For each active event that you want to service:
   (a) Read the event status bits in PDSTAT*n* and MDSTAT*n*, depending on the status bits read in the previous step to determine the event that caused the interrupt.
   (b) Service the interrupt as required by your application.
   (c) Write the M[n] bit in MERRCR*n* and the P[n] bit in PERRCR to clear corresponding status.
   (d) Set the ALLEV bit in INTEVAL. Setting this bit reasserts the PSC interrupt to the device interrupt controller, if there are still any active interrupt events.

Copyright © 2013–2016, Texas Instruments Incorporated

8.6 PSC Registers

Table 8-6 lists the memory-mapped registers for the PSC0 and Table 8-7 lists the memory-mapped registers for the PSC1.

Table 8-6. Power and Sleep Controller 0 (PSC0) Registers

Address	Acronym	Register Description	Section
01C1 0000h	REVID	Revision Identification Register	Section 8.6.1
01C1 0018h	INTEVAL	Interrupt Evaluation Register	Section 8.6.2
01C1 0040h	MERRPR0	Module Error Pending Register 0 (module 0-15)	Section 8.6.3
01C1 0050h	MERRCR0	Module Error Clear Register 0 (module 0-15)	Section 8.6.5
01C1 0060h	PERRPR	Power Error Pending Register	Section 8.6.7
01C1 0068h	PERRCR	Power Error Clear Register	Section 8.6.8
01C1 0120h	PTCMD	Power Domain Transition Command Register	Section 8.6.9
01C1 0128h	PTSTAT	Power Domain Transition Status Register	Section 8.6.10
01C1 0200h	PDSTAT0	Power Domain 0 Status Register	Section 8.6.11
01C1 0204h	PDSTAT1	Power Domain 1 Status Register	Section 8.6.12
01C1 0300h	PDCTL0	Power Domain 0 Control Register	Section 8.6.13
01C1 0304h	PDCTL1	Power Domain 1 Control Register	Section 8.6.14
01C1 0400h	PDCFG0	Power Domain 0 Configuration Register	Section 8.6.15
01C1 0404h	PDCFG1	Power Domain 1 Configuration Register	Section 8.6.16
01C1 0800h-01C1 083Ch	MDSTAT0-MDSTAT15	Module Status *n* Register (modules 0-15)	Section 8.6.17
01C1 0A00h-01C1 0A3Ch	MDCTL0-MDCTL15	Module Control *n* Register (modules 0-15)	Section 8.6.18

Table 8-7. Power and Sleep Controller 1 (PSC1) Registers

Address	Acronym	Register Description	Section
01E2 7000h	REVID	Revision Identification Register	Section 8.6.1
01E2 7018h	INTEVAL	Interrupt Evaluation Register	Section 8.6.2
01E2 7040h	MERRPR0	Module Error Pending Register 0 (module 0-31)	Section 8.6.4
01E2 7050h	MERRCR0	Module Error Clear Register 0 (module 0-31)	Section 8.6.6
01E2 7060h	PERRPR	Power Error Pending Register	Section 8.6.7
01E2 7068h	PERRCR	Power Error Clear Register	Section 8.6.8
01E2 7120h	PTCMD	Power Domain Transition Command Register	Section 8.6.9
01E2 7128h	PTSTAT	Power Domain Transition Status Register	Section 8.6.10
01E2 7200h	PDSTAT0	Power Domain 0 Status Register	Section 8.6.11
01E2 7204h	PDSTAT1	Power Domain 1 Status Register	Section 8.6.12
01E2 7300h	PDCTL0	Power Domain 0 Control Register	Section 8.6.13
01E2 7304h	PDCTL1	Power Domain 1 Control Register	Section 8.6.14
01E2 7400h	PDCFG0	Power Domain 0 Configuration Register	Section 8.6.15
01E2 7404h	PDCFG1	Power Domain 1 Configuration Register	Section 8.6.16
01E2 7800h-01E2 787Ch	MDSTAT0-MDSTAT31	Module Status *n* Register (modules 0-31)	Section 8.6.17
01E2 7A00h-01E2 7A7Ch	MDCTL0-MDCTL31	Module Control *n* Register (modules 0-31)	Section 8.6.19

Copyright © 2013–2016, Texas Instruments Incorporated

8.6.1 Revision Identification Register (REVID)

The revision identification register (REVID) is shown in Figure 8-1 and described in Table 8-8.

Figure 8-1. Revision Identification Register (REVID)

31	0
REV	
R-4482 3A00h	

LEGEND: R = Read only; -*n* = value after reset

Table 8-8. Revision Identification Register (REVID) Field Descriptions

Bit	Field	Value	Description
31-0	REV	4482 3A00h	Peripheral revision ID.

8.6.2 Interrupt Evaluation Register (INTEVAL)

The interrupt evaluation register (INTEVAL) is shown in Figure 8-2 and described in Table 8-9.

Figure 8-2. Interrupt Evaluation Register (INTEVAL)

31	16
Reserved	
R-0	

15	1	0
Reserved		ALLEV
R-0		W-0

LEGEND: R = Read only; W= Write only; -*n* = value after reset

Table 8-9. Interrupt Evaluation Register (INTEVAL) Field Descriptions

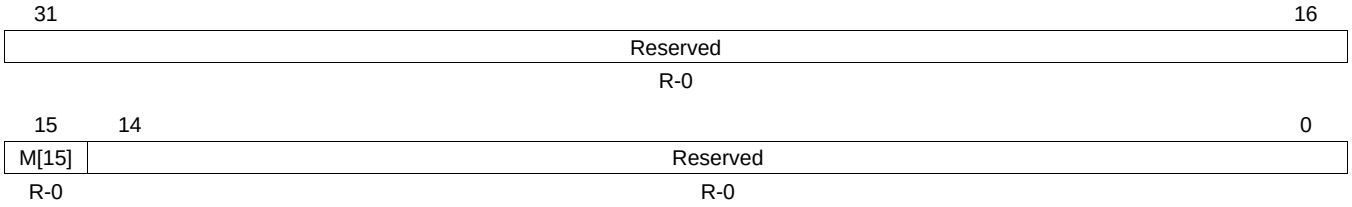
Bit	Field	Value	Description
31-1	Reserved	0	Reserved
0	ALLEV		Evaluate PSC interrupt (PSC*n*_ALLINT).
		0	A write of 0 has no effect.
		1	A write of 1 re-evaluates the interrupt condition.

Copyright © 2013–2016, Texas Instruments Incorporated

8.6.3 PSC0 Module Error Pending Register 0 (modules 0-15) (MERRPR0)

The PSC0 module error pending register 0 (MERRPR0) is shown in Figure 8-3 and described in Table 8-10.

Figure 8-3. PSC0 Module Error Pending Register 0 (MERRPR0)



LEGEND: R = Read only; -*n* = value after reset

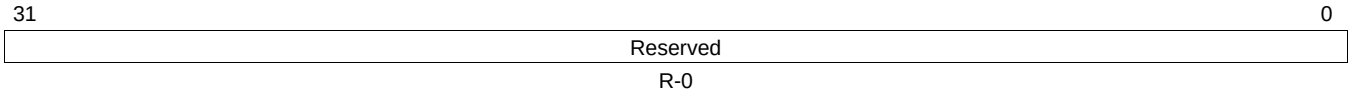
Table 8-10. PSC0 Module Error Pending Register 0 (MERRPR0) Field Descriptions

Bit	Field	Value	Description
31-16	Reserved	0	Reserved
15	M[15]		Module interrupt status bit for module 15 (DSP).
		0	Module 15 does not have an error condition.
		1	Module 15 has an error condition. See the module status 15 register (MDSTAT15) for the error condition.
14-0	Reserved	0	Reserved

8.6.4 PSC1 Module Error Pending Register 0 (modules 0-31) (MERRPR0)

The PSC1 module error pending register 0 (MERRPR0) is shown in Figure 8-4.

Figure 8-4. PSC1 Module Error Pending Register 0 (MERRPR0)



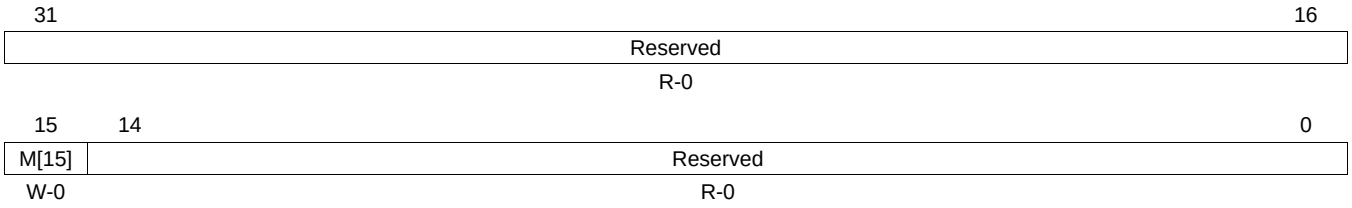
LEGEND: R = Read only; -*n* = value after reset

Copyright © 2013–2016, Texas Instruments Incorporated

8.6.5 PSC0 Module Error Clear Register 0 (modules 0-15) (MERRCR0)

The PSC0 module error clear register 0 (MERRCR0) is shown in Figure 8-5 and described in Table 8-11.

Figure 8-5. PSC0 Module Error Clear Register 0 (MERRCR0)



LEGEND: R = Read only; W = Write only; -*n* = value after reset

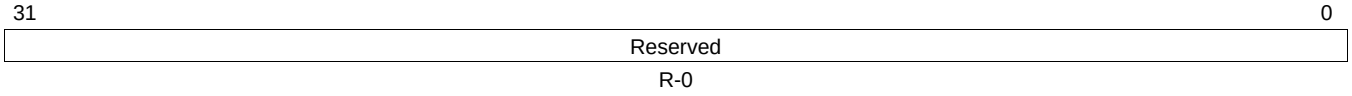
Table 8-11. PSC0 Module Error Clear Register 0 (MERRCR0) Field Descriptions

Bit	Field	Value	Description
31-16	Reserved	0	Reserved
15	M[15]		Clears the interrupt status bit (M[15]) set in the PSC0 module error pending register 0 (MERRPR0) and the interrupt status bits set in the module status 15 register (MDSTAT15).
		0	A write of 0 has no effect.
		1	A write of 1 clears the M[15] bit in MERRPR0 and the EMUIHB and EMURST bits in MDSTAT15.
14-0	Reserved	0	Reserved

8.6.6 PSC1 Module Error Clear Register 0 (modules 0-31) (MERRCR0)

The PSC1 module error clear register 0 (MERRCR0) is shown in Figure 8-6.

Figure 8-6. PSC1 Module Error Clear Register 0 (MERRCR0)



LEGEND: R = Read only; -*n* = value after reset

Copyright © 2013–2016, Texas Instruments Incorporated

8.6.7 Power Error Pending Register (PERRPR)

The power error pending register (PERRPR) is shown in Figure 8-7 and described in Table 8-12.

Figure 8-7. Power Error Pending Register (PERRPR)

31	16
Reserved	
R-0	

15	2	1	0
Reserved		P[1]	Rsvd
R-0		R-0	R-0

LEGEND: R = Read only; -*n* = value after reset

Table 8-12. Power Error Pending Register (PERRPR) Field Descriptions

Bit	Field	Value	Description
31-2	Reserved	0	Reserved
1	P[1]		RAM/Pseudo (PD1) power domain interrupt status.
		0	RAM/Pseudo power domain does not have an error condition.
		1	RAM/Pseudo power domain has an error condition. See the power domain 1 status register (PDSTAT1) for the error condition.
0	Reserved	0	Reserved

8.6.8 Power Error Clear Register (PERRCR)

The power error clear register (PERRCR) is shown in Figure 8-8 and described in Table 8-13.

Figure 8-8. Power Error Clear Register (PERRCR)

31	16
Reserved	
R-0	

15	2	1	0
Reserved		P[1]	Rsvd
R-0		W-0	R-0

LEGEND: R = Read only; W = Write only; -*n* = value after reset

Table 8-13. Power Error Clear Register (PERRCR) Field Descriptions

Bit	Field	Value	Description
31-2	Reserved	0	Reserved
1	P[1]		Clears the interrupt status bit (P) set in the power error pending register (PERRPR) and the interrupt status bits set in the power domain 1 status register (PDSTAT1).
		0	A write of 0 has no effect.
		1	A write of 1 clears the P bit in PERRPR and the interrupt status bits in PDSTAT1.
0	Reserved	0	Reserved

Copyright © 2013–2016, Texas Instruments Incorporated

8.6.9 Power Domain Transition Command Register (PTCMD)

The power domain transition command register (PTCMD) is shown in Figure 8-9 and described in Table 8-14.

Figure 8-9. Power Domain Transition Command Register (PTCMD)

31	16
Reserved	
R-0	

15	2	1	0
Reserved		GO[1]	GO[0]
R-0		W-0	W-0

LEGEND: R = Read only; W = Write only; -*n* = value after reset

Table 8-14. Power Domain Transition Command Register (PTCMD) Field Descriptions

Bit	Field	Value	Description
31-2	Reserved	0	Reserved
1	GO[1]		RAM/Pseudo (PD1) power domain GO transition command.
		0	A write of 0 has no effect.
		1	A write of 1 causes the PSC to evaluate all the NEXT fields relevant to this power domain (including PDCTL.NEXT for this domain, and MDCTL.NEXT for all the modules residing on this domain). If any of the NEXT fields are not matching the corresponding current state (PDSTAT.STATE, MDSTAT.STATE), the PSC will transition those respective domain/modules to the new NEXT state.
0	GO[0]		Always ON (PD0) power domain GO transition command.
		0	A write of 0 has no effect.
		1	A write of 1 causes the PSC to evaluate all the NEXT fields relevant to this power domain (including MDCTL.NEXT for all the modules residing on this domain). If any of the NEXT fields are not matching the corresponding current state (MDSTAT.STATE), the PSC will transition those respective domain/modules to the new NEXT state.

Copyright © 2013–2016, Texas Instruments Incorporated

8.6.10 Power Domain Transition Status Register (PTSTAT)

The power domain transition status register (PTSTAT) is shown in Figure 8-10 and described in Table 8-15 .

Figure 8-10. Power Domain Transition Status Register (PTSTAT)

31	16
Reserved	
R-0	

15	2	1	0
Reserved		GOSTAT[1]	GOSTAT[0]
R-0		R-0	R-0

LEGEND: R = Read only; -*n* = value after reset

Table 8-15. Power Domain Transition Status Register (PTSTAT) Field Descriptions

Bit	Field	Value	Description
31-2	Reserved	0	Reserved
1	GOSTAT[1]		RAM/Pseudo (PD1) power domain transition status.
		0	No transition in progress.
		1	RAM/Pseudo power domain is transitioning (that is, either the power domain is transitioning or modules in this power domain are transitioning).
0	GOSTAT[0]		Always ON (PD0) power domain transition status.
		0	No transition in progress.
		1	Modules in Always ON power domain are transitioning. Always On power domain is transitioning.

Copyright © 2013–2016, Texas Instruments Incorporated

8.6.11 Power Domain 0 Status Register (PDSTAT0)

The power domain 0 status register (PDSTAT0) is shown in Figure 8-11 and described in Table 8-16.

Figure 8-11. Power Domain 0 Status Register (PDSTAT0)

31	16
Reserved	
R-0	

15	12	11	10	9	8	7	5	4	0
Reserved		EMUIHB	Rsvd	PORDONE	POR	Reserved		STATE	
R-0		R-0	R-0	R-0	R-0	R-0		R-0	

LEGEND: R = Read only; -*n* = value after reset

Table 8-16. Power Domain 0 Status Register (PDSTAT0) Field Descriptions

Bit	Field	Value	Description
31-12	Reserved	0	Reserved
11	EMUIHB		Emulation alters domain state.
		0	Interrupt is not active. No emulation altering user-desired power domain states.
		1	Interrupt is active. Emulation alters user-desired power domain state.
10	Reserved	0	Reserved
9	PORDONE		Power_On_Reset (POR) Done status
		0	Power domain POR is not done.
		1	Power domain POR is done.
8	POR		Power Domain Power_On_Reset (POR) status. This bit reflects the POR status for this power domain including all modules in the domain.
		0	Power domain POR is asserted.
		1	Power domain POR is de-asserted.
7-5	Reserved	0	Reserved
4-0	STATE	0-1Fh	Power Domain Status.
		0	Power domain is in the off state.
		1h	Power domain is in the on state.
		2h-Fh	Reserved
		10h-1Ah	Power domain is in transition.
		1Bh-1Fh	Reserved

Copyright © 2013–2016, Texas Instruments Incorporated

8.6.12 Power Domain 1 Status Register (PDSTAT1)

The power domain 1 status register (PDSTAT1) is shown in Figure 8-12 and described in Table 8-17.

Figure 8-12. Power Domain 1 Status Register (PDSTAT1)

31	16
Reserved	
R-0	

15	12	11	10	9	8	7	5	4	0
Reserved		EMUIHB	Rsvd	PORDONE	POR	Reserved		STATE	
R-0		R-0	R-0	R-0	R-0	R-0		R-0	

LEGEND: R = Read only; -*n* = value after reset

Table 8-17. Power Domain 1 Status Register (PDSTAT1) Field Descriptions

Bit	Field	Value	Description
31-12	Reserved	0	Reserved
11	EMUIHB		Emulation alters domain state.
		0	Interrupt is not active. No emulation altering user-desired power domain states.
		1	Interrupt is active. Emulation alters user-desired power domain state.
10	Reserved	0	Reserved
9	PORDONE		Power_On_Reset (POR) Done status
		0	Power domain POR is not done.
		1	Power domain POR is done.
8	POR		Power Domain Power_On_Reset (POR) status. This bit reflects the POR status for this power domain including all modules in the domain.
		0	Power domain POR is asserted.
		1	Power domain POR is de-asserted.
7-5	Reserved	0	Reserved
4-0	STATE	0-1Fh	Power Domain Status.
		0	Power domain is in the off state.
		1h	Power domain is in the on state.
		2h-Fh	Reserved
		10h-1Ah	Power domain is in transition.
		1Bh-1Fh	Reserved

Copyright © 2013–2016, Texas Instruments Incorporated

8.6.13 Power Domain 0 Control Register (PDCTL0)

The power domain 0 control register (PDCTL0) is shown in Figure 8-13 and described in Table 8-18.

Figure 8-13. Power Domain 0 Control Register (PDCTL0)

31	24	23	16
Reserved		WAKECNT	
R-0		R/W-1Fh	

15	12	11	10	9	8	7	1	0
PDMODE		Reserved		EMUIHBIE	Rsvd	Reserved		NEXT
R-Fh		R-0		R/W-0	R-1	R-0		R/W-1

LEGEND: R/W = Read/Write; R = Read only; *-n* = value after reset

Table 8-18. Power Domain 0 Control Register (PDCTL0) Field Descriptions

Bit	Field	Value	Description
31-24	Reserved	0	Reserved
23-16	WAKECNT	0-FFh	RAM wake count delay value. Not recommended to change the default value (1Fh). Bits 23-30: GOOD2ACCESS wake delay. Bits 19-16: ON2GOOD wake delay.
15-12	PDMODE	0-Fh	Power down mode.
		0-Eh	Reserved
		Fh	Core on, RAM array on, RAM periphery on.
11-10	Reserved	0	Reserved
9	EMUIHBIE		Emulation alters power domain state interrupt enable.
		0	Disable interrupt.
		1	Enable interrupt.
8	Reserved	1	Reserved
7-1	Reserved	0	Reserved
0	NEXT		Power domain next state. For Always ON power domain this bit is read/write, but writes have no effect since internally this power domain always remains in the on state.
		0	Power domain off.
		1	Power domain on.

Copyright © 2013–2016, Texas Instruments Incorporated

8.6.14 Power Domain 1 Control Register (PDCTL1)

The power domain 1 control register (PDCTL1) is shown in Figure 8-14 and described in Table 8-19.

Figure 8-14. Power Domain 1 Control Register (PDCTL1)

31	24	23	16
Reserved		WAKECNT	
R-0		R/W-1Fh	

15	12	11	10	9	8	7	1	0
PDMODE		Reserved		EMUIHBIE	Rsvd	Reserved		NEXT
R-Fh		R-0		R/W-0	R-1	R-0		R/W-1

LEGEND: R/W = Read/Write; R = Read only; *-n* = value after reset

Table 8-19. Power Domain 1 Control Register (PDCTL1) Field Descriptions

Bit	Field	Value	Description
31-24	Reserved	0	Reserved
23-16	WAKECNT	0-FFh	RAM wake count delay value. Not recommended to change the default value (1Fh). Bits 23-30: GOOD2ACCESS wake delay. Bits 19-16: ON2GOOD wake delay.
15-12	PDMODE	0-Fh	Power down mode.
		0	Core off, RAM array off, RAM periphery off.
		1h	Core off, RAM array retention, RAM periphery off (deep sleep).
		2h-3h	Reserved
		4h	Core retention, RAM array off, RAM periphery off.
		5h	Core retention, RAM array retention, RAM periphery off (deep sleep).
		6h-7h	Reserved
		8h	Core on, RAM array off, RAM periphery off.
		9h	Core on, RAM array retention, RAM periphery off (deep sleep).
		Ah	Core on, RAM array retention, RAM periphery off (light sleep).
		Bh	Core on, RAM array retention, RAM periphery on.
		Ch-Eh	Reserved
		Fh	Core on, RAM array on, RAM periphery on.
11-10	Reserved	0	Reserved
9	EMUIHBIE		Emulation alters power domain state interrupt enable.
		0	Disable interrupt.
		1	Enable interrupt.
8	Reserved	1	Reserved
7-1	Reserved	0	Reserved
0	NEXT		User-desired power domain next state.
		0	Power domain off.
		1	Power domain on.

Copyright © 2013–2016, Texas Instruments Incorporated

8.6.15 Power Domain 0 Configuration Register (PDCFG0)

The power domain 0 configuration register (PDCFG0) is shown in Figure 8-15 and described in Table 8-20.

Figure 8-15. Power Domain 0 Configuration Register (PDCFG0)

31	16
Reserved	
R-0	

15	4	3	2	1	0
Reserved		PD_LOCK	ICEPICK	RAM_PSM	ALWAYSON
R-0		R-1	R-1	R-0	R-1

LEGEND: R = Read only; -*n* = value after reset

Table 8-20. Power Domain 0 Configuration Register (PDCFG0) Field Descriptions

Bit	Field	Value	Description
31-4	Reserved	0	Reserved
3	PD_LOCK		PDCTL.NEXT lock. For Always ON power domain this bit is a don't care.
		0	PDCTL.NEXT bit is locked and cannot be changed in software.
		1	PDCTL.NEXT bit is not locked.
2	ICEPICK		IcePick support.
		0	Not present
		1	Present
1	RAM_PSM		RAM power domain.
		0	Not a RAM power domain.
		1	RAM power domain.
0	ALWAYSON		Always ON power domain.
		0	Not an Always ON power domain.
		1	Always ON power domain.

Copyright © 2013–2016, Texas Instruments Incorporated

8.6.16 Power Domain 1 Configuration Register (PDCFG1)

The power domain 1 configuration register (PDCFG1) is shown in Figure 8-16 and described in Table 8-21.

Figure 8-16. Power Domain 1 Configuration Register (PDCFG1)

31	16
Reserved	
R-0	

15	4	3	2	1	0
Reserved		PD_LOCK	ICEPICK	RAM_PSM	ALWAYSON
R-0		R-1	R-1	R-0	R-1

LEGEND: R = Read only; -*n* = value after reset

Table 8-21. Power Domain 1 Configuration Register (PDCFG1) Field Descriptions

Bit	Field	Value	Description
31-4	Reserved	0	Reserved
3	PD_LOCK		PDCTL.NEXT lock. For Always ON power domain this bit is a don't care.
		0	PDCTL.NEXT bit is locked and cannot be changed in software.
		1	PDCTL.NEXT bit is not locked.
2	ICEPICK		IcePick support.
		0	Not present
		1	Present
1	RAM_PSM		RAM power domain.
		0	Not a RAM power domain.
		1	RAM power domain.
0	ALWAYSON		Always ON power domain.
		0	Not an Always ON power domain.
		1	Always ON power domain.

Copyright © 2013–2016, Texas Instruments Incorporated

8.6.17 Module Status *n* Register (MDSTAT*n*)

The module status *n* register (MDSTAT*n*) is shown in Figure 8-17 and described in Table 8-22.

Figure 8-17. Module Status *n* Register (MDSTAT*n*)

31	18	17	16
Reserved		EMUIHB	EMURST
R-0		R-0	R-0

15	13	12	11	10	9	8	7	6	5	0
Reserved		MCKOUT	Rsvd	MRST	LRSTDONE	LRST	Reserved		STATE	
R-0		R-0	R-1	R-0	R-1	R-1	R-0		R-0	

LEGEND: R = Read only; -*n* = value after reset

Table 8-22. Module Status *n* Register (MDSTAT*n*) Field Descriptions

Bit	Field	Value	Description
31-18	Reserved	0	Reserved
17	EMUIHB		Emulation alters module state. This bit applies to DSP module (module 15). This field is 0 for all other modules.
		0	No emulation altering user-desired module state programmed in the NEXT bit in the module control 15 register (MDCTL15).
		1	Emulation altered user-desired state programmed in the NEXT bit in MDCTL15. If you desire to generate a PSCINT upon this event, you must set the EMUIHBIE bit in MDCTL15.
16	EMURST		Emulation alters module reset. This bit applies to DSP module (module 15). This field is 0 for all other modules.
		0	No emulation altering user-desired module reset state.
		1	Emulation altered user-desired module reset state. If you desire to generate a PSCINT upon this event, you must set the EMURSTIE bit in the module control 15 register (MDCTL15).
15-13	Reserved	0	Reserved
12	MCKOUT		Module clock output status. Shows status of module clock.
		0	Module clock is off.
		1	Module clock is on.
11	Reserved	1	Reserved
10	MRST		Module reset status. Reflects actual state of module reset.
		0	Module reset is asserted.
		1	Module reset is de-asserted.
9	LRSTDONE		Local reset done. Software is responsible for checking if local reset is done before accessing this module. This bit applies to DSP module (module 15). This field is 1 for all other modules.
		0	Local reset is not done.
		1	Local reset is done.
8	LRST		Module local reset status. This bit applies to DSP module (module 15).
		0	Local reset is asserted.
		1	Local reset is de-asserted.
7-6	Reserved	0	Reserved
5-0	STATE	0-3Fh	Module state status: indicates current module status.
		0	SwRstDisable state
		1h	SyncReset state
		2h	Disable state
		3h	Enable state
		4h-3Fh	Indicates transition

Copyright © 2013–2016, Texas Instruments Incorporated

8.6.18 PSC0 Module Control *n* Register (modules 0-15) (MDCTL*n*)

The PSC0 module control *n* register (MDCTL*n*) is shown in Figure 8-18 and described in Table 8-23.

Figure 8-18. PSC0 Module Control *n* Register (MDCTL*n*)

31	30	16
FORCE	Reserved	
R/W-0	R-0	

15	11	10	9	8	7	3	2	0
Reserved		EMUIHBIE	EMURSTIE	LRST	Reserved		NEXT	
R-0		R/W-0	R/W-0	R/W-0	R-0		R/W-0	

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 8-23. PSC0 Module Control *n* Register (MDCTL*n*) Field Descriptions

Bit	Field	Value	Description
31	FORCE		Force enable. This bit forces the module state programmed in the NEXT bit in the module control 15 register (MDCTL15), ignoring and bypassing all the clock stop request handshakes managed by the PSC to change the state of the clocks to the module.
			Note: It is **not** recommended to use the FORCE bit to disable the module clock, unless specified.
		0	Force is disabled.
		1	Force is enabled.
30-11	Reserved	0	Reserved
10	EMUIHBIE		Interrupt enable for emulation alters module state. This bit applies to DSP module (module 15).
		0	Disable interrupt.
		1	Enable interrupt.
9	EMURSTIE		Interrupt enable for emulation alters reset. This bit applies to DSP module (module 15).
		0	Disable interrupt.
		1	Enable interrupt.
8	LRST		Module local reset control. This bit applies to DSP module (module 15).
		0	Assert local reset
		1	De-assert local reset
7-3	Reserved	0	Reserved
2-0	NEXT	0-3h	Module next state.
		0	SwRstDisable state
		1h	SyncReset state
		2h	Disable state
		3h	Enable state

Copyright © 2013–2016, Texas Instruments Incorporated

8.6.19 PSC1 Module Control *n* Register (modules 0-31) (MDCTL*n*)

The PSC1 module control *n* register (MDCTL*n*) is shown in Figure 8-19 and described in Table 8-24.

Figure 8-19. PSC1 Module Control *n* Register (MDCTL*n*)

31	30	16
FORCE	Reserved	
R/W-0	R-0	

15	3	2	0
Reserved		NEXT	
R-0		R/W-0	

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 8-24. PSC1 Module Control *n* Register (MDCTL*n*) Field Descriptions

Bit	Field	Value	Description
31	FORCE		Force enable. This bit forces the module state programmed in the NEXT bit in the module control 15 register (MDCTL15), ignoring and bypassing all the clock stop request handshakes managed by the PSC to change the state of the clocks to the module. Note: It is **not** recommended to use the FORCE bit to disable the module clock, unless specified.
		0	Force is disabled.
		1	Force is enabled.
30-3	Reserved	0	Reserved
2-0	NEXT	0-3h	Module next state.
		0	SwRstDisable state
		1h	SyncReset state
		2h	Disable state
		3h	Enable state

Copyright © 2013–2016, Texas Instruments Incorporated

Power Management

Topic Page

9.1 Introduction 122
9.2 Power Consumption Overview 122
9.3 PSC and PLLC Overview 122
9.4 Features 123
9.5 Clock Management 124
9.6 DSP Sleep Mode Management 125

Copyright © 2013–2016, Texas Instruments Incorporated

9.1 Introduction

Power management is an important aspect for most embedded applications. For several applications and target markets, there may be a specific power budget and requirements to minimize power consumption for both power supply sizing and battery life considerations. Additionally, lower power consumption results in more optimal and efficient designs from cost, design, and energy perspectives. This device has several means of managing the power consumption. This chapter discusses the various power management features.

9.2 Power Consumption Overview

Power consumed by semiconductor devices has two components: dynamic and static. This can be shown as:

$Ptotal = P_{dynamic} + P_{static}$

The dynamic power is the power consumed to perform work when the device is in active modes (clocks applied, busses, and I/O switching), that is, analog circuits changing states. The dynamic power is defined by:

$P_{dynamic} = Capacitance \times Voltage^2 \times Frequency$

From the above formula, the dynamic power scales with the clock frequency (device/module frequency for core operations and switching frequency for I/O). Dynamic power can be reduced by controlling the clocks in such a way as to either operate at a clock setting just high enough to complete the required operation in the required timeline or to run at a clock setting until the work is complete and then drastically reduce the clock frequency or cut off the clocks until additional work must be performed.

In the formula, the dynamic power varies with the voltage squared, so the voltage of operations has significant impact on overall power consumption and, thus, on the battery life. Dynamic power can be reduced by scaling the operating voltage, when the performance requirements are not that high and the device can be operated at a corresponding lower frequency.

The capacitance is the capacitance of the switching nodes, or the load capacitances on the switching I/O pins.

The static power, as the name suggests, is independent of the switching frequency of the logic. It can be shown as:

$P_{static} = f_{(leakage\ current)}$

It is essentially a function of the “leakage”, or the power consumed by the logic when it is not switching or is not performing any work. Leakage current is dependent mostly on the manufacturing process used, the size of the die, etc. Leakage current is unavoidable while power is applied and scales roughly with the operating junction temperatures. Leakage power can only be avoided by removing power completely from a device or subsystem. The static power consumption plays a significant role in the Standby Modes (when the application is not running and in a dormant state) and plays an important role in the battery life for portable applications, etc.

9.3 PSC and PLLC Overview

The power and sleep controller (PSC) module plays an important role in managing the enabling/disabling of the clocks to the core and various peripheral modules. The PSC provides a granular support to turn on/off clocks on a module by module basis. Similarly, the PLL controller (PLLC) plays an important role in device and module clock generation, and manages the frequency scaling operations for the device. Together, both of these modules play a significant role in managing the clocks from a power management feature standpoint. For detailed information on the PSC, see the *Power and Sleep Controller (PSC)* chapter. For detailed information on the PLLC, see the *Device Clocking* chapter and the *Phase-Locked Loop Controller (PLLC)* chapter.

Copyright © 2013–2016, Texas Instruments Incorporated

9.4 Features

This device has several means of managing power consumption, as detailed in the subsequent sections. This device uses the state-of-the-art 65 nm process, which provides a good balance on power and performance, providing high-performance transistors with relatively less leakage current and, thereby, low standby-power consumption modes.

There are several features in design as well as user driven software control to reduce dynamic power consumption. The design features (not under user control) include a power optimized clock tree design to reduce overall clock tree power consumption and automatic clock gating in several modules when the logic in the modules is not active.

The on-chip power and sleep controller (PSC) module provides granular software controlled module level clock gating, which reduces both clock tree and module power by basically disabling the clocks when the modules are not being used. Clock management also allows you to slow down the clocks, to reduce the dynamic power.

Table 9-1 describes the power management features.

Table 9-1. Power Management Features

Power Management	Description	Features
	Clock Management	
PLL power-down	The PLL can be powered-down and run in bypass modes when not in use.	Reduces the dynamic power consumption of the core.
Module clock ON/OFF	Module clocks can be turned on/off without requiring reconfiguring the registers.	Reduces the dynamic/switching power consumption of the core and I/O (if any free running I/O clocks).
Core/module clock frequency scaling	The device can be run at a lower frequency using the PLLM/PLL dividers. Many modules have internal clock dividers to scale module/IO frequency.	Reduces the dynamic/switching power consumption of core and I/O.
	Core Sleep Management	
DSP subsystem sleep mode	The DSP CPU can be put in sleep (IDLE) mode.	Reduces the dynamic power.

Copyright © 2013–2016, Texas Instruments Incorporated

9.5 Clock Management

9.5.1 Module Clock ON/OFF

The module clock on/off feature allows software to disable clocks to module individually, in order to reduce the module's dynamic/switching power consumption down to zero. This device is designed in full static CMOS; thus, when a module clock stops, the module's state is preserved and retained. When the clock is restarted, the module resumes operating from the stopping point.

NOTE: Stopping clocks to a module only affects dynamic power consumption, it does not affect static power consumption of the module or the device.

The power and sleep controller (PSC) module controls module clock gating. If a module's clock(s) is stopped while being accessed, the access may not occur, and it can potentially result in unexpected behavior. The PSC provides some protection against such erroneous conditions by monitoring the internal bus activity to ensure there are no accesses to the module from the internal bus, before allowing module's internal clock to be gated. However, it is still recommended that software must ensure that all of the transactions to the module are finished prior to disabling the clocks.

The procedure to turn module clocks on/off using the PSC is described in the *Power and Sleep Controller (PSC)* chapter.

Furthermore, special consideration must be given to DSP clock on/off. The procedure to turn the core clock on/off is further described in .

Additionally some peripherals implement additional power saving features by automatically shutting of clock to components within the module , when the logic is not active. This is transparent to you, but reduces overall dynamic power consumption when modules are not active.

9.5.2 Module Clock Frequency Scaling

Module clock frequency is scalable by programming the PLL multiply and divide parameters. Additionally, some modules might also have internal clock dividers. Reducing the clock frequency reduces the dynamic/switching power consumption, which scales linearly with frequency.

The *Device Clocking* chapter details the clocking structure of the device. The *Phase-Locked Loop Controller (PLLC)* chapter describes how to program the PLL0 and PLL1 frequency and the frequency constraints.

9.5.3 PLL Bypass and Power Down

You can bypass the PLL in the device. Bypassing the PLL sends the PLL reference clock (OSCIN) instead of the PLL VCO output (PLLOUT) to the system clocks of the PLLC. The PLL OSCIN is typically, at most, up to 50 MHz. You can use this mode to reduce the core and module clock frequencies to very low maintenance levels without using the PLL during periods of very low system activity, this again can lower the overall dynamic/switching power consumption, which is linearly proportional to the frequency. Furthermore, you can also power-down the PLL when bypassing it to minimize the overall power consumed by the PLL module.

The *Device Clocking* chapter and the *Phase-Locked Loop Controller (PLLC)* chapter describe PLL bypass and PLL power down.

Copyright © 2013–2016, Texas Instruments Incorporated

9.6 DSP Sleep Mode Management

9.6.1 C674x DSP CPU Sleep Mode

The DSP CPU can be put in a low-power state by executing the IDLE instruction. For information on the IDLE instruction, see the *TMS320C674x DSP CPU and Instruction Set Reference Guide* (SPRUFE8).

9.6.2 C674x Megamodule Sleep Mode

The IDLE instruction is used as part of the procedure for shutting down the entire C674x megamodule, by the power-down controller (PDC) module. In shutting down the entire C674x megamodule, the PDC can internally clock gate off the following components of the megamodule and internal memories of the DSP subsystem:

- C674x CPU
- Level 1 Program Memory Controller (PMC)
- Level 1 Data Memory Controller (DMC)
- Level 2 Unified Memory Controller (UMC)
- Extended Memory Controller (EMC)
- L1P Memory
- L1D Memory
- L2 Memory

Putting the entire C674x megamodule into the low-power sleep mode is typically more useful and saves a lot more power, as compared to just executing the IDLE instruction to put only the CPU in idle mode.

For information on putting the C674x megamodule in the low-power mode using the PDC, see the *TMS320C674x DSP Megamodule Reference Guide* (SPRUFK5).

Copyright © 2013–2016, Texas Instruments Incorporated

System Configuration (SYSCFG) Module

Topic **Page**

10.1 Introduction 127
10.2 Protection 128
10.3 Master Priority Control 129
10.4 Interrupt Support 130
10.5 SYSCFG Registers 130

Copyright © 2013–2016, Texas Instruments Incorporated

10.1 Introduction

The system configuration (SYSCFG) module is a system-level module containing status and top level control logic required by the device. The system configuration module consists of a set of memory-mapped status and control registers, accessible by the CPU, supporting all of the following system features, and miscellaneous functions and operations.

- Device Identification
- Device Configuration
  - Pin multiplexing control
  - Device Boot Configuration Status
- Master Priority Control
  - Controls the system priority for all master peripherals (including EDMA3TC)
- Emulation Control
  - Emulation suspend control for peripherals that support the feature
- Special Peripheral Status and Control
  - Locking of PLL control settings
  - Default burst size configuration for EDMA3 transfer controllers
  - Event source selection for the eCAP peripheral input capture
  - McASP AMUTEIN selection and clearing of AMUTE

The system configuration module controls several global operations of the device; therefore, the module supports protection against erroneous and illegal accesses to the registers in its memory-map. The protection mechanisms that are present in the module are:

- A special key sequence that needs to be written into a set of registers in the system configuration module, to allow write ability to the rest of registers in the system configuration module.
- Several registers in the module are only accessible when the CPU requesting read/write access is in privileged mode.

Copyright © 2013–2016, Texas Instruments Incorporated

10.2 Protection

Table 10-1 provides the list of registers in the SYSCFG module; it also indicates whether a particular register can be accessed only when the CPU is in privileged mode. See Section 10.5 for a description of these registers.

Table 10-1. System Configuration (SYSCFG) Module Register Access

Offset	Acronym	Register Description	Access
0h	REVID	Revision Identification Register	—
8h-14h	DIEIDR0-DIEIDR3	Die Identification 0-3 Registers	—
18h	DEVIDR0	Device Identification Register 0	—
20h	BOOTCFG	Boot Configuration Register	Privileged mode
24h	CHIPREVID	Silicon Revision Identification Register	Privileged mode
38h	KICK0R	Kick 0 Register	Privileged mode
3Ch	KICK1R	Kick 1 Register	Privileged mode
44h	HOST1CFG	Host 1 Configuration Register	—
E0h	IRAWSTAT	Interrupt Raw Status/Set Register	Privileged mode
E4h	IENSTAT	Interrupt Enable Status/Clear Register	Privileged mode
E8h	IENSET	Interrupt Enable Register	Privileged mode
ECh	IENCLR	Interrupt Enable Clear Register	Privileged mode
F0h	EOI	End of Interrupt Register	Privileged mode
F4h	FLTADDRR	Fault Address Register	Privileged mode
F8h	FLTSTAT	Fault Status Register	—
110h-118h	MSTPRI0-MSTPRI2	Master Priority 0-2 Registers	Privileged mode
120h-16Ch	PINMUX0-PINMUX19	Pin Multiplexing Control 0-19 Registers	Privileged mode
170h	SUSPSRC	Suspend Source Register	Privileged mode
174h	CHIPSIG	Chip Signal Register	—
178h	CHIPSIG_CLR	Chip Signal Clear Register	—
17Ch-18Ch	CFGCHIP0-CFGCHIP4	Chip Configuration 0-4 Registers	Privileged mode

10.2.1 Requirements to Access SYSCFG Registers

As mentioned previously, the SYSCFG module controls several global operations of the device; therefore, it has protection mechanism that prevents spurious and illegal accesses to the registers in its memory map. The protection mechanism enables accesses to these registers only if certain conditions are met. The protection mechanisms that are present in the module are described in the following sections.

10.2.1.1 Privilege Mode Protection

The CPU supports two privilege levels: Supervisor and User. Several registers in the SYSCFG memory-map can only be accessed when the accessing host (CPU or master peripheral) is operating in privileged mode, that is, in Supervisor mode. The registers that can only be accessed in privileged mode are listed in Section 10.5. See the for details on privilege levels.

Copyright © 2013–2016, Texas Instruments Incorporated

10.2.1.2 Kicker Mechanism Protection

NOTE: The Kick 0 and Kick 1 registers can only be accessed in privileged mode (the host needs to be in Supervisor mode). Any number of accesses may be performed to the SYSCFG module, while the module is unlocked.

The SYSCFG module remains unlocked after the unlock sequence, until locked again. Locking the module is accomplished by writing any value other then the key values to either KICK0 or KICK1.

To access any registers in the SYSCFG module, it is required to follow a special sequence of writes to the Kick registers (Kick0 and Kick1) with correct key values. Writing the correct key value to the kick registers unlocks the registers in the SYSCFG memory-map. In order to access the SYSCFG registers, the following unlock sequence needs to be executed in software:

1. Write the key value of 83E7 0B13h to Kick 0 register.
2. Write the key value of 95A4 F1E0h to Kick 1 register.

After steps 1 and 2, the SYSCFG module registers are accessible and can be configured as per the application requirements.

10.3 Master Priority Control

The on-chip peripherals/modules are essentially divided into two broad categories, masters and slaves. The master peripherals are typically capable of initiating their own read/write data access requests, this includes the EDMA3 transfer controllers, and peripherals that do not rely on the CPU or EDMA3 for initiating the data transfer to/from them. In order to determine allowed connection between masters and slave, each master request source must have a unique master ID (mstid) associated with it. The master ID is shown in Table 10-2. See the device-specific data manual to determine the masters present on your device.

Each switched central resource (SCR) performs prioritization based on priority level of the master that sends the read/write requests. For all peripherals/ports classified as masters on the device, the priority is programmed in the master priority registers (MSTPRI0-3) in the SYSCFG modules. The default priority levels for each bus master is shown in Table 10-3. Application software is expected to modify these values to obtain the desired performance.

Table 10-2. Master IDs

Master ID	Peripheral
0-1	Reserved
2	DSP MDMA
3	DSP CFG
4-7	Reserved
8	dMAX0
9	dMAX1
10	TPCC0
11-15	Reserved
16	TPTC0 - read
17	TPTC0 - write
18	TPTC1 - read
19	TPTC1 - write
20-255	Reserved

Copyright © 2013–2016, Texas Instruments Incorporated

Table 10-3. Default Master Priority

Master	Default Priority[1]	Master Priority Register
dMAX0	0	MSTPRI1
dMAX1	0	MSTPRI1
EDMA3TC0[2]	0	MSTPRI1
EDMA3TC1	0	MSTPRI1
DSP MDMA[3]	2	MSTPRI0
DSP CFG[4]	2	MSTPRI0

(1) The default priority settings might not be optimal for all applications. The master priority should be changed from default based on application specific requirement, in order to get optimal performance and prioritization for masters moving data that is real time sensitive.
(2) The priority for EDMA3TC0 and EDMA3TC1 is configurable through fields in MSTPRI1, not the EDMA3CC QUEPRI register.
(3) The priority for DSP MDMA and DSP CFG is controlled by fields in MSTPRI0 and not DSP.MDMAARBE.PRI (DSP Bandwidth manager module).
(4) The priority for DSP MDMA and DSP CFG is controlled by fields in MSTPRI0 and not DSP.MDMAARBE.PRI (DSP Bandwidth manager module).

10.4 Interrupt Support

10.4.1 Interrupt Events and Requests

The SYSCFG module generates two interrupts: an address error interrupt (BOOTCFG_ADDR_ERR) and a protection interrupt (BOOTCFG_PROT_ERR). The BOOTCFG_ADDR_ERR is generated when there is an addressing violation due to an access to a non-existent location in the SYSCFG register space. The BOOTCFG_PROT_ERR interrupt is generated when there is a protection violation of either in the defined ranges or to the SYSCFG registers. It is required to write a value of 0 to the end of interrupt register (EOI) after the software has processed the SYSCFG interrupt, this acts as an acknowledgement of completion of the SYSCFG interrupt so that the module can reliably generate subsequent interrupts.

The transfer parameters that caused the violation are saved in the fault address register (FLTADDRR) and the fault status register (FLTSTAT).

10.4.2 Interrupt Multiplexing

The interrupts from the SYSCFG module are combined with the interrupts from the MPU module into a single interrupt called MPU_BOOTCFG_ERR. The combined interrupt is routed to the DSP interrupt controller.

10.4.3 Host-DSP Communication Interrupts

The SYSCFG module also has a set of registers, the chip signal register (CHIPSIG) and the chip signal clear register (CHIPSIG_CLR), to facilitate host-to-processor communication. This is generally used to allow an external host and the DSP to coordinate.

Either of the processors can set specific bits in this SYSCFG register, which in turn can interrupt the other processor, if the interrupts have been appropriately enabled in the processor's interrupt controller.

10.5 SYSCFG Registers

Table 10-4 lists the memory-mapped registers for the system configuration module (SYSCFG).

Table 10-4. System Configuration Module (SYSCFG) Registers

Address	Acronym	Register Description	Section
01C1 4000h	REVID	Revision Identification Register	Section 10.5.1
01C1 4008h	DIEIDR0[1]	Die Identification Register 0	—

(1) This register is for internal-use only.

Copyright © 2013–2016, Texas Instruments Incorporated

Table 10-4. System Configuration Module (SYSCFG) Registers (continued)

Address	Acronym	Register Description	Section
01C1 400Ch	DIEIDR1[1]	Die Identification Register 1	—
01C1 4010h	DIEIDR2[1]	Die Identification Register 2	—
01C1 4014h	DIEIDR3[1]	Die Identification Register 3	—
01C1 4018h	DEVIDR0	Device Identification Register 0	Section 10.5.2
01C1 4020h	BOOTCFG	Boot Configuration Register	Section 10.5.3
01C1 4024h	CHIPREVID	Silicon Revision Identification Register	Section 10.5.4
01C1 4038h	KICK0R	Kick 0 Register	Section 10.5.5.1
01C1 403Ch	KICK1R	Kick 1 Register	Section 10.5.5.2
01C1 4044h	HOST1CFG	Host 1 Configuration Register	Section 10.5.6
01C1 40E0h	IRAWSTAT	Interrupt Raw Status/Set Register	Section 10.5.7.1
01C1 40E4h	IENSTAT	Interrupt Enable Status/Clear Register	Section 10.5.7.2
01C1 40E8h	IENSET	Interrupt Enable Register	Section 10.5.7.3
01C1 40ECh	IENCLR	Interrupt Enable Clear Register	Section 10.5.7.4
01C1 40F0h	EOI	End of Interrupt Register	Section 10.5.7.5
01C1 40F4h	FLTADDRR	Fault Address Register	Section 10.5.8.1
01C1 40F8h	FLTSTAT	Fault Status Register	Section 10.5.8.2
01C1 4110h	MSTPRI0	Master Priority 0 Register	Section 10.5.9.1
01C1 4114h	MSTPRI1	Master Priority 1 Register	Section 10.5.9.2
01C1 4118h	MSTPRI2	Master Priority 2 Register	Section 10.5.9.3
01C1 4120h	PINMUX0	Pin Multiplexing Control 0 Register	Section 10.5.10.1
01C1 4124h	PINMUX1	Pin Multiplexing Control 1 Register	Section 10.5.10.2
01C1 4128h	PINMUX2	Pin Multiplexing Control 2 Register	Section 10.5.10.3
01C1 412Ch	PINMUX3	Pin Multiplexing Control 3 Register	Section 10.5.10.4
01C1 4130h	PINMUX4	Pin Multiplexing Control 4 Register	Section 10.5.10.5
01C1 4134h	PINMUX5	Pin Multiplexing Control 5 Register	Section 10.5.10.6
01C1 4138h	PINMUX6	Pin Multiplexing Control 6 Register	Section 10.5.10.7
01C1 413Ch	PINMUX7	Pin Multiplexing Control 7 Register	Section 10.5.10.8
01C1 4140h	PINMUX8	Pin Multiplexing Control 8 Register	Section 10.5.10.9
01C1 4144h	PINMUX9	Pin Multiplexing Control 9 Register	Section 10.5.10.10
01C1 4148h	PINMUX10	Pin Multiplexing Control 10 Register	Section 10.5.10.11
01C1 414Ch	PINMUX11	Pin Multiplexing Control 11 Register	Section 10.5.10.12
01C1 4150h	PINMUX12	Pin Multiplexing Control 12 Register	Section 10.5.10.13
01C1 4154h	PINMUX13	Pin Multiplexing Control 13 Register	Section 10.5.10.14
01C1 4158h	PINMUX14	Pin Multiplexing Control 14 Register	Section 10.5.10.15
01C1 415Ch	PINMUX15	Pin Multiplexing Control 15 Register	Section 10.5.10.16
01C1 4160h	PINMUX16	Pin Multiplexing Control 16 Register	Section 10.5.10.17
01C1 4164h	PINMUX17	Pin Multiplexing Control 17 Register	Section 10.5.10.18
01C1 4168h	PINMUX18	Pin Multiplexing Control 18 Register	Section 10.5.10.19
01C1 416Ch	PINMUX19	Pin Multiplexing Control 19 Register	Section 10.5.10.20
01C1 4170h	SUSPSRC	Suspend Source Register	Section 10.5.11
01C1 4174h	CHIPSIG	Chip Signal Register	Section 10.5.12
01C1 4178h	CHIPSIG_CLR	Chip Signal Clear Register	Section 10.5.13
01C1 417Ch	CFGCHIP0	Chip Configuration 0 Register	Section 10.5.14
01C1 4180h	CFGCHIP1	Chip Configuration 1 Register	Section 10.5.15
01C1 418Ch	CFGCHIP4	Chip Configuration 4 Register	Section 10.5.16

Copyright © 2013–2016, Texas Instruments Incorporated

10.5.1 Revision Identification Register (REVID)

The revision identification register (REVID) provides the revision information for the SYSCFG module. The REVID is shown in Figure 10-1 and described in Table 10-5.

Figure 10-1. Revision Identification Register (REVID)

31	0
REV	
R-4E84 0102h	

LEGEND: R = Read only; -*n* = value after reset

Table 10-5. Revision Identification Register (REVID) Field Descriptions

Bit	Field	Value	Description
31-0	REV	4E84 0102h	**Revision ID.** Revision information for the SYSCFG module.

10.5.2 Device Identification Register 0 (DEVIDR0)

The device identification register 0 (DEVIDR0) contains a software readable version of the JTAG ID device. Software can use this register to determine the version of the device on which it is executing. The DEVIDR0 is shown in Figure 10-2 and described in Table 10-6.

Figure 10-2. Device Identification Register 0 (DEVIDR0)

31	0
DEVID0	
R-0B7D F02Fh	

LEGEND: R = Read only; -*n* = value after reset

Table 10-6. Device Identification Register 0 (DEVIDR0) Field Descriptions

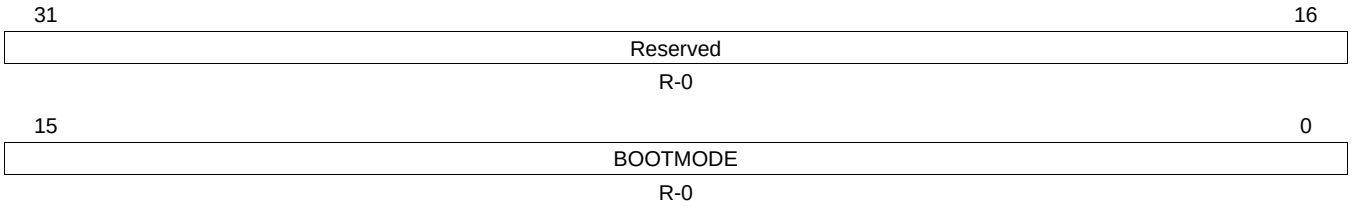
Bit	Field	Value	Description
31-0	DEVID0	R-0B7D F02Fh	Device identification.

Copyright © 2013–2016, Texas Instruments Incorporated

10.5.3 Boot Configuration Register (BOOTCFG)

The device boot and configuration settings are latched at device reset, and captured in the boot configuration register (BOOTCFG). See the device-specific data manual and the *Boot Considerations* chapter for details on boot and configuration settings. The BOOTCFG is shown in Figure 10-3 and described in Table 10-7.

Figure 10-3. Boot Configuration Register (BOOTCFG)



LEGEND: R = Read only; -*n* = value after reset

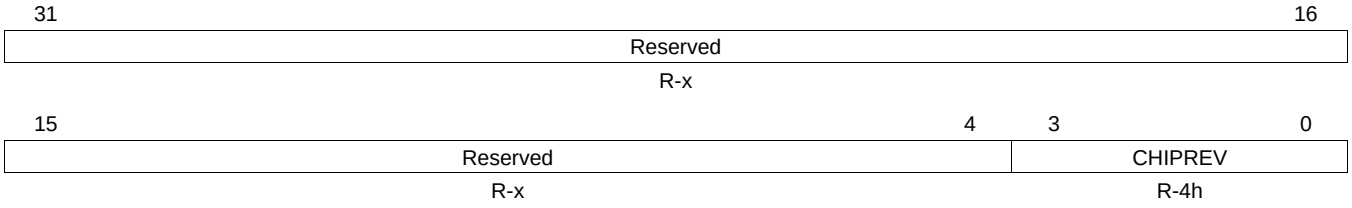
Table 10-7. Boot Configuration Register (BOOTCFG) Field Descriptions

Bit	Field	Value	Description
31-16	Reserved	0	Reserved
15-0	BOOTMODE	0-FFFFh	**Boot Mode.** This reflects the state of the boot mode pins.

10.5.4 Silicon Revision Identification Register (CHIPREVID)

The silicon revision identification register (CHIPREVID) provides software-readable silicon revision information for the device. The CHIPREVID is shown in Figure 10-4 and described in Table 10-8.

Figure 10-4. Silicon Revision Identification Register (CHIPREVID)



LEGEND: R = Read only; -*n* = value after reset; x = value is indeterminate after reset

Table 10-8. Silicon Revision Identification Register (CHIPREVID) Field Descriptions

Bit	Field	Value	Description
31-4	Reserved	0	Reserved
3-0	CHIPREV		**Identifies silicon revision of device.**
		0-3h	Older silicon revision
		4h	Silicon revision 3.0

Copyright © 2013–2016, Texas Instruments Incorporated

10.5.5 Kick Registers (KICK0R-KICK1R)

The SYSCFG module has a protection mechanism to prevent any spurious writes from changing any of the modules memory-mapped registers. At power-on reset, none of the SYSCFG module registers are writeable (they are readable). To allow writing to the registers in the module, it is required to “unlock” the registers by writing to two memory-mapped registers in the SYSCFG module, Kick0 and Kick1, with exact data values. Once these values are written, then all the registers in the SYSCFG module that are writeable can be written to. See Section 10.2.1.2 for the exact key values and sequence of steps. Writing any other data value to either of these kick registers will cause the memory mapped registers to be “locked” again and block out any write accesses to registers in the SYSCFG module.

10.5.5.1 Kick 0 Register (KICK0R)

The KICK0R is shown in Figure 10-5 and described in Table 10-9.

Figure 10-5. Kick 0 Register (KICK0R)

31	0
KICK1	
R/W-0	

LEGEND: R/W = Read/Write; -*n* = value after reset

Table 10-9. Kick 0 Register (KICK0R) Field Descriptions

Bit	Field	Value	Description
31-0	KICK0	0-FFFF FFFFh	**KICK0R allows writing to unlock the kick0 data.** The written data must be 83E7 0B13h to unlock this register. It must be written before writing to the kick1 register. Writing any other value will lock the other MMRs.

10.5.5.2 Kick 1 Register (KICK1R)

The KICK1R is shown in Figure 10-6 and described in Table 10-10.

Figure 10-6. Kick 1 Register (KICK1R)

31	0
KICK0	
R/W-0	

LEGEND: R/W = Read/Write; -*n* = value after reset

Table 10-10. Kick 1 Register (KICK1R) Field Descriptions

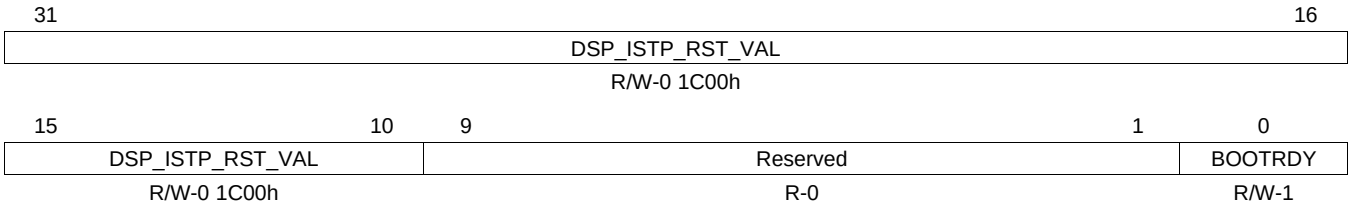
Bit	Field	Value	Description
31-0	KICK1	0-FFFF FFFFh	**KICK1R allows writing to unlock the kick1 data and the kicker mechanism to write to other MMRs.** The written data must be 95A4 F1E0h to unlock this register. KICK0R must be written before writing to the kick1 register. Writing any other value will lock the other MMRs.

Copyright © 2013–2016, Texas Instruments Incorporated

10.5.6 Host 1 Configuration Register (HOST1CFG)

The host 1 configuration register (HOST1CFG) provides information on the DSP boot address value at power-on reset. The boot address defaults to 0070 0000h (DSP ROM) on power-up. The address field is read/writeable after reset and can be modified to allow execution from an alternate location after a module level or local reset on the DSP. The HOST1CFG is shown in Figure 10-7 and described in Table 10-11.

Figure 10-7. Host 1 Configuration Register (HOST1CFG)



LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 10-11. Host 1 Configuration Register (HOST1CFG) Field Descriptions

Bit	Field	Value	Description
31-10	DSP_ISTP_RST_VAL	0-3F FFFFh	**DSP boot address vector.**
9-1	Reserved	0	Reserved
0	BOOTRDY		**DSP boot ready bit allowing DSP to boot.**
		0	DSP held in reset mode.
		1	DSP released from wait in reset mode.

Copyright © 2013–2016, Texas Instruments Incorporated

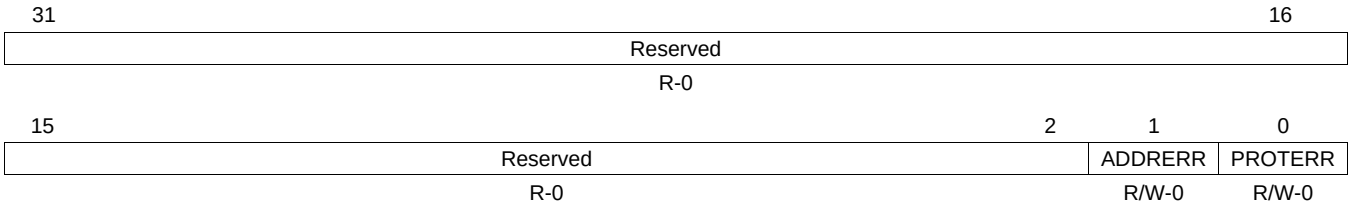
10.5.7 Interrupt Registers

The interrupt registers are a set of registers that provide control for the address and protection violation error interrupt generated by the SYSCFG module when there is an address or protection violation to the module's memory-mapped register address space. This includes enable control, interrupt set and clear control, and end of interrupt (EOI) control.

10.5.7.1 Interrupt Raw Status/Set Register (IRAWSTAT)

The interrupt raw status/set register (IRAWSTAT) shows the interrupt status before enabling the interrupt and allows setting of the interrupt status. The IRAWSTAT is shown in Figure 10-8 and described in Table 10-12.

Figure 10-8. Interrupt Raw Status/Set Register (IRAWSTAT)



LEGEND: R/W = Read/Write; R = Read only; *-n* = value after reset

Table 10-12. Interrupt Raw Status/Set Register (IRAWSTAT) Field Descriptions

Bit	Field	Value	Description
31-2	Reserved	0	Reserved. Always read 0.
1	ADDRERR		**Addressing violation error.** Reading this bit field reflects the raw status of the interrupt before enabling.
		0	Indicates the interrupt is not set. Writing 0 has no effect.
		1	Indicates the interrupt is set. Writing 1 sets the status.
0	PROTERR		**Protection violation error.** Reading this bit field reflects the raw status of the interrupt before enabling.
		0	Indicates the interrupt is not set. Writing 0 has no effect.
		1	Indicates the interrupt is set. Writing 1 sets the status.

Copyright © 2013–2016, Texas Instruments Incorporated

10.5.7.2 Interrupt Enable Status/Clear Register (IENSTAT)

The interrupt enable status/clear register (IENSTAT) shows the status of enabled interrupt and allows clearing of the interrupt status. The IENSTAT is shown in Figure 10-9 and described in Table 10-13.

Figure 10-9. Interrupt Enable Status/Clear Register (IENSTAT)

31	16
Reserved	
R-0	

15	2	1	0
Reserved		ADDRERR	PROTERR
R-0		R/W-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; *-n* = value after reset

Table 10-13. Interrupt Enable Status/Clear Register (IENSTAT) Field Descriptions

Bit	Field	Value	Description
31-2	Reserved	0	Reserved. Always read 0.
1	ADDRERR		**Addressing violation error.** Reading this bit field reflects the interrupt enabled status.
		0	Indicates the interrupt is not set. Writing 0 has no effect.
		1	Indicates the interrupt is set. Writing 1 clears the status.
0	PROTERR		**Protection violation error.** Reading this bit field reflects the interrupt enabled status.
		0	Indicates the interrupt is not set. Writing 0 has no effect.
		1	Indicates the interrupt is set. Writing 1 clears the status.

Copyright © 2013–2016, Texas Instruments Incorporated

10.5.7.3 Interrupt Enable Register (IENSET)

The interrupt enable register (IENSET) allows setting/enabling the interrupt for address and/or protection violation condition. It also shows the value of the register (whether or not interrupt is enabled). The IENSET is shown in Figure 10-10 and described in Table 10-14.

Figure 10-10. Interrupt Enable Register (IENSET)

31	16
Reserved	
R-0	

15	2	1	0
Reserved		ADDRERR_EN	PROTERR_EN
R-0		R/W-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 10-14. Interrupt Enable Register (IENSET) Field Descriptions

Bit	Field	Value	Description
31-2	Reserved	0	Reserved. Always read 0.
1	ADDRERR_EN		**Addressing violation error.**
		0	Writing a 0 has not effect.
		1	Writing a 1 enables this interrupt.
0	PROTERR_EN		**Protection violation error.**
		0	Writing a 0 has not effect.
		1	Writing a 1 enables this interrupt.

10.5.7.4 Interrupt Enable Clear Register (IENCLR)

The interrupt enable clear register (IENCLR) allows clearing/disable the interrupt for address and/or protection violation condition. It also shows the value of the interrupt enable register (IENSET). The IENCLR is shown in Figure 10-11 and described in Table 10-15.

Figure 10-11. Interrupt Enable Clear Register (IENCLR)

31	16
Reserved	
R-0	

15	2	1	0
Reserved		ADDRERR_CLR	PROTERR_CLR
R-0		R/W-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 10-15. Interrupt Enable Clear Register (IENCLR) Field Descriptions

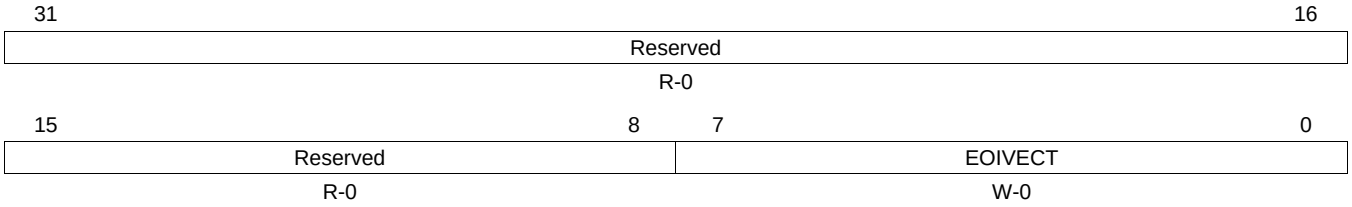
Bit	Field	Value	Description
31-2	Reserved	0	Reserved. Always read 0.
1	ADDRERR_CLR		**Addressing violation error.**
		0	Writing a 0 has not effect.
		1	Writing a 1 clears/disables this interrupt.
0	PROTERR_CLR		**Protection violation error.**
		0	Writing a 0 has not effect.
		1	Writing a 1 clears/disables this interrupt.

Copyright © 2013–2016, Texas Instruments Incorporated

10.5.7.5 End of Interrupt Register (EOI)

The end of interrupt register (EOI) is used in software to indicate completion of the interrupt servicing of the SYSCFG interrupt (for address/protection violation). It is required to write a value of 0 to the EOI register after the software has processed the SYSCFG interrupt, this acts as an acknowledgement of completion of the SYSCFG interrupt so that the module can reliably generate the subsequent interrupts. The EOI is shown in Figure 10-12 and described in Table 10-16.

Figure 10-12. End of Interrupt Register (EOI)



LEGEND: R = Read only; W = Write only; -*n* = value after reset

Table 10-16. End of Interrupt Register (EOI) Field Descriptions

Bit	Field	Value	Description
31-8	Reserved	0	Reserved. Always read 0.
7-0	EOIVECT	0-FFh	**EOI vector value.** Write the interrupt distribution value of the chip.

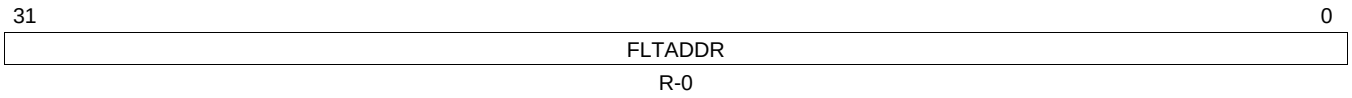
10.5.8 Fault Registers

The fault registers are a group of registers responsible for capturing the details on the faulty (address/protection violation errors) accesses, such as address and type of error.

10.5.8.1 Fault Address Register (FLTADDRR)

The fault address register (FLTADDRR) captures the address of the first transfer that causes the address or memory violation error. The FLTADDRR is shown in Figure 10-13 and described in Table 10-17.

Figure 10-13. Fault Address Register (FLTADDRR)



LEGEND: R = Read only; -*n* = value after reset

Table 10-17. Fault Address Register (FLTADDRR) Field Descriptions

Bit	Field	Value	Description
31-0	FLTADDR	0-FFFF FFFFh	**Fault address for the first fault transfer.**

Copyright © 2013–2016, Texas Instruments Incorporated

10.5.8.2 Fault Status Register (FLTSTAT)

The fault status register (FLTSTAT) holds/captures additional attributes and status of the first erroneous transaction. This includes things like the master id for the master that caused the address/memory violation error, details on whether it is a user or supervisor level read/write or execute fault. The FLTSTAT is shown in Figure 10-14 and described in Table 10-18.

Figure 10-14. Fault Status Register (FLTSTAT)

31	24	23	16
ID		MSTID	
R-0		R-0	

15	13	12	9	8	6	5	0
Reserved		PRIVID		Reserved		TYPE	
R-0		R-0		R-0		R-0	

LEGEND: R = Read only; -*n* = value after reset

Table 10-18. Fault Status Register (FLTSTAT) Field Descriptions

Bit	Field	Value	Description
31-24	ID	0-FFh	**Transfer ID of the first fault transfer.**
23-16	MSTID	0-FFh	**Master ID of the first fault transfer.**
15-13	Reserved	0	Reserved. Always read 0
12-9	PRIVID	0-Fh	**Privilege ID of the first fault transfer.**
8-6	Reserved	0	Reserved. Always read 0
5-0	TYPE		**Fault type of first fault transfer.**
		0	No transfer fault
		1h	User execute fault
		2h	User write fault
		3h	*Reserved*
		4h	User read fault
		5h-7h	*Reserved*
		8h	Supervisor execute fault
		9h-Fh	*Reserved*
		10h	Supervisor write fault
		11h-1Fh	*Reserved*
		20h	Supervisor read fault
		21h-3Fh	*Reserved*

Copyright © 2013–2016, Texas Instruments Incorporated

10.5.9 Master Priority Registers (MSTPRI0-MSTPRI2)

10.5.9.1 Master Priority 0 Register (MSTPRI0)

The master priority 0 register (MSTPRI0) is shown in Figure 10-15 and described in Table 10-19.

Figure 10-15. Master Priority 0 Register (MSTPRI0)

31	30 28	27	26 24	23	22 20	19	18 16
Rsvd	Reserved	Rsvd	Reserved	Rsvd	Reserved	Rsvd	Reserved
R/W-0	R/W-4h	R/W-0	R/W-4h	R/W-0	R/W-4h	R/W-0	R/W-4h

15	14 12	11	10 8	7	6 4	3	2 0
Rsvd	DSP_CFG	Rsvd	DSP_MDMA	Rsvd	Reserved	Rsvd	Reserved
R/W-0	R/W-2h	R-0	R/W-2h	R-0	R/W-2h	R-0	R/W-2h

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 10-19. Master Priority 0 Register (MSTPRI0) Field Descriptions

Bit	Field	Value	Description
31	Reserved	0	Reserved. Write the default value when modifying this register.
30-28	Reserved	4h	Reserved. Write the default value when modifying this register.
27	Reserved	0	Reserved. Write the default value when modifying this register.
26-24	Reserved	4h	Reserved. Write the default value when modifying this register.
23	Reserved	0	Reserved. Write the default value when modifying this register.
22-20	Reserved	4h	Reserved. Write the default value when modifying this register.
19	Reserved	0	Reserved. Write the default value when modifying this register.
18-16	Reserved	4h	Reserved. Write the default value when modifying this register.
15	Reserved	0	Reserved. Write the default value when modifying this register.
14-12	DSP_CFG	0-7h	**DSP CFG port priority.** Bit = 0 = priority 0 (highest); bit = 7h = priority 7 (lowest).
11	Reserved	0	Reserved. Always read as 0.
10-8	DSP_MDMA	0-7h	**DSP DMA port priority.** Bit = 0 = priority 0 (highest); bit = 7h = priority 7 (lowest).
7	Reserved	0	Reserved. Always read as 0.
6-4	Reserved	2h	Reserved. Write the default value when modifying this register.
3	Reserved	0	Reserved. Always read as 0.
2-0	Reserved	2h	Reserved. Write the default value when modifying this register.

Copyright © 2013–2016, Texas Instruments Incorporated

10.5.9.2 Master Priority 1 Register (MSTPRI1)

The master priority 1 register (MSTPRI1) is shown in Figure 10-16 and described in Table 10-20.

Figure 10-16. Master Priority 1 Register (MSTPRI1)

31	30 28	27	26 24	23	22 20	19	18 16
Rsvd	Reserved	Rsvd	Reserved	Rsvd	Reserved	Rsvd	Reserved
R/W-0	R/W-4h	R/W-0	R/W-4h	R/W-0	R/W-4h	R/W-0	R/W-4h

15	14 12	11	10 8	7	6 4	3	2 0
Rsvd	EDMATC1	Rsvd	EDMATC0	Rsvd	DMAX1	Rsvd	DMAX0
R/W-0	R/W-0	R-0	R/W-0	R-0	R/W-0	R-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; *-n* = value after reset

Table 10-20. Master Priority 1 Register (MSTPRI1) Field Descriptions

Bit	Field	Value	Description
31	Reserved	0	Reserved. Write the default value when modifying this register.
30-28	Reserved	4h	Reserved. Write the default value when modifying this register.
27	Reserved	0	Reserved. Write the default value when modifying this register.
26-24	Reserved	4h	Reserved. Write the default value when modifying this register.
23	Reserved	0	Reserved. Write the default value when modifying this register.
22-20	Reserved	4h	Reserved. Write the default value when modifying this register.
19	Reserved	0	Reserved. Write the default value when modifying this register.
18-16	Reserved	4h	Reserved. Write the default value when modifying this register.
15	Reserved	0	Reserved. Write the default value when modifying this register.
14-12	EDMATC1	0-7h	**EDMA3TC1 priority.** Bit = 0 = priority 0 (highest); bit = 7h = priority 7 (lowest).
11	Reserved	0	Reserved. Always read as 0.
10-8	EDMATC0	0-7h	**EDMA3TC0 priority.** Bit = 0 = priority 0 (highest); bit = 7h = priority 7 (lowest).
7	Reserved	0	Reserved. Always read as 0.
6-4	DMAX1	0-7h	**dMAX1 priority.** Bit = 0 = priority 0 (highest); bit = 7h = priority 7 (lowest).
3	Reserved	0	Reserved. Always read as 0.
2-0	DMAX0	0-7h	**dMAX0 priority.** Bit = 0 = priority 0 (highest); bit = 7h = priority 7 (lowest).

Copyright © 2013–2016, Texas Instruments Incorporated

10.5.9.3 Master Priority 2 Register (MSTPRI2)

The master priority 2 register (MSTPRI2) is shown in Figure 10-17 and described in Table 10-21.

Figure 10-17. Master Priority 2 Register (MSTPRI2)

31	30 28	27	26 24	23	22 20	19	18 16
Rsvd	Reserved	Rsvd	Reserved	Rsvd	Reserved	Rsvd	Reserved
R/W-0	R/W-5h	R/W-0	R/W-4h	R/W-0	R/W-6h	R/W-0	R/W-0

15	14 12	11	10 8	7	6 4	3	2 0
Rsvd	Reserved	Rsvd	Reserved	Rsvd	Reserved	Rsvd	Reserved
R/W-0	R/W-4h	R/W-0	R/W-4h	R/W-0	R/W-0	R/W-0	R/W-4h

LEGEND: R/W = Read/Write; -*n* = value after reset

Table 10-21. Master Priority 2 Register (MSTPRI2) Field Descriptions

Bit	Field	Value	Description
31	Reserved	0	Reserved. Write the default value when modifying this register.
30-28	Reserved	5h	Reserved. Write the default value when modifying this register.
27	Reserved	0	Reserved. Write the default value when modifying this register.
26-24	Reserved	4h	Reserved. Write the default value when modifying this register.
23	Reserved	0	Reserved. Write the default value when modifying this register.
22-20	Reserved	6h	Reserved. Write the default value when modifying this register.
19	Reserved	0	Reserved. Write the default value when modifying this register.
18-16	Reserved	0	Reserved. Write the default value when modifying this register.
15	Reserved	0	Reserved. Write the default value when modifying this register.
14-12	Reserved	4h	Reserved. Write the default value when modifying this register.
11	Reserved	0	Reserved. Write the default value when modifying this register.
10-8	Reserved	4h	Reserved. Write the default value when modifying this register.
7	Reserved	0	Reserved. Write the default value when modifying this register.
6-4	Reserved	0	Reserved. Write the default value when modifying this register.
3	Reserved	0	Reserved. Write the default value when modifying this register.
2-0	Reserved	4h	Reserved. Write the default value when modifying this register.

Copyright © 2013–2016, Texas Instruments Incorporated

10.5.10 Pin Multiplexing Control Registers (PINMUX0-PINMUX19)

Extensive use of pin multiplexing is used to accommodate the large number of peripheral functions in the smallest possible package. On the device, pin multiplexing can be controlled on a pin by pin basis. This is done by the pin multiplexing registers (PINMUX0-PINMUX19). Each pin that is multiplexed with several different functions has a corresponding 4-bit field in PINMUX*n*. Pin multiplexing selects which of several peripheral pin functions control the pins IO buffer output data and output enable values only. Note that the input from each pin is always routed to all of the peripherals that share the pin; the PINMUX registers have no effect on input from a pin. Hardware does not attempt to ensure that the proper pin multiplexing is selected for the peripherals or that interface mode is being used. Detailed information about the pin multiplexing and control is covered in the device-specific data manual.

10.5.10.1 Pin Multiplexing Control 0 Register (PINMUX0)

Figure 10-18. Pin Multiplexing Control 0 Register (PINMUX0)

31 28	27 24	23 20	19 16
PINMUX0_31_28	PINMUX0_27_24	PINMUX0_23_20	PINMUX0_19_16
R/W-0	R/W-0	R/W-0	R/W-0
15 12	**11 8**	**7 4**	**3 0**
PINMUX0_15_12	PINMUX0_11_8	PINMUX0_7_4	PINMUX0_3_0
R/W-0	R/W-0	R/W-0	R/W-0

LEGEND: R/W = Read/Write; -*n* = value after reset

Table 10-22. Pin Multiplexing Control 0 Register (PINMUX0) Field Descriptions

Bit	Field	PFP Pin	Value	Description
31-28	PINMUX0_31_28	—		**PINMUX0[31:28] Control**
			0	Pin is 3-stated.
			1h-Fh	*Reserved*
27-24	PINMUX0_27_24	—		**PINMUX0[27:24] Control**
			0	Pin is 3-stated.
			1h-Fh	*Reserved*
23-20	PINMUX0_23_20	—		**PINMUX0[23:20] Control**
			0	Pin is 3-stated.
			1h-Fh	*Reserved*
19-16	PINMUX0_19_16	—		**PINMUX0[19:16] Control**
			0	Pin is 3-stated.
			1h-Fh	*Reserved*
15-12	PINMUX0_15_12	—		**PINMUX0[15:12] Control**
			0	Pin is 3-stated.
			1h-Fh	*Reserved*
11-8	PINMUX0_11_8	—		**PINMUX0[11:8] Control**
			0	Pin is 3-stated.
			1h-Fh	*Reserved*
7-4	PINMUX0_7_4	—		**PINMUX0[7:4] Control**
			0	Pin is 3-stated.
			1h-Fh	*Reserved*
3-0	PINMUX0_3_0	—		**PINMUX0[3:0] Control**
			0-Fh	*Reserved*

Copyright © 2013–2016, Texas Instruments Incorporated

10.5.10.2 Pin Multiplexing Control 1 Register (PINMUX1)

Figure 10-19. Pin Multiplexing Control 1 Register (PINMUX1)

31 28	27 24	23 20	19 16
PINMUX1_31_28	PINMUX1_27_24	PINMUX1_23_20	PINMUX1_19_16
R/W-0	R/W-0	R/W-0	R/W-0
15 12	**11 8**	**7 4**	**3 0**
PINMUX1_15_12	PINMUX1_11_8	PINMUX1_7_4	PINMUX1_3_0
R/W-0	R/W-0	R/W-0	R/W-0

LEGEND: R/W = Read/Write; -*n* = value after reset

Table 10-23. Pin Multiplexing Control 1 Register (PINMUX1) Field Descriptions

Bit	Field	PFP Pin	Value	Description
31-28	PINMUX1_31_28	—		**PINMUX1[31:28] Control**
			0	Pin is 3-stated.
			1h-Fh	*Reserved*
27-24	PINMUX1_27_24	—		**PINMUX1[27:24] Control**
			0	Pin is 3-stated.
			1h-Fh	*Reserved*
23-20	PINMUX1_23_20	—		**PINMUX1[23:20] Control**
			0	Pin is 3-stated.
			1h-Fh	*Reserved*
19-16	PINMUX1_19_16	—		**PINMUX1[19:16] Control**
			0	Pin is 3-stated.
			1h-Fh	*Reserved*
15-12	PINMUX1_15_12	—		**PINMUX1[15:12] Control**
			0	Pin is 3-stated.
			1h-Fh	*Reserved*
11-8	PINMUX1_11_8	—		**PINMUX1[11:8] Control**
			0	Pin is 3-stated.
			1h-Fh	*Reserved*
7-4	PINMUX1_7_4	—		**PINMUX1[7:4] Control**
			0	Pin is 3-stated.
			1h-Fh	*Reserved*
3-0	PINMUX1_3_0	—		**PINMUX1[3:0] Control**
			0	Pin is 3-stated.
			1h-Fh	*Reserved*

Copyright © 2013–2016, Texas Instruments Incorporated

10.5.10.3 Pin Multiplexing Control 2 Register (PINMUX2)

Figure 10-20. Pin Multiplexing Control 2 Register (PINMUX2)

31 28	27 24	23 20	19 16
PINMUX2_31_28	PINMUX2_27_24	PINMUX2_23_20	PINMUX2_19_16
R/W-0	R/W-0	R/W-0	R/W-0
15 12	**11 8**	**7 4**	**3 0**
PINMUX2_15_12	PINMUX2_11_8	PINMUX2_7_4	PINMUX2_3_0
R/W-0	R/W-0	R/W-0	R/W-0

LEGEND: R/W = Read/Write; -*n* = value after reset

Table 10-24. Pin Multiplexing Control 2 Register (PINMUX2) Field Descriptions

Bit	Field	PFP Pin	Value	Description
31-28	PINMUX2_31_28	—		**PINMUX2[31:28] Control**
			0	Pin is 3-stated.
			1h-Fh	*Reserved*
27-24	PINMUX2_27_24	—		**PINMUX2[27:24] Control**
			0	Pin is 3-stated.
			1h-Fh	*Reserved*
23-20	PINMUX2_23_20	—		**PINMUX2[23:20] Control**
			0	Pin is 3-stated.
			1h-Fh	*Reserved*
19-16	PINMUX2_19_16	—		**PINMUX2[19:16] Control**
			0	Pin is 3-stated.
			1h-Fh	*Reserved*
15-12	PINMUX2_15_12	—		**PINMUX2[15:12] Control**
			0	Pin is 3-stated.
			1h-Fh	*Reserved*
11-8	PINMUX2_11_8	—		**PINMUX2[11:8] Control**
			0	Pin is 3-stated.
			1h-Fh	*Reserved*
7-4	PINMUX2_7_4	—		**PINMUX2[7:4] Control**
			0	Pin is 3-stated.
			1h-Fh	*Reserved*
3-0	PINMUX2_3_0	—		**PINMUX2[3:0] Control**
			0	Pin is 3-stated.
			1h-Fh	*Reserved*

Copyright © 2013–2016, Texas Instruments Incorporated

10.5.10.4 Pin Multiplexing Control 3 Register (PINMUX3)

Figure 10-21. Pin Multiplexing Control 3 Register (PINMUX3)

31 28	27 24	23 20	19 16
PINMUX3_31_28	PINMUX3_27_24	PINMUX3_23_20	PINMUX3_19_16
R/W-0	R/W-0	R/W-0	R/W-0
15 12	**11 8**	**7 4**	**3 0**
PINMUX3_15_12	PINMUX3_11_8	PINMUX3_7_4	PINMUX3_3_0
R/W-0	R/W-0	R/W-0	R/W-0

LEGEND: R/W = Read/Write; -*n* = value after reset

Table 10-25. Pin Multiplexing Control 3 Register (PINMUX3) Field Descriptions

Bit	Field	PFP Pin	Value	Description
31-28	PINMUX3_31_28	—		**PINMUX3[31:28] Control**
			0	Pin is 3-stated.
			1h-Fh	*Reserved*
27-24	PINMUX3_27_24	—		**PINMUX3[27:24] Control**
			0	Pin is 3-stated.
			1h-Fh	*Reserved*
23-20	PINMUX3_23_20	—		**PINMUX3[23:20] Control**
			0	Pin is 3-stated.
			1h-Fh	*Reserved*
19-16	PINMUX3_19_16	—		**PINMUX3[19:16] Control**
			0	Pin is 3-stated.
			1h-Fh	*Reserved*
15-12	PINMUX3_15_12	—		**PINMUX3[15:12] Control**
			0	Pin is 3-stated.
			1h-Fh	*Reserved*
11-8	PINMUX3_11_8	—		**PINMUX3[11:8] Control**
			0	Pin is 3-stated.
			1h-Fh	*Reserved*
7-4	PINMUX3_7_4	—		**PINMUX3[7:4] Control**
			0	Pin is 3-stated.
			1h-Fh	*Reserved*
3-0	PINMUX3_3_0	—		**PINMUX3[3:0] Control**
			0	Pin is 3-stated.
			1h-Fh	*Reserved*

Copyright © 2013–2016, Texas Instruments Incorporated

10.5.10.5 Pin Multiplexing Control 4 Register (PINMUX4)

Figure 10-22. Pin Multiplexing Control 4 Register (PINMUX4)

31 28	27 24	23 20	19 16
PINMUX4_31_28	PINMUX4_27_24	PINMUX4_23_20	PINMUX4_19_16
R/W-0	R/W-0	R/W-0	R/W-0
15 12	**11 8**	**7 4**	**3 0**
PINMUX4_15_12	PINMUX4_11_8	PINMUX4_7_4	PINMUX4_3_0
R/W-0	R/W-0	R/W-0	R/W-0

LEGEND: R/W = Read/Write; -*n* = value after reset

Table 10-26. Pin Multiplexing Control 4 Register (PINMUX4) Field Descriptions

Bit	Field	PFP Pin	Value	Description
31-28	PINMUX4_31_28	—		**PINMUX4[31:28] Control**
			0	Pin is 3-stated.
			1h-Fh	*Reserved*
27-24	PINMUX4_27_24	—		**PINMUX4[27:24] Control**
			0	Pin is 3-stated.
			1h-Fh	*Reserved*
23-20	PINMUX4_23_20	—		**PINMUX4[23:20] Control**
			0	Pin is 3-stated.
			1h-Fh	*Reserved*
19-16	PINMUX4_19_16	—		**PINMUX4[19:16] Control**
			0	Pin is 3-stated.
			1h-Fh	*Reserved*
15-12	PINMUX4_15_12	—		**PINMUX4[15:12] Control**
			0	Pin is 3-stated.
			1h-Fh	*Reserved*
11-8	PINMUX4_11_8	—		**PINMUX4[11:8] Control**
			0	Pin is 3-stated.
			1h-Fh	*Reserved*
7-4	PINMUX4_7_4	—		**PINMUX4[7:4] Control**
			0	Pin is 3-stated.
			1h-Fh	*Reserved*
3-0	PINMUX4_3_0	—		**PINMUX4[3:0] Control**
			0	Pin is 3-stated.
			1h-Fh	*Reserved*

Copyright © 2013–2016, Texas Instruments Incorporated

10.5.10.6 Pin Multiplexing Control 5 Register (PINMUX5)

Figure 10-23. Pin Multiplexing Control 5 Register (PINMUX5)

31 28	27 24	23 20	19 16
PINMUX5_31_28	PINMUX5_27_24	PINMUX5_23_20	PINMUX5_19_16
R/W-0	R/W-0	R/W-0	R/W-0
15 12	**11 8**	**7 4**	**3 0**
PINMUX5_15_12	PINMUX5_11_8	PINMUX5_7_4	PINMUX5_3_0
R/W-0	R/W-0	R/W-0	R/W-0

LEGEND: R/W = Read/Write; -*n* = value after reset

Table 10-27. Pin Multiplexing Control 5 Register (PINMUX5) Field Descriptions

Bit	Field	PFP Pin	Value	Description
31-28	PINMUX5_31_28	—		**PINMUX5[31:28] Control**
			0	Pin is 3-stated.
			1h-Fh	*Reserved*
27-24	PINMUX5_27_24	—		**PINMUX5[27:24] Control**
			0	Pin is 3-stated.
			1h-Fh	*Reserved*
23-20	PINMUX5_23_20	—		**PINMUX5[23:20] Control**
			0	Pin is 3-stated.
			1h-Fh	*Reserved*
19-16	PINMUX5_19_16	—		**PINMUX5[19:16] Control**
			0	Pin is 3-stated.
			1h-Fh	*Reserved*
15-12	PINMUX5_15_12	—		**PINMUX5[15:12] Control**
			0	Pin is 3-stated.
			1h-Fh	*Reserved*
11-8	PINMUX5_11_8	—		**PINMUX5[11:8] Control**
			0	Pin is 3-stated.
			1h-Fh	*Reserved*
7-4	PINMUX5_7_4	—		**PINMUX5[7:4] Control**
			0	Pin is 3-stated.
			1h-Fh	*Reserved*
3-0	PINMUX5_3_0	—		**PINMUX5[3:0] Control**
			0	Pin is 3-stated.
			1h-Fh	*Reserved*

Copyright © 2013–2016, Texas Instruments Incorporated

10.5.10.7 Pin Multiplexing Control 6 Register (PINMUX6)

Figure 10-24. Pin Multiplexing Control 6 Register (PINMUX6)

31 28	27 24	23 20	19 16
PINMUX6_31_28	PINMUX6_27_24	PINMUX6_23_20	PINMUX6_19_16
R/W-0	R/W-0	R/W-0	R/W-0
15 12	**11 8**	**7 4**	**3 0**
PINMUX6_15_12	PINMUX6_11_8	PINMUX6_7_4	PINMUX6_3_0
R/W-0	R/W-0	R/W-0	R/W-0

LEGEND: R/W = Read/Write; -*n* = value after reset

Table 10-28. Pin Multiplexing Control 6 Register (PINMUX6) Field Descriptions

Bit	Field	PFP Pin	Value	Description
31-28	PINMUX6_31_28	34		**GP6[14] Control**
			0	Pin is 3-stated.
			1h-7h	*Reserved*
			8h	Selects Function GP6[14]
			9h-Fh	*Reserved*
27-24	PINMUX6_27_24	35		**GP6[13] Control**
			0	Pin is 3-stated.
			1h-7h	*Reserved*
			8h	Selects Function GP6[13]
			9h-Fh	*Reserved*
23-20	PINMUX6_23_20	36		**GP6[12] Control**
			0	Pin is 3-stated.
			1h-7h	*Reserved*
			8h	Selects Function GP6[12]
			9h-Fh	*Reserved*
19-16	PINMUX6_19_16	37		**GP6[11] Control**
			0	Pin is 3-stated.
			1h-7h	*Reserved*
			8h	Selects Function GP6[11]
			9h-Fh	*Reserved*
15-12	PINMUX6_15_12	39		**GP6[10] Control**
			0	Pin is 3-stated.
			1h-7h	*Reserved*
			8h	Selects Function GP6[10]
			9h-Fh	*Reserved*
11-8	PINMUX6_11_8	40		**GP6[9] Control**
			0	Pin is 3-stated.
			1h-7h	*Reserved*
			8h	Selects Function GP6[9]
			9h-Fh	*Reserved*
7-4	PINMUX6_7_4	—		**PINMUX6[7:4] Control**
			0	Pin is 3-stated.
			1h-Fh	*Reserved*

Copyright © 2013–2016, Texas Instruments Incorporated

Table 10-28. Pin Multiplexing Control 6 Register (PINMUX6) Field Descriptions (continued)

Bit	Field	PFP Pin	Value	Description
3-0	PINMUX6_3_0	—		**PINMUX6[3:0] Control**
			0	Pin is 3-stated.
			1h-Fh	*Reserved*

Copyright © 2013–2016, Texas Instruments Incorporated

10.5.10.8 Pin Multiplexing Control 7 Register (PINMUX7)

Figure 10-25. Pin Multiplexing Control 7 Register (PINMUX7)

31 28	27 24	23 20	19 16
PINMUX7_31_28	PINMUX7_27_24	PINMUX7_23_20	PINMUX7_19_16
R/W-0	R/W-0	R/W-0	R/W-0
15 12	**11 8**	**7 4**	**3 0**
PINMUX7_15_12	PINMUX7_11_8	PINMUX7_7_4	PINMUX7_3_0
R/W-0	R/W-0	R/W-0	R/W-0

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 10-29. Pin Multiplexing Control 7 Register (PINMUX7) Field Descriptions

Bit	Field	PFP Pin	Value	Description
31-28	PINMUX7_31_28	9		**$\overline{\text{SPI0_SCS[0]}}$/$\overline{\text{UART0_RTS}}$/EQEP0B/GP5[4]/BOOT[4] Control**
			0	Pin is 3-stated.
			1h	Selects Function $\overline{\text{SPI0_SCS[0]}}$
			2h	Selects Function $\overline{\text{UART0_RTS}}$
			3h	*Reserved*
			4h	Selects Function EQEP0B
			5h-7h	*Reserved*
			8h	Selects Function GP5[4]/BOOT[4]
			9h-Fh	*Reserved*
27-24	PINMUX7_27_24	12		**$\overline{\text{SPI0_ENA}}$/$\overline{\text{UART0_CTS}}$/EQEP0A/GP5[3]/BOOT[3] Control**
			0	Pin is 3-stated.
			1h	Selects Function $\overline{\text{SPI0_ENA}}$
			2h	Selects Function $\overline{\text{UART0_CTS}}$
			3h	*Reserved*
			4h	Selects Function EQEP0A
			5h-7h	*Reserved*
			8h	Selects Function GP5[3]/BOOT[3]
			9h-Fh	*Reserved*
23-20	PINMUX7_23_20	11		**SPI0_CLK/GP5[2]/BOOT[2] Control**
			0	Pin is 3-stated.
			1h	Selects Function SPI0_CLK
			2h-7h	*Reserved*
			8h	Selects Function GP5[2]/BOOT[2]
			9h-Fh	*Reserved*
19-16	PINMUX7_19_16	18		**SPI0_SIMO[0]/EQEP0S/GP5[1]/BOOT[1] Control**
			0	Pin is 3-stated.
			1h	Selects Function SPI0_SIMO[0]
			2h	Selects Function EQEP0S
			3h-7h	*Reserved*
			8h	Selects Function GP5[1]/BOOT[1]
			9h-Fh	*Reserved*

Copyright © 2013–2016, Texas Instruments Incorporated

Table 10-29. Pin Multiplexing Control 7 Register (PINMUX7) Field Descriptions (continued)

Bit	Field	PFP Pin	Value	Description
15-12	PINMUX7_15_12	17		**SPI0_SOMI[0]/EQEP0I/GP5[0]/BOOT[0] Control**
			0	Pin is 3-stated.
			1h	Selects Function SPI0_SOMI[0]
			2h	Selects Function EQEP0I
			3h-7h	*Reserved*
			8h	Selects Function GP5[0]/BOOT[0]
			9h-Fh	*Reserved*
11-8	PINMUX7_11_8	—		**PINMUX7[11:8] Control**
			0	Pin is 3-stated.
			1h-Fh	*Reserved*
7-4	PINMUX7_7_4	—		**PINMUX7[7:4] Control**
			0	Pin is 3-stated.
			1h-Fh	*Reserved*
3-0	PINMUX7_3_0	33		**GP6[15] Control**
			0	Pin is 3-stated.
			1h-7h	*Reserved*
			8h	Selects Function GP6[15]
			9h-Fh	*Reserved*

Copyright © 2013–2016, Texas Instruments Incorporated

10.5.10.9 Pin Multiplexing Control 8 Register (PINMUX8)

Figure 10-26. Pin Multiplexing Control 8 Register (PINMUX8)

31 28	27 24	23 20	19 16
PINMUX8_31_28	PINMUX8_27_24	PINMUX8_23_20	PINMUX8_19_16
R/W-0	R/W-0	R/W-0	R/W-0
15 12	**11 8**	**7 4**	**3 0**
PINMUX8_15_12	PINMUX8_11_8	PINMUX8_7_4	PINMUX8_3_0
R/W-0	R/W-0	R/W-0	R/W-0

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 10-30. Pin Multiplexing Control 8 Register (PINMUX8) Field Descriptions

Bit	Field	PFP Pin	Value	Description
31-28	PINMUX8_31_28	7		**$\overline{\text{SPI1_ENA}}$/UART2_RXD/GP5[12] Control**
			0	Pin is 3-stated.
			1h	Selects Function $\overline{\text{SPI1_ENA}}$
			2h	Selects Function UART2_RXD
			3h-7h	*Reserved*
			8h	Selects Function GP5[12]
			9h-Fh	*Reserved*
27-24	PINMUX8_27_24	6		**AXR1[11]/GP5[11] Control**
			0	Pin is 3-stated.
			1h	Selects Function AXR1[11]
			2h-7h	*Reserved*
			8h	Selects Function GP5[11]
			9h-Fh	*Reserved*
23-20	PINMUX8_23_20	4		**AXR1[10]/GP5[10] Control**
			0	Pin is 3-stated.
			1h	Selects Function AXR1[10]
			2h-7h	*Reserved*
			8h	Selects Function GP5[10]
			9h-Fh	*Reserved*
19-16	PINMUX8_19_16	3		**UART0_TXD/I2C0_SCL/TM64P0_OUT12/GP5[9]/BOOT[9] Control**
			0	Pin is 3-stated.
			1h	Selects Function UART0_TXD
			2h	Selects Function I2C0_SCL
			3h	*Reserved*
			4h	Selects Function TM64P0_OUT12
			5h-7h	*Reserved*
			8h	Selects Function GP5[9]/BOOT[9]
			9h-Fh	*Reserved*

Copyright © 2013–2016, Texas Instruments Incorporated

Table 10-30. Pin Multiplexing Control 8 Register (PINMUX8) Field Descriptions (continued)

Bit	Field	PFP Pin	Value	Description
15-12	PINMUX8_15_12	2		**UART0_RXD/I2C0_SDA/TM64P0_IN12/GP5[8]/BOOT[8] Control**
			0	Pin is 3-stated.
			1h	Selects Function UART0_RXD
			2h	Selects Function I2C0_SDA
			3h	*Reserved*
			4h	Selects Function TM64P0_IN12
			5h-7h	*Reserved*
			8h	Selects Function GP5[8]/BOOT[8]
			9h-Fh	*Reserved*
11-8	PINMUX8_11_8	16		**SPI1_CLK/GP5[7]/BOOT[7] Control**
			0	Pin is 3-stated.
			1h	Selects Function SPI1_CLK
			2h-7h	*Reserved*
			8h	Selects Function GP5[7]/BOOT[7]
			9h-Fh	*Reserved*
7-4	PINMUX8_7_4	14		**SPI1_SIMO[0]/I2C1_SDA/GP5[6]/BOOT[6] Control**
			0	Pin is 3-stated.
			1h	Selects Function SPI1_SIMO[0]
			2h	Selects Function I2C1_SDA
			3h-7h	*Reserved*
			8h	Selects Function GP5[6]/BOOT[6]
			9h-Fh	*Reserved*
3-0	PINMUX8_3_0	13		**SPI1_SOMI[0]/I2C1_SCL/GP5[5]/BOOT[5] Control**
			0	Pin is 3-stated.
			1h	Selects Function SPI1_SOMI[0]
			2h	Selects Function I2C1_SCL
			3h-7h	*Reserved*
			8h	Selects Function GP5[5]/BOOT[5]
			9h-Fh	*Reserved*

Copyright © 2013–2016, Texas Instruments Incorporated

10.5.10.10 Pin Multiplexing Control 9 Register (PINMUX9)

Figure 10-27. Pin Multiplexing Control 9 Register (PINMUX9)

31 28	27 24	23 20	19 16
PINMUX9_31_28	PINMUX9_27_24	PINMUX9_23_20	PINMUX9_19_16
R/W-0	R/W-0	R/W-0	R/W-0
15 12	**11 8**	**7 4**	**3 0**
PINMUX9_15_12	PINMUX9_11_8	PINMUX9_7_4	PINMUX9_3_0
R/W-0	R/W-0	R/W-0	R/W-0

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 10-31. Pin Multiplexing Control 9 Register (PINMUX9) Field Descriptions

Bit	Field	PFP Pin	Value	Description
31-28	PINMUX9_31_28	60		**AFSR0/GP3[12] Control**
			0	Pin is 3-stated.
			1h	Selects Function AFSR0
			2h-7h	*Reserved*
			8h	Selects Function GP3[12]
			9h-Fh	*Reserved*
27-24	PINMUX9_27_24	59		**ACLKR0/ECAP1/APWM1/GP2[15] Control**
			0	Pin is 3-stated.
			1h	Selects Function ACLKR0
			2h	Selects Function ECAP1/APWM1
			3h-7h	*Reserved*
			8h	Selects Function GP2[15]
			9h-Fh	*Reserved*
23-20	PINMUX9_23_20	58		**AHCLKR0/GP2[14]/BOOT[11] Control**
			0	Pin is 3-stated.
			1h	Selects Function AHCLKR0
			2h-7h	*Reserved*
			8h	Selects Function GP2[14]/BOOT[11]
			9h-Fh	*Reserved*
19-16	PINMUX9_19_16	56		**AFSX0/GP2[13]/BOOT[10] Control**
			0	Pin is 3-stated.
			1h	Selects Function AFSX0
			2h-7h	*Reserved*
			8h	Selects Function GP2[13]/BOOT[10]
			9h-Fh	*Reserved*
15-12	PINMUX9_15_12	55		**ACLKX0/ECAP0/APWM0/GP2[12] Control**
			0	Pin is 3-stated.
			1h	Selects Function ACLKX0
			2h	Selects Function ECAP0/APWM0
			3h-7h	*Reserved*
			8h	Selects Function GP2[12]
			9h-Fh	*Reserved*

Copyright © 2013–2016, Texas Instruments Incorporated

Table 10-31. Pin Multiplexing Control 9 Register (PINMUX9) Field Descriptions (continued)

Bit	Field	PFP Pin	Value	Description
11-8	PINMUX9_11_8	54		**AHCLKX0/AHCLKX2/GP2[11] Control**
			0	Pin is 3-stated.
			1h	Selects Function AHCLKX0
			2h	Selects Function AHCLKX2
			3h-7h	*Reserved*
			8h	Selects Function GP2[11]
			9h-Fh	*Reserved*
7-4	PINMUX9_7_4	—		**PINMUX9[7:4] Control**
			0	Pin is 3-stated.
			1h-Fh	*Reserved*
3-0	PINMUX9_3_0	8		**$\overline{\text{SPI1_SCS[0]}}$/UART2_TXD/GP5[13] Control**
			0	Pin is 3-stated.
			1h	Selects Function $\overline{\text{SPI1_SCS[0]}}$
			2h	Selects Function UART2_TXD
			3h-7h	*Reserved*
			8h	Selects Function GP5[13]
			9h-Fh	*Reserved*

Copyright © 2013–2016, Texas Instruments Incorporated

10.5.10.11 Pin Multiplexing Control 10 Register (PINMUX10)

Figure 10-28. Pin Multiplexing Control 10 Register (PINMUX10)

31 28	27 24	23 20	19 16
PINMUX10_31_28	PINMUX10_27_24	PINMUX10_23_20	PINMUX10_19_16
R/W-0	R/W-0	R/W-0	R/W-0
15 12	**11 8**	**7 4**	**3 0**
PINMUX10_15_12	PINMUX10_11_8	PINMUX10_7_4	PINMUX10_3_0
R/W-0	R/W-0	R/W-0	R/W-0

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 10-32. Pin Multiplexing Control 10 Register (PINMUX10) Field Descriptions

Bit	Field	PFP Pin	Value	Description
31-28	PINMUX10_31_28	48		**AXR0[6]/ACLKR2/GP3[6] Control**
			0	Pin is 3-stated.
			1h	Selects Function AXR0[6]
			2h-3h	*Reserved*
			4h	Selects Function ACLKR2
			5h-7h	*Reserved*
			8h	Selects Function GP3[6]
			9h-Fh	*Reserved*
27-24	PINMUX10_27_24	47		**AXR0[5]/AFSX2/GP3[5] Control**
			0	Pin is 3-stated.
			1h	Selects Function AXR0[5]
			2h-3h	*Reserved*
			4h	Selects Function AFSX2
			5h-7h	*Reserved*
			8h	Selects Function GP3[5]
			9h-Fh	*Reserved*
23-20	PINMUX10_23_20	46		**AXR0[4]/AXR2[1]/GP3[4] Control**
			0	Pin is 3-stated.
			1h	Selects Function AXR0[4]
			2h-3h	*Reserved*
			4h	Selects Function AXR2[1]
			5h-7h	*Reserved*
			8h	Selects Function GP3[4]
			9h-Fh	*Reserved*
19-16	PINMUX10_19_16	45		**AXR0[3]/AXR2[2]/GP3[3] Control**
			0	Pin is 3-stated.
			1h	Selects Function AXR0[3]
			2h-3h	*Reserved*
			4h	Selects Function AXR2[2]
			5h-7h	*Reserved*
			8h	Selects Function GP3[3]
			9h-Fh	*Reserved*

Copyright © 2013–2016, Texas Instruments Incorporated

Table 10-32. Pin Multiplexing Control 10 Register (PINMUX10) Field Descriptions (continued)

Bit	Field	PFP Pin	Value	Description
15-12	PINMUX10_15_12	43		**AXR0[2]/AXR2[3]/GP3[2] Control**
			0	Pin is 3-stated.
			1h	Selects Function AXR0[2]
			2h-3h	*Reserved*
			4h	Selects Function AXR2[3]
			5h-7h	*Reserved*
			8h	Selects Function GP3[2]
			9h-Fh	*Reserved*
11-8	PINMUX10_11_8	42		**AXR0[1]/ACLKX2/GP3[1] Control**
			0	Pin is 3-stated.
			1h	Selects Function AXR0[1]
			2h-3h	*Reserved*
			4h	Selects Function ACLKX2
			5h-7h	*Reserved*
			8h	Selects Function GP3[1]
			9h-Fh	*Reserved*
7-4	PINMUX10_7_4	41		**AXR0[0]/AFSR2/GP3[0] Control**
			0	Pin is 3-stated.
			1h	Selects Function AXR0[0]
			2h-3h	*Reserved*
			4h	Selects Function AFSR2
			5h-7h	*Reserved*
			8h	Selects Function GP3[0]
			9h-Fh	*Reserved*
3-0	PINMUX10_3_0	—		**PINMUX10[3:0] Control**
			0-Fh	*Reserved*

Copyright © 2013–2016, Texas Instruments Incorporated

10.5.10.12 Pin Multiplexing Control 11 Register (PINMUX11)

Figure 10-29. Pin Multiplexing Control 11 Register (PINMUX11)

31 28	27 24	23 20	19 16
PINMUX11_31_28	PINMUX11_27_24	PINMUX11_23_20	PINMUX11_19_16
R/W-0	R/W-0	R/W-0	R/W-0
15 12	**11 8**	**7 4**	**3 0**
PINMUX11_15_12	PINMUX11_11_8	PINMUX11_7_4	PINMUX11_3_0
R/W-0	R/W-0	R/W-0	R/W-0

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 10-33. Pin Multiplexing Control 11 Register (PINMUX11) Field Descriptions

Bit	Field	PFP Pin	Value	Description
31-28	PINMUX11_31_28	76		**AFSX1/EPWMSYNCI/EPWMSYNCO/GP4[10] Control**
			0	Pin is 3-stated.
			1h	Selects Function AFSX1
			2h	Selects Function EPWMSYNCI
			3h	*Reserved*
			4h	Selects Function EPWMSYNCO
			5h-7h	*Reserved*
			8h	Selects Function GP4[10]
			9h-Fh	*Reserved*
27-24	PINMUX11_27_24	75		**ACLKX1/EPWM0A/GP3[15] Control**
			0	Pin is 3-stated.
			1h	Selects Function ACLKX1
			2h	Selects Function EPWM0A
			3h-7h	*Reserved*
			8h	Selects Function GP3[15]
			9h-Fh	*Reserved*
23-20	PINMUX11_23_20	74		**AHCLKX1/EPWM0B/GP3[14] Control**
			0	Pin is 3-stated.
			1h	Selects Function AHCLKX1
			2h	Selects Function EPWM0B
			3h-7h	*Reserved*
			8h	Selects Function GP3[14]
			9h-Fh	*Reserved*
19-16	PINMUX11_19_16	—		**PINMUX11[19:16] Control**
			0	Pin is 3-stated.
			1h-Fh	*Reserved*

Copyright © 2013–2016, Texas Instruments Incorporated

Table 10-33. Pin Multiplexing Control 11 Register (PINMUX11) Field Descriptions (continued)

Bit	Field	PFP Pin	Value	Description
15-12	PINMUX11_15_12	53		**UART1_TXD/AXR0[10]/GP3[10] Control**
			0	Pin is 3-stated.
			1h	Selects Function UART1_TXD
			2h	Selects Function AXR0[10]
			3h-7h	*Reserved*
			8h	Selects Function GP3[10]
			9h-Fh	*Reserved*
11-8	PINMUX11_11_8	52		**UART1_RXD/AXR0[9]/GP3[9] Control**
			0	Pin is 3-stated.
			1h	Selects Function UART1_RXD
			2h	Selects Function AXR0[9]
			3h-7h	*Reserved*
			8h	Selects Function GP3[9]
			9h-Fh	*Reserved*
7-4	PINMUX11_7_4	51		**AXR0[8]/GP3[8] Control**
			0	Pin is 3-stated.
			1h	Selects Function AXR0[8]
			2h-7h	*Reserved*
			8h	Selects Function GP3[8]
			9h-Fh	*Reserved*
3-0	PINMUX11_3_0	50		**AXR0[7]/GP3[7] Control**
			0	Pin is 3-stated.
			1h	Selects Function AXR0[7]
			2h-7h	*Reserved*
			8h	Selects Function GP3[7]
			9h-Fh	*Reserved*

Copyright © 2013–2016, Texas Instruments Incorporated

10.5.10.13 Pin Multiplexing Control 12 Register (PINMUX12)

Figure 10-30. Pin Multiplexing Control 12 Register (PINMUX12)

31 28	27 24	23 20	19 16
PINMUX12_31_28	PINMUX12_27_24	PINMUX12_23_20	PINMUX12_19_16
R/W-0	R/W-0	R/W-0	R/W-0
15 12	**11 8**	**7 4**	**3 0**
PINMUX12_15_12	PINMUX12_11_8	PINMUX12_7_4	PINMUX12_3_0
R/W-0	R/W-0	R/W-0	R/W-0

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 10-34. Pin Multiplexing Control 12 Register (PINMUX12) Field Descriptions

Bit	Field	PFP Pin	Value	Description
31-28	PINMUX12_31_28	—		**PINMUX12[31:28] Control**
			0	Pin is 3-stated.
			1h-Fh	*Reserved*
27-24	PINMUX12_27_24	—		**PINMUX12[27:24] Control**
			0	Pin is 3-stated.
			1h-Fh	*Reserved*
23-20	PINMUX12_23_20	—		**PINMUX12[23:20] Control**
			0	Pin is 3-stated.
			1h-Fh	*Reserved*
19-16	PINMUX12_19_16	1		**AXR1[0]/GP4[0] Control**
			0	Pin is 3-stated.
			1h	Selects Function AXR1[0]
			2h-7h	*Reserved*
			8h	Selects Function GP4[0]
			9h-Fh	*Reserved*
15-12	PINMUX12_15_12	—		**PINMUX12[15:12] Control**
			0	Pin is 3-stated.
			1h-Fh	*Reserved*
11-8	PINMUX12_11_8	79		**AFSR1/GP4[13] Control**
			0	Pin is 3-stated.
			1h	Selects Function AFSR1
			2h-7h	*Reserved*
			8h	Selects Function GP4[13]
			9h-Fh	*Reserved*
7-4	PINMUX12_7_4	78		**ACLKR1/ECAP2/APWM2/GP4[12] Control**
			0	Pin is 3-stated.
			1h	Selects Function ACLKR1
			2h	Selects Function ECAP2/APWM2
			3h-7h	*Reserved*
			8h	Selects Function GP4[12]
			9h-Fh	*Reserved*
3-0	PINMUX12_3_0	—		**PINMUX12[3:0] Control**
			0	Pin is 3-stated.
			1h-Fh	*Reserved*

Copyright © 2013–2016, Texas Instruments Incorporated

10.5.10.14 Pin Multiplexing Control 13 Register (PINMUX13)

Figure 10-31. Pin Multiplexing Control 13 Register (PINMUX13)

31 28	27 24	23 20	19 16
PINMUX13_31_28	PINMUX13_27_24	PINMUX13_23_20	PINMUX13_19_16
R/W-0	R/W-0	R/W-0	R/W-0
15 12	**11 8**	**7 4**	**3 0**
PINMUX13_15_12	PINMUX13_11_8	PINMUX13_7_4	PINMUX13_3_0
R/W-0	R/W-0	R/W-0	R/W-0

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 10-35. Pin Multiplexing Control 13 Register (PINMUX13) Field Descriptions

Bit	Field	PFP Pin	Value	Description
31-28	PINMUX13_31_28	21		**GP0[1] Control**
			0	Pin is 3-stated.
			1h-7h	*Reserved*
			8h	Selects Function GP0[1]
			9h-Fh	*Reserved*
27-24	PINMUX13_27_24	20		**GP0[0]/BOOT[12] Control**
			0	Pin is 3-stated.
			1h-7h	*Reserved*
			8h	Selects Function GP0[0]/BOOT[12]
			9h-Fh	*Reserved*
23-20	PINMUX13_23_20	—		**PINMUX13[23:20] Control**
			0	Pin is 3-stated.
			1h-Fh	*Reserved*
19-16	PINMUX13_19_16	—		**PINMUX13[19:16] Control**
			0	Pin is 3-stated.
			1h-Fh	*Reserved*
15-12	PINMUX13_15_12	—		**PINMUX13[15:12] Control**
			0	Pin is 3-stated.
			1h-Fh	*Reserved*
11-8	PINMUX13_11_8	—		**PINMUX13[11:8] Control**
			0	Pin is 3-stated.
			1h-Fh	*Reserved*
7-4	PINMUX13_7_4	—		**PINMUX13[7:4] Control**
			0	Pin is 3-stated.
			1h-Fh	*Reserved*
3-0	PINMUX13_3_0	—		**PINMUX13[3:0] Control**
			0	Pin is 3-stated.
			1h-Fh	*Reserved*

Copyright © 2013–2016, Texas Instruments Incorporated

10.5.10.15 Pin Multiplexing Control 14 Register (PINMUX14)

Figure 10-32. Pin Multiplexing Control 14 Register (PINMUX14)

31 28	27 24	23 20	19 16
PINMUX14_31_28	PINMUX14_27_24	PINMUX14_23_20	PINMUX14_19_16
R/W-0	R/W-0	R/W-0	R/W-0
15 12	**11 8**	**7 4**	**3 0**
PINMUX14_15_12	PINMUX14_11_8	PINMUX14_7_4	PINMUX14_3_0
R/W-0	R/W-0	R/W-0	R/W-0

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 10-36. Pin Multiplexing Control 14 Register (PINMUX14) Field Descriptions

Bit	Field	PFP Pin	Value	Description
31-28	PINMUX14_31_28	—		**PINMUX14[31:28] Control**
			0	Pin is 3-stated.
			1h-Fh	*Reserved*
27-24	PINMUX14_27_24	—		**PINMUX14[27:24] Control**
			0	Pin is 3-stated.
			1h-Fh	*Reserved*
23-20	PINMUX14_23_20	29		**GP0[7]/BOOT[13] Control**
			0	Pin is 3-stated.
			1h-7h	*Reserved*
			8h	Selects Function GP0[7]/BOOT[13]
			9h-Fh	*Reserved*
19-16	PINMUX14_19_16	27		**GP0[6] Control**
			0	Pin is 3-stated.
			1h-7h	*Reserved*
			8h	Selects Function GP0[6]
			9h-Fh	*Reserved*
15-12	PINMUX14_15_12	26		**GP0[5] Control**
			0	Pin is 3-stated.
			1h-7h	*Reserved*
			8h	Selects Function GP0[5]
			9h-Fh	*Reserved*
11-8	PINMUX14_11_8	24		**GP0[4] Control**
			0	Pin is 3-stated.
			1h-7h	*Reserved*
			8h	Selects Function GP0[4]
			9h-Fh	*Reserved*
7-4	PINMUX14_7_4	23		**GP0[3] Control**
			0	Pin is 3-stated.
			1h-7h	*Reserved*
			8h	Selects Function GP0[3]
			9h-Fh	*Reserved*

Copyright © 2013–2016, Texas Instruments Incorporated

Table 10-36. Pin Multiplexing Control 14 Register (PINMUX14) Field Descriptions (continued)

Bit	Field	PFP Pin	Value	Description
3-0	PINMUX14_3_0	22		**GP0[2] Control**
			0	Pin is 3-stated.
			1h-7h	*Reserved*
			8h	Selects Function GP0[2]
			9h-Fh	*Reserved*

Copyright © 2013–2016, Texas Instruments Incorporated

10.5.10.16 Pin Multiplexing Control 15 Register (PINMUX15)

Figure 10-33. Pin Multiplexing Control 15 Register (PINMUX15)

31 28	27 24	23 20	19 16
PINMUX15_31_28	PINMUX15_27_24	PINMUX15_23_20	PINMUX15_19_16
R/W-0	R/W-0	R/W-0	R/W-0
15 12	**11 8**	**7 4**	**3 0**
PINMUX15_15_12	PINMUX15_11_8	PINMUX15_7_4	PINMUX15_3_0
R/W-0	R/W-0	R/W-0	R/W-0

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 10-37. Pin Multiplexing Control 15 Register (PINMUX15) Field Descriptions

Bit	Field	PFP Pin	Value	Description
31-28	PINMUX15_31_28	—		**PINMUX15[31:28] Control**
			0	Pin is 3-stated.
			1h-Fh	*Reserved*
27-24	PINMUX15_27_24	—		**PINMUX15[27:24] Control**
			0	Pin is 3-stated.
			1h-Fh	*Reserved*
23-20	PINMUX15_23_20	—		**PINMUX15[23:20] Control**
			0	Pin is 3-stated.
			1h-Fh	*Reserved*
19-16	PINMUX15_19_16	—		**PINMUX15[19:16] Control**
			0	Pin is 3-stated.
			1h-Fh	*Reserved*
15-12	PINMUX15_15_12	—		**PINMUX15[15:12] Control**
			0	Pin is 3-stated.
			1h-Fh	*Reserved*
11-8	PINMUX15_11_8	—		**PINMUX15[11:8] Control**
			0	Pin is 3-stated.
			1h-Fh	*Reserved*
7-4	PINMUX15_7_4	—		**PINMUX15[7:4] Control**
			0	Pin is 3-stated.
			1h-Fh	*Reserved*
3-0	PINMUX15_3_0	—		**PINMUX15[3:0] Control**
			0	Pin is 3-stated.
			1h-Fh	*Reserved*

Copyright © 2013–2016, Texas Instruments Incorporated

10.5.10.17 Pin Multiplexing Control 16 Register (PINMUX16)

Figure 10-34. Pin Multiplexing Control 16 Register (PINMUX16)

31 28	27 24	23 20	19 16
PINMUX16_31_28	PINMUX16_27_24	PINMUX16_23_20	PINMUX16_19_16
R/W-0	R/W-0	R/W-0	R/W-0
15 12	**11 8**	**7 4**	**3 0**
PINMUX16_15_12	PINMUX16_11_8	PINMUX16_7_4	PINMUX16_3_0
R/W-0	R/W-0	R/W-0	R/W-0

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 10-38. Pin Multiplexing Control 16 Register (PINMUX16) Field Descriptions

Bit	Field	PFP Pin	Value	Description
31-28	PINMUX16_31_28	—		**PINMUX16[31:28] Control**
			0	Pin is 3-stated.
			1h-Fh	*Reserved*
27-24	PINMUX16_27_24	—		**PINMUX16[27:24] Control**
			0	Pin is 3-stated.
			1h-Fh	*Reserved*
23-20	PINMUX16_23_20	—		**PINMUX16[23:20] Control**
			0	Pin is 3-stated.
			1h-Fh	*Reserved*
19-16	PINMUX16_19_16	—		**PINMUX16[19:16] Control**
			0	Pin is 3-stated.
			1h-Fh	*Reserved*
15-12	PINMUX16_15_12	—		**PINMUX16[15:12] Control**
			0	Pin is 3-stated.
			1h-Fh	*Reserved*
11-8	PINMUX16_11_8	—		**PINMUX16[11:8] Control**
			0	Pin is 3-stated.
			1h-Fh	*Reserved*
7-4	PINMUX16_7_4	—		**PINMUX16[7:4] Control**
			0	Pin is 3-stated.
			1h-Fh	*Reserved*
3-0	PINMUX16_3_0	—		**PINMUX16[3:0] Control**
			0	Pin is 3-stated.
			1h-Fh	*Reserved*

Copyright © 2013–2016, Texas Instruments Incorporated

10.5.10.18 Pin Multiplexing Control 17 Register (PINMUX17)

Figure 10-35. Pin Multiplexing Control 17 Register (PINMUX17)

31 28	27 24	23 20	19 16
PINMUX17_31_28	PINMUX17_27_24	PINMUX17_23_20	PINMUX17_19_16
R/W-0	R/W-0	R/W-0	R/W-0
15 12	**11 8**	**7 4**	**3 0**
PINMUX17_15_12	PINMUX17_11_8	PINMUX17_7_4	PINMUX17_3_0
R/W-0	R/W-0	R/W-0	R/W-0

LEGEND: R/W = Read/Write; -*n* = value after reset

Table 10-39. Pin Multiplexing Control 17 Register (PINMUX17) Field Descriptions

Bit	Field	PFP Pin	Value	Description
31-28	PINMUX17_31_28	—		**PINMUX17[31:28] Control**
			0	Pin is 3-stated.
			1h-Fh	*Reserved*
27-24	PINMUX17_27_24	—		**PINMUX17[27:24] Control**
			0	Pin is 3-stated.
			1h-Fh	*Reserved*
23-20	PINMUX17_23_20	—		**PINMUX17[23:20] Control**
			0	Pin is 3-stated.
			1h-Fh	*Reserved*
19-16	PINMUX17_19_16	—		**PINMUX17[19:16] Control**
			0	Pin is 3-stated.
			1h-Fh	*Reserved*
15-12	PINMUX17_15_12	—		**PINMUX17[15:12] Control**
			0	Pin is 3-stated.
			1h-Fh	*Reserved*
11-8	PINMUX17_11_8	—		**PINMUX17[11:8] Control**
			0	Pin is 3-stated.
			1h-Fh	*Reserved*
7-4	PINMUX17_7_4	—		**PINMUX17[7:4] Control**
			0	Pin is 3-stated.
			1h-Fh	*Reserved*
3-0	PINMUX17_3_0	—		**PINMUX17[3:0] Control**
			0	Pin is 3-stated.
			1h-Fh	*Reserved*

Copyright © 2013–2016, Texas Instruments Incorporated

10.5.10.19 Pin Multiplexing Control 18 Register (PINMUX18)

Figure 10-36. Pin Multiplexing Control 18 Register (PINMUX18)

31 28	27 24	23 20	19 16
PINMUX18_31_28	PINMUX18_27_24	PINMUX18_23_20	PINMUX18_19_16
R/W-0	R/W-0	R/W-0	R/W-0
15 12	**11 8**	**7 4**	**3 0**
PINMUX18_15_12	PINMUX18_11_8	PINMUX18_7_4	PINMUX18_3_0
R/W-0	R/W-0	R/W-0	R/W-0

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 10-40. Pin Multiplexing Control 18 Register (PINMUX18) Field Descriptions

Bit	Field	PFP Pin	Value	Description
31-28	PINMUX18_31_28	—		**PINMUX18[31:28] Control**
			0	Pin is 3-stated.
			1h-Fh	*Reserved*
27-24	PINMUX18_27_24	—		**PINMUX18[27:24] Control**
			0	Pin is 3-stated.
			1h-Fh	*Reserved*
23-20	PINMUX18_23_20	—		**PINMUX18[23:20] Control**
			0	Pin is 3-stated.
			1h-Fh	*Reserved*
19-16	PINMUX18_19_16	—		**PINMUX18[19:16] Control**
			0	Pin is 3-stated.
			1h-Fh	*Reserved*
15-12	PINMUX18_15_12	—		**PINMUX18[15:12] Control**
			0	Pin is 3-stated.
			1h-Fh	*Reserved*
11-8	PINMUX18_11_8	—		**PINMUX18[11:8] Control**
			0	Pin is 3-stated.
			1h-Fh	*Reserved*
7-4	PINMUX18_7_4	30		**AXR0[12]/GP2[3]/BOOT[14] Control**
			0	Pin is 3-stated.
			1h-3h	*Reserved*
			4h	Selects Function AXR0[12]
			5h-7h	*Reserved*
			8h	Selects Function GP2[3]/BOOT[14]
			9h-Fh	*Reserved*
3-0	PINMUX18_3_0	—		**PINMUX18[3:0] Control**
			0	Pin is 3-stated.
			1h-Fh	*Reserved*

Copyright © 2013–2016, Texas Instruments Incorporated

10.5.10.20 Pin Multiplexing Control 19 Register (PINMUX19)

Figure 10-37. Pin Multiplexing Control 19 Register (PINMUX19)

31	16
Reserved	
R/W-0	

15	4	3	0
Reserved		PINMUX19_3_0	
R/W-0		R/W-0	

LEGEND: R/W = Read/Write; -*n* = value after reset

Table 10-41. Pin Multiplexing Control 19 Register (PINMUX19) Field Descriptions

Bit	Field	PFP Pin	Value	Description
31-4	Reserved	—	0	Reserved
3-0	PINMUX19_3_0	19		**GP2[10] Control**
			0	Pin is 3-stated.
			1h-7h	*Reserved*
			8h	Selects Function GP2[10]
			9h-Fh	*Reserved*

Copyright © 2013–2016, Texas Instruments Incorporated

10.5.11 Suspend Source Register (SUSPSRC)

The SUSPSRC is shown in Figure 10-38 and described in Table 10-42.

Figure 10-38. Suspend Source Register (SUSPSRC)

31	30	29	28	27	26	25	24
Reserved	Reserved	Reserved	Reserved	TIMER64_0SRC	Reserved	Reserved	Reserved
R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1
23	22	21	20	19	18	17	16
EPWM0SRC	SPI1SRC	SPI0SRC	UART2SRC	UART1SRC	UART0SRC	I2C1SRC	I2C0SRC
R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1
15	14	13	12	11	10	9	8
Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved
R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1
7	6	5	4	3	2	1	0
Reserved	DMAXSRC	Reserved	Reserved	EQEP0SRC	ECAP2SRC	ECAP1SRC	ECAP0SRC
R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1

LEGEND: R/W = Read/Write; -*n* = value after reset

Table 10-42. Suspend Source Register (SUSPSRC) Field Descriptions

Bit	Field	Value	Description
31-28	Reserved	1	Reserved. Write the default value to all bits when modifying this register.
27	TIMER64_0SRC		**Timer0 64 Emulation Suspend Source.**
		0	No emulation suspend.
		1	DSP is the source of the emulation suspend.
26-24	Reserved	1	Reserved. Write the default value to all bits when modifying this register.
23	EPWM0SRC		**EPWM0 Emulation Suspend Source.**
		0	No emulation suspend.
		1	DSP is the source of the emulation suspend.
22	SPI1SRC		**SPI1 Emulation Suspend Source.**
		0	No emulation suspend.
		1	DSP is the source of the emulation suspend.
21	SPI0SRC		**SPI0 Emulation Suspend Source.**
		0	No emulation suspend.
		1	DSP is the source of the emulation suspend.
20	UART2SRC		**UART2 Emulation Suspend Source.**
		0	No emulation suspend.
		1	DSP is the source of the emulation suspend.
19	UART1SRC		**UART1 Emulation Suspend Source.**
		0	No emulation suspend.
		1	DSP is the source of the emulation suspend.
18	UART0SRC		**UART0 Emulation Suspend Source.**
		0	No emulation suspend.
		1	DSP is the source of the emulation suspend.
17	I2C1SRC		**I2C1 Emulation Suspend Source.**
		0	No emulation suspend.
		1	DSP is the source of the emulation suspend.
16	I2C0SRC		**I2C0 Emulation Suspend Source.**
		0	No emulation suspend.
		1	DSP is the source of the emulation suspend.

Copyright © 2013–2016, Texas Instruments Incorporated

Table 10-42. Suspend Source Register (SUSPSRC) Field Descriptions (continued)

Bit	Field	Value	Description
15-7	Reserved	1	Reserved. Write the default value to all bits when modifying this register.
6	DMAXSRC		**dMAX Emulation Suspend Source.**
		0	No emulation suspend.
		1	DSP is the source of the emulation suspend.
5-4	Reserved	1	Reserved. Write the default value when modifying this register.
3	EQEP0SRC		**EQEP0 Emulation Suspend Source.**
		0	No emulation suspend.
		1	DSP is the source of the emulation suspend.
2	ECAP2SRC		**ECAP2 Emulation Suspend Source.**
		0	No emulation suspend.
		1	DSP is the source of the emulation suspend.
1	ECAP1SRC		**ECAP1 Emulation Suspend Source.**
		0	No emulation suspend.
		1	DSP is the source of the emulation suspend.
0	ECAP0SRC		**ECAP0 Emulation Suspend Source.**
		0	No emulation suspend.
		1	DSP is the source of the emulation suspend.

Copyright © 2013–2016, Texas Instruments Incorporated

10.5.12 Chip Signal Register (CHIPSIG)

The interrupts to the DSP can be generated by setting one of the two CHIPSIG[3-2] bits or an NMI interrupt by setting the CHIPSIG[4] bit in the chip signal register (CHIPSIG). Writing a 1 to these bits sets the interrupts, writing a 0 has no effect. Reads return the value of these bits and can also be used as status bits. The CHIPSIG is shown in Figure 10-39 and described in Table 10-43.

Figure 10-39. Chip Signal Register (CHIPSIG)

31	16
Reserved	
R-0	

15	5	4	3	2	1	0
Reserved		CHIPSIG4	CHIPSIG3	CHIPSIG2	Reserved	
R-0		R/W-0	R/W-0	R/W-0	R/W-0	

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 10-43. Chip Signal Register (CHIPSIG) Field Descriptions

Bit	Field	Value	Description
31-5	Reserved	0	Reserved
4	CHIPSIG4		**Asserts DSP NMI interrupt.**
		0	No effect
		1	Asserts interrupt
3	CHIPSIG3		**Asserts SYSCFG_CHIPINT3 interrupt.**
		0	No effect
		1	Asserts interrupt
2	CHIPSIG2		**Asserts SYSCFG_CHIPINT2 interrupt.**
		0	No effect
		1	Asserts interrupt
1-0	Reserved	0	Reserved. Write the default value to all bits when modifying this register.

Copyright © 2013–2016, Texas Instruments Incorporated

10.5.13 Chip Signal Clear Register (CHIPSIG_CLR)

The chip signal clear register (CHIPSIG_CLR) is used to clear the bits set in the chip signal register (CHIPSIG). Writing a 1 to a CHIPSIG[*n*] bit in CHIPSIG_CLR clears the corresponding CHIPSIG[*n*] bit in CHIPSIG; writing a 0 has no effect. After servicing the interrupt, the interrupted processor can clear the bits set in CHIPSIG by writing 1 to the corresponding bits in CHIPSIG_CLR. The other processor may poll the CHIPSIG[*n*] bit to determine when the interrupted processor has completed the interrupt service. The CHIPSIG_CLR is shown in Figure 10-40 and described in Table 10-44.

For more information on DSP interrupts, see the *DSP Subsystem* chapter.

Figure 10-40. Chip Signal Clear Register (CHIPSIG_CLR)

31	16
Reserved	
R-0	

15	5	4	3	2	1	0
Reserved		CHIPSIG4	CHIPSIG3	CHIPSIG2	Reserved	
R-0		R/W-0	R/W-0	R/W-0	R/W-0	

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 10-44. Chip Signal Clear Register (CHIPSIG_CLR) Field Descriptions

Bit	Field	Value	Description
31-5	Reserved	0	Reserved
4	CHIPSIG4		**Clears DSP NMI interrupt.**
		0	No effect
		1	Clears interrupt
3	CHIPSIG3		**Clears SYSCFG_CHIPINT3 interrupt.**
		0	No effect
		1	Clears interrupt
2	CHIPSIG2		**Clears SYSCFG_CHIPINT2 interrupt.**
		0	No effect
		1	Clears interrupt
1-0	Reserved	0	Reserved. Write the default value to all bits when modifying this register.

Copyright © 2013–2016, Texas Instruments Incorporated

10.5.14 Chip Configuration 0 Register (CFGCHIP0)

The chip configuration 0 register (CFGCHIP0) controls the following functions:

- PLL Controller memory-mapped register lock: Used to lock out writes to the PLL controller memory-mapped registers (MMRs) to prevent any erroneous writes in software to the PLL controller register space.
- EDMA3 Transfer Controller Default Burst Size (DBS) Control: This controls the maximum number of bytes issued per read/write command or the burst size for the individual transfer controllers (TCs) on the device. By default for all transfer controllers, the burst size is set to 16 bytes. However, CFGCHIP0 allows configurability of this parameter so that the TC can have a burst size of 16, 32, or 64 bytes. The burst size determines the intra packet efficiency for the EDMA3 transfers. Additionally, it also facilitates preemption at a system level, as all transfer requests are internally broken down by the transfer controller up to DBS size byte chunks and on a system level, each master's priority (configured by the MSTPRI register) is evaluated at burst size boundaries. The DBS value can significantly impact the standalone throughput performance depending on the source and destination (bus width/frequency/burst support etc) and the TC FIFO size, etc. Therefore, the DBS size configuration should be carefully analyzed to meet the system's throughput/performance requirements.

The CFGCHIP0 is shown in Figure 10-41 and described in Table 10-45.

Figure 10-41. Chip Configuration 0 Register (CFGCHIP0)

31	16
Reserved	
R-0	

15	5	4	3	2	1	0
Reserved		PLL_MASTER_LOCK	TC1DBS		TC0DBS	
R-0		R/W-0	R/W-0		R/W-0	

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 10-45. Chip Configuration 0 Register (CFGCHIP0) Field Descriptions

Bit	Field	Value	Description
31-5	Reserved	0	Reserved
4	PLL_MASTER_LOCK		**PLL MMRs lock.**
		0	PLLC MMRs are freely accessible.
		1	All PLLC MMRs are locked.
3-2	TC1DBS		**TC1 Default Burst Size (DBS).**
		0	16 bytes
		1h	32 bytes
		2h	64 bytes
		3h	*Reserved*
1-0	TC0DBS		**TC0 Default Burst Size (DBS).**
		0	16 bytes
		1h	32 bytes
		2h	64 bytes
		3h	*Reserved*

Copyright © 2013–2016, Texas Instruments Incorporated

10.5.15 Chip Configuration 1 Register (CFGCHIP1)

The chip configuration 1 register (CFGCHIP1) controls the following functions:

- eCAP0/1/2 event input source: Allows using McASP TX/RX events as eCAP event input sources.
- eHRPWM Time Base Clock (TBCLK) Synchronization: Allows the software to globally synchronize all enabled eHRPWM modules to the time base clock (TBCLK).
- McASP AMUTEIN signal source control: Allows selecting GPIO interrupt from different banks as source for the McASP AMUTEIN signal. CFGCHIP1 provides this signal source control for all McASPs on the device.

The CFGCHIP1 is shown in Figure 10-42 and described in Table 10-46.

Figure 10-42. Chip Configuration 1 Register (CFGCHIP1)

31	27	26	22	21	17	16
CAP2SRC		CAP1SRC		CAP0SRC		Reserved
R/W-0		R/W-0		R/W-0		R/W-0

15	14	13	12	11	8
Reserved	Reserved		TBCLKSYNC	AMUTESEL2	
R/W-0	R-0		R/W-0	R/W-0	

7	4	3	0
AMUTESEL1		AMUTESEL0	
R/W-0		R/W-0	

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 10-46. Chip Configuration 1 Register (CFGCHIP1) Field Descriptions

Bit	Field	Value	Description
31-27	CAP2SRC		**Selects the eCAP2 module event input.**
		0	eCAP2 Pin input
		1h	McASP0 TX DMA Event
		2h	McASP0 RX DMA Event
		3h	McASP1 TX DMA Event
		4h	McASP1 RX DMA Event
		5h	McASP2 TX DMA Event
		6h	McASP2 RX DMA Event
		7h-1Fh	*Reserved*
26-22	CAP1SRC		**Selects the eCAP1 module event input.**
		0	eCAP1 Pin input
		1h	McASP0 TX DMA Event
		2h	McASP0 RX DMA Event
		3h	McASP1 TX DMA Event
		4h	McASP1 RX DMA Event
		5h	McASP2 TX DMA Event
		6h	McASP2 RX DMA Event
		7h-1Fh	*Reserved*

Copyright © 2013–2016, Texas Instruments Incorporated

Table 10-46. Chip Configuration 1 Register (CFGCHIP1) Field Descriptions (continued)

Bit	Field	Value	Description
21-17	CAP0SRC		**Selects the eCAP0 module event input.**
		0	eCAP0 Pin input
		1h	McASP0 TX DMA Event
		2h	McASP0 RX DMA Event
		3h	McASP1 TX DMA Event
		4h	McASP1 RX DMA Event
		5h	McASP2 TX DMA Event
		6h	McASP2 RX DMA Event
		7h-1Fh	*Reserved*
16-15	Reserved	0	Reserved. Write the default value to all bits when modifying this register.
14-13	Reserved	0	Reserved. Always read as 0.
12	TBCLKSYNC		**eHRPWM Module Time Base Clock (TBCLK) Synchronization.** Allows you to globally synchronize all enabled eHRPWM modules to the time base clock (TBCLK).
		0	Time base clock (TBCLK) within each enabled eHRPWM module is stopped.
		1	All enabled eHRPWM module clocks are started with the first rising edge of TBCLK aligned. For perfectly synchronized TBCLKs, the prescaler bits in the TBCTL register of each eHRPWM module must be set identically.
11-8	AMUTESEL2		**Selects the source of McASP2 AMUTEIN signal.**
		0	Drive McASP2 AMUTEIN signal low
		1h	GPIO Interrupt from Bank 0
		2h	GPIO Interrupt from Bank 1
		3h	GPIO Interrupt from Bank 2
		4h	GPIO Interrupt from Bank 3
		5h	GPIO Interrupt from Bank 4
		6h	GPIO Interrupt from Bank 5
		7h	GPIO Interrupt from Bank 6
		8h	GPIO Interrupt from Bank 7
		9h-Fh	*Reserved*
7-4	AMUTESEL1		**Selects the source of McASP1 AMUTEIN signal.**
		0	Drive McASP1 AMUTEIN signal low
		1h	GPIO Interrupt from Bank 0
		2h	GPIO Interrupt from Bank 1
		3h	GPIO Interrupt from Bank 2
		4h	GPIO Interrupt from Bank 3
		5h	GPIO Interrupt from Bank 4
		6h	GPIO Interrupt from Bank 5
		7h	GPIO Interrupt from Bank 6
		8h	GPIO Interrupt from Bank 7
		9h-Fh	*Reserved*

Copyright © 2013–2016, Texas Instruments Incorporated

Table 10-46. Chip Configuration 1 Register (CFGCHIP1) Field Descriptions (continued)

Bit	Field	Value	Description
3-0	AMUTESEL0		**Selects the source of McASP0 AMUTEIN signal.**
		0	Drive McASP0 AMUTEIN signal low
		1h	GPIO Interrupt from Bank 0
		2h	GPIO Interrupt from Bank 1
		3h	GPIO Interrupt from Bank 2
		4h	GPIO Interrupt from Bank 3
		5h	GPIO Interrupt from Bank 4
		6h	GPIO Interrupt from Bank 5
		7h	GPIO Interrupt from Bank 6
		8h	GPIO Interrupt from Bank 7
		9h-Fh	*Reserved*

10.5.16 Chip Configuration 4 Register (CFGCHIP4)

The CFGCHIP4 register is used for clearing the AMUNTEIN signal for the McASPs. Writing a 1 causes a single pulse that clears the 'latched' GPIO interrupt for AMUTEIN of McASP if it was previously set. Reads always return a value of 0. The register has individual bits for each McASP supported on the device. The CFGCHIP4 is shown in Figure 10-43 and described in Table 10-47.

Figure 10-43. Chip Configuration 4 Register (CFGCHIP4)

31	16
Reserved	
R-0	

15	8	7	3	2	1	0
Reserved		Reserved		AMUTECLR2	AMUTECLR1	AMUTECLR0
R/W-FFh		R/W-0		R/W-0	R/W-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; *-n* = value after reset

Table 10-47. Chip Configuration 4 Register (CFGCHIP4) Field Descriptions

Bit	Field	Value	Description
31-16	Reserved	0	Reserved
15-8	Reserved	FFh	Reserved. Write the default value when modifying this register.
7-3	Reserved	0	Reserved. Write the default value to all bits when modifying this register.
2	AMUTECLR2		**Clears the 'latched' GPIO interrupt for AMUTEIN of McASP2 when set to 1.**
		0	No effect
		1	Clears interrupt
1	AMUTECLR1		**Clears the 'latched' GPIO interrupt for AMUTEIN of McASP1 when set to 1.**
		0	No effect
		1	Clears interrupt
0	AMUTECLR0		**Clears the 'latched' GPIO interrupt for AMUTEIN of McASP0 when set to 1.**
		0	No effect
		1	Clears interrupt

Copyright © 2013–2016, Texas Instruments Incorporated

Boot Considerations

Topic Page

11.1 Introduction 180

Copyright © 2013–2016, Texas Instruments Incorporated

11.1 Introduction

This device supports a variety of boot modes through an internal DSP ROM bootloader. This device does not support dedicated hardware boot modes; therefore, all boot modes utilize the internal DSP ROM. The input states of the BOOT pins are sampled and latched into the BOOTCFG register, which is part of the system configuration (SYSCFG) module, when device reset is deasserted. Boot mode selection is determined by the values of the BOOT pins.

The following boot modes are supported:

- NAND Flash boot
  - 8-bit NAND
  - 16-bit NAND
- NOR Flash boot
  - NOR Direct boot
  - NOR Legacy boot
  - NOR AIS boot
- I2C0/I2C1 Boot
  - Master boot
  - Slave boot
- SPI0/SPI1 Boot
  - Master boot
  - Slave boot
- UART0/1/2 Boot

See *Using the C6747/45/43 Bootloader Application Report* (SPRABB1) for more details on the ROM Boot Loader, a list of boot pins used, and the complete list of supported boot modes.

Copyright © 2013–2016, Texas Instruments Incorporated

Dual Data Movement Accelerator (dMAX)

Topic	Page

Copyright © 2013–2016, Texas Instruments Incorporated

The Dual Data Movement Accelerator (dMAX) module consists of:

- Two data transfers units (dMAX0 and dMAX1) and their associated memories.
- An interrupt controller (INTC) for handling system interrupt events. The INTC also supports posting events back to the device level host CPU.
- A Switched Central Resource (SCR) for connecting the various internal and external masters to the resources inside the dMAX.

The two dMAX units can operate completely independently or in coordination with each other. The dMAX units can also work in coordination with the device level host CPU. This is determined by the nature of the program that is loaded into their instruction memory. Several different signaling mechanisms are available between the dMAX units and the device level host CPU.

The dMAX units are optimized for performing embedded tasks that require manipulation of packed memory-mapped data structures, handling of system events that have tight real-time constraints and interfacing with systems external to the device.

The dMAX (also referred to as the Programmable Real-Time Unit) documentation (peripheral guide) is on the external wiki:Programmable_Realtime_Unit.

Copyright © 2013–2016, Texas Instruments Incorporated

Enhanced Capture (eCAP) Module

The enhanced capture (eCAP) module is essential in systems where accurate timing of external events is important. This chapter describes the eCAP module.

Topic **Page**

13.1 Introduction **184**
13.2 Architecture **185**
13.3 Applications **194**
13.4 Registers **210**

Copyright © 2013–2016, Texas Instruments Incorporated

13.1 Introduction

13.1.1 Purpose of the Peripheral

Uses for eCAP include:

- Sample rate measurements of audio inputs
- Speed measurements of rotating machinery (for example, toothed sprockets sensed via Hall sensors)
- Elapsed time measurements between position sensor pulses
- Period and duty cycle measurements of pulse train signals
- Decoding current or voltage amplitude derived from duty cycle encoded current/voltage sensors

13.1.2 Features

The eCAP module includes the following features:

- 32-bit time base counter
- 4-event time-stamp registers (each 32 bits)
- Edge polarity selection for up to four sequenced time-stamp capture events
- Interrupt on either of the four events
- Single shot capture of up to four event time-stamps
- Continuous mode capture of time-stamps in a four-deep circular buffer
- Absolute time-stamp capture
- Difference (Delta) mode time-stamp capture
- All above resources dedicated to a single input pin
- When not used in capture mode, the ECAP module can be configured as a single channel PWM output

Copyright © 2013–2016, Texas Instruments Incorporated

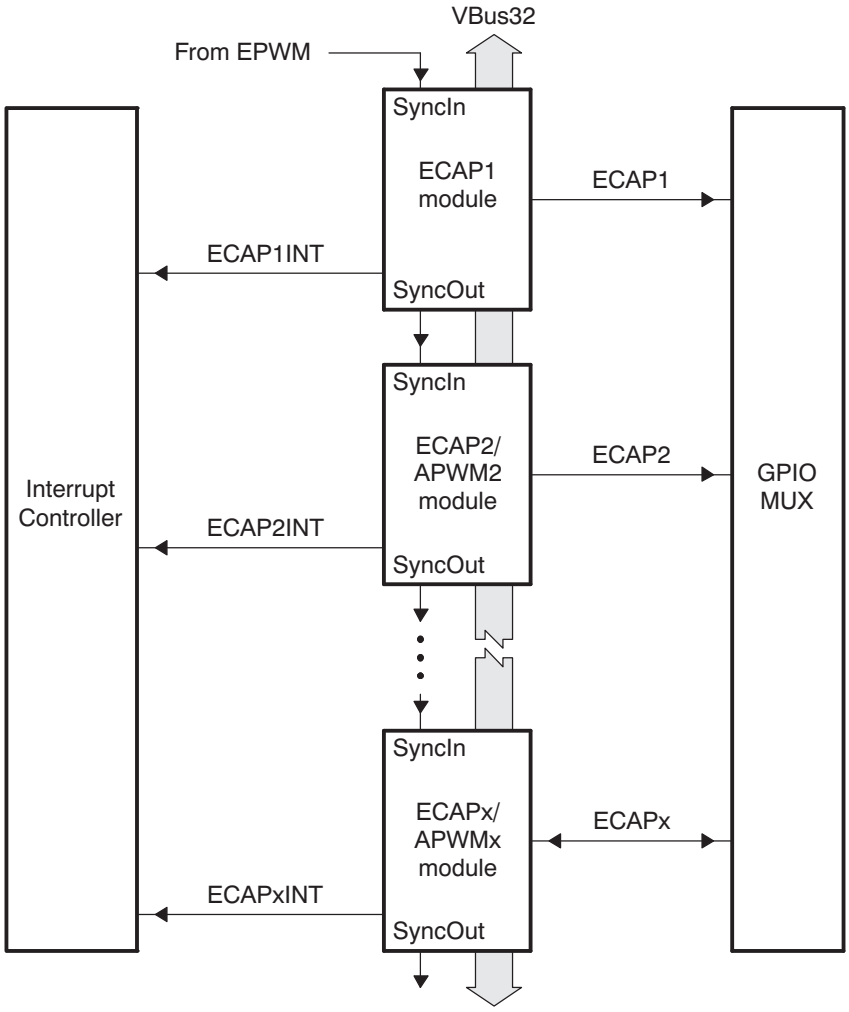
13.2 Architecture

The eCAP module represents one complete capture channel that can be instantiated multiple times depending on the target device. In the context of this guide, one eCAP channel has the following independent key resources:

- Dedicated input capture pin
- 32-bit time base counter
- 4 × 32-bit time-stamp capture registers (CAP1-CAP4)
- 4-stage sequencer (Modulo4 counter) that is synchronized to external events, ECAP pin rising/falling edges.
- Independent edge polarity (rising/falling edge) selection for all 4 events
- Input capture signal prescaling (from 2-62)
- One-shot compare register (2 bits) to freeze captures after 1 to 4 time-stamp events
- Control for continuous time-stamp captures using a 4-deep circular buffer (CAP1-CAP4) scheme
- Interrupt capabilities on any of the 4 capture events

Multiple identical eCAP modules can be contained in a system as shown in Figure 13-1. The number of modules is device-dependent and is based on target application needs. In this chapter, the letter x within a signal or module name is used to indicate a generic eCAP instance on a device.

Figure 13-1. Multiple eCAP Modules

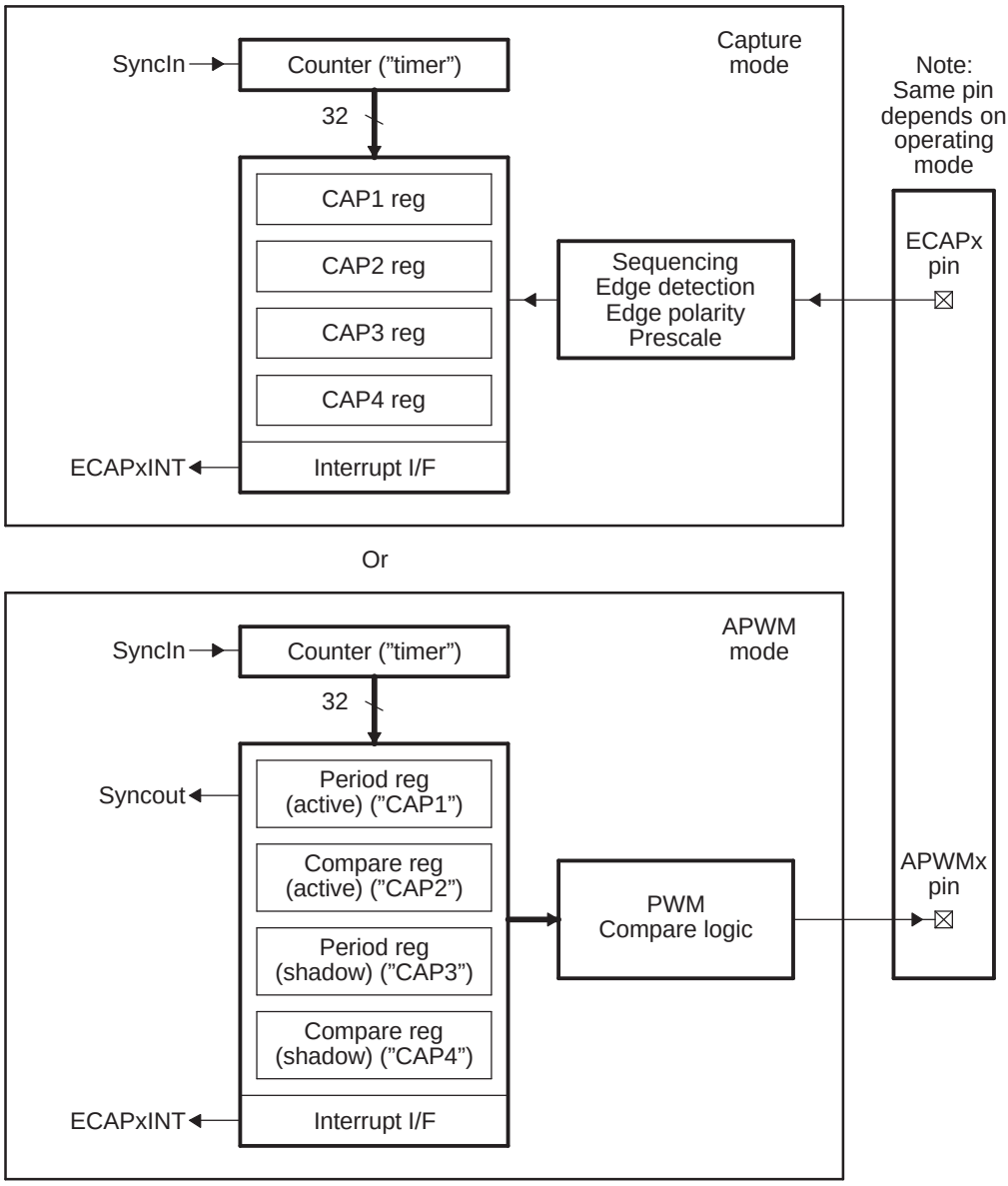


Copyright © 2013–2016, Texas Instruments Incorporated

13.2.1 Capture and APWM Operating Mode

You can use the eCAP module resources to implement a single-channel PWM generator (with 32 bit capabilities) when it is not being used for input captures. The counter operates in count-up mode, providing a time-base for asymmetrical pulse width modulation (PWM) waveforms. The CAP1 and CAP2 registers become the active period and compare registers, respectively, while CAP3 and CAP4 registers become the period and capture shadow registers, respectively. Figure 13-2 is a high-level view of both the capture and auxiliary pulse-width modulator (APWM) modes of operation.

Figure 13-2. Capture and APWM Modes of Operation



(1) A single pin is shared between CAP and APWM functions. In capture mode, it is an input; in APWM mode, it is an output.

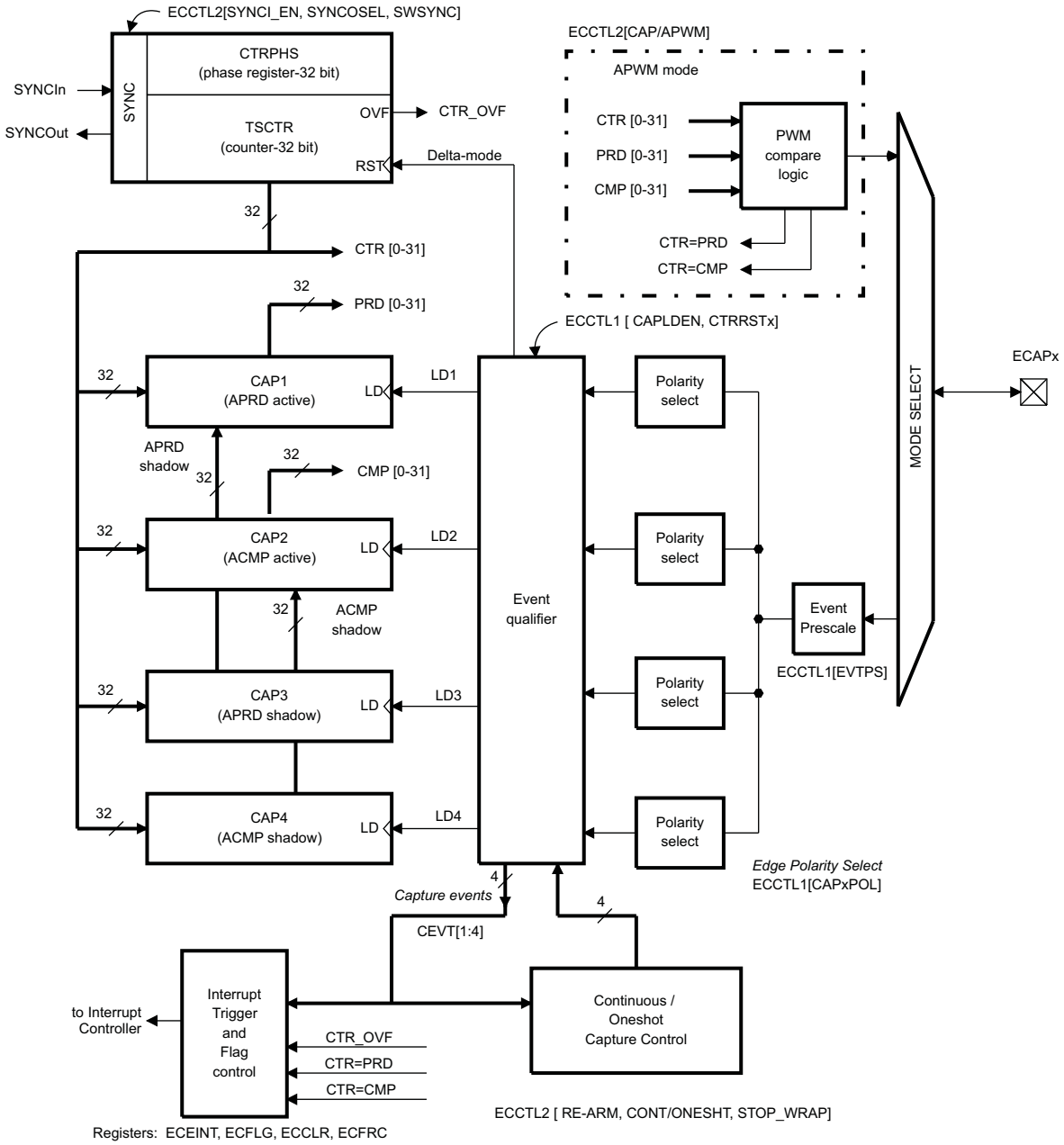
(2) In APWM mode, writing any value to CAP1/CAP2 active registers also writes the same value to the corresponding shadow registers CAP3/CAP4. This emulates immediate mode. Writing to the shadow registers CAP3/CAP4 invokes the shadow mode.

Copyright © 2013–2016, Texas Instruments Incorporated

13.2.2 Capture Mode Description

Figure 13-3 shows the various components that implement the capture function.

Figure 13-3. Capture Function Diagram

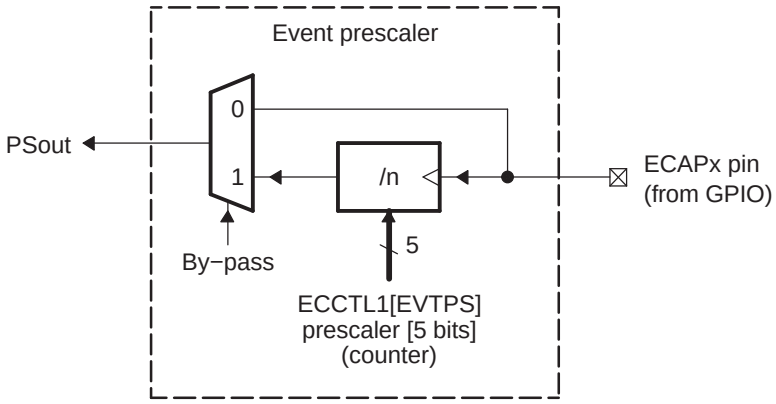


Copyright © 2013–2016, Texas Instruments Incorporated

13.2.2.1 Event Prescaler

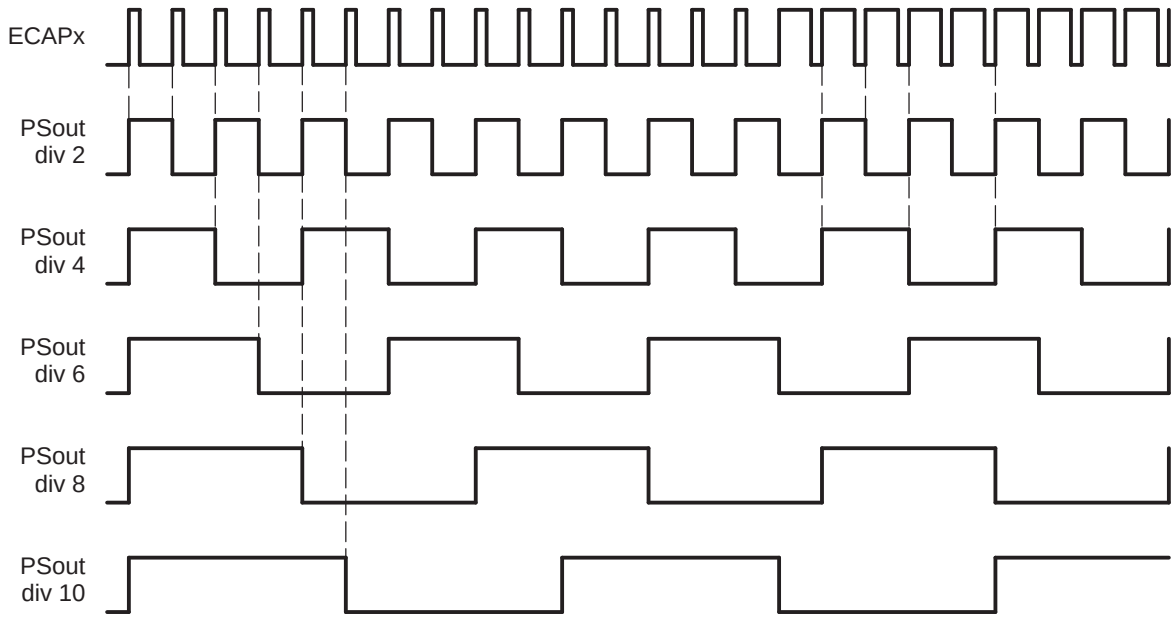
An input capture signal (pulse train) can be prescaled by N = 2-62 (in multiples of 2) or can bypass the prescaler. This is useful when very high frequency signals are used as inputs. Figure 13-4 shows a functional diagram and Figure 13-5 shows the operation of the prescale function.

Figure 13-4. Event Prescale Control



(1) When a prescale value of 1 is chosen (ECCTL1[13:9] = 0000) the input capture signal by-passes the prescale logic completely.

Figure 13-5. Prescale Function Waveforms



Copyright © 2013–2016, Texas Instruments Incorporated

13.2.2.2 Edge Polarity Select and Qualifier

- Four independent edge polarity (rising edge/falling edge) selection multiplexers are used, one for each capture event.
- Each edge (up to 4) is event qualified by the Modulo4 sequencer.
- The edge event is gated to its respective CAP*n* register by the Mod4 counter. The CAP*n* register is loaded on the falling edge.

13.2.2.3 Continuous/One-Shot Control

- The Mod4 (2 bit) counter is incremented via edge qualified events (CEVT1-CEVT4).
- The Mod4 counter continues counting (0->1->2->3->0) and wraps around unless stopped.
- A 2-bit stop register is used to compare the Mod4 counter output, and when equal stops the Mod4 counter and inhibits further loads of the CAP1-CAP4 registers. This occurs during one-shot operation.

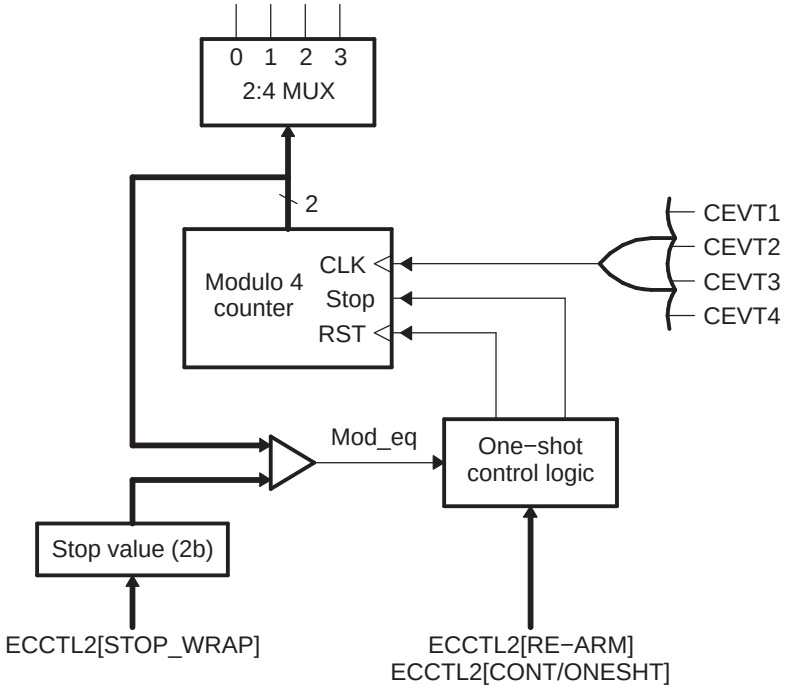
The continuous/one-shot block (Figure 13-6) controls the start/stop and reset (zero) functions of the Mod4 counter via a mono-shot type of action that can be triggered by the stop-value comparator and re-armed via software control.

Once armed, the eCAP module waits for 1-4 (defined by stop-value) capture events before freezing both the Mod4 counter and contents of CAP1-4 registers (time-stamps).

Re-arming prepares the eCAP module for another capture sequence. Also re-arming clears (to zero) the Mod4 counter and permits loading of CAP1-4 registers again, providing the CAPLDEN bit is set.

In continuous mode, the Mod4 counter continues to run (0->1->2->3->0, the one-shot action is ignored, and capture values continue to be written to CAP1-4 in a circular buffer sequence.

Figure 13-6. Continuous/One-shot Block Diagram



Copyright © 2013–2016, Texas Instruments Incorporated

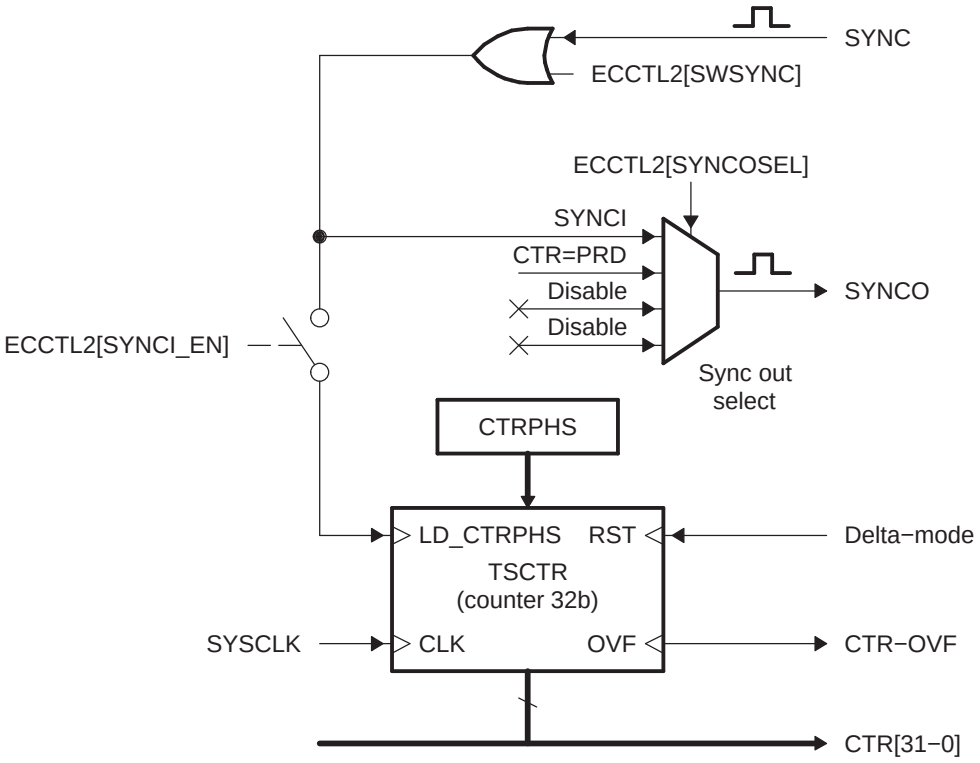
13.2.2.4 32-Bit Counter and Phase Control

This counter (Figure 13-7) provides the time-base for event captures, and is clocked via the system clock.

A phase register is provided to achieve synchronization with other counters, via a hardware and software forced sync. This is useful in APWM mode when a phase offset between modules is needed.

On any of the four event loads, an option to reset the 32-bit counter is given. This is useful for time difference capture. The 32-bit counter value is captured first, then it is reset to 0 by any of the LD1-LD4 signals.

Figure 13-7. Counter and Synchronization Block Diagram



Copyright © 2013–2016, Texas Instruments Incorporated

13.2.2.5 CAP1-CAP4 Registers

These 32-bit registers are fed by the 32-bit counter timer bus, CTR[0-31] and are loaded (capture a time-stamp) when their respective LD inputs are strobed.

Loading of the capture registers can be inhibited via control bit CAPLDEN. During one-shot operation, this bit is cleared (loading is inhibited) automatically when a stop condition occurs, StopValue = Mod4.

CAP1 and CAP2 registers become the active period and compare registers, respectively, in APWM mode.

CAP3 and CAP4 registers become the respective shadow registers (APRD and ACMP) for CAP1 and CAP2 during APWM operation.

13.2.2.6 Interrupt Control

An Interrupt can be generated on capture events (CEVT1-CEVT4, CTROVF) or APWM events (CTR = PRD, CTR = CMP). See Figure 13-8.

A counter overflow event (FFFF FFFFh->0000 0000h) is also provided as an interrupt source (CTROVF).

The capture events are edge and sequencer qualified (that is, ordered in time) by the polarity select and Mod4 gating, respectively.

One of these events can be selected as the interrupt source (from the eCAP*n* module) going to the interrupt controller.

Seven interrupt events (CEVT1, CEVT2, CEVT3, CEVT4, CNTOVF, CTR = PRD, CTR = CMP) can be generated. The interrupt enable register (ECEINT) is used to enable/disable individual interrupt event sources. The interrupt flag register (ECFLG) indicates if any interrupt event has been latched and contains the global interrupt flag bit (INT). An interrupt pulse is generated to the interrupt controller only if any of the interrupt events are enabled, the flag bit is 1, and the INT flag bit is 0. The interrupt service routine must clear the global interrupt flag bit and the serviced event via the interrupt clear register (ECCLR) before any other interrupt pulses are generated. You can force an interrupt event via the interrupt force register (ECFRC). This is useful for test purposes.

13.2.2.7 Shadow Load and Lockout Control

In capture mode, this logic inhibits (locks out) any shadow loading of CAP1 or CAP2 from APRD and ACMP registers, respectively.

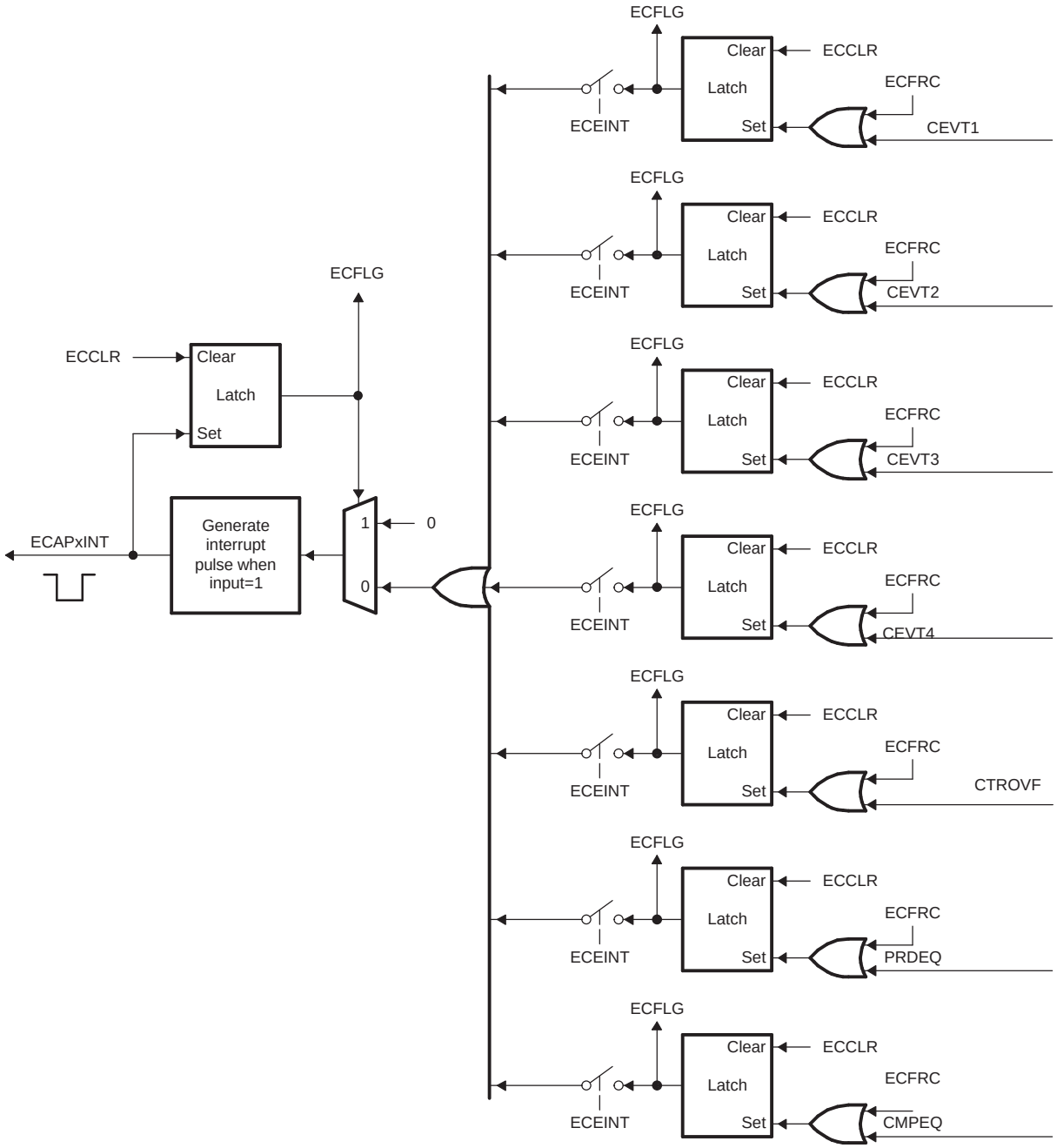
In APWM mode, shadow loading is active and two choices are permitted:

- Immediate - APRD or ACMP are transferred to CAP1 or CAP2 immediately upon writing a new value.
- On period equal, CTR[31:0] = PRD[31:0]

NOTE: The CEVT1, CEVT2, CEVT3, CEVT4 flags are only active in capture mode (ECCTL2[CAP/APWM == 0]). The CTR = PRD, CTR = CMP flags are only valid in APWM mode (ECCTL2[CAP/APWM == 1]). CNTOVF flag is valid in both modes.

Copyright © 2013–2016, Texas Instruments Incorporated

Figure 13-8. Interrupts in eCAP Module



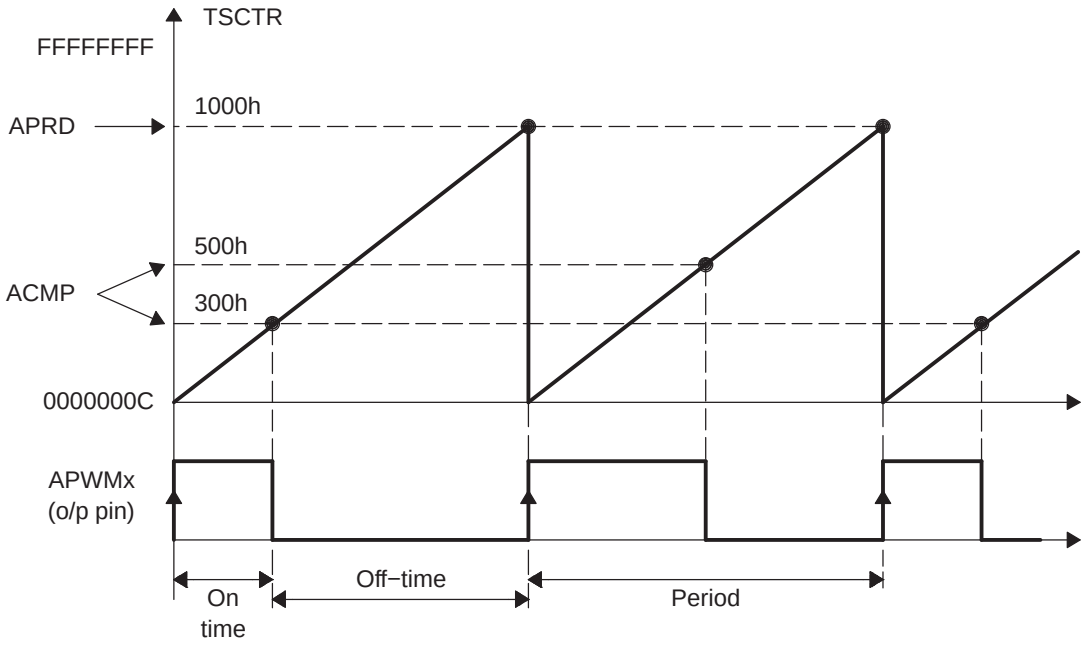
Copyright © 2013–2016, Texas Instruments Incorporated

13.2.2.8 APWM Mode Operation

Main operating highlights of the APWM section:

- The time-stamp counter bus is made available for comparison via 2 digital (32-bit) comparators.
- When CAP1/2 registers are not used in capture mode, their contents can be used as Period and Compare values in APWM mode.
- Double buffering is achieved via shadow registers APRD and ACMP (CAP3/4). The shadow register contents are transferred over to CAP1/2 registers either immediately upon a write, or on a CTR = PRD trigger.
- In APWM mode, writing to CAP1/CAP2 active registers will also write the same value to the corresponding shadow registers CAP3/CAP4. This emulates immediate mode. Writing to the shadow registers CAP3/CAP4 will invoke the shadow mode.
- During initialization, you must write to the active registers for both period and compare. This automatically copies the initial values into the shadow values. For subsequent compare updates, during run-time, you only need to use the shadow registers.

Figure 13-9. PWM Waveform Details Of APWM Mode Operation



The behavior of APWM active-high mode (APWMPOL == 0) is:

```
CMP = 0x00000000, output low for duration of period (0% duty)
CMP = 0x00000001, output high 1 cycle
CMP = 0x00000002, output high 2 cycles
CMP = PERIOD, output high except for 1 cycle (<100% duty)
CMP = PERIOD+1, output high for complete period (100% duty)
CMP > PERIOD+1, output high for complete period
```

The behavior of APWM active-low mode (APWMPOL == 1) is:

```
CMP = 0x00000000, output high for duration of period (0% duty)
CMP = 0x00000001, output low 1 cycle
CMP = 0x00000002, output low 2 cycles
CMP = PERIOD, output low except for 1 cycle (<100% duty)
CMP = PERIOD+1, output low for complete period (100% duty)
```


Copyright © 2013–2016, Texas Instruments Incorporated

```
CMP > PERIOD+1, output low for complete period
```

13.3 Applications

The following sections will provide Applications examples and code snippets to show how to configure and operate the eCAP module. For clarity and ease of use, below are useful #defines which will help in the understanding of the examples.

```
// ECCTL1 ( ECAP Control Reg 1)
//============================
// CAPxPOL bits
#define    EC_RISING              0x0
#define    EC_FALLING             0x1

// CTRRSTx bits
#define    EC_ABS_MODE            0x0
#define    EC_DELTA_MODE          0x1

// PRESCALE bits
#define    EC_BYPASS              0x0
#define    EC_DIV1                0x0
#define    EC_DIV2                0x1
#define    EC_DIV4                0x2
#define    EC_DIV6                0x3
#define    EC_DIV8                0x4
#define    EC_DIV10               0x5

// ECCTL2 ( ECAP Control Reg 2)
//============================
// CONT/ONESHOT bit
#define    EC_CONTINUOUS          0x0
#define    EC_ONESHOT             0x1

// STOPVALUE bit
#define    EC_EVENT1              0x0
#define    EC_EVENT2              0x1
#define    EC_EVENT3              0x2
#define    EC_EVENT4              0x3

// RE-ARM bit
#define    EC_ARM                 0x1

// TSCTRSTOP bit
#define    EC_FREEZE              0x0
#define    EC_RUN                 0x1

// SYNCO_SEL bit
#define    EC_SYNCIN              0x0
#define    EC_CTR_PRD             0x1
#define    EC_SYNCO_DIS           0x2

// CAP/APWM mode bit
#define    EC_CAP_MODE            0x0
#define    EC_APWM_MODE           0x1

// APWMPOL bit
#define    EC_ACTV_HI             0x0
#define    EC_ACTV_LO             0x1

// Generic
#define    EC_DISABLE             0x0
#define    EC_ENABLE              0x1
#define    EC_FORCE               0x1
```

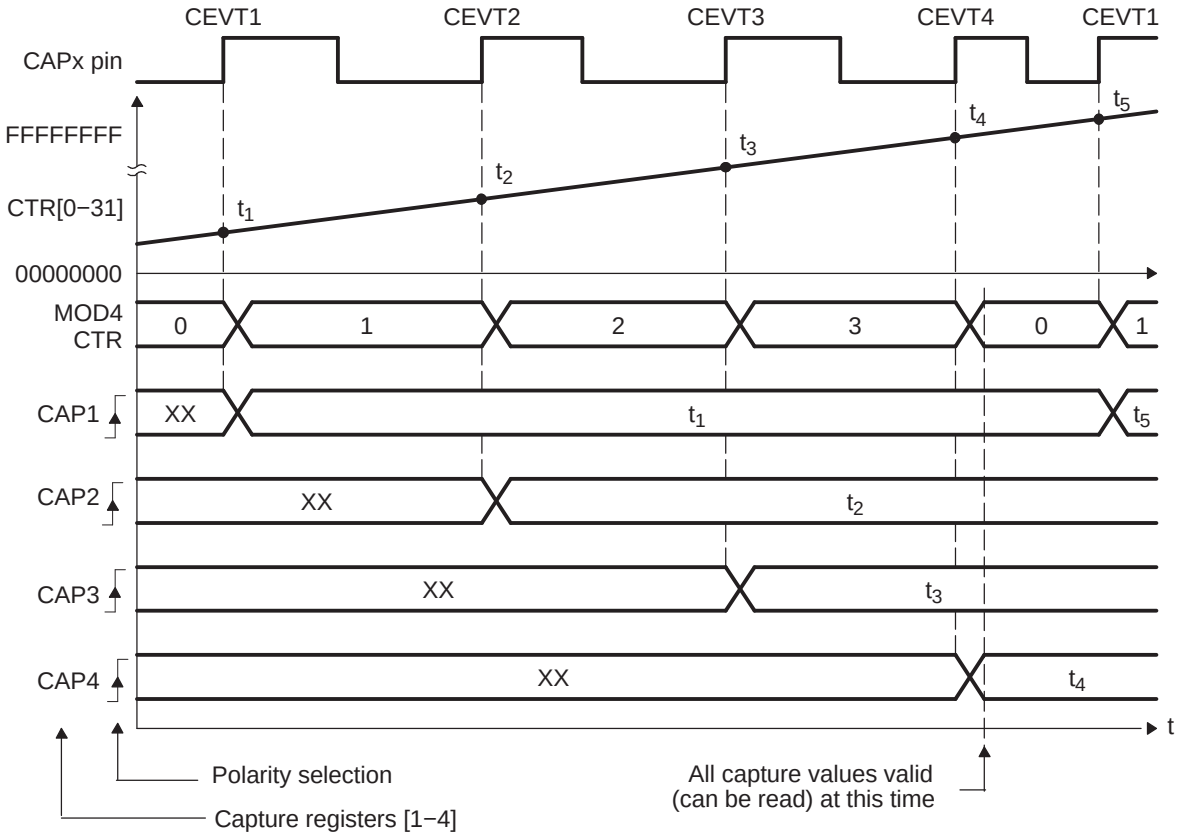

Copyright © 2013–2016, Texas Instruments Incorporated

13.3.1 Absolute Time-Stamp Operation Rising Edge Trigger Example

Figure 13-10 shows an example of continuous capture operation (Mod4 counter wraps around). In this figure, TSCTR counts-up without resetting and capture events are qualified on the rising edge only, this gives period (and frequency) information.

On an event, the TSCTR contents (time-stamp) is first captured, then Mod4 counter is incremented to the next state. When the TSCTR reaches FFFF FFFFh (maximum value), it wraps around to 0000 0000h (not shown in Figure 13-10), if this occurs, the CTROVF (counter overflow) flag is set, and an interrupt (if enabled) occurs, CTROVF (counter overflow) Flag is set, and an Interrupt (if enabled) occurs. Captured time-stamps are valid at the point indicated by the diagram, after the 4th event, hence event CEVT4 can conveniently be used to trigger an interrupt and the CPU can read data from the CAP*n* registers.

Figure 13-10. Capture Sequence for Absolute Time-Stamp, Rising Edge Detect



Copyright © 2013–2016, Texas Instruments Incorporated

Table 13-1. ECAP Initialization for CAP Mode Absolute Time, Rising Edge Trigger

Register	Bit	Value
ECCTL1	CAP1POL	EC_RISING
ECCTL1	CAP2POL	EC_RISING
ECCTL1	CAP3POL	EC_RISING
ECCTL1	CAP4POL	EC_RISING
ECCTL1	CTRRST1	EC_ABS_MODE
ECCTL1	CTRRST2	EC_ABS_MODE
ECCTL1	CTRRST3	EC_ABS_MODE
ECCTL1	CTRRST4	EC_ABS_MODE
ECCTL1	CAPLDEN	EC_ENABLE
ECCTL1	PRESCALE	EC_DIV1
ECCTL2	CAP_APWM	EC_CAP_MODE
ECCTL2	CONT_ONESHT	EC_CONTINUOUS
ECCTL2	SYNCO_SEL	EC_SYNCO_DIS
ECCTL2	SYNCI_EN	EC_DISABLE
ECCTL2	TSCTRSTOP	EC_RUN

Example 13-1. Code Snippet for CAP Mode Absolute Time, Rising Edge Trigger

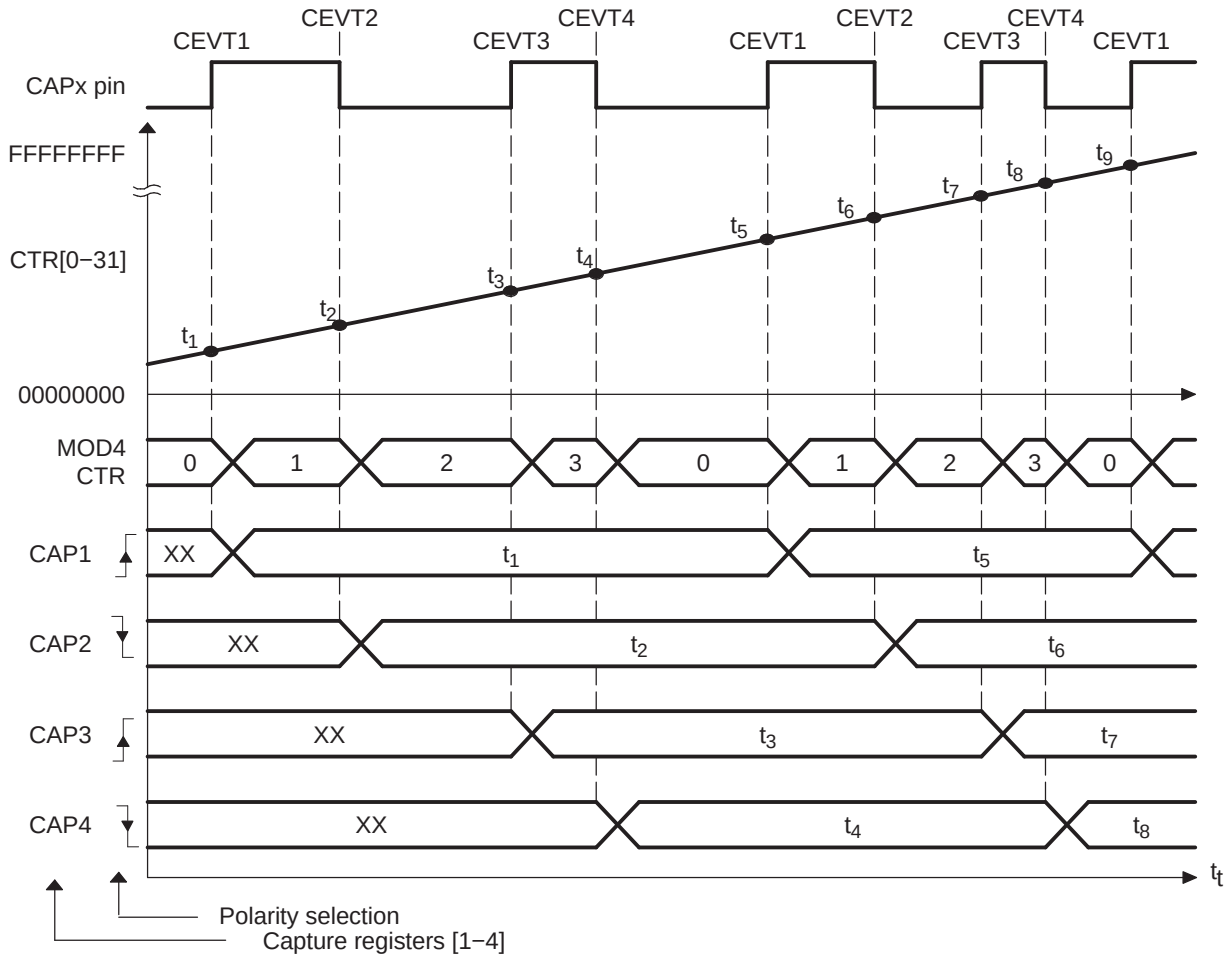
```
// Code snippet for CAP mode Absolute Time, Rising edge trigger

// Run Time ( e.g. CEVT4 triggered ISR call)
//=========================================
    TSt1 = ECAPxRegs.CAP1;        // Fetch Time-Stamp captured at t1
    TSt2 = ECAPxRegs.CAP2;        // Fetch Time-Stamp captured at t2
    TSt3 = ECAPxRegs.CAP3;        // Fetch Time-Stamp captured at t3
    TSt4 = ECAPxRegs.CAP4;        // Fetch Time-Stamp captured at t4

    Period1 = TSt2-TSt1;        // Calculate 1st period
    Period2 = TSt3-TSt2;        // Calculate 2nd period
    Period3 = TSt4-TSt3;        // Calculate 3rd period
```


Copyright © 2013–2016, Texas Instruments Incorporated

13.3.2 Absolute Time-Stamp Operation Rising and Falling Edge Trigger Example

In Figure 13-11 the eCAP operating mode is almost the same as in the previous section except capture events are qualified as either rising or falling edge, this now gives both period and duty cycle information: Period1 = $t_3 - t_1$, Period2 = $t_5 - t_3$, …etc. Duty Cycle1 (on-time %) = ($t_2 - t_1$) / Period1 x 100%, etc. Duty Cycle1 (off-time %) = ($t_3 - t_2$) / Period1 x 100%, etc.

Figure 13-11. Capture Sequence for Absolute Time-Stamp, Rising and Falling Edge Detect



Copyright © 2013–2016, Texas Instruments Incorporated

Table 13-2. ECAP Initialization for CAP Mode Absolute Time, Rising and Falling Edge Trigger

Register	Bit	Value
ECCTL1	CAP1POL	EC_RISING
ECCTL1	CAP2POL	EC_FALLING
ECCTL1	CAP3POL	EC_RISING
ECCTL1	CAP4POL	EC_FALLING
ECCTL1	CTRRST1	EC_ABS_MODE
ECCTL1	CTRRST2	EC_ABS_MODE
ECCTL1	CTRRST3	EC_ABS_MODE
ECCTL1	CTRRST4	EC_ABS_MODE
ECCTL1	CAPLDEN	EC_ENABLE
ECCTL1	PRESCALE	EC_DIV1
ECCTL2	CAP_APWM	EC_CAP_MODE
ECCTL2	CONT_ONESHT	EC_CONTINUOUS
ECCTL2	SYNCO_SEL	EC_SYNCO_DIS
ECCTL2	SYNCI_EN	EC_DISABLE
ECCTL2	TSCTRSTOP	EC_RUN

Example 13-2. Code Snippet for CAP Mode Absolute Time, Rising and Falling Edge Trigger

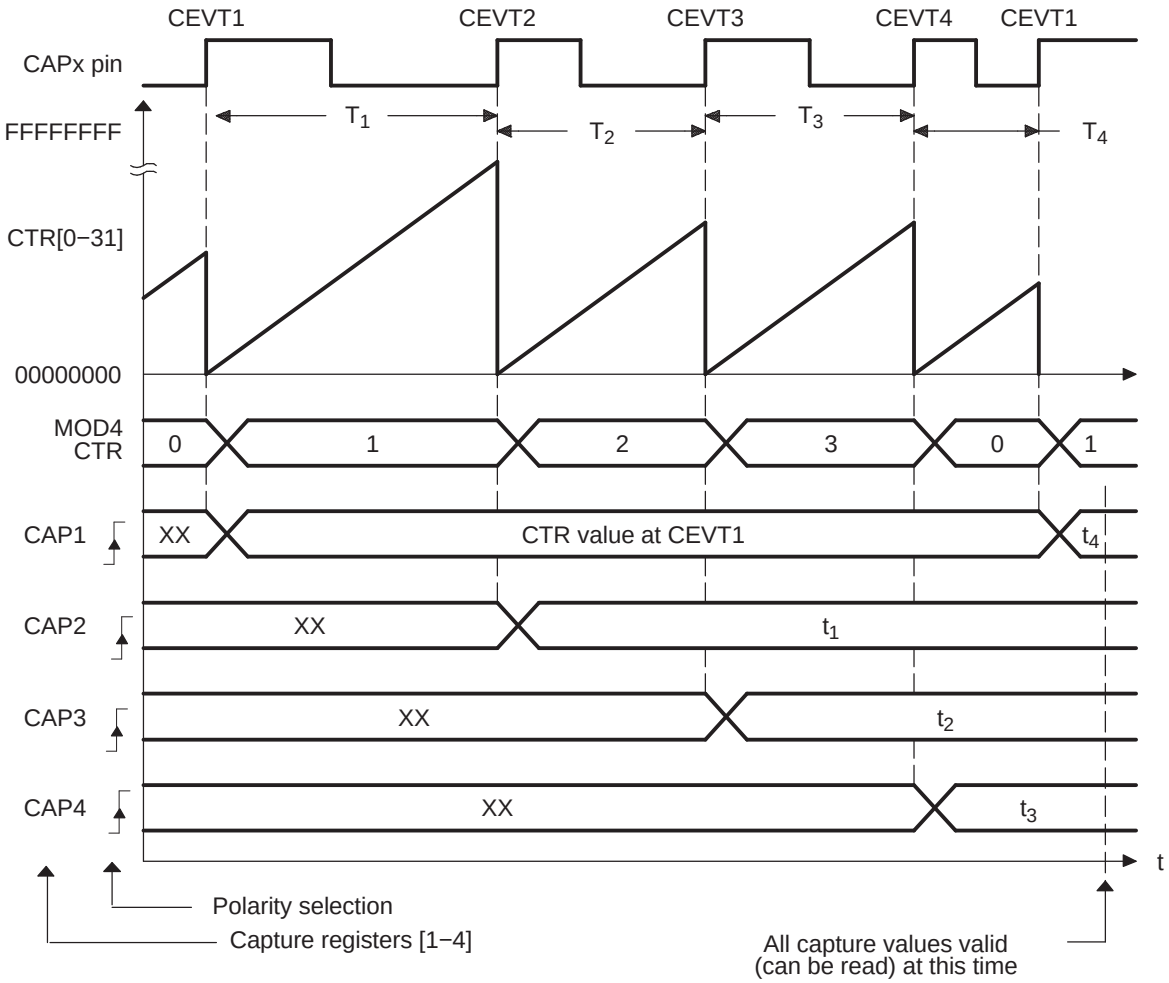
```
// Code snippet for CAP mode Absolute Time, Rising & Falling edge triggers

// Run Time ( e.g. CEVT4 triggered ISR call)
//=========================================
    TSt1 = ECAPxRegs.CAP1;        // Fetch Time-Stamp captured at t1
    TSt2 = ECAPxRegs.CAP2;        // Fetch Time-Stamp captured at t2
    TSt3 = ECAPxRegs.CAP3;        // Fetch Time-Stamp captured at t3
    TSt4 = ECAPxRegs.CAP4;        // Fetch Time-Stamp captured at t4

    Period1 = TSt3-TSt1;        // Calculate 1st period
    DutyOnTime1 = TSt2-TSt1;    // Calculate On time
    DutyOffTime1 = TSt3-TSt2;    // Calculate Off time
```


Copyright © 2013–2016, Texas Instruments Incorporated

13.3.3 Time Difference (Delta) Operation Rising Edge Trigger Example

Figure 13-12 shows how the eCAP module can be used to collect Delta timing data from pulse train waveforms. Here Continuous Capture mode (TSCTR counts-up without resetting, and Mod4 counter wraps around) is used. In Delta-time mode, TSCTR is Reset back to Zero on every valid event. Here Capture events are qualified as Rising edge only. On an event, TSCTR contents (time-stamp) is captured first, and then TSCTR is reset to Zero. The Mod4 counter then increments to the next state. If TSCTR reaches FFFF FFFFh (maximum value), before the next event, it wraps around to 0000 0000h and continues, a CNTOVF (counter overflow) Flag is set, and an Interrupt (if enabled) occurs. The advantage of Delta-time Mode is that the CAP*n* contents directly give timing data without the need for CPU calculations: Period1 = T_1, Period2 = T_2,…etc. As shown in Figure 13-12, the CEVT1 event is a good trigger point to read the timing data, T_1, T_2, T_3, T_4 are all valid here.

Figure 13-12. Capture Sequence for Delta Mode Time-Stamp, Rising Edge Detect



Copyright © 2013–2016, Texas Instruments Incorporated

Table 13-3. ECAP Initialization for CAP Mode Delta Time, Rising Edge Trigger

Register	Bit	Value
ECCTL1	CAP1POL	EC_RISING
ECCTL1	CAP2POL	EC_RISING
ECCTL1	CAP3POL	EC_RISING
ECCTL1	CAP4POL	EC_RISING
ECCTL1	CTRRST1	EC_DELTA_MODE
ECCTL1	CTRRST2	EC_DELTA_MODE
ECCTL1	CTRRST3	EC_DELTA_MODE
ECCTL1	CTRRST4	EC_DELTA_MODE
ECCTL1	CAPLDEN	EC_ENABLE
ECCTL1	PRESCALE	EC_DIV1
ECCTL2	CAP_APWM	EC_CAP_MODE
ECCTL2	CONT_ONESHT	EC_CONTINUOUS
ECCTL2	SYNCO_SEL	EC_SYNCO_DIS
ECCTL2	SYNCI_EN	EC_DISABLE
ECCTL2	TSCTRSTOP	EC_RUN

Example 13-3. Code Snippet for CAP Mode Delta Time, Rising Edge Trigger

```
// Code snippet for CAP mode Delta Time, Rising edge trigger

// Run Time ( e.g. CEVT1 triggered ISR call)
//=========================================
// Note: here Time-stamp directly represents the Period value.
    Period4 = ECAPxRegs.CAP1;    // Fetch Time-Stamp captured at T1
    Period1 = ECAPxRegs.CAP2;    // Fetch Time-Stamp captured at T2
    Period2 = ECAPxRegs.CAP3;    // Fetch Time-Stamp captured at T3
    Period3 = ECAPxRegs.CAP4;    // Fetch Time-Stamp captured at T4
```

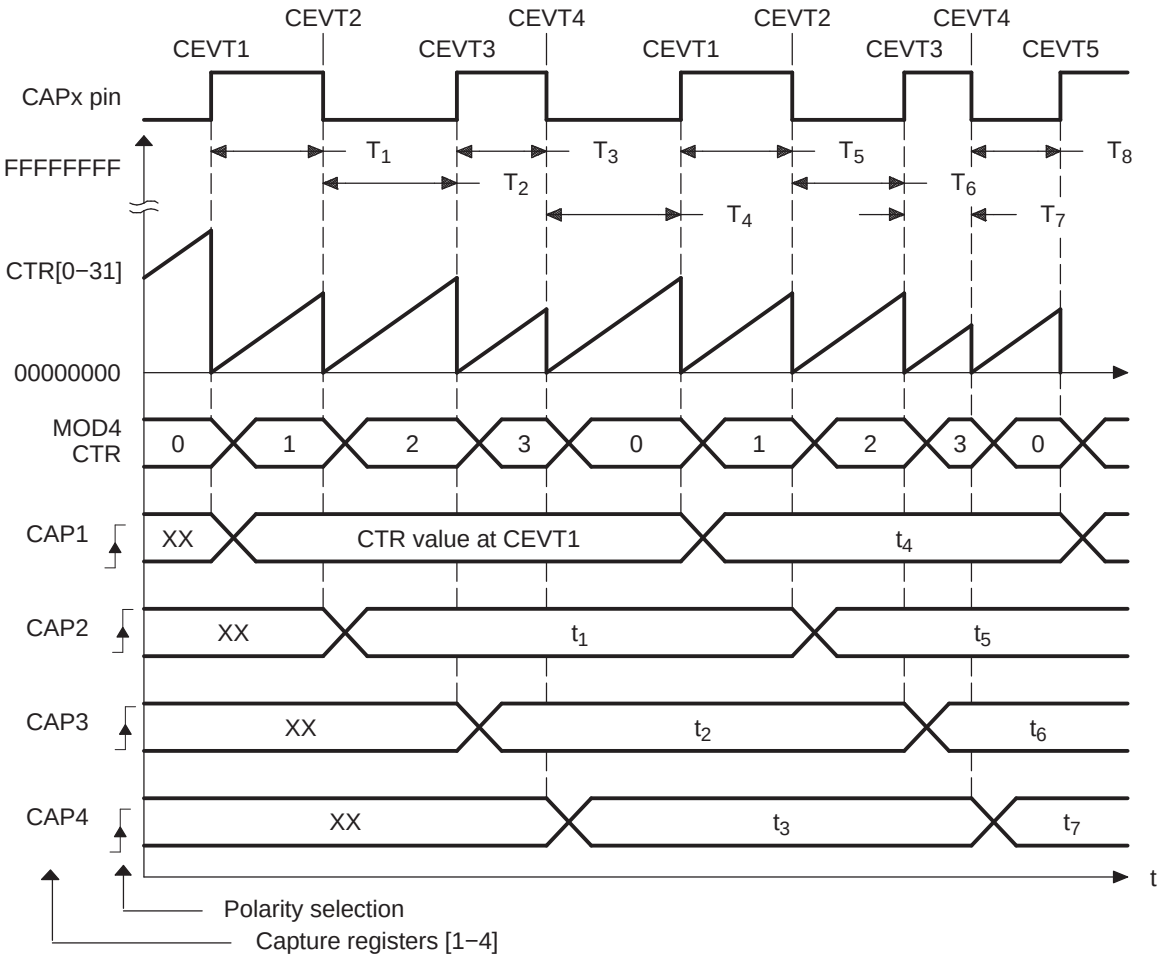

Copyright © 2013–2016, Texas Instruments Incorporated

13.3.4 Time Difference (Delta) Operation Rising and Falling Edge Trigger Example

In Figure 13-13 the eCAP operating mode is almost the same as in previous section except Capture events are qualified as either Rising or Falling edge, this now gives both Period and Duty cycle information: Period1 = T_1 + T_2, Period2 = T_3 + T_4, …etc Duty Cycle1 (on-time %) = T_1 / Period1 × 100%, etc Duty Cycle1 (off-time %) = T_2 / Period1 × 100%, etc

During initialization, you must write to the active registers for both period and compare. This will then automatically copy the init values into the shadow values. For subsequent compare updates, that is, during run-time, only the shadow registers must be used.

Figure 13-13. Capture Sequence for Delta Mode Time-Stamp, Rising and Falling Edge Detect



Copyright © 2013–2016, Texas Instruments Incorporated

Table 13-4. ECAP Initialization for CAP Mode Delta Time, Rising and Falling Edge Triggers

Register	Bit	Value
ECCTL1	CAP1POL	EC_RISING
ECCTL1	CAP2POL	EC_FALLING
ECCTL1	CAP3POL	EC_RISING
ECCTL1	CAP4POL	EC_FALLING
ECCTL1	CTRRST1	EC_DELTA_MODE
ECCTL1	CTRRST2	EC_DELTA_MODE
ECCTL1	CTRRST3	EC_DELTA_MODE
ECCTL1	CTRRST4	EC_DELTA_MODE
ECCTL1	CAPLDEN	EC_ENABLE
ECCTL1	PRESCALE	EC_DIV1
ECCTL2	CAP_APWM	EC_CAP_MODE
ECCTL2	CONT_ONESHT	EC_CONTINUOUS
ECCTL2	SYNCO_SEL	EC_SYNCO_DIS
ECCTL2	SYNCI_EN	EC_DISABLE
ECCTL2	TSCTRSTOP	EC_RUN

Example 13-4. Code Snippet for CAP Mode Delta Time, Rising and Falling Edge Triggers

```
// Code snippet for CAP mode Delta Time, Rising and Falling edge triggers

// Run Time ( e.g. CEVT1 triggered ISR call)
//=========================================
// Note: here Time-stamp directly represents the Duty cycle values.
    DutyOnTime1 =  ECAPxRegs.CAP2;    // Fetch Time-Stamp captured at T2
    DutyOffTime1 = ECAPxRegs.CAP3;    // Fetch Time-Stamp captured at T3
    DutyOnTime2 =  ECAPxRegs.CAP4;    // Fetch Time-Stamp captured at T4
    DutyOffTime2 = ECAPxRegs.CAP1;    // Fetch Time-Stamp captured at T1

    Period1 = DutyOnTime1 + DutyOffTime1;
    Period2 = DutyOnTime2 + DutyOffTime2;
```

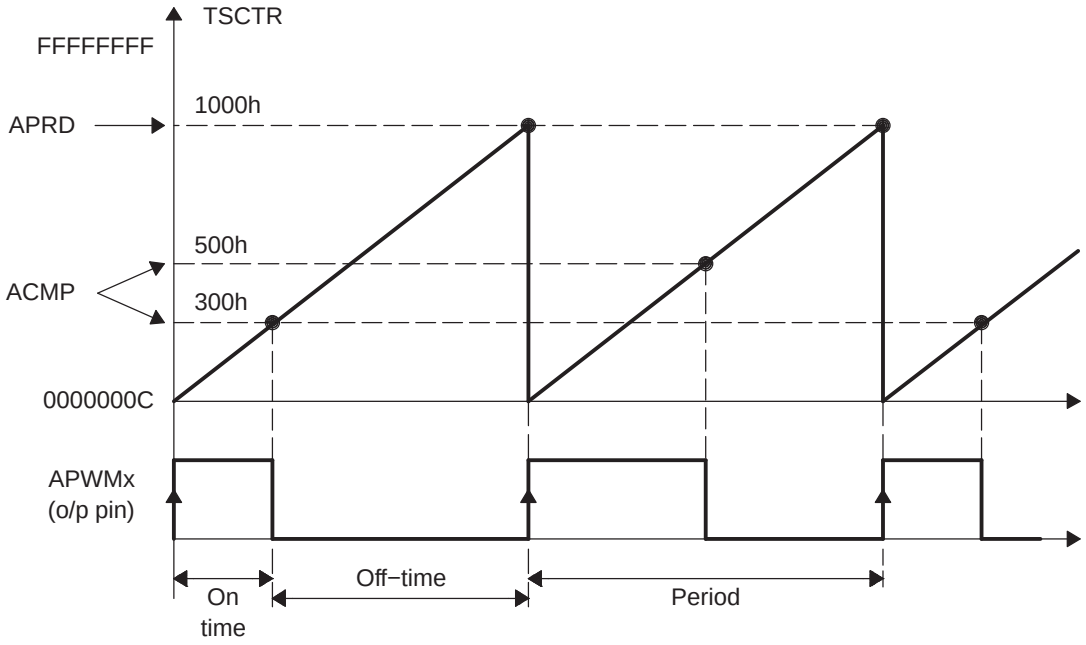

Copyright © 2013–2016, Texas Instruments Incorporated

13.3.5 Application of the APWM Mode

13.3.5.1 Simple PWM Generation (Independent Channel/s) Example

In this example, the eCAP module is configured to operate as a PWM generator. Here a very simple single channel PWM waveform is generated from output pin APWM*n*. The PWM polarity is active high, which means that the compare value (CAP2 reg is now a compare register) represents the on-time (high level) of the period. Alternatively, if the APWMPOL bit is configured for active low, then the compare value represents the off-time.

Figure 13-14. PWM Waveform Details of APWM Mode Operation



Copyright © 2013–2016, Texas Instruments Incorporated

Table 13-5. ECAP Initialization for APWM Mode

Register	Bit	Value
CAP1	CAP1	0x1000
CTRPHS	CTRPHS	0x0
ECCTL2	CAP_APWM	EC_APWM_MODE
ECCTL2	APWMPOL	EC_ACTV_HI
ECCTL2	SYNCI_EN	EC_DISABLE
ECCTL2	SYNCO_SEL	EC_SYNCO_DIS
ECCTL2	TSCTRSTOP	EC_RUN

Example 13-5. Code Snippet for APWM Mode

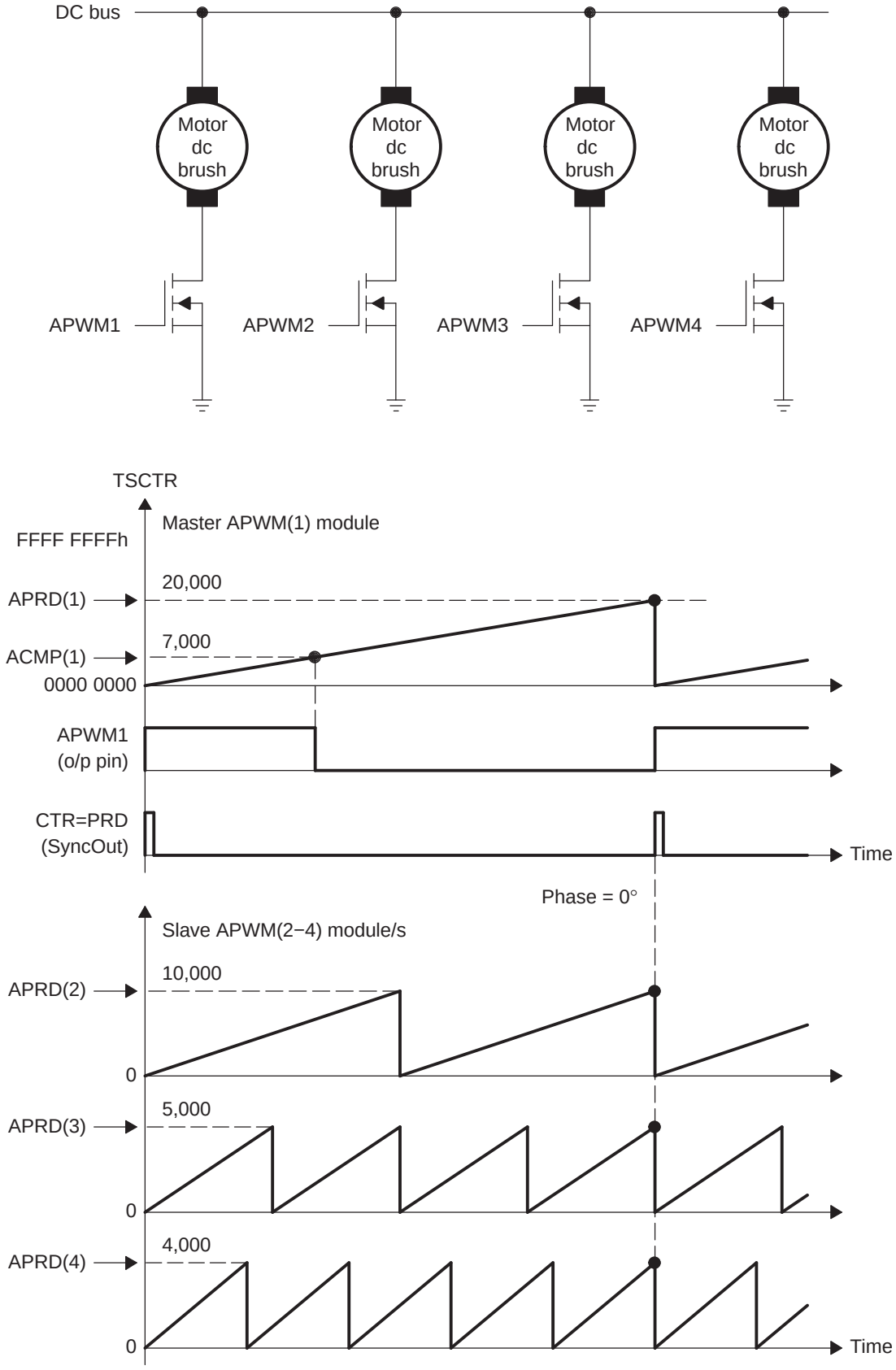
```
// Code snippet for APWM mode Example 1

// Run Time (Instant 1, e.g. ISR call)
//======================
   ECAPxRegs.CAP2 = 0x300;      // Set Duty cycle i.e. compare value

// Run Time (Instant 2, e.g. another ISR call)
//======================
   ECAPxRegs.CAP2 = 0x500;      // Set Duty cycle i.e. compare value
```

13.3.5.2 Multichannel PWM Generation with Synchronization Example

Figure 13-15 takes advantage of the synchronization feature between eCAP modules. Here 4 independent PWM channels are required with different frequencies, but at integer multiples of each other to avoid "beat" frequencies. Hence one eCAP module is configured as the Master and the remaining 3 are Slaves all receiving their synch pulse (CTR = PRD) from the master. Note the Master is chosen to have the lower frequency (F1 = 1/20,000) requirement. Here Slave2 Freq = 2 × F1, Slave3 Freq = 4 × F1 and Slave4 Freq = 5 × F1. Note here values are in decimal notation. Also, only the APWM1 output waveform is shown.

Copyright © 2013–2016, Texas Instruments Incorporated

Figure 13-15. Multichannel PWM Example Using 4 eCAP Modules



Copyright © 2013–2016, Texas Instruments Incorporated

Table 13-6. ECAP1 Initialization for Multichannel PWM Generation with Synchronization

Register	Bit	Value
CAP1	CAP1	20000
CTRPHS	CTRPHS	0
ECCTL2	CAP_APWM	EC_APWM_MODE
ECCTL2	APWMPOL	EC_ACTV_HI
ECCTL2	SYNCI_EN	EC_DISABLE
ECCTL2	SYNCO_SEL	EC_CTR_PRD
ECCTL2	TSCTRSTOP	EC_RUN

Table 13-7. ECAP2 Initialization for Multichannel PWM Generation with Synchronization

Register	Bit	Value
CAP1	CAP1	10000
CTRPHS	CTRPHS	0
ECCTL2	CAP_APWM	EC_APWM_MODE
ECCTL2	APWMPOL	EC_ACTV_HI
ECCTL2	SYNCI_EN	EC_ENABLE
ECCTL2	SYNCO_SEL	EC_SYNCI
ECCTL2	TSCTRSTOP	EC_RUN

Table 13-8. ECAP3 Initialization for Multichannel PWM Generation with Synchronization

Register	Bit	Value
CAP1	CAP1	5000
CTRPHS	CTRPHS	0
ECCTL2	CAP_APWM	EC_APWM_MODE
ECCTL2	APWMPOL	EC_ACTV_HI
ECCTL2	SYNCI_EN	EC_ENABLE
ECCTL2	SYNCO_SEL	EC_SYNCI
ECCTL2	TSCTRSTOP	EC_RUN

Table 13-9. ECAP4 Initialization for Multichannel PWM Generation with Synchronization

Register	Bit	Value
CAP1	CAP1	4000
CTRPHS	CTRPHS	0
ECCTL2	CAP_APWM	EC_APWM_MODE
ECCTL2	APWMPOL	EC_ACTV_HI
ECCTL2	SYNCI_EN	EC_ENABLE
ECCTL2	SYNCO_SEL	EC_SYNCO_DIS
ECCTL2	TSCTRSTOP	EC_RUN

Copyright © 2013–2016, Texas Instruments Incorporated

Example 13-6. Code Snippet for Multichannel PWM Generation with Synchronization

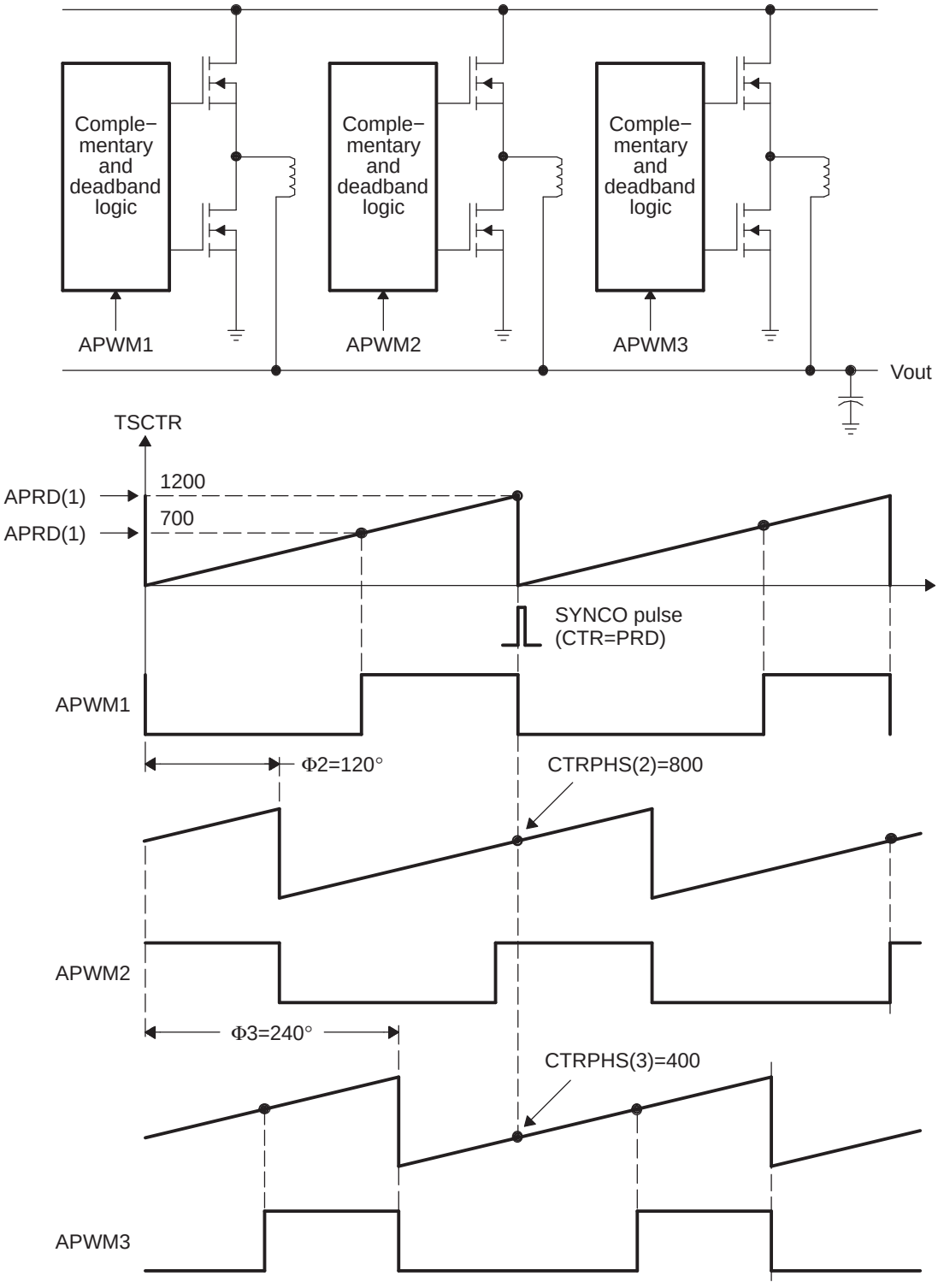
```
// Code snippet for APWM mode Example 2

// Run Time (Note: Example execution of one run-time instant)
//=============================================================
   ECAP1Regs.CAP2 = 7000;    // Set Duty cycle i.e., compare value = 7000
   ECAP2Regs.CAP2 = 2000;    // Set Duty cycle i.e., compare value = 2000
   ECAP3Regs.CAP2 = 550;     // Set Duty cycle i.e., compare value = 550
   ECAP4Regs.CAP2 = 6500;    // Set Duty cycle i.e., compare value = 6500
```

13.3.5.3 Multichannel PWM Generation with Phase Control Example

In Figure 13-16, the Phase control feature of the APWM mode is used to control a 3 phase Interleaved DC/DC converter topology. This topology requires each phase to be off-set by 120° from each other. Hence if “Leg” 1 (controlled by APWM1) is the reference Leg (or phase), that is, 0°, then Leg 2 need 120° off-set and Leg 3 needs 240° off-set. The waveforms in Figure 13-16 show the timing relationship between each of the phases (Legs). Note eCAP1 module is the Master and issues a sync out pulse to the slaves (modules 2, 3) whenever TSCTR = Period value.

Copyright © 2013–2016, Texas Instruments Incorporated

Figure 13-16. Multiphase (channel) Interleaved PWM Example Using 3 eCAP Modules



Copyright © 2013–2016, Texas Instruments Incorporated

Table 13-10. ECAP1 Initialization for Multichannel PWM Generation with Phase Control

Register	Bit	Value
CAP1	CAP1	1200
CTRPHS	CTRPHS	0
ECCTL2	CAP_APWM	EC_APWM_MODE
ECCTL2	APWMPOL	EC_ACTV_HI
ECCTL2	SYNCI_EN	EC_DISABLE
ECCTL2	SYNCO_SEL	EC_CTR_PRD
ECCTL2	TSCTRSTOP	EC_RUN

Table 13-11. ECAP2 Initialization for Multichannel PWM Generation with Phase Control

Register	Bit	Value
CAP1	CAP1	1200
CTRPHS	CTRPHS	800
ECCTL2	CAP_APWM	EC_APWM_MODE
ECCTL2	APWMPOL	EC_ACTV_HI
ECCTL2	SYNCI_EN	EC_ENABLE
ECCTL2	SYNCO_SEL	EC_SYNCI
ECCTL2	TSCTRSTOP	EC_RUN

Table 13-12. ECAP3 Initialization for Multichannel PWM Generation with Phase Control

Register	Bit	Value
CAP1	CAP1	1200
CTRPHS	CTRPHS	400
ECCTL2	CAP_APWM	EC_APWM_MODE
ECCTL2	APWMPOL	EC_ACTV_HI
ECCTL2	SYNCI_EN	EC_ENABLE
ECCTL2	SYNCO_SEL	EC_SYNCO_DIS
ECCTL2	TSCTRSTOP	EC_RUN

Example 13-7. Code Snippet for Multichannel PWM Generation with Phase Control

```
// Code snippet for APWM mode Example 3


// Run Time (Note: Example execution of one run-time instant)
//=============================================================
// All phases are set to the same duty cycle
  ECAP1Regs.CAP2 = 700;     // Set Duty cycle i.e. compare value = 700
  ECAP2Regs.CAP2 = 700;     // Set Duty cycle i.e. compare value = 700
  ECAP3Regs.CAP2 = 700;     // Set Duty cycle i.e. compare value = 700
```


Copyright © 2013–2016, Texas Instruments Incorporated

13.4 Registers

Table 13-13 shows the eCAP module control and status register set. All 32-bit registers are aligned on even address boundaries and are organized in little-endian mode. The 16 least-significant bits of a 32-bit register are located on lowest address (even address).

NOTE: In APWM mode, writing to CAP1/CAP2 active registers also writes the same value to the corresponding shadow registers CAP3/CAP4. This emulates immediate mode. Writing to the shadow registers CAP3/CAP4 invokes the shadow mode.

Table 13-13. Control and Status Register Set

Offset	Acronym	Description	Size (×16)	Section
0h	TSCTR	Time-Stamp Counter Register	2	Section 13.4.1
4h	CTRPHS	Counter Phase Offset Value Register	2	Section 13.4.2
8h	CAP1	Capture 1 Register	2	Section 13.4.3
Ch	CAP2	Capture 2 Register	2	Section 13.4.4
10h	CAP3	Capture 3 Register	2	Section 13.4.5
14h	CAP4	Capture 4 Register	2	Section 13.4.6
28h	ECCTL1	Capture Control Register 1	1	Section 13.4.7
2Ah	ECCTL2	Capture Control Register 2	1	Section 13.4.8
2Ch	ECEINT	Capture Interrupt Enable Register	1	Section 13.4.9
2Eh	ECFLG	Capture Interrupt Flag Register	1	Section 13.4.10
30h	ECCLR	Capture Interrupt Clear Register	1	Section 13.4.11
32h	ECFRC	Capture Interrupt Force Register	1	Section 13.4.12
5Ch	REVID	Revision ID Register	2	Section 13.4.13

13.4.1 Time-Stamp Counter Register (TSCTR)

The time-stamp counter register (TSCTR) is shown in Figure 13-17 and described in Table 13-14.

Figure 13-17. Time-Stamp Counter Register (TSCTR)

31	0
TSCTR	
R/W-0	

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 13-14. Time-Stamp Counter Register (TSCTR) Field Descriptions

Bit	Field	Value	Description
31-0	TSCTR	0-FFFF FFFFh	Active 32-bit counter register that is used as the capture time-base

Copyright © 2013–2016, Texas Instruments Incorporated

13.4.2 Counter Phase Control Register (CTRPHS)

The counter phase control register (CTRPHS) is shown in Figure 13-18 and described in Table 13-15.

Figure 13-18. Counter Phase Control Register (CTRPHS)

31	0
CTRPHS	
R/W-0	

LEGEND: R/W = Read/Write; -*n* = value after reset

Table 13-15. Counter Phase Control Register (CTRPHS) Field Descriptions

Bit	Field	Value	Description
31-0	CTRPHS	0-FFFF FFFFh	Counter phase value register that can be programmed for phase lag/lead. This register shadows TSCTR and is loaded into TSCTR upon either a SYNCI event or S/W force via a control bit. Used to achieve phase control synchronization with respect to other eCAP and EPWM time-bases.

13.4.3 Capture 1 Register (CAP1)

The capture 1 register (CAP1) is shown in Figure 13-19 and described in Table 13-16.

Figure 13-19. Capture 1 Register (CAP1)

31	0
CAP1	
R/W-0	

LEGEND: R/W = Read/Write; -*n* = value after reset

Table 13-16. Capture 1 Register (CAP1) Field Descriptions

Bit	Field	Value	Description
31-0	CAP1	0-FFFF FFFFh	This register can be loaded (written) by: • Time-Stamp (i.e., counter value) during a capture event • Software - may be useful for test purposes • APRD active register when used in APWM mode

Copyright © 2013–2016, Texas Instruments Incorporated

13.4.4 Capture 2 Register (CAP2)

The capture 2 register (CAP2) is shown in Figure 13-20 and described in Table 13-17.

Figure 13-20. Capture 2 Register (CAP2)

31	0
CAP2	
R/W-0	

LEGEND: R/W = Read/Write; -*n* = value after reset

Table 13-17. Capture 2 Register (CAP2) Field Descriptions

Bit	Field	Value	Description
31-0	CAP2	0-FFFF FFFFh	This register can be loaded (written) by: • Time-Stamp (i.e., counter value) during a capture event • Software - may be useful for test purposes • ACMP active register when used in APWM mode

13.4.5 Capture 3 Register (CAP3)

The capture 3 register (CAP3) is shown in Figure 13-21 and described in Table 13-18.

Figure 13-21. Capture 3 Register (CAP3)

31	0
CAP3	
R/W-0	

LEGEND: R/W = Read/Write; -*n* = value after reset

Table 13-18. Capture 3 Register (CAP3) Field Descriptions

Bit	Field	Value	Description
31-0	CAP3	0-FFFF FFFFh	In CMP mode, this is a time-stamp capture register. In APWM mode, this is the period shadow (APRD) register. You update the PWM period value through this register. In this mode, CAP3 shadows CAP1.

Copyright © 2013–2016, Texas Instruments Incorporated

13.4.6 Capture 4 Register (CAP4)

The capture 4 register (CAP4) is shown in Figure 13-22 and described in Table 13-19.

Figure 13-22. Capture 4 Register (CAP4)

31	0
CAP4	
R/W-0	

LEGEND: R/W = Read/Write; -*n* = value after reset

Table 13-19. Capture 4 Register (CAP4) Field Descriptions

Bit	Field	Value	Description
31-0	CAP4	0-FFFF FFFFh	In CMP mode, this is a time-stamp capture register. In APWM mode, this is the compare shadow (ACMP) register. You update the PWM compare value through this register. In this mode, CAP4 shadows CAP2.

13.4.7 ECAP Control Register 1 (ECCTL1)

The ECAP control register 1 (ECCTL1) is shown in Figure 13-23 and described in Table 13-20.

Figure 13-23. ECAP Control Register 1 (ECCTL1)

15	14	13	9	8
FREE/SOFT		PRESCALE		CAPLDEN
R/W-0		R/W-0		R/W-0

7	6	5	4	3	2	1	0
CTRRST4	CAP4POL	CTRRST3	CAP3POL	CTRRST2	CAP2POL	CTRRST1	CAP1POL
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0

LEGEND: R/W = Read/Write; -*n* = value after reset

Table 13-20. ECAP Control Register 1 (ECCTL1) Field Descriptions

Bit	Field	Value	Description
15-14	FREE/SOFT	0-3h	Emulation Control
		0	TSCTR counter stops immediately on emulation suspend
		1h	TSCTR counter runs until = 0
		2h-3h	TSCTR counter is unaffected by emulation suspend (Run Free)
13-9	PRESCALE	0-1Fh	Event Filter prescale select
		0	Divide by 1 (i.e,. no prescale, by-pass the prescaler)
		1	Divide by 2
		2h	Divide by 4
		3h	Divide by 6
		4h	Divide by 8
		5h	Divide by 10
		...	
		1Eh	Divide by 60
		1Fh	Divide by 62
8	CAPLDEN		Enable Loading of CAP1-4 registers on a capture event
		0	Disable CAP1-4 register loads at capture event time.
		1	Enable CAP1-4 register loads at capture event time.

Copyright © 2013–2016, Texas Instruments Incorporated

Table 13-20. ECAP Control Register 1 (ECCTL1) Field Descriptions (continued)

Bit	Field	Value	Description
7	CTRRST4		Counter Reset on Capture Event 4
		0	*Do not* reset counter on Capture Event 4 (absolute time stamp operation)
		1	Reset counter after Capture Event 4 time-stamp has been captured (used in difference mode operation)
6	CAP4POL		Capture Event 4 Polarity select
		0	Capture Event 4 triggered on a rising edge (RE)
		1	Capture Event 4 triggered on a falling edge (FE)
5	CTRRST3		Counter Reset on Capture Event 3
		0	*Do not* reset counter on Capture Event 3 (absolute time stamp)
		1	Reset counter after Event 3 time-stamp has been captured (used in difference mode operation)
4	CAP3POL		Capture Event 3 Polarity select
		0	Capture Event 3 triggered on a rising edge (RE)
		1	Capture Event 3 triggered on a falling edge (FE)
3	CTRRST2		Counter Reset on Capture Event 2
		0	*Do not* reset counter on Capture Event 2 (absolute time stamp)
		1	Reset counter after Event 2 time-stamp has been captured (used in difference mode operation)
2	CAP2POL		Capture Event 2 Polarity select
		0	Capture Event 2 triggered on a rising edge (RE)
		1	Capture Event 2 triggered on a falling edge (FE)
1	CTRRST1		Counter Reset on Capture Event 1
		0	*Do not* reset counter on Capture Event 1 (absolute time stamp)
		1	Reset counter after Event 1 time-stamp has been captured (used in difference mode operation)
0	CAP1POL		Capture Event 1 Polarity select
		0	Capture Event 1 triggered on a rising edge (RE)
		1	Capture Event 1 triggered on a falling edge (FE)

Copyright © 2013–2016, Texas Instruments Incorporated

13.4.8 ECAP Control Register 2 (ECCTL2)

The ECAP control register 2 (ECCTL2) is shown in Figure 13-24 and described in Table 13-21.

Figure 13-24. ECAP Control Register 2 (ECCTL2)

15	14	13	12	11	10	9	8
Reserved					APWMPOL	CAP/APWM	SWSYNC
R-0					R/W-0	R/W-0	R/W-0

7	6	5	4	3	2	1	0
SYNCO_SEL		SYNCI_EN	TSCTRSTOP	RE-ARM	STOP_WRAP		CONT/ONESHT
R/W-0		R/W-0	R/W-0	R/W-0	R/W-1		R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 13-21. ECAP Control Register 2 (ECCTL2) Field Descriptions

Bit	Field	Value	Description
15-11	Reserved	0	Reserved
10	APWMPOL		APWM output polarity select. This is applicable only in APWM operating mode
		0	Output is active high (Compare value defines high time)
		1	Output is active low (Compare value defines low time)
9	CAP/APWM		CAP/APWM operating mode select
		0	ECAP module operates in capture mode. This mode forces the following configuration: • Inhibits TSCTR resets via CTR = PRD event • Inhibits shadow loads on CAP1 and 2 registers • Permits user to enable CAP1-4 register load • ECAP*n*/APWM*n* pin operates as a capture input
		1	ECAP module operates in APWM mode. This mode forces the following configuration: • Resets TSCTR on CTR = PRD event (period boundary • Permits shadow loading on CAP1 and 2 registers • Disables loading of time-stamps into CAP1-4 registers • ECAP*n*/APWM*n* pin operates as a APWM output
8	SWSYNC		Software-forced Counter (TSCTR) Synchronizing. This provides a convenient software method to synchronize some or all ECAP time bases. In APWM mode, the synchronizing can also be done via the CTR = PRD event.
		0	Writing a zero has no effect. Reading always returns a zero
		1	Writing a one forces a TSCTR shadow load of current ECAP module and any ECAP modules down-stream providing the SYNCO_SEL bits are 0,0. After writing a 1, this bit returns to a zero. Note: Selection CTR = PRD is meaningful only in APWM mode; however, you can choose it in CAP mode if you find doing so useful.
7-6	SYNCO_SEL	0-3h	Sync-Out Select
		0	Select sync-in event to be the sync-out signal (pass through)
		1h	Select CTR = PRD event to be the sync-out signal
		2h	Disable sync out signal
		3h	Disable sync out signal
5	SYNCI_EN		Counter (TSCTR) Sync-In select mode
		0	Disable sync-in option
		1	Enable counter (TSCTR) to be loaded from CTRPHS register upon either a SYNCI signal or a S/W force event.
4	TSCTRSTOP		Time Stamp (TSCTR) Counter Stop (freeze) Control
		0	TSCTR stopped
		1	TSCTR free-running

Copyright © 2013–2016, Texas Instruments Incorporated

Table 13-21. ECAP Control Register 2 (ECCTL2) Field Descriptions (continued)

Bit	Field	Value	Description
3	RE-ARM		One-Shot Re-Arming Control, that is, wait for stop trigger. Note: The re-arm function is valid in one shot or continuous mode.
		0	Has no effect (reading always returns a 0)
		1	Arms the one-shot sequence as follows: 1) Resets the Mod4 counter to zero 2) Unfreezes the Mod4 counter 3) Enables capture register loads
2-1	STOP_WRAP	0-3h	Stop value for one-shot mode. This is the number (between 1-4) of captures allowed to occur before the CAP(1-4) registers are frozen, that is, capture sequence is stopped. Wrap value for continuous mode. This is the number (between 1-4) of the capture register in which the circular buffer wraps around and starts again.
		0	Stop after Capture Event 1 in one-shot mode. Wrap after Capture Event 1 in continuous mode.
		1h	Stop after Capture Event 2 in one-shot mode. Wrap after Capture Event 2 in continuous mode.
		2h	Stop after Capture Event 3 in one-shot mode. Wrap after Capture Event 3 in continuous mode.
		3h	Stop after Capture Event 4 in one-shot mode. Wrap after Capture Event 4 in continuous mode.
			Notes: STOP_WRAP is compared to Mod4 counter and, when equal, 2 actions occur: • Mod4 counter is stopped (frozen) • Capture register loads are inhibited In one-shot mode, further interrupt events are blocked until re-armed.
0	CONT/ONESHT		Continuous or one-shot mode control (applicable only in capture mode)
		0	Operate in continuous mode
		1	Operate in one-shot mode

13.4.9 ECAP Interrupt Enable Register (ECEINT)

The ECAP interrupt enable register (ECEINT) is shown in Figure 13-25 and described in Table 13-22.

The interrupt enable bits (CEVT*n*) block any of the selected events from generating an interrupt. Events will still be latched into the flag bit (ECFLG register) and can be forced/cleared via the ECFRC/ECCLR registers.

The proper procedure for configuring peripheral modes and interrupts is:

1. Disable global interrupts
2. Stop eCAP counter
3. Disable eCAP interrupts
4. Configure peripheral registers
5. Clear spurious eCAP interrupt flags
6. Enable eCAP interrupts
7. Start eCAP counter
8. Enable global interrupts

Copyright © 2013–2016, Texas Instruments Incorporated

Figure 13-25. ECAP Interrupt Enable Register (ECEINT)

15							8
Reserved							
R-0							
7	6	5	4	3	2	1	0
CTR=CMP	CTR=PRD	CTROVF	CEVT4	CEVT3	CEVT2	CETV1	Reserved
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 13-22. ECAP Interrupt Enable Register (ECEINT) Field Descriptions

Bit	Field	Value	Description
15-8	Reserved	0	Reserved
7	CTR=CMP		Counter Equal Compare Interrupt Enable
		0	Disable Compare Equal as an Interrupt source
		1	Enable Compare Equal as an Interrupt source
6	CTR=PRD		Counter Equal Period Interrupt Enable
		0	Disable Period Equal as an Interrupt source
		1	Enable Period Equal as an Interrupt source
5	CTROVF		Counter Overflow Interrupt Enable
		0	Disable counter Overflow as an Interrupt source
		1	Enable counter Overflow as an Interrupt source
4	CEVT4		Capture Event 4 Interrupt Enable
		0	Disable Capture Event 4 as an Interrupt source
		1	Enable Capture Event 4 as an Interrupt source
3	CEVT3		Capture Event 3 Interrupt Enable
		0	Disable Capture Event 3 as an Interrupt source
		1	Enable Capture Event 3 as an Interrupt source
2	CEVT2		Capture Event 2 Interrupt Enable
		0	Disable Capture Event 2 as an Interrupt source
		1	Enable Capture Event 2 as an Interrupt source
1	CEVT1		Capture Event 1 Interrupt Enable
		0	Disable Capture Event 1 as an Interrupt source
		1	Enable Capture Event 1 as an Interrupt source
0	Reserved	0	Reserved

Copyright © 2013–2016, Texas Instruments Incorporated

13.4.10 ECAP Interrupt Flag Register (ECFLG)

The ECAP interrupt flag register (ECFLG) is shown in Figure 13-26 and described in Table 13-23.

Figure 13-26. ECAP Interrupt Flag Register (ECFLG)

15							8
Reserved							
R-0							
7	6	5	4	3	2	1	0
CTR=CMP	CTR=PRD	CTROVF	CEVT4	CETV3	CEVT2	CETV1	INT
R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0

LEGEND: R = Read only; -*n* = value after reset

Table 13-23. ECAP Interrupt Flag Register (ECFLG) Field Descriptions

Bit	Field	Value	Description
15-8	Reserved	0	Reserved
7	CTR=CMP		Compare Equal Compare Status Flag. This flag is only active in APWM mode.
		0	Indicates no event occurred
		1	Indicates the counter (TSCTR) reached the compare register value (ACMP)
6	CTR=PRD		Counter Equal Period Status Flag. This flag is only active in APWM mode.
		0	Indicates no event occurred
		1	Indicates the counter (TSCTR) reached the period register value (APRD) and was reset.
5	CTROVF		Counter Overflow Status Flag. This flag is active in CAP and APWM mode.
		0	Indicates no event occurred.
		1	Indicates the counter (TSCTR) has made the transition from 0xFFFFFFFF to 0x00000000
4	CEVT4		Capture Event 4 Status Flag This flag is only active in CAP mode.
		0	Indicates no event occurred
		1	Indicates the fourth event occurred at ECAP*n* pin
3	CEVT3		Capture Event 3 Status Flag. This flag is active only in CAP mode.
		0	Indicates no event occurred.
		1	Indicates the third event occurred at ECAP*n* pin.
2	CEVT2		Capture Event 2 Status Flag. This flag is only active in CAP mode.
		0	Indicates no event occurred.
		1	Indicates the second event occurred at ECAP*n* pin.
1	CEVT1		Capture Event 1 Status Flag. This flag is only active in CAP mode.
		0	Indicates no event occurred.
		1	Indicates the first event occurred at ECAP*n* pin.
0	INT		Global Interrupt Status Flag
		0	Indicates no interrupt generated.
		1	Indicates that an interrupt was generated.

Copyright © 2013–2016, Texas Instruments Incorporated

13.4.11 ECAP Interrupt Clear Register (ECCLR)

The ECAP interrupt clear register (ECCLR) is shown in Figure 13-27 and described in Table 13-24.

Figure 13-27. ECAP Interrupt Clear Register (ECCLR)

15							8
Reserved							
R-0							
7	6	5	4	3	2	1	0
CTR=CMP	CTR=PRD	CTROVF	CEVT4	CETV3	CETV2	CETV1	INT
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 13-24. ECAP Interrupt Clear Register (ECCLR) Field Descriptions

Bit	Field	Value	Description
15-8	Reserved	0	Reserved
7	CTR=CMP		Counter Equal Compare Status Flag
		0	Writing a 0 has no effect. Always reads back a 0
		1	Writing a 1 clears the CTR=CMP flag condition
6	CTR=PRD		Counter Equal Period Status Flag
		0	Writing a 0 has no effect. Always reads back a 0
		1	Writing a 1 clears the CTR=PRD flag condition
5	CTROVF		Counter Overflow Status Flag
		0	Writing a 0 has no effect. Always reads back a 0
		1	Writing a 1 clears the CTROVF flag condition
4	CEVT4		Capture Event 4 Status Flag
		0	Writing a 0 has no effect. Always reads back a 0.
		1	Writing a 1 clears the CEVT3 flag condition.
3	CEVT3		Capture Event 3 Status Flag
		0	Writing a 0 has no effect. Always reads back a 0.
		1	Writing a 1 clears the CEVT3 flag condition.
2	CEVT2		Capture Event 2 Status Flag
		0	Writing a 0 has no effect. Always reads back a 0.
		0	Writing a 1 clears the CEVT2 flag condition.
1	CEVT1		Capture Event 1 Status Flag
		0	Writing a 0 has no effect. Always reads back a 0.
		1	Writing a 1 clears the CEVT1 flag condition.
0	INT		Global Interrupt Clear Flag
		0	Writing a 0 has no effect. Always reads back a 0.
		1	Writing a 1 clears the INT flag and enable further interrupts to be generated if any of the event flags are set to 1.

Copyright © 2013–2016, Texas Instruments Incorporated

13.4.12 ECAP Interrupt Forcing Register (ECFRC)

The ECAP interrupt forcing register (ECFRC) is shown in Figure 13-28 and described in Table 13-25.

Figure 13-28. ECAP Interrupt Forcing Register (ECFRC)

15	14	13	12	11	10	9	8
Reserved							
R-0							

7	6	5	4	3	2	1	0
CTR=CMP	CTR=PRD	CTROVF	CEVT4	CETV3	CETV2	CETV1	Reserved
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 13-25. ECAP Interrupt Forcing Register (ECFRC) Field Descriptions

Bit	Field	Value	Description
15-8	Reserved	0	Reserved
7	CTR=CMP		Force Counter Equal Compare Interrupt
		0	No effect. Always reads back a 0.
		1	Writing a 1 sets the CTR=CMP flag bit.
6	CTR=PRD		Force Counter Equal Period Interrupt
		0	No effect. Always reads back a 0.
		1	Writing a 1 sets the CTR=PRD flag bit.
5	CTROVF		Force Counter Overflow
		0	No effect. Always reads back a 0.
		1	Writing a 1 to this bit sets the CTROVF flag bit.
4	CEVT4		Force Capture Event 4
		0	No effect. Always reads back a 0.
		1	Writing a 1 sets the CEVT4 flag bit
3	CEVT3		Force Capture Event 3
		0	No effect. Always reads back a 0.
		1	Writing a 1 sets the CEVT3 flag bit
2	CEVT2		Force Capture Event 2
		0	No effect. Always reads back a 0.
		1	Writing a 1 sets the CEVT2 flag bit.
1	CEVT1		Force Capture Event 1
		0	No effect. Always reads back a 0.
		1	Writing a 1 sets the CEVT1 flag bit.
0	Reserved	0	Reserved

Copyright © 2013–2016, Texas Instruments Incorporated

13.4.13 Revision ID Register (REVID)

The revision ID register (REVID) is shown in Figure 13-29 and described in Table 13-26.

Figure 13-29. Revision ID Register (REVID)

31	0
REV	
R-44D2 2100h	

LEGEND: R = Read only; -*n* = value after reset

Table 13-26. Revision ID Register (REVID) Field Descriptions

Bit	Field	Value	Description
31-0	REV	44D2 2100h	Revision ID.

Copyright © 2013–2016, Texas Instruments Incorporated

Enhanced High-Resolution Pulse-Width Modulator (eHRPWM)

This chapter describes the enhanced high-resolution pulse-width modulator (eHRPWM).

Topic **Page**

14.1 Introduction **223**
14.2 Architecture **228**
14.3 Applications to Power Topologies **287**
14.4 Registers **311**

Copyright © 2013–2016, Texas Instruments Incorporated

14.1 Introduction

14.1.1 Introduction

An effective PWM peripheral must be able to generate complex pulse width waveforms with minimal CPU overhead or intervention. It needs to be highly programmable and very flexible while being easy to understand and use. The ePWM unit described here addresses these requirements by allocating all needed timing and control resources on a per PWM channel basis. Cross coupling or sharing of resources has been avoided; instead, the ePWM is built up from smaller single channel modules with separate resources and that can operate together as required to form a system. This modular approach results in an orthogonal architecture and provides a more transparent view of the peripheral structure, helping users to understand its operation quickly.

In this chapter, the letter x within a signal or module name is used to indicate a generic ePWM instance on a device. For example, output signals EPWMxA and EPWMxB refer to the output signals from the ePWMx instance. Thus, EPWM1A and EPWM1B belong to ePWM1 and, likewise, EPWM4A and EPWM4B belong to ePWM4.

14.1.2 Submodule Overview

The ePWM module represents one complete PWM channel composed of two PWM outputs: EPWMxA and EPWMxB. Multiple ePWM modules are instanced within a device as shown in Figure 14-1. Each ePWM instance is identical with one exception. Some instances include a hardware extension that allows more precise control of the PWM outputs. This extension is the high-resolution pulse width modulator (HRPWM) and is described in Section 14.2.10. See your device-specific data manual to determine which ePWM instances include this feature. Each ePWM module is indicated by a numerical value starting with 1. For example ePWM1 is the first instance and ePWM3 is the 3rd instance in the system and ePWMx indicates any instance.

The ePWM modules are chained together via a clock synchronization scheme that allows them to operate as a single system when required. Additionally, this synchronization scheme can be extended to the capture peripheral modules (eCAP). The number of modules is device-dependent and based on target application needs. Modules can also operate stand-alone.

Each ePWM module supports the following features:

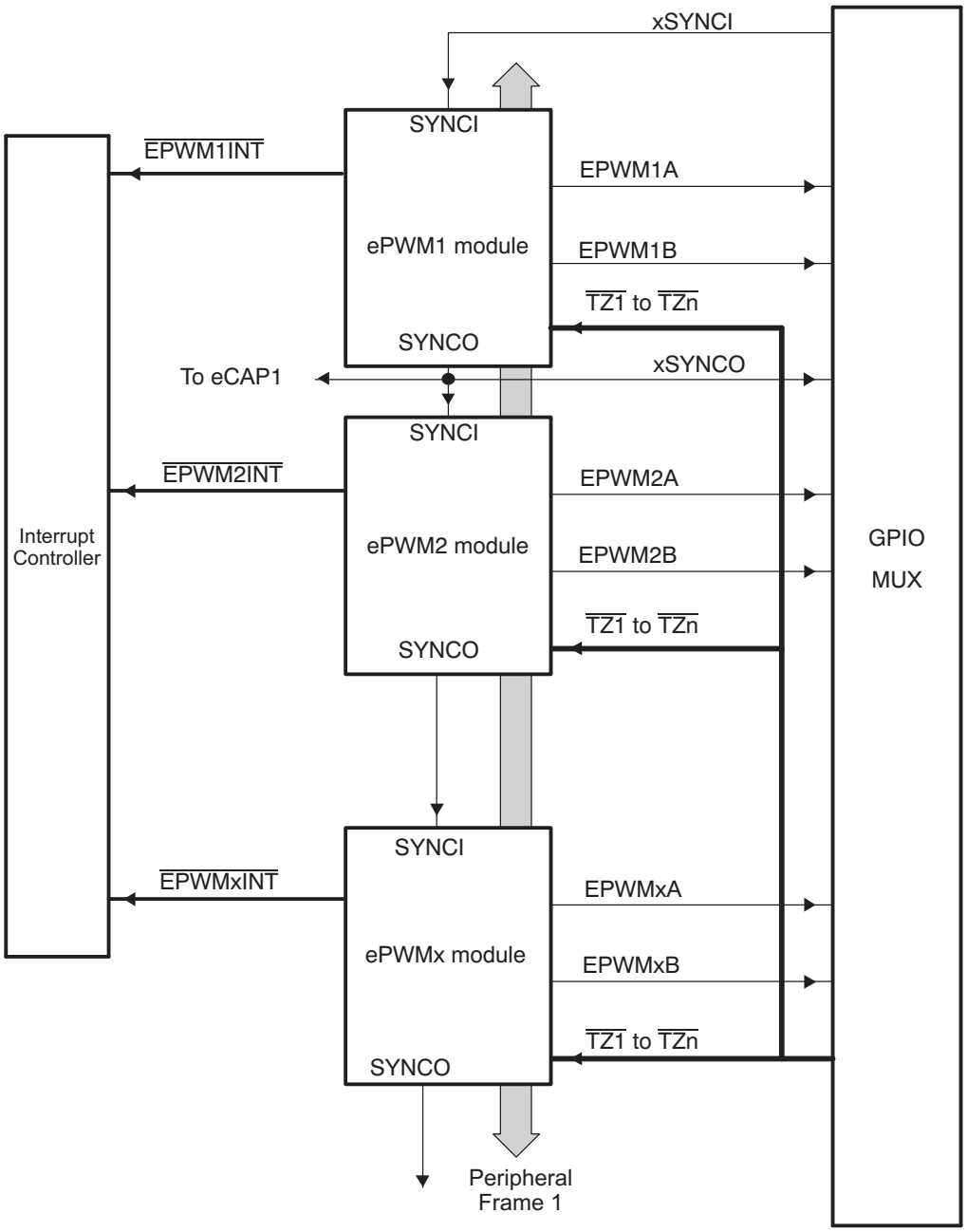
- Dedicated 16-bit time-base counter with period and frequency control
- Two PWM outputs (EPWMxA and EPWMxB) that can be used in the following configurations::
  - Two independent PWM outputs with single-edge operation
  - Two independent PWM outputs with dual-edge symmetric operation
  - One independent PWM output with dual-edge asymmetric operation
- Asynchronous override control of PWM signals through software.
- Programmable phase-control support for lag or lead operation relative to other ePWM modules.
- Hardware-locked (synchronized) phase relationship on a cycle-by-cycle basis.
- Dead-band generation with independent rising and falling edge delay control.
- Programmable trip zone allocation of both cycle-by-cycle trip and one-shot trip on fault conditions.
- A trip condition can force either high, low, or high-impedance state logic levels at PWM outputs.
- Programmable event prescaling minimizes CPU overhead on interrupts.
- PWM chopping by high-frequency carrier signal, useful for pulse transformer gate drives.

Each ePWM module is connected to the input/output signals shown in Figure 14-1. The signals are described in detail in subsequent sections.

The order in which the ePWM modules are connected may differ from what is shown in Figure 14-1. See Section 14.2.3.3.2 for the synchronization scheme for a particular device. Each ePWM module consists of seven submodules and is connected within a system via the signals shown in Figure 14-2.

Copyright © 2013–2016, Texas Instruments Incorporated

Figure 14-1. Multiple ePWM Modules



Copyright © 2013–2016, Texas Instruments Incorporated

Figure 14-2. Submodules and Signal Connections for an ePWM Module

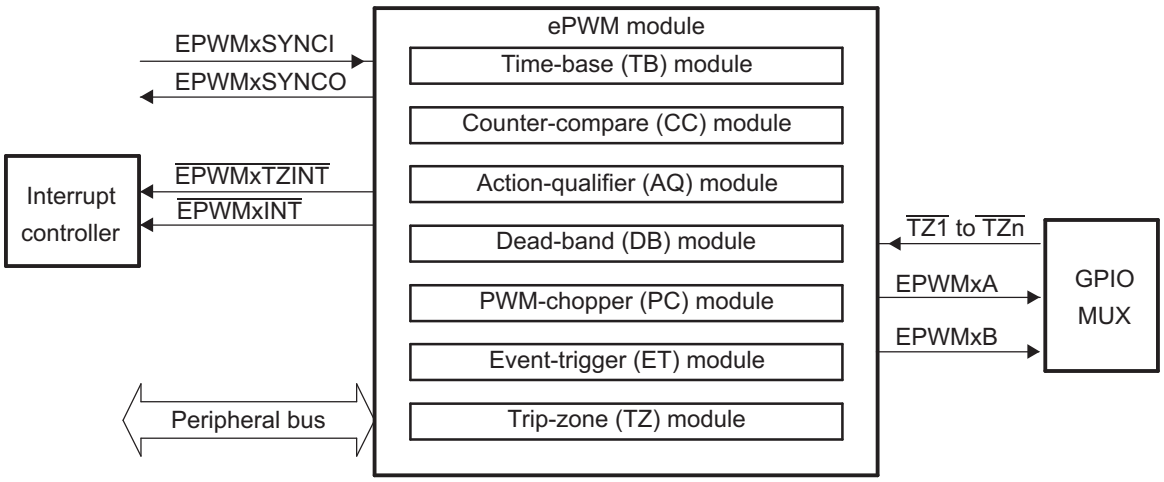


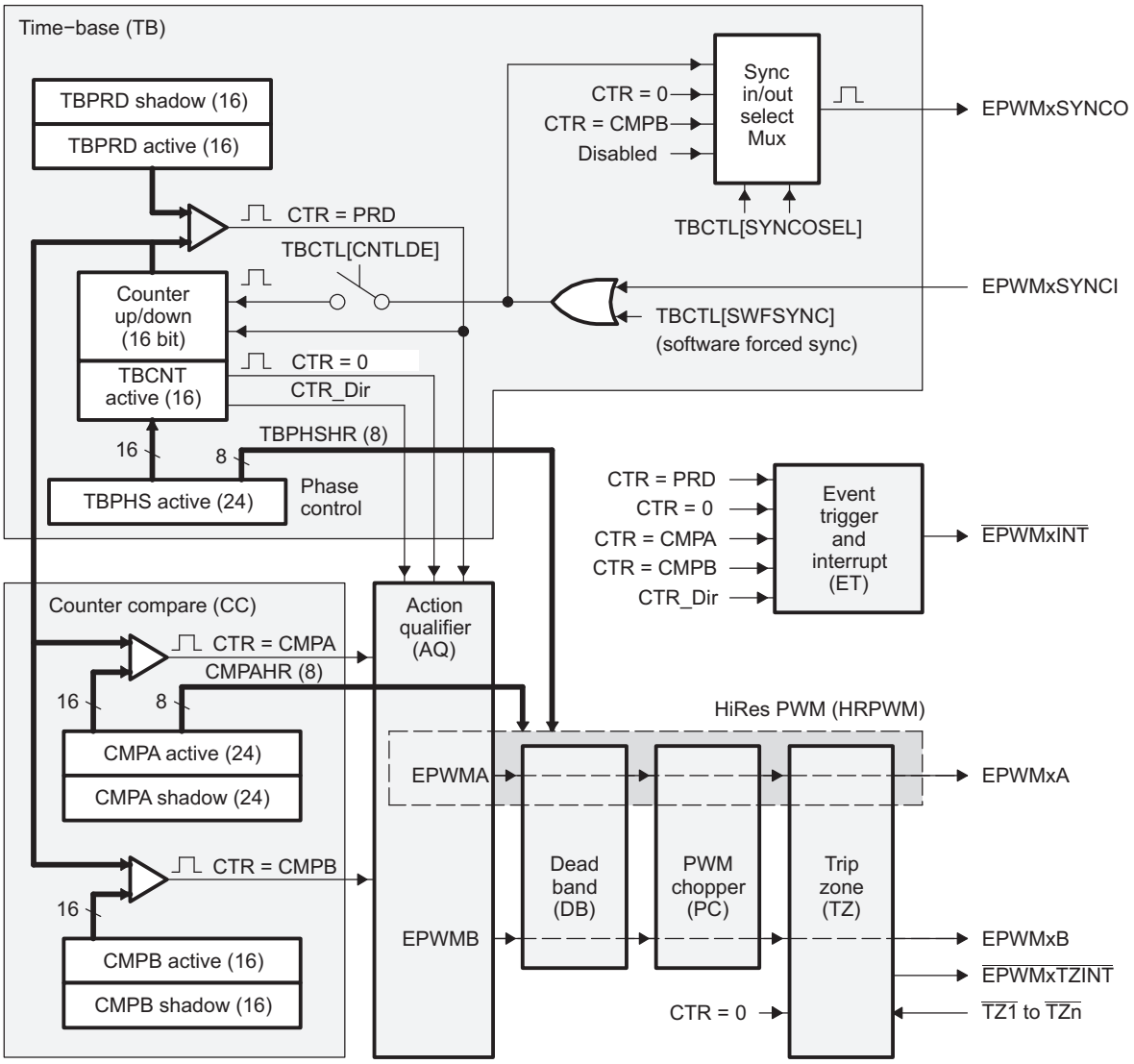
Figure 14-3 shows more internal details of a single ePWM module. The main signals used by the ePWM module are:

- **PWM output signals (EPWMxA and EPWMxB).** The PWM output signals are made available external to the device through the GPIO peripheral described in the system control and interrupts guide for your device.
- **Trip-zone signals ($\overline{\text{TZ1}}$ to $\overline{\text{TZn}}$).** These input signals alert the ePWM module of an external fault condition. Each module on a device can be configured to either use or ignore any of the trip-zone signals. The trip-zone signal can be configured as an asynchronous input through the GPIO peripheral. See your device-specific data manual to determine how many trip-zone pins are available in the device.
- **Time-base synchronization input (EPWMxSYNCI) and output (EPWMxSYNCO) signals.** The synchronization signals daisy chain the ePWM modules together. Each module can be configured to either use or ignore its synchronization input. The clock synchronization input and output signal are brought out to pins only for ePWM1 (ePWM module #1). The synchronization output for ePWM1 (EPWM1SYNCO) is also connected to the SYNCI of the first enhanced capture module (eCAP1).
- **Peripheral Bus.** The peripheral bus is 32-bits wide and allows both 16-bit and 32-bit writes to the ePWM register file.

Figure 14-3 also shows the key internal submodule interconnect signals. Each submodule is described in detail in Section 14.2.

Copyright © 2013–2016, Texas Instruments Incorporated

Figure 14-3. ePWM Submodules and Critical Internal Signal Interconnects



Copyright © 2013–2016, Texas Instruments Incorporated

14.1.3 Register Mapping

Table 14-1 shows the complete ePWM module control and status register set grouped by submodule. Each register set is duplicated for each instance of the ePWM module. The start address for each ePWM register file instance on a device is specified in the appropriate data manual.

Table 14-1. ePWM Module Control and Status Registers Grouped by Submodule

Acronym	Offset[1]	Size (×16)	Shadow	Register Description
Time-Base Submodule Registers				
TBCTL	0h	1	No	Time-Base Control Register
TBSTS	2h	1	No	Time-Base Status Register
TBPHSHR	4h	1	No	Extension for HRPWM Phase Register [2]
TBPHS	6h	1	No	Time-Base Phase Register
TBCNT	8h	1	No	Time-Base Counter Register
TBPRD	Ah	1	Yes	Time-Base Period Register
Counter-Compare Submodule Registers				
CMPCTL	Eh	1	No	Counter-Compare Control Register
CMPAHR	10h	1	No	Extension for HRPWM Counter-Compare A Register [2]
CMPA	12h	1	Yes	Counter-Compare A Register
CMPB	14h	1	Yes	Counter-Compare B Register
Action-Qualifier Submodule Registers				
AQCTLA	16h	1	No	Action-Qualifier Control Register for Output A (EPWMxA)
AQCTLB	18h	1	No	Action-Qualifier Control Register for Output B (EPWMxB)
AQSFRC	1Ah	1	No	Action-Qualifier Software Force Register
AQCSFRC	1Ch	1	Yes	Action-Qualifier Continuous S/W Force Register Set
Dead-Band Generator Submodule Registers				
DBCTL	1Eh	1	No	Dead-Band Generator Control Register
DBRED	20h	1	No	Dead-Band Generator Rising Edge Delay Count Register
DBFED	22h	1	No	Dead-Band Generator Falling Edge Delay Count Register
PWM-Chopper Submodule Registers				
PCCTL	3Ch	1	No	PWM-Chopper Control Register
Trip-Zone Submodule Registers				
TZSEL	24h	1	No	Trip-Zone Select Register
TZCTL	28h	1	No	Trip-Zone Control Register
TZEINT	2Ah	1	No	Trip-Zone Enable Interrupt Register
TZFLG	2Ch	1	No	Trip-Zone Flag Register
TZCLR	2Eh	1	No	Trip-Zone Clear Register
TZFRC	30h	1	No	Trip-Zone Force Register
Event-Trigger Submodule Registers				
ETSEL	32h	1	No	Event-Trigger Selection Register
ETPS	34h	1	No	Event-Trigger Pre-Scale Register
ETFLG	36h	1	No	Event-Trigger Flag Register
ETCLR	38h	1	No	Event-Trigger Clear Register
ETFRC	3Ah	1	No	Event-Trigger Force Register
High-Resolution PWM (HRPWM) Submodule Registers				
HRCNFG	1040h	1	No	HRPWM Configuration Register [2]

(1) Locations not shown are reserved.

(2) These registers are only available on ePWM instances that include the high-resolution PWM (HRPWM) extension; otherwise, these locations are reserved. See your device-specific data manual to determine which instances include the HRPWM.

Copyright © 2013–2016, Texas Instruments Incorporated

14.2 Architecture

Seven submodules are included in every ePWM peripheral. There are some instances that include a high-resolution submodule that allows more precise control of the PWM outputs. Each of these submodules performs specific tasks that can be configured by software.

14.2.1 Overview

Table 14-2 lists the eight key submodules together with a list of their main configuration parameters. For example, if you need to adjust or control the duty cycle of a PWM waveform, then you should see the counter-compare submodule in Section 14.2.4 for relevant details.

Table 14-2. Submodule Configuration Parameters

Submodule	Configuration Parameter or Option	Reference
Time-base (TB)	• Scale the time-base clock (TBCLK) relative to the system clock (SYSCLKOUT). • Configure the PWM time-base counter (TBCNT) frequency or period. • Set the mode for the time-base counter: – count-up mode: used for asymmetric PWM – count-down mode: used for asymmetric PWM – count-up-and-down mode: used for symmetric PWM • Configure the time-base phase relative to another ePWM module. • Synchronize the time-base counter between modules through hardware or software. • Configure the direction (up or down) of the time-base counter after a synchronization event. • Configure how the time-base counter will behave when the device is halted by an emulator. • Specify the source for the synchronization output of the ePWM module: – Synchronization input signal – Time-base counter equal to zero – Time-base counter equal to counter-compare B (CMPB) – No output synchronization signal generated.	Section 14.2.3
Counter-compare (CC)	• Specify the PWM duty cycle for output EPWMxA and/or output EPWMxB • Specify the time at which switching events occur on the EPWMxA or EPWMxB output	Section 14.2.4
Action-qualifier (AQ)	• Specify the type of action taken when a time-base or counter-compare submodule event occurs: – No action taken – Output EPWMxA and/or EPWMxB switched high – Output EPWMxA and/or EPWMxB switched low – Output EPWMxA and/or EPWMxB toggled • Force the PWM output state through software control • Configure and control the PWM dead-band through software	Section 14.2.5
Dead-band (DB)	• Control of traditional complementary dead-band relationship between upper and lower switches • Specify the output rising-edge-delay value • Specify the output falling-edge delay value • Bypass the dead-band module entirely. In this case the PWM waveform is passed through without modification.	Section 14.2.6
PWM-chopper (PC)	• Create a chopping (carrier) frequency. • Pulse width of the first pulse in the chopped pulse train. • Duty cycle of the second and subsequent pulses. • Bypass the PWM-chopper module entirely. In this case the PWM waveform is passed through without modification.	Section 14.2.7

Copyright © 2013–2016, Texas Instruments Incorporated

Table 14-2. Submodule Configuration Parameters (continued)

Submodule	Configuration Parameter or Option	Reference
Trip-zone (TZ)	• Configure the ePWM module to react to one, all, or none of the trip-zone pins. • Specify the tripping action taken when a fault occurs: – Force EPWMxA and/or EPWMxB high – Force EPWMxA and/or EPWMxB low – Force EPWMxA and/or EPWMxB to a high-impedance state – Configure EPWMxA and/or EPWMxB to ignore any trip condition. • Configure how often the ePWM will react to each trip-zone pin: – One-shot – Cycle-by-cycle • Enable the trip-zone to initiate an interrupt. • Bypass the trip-zone module entirely.	Section 14.2.8
Event-trigger (ET)	• Enable the ePWM events that will trigger an interrupt. • Specify the rate at which events cause triggers (every occurrence or every second or third occurrence) • Poll, set, or clear event flags	Section 14.2.9
High-Resolution PWM (HRPWM)	• Enable extended time resolution capabilities • Configure finer time granularity control or edge positioning	Section 14.2.10

Copyright © 2013–2016, Texas Instruments Incorporated

Code examples are provided in the remainder of this chapter that show how to implement various ePWM module configurations. These examples use the constant definitions shown in Example 14-1.

Example 14-1. Constant Definitions Used in the Code Examples

```
// TBCTL (Time-Base Control)
// = = = = = = = = = = = = = = = = = = = = = = = = = = = = =
// TBCNT MODE bit
#define         TB_COUNT_UP         0x0
#define         TB_COUNT_DOWN       0x1
#define         TB_COUNT_UPDOWN     0x2
#define         TB_FREEZE           0x3
// PHSEN bit
#define         TB_DISABLE          0x0
#define         TB_ENABLE           0x1
// PRDLD bit
#define         TB_SHADOW           0x0
#define         TB_IMMEDIATE        0x1
// SYNCOSEL bit
#define         TB_SYNC_IN          0x0
#define         TB_CTR_ZERO         0x1
#define         TB_CTR_CMPB         0x2
#define         TB_SYNC_DISABLE     0x3
// HSPCLKDIV and CLKDIV bits
#define         TB_DIV1             0x0
#define         TB_DIV2             0x1
#define         TB_DIV4             0x2
// PHSDIR bit
#define         TB_DOWN             0x0
#define         TB_UP               0x1
// CMPCTL (Compare Control)
// = = = = = = = = = = = = = = = = = = = = = = = = = = = = =
// LOADAMODE and LOADBMODE bits
#define         CC_CTR_ZERO         0x0
#define         CC_CTR_PRD          0x1
#define         CC_CTR_ZERO_PRD     0x2
#define         CC_LD_DISABLE       0x3
// SHDWAMODE and SHDWBMODE bits
#define         CC_SHADOW           0x0
#define         CC_IMMEDIATE        0x1
// AQCTLA and AQCTLB (Action-qualifier Control)
// = = = = = = = = = = = = = = = = = = = = = = = = = = = = =
// ZRO, PRD, CAU, CAD, CBU, CBD bits
#define         AQ_NO_ACTION        0x0
#define         AQ_CLEAR            0x1
#define         AQ_SET              0x2
#define         AQ_TOGGLE           0x3
// DBCTL (Dead-Band Control)
// = = = = = = = = = = = = = = = = = = = = = = = = = = = = =
// MODE bit
#define         DB_DISABLE          0x0
#define         DBA_ENABLE          0x1
#define         DBB_ENABLE          0x2
#define         DB_FULL_ENABLE      0x3
// POLSEL bit
#define         DB_ACTV_HI          0x0
#define         DB_ACTV_LOC         0x1
#define         DB_ACTV_HIC         0x2
#define         DB_ACTV_LO          0x3
// PCCTL (chopper control)
// = = = = = = = = = = = = = = = = = = = = = = = = = = = = =
// CHPEN bit
#define         CHP_DISABLE         0x0
#define         CHP_ENABLE          0x1
```


Copyright © 2013–2016, Texas Instruments Incorporated

Example 14-1. Constant Definitions Used in the Code Examples (continued)

```
// CHPFREQ bit
#define         CHP_DIV1           0x0
#define         CHP_DIV2           0x1
#define         CHP_DIV3           0x2
#define         CHP_DIV4           0x3
#define         CHP_DIV5           0x4
#define         CHP_DIV6           0x5
#define         CHP_DIV7           0x6
#define         CHP_DIV8           0x7
// CHPDUTY bit
#define         CHP1_8TH           0x0
#define         CHP2_8TH           0x1
#define         CHP3_8TH           0x2
#define         CHP4_8TH           0x3
#define         CHP5_8TH           0x4
#define         CHP6_8TH           0x5
#define         CHP7_8TH           0x6
// TZSEL (Trip-zone Select)
// = = = = = = = = = = = = = = = = = = = = = = = = = = = = =
// CBCn and OSHTn bits
#define         TZ_DISABLE         0x0
#define         TZ_ENABLE          0x1
// TZCTL (Trip-zone Control)
// = = = = = = = = = = = = = = = = = = = = = = = = = = = = =
// TZA and TZB bits
#define         TZ_HIZ             0x0
#define         TZ_FORCE_HI        0x1
#define         TZ_FORCE_LO        0x2
#define         TZ_NONE            0x3
// ETSEL (Event-trigger Select)
// = = = = = = = = = = = = = = = = = = = = = = = = = = = = =
// INTSEL bit
#define         ET_CTR_ZERO        0x1
#define         ET_CTR_PRD         0x2
#define         ET_CTRU_CMPA       0x4
#define         ET_CTRD_CMPA       0x5
#define         ET_CTRU_CMPB       0x6
#define         ET_CTRD_CMPB       0x7
// ETPS (Event-trigger Prescale)
// = = = = = = = = = = = = = = = = = = = = = = = = = = = = =
// INTPRD bit
#define         ET_DISABLE         0x0
#define         ET_1ST             0x1
#define         ET_2ND             0x2
#define         ET_3RD             0x3
```

14.2.2 Proper Interrupt Initialization Procedure

When the ePWM peripheral clock is enabled it may be possible that interrupt flags may be set due to spurious events due to the ePWM registers not being properly initialized. The proper procedure for initializing the ePWM peripheral is:

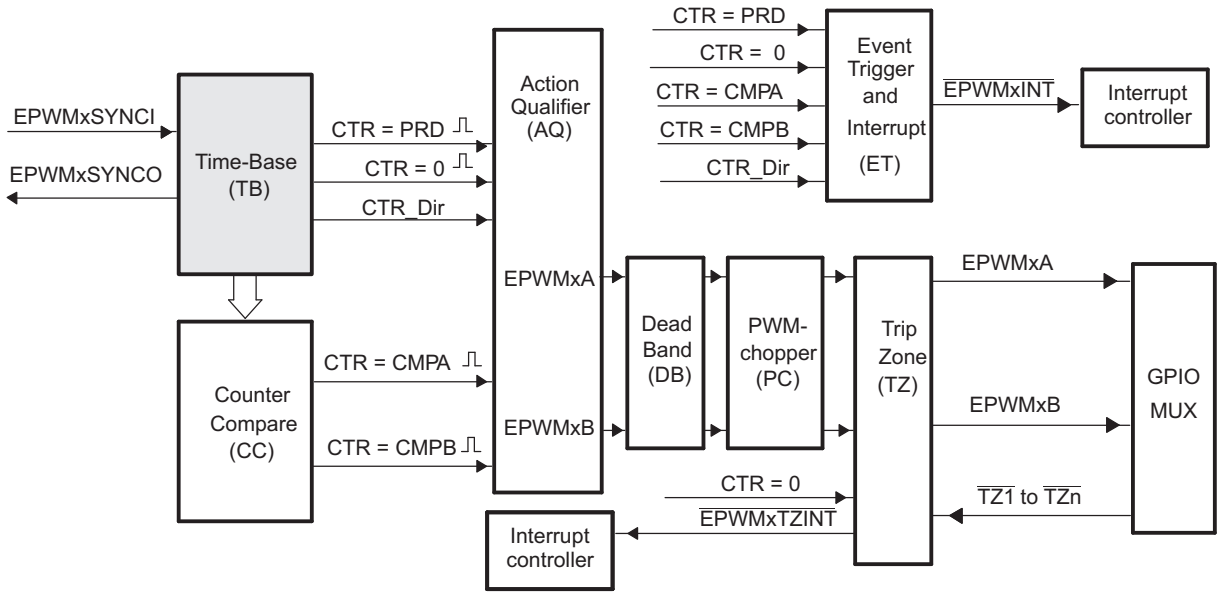
1. Disable global interrupts (CPU INTM flag)
2. Disable ePWM interrupts
3. Initialize peripheral registers
4. Clear any spurious ePWM flags
5. Enable ePWM interrupts
6. Enable global interrupts

Copyright © 2013–2016, Texas Instruments Incorporated

14.2.3 Time-Base (TB) Submodule

Each ePWM module has its own time-base submodule that determines all of the event timing for the ePWM module. Built-in synchronization logic allows the time-base of multiple ePWM modules to work together as a single system. Figure 14-4 illustrates the time-base module's place within the ePWM.

Figure 14-4. Time-Base Submodule Block Diagram



14.2.3.1 Purpose of the Time-Base Submodule

You can configure the time-base submodule for the following:

- Specify the ePWM time-base counter (TBCNT) frequency or period to control how often events occur.
- Manage time-base synchronization with other ePWM modules.
- Maintain a phase relationship with other ePWM modules.
- Set the time-base counter to count-up, count-down, or count-up-and-down mode.
- Generate the following events:
  - CTR = PRD: Time-base counter equal to the specified period (TBCNT = TBPRD) .
  - CTR = 0: Time-base counter equal to zero (TBCNT = 0000h).
- Configure the rate of the time-base clock; a prescaled version of the CPU system clock (SYSCLKOUT). This allows the time-base counter to increment/decrement at a slower rate.

Copyright © 2013–2016, Texas Instruments Incorporated

14.2.3.2 Controlling and Monitoring the Time-Base Submodule

Table 14-3 lists the registers used to control and monitor the time-base submodule.

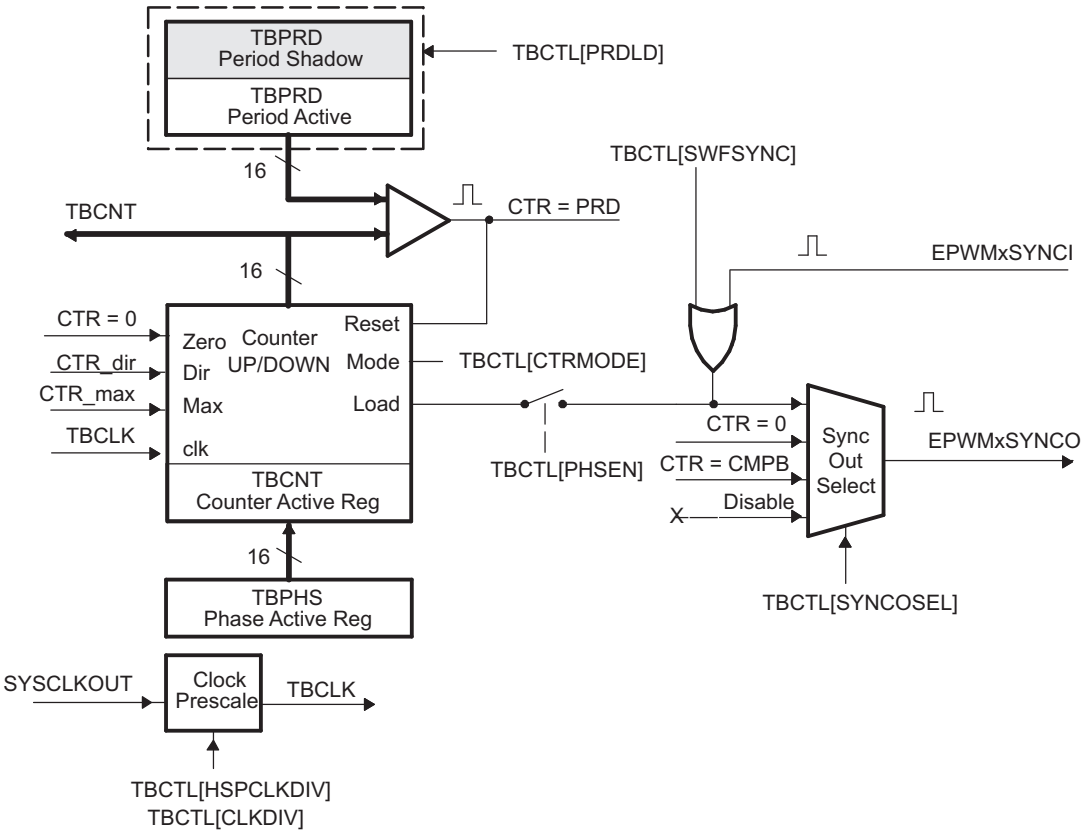
Table 14-3. Time-Base Submodule Registers

Acronym	Register Description	Address Offset	Shadowed
TBCTL	Time-Base Control Register	0h	No
TBSTS	Time-Base Status Register	2h	No
TBPHSHR	HRPWM extension Phase Register (1)	4h	No
TBPHS	Time-Base Phase Register	6h	No
TBCNT	Time-Base Counter Register	8h	No
TBPRD	Time-Base Period Register	Ah	Yes

(1) This register is available only on ePWM instances that include the high-resolution extension (HRPWM). On ePWM modules that do not include the HRPWM, this location is reserved. See your device-specific data manual to determine which ePWM instances include this feature.

Figure 14-5 shows the critical signals and registers of the time-base submodule. Table 14-4 provides descriptions of the key signals associated with the time-base submodule.

Figure 14-5. Time-Base Submodule Signals and Registers



Copyright © 2013–2016, Texas Instruments Incorporated

Table 14-4. Key Time-Base Signals

Signal	Description
EPWMxSYNCI	Time-base synchronization input.
	Input pulse used to synchronize the time-base counter with the counter of ePWM module earlier in the synchronization chain. An ePWM peripheral can be configured to use or ignore this signal. For the first ePWM module (EPWM1) this signal comes from a device pin. For subsequent ePWM modules this signal is passed from another ePWM peripheral. For example, EPWM2SYNCI is generated by the ePWM1 peripheral, EPWM3SYNCI is generated by ePWM2 and so forth. See Section 14.2.3.3.2 for information on the synchronization order of a particular device.
EPWMxSYNCO	Time-base synchronization output.
	This output pulse is used to synchronize the counter of an ePWM module later in the synchronization chain. The ePWM module generates this signal from one of three event sources: 1. EPWMxSYNCI (Synchronization input pulse) 2. CTR = 0: The time-base counter equal to zero (TBCNT = 0000h). 3. CTR = CMPB: The time-base counter equal to the counter-compare B (TBCNT = CMPB) register.
CTR = PRD	Time-base counter equal to the specified period.
	This signal is generated whenever the counter value is equal to the active period register value. That is when TBCNT = TBPRD.
CTR = 0	Time-base counter equal to zero.
	This signal is generated whenever the counter value is zero. That is when TBCNT equals 0000h.
CTR = CMPB	Time-base counter equal to active counter-compare B register (TBCNT = CMPB).
	This event is generated by the counter-compare submodule and used by the synchronization out logic.
CTR_dir	Time-base counter direction.
	Indicates the current direction of the ePWM's time-base counter. This signal is high when the counter is increasing and low when it is decreasing.
CTR_max	Time-base counter equal max value. (TBCNT = FFFFh)
	Generated event when the TBCNT value reaches its maximum value. This signal is only used only as a status bit.
TBCLK	Time-base clock.
	This is a prescaled version of the system clock (SYSCLKOUT) and is used by all submodules within the ePWM. This clock determines the rate at which time-base counter increments or decrements.

14.2.3.3 Calculating PWM Period and Frequency

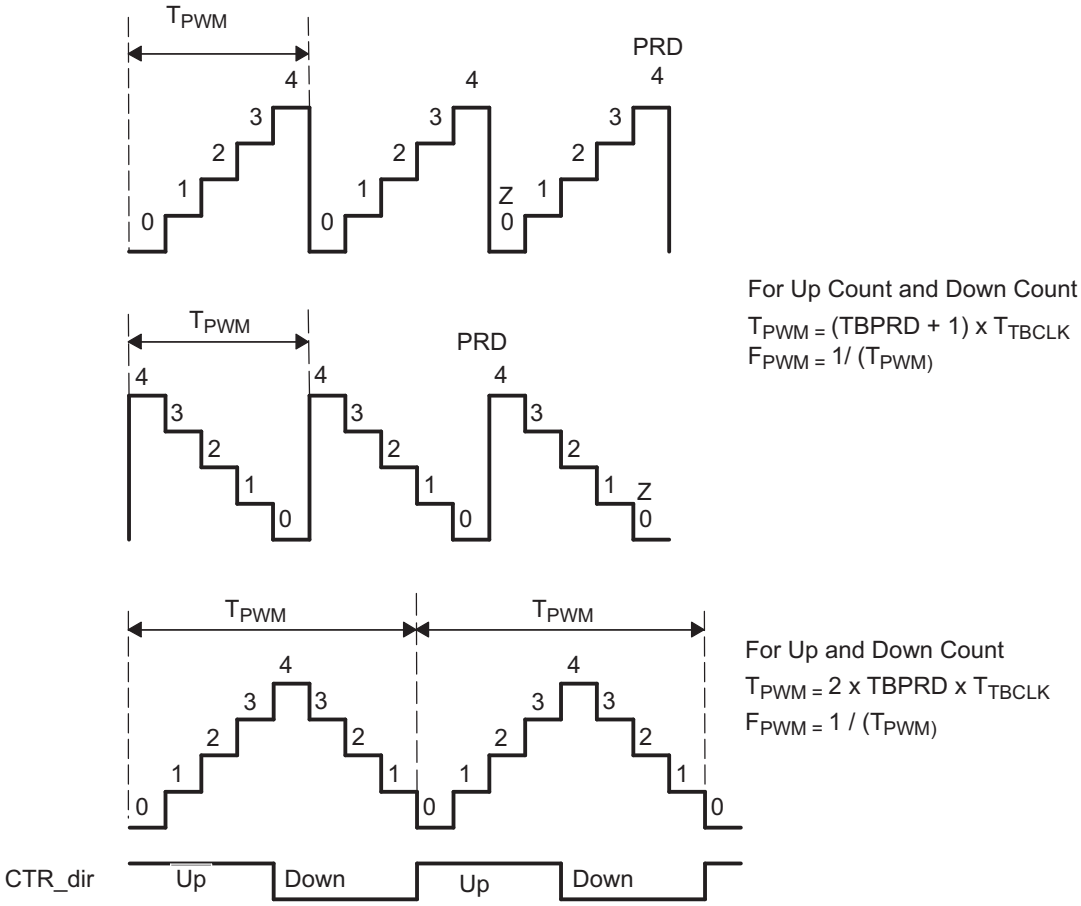
The frequency of PWM events is controlled by the time-base period (TBPRD) register and the mode of the time-base counter. Figure 14-6 shows the period (T_{pwm}) and frequency (F_{pwm}) relationships for the up-count, down-count, and up-down-count time-base counter modes when when the period is set to 4 (TBPRD = 4). The time increment for each step is defined by the time-base clock (TBCLK) which is a prescaled version of the system clock (SYSCLKOUT).

The time-base counter has three modes of operation selected by the time-base control register (TBCTL):

- **Up-Down-Count Mode:** In up-down-count mode, the time-base counter starts from zero and increments until the period (TBPRD) value is reached. When the period value is reached, the time-base counter then decrements until it reaches zero. At this point the counter repeats the pattern and begins to increment.
- **Up-Count Mode:** In this mode, the time-base counter starts from zero and increments until it reaches the value in the period register (TBPRD). When the period value is reached, the time-base counter resets to zero and begins to increment once again.
- **Down-Count Mode:** In down-count mode, the time-base counter starts from the period (TBPRD) value and decrements until it reaches zero. When it reaches zero, the time-base counter is reset to the period value and it begins to decrement once again.

Copyright © 2013–2016, Texas Instruments Incorporated

Figure 14-6. Time-Base Frequency and Period



14.2.3.3.1 Time-Base Period Shadow Register

The time-base period register (TBPRD) has a shadow register. Shadowing allows the register update to be synchronized with the hardware. The following definitions are used to describe all shadow registers in the ePWM module:

- **Active Register:** The active register controls the hardware and is responsible for actions that the hardware causes or invokes.
- **Shadow Register:** The shadow register buffers or provides a temporary holding location for the active register. It has no direct effect on any control hardware. At a strategic point in time the shadow register's content is transferred to the active register. This prevents corruption or spurious operation due to the register being asynchronously modified by software.

The memory address of the shadow period register is the same as the active register. Which register is written to or read from is determined by the TBCTL[PRDLD] bit. This bit enables and disables the TBPRD shadow register as follows:

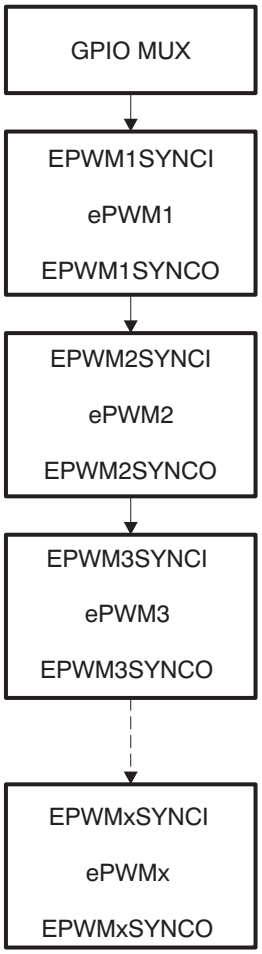
- **Time-Base Period Shadow Mode:** The TBPRD shadow register is enabled when TBCTL[PRDLD] = 0. Reads from and writes to the TBPRD memory address go to the shadow register. The shadow register contents are transferred to the active register (TBPRD (Active) ← TBPRD (shadow)) when the time-base counter equals zero (TBCNT = 0000h). By default the TBPRD shadow register is enabled.
- **Time-Base Period Immediate Load Mode:** If immediate load mode is selected (TBCTL[PRDLD] = 1), then a read from or a write to the TBPRD memory address goes directly to the active register.

Copyright © 2013–2016, Texas Instruments Incorporated

14.2.3.3.2 Time-Base Counter Synchronization

A time-base synchronization scheme connects all of the ePWM modules on a device. Each ePWM module has a synchronization input (EPWMxSYNCI) and a synchronization output (EPWMxSYNCO). The input synchronization for the first instance (ePWM1) comes from an external pin. The possible synchronization connections for the remaining ePWM modules is shown in Figure 14-7.

Figure 14-7. Time-Base Counter Synchronization Scheme 1



Copyright © 2013–2016, Texas Instruments Incorporated

Each ePWM module can be configured to use or ignore the synchronization input. If the TBCTL[PHSEN] bit is set, then the time-base counter (TBCNT) of the ePWM module will be automatically loaded with the phase register (TBPHS) contents when one of the following conditions occur:

- **EPWMxSYNCI: Synchronization Input Pulse:** The value of the phase register is loaded into the counter register when an input synchronization pulse is detected (TBPHS → TBCNT). This operation occurs on the next valid time-base clock (TBCLK) edge.
- **Software Forced Synchronization Pulse:** Writing a 1 to the TBCTL[SWFSYNC] control bit invokes a software forced synchronization. This pulse is ORed with the synchronization input signal, and therefore has the same effect as a pulse on EPWMxSYNCI.

This feature enables the ePWM module to be automatically synchronized to the time base of another ePWM module. Lead or lag phase control can be added to the waveforms generated by different ePWM modules to synchronize them. In up-down-count mode, the TBCTL[PSHDIR] bit configures the direction of the time-base counter immediately after a synchronization event. The new direction is independent of the direction prior to the synchronization event. The TBPHS bit is ignored in count-up or count-down modes. See Figure 14-8 through Figure 14-11 for examples.

Clearing the TBCTL[PHSEN] bit configures the ePWM to ignore the synchronization input pulse. The synchronization pulse can still be allowed to flow-through to the EPWMxSYNCO and be used to synchronize other ePWM modules. In this way, you can set up a master time-base (for example, ePWM1) and downstream modules (ePWM2 - ePWMx) may elect to run in synchronization with the master.

14.2.3.4 Phase Locking the Time-Base Clocks of Multiple ePWM Modules

The TBCLKSYNC bit in the chip configuration register 1 (CFGCHIP1) in the System Module can be used to globally synchronize the time-base clocks of all enabled ePWM modules on a device. The TBCLKSYNC bit is part of the chip configuration registers and is described in the device-specific data manual. When TBCLKSYNC = 0, the time-base clock of all ePWM modules is stopped (default). When TBCLKSYNC = 1, all ePWM time-base clocks are started with the rising edge of TBCLK aligned. For perfectly synchronized TBCLKs, the prescaler bits in the TBCTL register of each ePWM module must be set identically. The proper procedure for enabling the ePWM clocks is as follows:

1. Enable the ePWM module clocks.
2. Set TBCLKSYNC = 0. This will stop the time-base clock within any enabled ePWM module.
3. Configure the prescaler values and desired ePWM modes.
4. Set TBCLKSYNC = 1.

14.2.3.5 Time-Base Counter Modes and Timing Waveforms

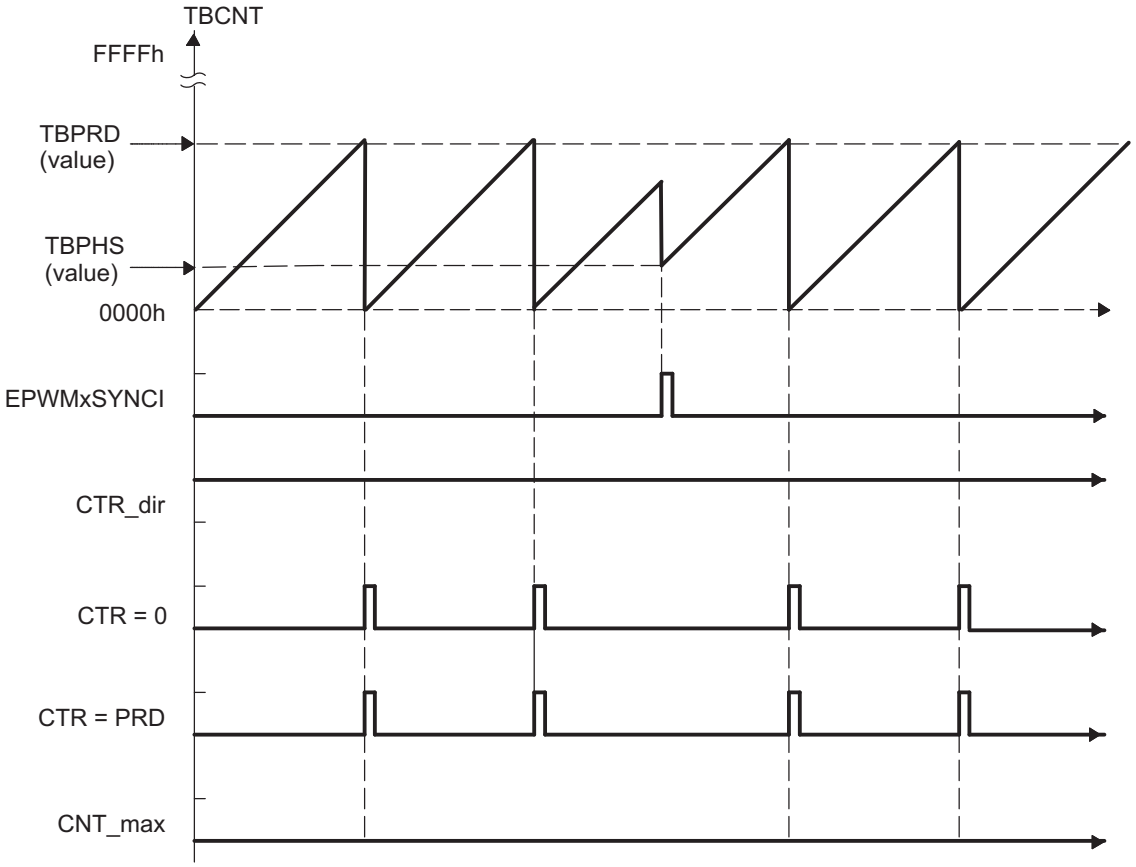
The time-base counter operates in one of four modes:

- Up-count mode which is asymmetrical.
- Down-count mode which is asymmetrical.
- Up-down-count which is symmetrical.
- Frozen where the time-base counter is held constant at the current value.

To illustrate the operation of the first three modes, Figure 14-8 to Figure 14-11 show when events are generated and how the time-base responds to an EPWMxSYNCI signal.

Copyright © 2013–2016, Texas Instruments Incorporated

Figure 14-8. Time-Base Up-Count Mode Waveforms



Copyright © 2013–2016, Texas Instruments Incorporated

Figure 14-9. Time-Base Down-Count Mode Waveforms

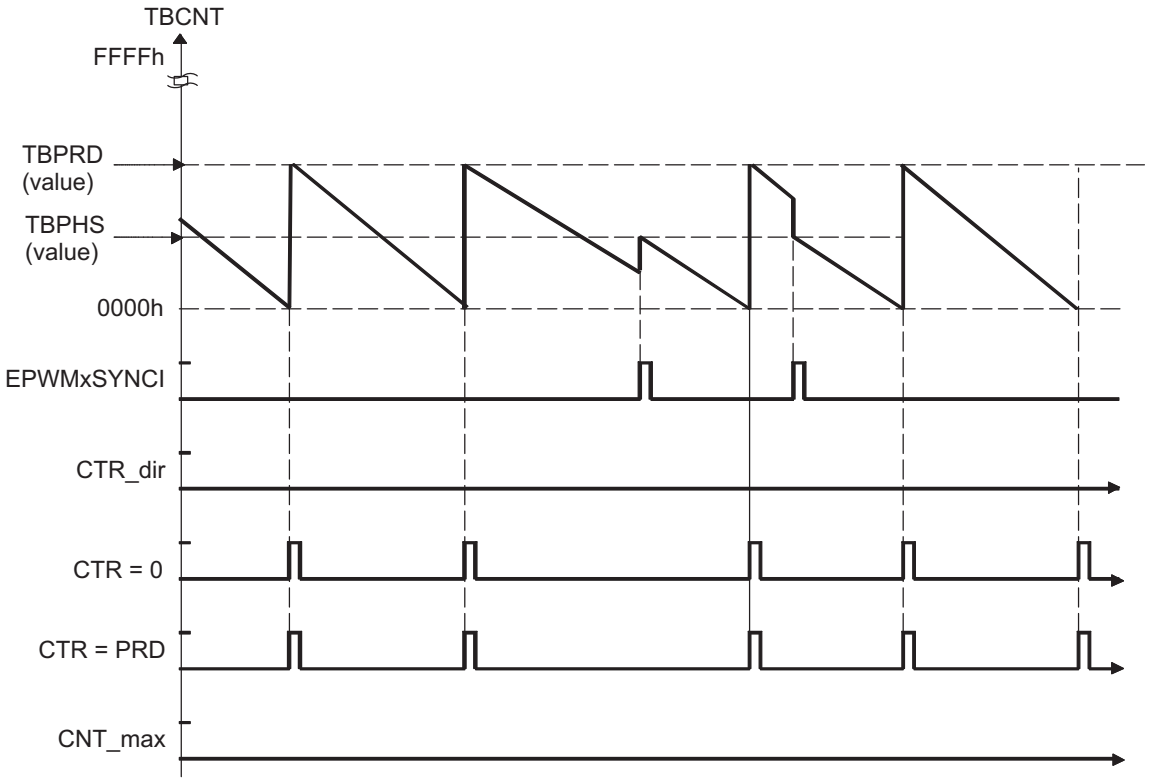
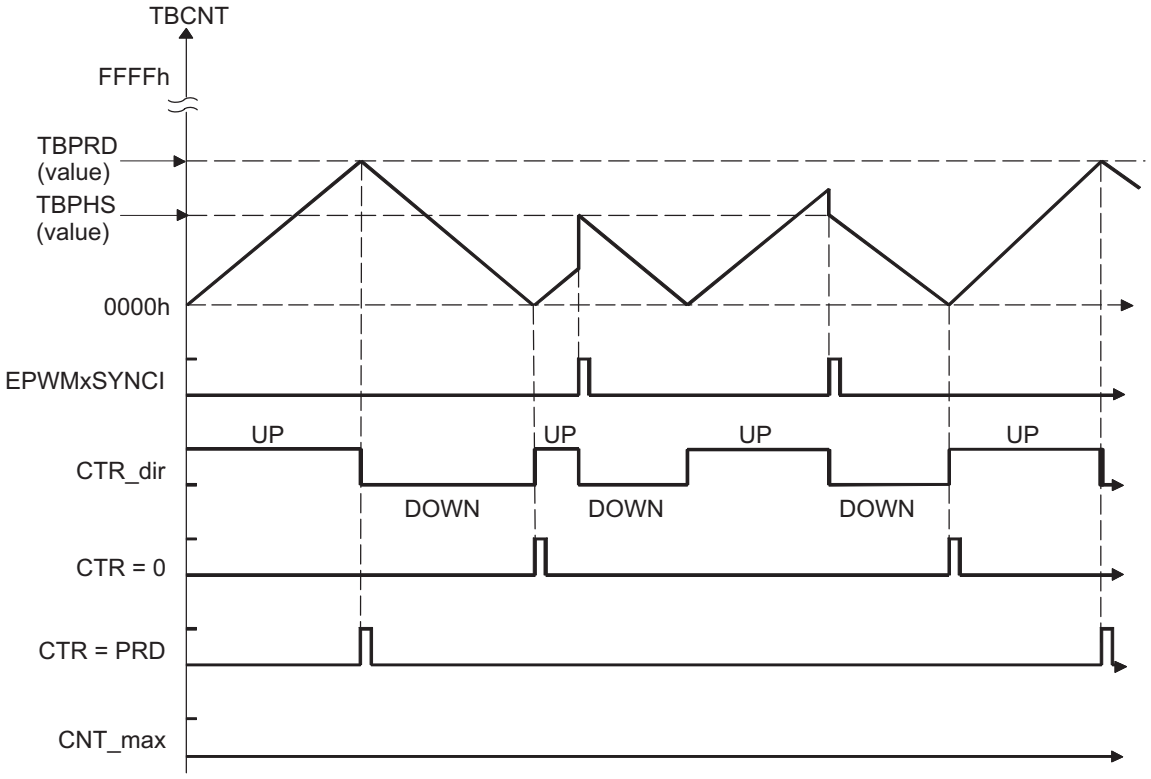
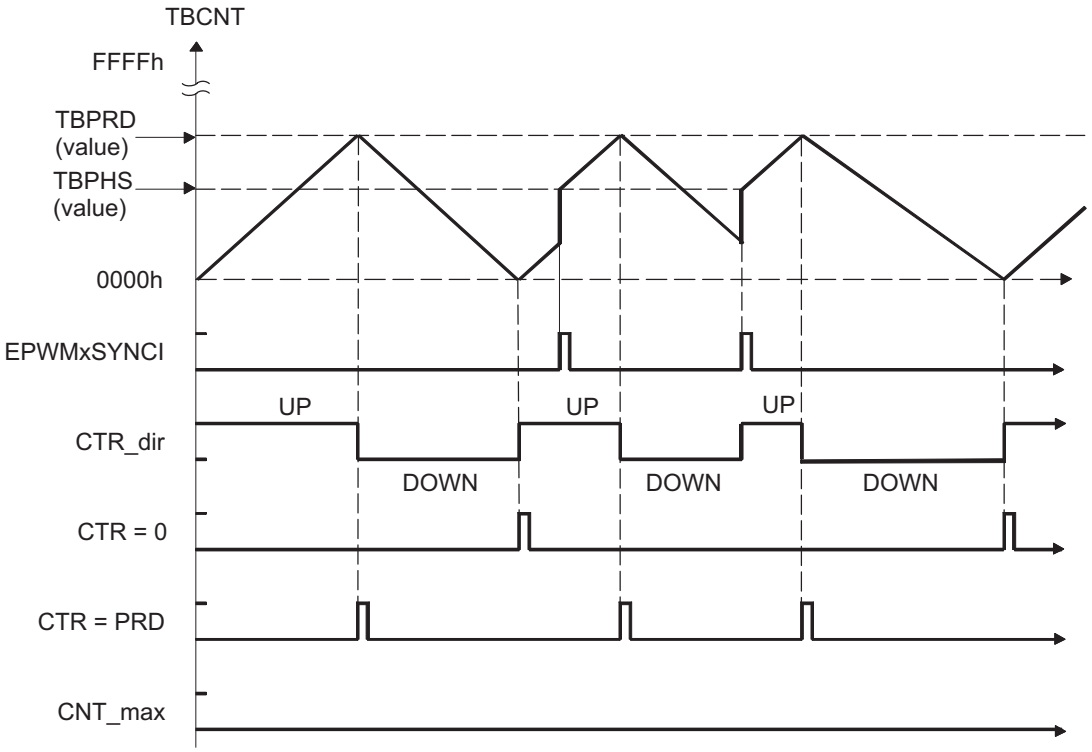


Figure 14-10. Time-Base Up-Down-Count Waveforms, TBCTL[PHSDIR = 0] Count Down on Synchronization Event



Copyright © 2013–2016, Texas Instruments Incorporated

Figure 14-11. Time-Base Up-Down Count Waveforms, TBCTL[PHSDIR = 1] Count Up on Synchronization Event



Copyright © 2013–2016, Texas Instruments Incorporated

14.2.4 Counter-Compare (CC) Submodule

Figure 14-12 illustrates the counter-compare submodule within the ePWM. Figure 14-13 shows the basic structure of the counter-compare submodule.

Figure 14-12. Counter-Compare Submodule

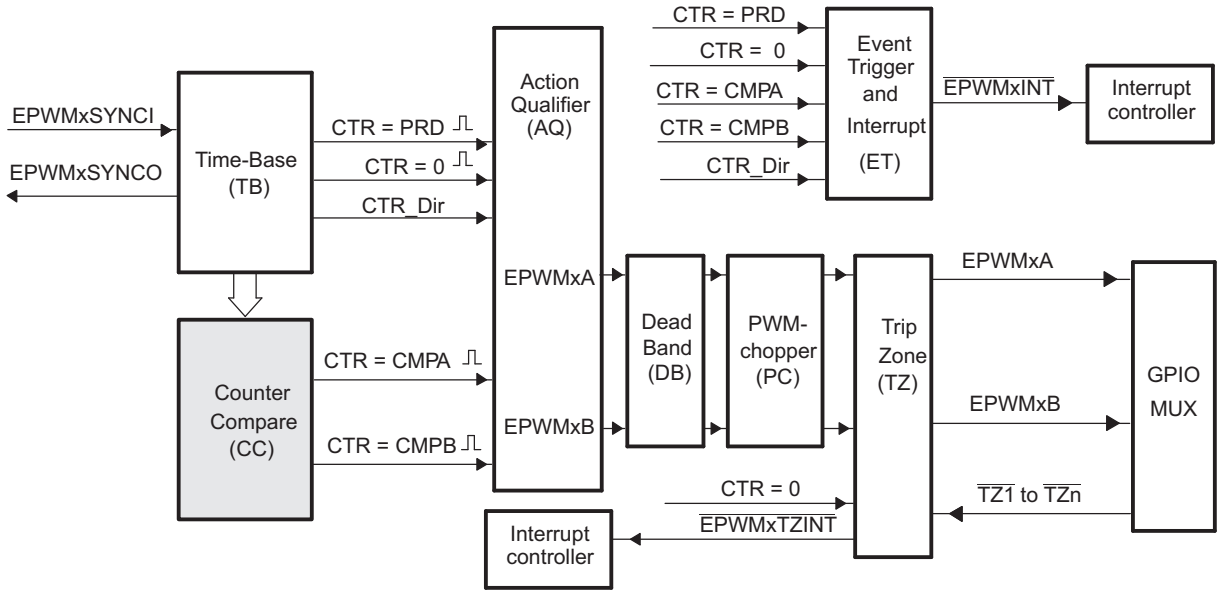
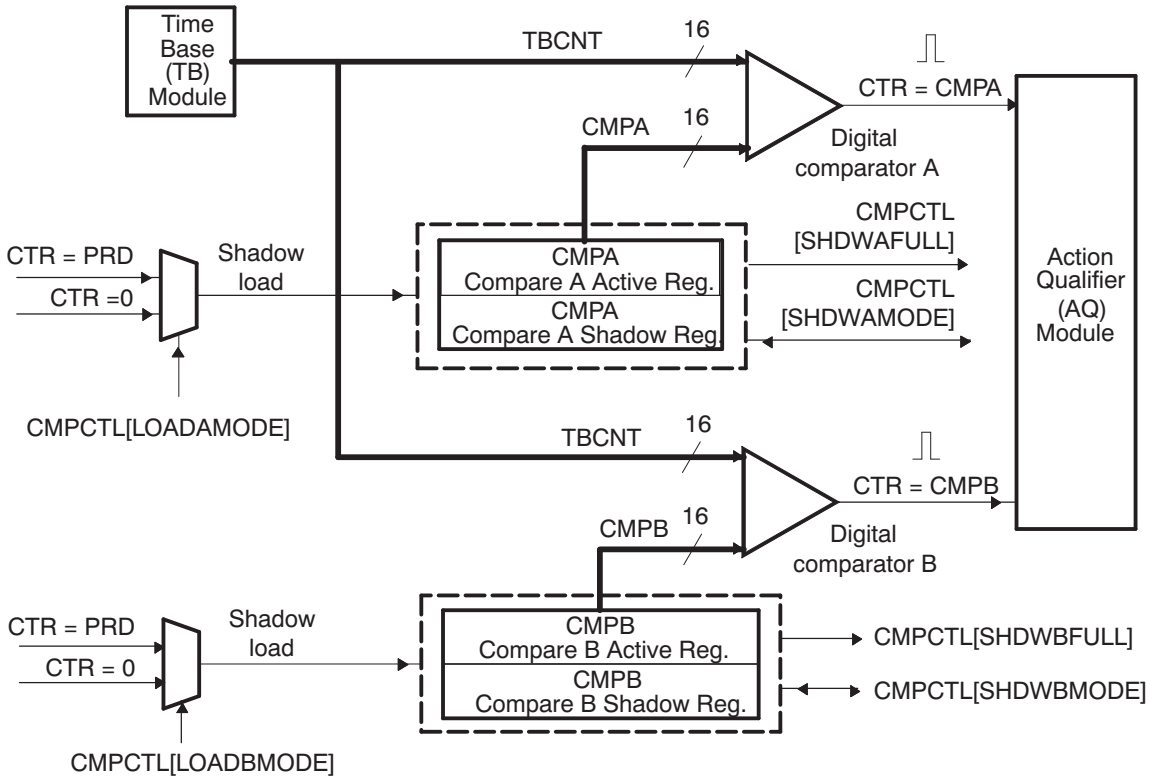


Figure 14-13. Counter-Compare Submodule Signals and Registers



Copyright © 2013–2016, Texas Instruments Incorporated

14.2.4.1 Purpose of the Counter-Compare Submodule

The counter-compare submodule takes as input the time-base counter value. This value is continuously compared to the counter-compare A (CMPA) and counter-compare B (CMPB) registers. When the time-base counter is equal to one of the compare registers, the counter-compare unit generates an appropriate event.

The counter-compare submodule:

- Generates events based on programmable time stamps using the CMPA and CMPB registers
  - CTR = CMPA: Time-base counter equals counter-compare A register (TBCNT = CMPA).
  - CTR = CMPB: Time-base counter equals counter-compare B register (TBCNT = CMPB)
- Controls the PWM duty cycle if the action-qualifier submodule is configured appropriately
- Shadows new compare values to prevent corruption or glitches during the active PWM cycle

14.2.4.2 Controlling and Monitoring the Counter-Compare Submodule

Table 14-5 lists the registers used to control and monitor the counter-compare submodule. Table 14-6 lists the key signals associated with the counter-compare submodule.

Table 14-5. Counter-Compare Submodule Registers

Acronym	Register Description	Address Offset	Shadowed
CMPCTL	Counter-Compare Control Register.	Eh	No
CMPAHR	HRPWM Counter-Compare A Extension Register (1)	10h	Yes
CMPA	Counter-Compare A Register	12h	Yes
CMPB	Counter-Compare B Register	14h	Yes

(1) This register is available only on ePWM modules with the high-resolution extension (HRPWM). On ePWM modules that do not include the HRPWM, this location is reserved. Refer to the device-specific data manual to determine which ePWM instances include this feature.

Table 14-6. Counter-Compare Submodule Key Signals

Signal	Description of Event	Registers Compared
CTR = CMPA	Time-base counter equal to the active counter-compare A value	TBCNT = CMPA
CTR = CMPB	Time-base counter equal to the active counter-compare B value	TBCNT = CMPB
CTR = PRD	Time-base counter equal to the active period. Used to load active counter-compare A and B registers from the shadow register	TBCNT = TBPRD
CTR = 0	Time-base counter equal to zero. Used to load active counter-compare A and B registers from the shadow register	TBCNT = 0000h

Copyright © 2013–2016, Texas Instruments Incorporated

14.2.4.3 Operational Highlights for the Counter-Compare Submodule

The counter-compare submodule is responsible for generating two independent compare events based on two compare registers:

1. CTR = CMPA: Time-base counter equal to counter-compare A register (TBCNT = CMPA).
2. CTR = CMPB: Time-base counter equal to counter-compare B register (TBCNT = CMPB).

For up-count or down-count mode, each event occurs only once per cycle. For up-down-count mode each event occurs twice per cycle, if the compare value is between 0000h and TBPRD; and occurs once per cycle, if the compare value is equal to 0000h or equal to TBPRD. These events are fed into the action-qualifier submodule where they are qualified by the counter direction and converted into actions if enabled. Refer to Section 14.2.5.1 for more details.

The counter-compare registers CMPA and CMPB each have an associated shadow register. Shadowing provides a way to keep updates to the registers synchronized with the hardware. When shadowing is used, updates to the active registers only occurs at strategic points. This prevents corruption or spurious operation due to the register being asynchronously modified by software. The memory address of the active register and the shadow register is identical. Which register is written to or read from is determined by the CMPCTL[SHDWAMODE] and CMPCTL[SHDWBMODE] bits. These bits enable and disable the CMPA shadow register and CMPB shadow register respectively. The behavior of the two load modes is described below:

- **Shadow Mode:** The shadow mode for the CMPA is enabled by clearing the CMPCTL[SHDWAMODE] bit and the shadow register for CMPB is enabled by clearing the CMPCTL[SHDWBMODE] bit. Shadow mode is enabled by default for both CMPA and CMPB.

  If the shadow register is enabled then the content of the shadow register is transferred to the active register on one of the following events:

  – CTR = PRD: Time-base counter equal to the period (TBCNT = TBPRD).
  – CTR = 0: Time-base counter equal to zero (TBCNT = 0000h)
  – Both CTR = PRD and CTR = 0

  Which of these three events is specified by the CMPCTL[LOADAMODE] and CMPCTL[LOADBMODE] register bits. Only the active register contents are used by the counter-compare submodule to generate events to be sent to the action-qualifier.
- **Immediate Load Mode:** If immediate load mode is selected (TBCTL[SHADWAMODE] = 1 or TBCTL[SHADWBMODE] = 1), then a read from or a write to the register will go directly to the active register.

14.2.4.4 Count Mode Timing Waveforms

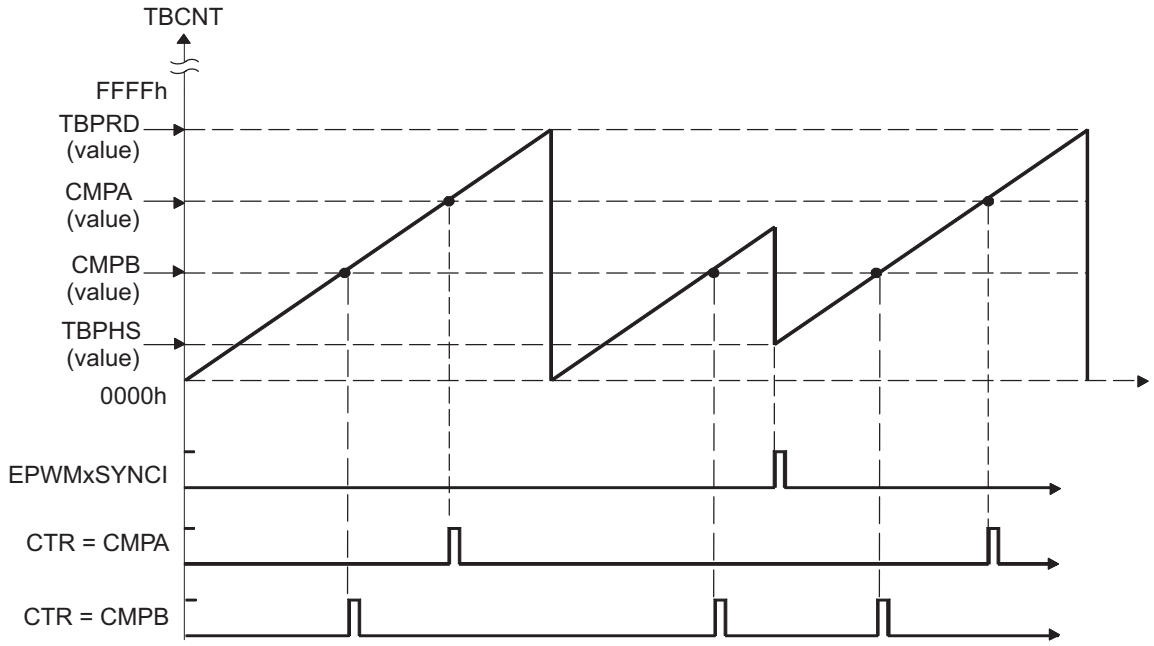
The counter-compare module can generate compare events in all three count modes:

- Up-count mode: used to generate an asymmetrical PWM waveform.
- Down-count mode: used to generate an asymmetrical PWM waveform.
- Up-down-count mode: used to generate a symmetrical PWM waveform.

To best illustrate the operation of the first three modes, the timing diagrams in Figure 14-14 to Figure 14-17 show when events are generated and how the EPWMxSYNCI signal interacts.

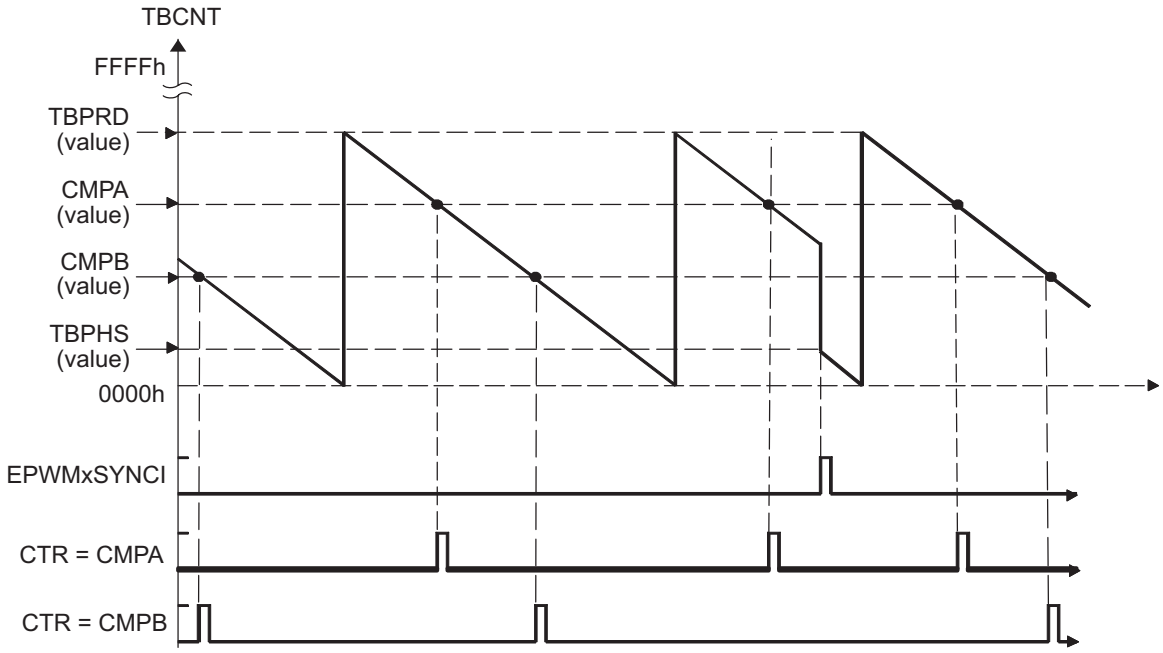
Copyright © 2013–2016, Texas Instruments Incorporated

Figure 14-14. Counter-Compare Event Waveforms in Up-Count Mode



NOTE: An EPWMxSYNCI external synchronization event can cause a discontinuity in the TBCNT count sequence. This can lead to a compare event being skipped. This skipping is considered normal operation and must be taken into account.

Figure 14-15. Counter-Compare Events in Down-Count Mode



Copyright © 2013–2016, Texas Instruments Incorporated

Figure 14-16. Counter-Compare Events in Up-Down-Count Mode, TBCTL[PHSDIR = 0] Count Down on Synchronization Event

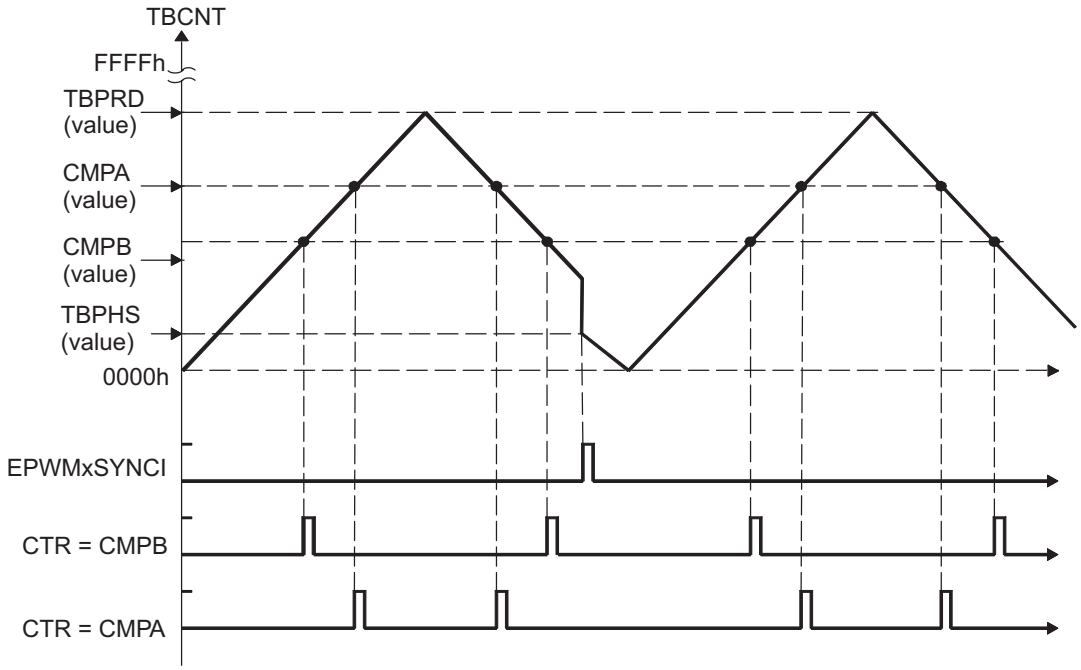
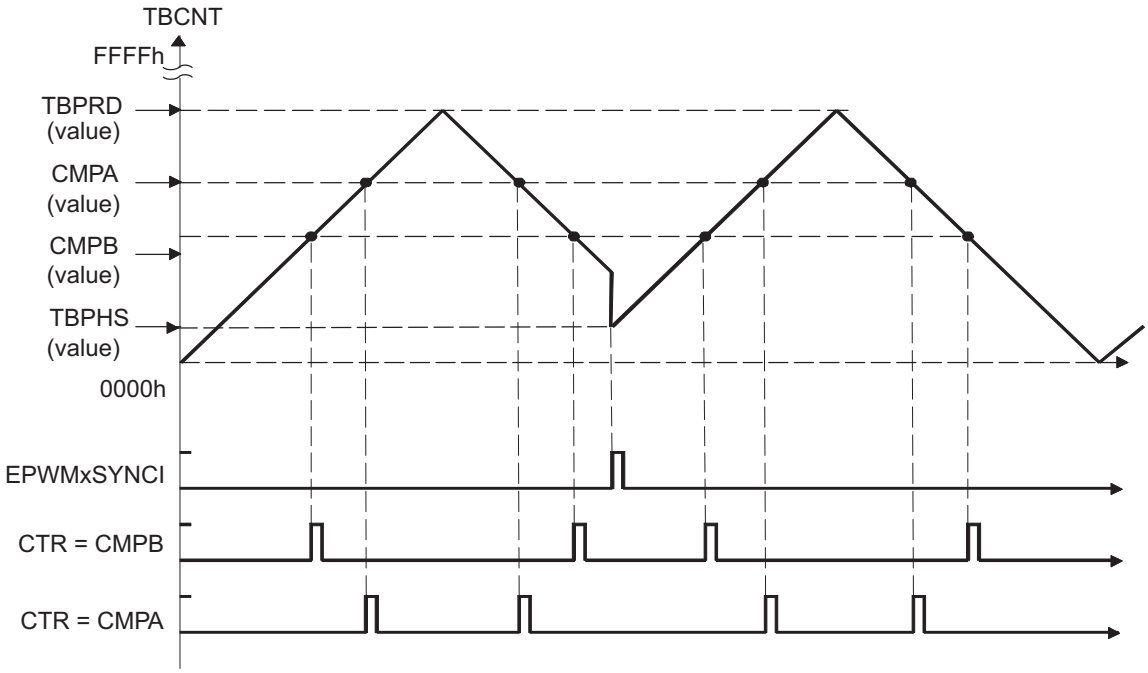


Figure 14-17. Counter-Compare Events in Up-Down-Count Mode, TBCTL[PHSDIR = 1] Count Up on Synchronization Event

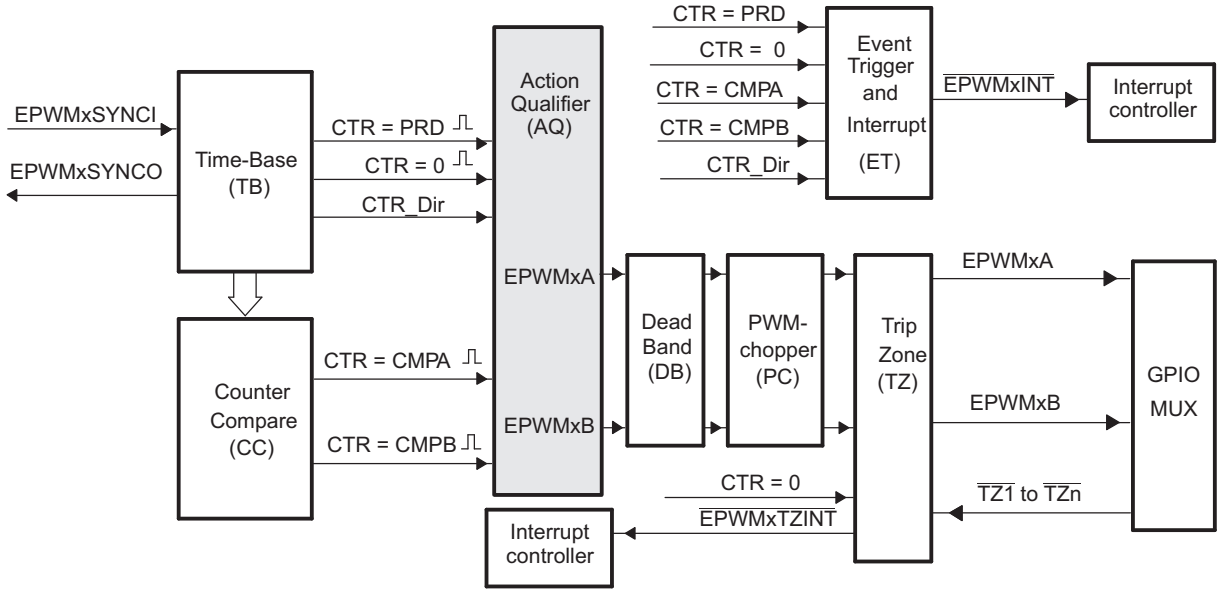


Copyright © 2013–2016, Texas Instruments Incorporated

14.2.5 Action-Qualifier (AQ) Submodule

Figure 14-18 shows the action-qualifier (AQ) submodule (see shaded block) in the ePWM system. The action-qualifier submodule has the most important role in waveform construction and PWM generation. It decides which events are converted into various action types, thereby producing the required switched waveforms at the EPWMxA and EPWMxB outputs.

Figure 14-18. Action-Qualifier Submodule



14.2.5.1 Purpose of the Action-Qualifier Submodule

The action-qualifier submodule is responsible for the following:

- Qualifying and generating actions (set, clear, toggle) based on the following events:
  - CTR = PRD: Time-base counter equal to the period (TBCNT = TBPRD)
  - CTR = 0: Time-base counter equal to zero (TBCNT = 0000h)
  - CTR = CMPA: Time-base counter equal to the counter-compare A register (TBCNT = CMPA)
  - CTR = CMPB: Time-base counter equal to the counter-compare B register (TBCNT = CMPB)
- Managing priority when these events occur concurrently
- Providing independent control of events when the time-base counter is increasing and when it is decreasing.

14.2.5.2 Controlling and Monitoring the Action-Qualifier Submodule

Table 14-7 lists the registers used to control and monitor the action-qualifier submodule.

Table 14-7. Action-Qualifier Submodule Registers

Acronym	Register Description	Address Offset	Shadowed
AQCTLA	Action-Qualifier Control Register For Output A (EPWMxA)	16h	No
AQCTLB	Action-Qualifier Control Register For Output B (EPWMxB)	18h	No
AQSFRC	Action-Qualifier Software Force Register	1Ah	No
AQCSFRC	Action-Qualifier Continuous Software Force	1Ch	Yes

Copyright © 2013–2016, Texas Instruments Incorporated

The action-qualifier submodule is based on event-driven logic. It can be thought of as a programmable cross switch with events at the input and actions at the output, all of which are software controlled via the set of registers shown in Figure 14-19. The possible input events are summarized again in Table 14-8.

Figure 14-19. Action-Qualifier Submodule Inputs and Outputs

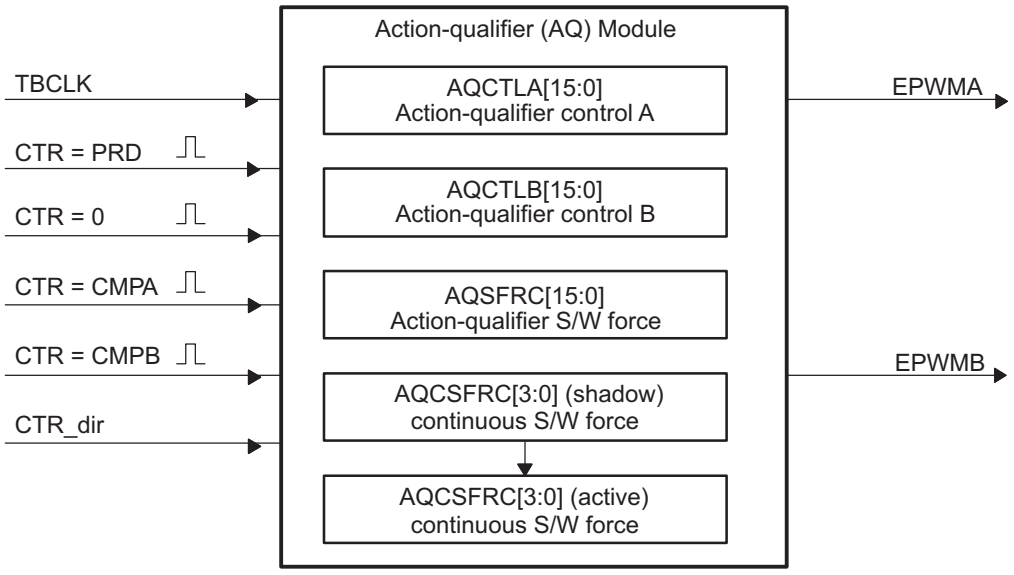


Table 14-8. Action-Qualifier Submodule Possible Input Events

Signal	Description	Registers Compared
CTR = PRD	Time-base counter equal to the period value	TBCNT = TBPRD
CTR = 0	Time-base counter equal to zero	TBCNT = 0000h
CTR = CMPA	Time-base counter equal to the counter-compare A	TBCNT = CMPA
CTR = CMPB	Time-base counter equal to the counter-compare B	TBCNT = CMPB
Software forced event	Asynchronous event initiated by software	

The software forced action is a useful asynchronous event. This control is handled by registers AQSFRC and AQCSFRC.

The action-qualifier submodule controls how the two outputs EPWMxA and EPWMxB behave when a particular event occurs. The event inputs to the action-qualifier submodule are further qualified by the counter direction (up or down). This allows for independent action on outputs on both the count-up and count-down phases.

The possible actions imposed on outputs EPWMxA and EPWMxB are:

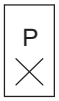
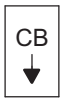
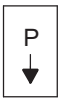
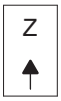
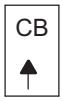
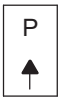
- **Set High:** Set output EPWMxA or EPWMxB to a high level.
- **Clear Low:** Set output EPWMxA or EPWMxB to a low level.
- **Toggle:** If EPWMxA or EPWMxB is currently pulled high, then pull the output low. If EPWMxA or EPWMxB is currently pulled low, then pull the output high.
- **Do Nothing:** Keep outputs EPWMxA and EPWMxB at same level as currently set. Although the "Do Nothing" option prevents an event from causing an action on the EPWMxA and EPWMxB outputs, this event can still trigger interrupts. See the event-trigger submodule description in Section 14.2.9 for details.

Copyright © 2013–2016, Texas Instruments Incorporated

Actions are specified independently for either output (EPWMxA or EPWMxB). Any or all events can be configured to generate actions on a given output. For example, both CTR = CMPA and CTR = CMPB can operate on output EPWMxA. All qualifier actions are configured via the control registers found at the end of this section.

For clarity, the drawings in this chapter use a set of symbolic actions. These symbols are summarized in Figure 14-20. Each symbol represents an action as a marker in time. Some actions are fixed in time (zero and period) while the CMPA and CMPB actions are moveable and their time positions are programmed via the counter-compare A and B registers, respectively. To turn off or disable an action, use the "Do Nothing option"; it is the default at reset.

Figure 14-20. Possible Action-Qualifier Actions for EPWMxA and EPWMxB Outputs

S/W force	TB Counter equals:				Actions
	Zero	Comp A	Comp B	Period	
					Do Nothing
					Clear Low
					Set High
					Toggle

Copyright © 2013–2016, Texas Instruments Incorporated

14.2.5.3 Action-Qualifier Event Priority

It is possible for the ePWM action qualifier to receive more than one event at the same time. In this case events are assigned a priority by the hardware. The general rule is events occurring later in time have a higher priority and software forced events always have the highest priority. The event priority levels for up-down-count mode are shown in Table 14-9. A priority level of 1 is the highest priority and level 7 is the lowest. The priority changes slightly depending on the direction of TBCNT.

Table 14-9. Action-Qualifier Event Priority for Up-Down-Count Mode

Priority Level	Event if TBCNT is Incrementing TBCNT = 0 up to TBCNT = TBPRD	Event if TBCNT is Decrementing TBCNT = TBPRD down to TBCNT = 1
1 (Highest)	Software forced event	Software forced event
2	Counter equals CMPB on up-count (CBU)	Counter equals CMPB on down-count (CBD)
3	Counter equals CMPA on up-count (CAU)	Counter equals CMPA on down-count (CAD)
4	Counter equals zero	Counter equals period (TBPRD)
5	Counter equals CMPB on down-count (CBD) (1)	Counter equals CMPB on up-count (CBU) (1)
6 (Lowest)	Counter equals CMPA on down-count (CAD) (1)	Counter equals CMPA on up-count (CBU) (1)

(1) To maintain symmetry for up-down-count mode, both up-events (CAU/CBU) and down-events (CAD/CBD) can be generated for TBPRD. Otherwise, up-events can occur only when the counter is incrementing and down-events can occur only when the counter is decrementing.

Table 14-10 shows the action-qualifier priority for up-count mode. In this case, the counter direction is always defined as up and thus down-count events will never be taken.

Table 14-10. Action-Qualifier Event Priority for Up-Count Mode

Priority Level	Event
1 (Highest)	Software forced event
2	Counter equal to period (TBPRD)
3	Counter equal to CMPB on up-count (CBU)
4	Counter equal to CMPA on up-count (CAU)
5 (Lowest)	Counter equal to Zero

Table 14-11 shows the action-qualifier priority for down-count mode. In this case, the counter direction is always defined as down and thus up-count events will never be taken.

Table 14-11. Action-Qualifier Event Priority for Down-Count Mode

Priority Level	Event
1 (Highest)	Software forced event
2	Counter equal to Zero
3	Counter equal to CMPB on down-count (CBD)
4	Counter equal to CMPA on down-count (CAD)
5 (Lowest)	Counter equal to period (TBPRD)

Copyright © 2013–2016, Texas Instruments Incorporated

It is possible to set the compare value greater than the period. In this case the action will take place as shown in Table 14-12.

Table 14-12. Behavior if CMPA/CMPB is Greater than the Period

Counter Mode	Compare on Up-Count Event CAU/CBU	Compare on Down-Count Event CAU/CBU
Up-Count Mode	If CMPA/CMPB ≤ TBPRD period, then the event occurs on a compare match (TBCNT = CMPA or CMPB). If CMPA/CMPB > TBPRD, then the event will not occur.	Never occurs.
Down-Count Mode	Never occurs.	If CMPA/CMPB < TBPRD, the event will occur on a compare match (TBCNT = CMPA or CMPB). If CMPA/CMPB ≥ TBPRD, the event will occur on a period match (TBCNT = TBPRD).
Up-Down-Count Mode	If CMPA/CMPB < TBPRD and the counter is incrementing, the event occurs on a compare match (TBCNT = CMPA or CMPB). If CMPA/CMPB is ≥ TBPRD, the event will occur on a period match (TBCNT = TBPRD).	If CMPA/CMPB < TBPRD and the counter is decrementing, the event occurs on a compare match (TBCNT = CMPA or CMPB). If CMPA/CMPB ≥ TBPRD, the event occurs on a period match (TBCNT = TBPRD).

14.2.5.4 Waveforms for Common Configurations

NOTE: The waveforms in this chapter show the ePWMs behavior for a static compare register value. In a running system, the active compare registers (CMPA and CMPB) are typically updated from their respective shadow registers once every period. The user specifies when the update will take place; either when the time-base counter reaches zero or when the time-base counter reaches period. There are some cases when the action based on the new value can be delayed by one period or the action based on the old value can take effect for an extra period. Some PWM configurations avoid this situation. These include, but are not limited to, the following:

Use up-down-count mode to generate a symmetric PWM:

- If you load CMPA/CMPB on zero, then use CMPA/CMPB values greater than or equal to 1.
- If you load CMPA/CMPB on period, then use CMPA/CMPB values less than or equal to TBPRD - 1.

  This means there will always be a pulse of at least one TBCLK cycle in a PWM period which, when very short, tend to be ignored by the system.

Use up-down-count mode to generate an asymmetric PWM:

- To achieve 50%-0% asymmetric PWM use the following configuration: Load CMPA/CMPB on period and use the period action to clear the PWM and a compare-up action to set the PWM. Modulate the compare value from 0 to TBPRD to achieve 50%-0% PWM duty.

When using up-count mode to generate an asymmetric PWM:

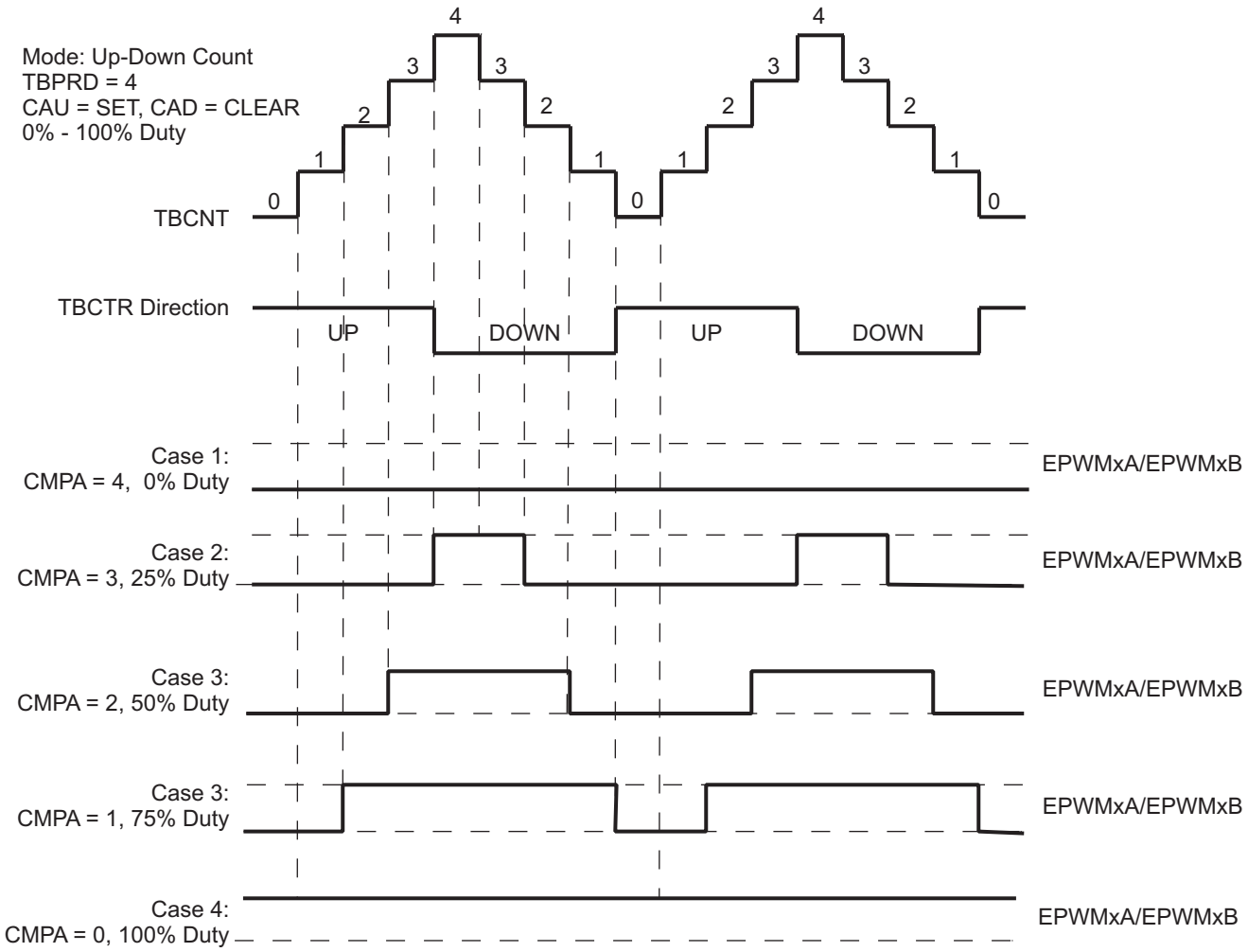
- To achieve 0-100% asymmetric PWM use the following configuration: Load CMPA/CMPB on TBPRD. Use the Zero action to set the PWM and a compare-up action to clear the PWM. Modulate the compare value from 0 to TBPRD+1 to achieve 0-100% PWM duty.

Copyright © 2013–2016, Texas Instruments Incorporated

Figure 14-21 shows how a symmetric PWM waveform can be generated using the up-down-count mode of the TBCNT. In this mode 0%-100% DC modulation is achieved by using equal compare matches on the up count and down count portions of the waveform. In the example shown, CMPA is used to make the comparison. When the counter is incrementing the CMPA match will pull the PWM output high. Likewise, when the counter is decrementing the compare match will pull the PWM signal low. When CMPA = 0, the PWM signal is low for the entire period giving the 0% duty waveform. When CMPA = TBPRD, the PWM signal is high achieving 100% duty.

When using this configuration in practice, if you load CMPA/CMPB on zero, then use CMPA/CMPB values greater than or equal to 1. If you load CMPA/CMPB on period, then use CMPA/CMPB values less than or equal to TBPRD-1. This means there will always be a pulse of at least one TBCLK cycle in a PWM period which, when very short, tend to be ignored by the system.

Figure 14-21. Up-Down-Count Mode Symmetrical Waveform



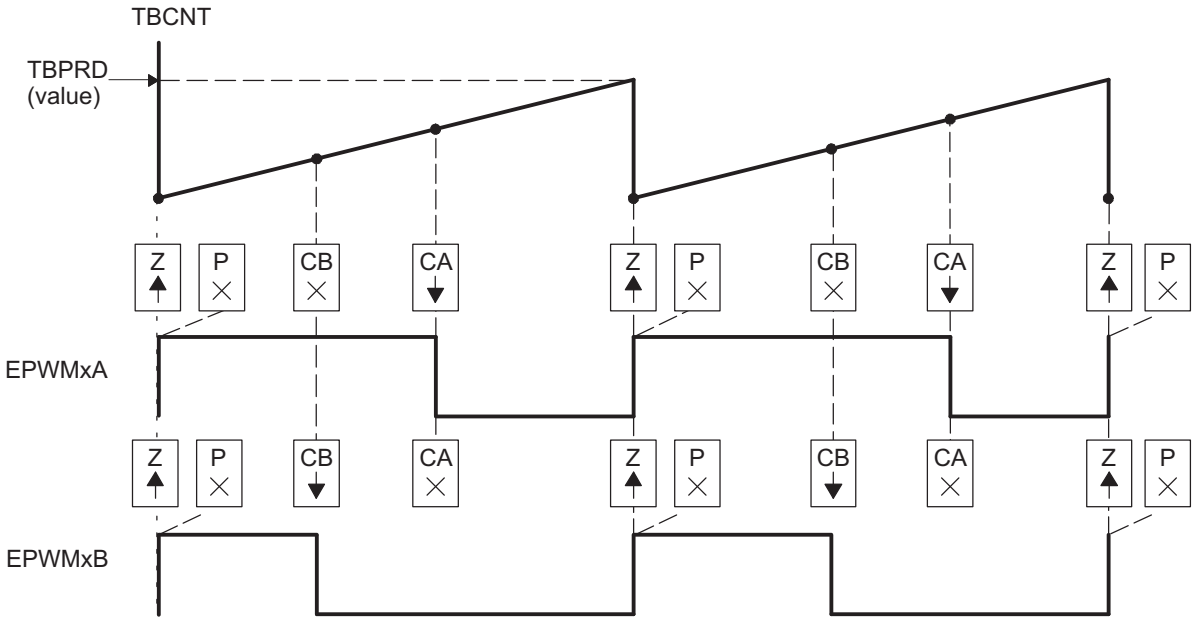
Copyright © 2013–2016, Texas Instruments Incorporated

The PWM waveforms in Figure 14-22 through Figure 14-27 show some common action-qualifier configurations. Some conventions used in the figures are as follows:

- TBPRD, CMPA, and CMPB refer to the value written in their respective registers. The active register, not the shadow register, is used by the hardware.
- CMPx, refers to either CMPA or CMPB.
- EPWMxA and EPWMxB refer to the output signals from ePWMx
- Up-Down means Count-up-and-down mode, Up means up-count mode and Dwn means down-count mode
- Sym = Symmetric, Asym = Asymmetric

Table 14-13 and Table 14-14 contains initialization and runtime register configurations for the waveforms in Figure 14-22.

Figure 14-22. Up, Single Edge Asymmetric Waveform, With Independent Modulation on EPWMxA and EPWMxB—Active High



(1) PWM period = (TBPRD + 1) × T_{TBCLK}

(2) Duty modulation for EPWMxA is set by CMPA, and is active high (that is, high time duty proportional to CMPA).

(3) Duty modulation for EPWMxB is set by CMPB and is active high (that is, high time duty proportional to CMPB).

(4) The "Do Nothing" actions (X) are shown for completeness, but will not be shown on subsequent diagrams.

(5) Actions at zero and period, although appearing to occur concurrently, are actually separated by one TBCLK period. TBCNT wraps from period to 0000h.

Copyright © 2013–2016, Texas Instruments Incorporated

Table 14-13. EPWMx Initialization for Figure 14-22

Register	Bit	Value	Comments
TBPRD	TBPRD	600 (258h)	Period = 601 TBCLK counts
TBPHS	TBPHS	0	Clear Phase Register to 0
TBCNT	TBCNT	0	Clear TB counter
TBCTL	CTRMODE	TB_UP	
	PHSEN	TB_DISABLE	Phase loading disabled
	PRDLD	TB_SHADOW	
	SYNCOSEL	TB_SYNC_DISABLE	
	HSPCLKDIV	TB_DIV1	TBCLK = SYSCLK
	CLKDIV	TB_DIV1	
CMPA	CMPA	350 (15Eh)	Compare A = 350 TBCLK counts
CMPB	CMPB	200 (C8h)	Compare B = 200 TBCLK counts
CMPCTL	SHDWAMODE	CC_SHADOW	
	SHDWBMODE	CC_SHADOW	
	LOADAMODE	CC_CTR_ZERO	Load on CTR = 0
	LOADBMODE	CC_CTR_ZERO	Load on CTR = 0
AQCTLA	ZRO	AQ_SET	
	CAU	AQ_CLEAR	
AQCTLB	ZRO	AQ_SET	
	CBU	AQ_CLEAR	

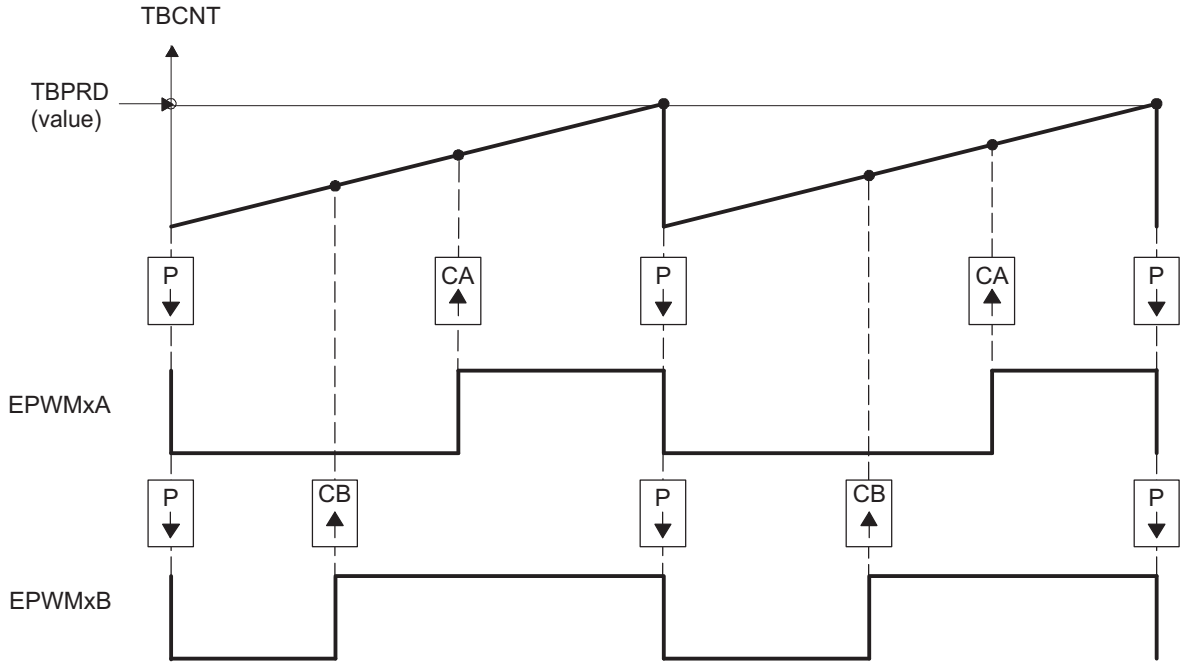
Table 14-14. EPWMx Run Time Changes for Figure 14-22

Register	Bit	Value	Comments
CMPA	CMPA	Duty1A	Adjust duty for output EPWM1A
CMPB	CMPB	Duty1B	Adjust duty for output EPWM1B

Copyright © 2013–2016, Texas Instruments Incorporated

Table 14-15 and Table 14-16 contains initialization and runtime register configurations for the waveforms in Figure 14-23.

Figure 14-23. Up, Single Edge Asymmetric Waveform With Independent Modulation on EPWMxA and EPWMxB—Active Low



(1) PWM period = (TBPRD + 1) × T_{TBCLK}

(2) Duty modulation for EPWMxA is set by CMPA, and is active low (that is, the low time duty is proportional to CMPA).

(3) Duty modulation for EPWMxB is set by CMPB and is active low (that is, the low time duty is proportional to CMPB).

(4) The Do Nothing actions (X) are shown for completeness here, but will not be shown on subsequent diagrams.

(5) Actions at zero and period, although appearing to occur concurrently, are actually separated by one TBCLK period. TBCNT wraps from period to 0000h.

Copyright © 2013–2016, Texas Instruments Incorporated

Table 14-15. EPWMx Initialization for Figure 14-23

Register	Bit	Value	Comments
TBPRD	TBPRD	600 (258h)	Period = 601 TBCLK counts
TBPHS	TBPHS	0	Clear Phase Register to 0
TBCNT	TBCNT	0	Clear TB counter
TBCTL	CTRMODE	TB_UP	
	PHSEN	TB_DISABLE	Phase loading disabled
	PRDLD	TB_SHADOW	
	SYNCOSEL	TB_SYNC_DISABLE	
	HSPCLKDIV	TB_DIV1	TBCLK = SYSCLK
	CLKDIV	TB_DIV1	
CMPA	CMPA	350 (15Eh)	Compare A = 350 TBCLK counts
CMPB	CMPB	200 (C8h)	Compare B = 200 TBCLK counts
CMPCTL	SHDWAMODE	CC_SHADOW	
	SHDWBMODE	CC_SHADOW	
	LOADAMODE	CC_CTR_ZERO	Load on CTR = 0
	LOADBMODE	CC_CTR_ZERO	Load on CTR = 0
AQCTLA	PRD	AQ_CLEAR	
	CAU	AQ_SET	
AQCTLB	PRD	AQ_CLEAR	
	CBU	AQ_SET	

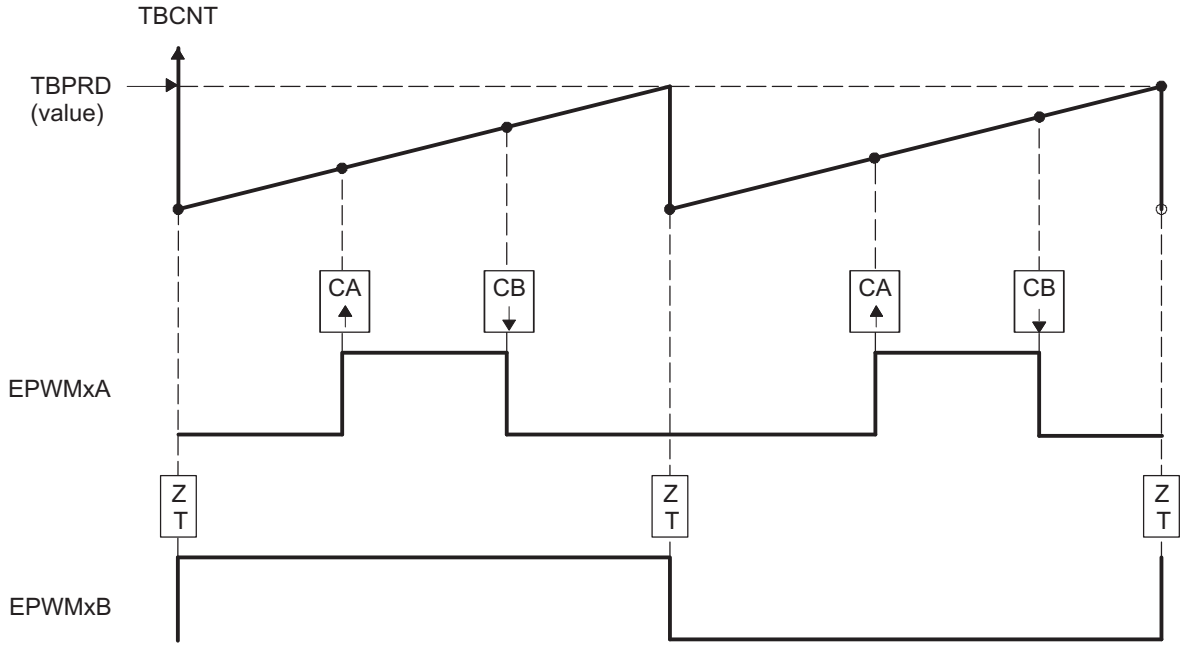
Table 14-16. EPWMx Run Time Changes for Figure 14-23

Register	Bit	Value	Comments
CMPA	CMPA	Duty1A	Adjust duty for output EPWM1A
CMPB	CMPB	Duty1B	Adjust duty for output EPWM1B

Copyright © 2013–2016, Texas Instruments Incorporated

Table 14-17 and Table 14-18 contains initialization and runtime register configurations for the waveforms Figure 14-24. Use the code in Example 14-1 to define the headers.

Figure 14-24. Up-Count, Pulse Placement Asymmetric Waveform With Independent Modulation on EPWMxA



(1) PWM frequency = 1/((TBPRD + 1) × T_{TBCLK})

(2) Pulse can be placed anywhere within the PWM cycle (0000h - TBPRD)

(3) High time duty proportional to (CMPB - CMPA)

(4) EPWMxB can be used to generate a 50% duty square wave with frequency = 1/2 × ((TBPRD + 1) × TBCLK)

Copyright © 2013–2016, Texas Instruments Incorporated

Table 14-17. EPWMx Initialization for Figure 14-24

Register	Bit	Value	Comments
TBPRD	TBPRD	600 (258h)	Period = 601 TBCLK counts
TBPHS	TBPHS	0	Clear Phase Register to 0
TBCNT	TBCNT	0	Clear TB counter
TBCTL	CTRMODE	TB_UP	
	PHSEN	TB_DISABLE	Phase loading disabled
	PRDLD	TB_SHADOW	
	SYNCOSEL	TB_SYNC_DISABLE	
	HSPCLKDIV	TB_DIV1	TBCLK = SYSCLK
	CLKDIV	TB_DIV1	
CMPA	CMPA	200 (C8h)	Compare A = 200 TBCLK counts
CMPB	CMPB	400 (190h)	Compare B = 400 TBCLK counts
CMPCTL	SHDWAMODE	CC_SHADOW	
	SHDWBMODE	CC_SHADOW	
	LOADAMODE	CC_CTR_ZERO	Load on CTR = 0
	LOADBMODE	CC_CTR_ZERO	Load on CTR = 0
AQCTLA	CAU	AQ_SET	
	CBU	AQ_CLEAR	
AQCTLB	ZRO	AQ_TOGGLE	

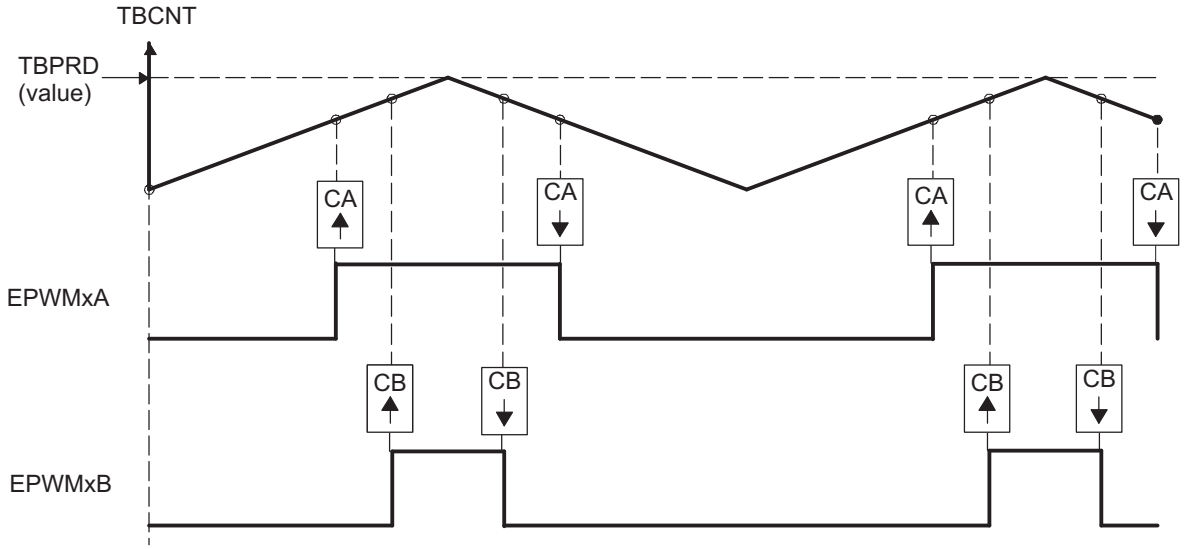
Table 14-18. EPWMx Run Time Changes for Figure 14-24

Register	Bit	Value	Comments
CMPA	CMPA	EdgePosA	Adjust duty for output EPWM1A
CMPB	CMPB	EdgePosB	

Copyright © 2013–2016, Texas Instruments Incorporated

Table 14-19 and Table 14-20 contains initialization and runtime register configurations for the waveforms in Figure 14-25. Use the code in Example 14-1 to define the headers.

Figure 14-25. Up-Down-Count, Dual Edge Symmetric Waveform, With Independent Modulation on EPWMxA and EPWMxB — Active Low



(1) PWM period = 2 x TBPRD × T_{TBCLK}

(2) Duty modulation for EPWMxA is set by CMPA, and is active low (that is, the low time duty is proportional to CMPA).

(3) Duty modulation for EPWMxB is set by CMPB and is active low (that is, the low time duty is proportional to CMPB).

(4) Outputs EPWMxA and EPWMxB can drive independent power switches

Copyright © 2013–2016, Texas Instruments Incorporated

Table 14-19. EPWMx Initialization for Figure 14-25

Register	Bit	Value	Comments
TBPRD	TBPRD	600 (258h)	Period = 601 TBCLK counts
TBPHS	TBPHS	0	Clear Phase Register to 0
TBCNT	TBCNT	0	Clear TB counter
TBCTL	CTRMODE	TB_UPDOWN	
	PHSEN	TB_DISABLE	Phase loading disabled
	PRDLD	TB_SHADOW	
	SYNCOSEL	TB_SYNC_DISABLE	
	HSPCLKDIV	TB_DIV1	TBCLK = SYSCLK
	CLKDIV	TB_DIV1	
CMPA	CMPA	400 (190h)	Compare A = 400 TBCLK counts
CMPB	CMPB	500 (1F4h)	Compare B = 500 TBCLK counts
CMPCTL	SHDWAMODE	CC_SHADOW	
	SHDWBMODE	CC_SHADOW	
	LOADAMODE	CC_CTR_ZERO	Load on CTR = 0
	LOADBMODE	CC_CTR_ZERO	Load on CTR = 0
AQCTLA	CAU	AQ_SET	
	CAD	AQ_CLEAR	
AQCTLB	CBU	AQ_SET	
	CBD	AQ_CLEAR	

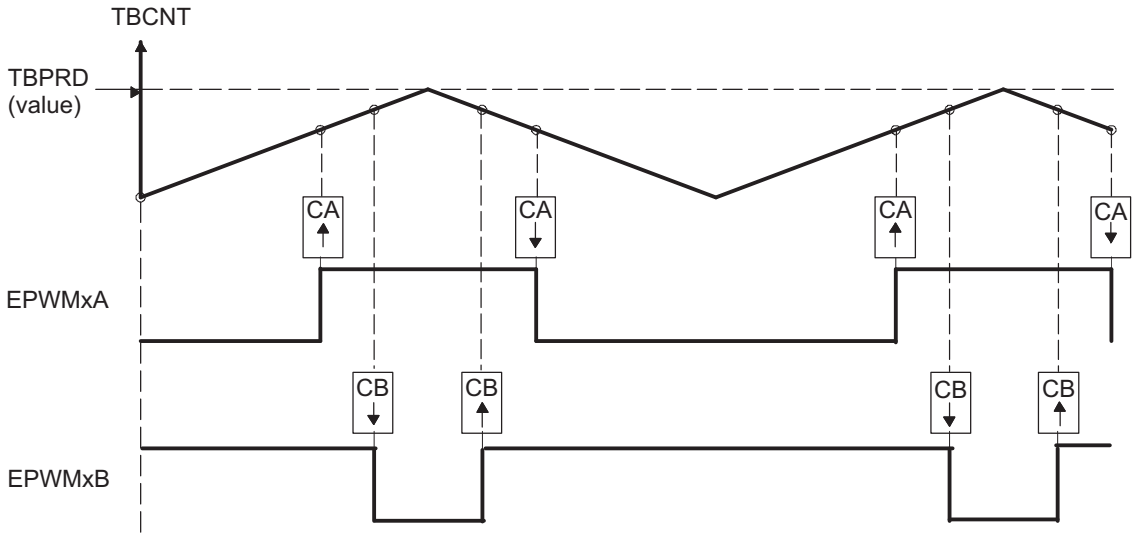
Table 14-20. EPWMx Run Time Changes for Figure 14-25

Register	Bit	Value	Comments
CMPA	CMPA	Duty1A	Adjust duty for output EPWM1A
CMPB	CMPB	Duty1B	Adjust duty for output EPWM1B

Copyright © 2013–2016, Texas Instruments Incorporated

Table 14-21 and Table 14-22 contains initialization and runtime register configurations for the waveforms in Figure 14-26. Use the code in Example 14-1 to define the headers.

Figure 14-26. Up-Down-Count, Dual Edge Symmetric Waveform, With Independent Modulation on EPWMxA and EPWMxB — Complementary



(1) PWM period = 2 × TBPRD × T_{TBCLK}

(2) Duty modulation for EPWMxA is set by CMPA, and is active low, i.e., low time duty proportional to CMPA

(3) Duty modulation for EPWMxB is set by CMPB and is active high, i.e., high time duty proportional to CMPB

(4) Outputs EPWMx can drive upper/lower (complementary) power switches

(5) Dead-band = CMPB - CMPA (fully programmable edge placement by software). Note the dead-band module is also available if the more classical edge delay method is required.

Copyright © 2013–2016, Texas Instruments Incorporated

Table 14-21. EPWMx Initialization for Figure 14-26

Register	Bit	Value	Comments
TBPRD	TBPRD	600 (258h)	Period = 601 TBCLK counts
TBPHS	TBPHS	0	Clear Phase Register to 0
TBCNT	TBCNT	0	Clear TB counter
TBCTL	CTRMODE	TB_UPDOWN	
	PHSEN	TB_DISABLE	Phase loading disabled
	PRDLD	TB_SHADOW	
	SYNCOSEL	TB_SYNC_DISABLE	
	HSPCLKDIV	TB_DIV1	TBCLK = SYSCLK
	CLKDIV	TB_DIV1	
CMPA	CMPA	350 (15Eh)	Compare A = 350 TBCLK counts
CMPB	CMPB	400 (190h)	Compare B = 400 TBCLK counts
CMPCTL	SHDWAMODE	CC_SHADOW	
	SHDWBMODE	CC_SHADOW	
	LOADAMODE	CC_CTR_ZERO	Load on CTR = 0
	LOADBMODE	CC_CTR_ZERO	Load on CTR = 0
AQCTLA	CAU	AQ_SET	
	CAD	AQ_CLEAR	
AQCTLB	CBU	AQ_CLEAR	
	CBD	AQ_SET	

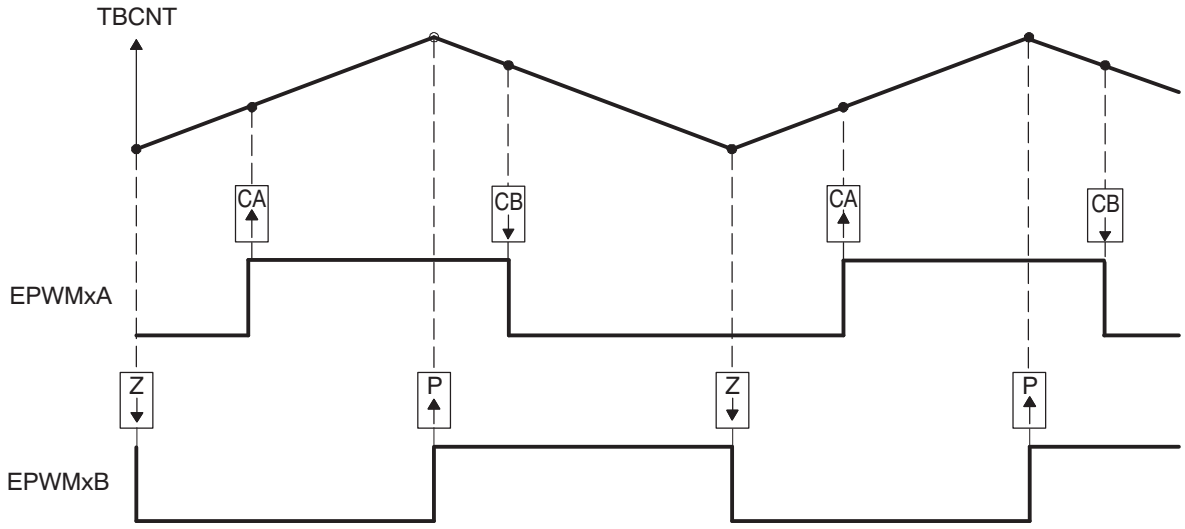
Table 14-22. EPWMx Run Time Changes for Figure 14-26

Register	Bit	Value	Comments
CMPA	CMPA	Duty1A	Adjust duty for output EPWM1A
CMPB	CMPB	Duty1B	Adjust duty for output EPWM1B

Copyright © 2013–2016, Texas Instruments Incorporated

Table 14-23 and Table 14-24 contains initialization and runtime register configurations for the waveforms in Figure 14-27. Use the code in Example 14-1 to define the headers.

Figure 14-27. Up-Down-Count, Dual Edge Asymmetric Waveform, With Independent Modulation on EPWMxA—Active Low



(1) PWM period = 2 × TBPRD × TBCLK

(2) Rising edge and falling edge can be asymmetrically positioned within a PWM cycle. This allows for pulse placement techniques.

(3) Duty modulation for EPWMxA is set by CMPA and CMPB.

(4) Low time duty for EPWMxA is proportional to (CMPA + CMPB).

(5) To change this example to active high, CMPA and CMPB actions need to be inverted (i.e., Set ! Clear and Clear Set).

(6) Duty modulation for EPWMxB is fixed at 50% (utilizes spare action resources for EPWMxB)

Copyright © 2013–2016, Texas Instruments Incorporated

Table 14-23. EPWMx Initialization for Figure 14-27

Register	Bit	Value	Comments
TBPRD	TBPRD	600 (258h)	Period = 601 TBCLK counts
TBPHS	TBPHS	0	Clear Phase Register to 0
TBCNT	TBCNT	0	Clear TB counter
TBCTL	CTRMODE	TB_UPDOWN	
	PHSEN	TB_DISABLE	Phase loading disabled
	PRDLD	TB_SHADOW	
	SYNCOSEL	TB_SYNC_DISABLE	
	HSPCLKDIV	TB_DIV1	TBCLK = SYSCLK
	CLKDIV	TB_DIV1	
CMPA	CMPA	250 (FAh)	Compare A = 250 TBCLK counts
CMPB	CMPB	450 (1C2h)	Compare B = 450 TBCLK counts
CMPCTL	SHDWAMODE	CC_SHADOW	
	SHDWBMODE	CC_SHADOW	
	LOADAMODE	CC_CTR_ZERO	Load on CTR = 0
	LOADBMODE	CC_CTR_ZERO	Load on CTR = 0
AQCTLA	CAU	AQ_SET	
	CBD	AQ_CLEAR	
AQCTLB	ZRO	AQ_CLEAR	
	PRD	AQ_SET	

Table 14-24. EPWMx Run Time Changes for Figure 14-27

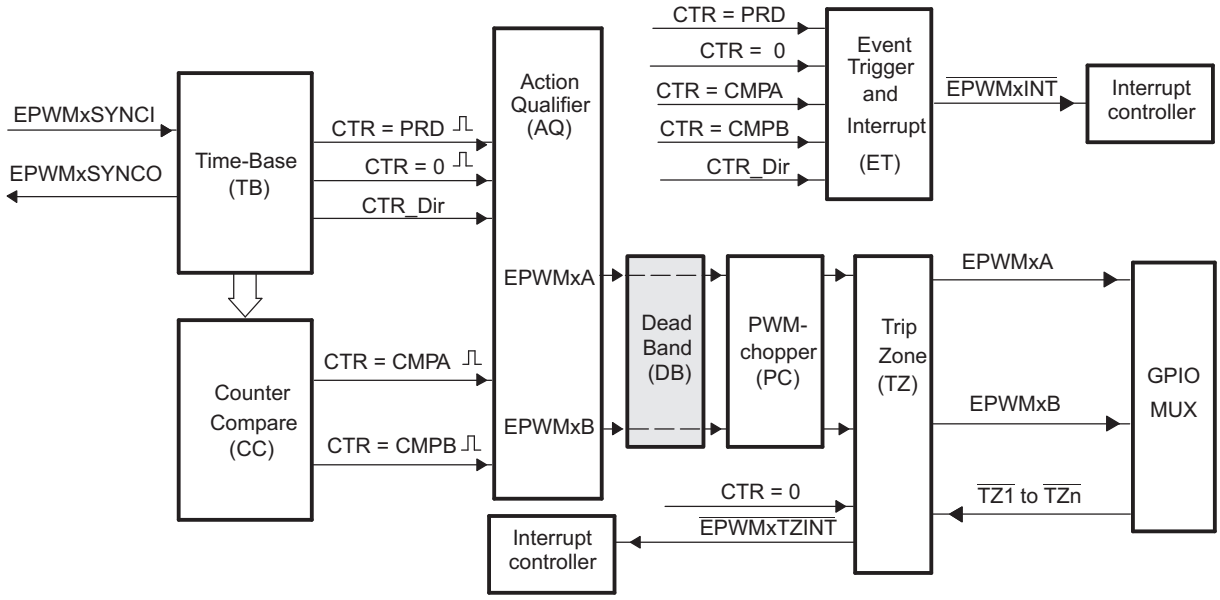
Register	Bit	Value	Comments
CMPA	CMPA	EdgePosA	Adjust duty for output EPWM1A
CMPB	CMPB	EdgePosB	

Copyright © 2013–2016, Texas Instruments Incorporated

14.2.6 Dead-Band Generator (DB) Submodule

Figure 14-28 illustrates the dead-band generator submodule within the ePWM module.

Figure 14-28. Dead-Band Generator Submodule



14.2.6.1 Purpose of the Dead-Band Submodule

The "Action-qualifier (AQ) Module" section discussed how it is possible to generate the required dead-band by having full control over edge placement using both the CMPA and CMPB resources of the ePWM module. However, if the more classical edge delay-based dead-band with polarity control is required, then the dead-band generator submodule should be used.

The key functions of the dead-band generator submodule are:

- Generating appropriate signal pairs (EPWMxA and EPWMxB) with dead-band relationship from a single EPWMxA input
- Programming signal pairs for:
  - Active high (AH)
  - Active low (AL)
  - Active high complementary (AHC)
  - Active low complementary (ALC)
- Adding programmable delay to rising edges (RED)
- Adding programmable delay to falling edges (FED)
- Can be totally bypassed from the signal path (note dotted lines in diagram)

14.2.6.2 Controlling and Monitoring the Dead-Band Submodule

The dead-band generator submodule operation is controlled and monitored via the following registers:

Table 14-25. Dead-Band Generator Submodule Registers

Acronym	Register Description	Address Offset	Shadowed
DBCTL	Dead-Band Control Register	1Eh	No
DBRED	Dead-Band Rising Edge Delay Count Register	20h	No
DBFED	Dead-Band Falling Edge Delay Count Register	22h	No

Copyright © 2013–2016, Texas Instruments Incorporated

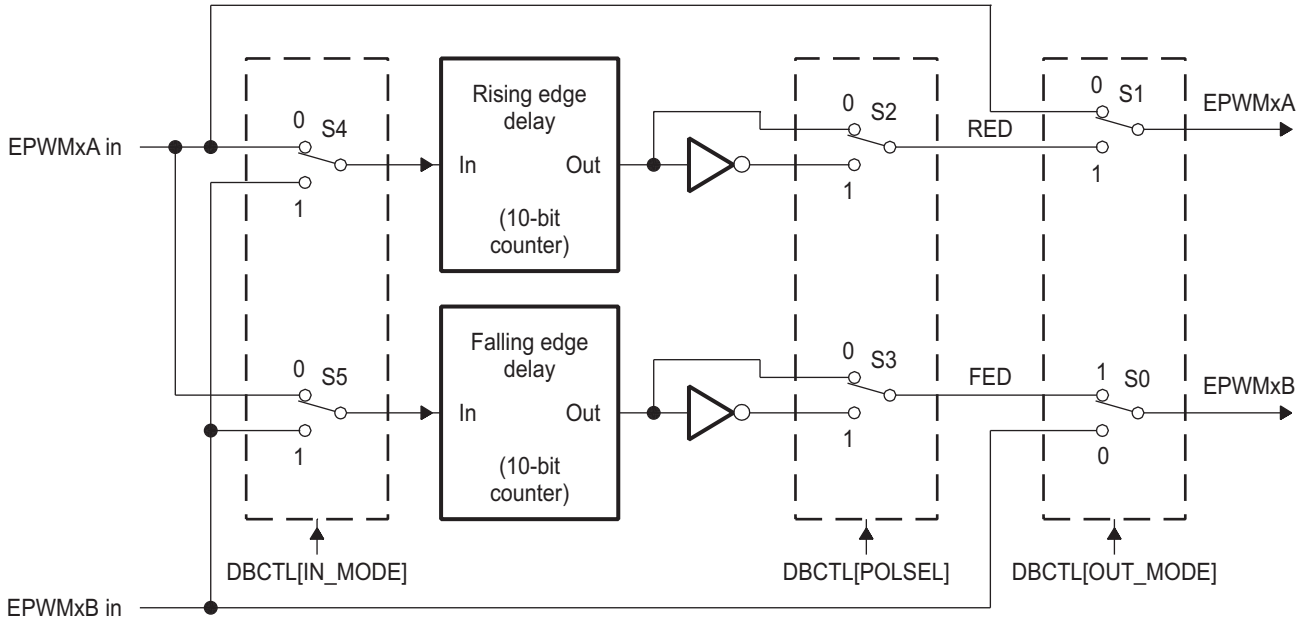
14.2.6.3 Operational Highlights for the Dead-Band Generator Submodule

The following sections provide the operational highlights.

The dead-band submodule has two groups of independent selection options as shown in Figure 14-29.

- **Input Source Selection:** The input signals to the dead-band module are the EPWMxA and EPWMxB output signals from the action-qualifier. In this section they will be referred to as EPWMxA In and EPWMxB In. Using the DBCTL[IN_MODE) control bits, the signal source for each delay, falling-edge or rising-edge, can be selected:
  - EPWMxA In is the source for both falling-edge and rising-edge delay. This is the default mode.
  - EPWMxA In is the source for falling-edge delay, EPWMxB In is the source for rising-edge delay.
  - EPWMxA In is the source for rising edge delay, EPWMxB In is the source for falling-edge delay.
  - EPWMxB In is the source for both falling-edge and rising-edge delay.
- **Output Mode Control:** The output mode is configured by way of the DBCTL[OUT_MODE] bits. These bits determine if the falling-edge delay, rising-edge delay, neither, or both are applied to the input signals.
- **Polarity Control:** The polarity control (DBCTL[POLSEL]) allows you to specify whether the rising-edge delayed signal and/or the falling-edge delayed signal is to be inverted before being sent out of the dead-band submodule.

Figure 14-29. Configuration Options for the Dead-Band Generator Submodule



Copyright © 2013–2016, Texas Instruments Incorporated

Although all combinations are supported, not all are typical usage modes. Table 14-26 lists some classical dead-band configurations. These modes assume that the DBCTL[IN_MODE] is configured such that EPWMxA In is the source for both falling-edge and rising-edge delay. Enhanced, or non-traditional modes can be achieved by changing the input signal source. The modes shown in Table 14-26 fall into the following categories:

- **Mode 1: Bypass both falling-edge delay (FED) and rising-edge delay (RED)** Allows you to fully disable the dead-band submodule from the PWM signal path.
- **Mode 2-5: Classical Dead-Band Polarity Settings** These represent typical polarity configurations that should address all the active high/low modes required by available industry power switch gate drivers. The waveforms for these typical cases are shown in Figure 14-30. Note that to generate equivalent waveforms to Figure 14-30, configure the action-qualifier submodule to generate the signal as shown for EPWMxA.
- **Mode 6: Bypass rising-edge-delay and Mode 7: Bypass falling-edge-delay** Finally the last two entries in Table 14-26 show combinations where either the falling-edge-delay (FED) or rising-edge-delay (RED) blocks are bypassed.

Table 14-26. Classical Dead-Band Operating Modes

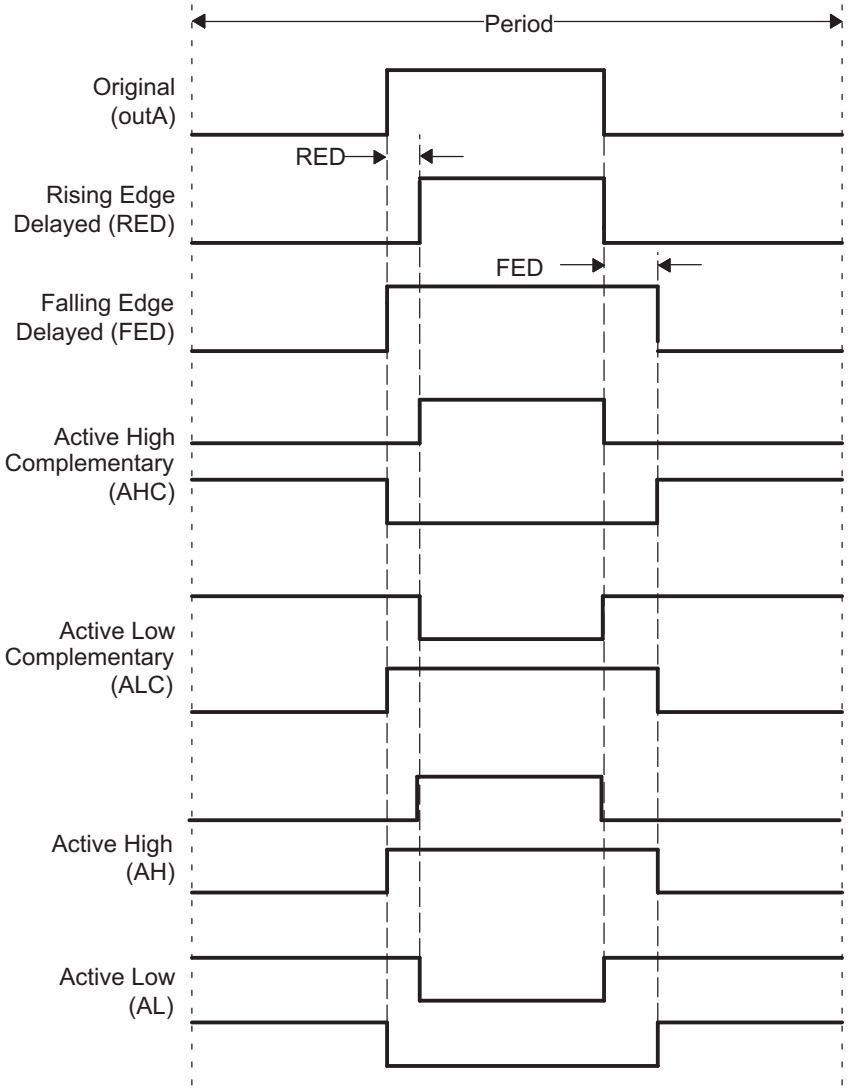
Mode	Mode Description [1]	DBCTL[POLSEL]		DBCTL[OUT_MODE]	
		S3	S2	S1	S0
1	EPWMxA and EPWMxB Passed Through (No Delay)	x	x	0	0
2	Active High Complementary (AHC)	1	0	1	1
3	Active Low Complementary (ALC)	0	1	1	1
4	Active High (AH)	0	0	1	1
5	Active Low (AL)	1	1	1	1
6	EPWMxA Out = EPWMxA In (No Delay) EPWMxB Out = EPWMxA In with Falling Edge Delay	0 or 1	0 or 1	0	1
7	EPWMxA Out = EPWMxA In with Rising Edge Delay EPWMxB Out = EPWMxB In with No Delay	0 or 1	0 or 1	1	0

(1) These are classical dead-band modes and assume that DBCTL[IN_MODE] = 0,0. That is, EPWMxA in is the source for both the falling-edge and rising-edge delays. Enhanced, non-traditional modes can be achieved by changing the IN_MODE configuration.

Copyright © 2013–2016, Texas Instruments Incorporated

Figure 14-30 shows waveforms for typical cases where 0% < duty < 100%.

Figure 14-30. Dead-Band Waveforms for Typical Cases (0% < Duty < 100%)



The dead-band submodule supports independent values for rising-edge (RED) and falling-edge (FED) delays. The amount of delay is programmed using the DBRED and DBFED registers. These are 10-bit registers and their value represents the number of time-base clock, TBCLK, periods a signal edge is delayed by. For example, the formula to calculate falling-edge-delay and rising-edge-delay are:

$FED = DBFED \times T_{TBCLK}$

$RED = DBRED \times T_{TBCLK}$

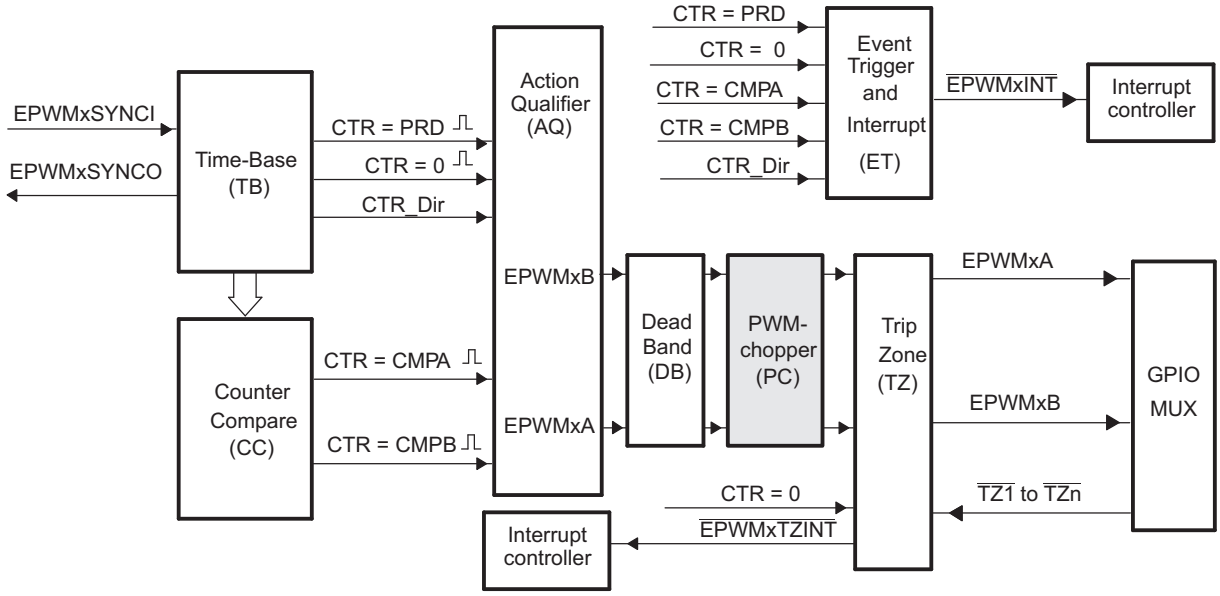
Where T_{TBCLK} is the period of TBCLK, the prescaled version of SYSCLKOUT.

Copyright © 2013–2016, Texas Instruments Incorporated

14.2.7 PWM-Chopper (PC) Submodule

Figure 14-31 illustrates the PWM-chopper (PC) submodule within the ePWM module. The PWM-chopper submodule allows a high-frequency carrier signal to modulate the PWM waveform generated by the action-qualifier and dead-band submodules. This capability is important if you need pulse transformer-based gate drivers to control the power switching elements.

Figure 14-31. PWM-Chopper Submodule



14.2.7.1 Purpose of the PWM-Chopper Submodule

The key functions of the PWM-chopper submodule are:

- Programmable chopping (carrier) frequency
- Programmable pulse width of first pulse
- Programmable duty cycle of second and subsequent pulses
- Can be fully bypassed if not required

14.2.7.2 Controlling the PWM-Chopper Submodule

The PWM-chopper submodule operation is controlled via the register in Table 14-27.

Table 14-27. PWM-Chopper Submodule Registers

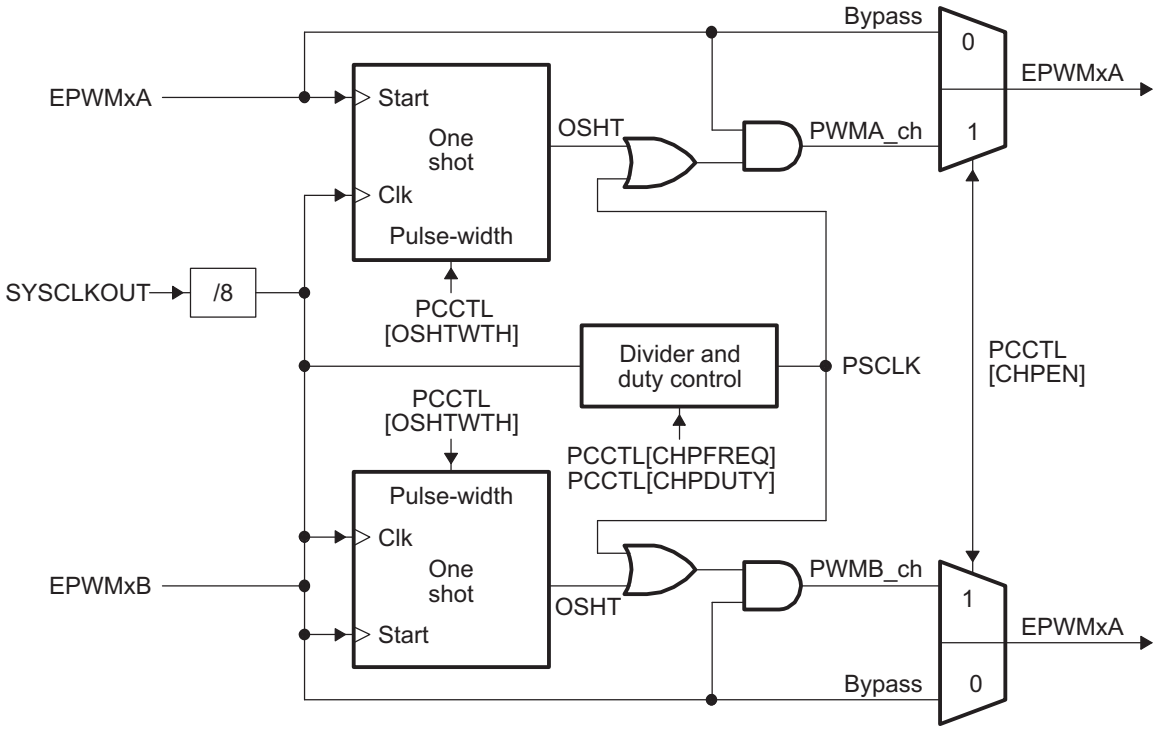
Acronym	Register Description	Address Offset	Shadowed
PCCTL	PWM-chopper Control Register	3Ch	No

Copyright © 2013–2016, Texas Instruments Incorporated

14.2.7.3 Operational Highlights for the PWM-Chopper Submodule

Figure 14-32 shows the operational details of the PWM-chopper submodule. The carrier clock is derived from SYSCLKOUT. Its frequency and duty cycle are controlled via the CHPFREQ and CHPDUTY bits in the PCCTL register. The one-shot block is a feature that provides a high energy first pulse to ensure hard and fast power switch turn on, while the subsequent pulses sustain pulses, ensuring the power switch remains on. The one-shot width is programmed via the OSHTWTH bits. The PWM-chopper submodule can be fully disabled (bypassed) via the CHPEN bit.

Figure 14-32. PWM-Chopper Submodule Signals and Registers

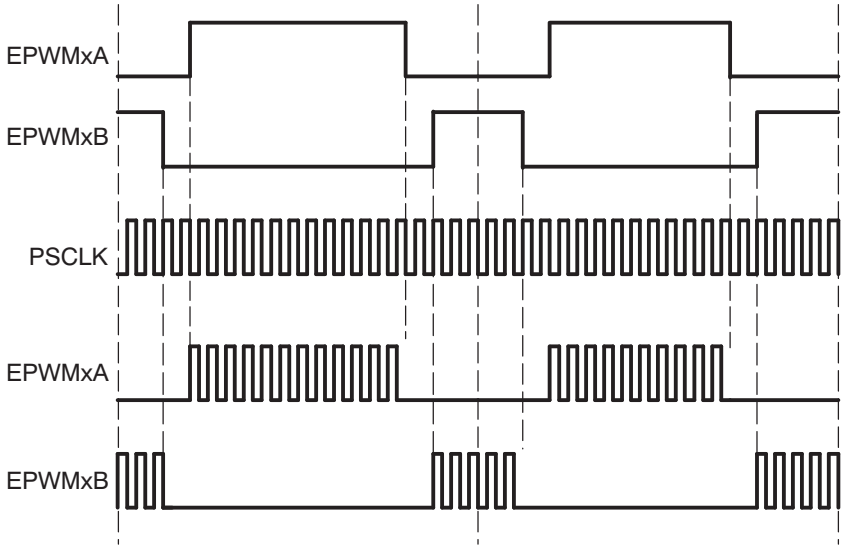


Copyright © 2013–2016, Texas Instruments Incorporated

14.2.7.4 Waveforms

Figure 14-33 shows simplified waveforms of the chopping action only; one-shot and duty-cycle control are not shown. Details of the one-shot and duty-cycle control are discussed in the following sections.

Figure 14-33. Simple PWM-Chopper Submodule Waveforms Showing Chopping Action Only



14.2.7.4.1 One-Shot Pulse

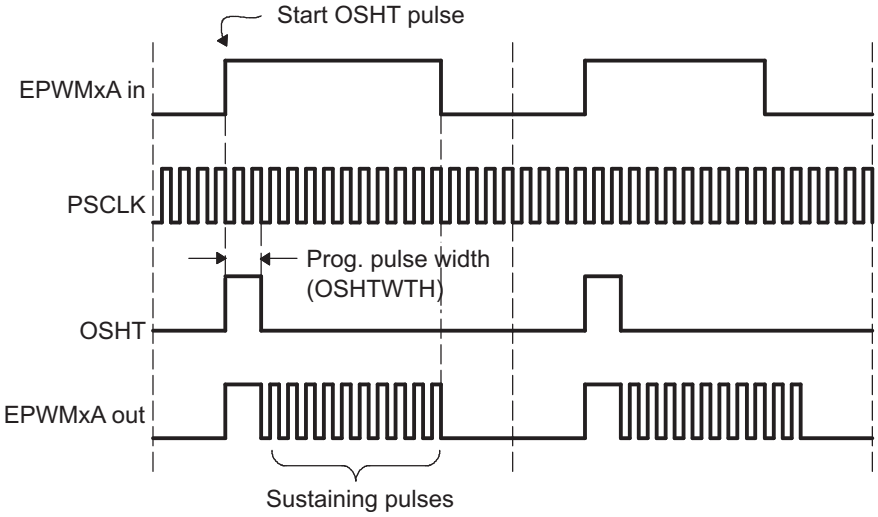
The width of the first pulse can be programmed to any of 16 possible pulse width values. The width or period of the first pulse is given by:

$T_{1stpulse} = T_{SYSCLKOUT} \times 8 \times OSHTWTH$

Where $T_{SYSCLKOUT}$ is the period of the system clock (SYSCLKOUT) and OSHTWTH is the four control bits (value from 1 to 16)

Figure 14-34 shows the first and subsequent sustaining pulses.

Figure 14-34. PWM-Chopper Submodule Waveforms Showing the First Pulse and Subsequent Sustaining Pulses



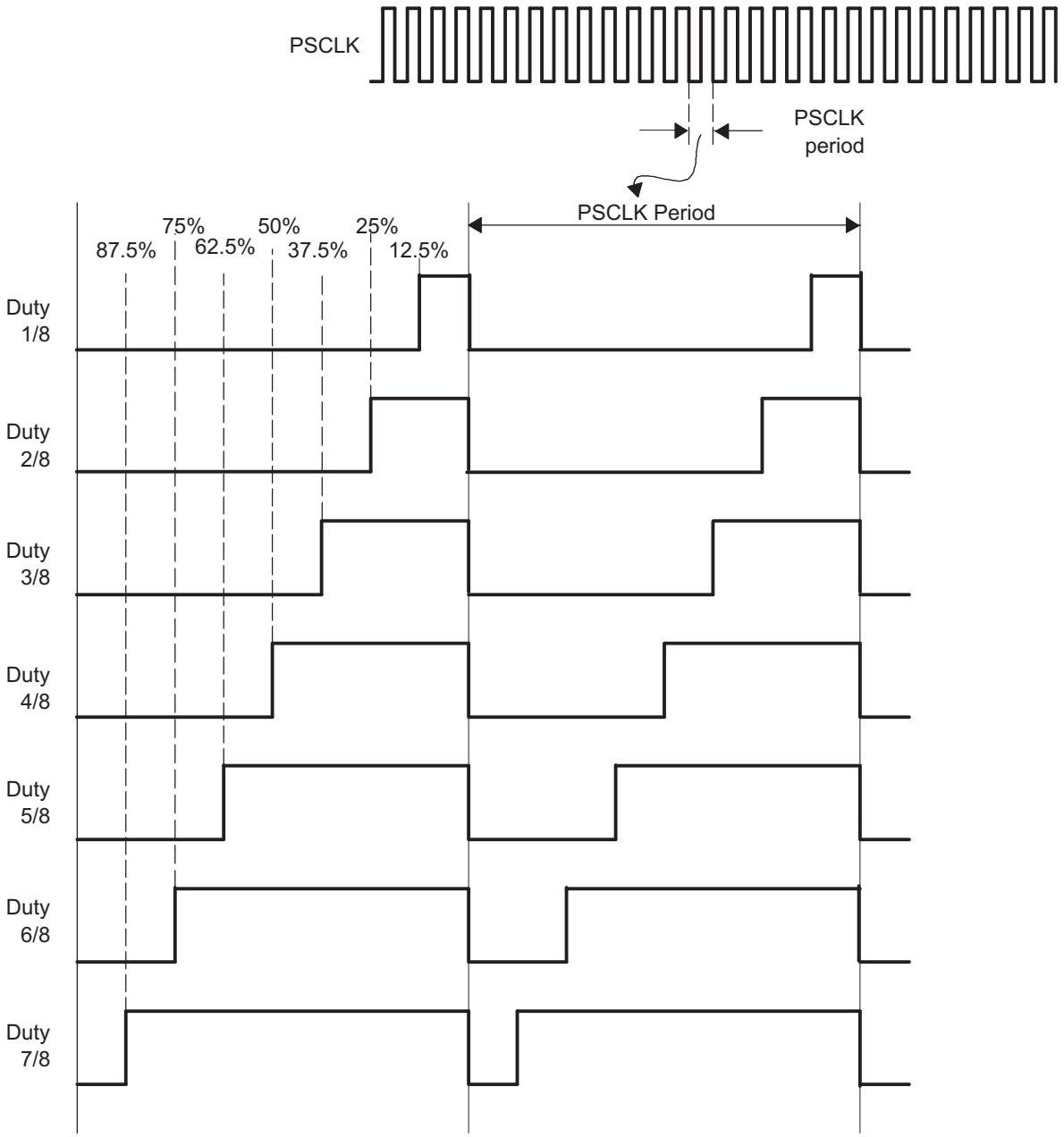
Copyright © 2013–2016, Texas Instruments Incorporated

14.2.7.4.2 Duty Cycle Control

Pulse transformer-based gate drive designs need to comprehend the magnetic properties or characteristics of the transformer and associated circuitry. Saturation is one such consideration. To assist the gate drive designer, the duty cycles of the second and subsequent pulses have been made programmable. These sustaining pulses ensure the correct drive strength and polarity is maintained on the power switch gate during the on period, and hence a programmable duty cycle allows a design to be tuned or optimized via software control.

Figure 14-35 shows the duty cycle control that is possible by programming the CHPDUTY bits. One of seven possible duty ratios can be selected ranging from 12.5% to 87.5%.

Figure 14-35. PWM-Chopper Submodule Waveforms Showing the Pulse Width (Duty Cycle) Control of Sustaining Pulses

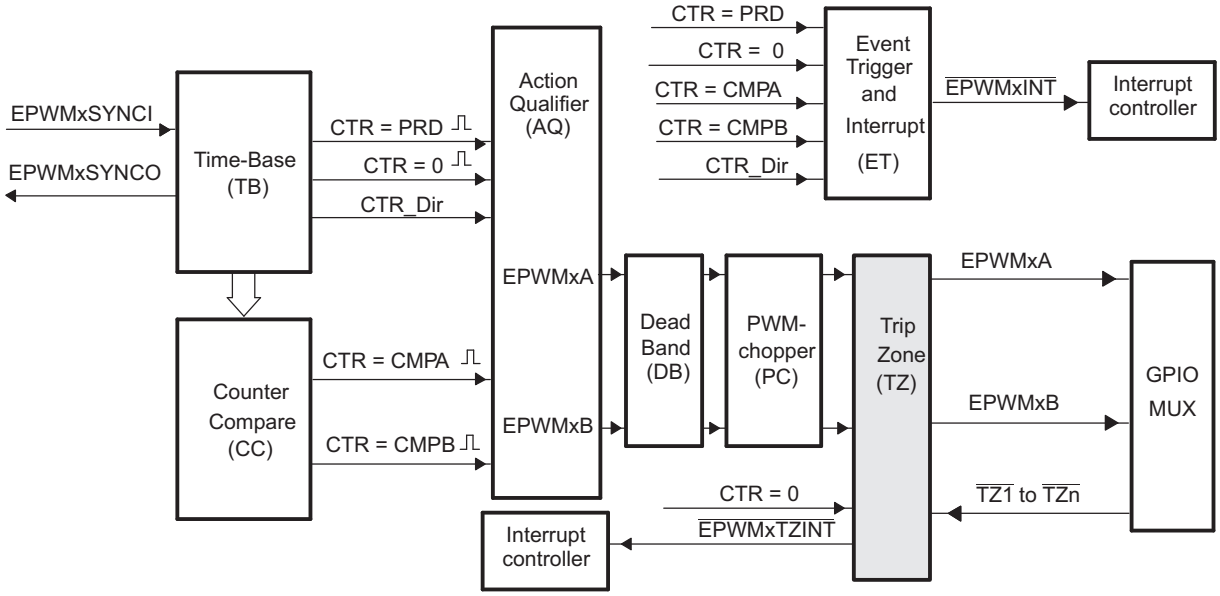


Copyright © 2013–2016, Texas Instruments Incorporated

14.2.8 Trip-Zone (TZ) Submodule

Figure 14-36 shows how the trip-zone (TZ) submodule fits within the ePWM module. Each ePWM module is connected to every $\overline{TZ}$ signal that are sourced from the GPIO MUX. These signals indicates external fault or trip conditions, and the ePWM outputs can be programmed to respond accordingly when faults occur. See your device-specific data manual to determine the number of trip-zone pins available for the device.

Figure 14-36. Trip-Zone Submodule



14.2.8.1 Purpose of the Trip-Zone Submodule

The key functions of the trip-zone submodule are:

- Trip inputs $\overline{TZ1}$ to $\overline{TZn}$ can be flexibly mapped to any ePWM module.
- Upon a fault condition, outputs EPWMxA and EPWMxB can be forced to one of the following:
  - High
  - Low
  - High-impedance
  - No action taken
- Support for one-shot trip (OSHT) for major short circuits or over-current conditions.
- Support for cycle-by-cycle tripping (CBC) for current limiting operation.
- Each trip-zone input pin can be allocated to either one-shot or cycle-by-cycle operation.
- Interrupt generation is possible on any trip-zone pin.
- Software-forced tripping is also supported.
- The trip-zone submodule can be fully bypassed if it is not required.

Copyright © 2013–2016, Texas Instruments Incorporated

14.2.8.2 Controlling and Monitoring the Trip-Zone Submodule

The trip-zone submodule operation is controlled and monitored through the following registers:

Table 14-28. Trip-Zone Submodule Registers

Acronym	Register Description	Address Offset	Shadowed
TZSEL	Trip-Zone Select Register	24h	No
TZCTL	Trip-Zone Control Register	28h	No
TZEINT	Trip-Zone Enable Interrupt Register	2Ah	No
TZFLG	Trip-Zone Flag Register	2Ch	No
TZCLR	Trip-Zone Clear Register	2Eh	No
TZFRC	Trip-Zone Force Register	30h	No

14.2.8.3 Operational Highlights for the Trip-Zone Submodule

The following sections describe the operational highlights and configuration options for the trip-zone submodule.

The trip-zone signals at pin $\overline{TZ1}$ to $\overline{TZn}$ is an active-low input signal. When the pin goes low, it indicates that a trip event has occurred. Each ePWM module can be individually configured to ignore or use each of the trip-zone pins. Which trip-zone pins are used by a particular ePWM module is determined by the TZSEL register for that specific ePWM module. The trip-zone signal may or may not be synchronized to the system clock (SYSCLKOUT). A minimum of 1 SYSCLKOUT low pulse on the $\overline{TZ}$ $\overline{n}$ inputs is sufficient to trigger a fault condition in the ePWM module. The asynchronous trip makes sure that if clocks are missing for any reason, the outputs can still be tripped by a valid event present on the $\overline{TZn}$ inputs.

The $\overline{TZ}$ $\overline{n}$ input can be individually configured to provide either a cycle-by-cycle or one-shot trip event for a ePWM module. The configuration is determined by the TZSEL[CBCn] and TZSEL[OSHTn] bits (where n corresponds to the trip pin) respectively.

- **Cycle-by-Cycle (CBC):** When a cycle-by-cycle trip event occurs, the action specified in the TZCTL register is carried out immediately on the EPWMxA and/or EPWMxB output. Table 14-29 lists the possible actions. In addition, the cycle-by-cycle trip event flag (TZFLG[CBC]) is set and a EPWMxTZINT interrupt is generated if it is enabled in the TZEINT register.

  The specified condition on the pins is automatically cleared when the ePWM time-base counter reaches zero (TBCNT = 0000h) if the trip event is no longer present. Therefore, in this mode, the trip event is cleared or reset every PWM cycle. The TZFLG[CBC] flag bit will remain set until it is manually cleared by writing to the TZCLR[CBC] bit. If the cycle-by-cycle trip event is still present when the TZFLG[CBC] bit is cleared, then it will again be immediately set.
- **One-Shot (OSHT):** When a one-shot trip event occurs, the action specified in the TZCTL register is carried out immediately on the EPWMxA and/or EPWMxB output. Table 14-29 lists the possible actions. In addition, the one-shot trip event flag (TZFLG[OST]) is set and a EPWMxTZINT interrupt is generated if it is enabled in the TZEINT register. The one-shot trip condition must be cleared manually by writing to the TZCLR[OST] bit.

The action taken when a trip event occurs can be configured individually for each of the ePWM output pins by way of the TZCTL[TZA] and TZCTL[TZB] register bits. One of four possible actions, shown in Table 14-29, can be taken on a trip event.

Copyright © 2013–2016, Texas Instruments Incorporated

Table 14-29. Possible Actions On a Trip Event

TZCTL[TZA] and/or TZCTL[TZB]	EPWMxA and/or EPWMxB	Comment
0	High-Impedance	Tripped
1h	Force to High State	Tripped
2h	Force to Low State	Tripped
3h	No Change	Do Nothing. No change is made to the output.

Example 14-2. Trip-Zone Configurations

Scenario A:
A one-shot trip event on $\overline{TZ1}$ pulls both EPWM1A, EPWM1B low and also forces EPWM2A and EPWM2B high.

- Configure the ePWM1 registers as follows:
  - TZSEL[OSHT1] = 1: enables $\overline{TZ}$ as a one-shot event source for ePWM1
  - TZCTL[TZA] = 2: EPWM1A will be forced low on a trip event.
  - TZCTL[TZB] = 2: EPWM1B will be forced low on a trip event.
- Configure the ePWM2 registers as follows:
  - TZSEL[OSHT1] = 1: enables $\overline{TZ}$ as a one-shot event source for ePWM2
  - TZCTL[TZA] = 1: EPWM2A will be forced high on a trip event.
  - TZCTL[TZB] = 1: EPWM2B will be forced high on a trip event.

Scenario B:
A cycle-by-cycle event on $\overline{TZ5}$ pulls both EPWM1A, EPWM1B low.

A one-shot event on $\overline{TZ1}$ or $\overline{TZ6}$ puts EPWM2A into a high impedance state.

- Configure the ePWM1 registers as follows:
  - TZSEL[CBC5] = 1: enables $\overline{TZ5}$ as a one-shot event source for ePWM1
  - TZCTL[TZA] = 2: EPWM1A will be forced low on a trip event.
  - TZCTL[TZB] = 2: EPWM1B will be forced low on a trip event.
- Configure the ePWM2 registers as follows:
  - TZSEL[OSHT1] = 1: enables $\overline{TZ1}$ as a one-shot event source for ePWM2
  - TZSEL[OSHT6] = 1: enables $\overline{TZ6}$ as a one-shot event source for ePWM1
  - TZCTL[TZA] = 0: EPWM1A will be put into a high-impedance state on a trip event.
  - TZCTL[TZB] = 3: EPWM1B will ignore the trip event.

14.2.8.4 Generating Trip Event Interrupts

Figure 14-37 and Figure 14-38 illustrate the trip-zone submodule control and interrupt logic, respectively.

Copyright © 2013–2016, Texas Instruments Incorporated

Figure 14-37. Trip-Zone Submodule Mode Control Logic

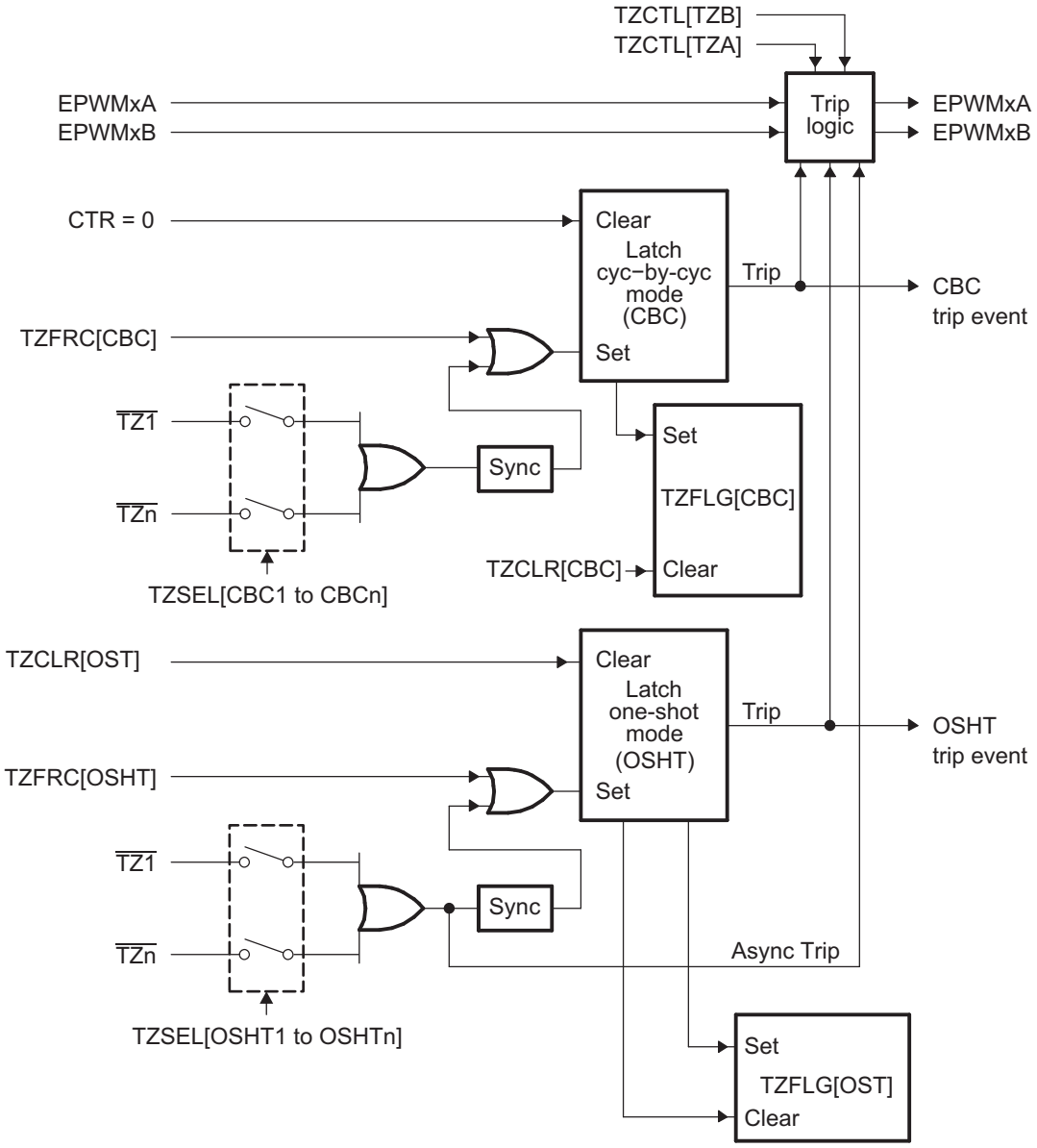
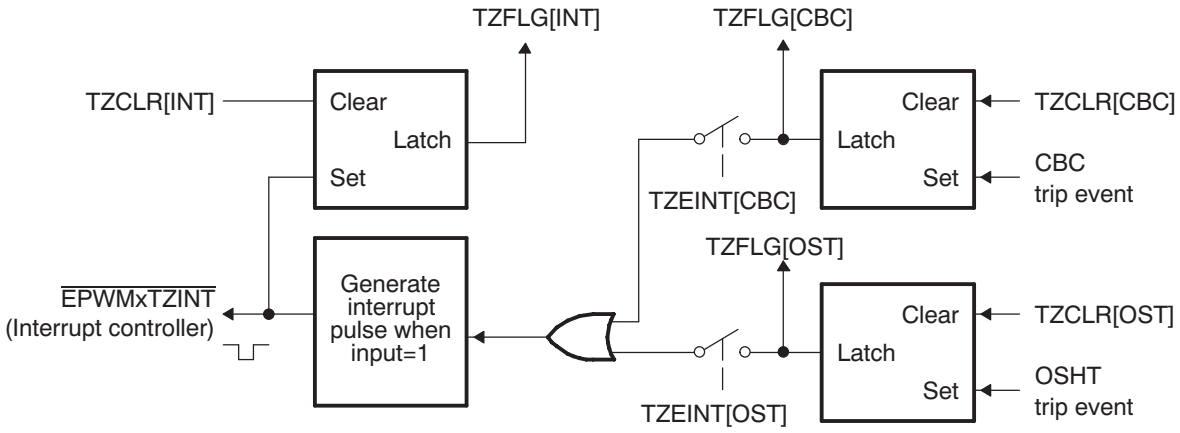


Figure 14-38. Trip-Zone Submodule Interrupt Logic

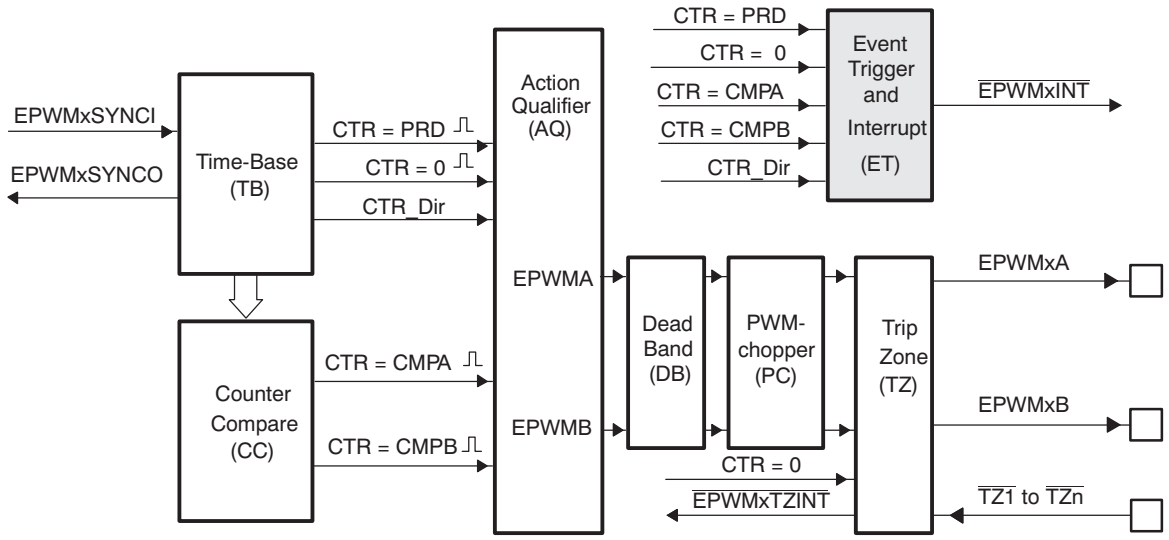


Copyright © 2013–2016, Texas Instruments Incorporated

14.2.9 Event-Trigger (ET) Submodule

Figure 14-39 shows the event-trigger (ET) submodule in the ePWM system. The event-trigger submodule manages the events generated by the time-base submodule and the counter-compare submodule to generate an interrupt to the CPU.

Figure 14-39. Event-Trigger Submodule



14.2.9.1 Purpose of the Event-Trigger Submodule

The key functions of the event-trigger submodule are:

- Receives event inputs generated by the time-base and counter-compare submodules
- Uses the time-base direction information for up/down event qualification
- Uses prescaling logic to issue interrupt requests at:
  - Every event
  - Every second event
  - Every third event
- Provides full visibility of event generation via event counters and flags

14.2.9.2 Controlling and Monitoring the Event-Trigger Submodule

The key registers used to configure the event-trigger submodule are shown in Table 14-30:

Table 14-30. Event-Trigger Submodule Registers

Acronym	Register Description	Address Offset	Shadowed
ETSEL	Event-Trigger Selection Register	32h	No
ETPS	Event-Trigger Prescale Register	34h	No
ETFLG	Event-Trigger Flag Register	36h	No
ETCLR	Event-Trigger Clear Register	38h	No
ETFRC	Event-Trigger Force Register	3Ah	No

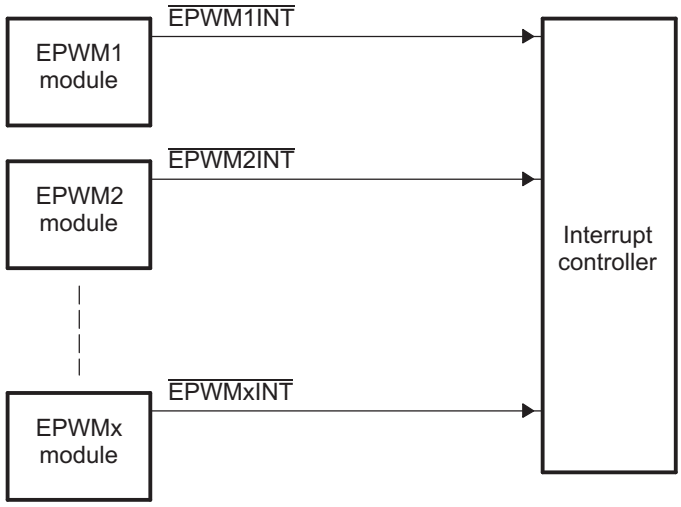
Copyright © 2013–2016, Texas Instruments Incorporated

14.2.9.3 Operational Overview of the Event-Trigger Submodule

The following sections describe the event-trigger submodule's operational highlights.

Each ePWM module has one interrupt request line connected to the interrupt controller as shown in Figure 14-40.

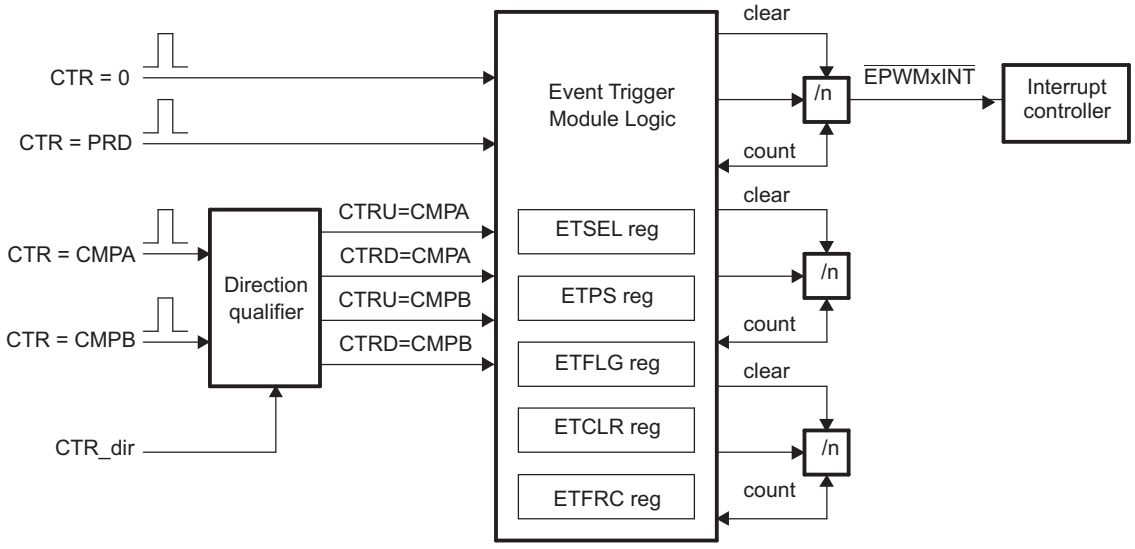
Figure 14-40. Event-Trigger Submodule Inter-Connectivity to Interrupt Controller



The event-trigger submodule monitors various event conditions (the left side inputs to event-trigger submodule shown in Figure 14-41) and can be configured to prescale these events before issuing an Interrupt request. The event-trigger prescaling logic can issue Interrupt requests at:

- Every event
- Every second event
- Every third event

Figure 14-41. Event-Trigger Submodule Showing Event Inputs and Prescaled Outputs



Copyright © 2013–2016, Texas Instruments Incorporated

- ETSEL—This selects which of the possible events will trigger an interrupt.
- ETPS—This programs the event prescaling options previously mentioned.
- ETFLG—These are flag bits indicating status of the selected and prescaled events.
- ETCLR—These bits allow you to clear the flag bits in the ETFLG register via software.
- ETFRC—These bits allow software forcing of an event. Useful for debugging or software intervention.

A more detailed look at how the various register bits interact with the Interrupt is shown in Figure 14-42.

Figure 14-42 shows the event-trigger's interrupt generation logic. The interrupt-period (ETPS[INTPRD]) bits specify the number of events required to cause an interrupt pulse to be generated. The choices available are:

- Do not generate an interrupt
- Generate an interrupt on every event
- Generate an interrupt on every second event
- Generate an interrupt on every third event

An interrupt cannot be generated on every fourth or more events.

Which event can cause an interrupt is configured by the interrupt selection (ETSEL[INTSEL]) bits. The event can be one of the following:

- Time-base counter equal to zero (TBCNT = 0000h).
- Time-base counter equal to period (TBCNT = TBPRD).
- Time-base counter equal to the compare A register (CMPA) when the timer is incrementing.
- Time-base counter equal to the compare A register (CMPA) when the timer is decrementing.
- Time-base counter equal to the compare B register (CMPB) when the timer is incrementing.
- Time-base counter equal to the compare B register (CMPB) when the timer is decrementing.

The number of events that have occurred can be read from the interrupt event counter (ETPS[INTCNT]) register bits. That is, when the specified event occurs the ETPS[INTCNT] bits are incremented until they reach the value specified by ETPS[INTPRD]. When ETPS[INTCNT] = ETPS[INTPRD] the counter stops counting and its output is set. The counter is only cleared when an interrupt is sent to the interrupt controller.

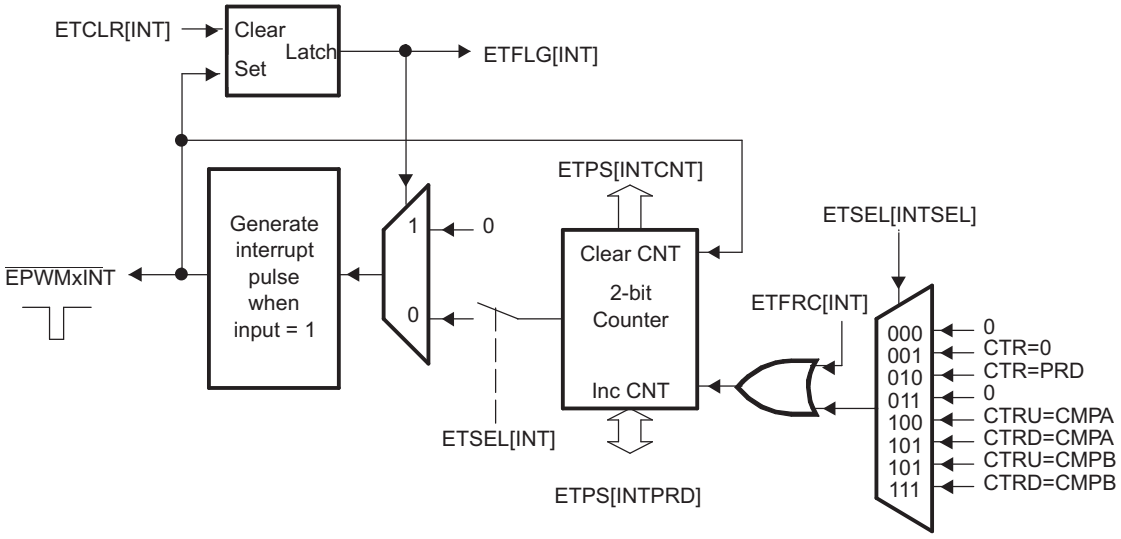
When ETPS[INTCNT] reaches ETPS[INTPRD], one of the following behaviors will occur:

- If interrupts are enabled, ETSEL[INTEN] = 1 and the interrupt flag is clear, ETFLG[INT] = 0, then an interrupt pulse is generated and the interrupt flag is set, ETFLG[INT] = 1, and the event counter is cleared ETPS[INTCNT] = 0. The counter will begin counting events again.
- If interrupts are disabled, ETSEL[INTEN] = 0, or the interrupt flag is set, ETFLG[INT] = 1, the counter stops counting events when it reaches the period value ETPS[INTCNT] = ETPS[INTPRD].
- If interrupts are enabled, but the interrupt flag is already set, then the counter will hold its output high until the ENTFLG[INT] flag is cleared. This allows for one interrupt to be pending while one is serviced.

Writing to the INTPRD bits will automatically clear the counter INTCNT = 0 and the counter output will be reset (so no interrupts are generated). Writing a 1 to the ETFRC[INT] bit will increment the event counter INTCNT. The counter will behave as described above when INTCNT = INTPRD. When INTPRD = 0, the counter is disabled and hence no events will be detected and the ETFRC[INT] bit is also ignored.

Copyright © 2013–2016, Texas Instruments Incorporated

Figure 14-42. Event-Trigger Interrupt Generator

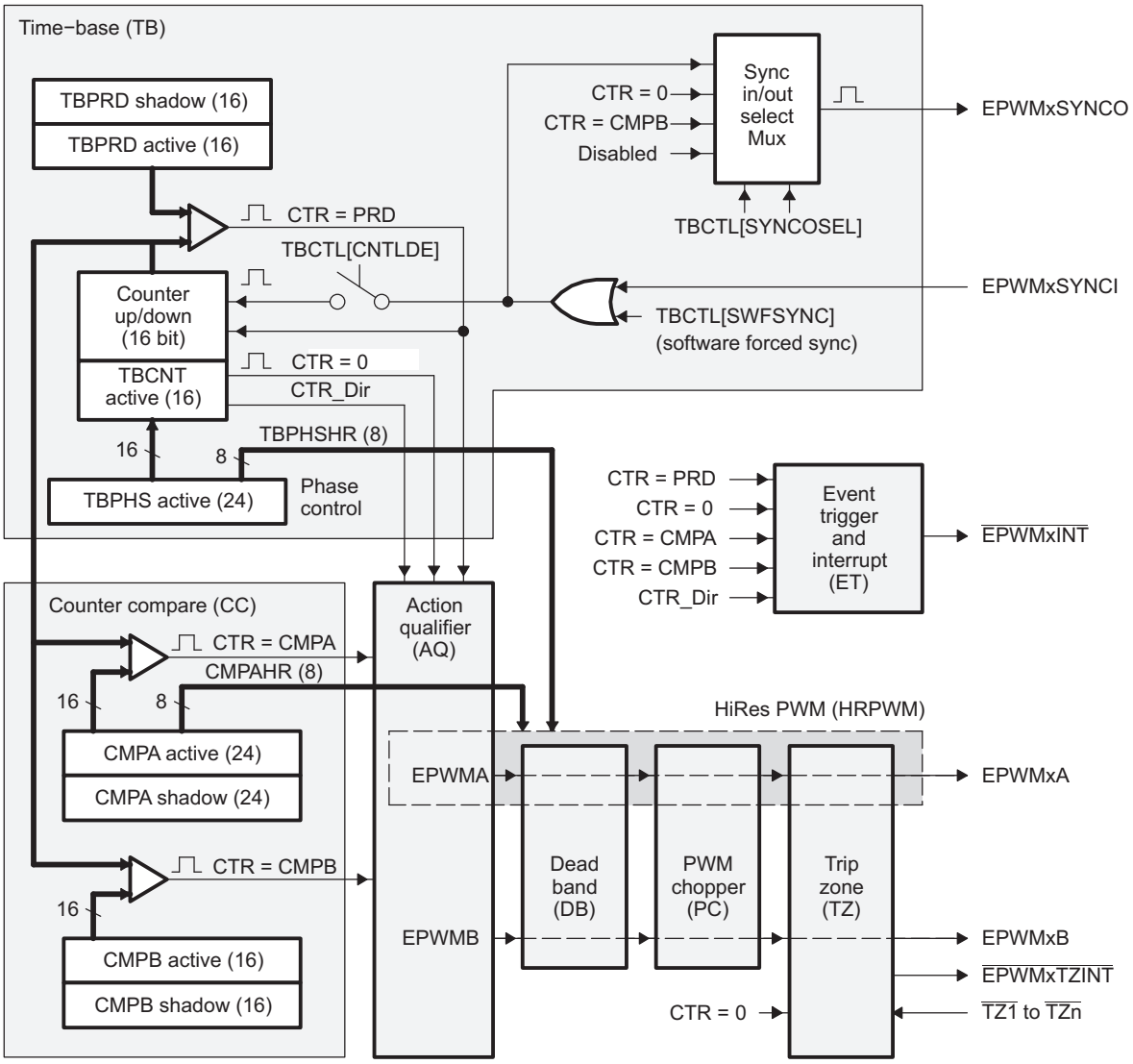


Copyright © 2013–2016, Texas Instruments Incorporated

14.2.10 High-Resolution PWM (HRPWM) Submodule

Figure 14-43 shows the high-resolution PWM (HRPWM) submodule in the ePWM system. Some devices include the high-resolution PWM submodule, see your device-specific data manual to determine which ePWM instances include this feature.

Figure 14-43. HRPWM System Interface



Copyright © 2013–2016, Texas Instruments Incorporated

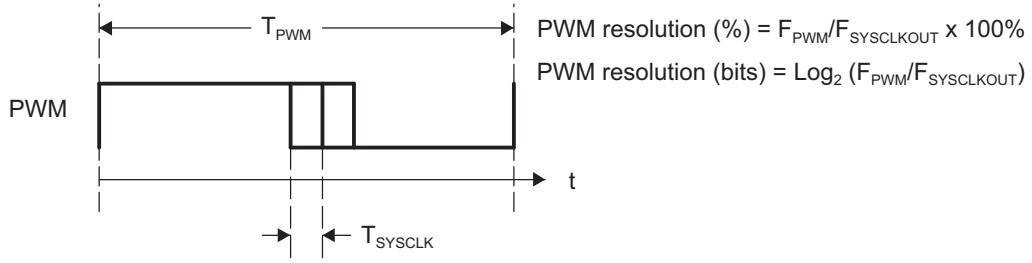
14.2.10.1 Purpose of the High-Resolution PWM Submodule

The enhanced high-resolution pulse-width modulator (eHRPWM) extends the time resolution capabilities of the conventionally derived digital pulse-width modulator (PWM). HRPWM is typically used when PWM resolution falls below ~9-10 bits. The key features of HRPWM are:

- Extended time resolution capability
- Used in both duty cycle and phase-shift control methods
- Finer time granularity control or edge positioning using extensions to the Compare A and Phase registers
- Implemented using the A signal path of PWM, that is, on the EPWMxA output. EPWMxB output has conventional PWM capabilities

The ePWM peripheral is used to perform a function that is mathematically equivalent to a digital-to-analog converter (DAC). As shown in Figure 14-44, the effective resolution for conventionally generated PWM is a function of PWM frequency (or period) and system clock frequency.

Figure 14-44. Resolution Calculations for Conventionally Generated PWM



If the required PWM operating frequency does not offer sufficient resolution in PWM mode, you may want to consider HRPWM. As an example of improved performance offered by HRPWM, Table 14-31 shows resolution in bits for various PWM frequencies. Table 14-31 values assume a MEP step size of 180 ps. See your device-specific data manual for typical and maximum performance specifications for the MEP.

Table 14-31. Resolution for PWM and HRPWM

	Regular Resolution (PWM)		High Resolution (HRPWM)	
PWM Frequency (kHz)	Bits	%	Bits	%
20	12.3	0.0	18.1	0.000
50	11.0	0.0	16.8	0.001
100	10.0	0.1	15.8	0.002
150	9.4	0.2	15.2	0.003
200	9.0	0.2	14.8	0.004
250	8.6	0.3	14.4	0.005
500	7.6	0.5	13.8	0.007
1000	6.6	1.0	12.4	0.018
1500	6.1	1.5	11.9	0.027
2000	5.6	2.0	11.4	0.036

Although each application may differ, typical low-frequency PWM operation (below 250 kHz) may not require HRPWM. HRPWM capability is most useful for high-frequency PWM requirements of power conversion topologies such as:

- Single-phase buck, boost, and flyback
- Multi-phase buck, boost, and flyback
- Phase-shifted full bridge
- Direct modulation of D-Class power amplifiers

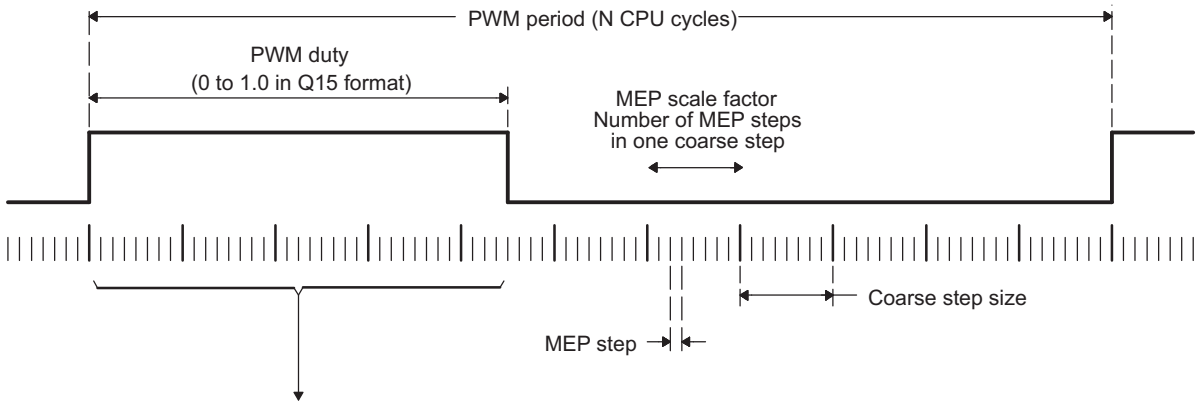
Copyright © 2013–2016, Texas Instruments Incorporated

14.2.10.2 Architecture of the High-Resolution PWM Submodule

The HRPWM is based on micro edge positioner (MEP) technology. MEP logic is capable of positioning an edge very finely by sub-dividing one coarse system clock of a conventional PWM generator. The time step accuracy is on the order of 150 ps. The HRPWM also has a self-check software diagnostics mode to check if the MEP logic is running optimally, under all operating conditions.

Figure 14-45 shows the relationship between one coarse system clock and edge position in terms of MEP steps, which are controlled via an 8-bit field in the Compare A extension register (CMPAHR).

Figure 14-45. Operating Logic Using MEP



Number of coarse steps = integer(PWMduty * PWMperiod)
Number of MEP steps = fraction(PWMduty * PWMperiod) * (MEPScaleFactor)

16−bit CMPA register value	= number of coarse steps
16−bit CMPAHR register value	= (number of MEP steps) << 8 + 0x180 (rounding) (A)

A For MEP range and rounding adjustment.

To generate an HRPWM waveform, configure the TBM, CCM, and AQM registers as you would to generate a conventional PWM of a given frequency and polarity. The HRPWM works together with the TBM, CCM, and AQM registers to extend edge resolution, and should be configured accordingly. Although many programming combinations are possible, only a few are needed and practical.

14.2.10.3 Controlling and Monitoring the High-Resolution PWM Submodule

The MEP of the HRPWM is controlled by two extension registers, each 8-bits wide. These two HRPWM registers are concatenated with the 16-bit TBPHS and CMPA registers used to control PWM operation.

- TBPHSHR - Time-Base Phase High-Resolution Register
- CMPAHR - Counter-Compare A High-Resolution Register

Table 14-32 lists the registers used to control and monitor the high-resolution PWM submodule.

Table 14-32. HRPWM Submodule Registers

Acronym	Register Description	Address Offset	Shadowed
TBPHSHR	Extension Register for HRPWM Phase	4h	No
CMPAHR	Extension Register for HRPWM Duty	10h	Yes
HRCNFG	HRPWM Configuration Register	1040h	No

Copyright © 2013–2016, Texas Instruments Incorporated

14.2.10.4 Configuring the High-Resolution PWM Submodule

Once the ePWM has been configured to provide conventional PWM of a given frequency and polarity, the HRPWM is configured by programming the HRCNFG register located at offset address 1040h. This register provides configuration options for the following key operating modes:

- **Edge Mode:** The MEP can be programmed to provide precise position control on the rising edge (RE), falling edge (FE), or both edges (BE) at the same time. FE and RE are used for power topologies requiring duty cycle control, while BE is used for topologies requiring phase shifting, for example, phase shifted full bridge.
- **Control Mode:** The MEP is programmed to be controlled either from the CMPAHR register (duty cycle control) or the TBPHSHR register (phase control). RE or FE control mode should be used with CMPAHR register. BE control mode should be used with TBPHSHR register.
- **Shadow Mode:** This mode provides the same shadowing (double buffering) option as in regular PWM mode. This option is valid only when operating from the CMPAHR register and should be chosen to be the same as the regular load option for the CMPA register. If TBPHSHR is used, then this option has no effect.

14.2.10.5 Operational Highlights for the High-Resolution PWM Submodule

The MEP logic is capable of placing an edge in one of 255 (8 bits) discrete time steps, each of which has a time resolution on the order of 150 ps. The MEP works with the TBM and CCM registers to be certain that time steps are optimally applied and that edge placement accuracy is maintained over a wide range of PWM frequencies, system clock frequencies and other operating conditions. Table 14-33 shows the typical range of operating frequencies supported by the HRPWM.

Table 14-33. Relationship Between MEP Steps, PWM Frequency and Resolution

System (MHz)	MEP Steps Per SYSCLKOUT(1) (2) (3)	PWM Minimum (Hz)(4)	PWM Maximum (MHz)	Resolution at Maximum (Bits)(5)
50.0	111	763	2.50	11.1
60.0	93	916	3.00	10.9
70.0	79	1068	3.50	10.6
80.0	69	1221	4.00	10.4
90.0	62	1373	4.50	10.3
100.0	56	1526	5.00	10.1

(1) System frequency = SYSCLKOUT, that is, CPU clock. TBCLK = SYSCLKOUT
(2) Table data based on a MEP time resolution of 180 ps (this is an example value)
(3) MEP steps applied = $T_{SYSCLKOUT}$/180 ps in this example.
(4) PWM minimum frequency is based on a maximum period value, TBPRD = 65 535. PWM mode is asymmetrical up-count.
(5) Resolution in bits is given for the maximum PWM frequency stated.

Copyright © 2013–2016, Texas Instruments Incorporated

14.2.10.5.1 Edge Positioning

In a typical power control loop (switch modes, digital motor control (DMC), uninterruptible power supply (UPS)), a digital controller (PID, 2pole/2zero, lag/lead, etc.) issues a duty command, usually expressed in a per unit or percentage terms.

In the following example, assume that for a particular operating point, the demanded duty cycle is 0.405 or 40.5% on-time and the required converter PWM frequency is 1.25 MHz. In conventional PWM generation with a system clock of 100 MHz, the duty cycle choices are in the vicinity of 40.5%. In Figure 14-46, a compare value of 32 counts (duty = 40%) is the closest to 40.5% that you can attain. This is equivalent to an edge position of 320 ns instead of the desired 324 ns. This data is shown in Table 14-34.

By utilizing the MEP, you can achieve an edge position much closer to the desired point of 324 ns. Table 14-34 shows that in addition to the CMPA value, 22 steps of the MEP (CMPAHR register) will position the edge at 323.96 ns, resulting in almost zero error. In this example, it is assumed that the MEP has a step resolution of 180 ns.

Figure 14-46. Required PWM Waveform for a Requested Duty = 40.5%

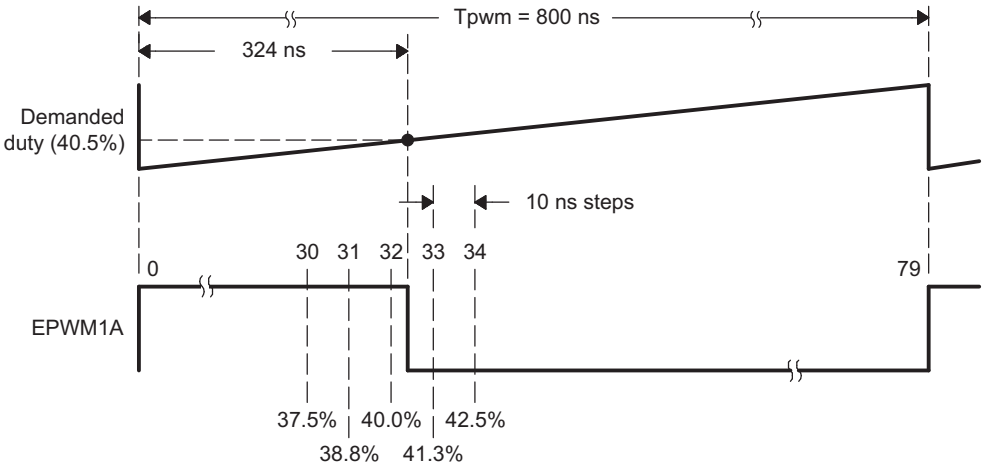


Table 14-34. CMPA vs Duty (left), and [CMPA:CMPAHR] vs Duty (right)

CMPA (count)[1] [2] [3]	DUTY (%)	High Time (ns)	CMPA (count)	CMPAHR (count)	Duty (%)	High Time (ns)
28	35.0	280	32	18	40.405	323.24
29	36.3	290	32	19	40.428	323.42
30	37.5	300	32	20	40.450	323.60
31	38.8	310	32	21	40.473	323.78
32	40.0	320	32	22	40.495	323.96
33	41.3	330	32	23	40.518	324.14
34	42.5	340	32	24	40.540	324.32
			32	25	40.563	324.50
Required			32	26	40.585	324.68
32.40	40.5	324	32	27	40.608	324.86

(1) System clock, SYSCLKOUT and TBCLK = 100 MHz, 10 ns
(2) For a PWM Period register value of 80 counts, PWM Period = 80 × 10 ns = 800 ns, PWM frequency = 1/800 ns = 1.25 MHz
(3) Assumed MEP step size for the above example = 180 ps

Copyright © 2013–2016, Texas Instruments Incorporated

14.2.10.5.2 Scaling Considerations

The mechanics of how to position an edge precisely in time has been demonstrated using the resources of the standard (CMPA) and MEP (CMPAHR) registers. In a practical application, however, it is necessary to seamlessly provide the CPU a mapping function from a per-unit (fractional) duty cycle to a final integer (non-fractional) representation that is written to the [CMPA:CMPAHR] register combination.

To do this, first examine the scaling or mapping steps involved. It is common in control software to express duty cycle in a per-unit or percentage basis. This has the advantage of performing all needed math calculations without concern for the final absolute duty cycle, expressed in clock counts or high time in ns. Furthermore, it makes the code more transportable across multiple converter types running different PWM frequencies.

To implement the mapping scheme, a two-step scaling procedure is required.

Assumptions for this example:

System clock, SYSCLKOUT	=	10 ns (100 MHz)
PWM frequency	=	1.25 MHz (1/800 ns)
Required PWM duty cycle, **PWMDuty**	=	0.405 (40.5%)
PWM period in terms of coarse steps, **PWMperiod** (800 ns/10 ns)	=	80
Number of MEP steps per coarse step at 180 ps (10 ns/180 ps), **MEP_SF**	=	55
Value to keep CMPAHR within the range of 1-255 and fractional rounding constant (default value)	=	180h

Step 1: Percentage Integer Duty value conversion for CMPA register

CMPA register value	=	int(**PWMDuty** × **PWMperiod**); int means integer part
	=	int(0.405 × 80)
	=	int(32.4)
CMPA register value	=	32 (20h)

Step 2: Fractional value conversion for CMPAHR register

CMPAHR register value	=	(frac(**PWMDuty** × **PWMperiod**) × **MEP_SF**) << 8) + 180h; frac means fractional part
	=	(frac(32.4) × 55 <<8) + 180h; Shift is to move the value as CMPAHR high byte
	=	((0.4 × 55) <<8) + 180h
	=	(22 <<8) + 180h
	=	22 × 256 + 180h ; Shifting left by 8 is the same multiplying by 256.
	=	5632 + 180h
	=	1600h + 180h
CMPAHR value	=	1780h; CMPAHR value = 1700h, lower 8 bits will be ignored by hardware.

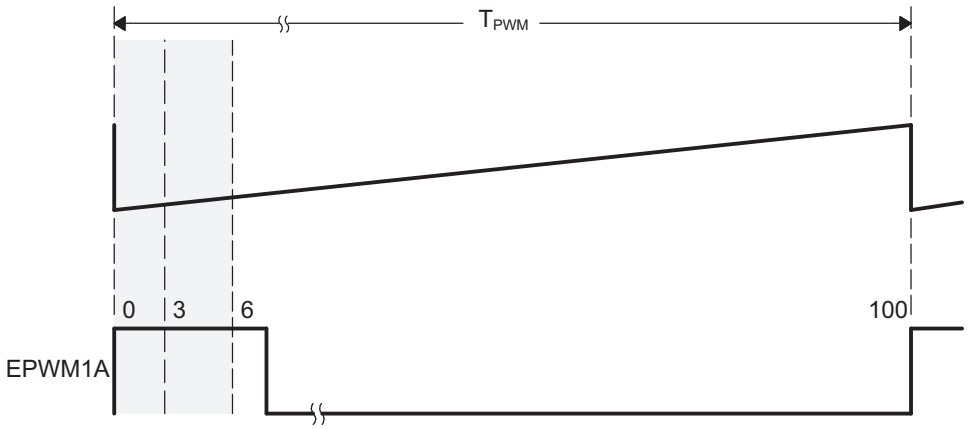
Copyright © 2013–2016, Texas Instruments Incorporated

14.2.10.5.3 Duty Cycle Range Limitation

In high resolution mode, the MEP is not active for 100% of the PWM period. It becomes operational 3 SYSCLK cycles after the period starts.

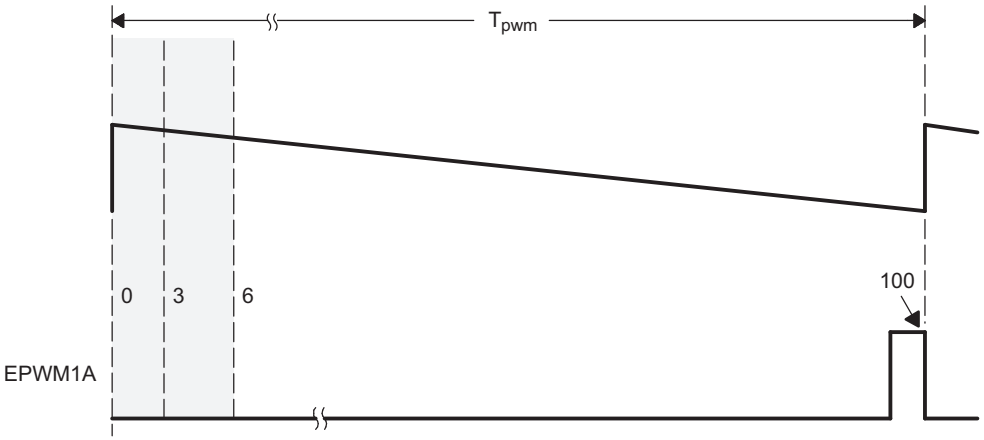
Duty cycle range limitations are illustrated in Figure 14-47. This limitation imposes a lower duty cycle limit on the MEP. For example, precision edge control is not available all the way down to 0% duty cycle. Although for the first 3 or 6 cycles, the HRPWM capabilities are not available, regular PWM duty control is still fully operational down to 0% duty. In most applications this should not be an issue as the controller regulation point is usually not designed to be close to 0% duty cycle.

Figure 14-47. Low % Duty Cycle Range Limitation Example When PWM Frequency = 1 MHz



If the application demands HRPWM operation in the low percent duty cycle region, then the HRPWM can be configured to operate in count-down mode with the rising edge position (REP) controlled by the MEP. This is illustrated in Figure 14-48. In this case low percent duty limitation is no longer an issue.

Figure 14-48. High % Duty Cycle Range Limitation Example when PWM Frequency = 1 MHz



Copyright © 2013–2016, Texas Instruments Incorporated

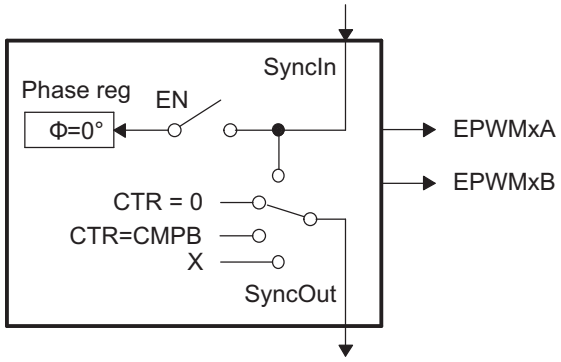
14.3 Applications to Power Topologies

An ePWM module has all the local resources necessary to operate completely as a standalone module or to operate in synchronization with other identical ePWM modules.

14.3.1 Overview of Multiple Modules

Previously in this user's guide, all discussions have described the operation of a single module. To facilitate the understanding of multiple modules working together in a system, the ePWM module described in reference is represented by the more simplified block diagram shown in Figure 14-49. This simplified ePWM block shows only the key resources needed to explain how a multiswitch power topology is controlled with multiple ePWM modules working together.

Figure 14-49. Simplified ePWM Module



Copyright © 2013–2016, Texas Instruments Incorporated

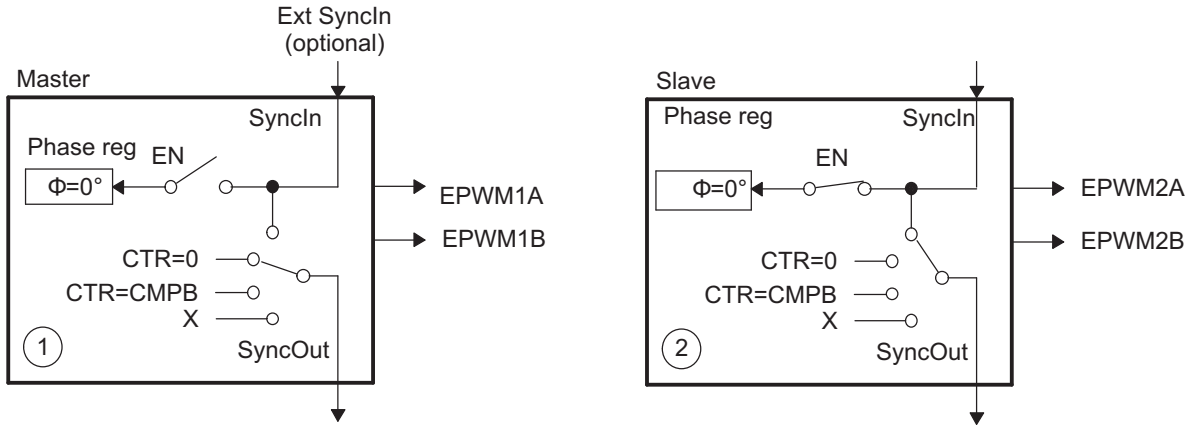
14.3.2 Key Configuration Capabilities

The key configuration choices available to each module are as follows:

- Options for SyncIn
  - Load own counter with phase register on an incoming sync strobe—enable (EN) switch closed
  - Do nothing or ignore incoming sync strobe—enable switch open
  - Sync flow-through - SyncOut connected to SyncIn
  - Master mode, provides a sync at PWM boundaries—SyncOut connected to CTR = PRD
  - Master mode, provides a sync at any programmable point in time—SyncOut connected to CTR = CMPB
  - Module is in standalone mode and provides No sync to other modules—SyncOut connected to X (disabled)
- Options for SyncOut
  - Sync flow-through - SyncOut connected to SyncIn
  - Master mode, provides a sync at PWM boundaries—SyncOut connected to CTR = PRD
  - Master mode, provides a sync at any programmable point in time—SyncOut connected to CTR = CMPB
  - Module is in standalone mode and provides No sync to other modules—SyncOut connected to X (disabled)

For each choice of SyncOut, a module may also choose to load its own counter with a new phase value on a SyncIn strobe input or choose to ignore it, i.e., via the enable switch. Although various combinations are possible, the two most common—master module and slave module modes—are shown in Figure 14-50.

Figure 14-50. EPWM1 Configured as a Typical Master, EPWM2 Configured as a Slave

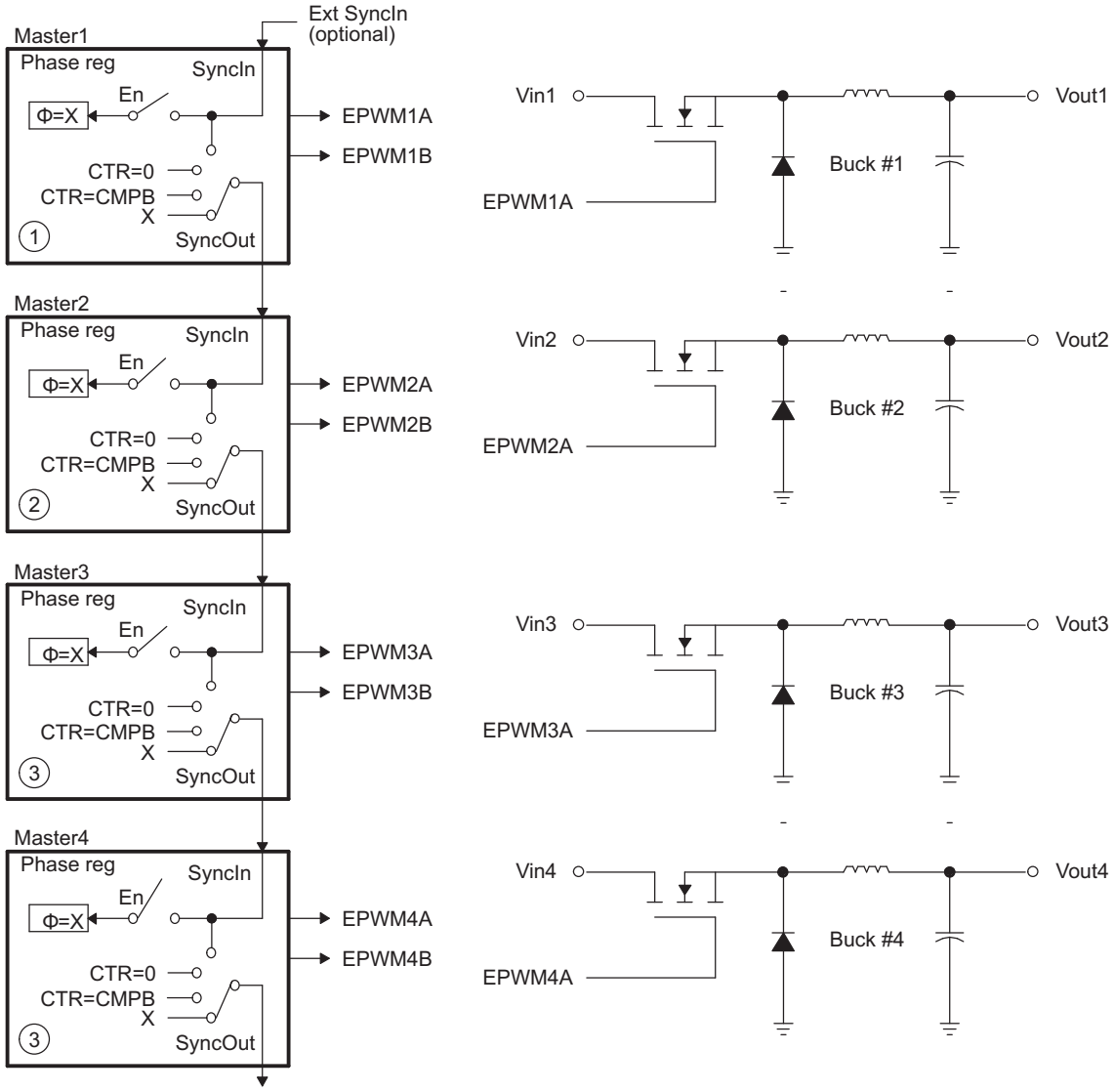


Copyright © 2013–2016, Texas Instruments Incorporated

14.3.3 Controlling Multiple Buck Converters With Independent Frequencies

One of the simplest power converter topologies is the buck. A single ePWM module configured as a master can control two buck stages with the same PWM frequency. If independent frequency control is required for each buck converter, then one ePWM module must be allocated for each converter stage. Figure 14-51 shows four buck stages, each running at independent frequencies. In this case, all four ePWM modules are configured as Masters and no synchronization is used. Figure 14-52 shows the waveforms generated by the setup shown in Figure 14-51; note that only three waveforms are shown, although there are four stages.

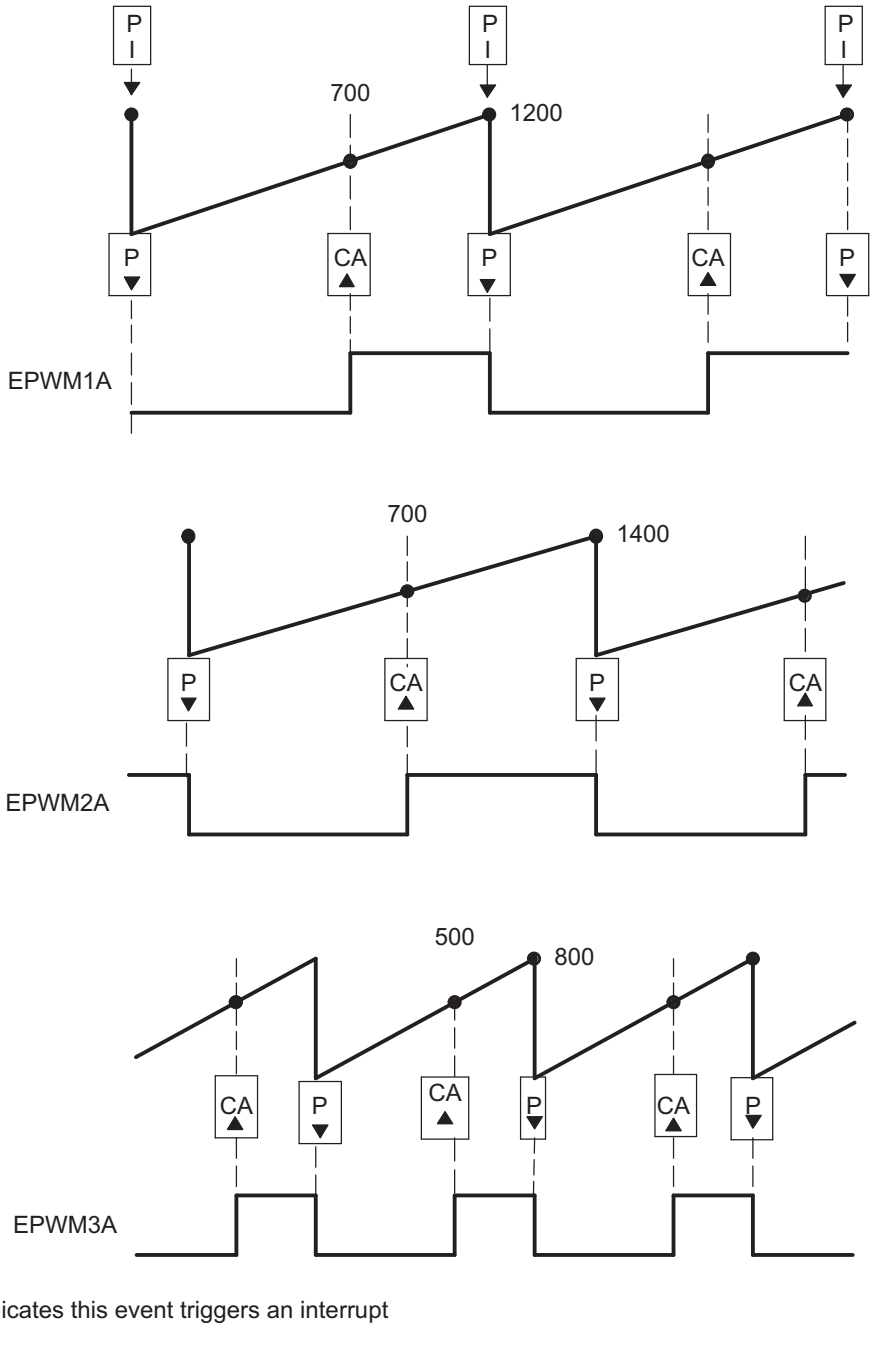
Figure 14-51. Control of Four Buck Stages. (Note: $F_{PWM1} \neq F_{PWM2} \neq F_{PWM3} \neq F_{PWM4}$)



NOTE: Θ = X indicates value in phase register is a "don't care"

Copyright © 2013–2016, Texas Instruments Incorporated

Figure 14-52. Buck Waveforms for Figure 14-51 (Note: Only three bucks shown here)



Copyright © 2013–2016, Texas Instruments Incorporated

Table 14-35. EPWM1 Initialization for Figure 14-52

Register	Bit	Value	Comments
TBPRD	TBPRD	1200 (4B0h)	Period = 1201 TBCLK counts
TBPHS	TBPHS	0	Clear Phase Register to 0
TBCTL	CTRMODE	TB_UP	
	PHSEN	TB_DISABLE	Phase loading disabled
	PRDLD	TB_SHADOW	
	SYNCOSEL	TB_SYNC_DISABLE	
CMPCTL	SHDWAMODE	CC_SHADOW	
	SHDWBMODE	CC_SHADOW	
	LOADAMODE	CC_CTR_ZERO	Load on CTR = 0
	LOADBMODE	CC_CTR_ZERO	Load on CTR = 0
AQCTLA	PRD	AQ_CLEAR	
	CAU	AQ_SET	

Table 14-36. EPWM2 Initialization for Figure 14-52

Register	Bit	Value	Comments
TBPRD	TBPRD	1400 (578h)	Period = 1401 TBCLK counts
TBPHS	TBPHS	0	Clear Phase Register to 0
TBCTL	CTRMODE	TB_UP	
	PHSEN	TB_DISABLE	Phase loading disabled
	PRDLD	TB_SHADOW	
	SYNCOSEL	TB_SYNC_DISABLE	
CMPCTL	SHDWAMODE	CC_SHADOW	
	SHDWBMODE	CC_SHADOW	
	LOADAMODE	CC_CTR_ZERO	Load on CTR = 0
	LOADBMODE	CC_CTR_ZERO	Load on CTR = 0
AQCTLA	PRD	AQ_CLEAR	
	CAU	AQ_SET	

Table 14-37. EPWM3 Initialization for Figure 14-52

Register	Bit	Value	Comments
TBPRD	TBPRD	800 (320h)	Period = 801 TBCLK counts
TBPHS	TBPHS	0	Clear Phase Register to 0
TBCTL	CTRMODE	TB_UP	
	PHSEN	TB_DISABLE	Phase loading disabled
	PRDLD	TB_SHADOW	
	SYNCOSEL	TB_SYNC_DISABLE	
CMPCTL	SHDWAMODE	CC_SHADOW	
	SHDWBMODE	CC_SHADOW	
	LOADAMODE	CC_CTR_ZERO	Load on CTR = 0
	LOADBMODE	CC_CTR_ZERO	Load on CTR = 0
AQCTLA	PRD	AQ_CLEAR	
	CAU	AQ_SET	

Copyright © 2013–2016, Texas Instruments Incorporated

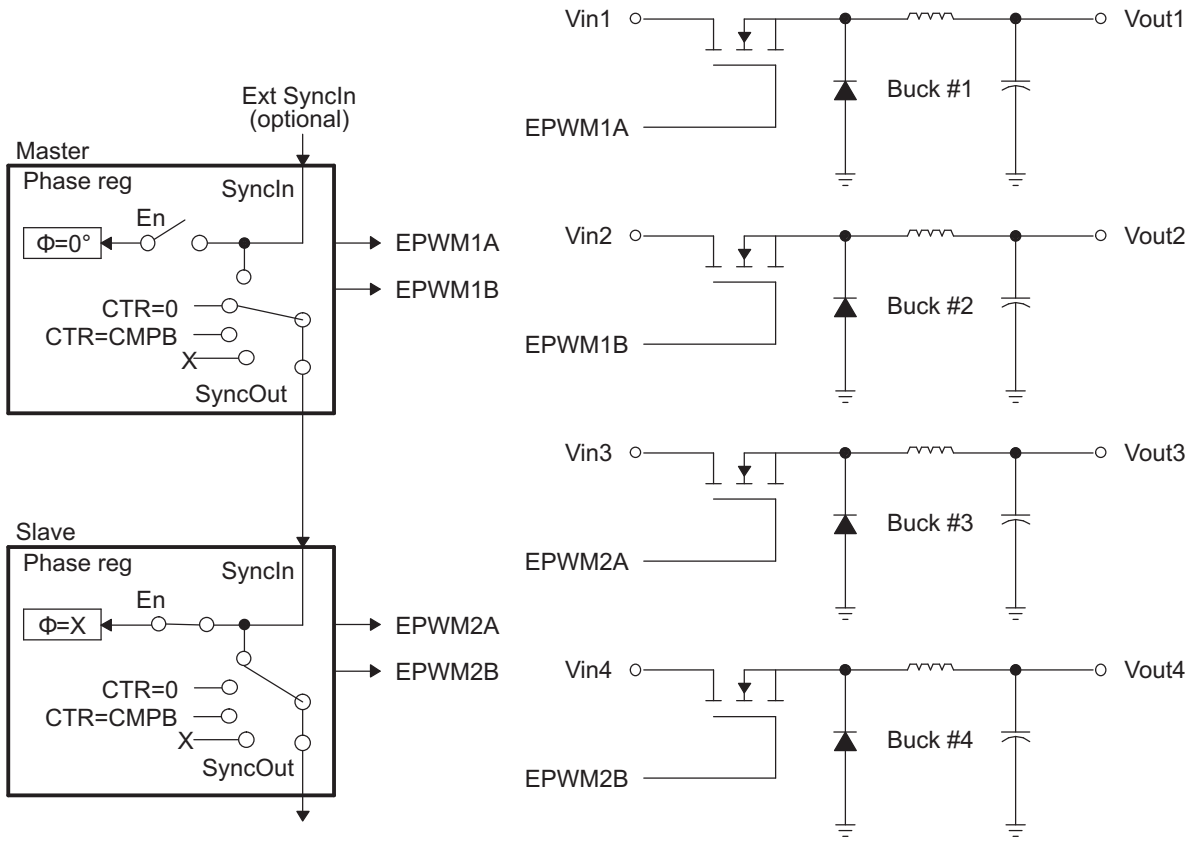
Example 14-3. Configuration for Example in Figure 14-52

```
// Run Time (Note: Example execution of one run-time instance)
//=====================================================
    EPwm1Regs.CMPA.half.CMPA = 700;              // adjust duty for output EPWM1A
    EPwm2Regs.CMPA.half.CMPA = 700;              // adjust duty for output EPWM2A
    EPwm3Regs.CMPA.half.CMPA = 500;              // adjust duty for output EPWM3A
```

14.3.4 Controlling Multiple Buck Converters With Same Frequencies

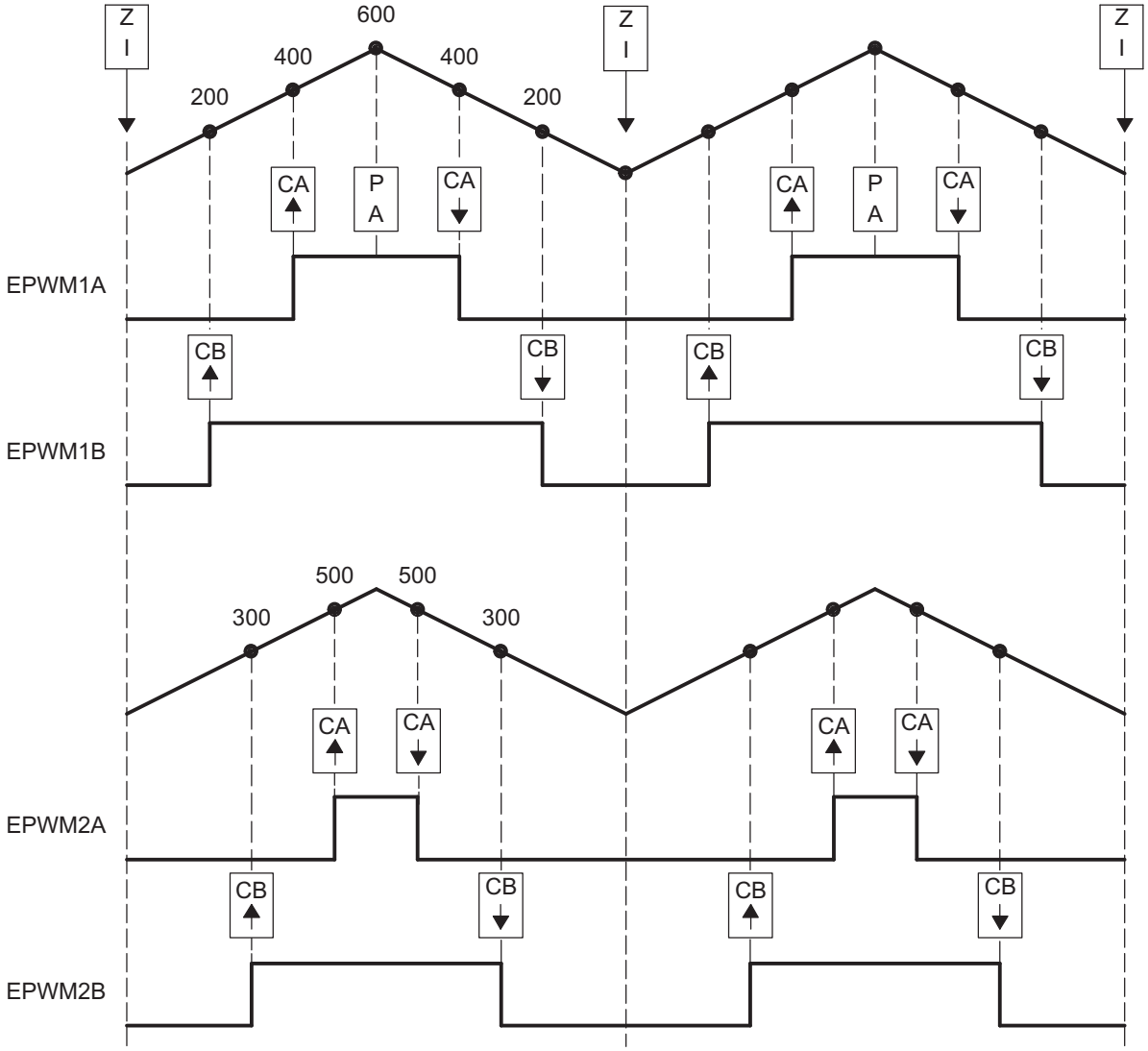
If synchronization is a requirement, ePWM module 2 can be configured as a slave and can operate at integer multiple (N) frequencies of module 1. The sync signal from master to slave ensures these modules remain locked. Figure 14-53 shows such a configuration; Figure 14-54 shows the waveforms generated by the configuration.

Figure 14-53. Control of Four Buck Stages. (Note: $F_{PWM2} = N \times F_{PWM1}$)



Copyright © 2013–2016, Texas Instruments Incorporated

Figure 14-54. Buck Waveforms for Figure 14-53 (Note: $F_{PWM2} = F_{PWM1}$)



Copyright © 2013–2016, Texas Instruments Incorporated

Table 14-38. EPWM1 Initialization for Figure 14-53

Register	Bit	Value	Comments
TBPRD	TBPRD	600 (258h)	Period = 1200 TBCLK counts
TBPHS	TBPHS	0	Clear Phase Register to 0
TBCTL	CTRMODE	TB_UPDOWN	
	PHSEN	TB_DISABLE	Phase loading disabled
	PRDLD	TB_SHADOW	
	SYNCOSEL	TB_CTR_ZERO	Sync down-stream module
CMPCTL	SHDWAMODE	CC_SHADOW	
	SHDWBMODE	CC_SHADOW	
	LOADAMODE	CC_CTR_ZERO	Load on CTR = 0
	LOADBMODE	CC_CTR_ZERO	Load on CTR = 0
AQCTLA	CAU	AQ_SET	Set actions for EPWM1A
	CAD	AQ_CLEAR	
AQCTLB	CBU	AQ_SET	Set actions for EPWM1B
	CBD	AQ_CLEAR	

Table 14-39. EPWM2 Initialization for Figure 14-53

Register	Bit	Value	Comments
TBPRD	TBPRD	600 (258h)	Period = 1200 TBCLK counts
TBPHS	TBPHS	0	Clear Phase Register to 0
TBCTL	CTRMODE	TB_UPDOWN	
	PHSEN	TB_ENABLE	Phase loading enabled
	PRDLD	TB_SHADOW	
	SYNCOSEL	TB_SYNC_IN	Sync flow-through
CMPCTL	SHDWAMODE	CC_SHADOW	
	SHDWBMODE	CC_SHADOW	
	LOADAMODE	CC_CTR_ZERO	Load on CTR = 0
	LOADBMODE	CC_CTR_ZERO	Load on CTR = 0
AQCTLA	CAU	AQ_SET	Set actions for EPWM2A
	CAD	AQ_CLEAR	
AQCTLB	CBU	AQ_SET	Set actions for EPWM2B
	CBD	AQ_CLEAR	

Example 14-4. Code Snippet for Configuration in Figure 14-53

```
// Run Time (Note: Example execution of one run-time instance)
//=========================================================
     EPwm1Regs.CMPA.half.CMPA = 400;       // adjust duty for output EPWM1A
     EPwm1Regs.CMPB = 200;                 // adjust duty for output EPWM1B
     EPwm2Regs.CMPA.half.CMPA = 500;       // adjust duty for output EPWM2A
     EPwm2Regs.CMPB = 300;                 // adjust duty for output EPWM2B
```

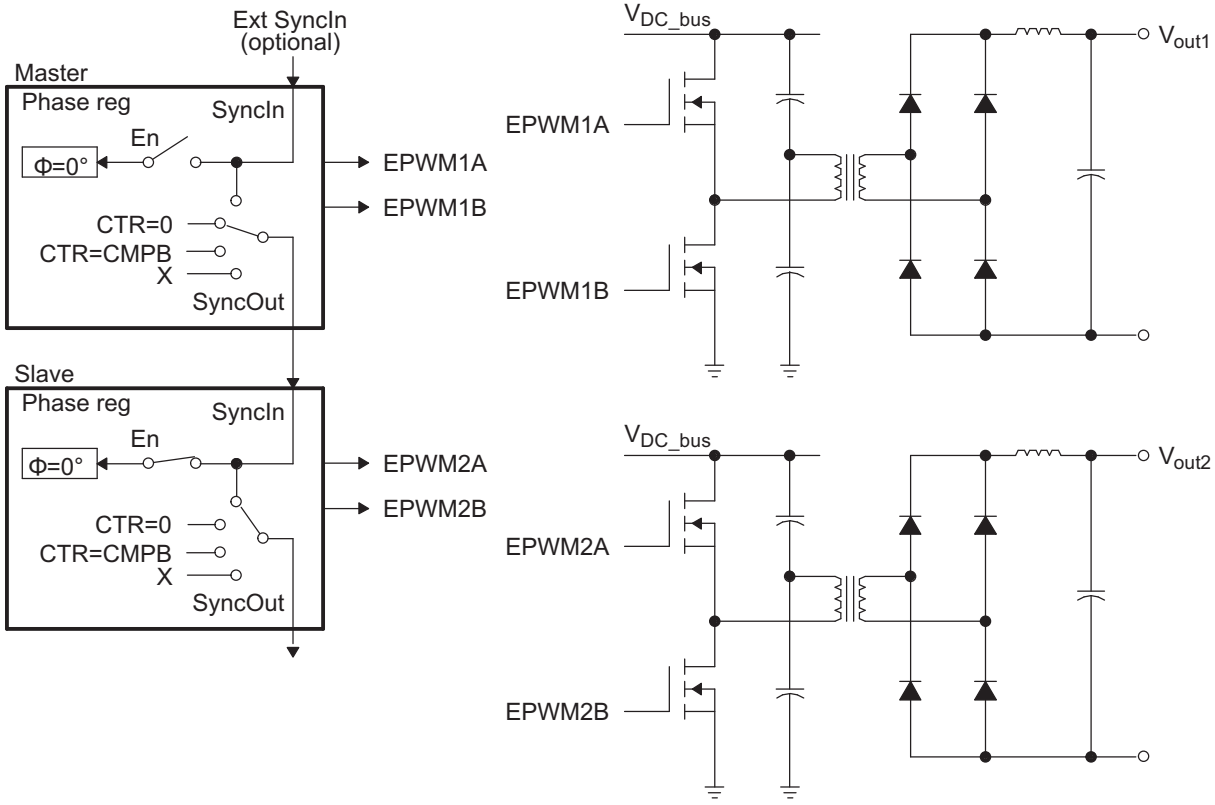

Copyright © 2013–2016, Texas Instruments Incorporated

14.3.5 Controlling Multiple Half H-Bridge (HHB) Converters

Topologies that require control of multiple switching elements can also be addressed with these same ePWM modules. It is possible to control a Half-H bridge stage with a single ePWM module. This control can be extended to multiple stages. Figure 14-55 shows control of two synchronized Half-H bridge stages where stage 2 can operate at integer multiple (N) frequencies of stage 1. Figure 14-56 shows the waveforms generated by the configuration shown in Figure 14-55.

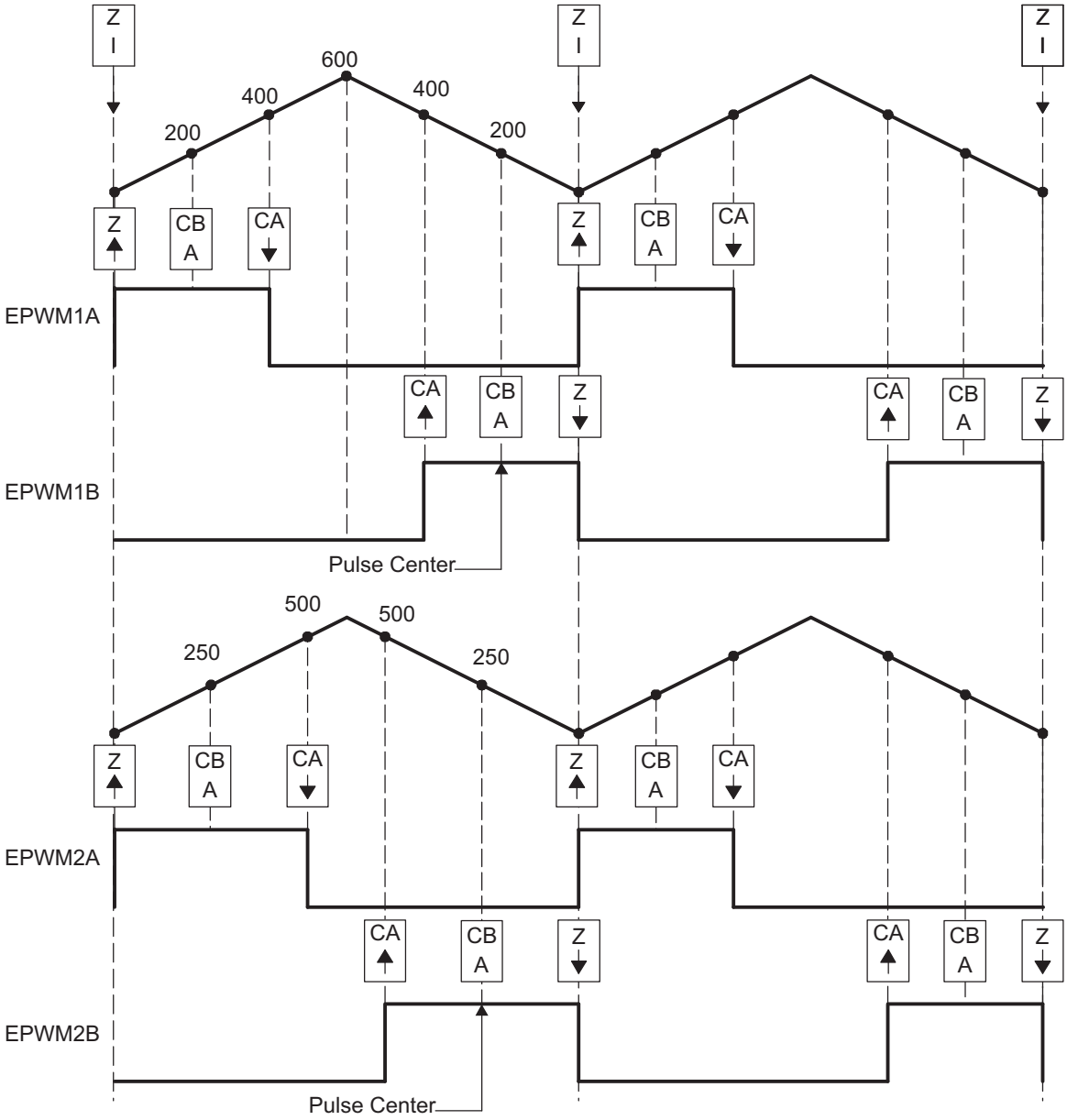
Module 2 (slave) is configured for Sync flow-through; if required, this configuration allows for a third Half-H bridge to be controlled by PWM module 3 and also, most importantly, to remain in synchronization with master module 1.

Figure 14-55. Control of Two Half-H Bridge Stages ($F_{PWM2} = N \times F_{PWM1}$)



Copyright © 2013–2016, Texas Instruments Incorporated

Figure 14-56. Half-H Bridge Waveforms for Figure 14-55 (Note: $F_{PWM2} = F_{PWM1}$)



Copyright © 2013–2016, Texas Instruments Incorporated

Table 14-40. EPWM1 Initialization for Figure 14-55

Register	Bit	Value	Comments
TBPRD	TBPRD	600 (258h)	Period = 1200 TBCLK counts
TBPHS	TBPHS	0	Clear Phase Register to 0
TBCTL	CTRMODE	TB_UPDOWN	
	PHSEN	TB_DISABLE	Phase loading disabled
	PRDLD	TB_SHADOW	
	SYNCOSEL	TB_CTR_ZERO	Sync down-stream module
CMPCTL	SHDWAMODE	CC_SHADOW	
	SHDWBMODE	CC_SHADOW	
	LOADAMODE	CC_CTR_ZERO	Load on CTR = 0
	LOADBMODE	CC_CTR_ZERO	Load on CTR = 0
AQCTLA	ZRO	AQ_SET	Set actions for EPWM1A
	CAU	AQ_CLEAR	
AQCTLB	ZRO	AQ_CLEAR	Set actions for EPWM1B
	CAD	AQ_SET	

Table 14-41. EPWM2 Initialization for Figure 14-55

Register	Bit	Value	Comments
TBPRD	TBPRD	600 (258h)	Period = 1200 TBCLK counts
TBPHS	TBPHS	0	Clear Phase Register to 0
TBCTL	CTRMODE	TB_UPDOWN	
	PHSEN	TB_ENABLE	Phase loading enabled
	PRDLD	TB_SHADOW	
	SYNCOSEL	TB_SYNC_IN	Sync flow-through
CMPCTL	SHDWAMODE	CC_SHADOW	
	SHDWBMODE	CC_SHADOW	
	LOADAMODE	CC_CTR_ZERO	Load on CTR = 0
	LOADBMODE	CC_CTR_ZERO	Load on CTR = 0
AQCTLA	ZRO	AQ_SET	Set actions for EPWM2A
	CAU	AQ_CLEAR	
AQCTLB	ZRO	AQ_CLEAR	Set actions for EPWM2B
	CAD	AQ_SET	

Example 14-5. Code Snippet for Configuration in Figure 14-55

```
// Run Time (Note: Example execution of one run-time instance)
//=========================================================
     EPwm1Regs.CMPA.half.CMPA = 400; // adjust duty for output EPWM1A
     EPwm1Regs.CMPB = 200;           // adjust duty for output EPWM1B
     EPwm2Regs.CMPA.half.CMPA = 500; // adjust duty for output EPWM2A
     EPwm2Regs.CMPB = 250;           // adjust duty for output EPWM2B
```

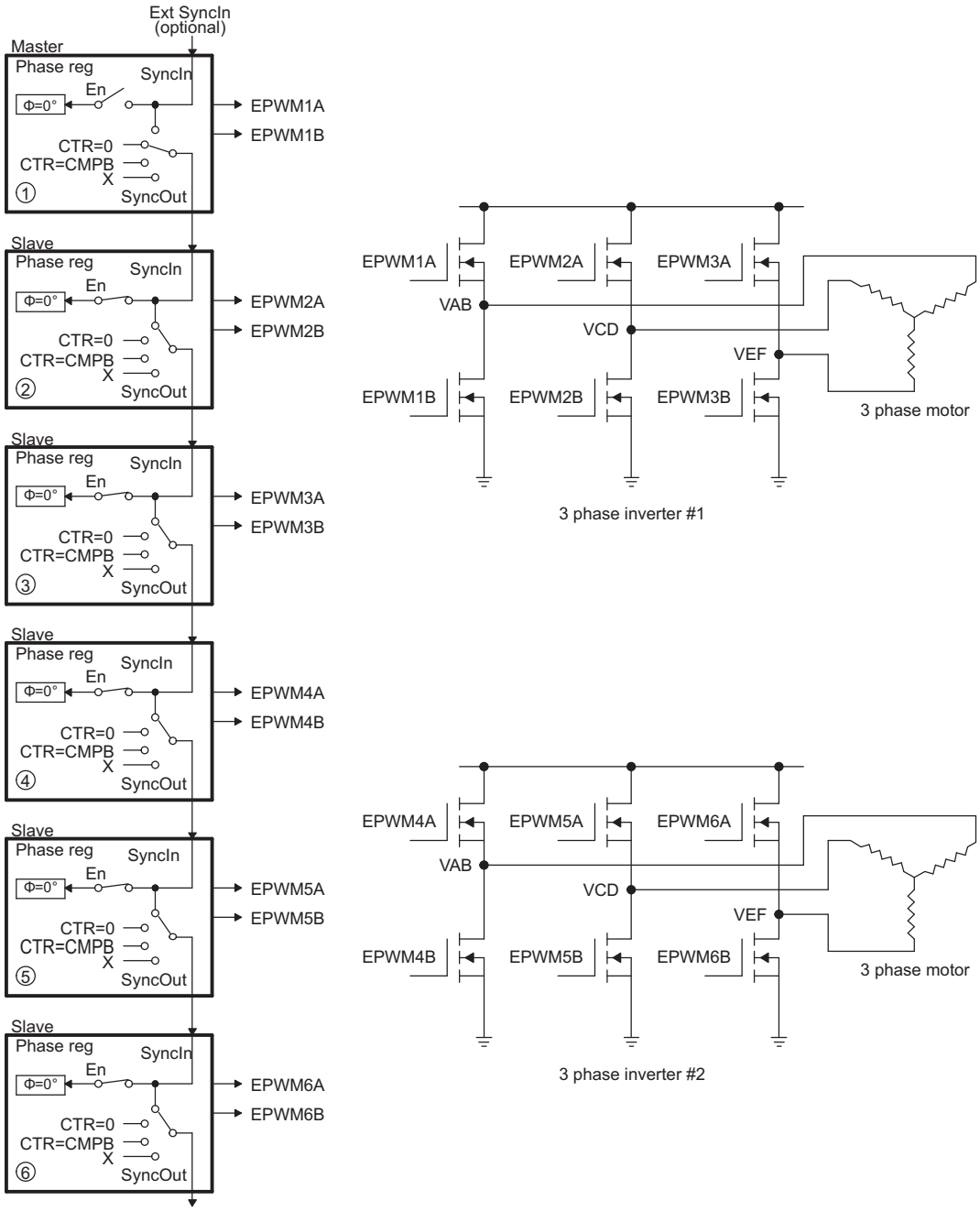

Copyright © 2013–2016, Texas Instruments Incorporated

14.3.6 Controlling Dual 3-Phase Inverters for Motors (ACI and PMSM)

The idea of multiple modules controlling a single power stage can be extended to the 3-phase Inverter case. In such a case, six switching elements can be controlled using three PWM modules, one for each leg of the inverter. Each leg must switch at the same frequency and all legs must be synchronized. A master + two slaves configuration can easily address this requirement. Figure 14-57 shows how six PWM modules can control two independent 3-phase Inverters; each running a motor.

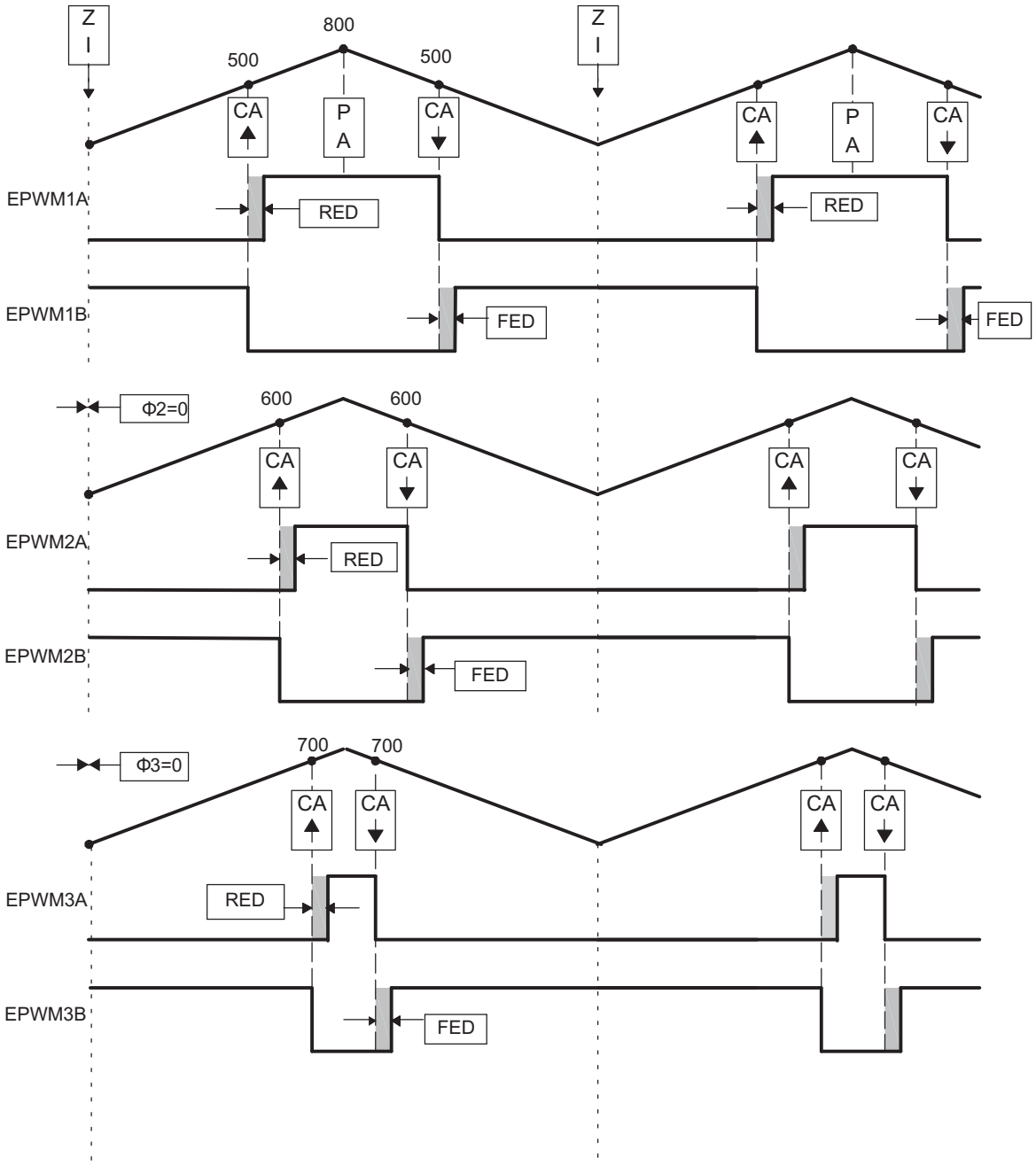
As in the cases shown in the previous sections, we have a choice of running each inverter at a different frequency (module 1 and module 4 are masters as in Figure 14-57), or both inverters can be synchronized by using one master (module 1) and five slaves. In this case, the frequency of modules 4, 5, and 6 (all equal) can be integer multiples of the frequency for modules 1, 2, 3 (also all equal).

Figure 14-57. Control of Dual 3-Phase Inverter Stages as Is Commonly Used in Motor Control



Copyright © 2013–2016, Texas Instruments Incorporated

Figure 14-58. 3-Phase Inverter Waveforms for Figure 14-57 (Only One Inverter Shown)



Copyright © 2013–2016, Texas Instruments Incorporated

Table 14-42. EPWM1 Initialization for Figure 14-57

Register	Bit	Value	Comments
TBPRD	TBPRD	800 (320h)	Period = 1600 TBCLK counts
TBPHS	TBPHS	0	Clear Phase Register to 0
TBCTL	CTRMODE	TB_UPDOWN	
	PHSEN	TB_DISABLE	Phase loading disabled
	PRDLD	TB_SHADOW	
	SYNCOSEL	TB_CTR_ZERO	Sync down-stream module
CMPCTL	SHDWAMODE	CC_SHADOW	
	SHDWBMODE	CC_SHADOW	
	LOADAMODE	CC_CTR_ZERO	Load on CTR = 0
	LOADBMODE	CC_CTR_ZERO	Load on CTR = 0
AQCTLA	CAU	AQ_SET	Set actions for EPWM1A
	CAD	AQ_CLEAR	
DBCTL	MODE	DB_FULL_ENABLE	Enable Dead-band module
	POLSEL	DB_ACTV_HIC	Active Hi complementary
DBFED	DBFED	50	FED = 50 TBCLKs
	DBRED	50	RED = 50 TBCLKs

Table 14-43. EPWM2 Initialization for Figure 14-57

Register	Bit	Value	Comments
TBPRD	TBPRD	800 (320h)	Period = 1600 TBCLK counts
TBPHS	TBPHS	0	Clear Phase Register to 0
TBCTL	CTRMODE	TB_UPDOWN	
	PHSEN	TB_ENABLE	Slave module
	PRDLD	TB_SHADOW	
	SYNCOSEL	TB_SYNC_IN	Sync flow-through
CMPCTL	SHDWAMODE	CC_SHADOW	
	SHDWBMODE	CC_SHADOW	
	LOADAMODE	CC_CTR_ZERO	Load on CTR = 0
	LOADBMODE	CC_CTR_ZERO	Load on CTR = 0
AQCTLA	CAU	AQ_SET	Set actions for EPWM2A
	CAD	AQ_CLEAR	
DBCTL	MODE	DB_FULL_ENABLE	Enable Dead-band module
	POLSEL	DB_ACTV_HIC	Active Hi complementary
DBFED	DBFED	50	FED = 50 TBCLKs
	DBRED	50	RED = 50 TBCLKs

Copyright © 2013–2016, Texas Instruments Incorporated

Table 14-44. EPWM3 Initialization for Figure 14-57

Register	Bit	Value	Comments
TBPRD	TBPRD	800 (320h)	Period = 1600 TBCLK counts
TBPHS	TBPHS	0	Clear Phase Register to 0
TBCTL	CTRMODE	TB_UPDOWN	
	PHSEN	TB_ENABLE	Slave module
	PRDLD	TB_SHADOW	
	SYNCOSEL	TB_SYNC_IN	Sync flow-through
CMPCTL	SHDWAMODE	CC_SHADOW	
	SHDWBMODE	CC_SHADOW	
	LOADAMODE	CC_CTR_ZERO	Load on CTR = 0
	LOADBMODE	CC_CTR_ZERO	Load on CTR = 0
AQCTLA	CAU	AQ_SET	Set actions for EPWM3A
	CAD	AQ_CLEAR	
DBCTL	MODE	DB_FULL_ENABLE	Enable Dead-band module
	POLSEL	DB_ACTV_HIC	Active Hi complementary
DBFED	DBFED	50	FED = 50 TBCLKs
	DBRED	50	RED = 50 TBCLKs

Example 14-6. Code Snippet for Configuration in Figure 14-57

```
// Run Time (Note: Example execution of one run-time instance)
//=====================================================
EPwm1Regs.CMPA.half.CMPA = 500; // adjust duty for output EPWM1A
EPwm2Regs.CMPA.half.CMPA = 600; // adjust duty for output EPWM2A
EPwm3Regs.CMPA.half.CMPA = 700; // adjust duty for output EPWM3A
```


Copyright © 2013–2016, Texas Instruments Incorporated

14.3.7 Practical Applications Using Phase Control Between PWM Modules

So far, none of the examples have made use of the phase register (TBPHS). It has either been set to zero or its value has been a don't care. However, by programming appropriate values into TBPHS, multiple PWM modules can address another class of power topologies that rely on phase relationship between legs (or stages) for correct operation. As described in the TB module section, a PWM module can be configured to allow a SyncIn pulse to cause the TBPHS register to be loaded into the TBCNT register. To illustrate this concept, Figure 14-59 shows a master and slave module with a phase relationship of 120°, that is, the slave leads the master.

Figure 14-59. Configuring Two PWM Modules for Phase Control

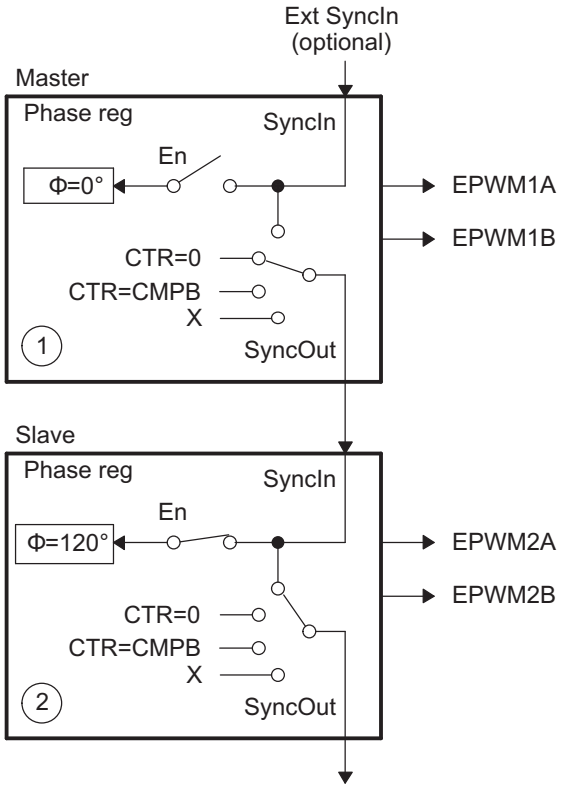
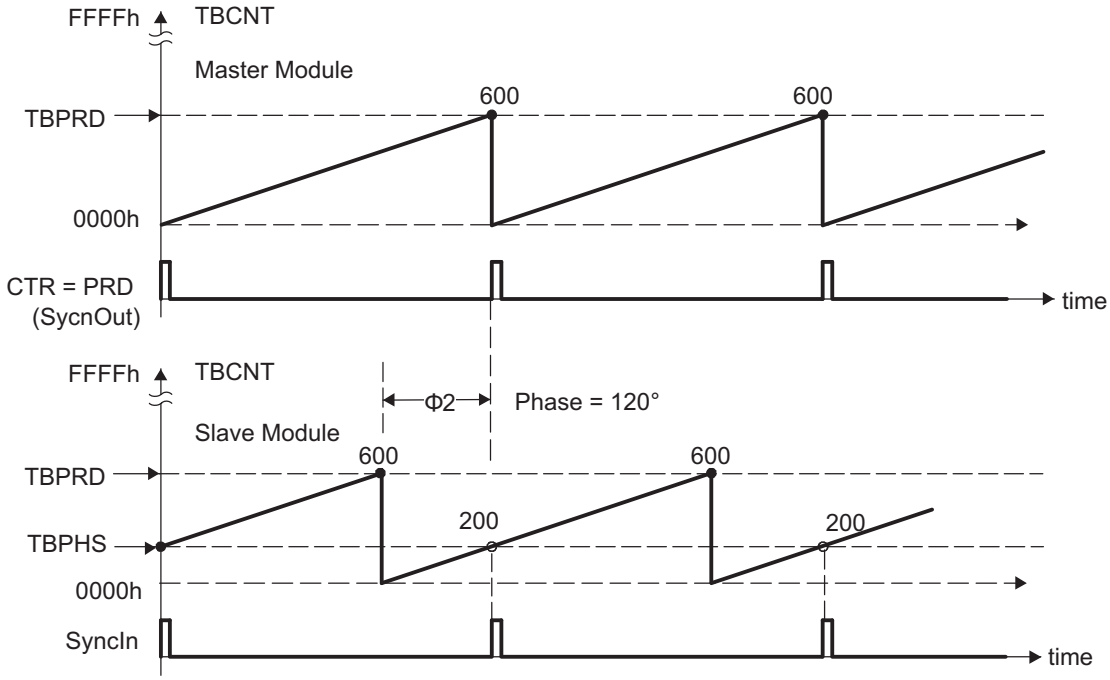


Figure 14-60 shows the associated timing waveforms for this configuration. Here, TBPRD = 600 for both master and slave. For the slave, TBPHS = 200 (200/600 × 360° = 120°). Whenever the master generates a SyncIn pulse (CTR = PRD), the value of TBPHS = 200 is loaded into the slave TBCNT register so the slave time-base is always leading the master's time-base by 120°.

Copyright © 2013–2016, Texas Instruments Incorporated

Figure 14-60. Timing Waveforms Associated With Phase Control Between 2 Modules



14.3.8 Controlling a 3-Phase Interleaved DC/DC Converter

A popular power topology that makes use of phase-offset between modules is shown in Figure 14-61. This system uses three PWM modules, with module 1 configured as the master. To work, the phase relationship between adjacent modules must be F = 120°. This is achieved by setting the slave TBPHS registers 2 and 3 with values of 1/3 and 2/3 of the period value, respectively. For example, if the period register is loaded with a value of 600 counts, then TBPHS (slave 2) = 200 and TBPHS (slave 3) = 400. Both slave modules are synchronized to the master 1 module.

This concept can be extended to four or more phases, by setting the TBPHS values appropriately. The following formula gives the TBPHS values for N phases:

TBPHS(N,M) = (TBPRD/N) × (M - 1)

Where:

N = number of phases

M = PWM module number

For example, for the 3-phase case (N = 3), TBPRD = 600,

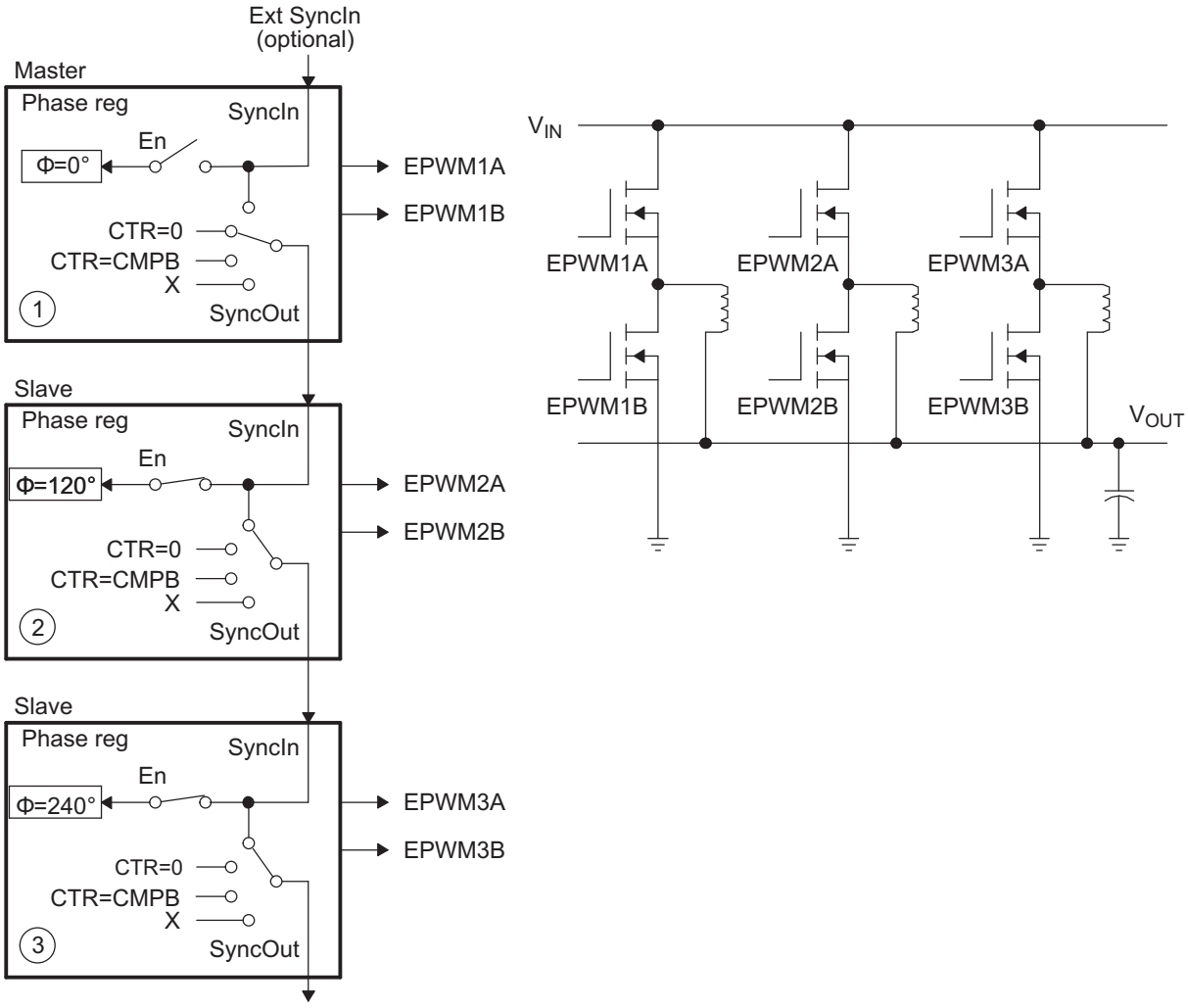
TBPHS(3,2) = (600/3) × (2 - 1) = 200 × 1 = 200 (Phase value for Slave module 2)

TBPHS(3,3) = (600/3) × (3 - 1) = 200 × 2 = 400 (Phase value for Slave module 3)

Figure 14-62 shows the waveforms for the configuration in Figure 14-61.

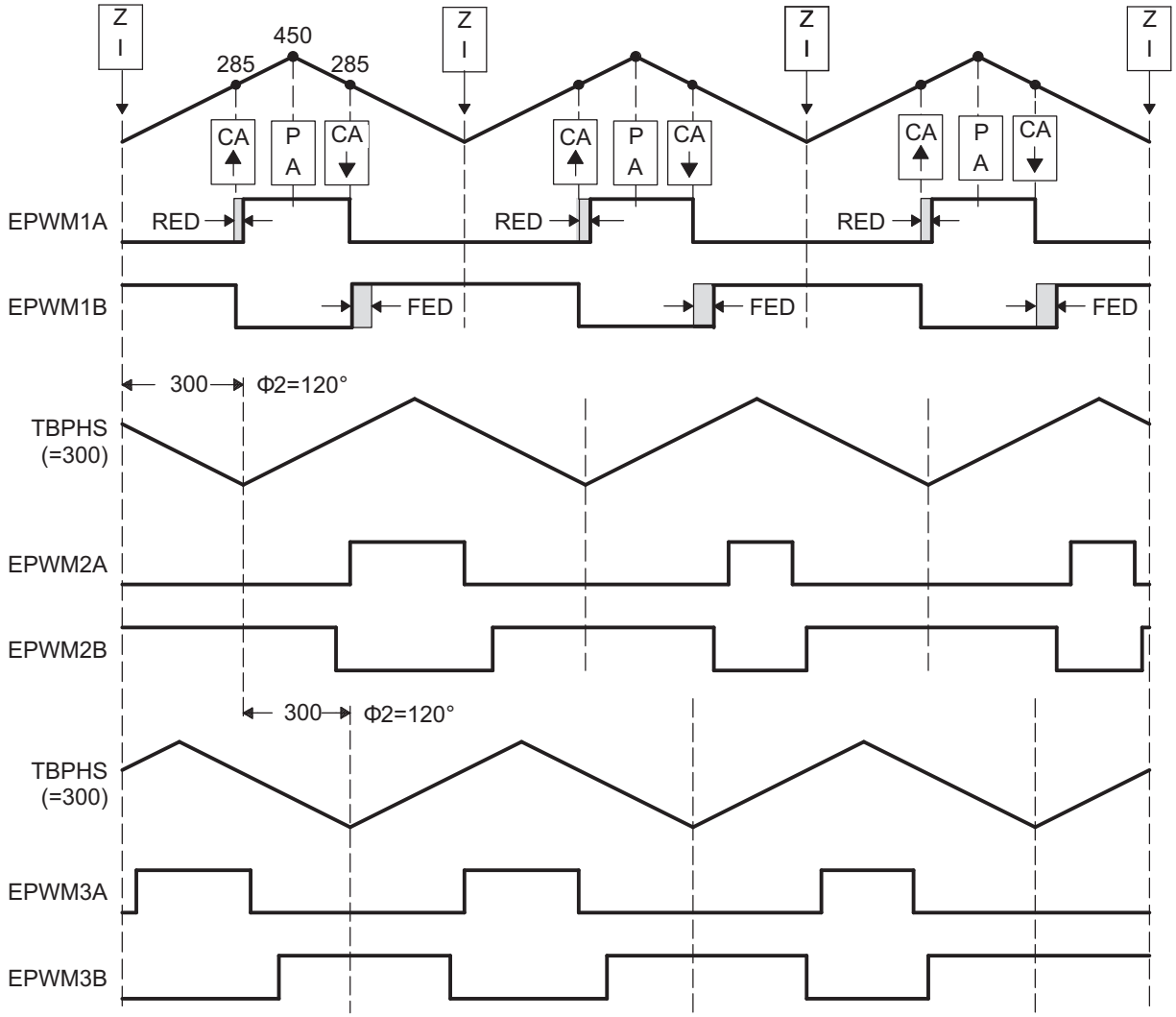
Copyright © 2013–2016, Texas Instruments Incorporated

Figure 14-61. Control of a 3-Phase Interleaved DC/DC Converter



Copyright © 2013–2016, Texas Instruments Incorporated

Figure 14-62. 3-Phase Interleaved DC/DC Converter Waveforms for Figure 14-61



Copyright © 2013–2016, Texas Instruments Incorporated

Table 14-45. EPWM1 Initialization for Figure 14-61

Register	Bit	Value	Comments
TBPRD	TBPRD	450 (1C2h)	Period = 900 TBCLK counts
TBPHS	TBPHS	0	Clear Phase Register to 0
TBCTL	CTRMODE	TB_UPDOWN	
	PHSEN	TB_DISABLE	Phase loading disabled
	PRDLD	TB_SHADOW	
	SYNCOSEL	TB_CTR_ZERO	Sync down-stream module
CMPCTL	SHDWAMODE	CC_SHADOW	
	SHDWBMODE	CC_SHADOW	
	LOADAMODE	CC_CTR_ZERO	Load on CTR = 0
	LOADBMODE	CC_CTR_ZERO	Load on CTR = 0
AQCTLA	CAU	AQ_SET	Set actions for EPWM1A
	CAD	AQ_CLEAR	
DBCTL	MODE	DB_FULL_ENABLE	Enable Dead-band module
	POLSEL	DB_ACTV_HIC	Active Hi complementary
DBFED	DBFED	20	FED = 20 TBCLKs
	DBRED	20	RED = 20 TBCLKs

Table 14-46. EPWM2 Initialization for Figure 14-61

Register	Bit	Value	Comments
TBPRD	TBPRD	450 (1C2h)	Period = 900 TBCLK counts
TBPHS	TBPHS	300	Phase = (300/900) × 360 = 120°
TBCTL	CTRMODE	TB_UPDOWN	
	PHSEN	TB_ENABLE	Slave module
	PRDLD	TB_SHADOW	
	SYNCOSEL	TB_SYNC_IN	Sync flow-through
	PHSDIR	TB_DOWN	Count DOWN on sync
CMPCTL	SHDWAMODE	CC_SHADOW	
	SHDWBMODE	CC_SHADOW	
	LOADAMODE	CC_CTR_ZERO	Load on CTR = 0
	LOADBMODE	CC_CTR_ZERO	Load on CTR = 0
AQCTLA	CAU	AQ_SET	Set actions for EPWM2A
	CAD	AQ_CLEAR	
DBCTL	MODE	DB_FULL_ENABLE	Enable Dead-band module
	POLSEL	DB_ACTV_HIC	Active Hi complementary
DBFED	DBFED	20	FED = 20 TBCLKs
	DBRED	20	RED = 20 TBCLKs

Copyright © 2013–2016, Texas Instruments Incorporated

Table 14-47. EPWM3 Initialization for Figure 14-61

Register	Bit	Value	Comments
TBPRD	TBPRD	450 (1C2h)	Period = 900 TBCLK counts
TBPHS	TBPHS	300	Phase = (300/900) × 360 = 120°
TBCTL	CTRMODE	TB_UPDOWN	
	PHSEN	TB_ENABLE	Slave module
	PRDLD	TB_SHADOW	
	SYNCOSEL	TB_SYNC_IN	Sync flow-through
	PHSDIR	TB_UP	Count UP on sync
CMPCTL	SHDWAMODE	CC_SHADOW	
	SHDWBMODE	CC_SHADOW	
	LOADAMODE	CC_CTR_ZERO	Load on CTR = 0
	LOADBMODE	CC_CTR_ZERO	Load on CTR = 0
AQCTLA	CAU	AQ_SET	Set actions for EPWM3A
	CAD	AQ_CLEAR	
DBCTL	MODE	DB_FULL_ENABLE	Enable Dead-band module
	POLSEL	DB_ACTV_HIC	Active Hi complementary
DBFED	DBFED	20	FED = 20 TBCLKs
	DBRED	20	RED = 20 TBCLKs

Example 14-7. Code Snippet for Configuration in Figure 14-61

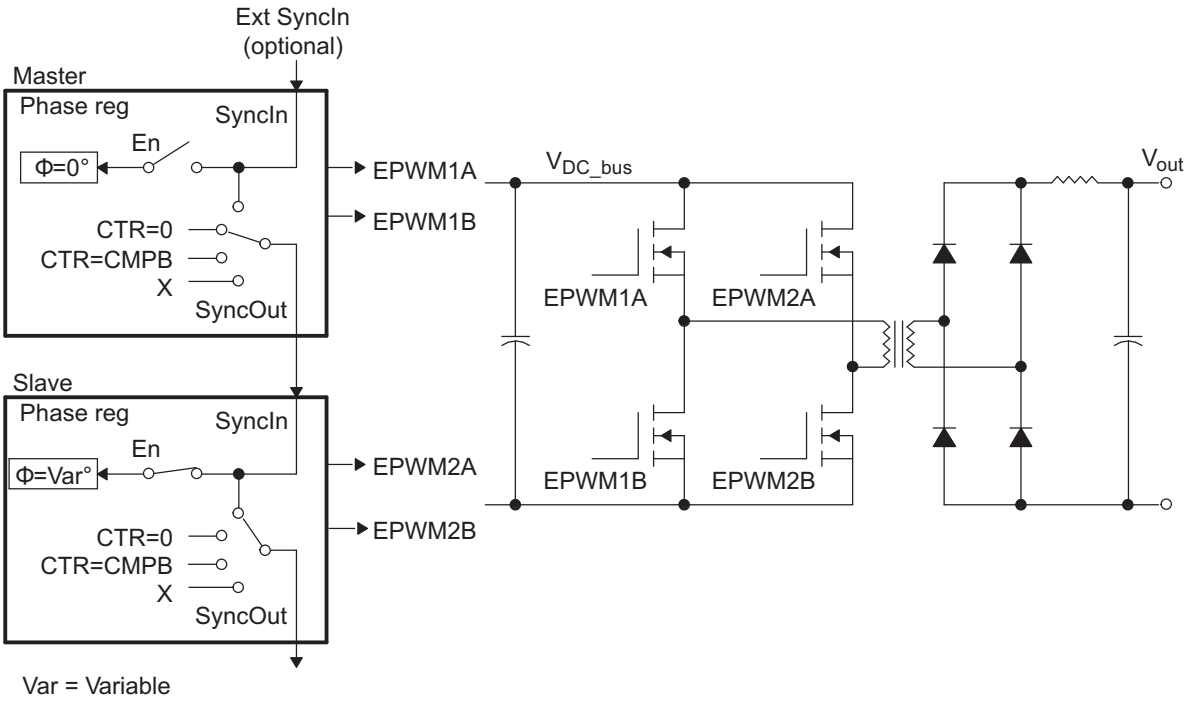
```
// Run Time (Note: Example execution of one run-time instance)
//=========================================================
EPwm1Regs.CMPA.half.CMPA = 285;         // adjust duty for output EPWM1A
EPwm2Regs.CMPA.half.CMPA = 285;         // adjust duty for output EPWM2A
EPwm3Regs.CMPA.half.CMPA = 285;         // adjust duty for output EPWM3A
```


Copyright © 2013–2016, Texas Instruments Incorporated

14.3.9 Controlling Zero Voltage Switched Full Bridge (ZVSFB) Converter

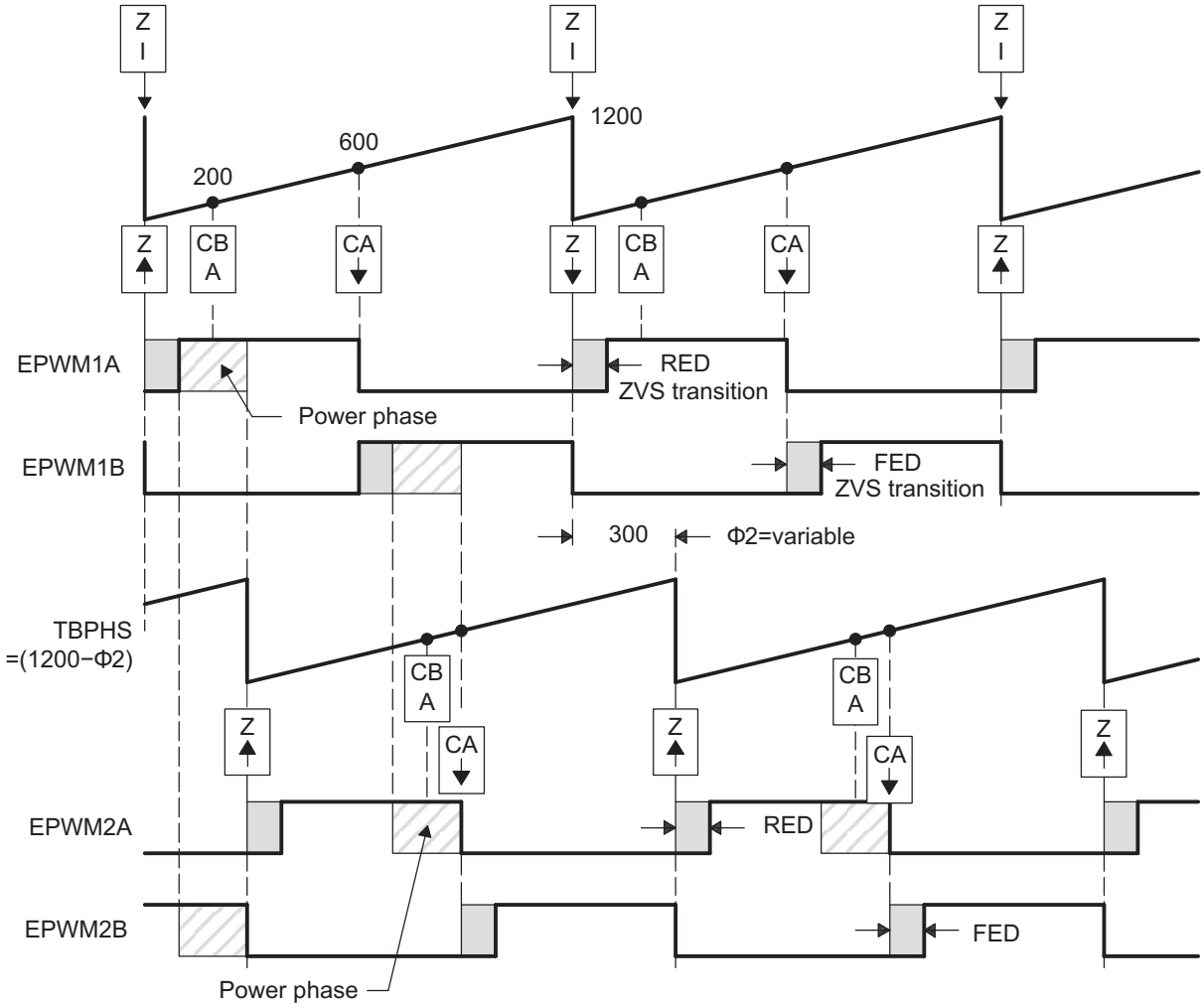
The example given in Figure 14-63 assumes a static or constant phase relationship between legs (modules). In such a case, control is achieved by modulating the duty cycle. It is also possible to dynamically change the phase value on a cycle-by-cycle basis. This feature lends itself to controlling a class of power topologies known as *phase-shifted full bridge*, or *zero voltage switched full bridge.* Here the controlled parameter is not duty cycle (this is kept constant at approximately 50 percent); instead it is the phase relationship between legs. Such a system can be implemented by allocating the resources of two PWM modules to control a single power stage, which in turn requires control of four switching elements. Figure 14-64 shows a master/slave module combination synchronized together to control a full H-bridge. In this case, both master and slave modules are required to switch at the same PWM frequency. The phase is controlled by using the slave's phase register (TBPHS). The master's phase register is not used and therefore can be initialized to zero.

Figure 14-63. Controlling a Full-H Bridge Stage (F_{PWM2} = F_{PWM1})



Copyright © 2013–2016, Texas Instruments Incorporated

Figure 14-64. ZVS Full-H Bridge Waveforms



Copyright © 2013–2016, Texas Instruments Incorporated

Table 14-48. EPWM1 Initialization for Figure 14-63

Register	Bit	Value	Comments
TBPRD	TBPRD	1200 (4B0h)	Period = 1201 TBCLK counts
TBPHS	TBPHS	0	Clear Phase Register to 0
TBCTL	CTRMODE	TB_UP	
	PHSEN	TB_DISABLE	Phase loading disabled
	PRDLD	TB_SHADOW	
	SYNCOSEL	TB_CTR_ZERO	Sync down-stream module
CMPA	CMPA	600 (258h)	Set 50% duty for EPWM1A
CMPCTL	SHDWAMODE	CC_SHADOW	
	SHDWBMODE	CC_SHADOW	
	LOADAMODE	CC_CTR_ZERO	Load on CTR = 0
	LOADBMODE	CC_CTR_ZERO	Load on CTR = 0
AQCTLA	ZRO	AQ_SET	Set actions for EPWM1A
	CAU	AQ_CLEAR	
DBCTL	MODE	DB_FULL_ENABLE	Enable Dead-band module
	POLSEL	DB_ACTV_HIC	Active Hi complementary
DBFED	DBFED	50	FED = 50 TBCLKs
	DBRED	70	RED = 70 TBCLKs

Table 14-49. EPWM2 Initialization for Figure 14-63

Register	Bit	Value	Comments
TBPRD	TBPRD	1200 (4B0h)	Period = 1201 TBCLK counts
TBPHS	TBPHS	0	Clear Phase Register to 0
TBCTL	CTRMODE	TB_UP	
	PHSEN	TB_ENABLE	Slave module
	PRDLD	TB_SHADOW	
	SYNCOSEL	TB_SYNC_IN	Sync flow-through
CMPA	CMPA	600 (258h)	Set 50% duty for EPWM2A
CMPCTL	SHDWAMODE	CC_SHADOW	
	SHDWBMODE	CC_SHADOW	
	LOADAMODE	CC_CTR_ZERO	Load on CTR = 0
	LOADBMODE	CC_CTR_ZERO	Load on CTR = 0
AQCTLA	ZRO	AQ_SET	Set actions for EPWM2A
	CAU	AQ_CLEAR	
DBCTL	MODE	DB_FULL_ENABLE	Enable Dead-band module
	POLSEL	DB_ACTV_HIC	Active Hi complementary
DBFED	DBFED	30	FED = 30 TBCLKs
	DBRED	40	RED = 40 TBCLKs

Example 14-8. Code Snippet for Configuration in Figure 14-63

```
// Run Time (Note: Example execution of one run-time instance)
//=========================================================
EPwm2Regs.TBPHS = 1200-300;                       // Set Phase reg to 300/1200 * 360 = 90 deg
EPwm1Regs.DBFED = FED1_NewValue;                  // Update ZVS transition interval
EPwm1Regs.DBRED = RED1_NewValue;                  // Update ZVS transition interval
EPwm2Regs.DBFED = FED2_NewValue;                  // Update ZVS transition interval
EPwm2Regs.DBRED = RED2_NewValue;                  // Update ZVS transition interval
```


Copyright © 2013–2016, Texas Instruments Incorporated

14.4 Registers

This section includes the registers for the submodules.

Table 14-50. Submodule Registers

Submodule	Section
Time-Base Submodule Registers	Section 14.4.1
Counter-Compare Submodule Registers	Section 14.4.2
Action-Qualifier Submodule Registers	Section 14.4.3
Dead-Band Generator Submodule Registers	Section 14.4.4
PWM-Chopper Submodule Registers	Section 14.4.5
Trip-Zone Submodule Registers	Section 14.4.6
Event-Trigger Submodule Registers	Section 14.4.7
High-Resolution PWM Registers	Section 14.4.8

14.4.1 Time-Base Submodule Registers

Table 14-51 lists the memory-mapped registers for the time-base submodule. See your device-specific data manual for the memory address of these registers. All other register offset addresses not listed in Table 14-51 should be considered as reserved locations and the register contents should not be modified.

Table 14-51. Time-Base Submodule Registers

Offset	Acronym	Register Description	Section
0h	TBCTL	Time-Base Control Register	Section 14.4.1.1
2h	TBSTS	Time-Base Status Register	Section 14.4.1.2
4h	TBPHSHR	Time-Base Phase High-Resolution Register(1)	Section 14.4.8.1
6h	TBPHS	Time-Base Phase Register	Section 14.4.1.3
8h	TBCNT	Time-Base Counter Register	Section 14.4.1.4
Ah	TBPRD	Time-Base Period Register	Section 14.4.1.5

(1) This register is only available on ePWM instances that include the high-resolution PWM (HRPWM) extension; otherwise, this location is reserved. See your device-specific data manual to determine which instances include the HRPWM.

14.4.1.1 Time-Base Control Register (TBCTL)

The time-base control register (TBCTL) is shown in Figure 14-65 and described in Table 14-52.

Figure 14-65. Time-Base Control Register (TBCTL)

15	14	13	12	10	9	8
FREE, SOFT		PHSDIR	CLKDIV		HSPCLKDIV	
R/W-0		R/W-0	R/W-0		R/W-0	

7	6	5	4	3	2	1	0
HSPCLKDIV	SWFSYNC	SYNCOSEL		PRDLD	PHSEN	CTRMODE	
R/W-1	R/W-0	R/W-0		R/W-0	R/W-0	R/W-3h	

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Copyright © 2013–2016, Texas Instruments Incorporated

Table 14-52. Time-Base Control Register (TBCTL) Field Descriptions

Bit	Field	Value	Description
15-14	FREE, SOFT	0-3h	Emulation Mode Bits. These bits select the behavior of the ePWM time-base counter during emulation events:
		0	Stop after the next time-base counter increment or decrement
		1h	Stop when counter completes a whole cycle: • Up-count mode: stop when the time-base counter = period (TBCNT = TBPRD) • Down-count mode: stop when the time-base counter = 0000 (TBCNT = 0000h) • Up-down-count mode: stop when the time-base counter = 0000 (TBCNT = 0000h)
		2h-3h	Free run
13	PHSDIR		Phase Direction Bit. This bit is only used when the time-base counter is configured in the up-down-count mode. The PHSDIR bit indicates the direction the time-base counter (TBCNT) will count after a synchronization event occurs and a new phase value is loaded from the phase (TBPHS) register. This is irrespective of the direction of the counter before the synchronization event.. In the up-count and down-count modes this bit is ignored.
		0	Count down after the synchronization event.
		1	Count up after the synchronization event.
12:10	CLKDIV	0-7h	Time-base Clock Prescale Bits. These bits determine part of the time-base clock prescale value. TBCLK = SYSCLKOUT/(HSPCLKDIV × CLKDIV)
		0	/1 (default on reset)
		1h	/2
		2h	/4
		3h	/8
		4h	/16
		5h	/32
		6h	/64
		7h	/128
9-7	HSPCLKDIV	0-7h	High-Speed Time-base Clock Prescale Bits. These bits determine part of the time-base clock prescale value. TBCLK = SYSCLKOUT/(HSPCLKDIV × CLKDIV) This divisor emulates the HSPCLK in the TMS320x281x system as used on the Event Manager (EV) peripheral.
		0	/1
		1h	/2 (default on reset)
		2h	/4
		3h	/6
		4h	/8
		5h	/10
		6h	/12
		7h	/14
6	SWFSYNC		Software Forced Synchronization Pulse
		0	Writing a 0 has no effect and reads always return a 0.
		1	Writing a 1 forces a one-time synchronization pulse to be generated. This event is ORed with the EPWMxSYNCI input of the ePWM module. SWFSYNC is valid (operates) only when EPWMxSYNCI is selected by SYNCOSEL = 00.
5-4	SYNCOSEL	0-3h	Synchronization Output Select. These bits select the source of the EPWMxSYNCO signal.
		0	EPWMxSYNC:
		1h	CTR = 0: Time-base counter equal to zero (TBCNT = 0000h)
		2h	CTR = CMPB : Time-base counter equal to counter-compare B (TBCNT = CMPB)
		3h	Disable EPWMxSYNCO signal

Copyright © 2013–2016, Texas Instruments Incorporated

Table 14-52. Time-Base Control Register (TBCTL) Field Descriptions (continued)

Bit	Field	Value	Description
3	PRDLD		Active Period Register Load From Shadow Register Select
		0	The period register (TBPRD) is loaded from its shadow register when the time-base counter, TBCNT, is equal to zero.
			A write or read to the TBPRD register accesses the shadow register.
		1	Load the TBPRD register immediately without using a shadow register.
			A write or read to the TBPRD register directly accesses the active register.
2	PHSEN		Counter Register Load From Phase Register Enable
		0	Do not load the time-base counter (TBCNT) from the time-base phase register (TBPHS)
		1	Load the time-base counter with the phase register when an EPWMxSYNCI input signal occurs or when a software synchronization is forced by the SWFSYNC bit.
1-0	CTRMODE	0-3h	Counter Mode. The time-base counter mode is normally configured once and not changed during normal operation. If you change the mode of the counter, the change will take effect at the next TBCLK edge and the current counter value shall increment or decrement from the value before the mode change.
			These bits set the time-base counter mode of operation as follows:
		0	Up-count mode
		1h	Down-count mode
		2h	Up-down-count mode
		3h	Stop-freeze counter operation (default on reset)

14.4.1.2 Time-Base Status Register (TBSTS)

The time-base status register (TBSTS) is shown in Figure 14-66 and described in Table 14-53.

Figure 14-66. Time-Base Status Register (TBSTS)

15	3	2	1	0
Reserved		CTRMAX	SYNCI	CTRDIR
R-0		R/W1C-0	R/W1C-0	R-1

LEGEND: R/W = Read/Write; R/W1C = Read/Write 1 to clear; *-n* = value after reset

Table 14-53. Time-Base Status Register (TBSTS) Field Descriptions

Bit	Field	Value	Description
15-3	Reserved	0	Reserved
2	CTRMAX		Time-Base Counter Max Latched Status Bit
		0	Reading a 0 indicates the time-base counter never reached its maximum value. Writing a 0 will have no effect.
		1	Reading a 1 on this bit indicates that the time-base counter reached the max value 0xFFFF. Writing a 1 to this bit will clear the latched event.
1	SYNCI		Input Synchronization Latched Status Bit
		0	Writing a 0 will have no effect. Reading a 0 indicates no external synchronization event has occurred.
		1	Reading a 1 on this bit indicates that an external synchronization event has occurred (EPWMxSYNCI). Writing a 1 to this bit will clear the latched event.
0	CTRDIR		Time-Base Counter Direction Status Bit. At reset, the counter is frozen; therefore, this bit has no meaning. To make this bit meaningful, you must first set the appropriate mode via TBCTL[CTRMODE].
		0	Time-Base Counter is currently counting down.
		1	Time-Base Counter is currently counting up.

14.4.1.3 Time-Base Phase Register (TBPHS)

The time-base phase register (TBPHS) is shown in Figure 14-67 and described in Table 14-54.

Copyright © 2013–2016, Texas Instruments Incorporated

Figure 14-67. Time-Base Phase Register (TBPHS)

15	0
TBPHS	
R/W-0	

LEGEND: R/W = Read/Write; -*n* = value after reset

Table 14-54. Time-Base Phase Register (TBPHS) Field Descriptions

Bits	Name	Value	Description
15-0	TBPHS	0-FFFFh	These bits set time-base counter phase of the selected ePWM relative to the time-base that is supplying the synchronization input signal. • If TBCTL[PHSEN] = 0, then the synchronization event is ignored and the time-base counter is not loaded with the phase. • If TBCTL[PHSEN] = 1, then the time-base counter (TBCNT) will be loaded with the phase (TBPHS) when a synchronization event occurs. The synchronization event can be initiated by the input synchronization signal (EPWMxSYNCI) or by a software forced synchronization.

14.4.1.4 Time-Base Counter Register (TBCNT)

The time-base counter register (TBCNT) is shown in Figure 14-68 and described in Table 14-55.

Figure 14-68. Time-Base Counter Register (TBCNT)

15	0
TBCNT	
R/W-0	

LEGEND: R/W = Read/Write; -*n* = value after reset

Table 14-55. Time-Base Counter Register (TBCNT) Field Descriptions

Bits	Name	Value	Description
15-0	TBCNT	0-FFFFh	Reading these bits gives the current time-base counter value. Writing to these bits sets the current time-base counter value. The update happens as soon as the write occurs; the write is NOT synchronized to the time-base clock (TBCLK) and the register is not shadowed.

Copyright © 2013–2016, Texas Instruments Incorporated

14.4.1.5 Time-Base Period Register (TBPRD)

The time-base period register (TBPRD) is shown in Figure 14-69 and described in Table 14-56.

Figure 14-69. Time-Base Period Register (TBPRD)

15		0
TBPRD		
R/W-0		

LEGEND: R/W = Read/Write; -*n* = value after reset

Table 14-56. Time-Base Period Register (TBPRD) Field Descriptions

Bits	Name	Value	Description
15-0	TBPRD	0-FFFFh	These bits determine the period of the time-base counter. This sets the PWM frequency. Shadowing of this register is enabled and disabled by the TBCTL[PRDLD] bit. By default this register is shadowed. • If TBCTL[PRDLD] = 0, then the shadow is enabled and any write or read will automatically go to the shadow register. In this case, the active register will be loaded from the shadow register when the time-base counter equals zero. • If TBCTL[PRDLD] = 1, then the shadow is disabled and any write or read will go directly to the active register, that is the register actively controlling the hardware. • The active and shadow registers share the same memory map address.

14.4.2 Counter-Compare Submodule Registers

Table 14-57 lists the memory-mapped registers for the counter-compare submodule. See your device-specific data manual for the memory address of these registers. All other register offset addresses not listed in Table 14-57 should be considered as reserved locations and the register contents should not be modified.

Table 14-57. Counter-Compare Submodule Registers

Offset	Acronym	Register Description	Section
Eh	CMPCTL	Counter-Compare Control Register	Section 14.4.2.1
10h	CMPAHR	Counter-Compare A High-Resolution Register(1)	Section 14.4.8.2
12h	CMPA	Counter-Compare A Register	Section 14.4.2.2
14h	CMPB	Counter-Compare B Register	Section 14.4.2.3

(1) This register is only available on ePWM instances that include the high-resolution PWM (HRPWM) extension; otherwise, this location is reserved. See your device-specific data manual to determine which instances include the HRPWM.

Copyright © 2013–2016, Texas Instruments Incorporated

14.4.2.1 Counter-Compare Control Register (CMPCTL)

The counter-compare control register (CMPCTL) is shown in Figure 14-70 and described in Table 14-58.

Figure 14-70. Counter-Compare Control Register (CMPCTL)

15					10	9	8
Reserved						SHDWBFULL	SHDWAFULL
R-0						R-0	R-0
7	6	5	4	3	2	1	0
Reserved	SHDWBMODE	Reserved	SHDWAMODE	LOADBMODE		LOADAMODE	
R-0	R/W-0	R-0	R/W-0	R/W-0		R/W-0	

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 14-58. Counter-Compare Control Register (CMPCTL) Field Descriptions

Bits	Name	Value	Description
15-10	Reserved	0	Reserved
9	SHDWBFULL		Counter-compare B (CMPB) Shadow Register Full Status Flag. This bit self clears once a load-strobe occurs.
		0	CMPB shadow FIFO not full yet
		1	Indicates the CMPB shadow FIFO is full; a CPU write will overwrite current shadow value.
8	SHDWAFULL		Counter-compare A (CMPA) Shadow Register Full Status Flag. The flag bit is set when a 32-bit write to CMPA:CMPAHR register or a 16-bit write to CMPA register is made. A 16-bit write to CMPAHR register will not affect the flag. This bit self clears once a load-strobe occurs.
		0	CMPA shadow FIFO not full yet
		1	Indicates the CMPA shadow FIFO is full, a CPU write will overwrite the current shadow value.
7	Reserved	0	Reserved
6	SHDWBMODE		Counter-compare B (CMPB) Register Operating Mode
		0	Shadow mode. Operates as a double buffer. All writes via the CPU access the shadow register.
		1	Immediate mode. Only the active compare B register is used. All writes and reads directly access the active register for immediate compare action.
5	Reserved		Reserved
4	SHDWAMODE		Counter-compare A (CMPA) Register Operating Mode
		0	Shadow mode. Operates as a double buffer. All writes via the CPU access the shadow register.
		1	Immediate mode. Only the active compare register is used. All writes and reads directly access the active register for immediate compare action
3-2	LOADBMODE	0-3h	Active Counter-Compare B (CMPB) Load From Shadow Select Mode. This bit has no effect in immediate mode (CMPCTL[SHDWBMODE] = 1).
		0	Load on CTR = 0: Time-base counter equal to zero (TBCNT = 0000h)
		1h	Load on CTR = PRD: Time-base counter equal to period (TBCNT = TBPRD)
		2h	Load on either CTR = 0 or CTR = PRD
		3h	Freeze (no loads possible)
1-0	LOADAMODE	0-3h	Active Counter-Compare A (CMPA) Load From Shadow Select Mode. This bit has no effect in immediate mode (CMPCTL[SHDWAMODE] = 1).
		0	Load on CTR = 0: Time-base counter equal to zero (TBCNT = 0000h)
		1h	Load on CTR = PRD: Time-base counter equal to period (TBCNT = TBPRD)
		2h	Load on either CTR = 0 or CTR = PRD
		3h	Freeze (no loads possible)

Copyright © 2013–2016, Texas Instruments Incorporated

14.4.2.2 Counter-Compare A Register (CMPA)

The counter-compare A register (CMPA) is shown in Figure 14-71 and described in Table 14-59.

Figure 14-71. Counter-Compare A Register (CMPA)

15		0
CMPA		
R/W-0		

LEGEND: R/W = Read/Write; -*n* = value after reset

Table 14-59. Counter-Compare A Register (CMPA) Field Descriptions

Bits	Name	Value	Description
15-0	CMPA	0-FFFFh	The value in the active CMPA register is continuously compared to the time-base counter (TBCNT). When the values are equal, the counter-compare module generates a "time-base counter equal to counter compare A" event. This event is sent to the action-qualifier where it is qualified and converted it into one or more actions. These actions can be applied to either the EPWMxA or the EPWMxB output depending on the configuration of the AQCTLA and AQCTLB registers. The actions that can be defined in the AQCTLA and AQCTLB registers include: • Do nothing; the event is ignored. • Clear: Pull the EPWMxA and/or EPWMxB signal low • Set: Pull the EPWMxA and/or EPWMxB signal high • Toggle the EPWMxA and/or EPWMxB signal Shadowing of this register is enabled and disabled by the CMPCTL[SHDWAMODE] bit. By default this register is shadowed. • If CMPCTL[SHDWAMODE] = 0, then the shadow is enabled and any write or read will automatically go to the shadow register. In this case, the CMPCTL[LOADAMODE] bit field determines which event will load the active register from the shadow register. • Before a write, the CMPCTL[SHDWAFULL] bit can be read to determine if the shadow register is currently full. • If CMPCTL[SHDWAMODE] = 1, then the shadow register is disabled and any write or read will go directly to the active register, that is the register actively controlling the hardware. • In either mode, the active and shadow registers share the same memory map address.

Copyright © 2013–2016, Texas Instruments Incorporated

14.4.2.3 Counter-Compare B Register (CMPB)

The counter-compare B register (CMPB) is shown in Figure 14-72 and described in Table 14-60.

Figure 14-72. Counter-Compare B Register (CMPB)

15	0
CMPB	
R/W-0	

LEGEND: R/W = Read/Write; -*n* = value after reset

Table 14-60. Counter-Compare B Register (CMPB) Field Descriptions

Bits	Name	Value	Description
15-0	CMPB	0-FFFFh	The value in the active CMPB register is continuously compared to the time-base counter (TBCNT). When the values are equal, the counter-compare module generates a "time-base counter equal to counter compare B" event. This event is sent to the action-qualifier where it is qualified and converted it into one or more actions. These actions can be applied to either the EPWMxA or the EPWMxB output depending on the configuration of the AQCTLA and AQCTLB registers. The actions that can be defined in the AQCTLA and AQCTLB registers include: • Do nothing. event is ignored. • Clear: Pull the EPWMxA and/or EPWMxB signal low • Set: Pull the EPWMxA and/or EPWMxB signal high • Toggle the EPWMxA and/or EPWMxB signal Shadowing of this register is enabled and disabled by the CMPCTL[SHDWBMODE] bit. By default this register is shadowed. • If CMPCTL[SHDWBMODE] = 0, then the shadow is enabled and any write or read will automatically go to the shadow register. In this case, the CMPCTL[LOADBMODE] bit field determines which event will load the active register from the shadow register: • Before a write, the CMPCTL[SHDWBFULL] bit can be read to determine if the shadow register is currently full. • If CMPCTL[SHDWBMODE] = 1, then the shadow register is disabled and any write or read will go directly to the active register, that is the register actively controlling the hardware. • In either mode, the active and shadow registers share the same memory map address.

14.4.3 Action-Qualifier Submodule Registers

Table 14-61 lists the memory-mapped registers for the action-qualifier submodule. See your device-specific data manual for the memory address of these registers. All other register offset addresses not listed in Table 14-61 should be considered as reserved locations and the register contents should not be modified.

Table 14-61. Action-Qualifier Submodule Registers

Offset	Acronym	Register Description	Section
16h	AQCTLA	Action-Qualifier Output A Control Register	Section 14.4.3.1
18h	AQCTLB	Action-Qualifier Output B Control Register	Section 14.4.3.2
1Ah	AQSFRC	Action-Qualifier Software Force Register	Section 14.4.3.3
1Ch	AQCSFRC	Action-Qualifier Continuous Software Force Register	Section 14.4.3.4

Copyright © 2013–2016, Texas Instruments Incorporated

14.4.3.1 Action-Qualifier Output A Control Register (AQCTLA)

The action-qualifier output A control register (AQCTLA) is shown in Figure 14-73 and described in Table 14-62.

Figure 14-73. Action-Qualifier Output A Control Register (AQCTLA)

15 12	11 10	9 8	7 6	5 4	3 2	1 0
Reserved	CBD	CBU	CAD	CAU	PRD	ZRO
R-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 14-62. Action-Qualifier Output A Control Register (AQCTLA) Field Descriptions

Bits	Name	Value	Description
15-12	Reserved	0	Reserved
11-10	CBD	0-3h	Action when the time-base counter equals the active CMPB register and the counter is decrementing.
		0	Do nothing (action disabled)
		1h	Clear: force EPWMxA output low.
		2h	Set: force EPWMxA output high.
		3h	Toggle EPWMxA output: low output signal will be forced high, and a high signal will be forced low.
9-8	CBU	0-3h	Action when the counter equals the active CMPB register and the counter is incrementing.
		0	Do nothing (action disabled)
		1h	Clear: force EPWMxA output low.
		2h	Set: force EPWMxA output high.
		3h	Toggle EPWMxA output: low output signal will be forced high, and a high signal will be forced low.
7-6	CAD	0-3h	Action when the counter equals the active CMPA register and the counter is decrementing.
		0	Do nothing (action disabled)
		1h	Clear: force EPWMxA output low.
		2h	Set: force EPWMxA output high.
		3h	Toggle EPWMxA output: low output signal will be forced high, and a high signal will be forced low.
5-4	CAU	0-3h	Action when the counter equals the active CMPA register and the counter is incrementing.
		0	Do nothing (action disabled)
		1h	Clear: force EPWMxA output low.
		2h	Set: force EPWMxA output high.
		3h	Toggle EPWMxA output: low output signal will be forced high, and a high signal will be forced low.
3-2	PRD	0-3h	Action when the counter equals the period. Note: By definition, in count up-down mode when the counter equals period the direction is defined as 0 or counting down.
		0	Do nothing (action disabled)
		1h	Clear: force EPWMxA output low.
		2h	Set: force EPWMxA output high.
		3h	Toggle EPWMxA output: low output signal will be forced high, and a high signal will be forced low.
1-0	ZRO	0-3h	Action when counter equals zero. Note: By definition, in count up-down mode when the counter equals 0 the direction is defined as 1 or counting up.
		0	Do nothing (action disabled)
		1h	Clear: force EPWMxA output low.
		2h	Set: force EPWMxA output high.
		3h	Toggle EPWMxA output: low output signal will be forced high, and a high signal will be forced low.

Copyright © 2013–2016, Texas Instruments Incorporated

14.4.3.2 Action-Qualifier Output B Control Register (AQCTLB)

The action-qualifier output B control register (AQCTLB) is shown in Figure 14-74 and described in Table 14-63.

Figure 14-74. Action-Qualifier Output B Control Register (AQCTLB)

15 12	11 10	9 8	7 6	5 4	3 2	1 0
Reserved	CBD	CBU	CAD	CAU	PRD	ZRO
R-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 14-63. Action-Qualifier Output B Control Register (AQCTLB) Field Descriptions

Bits	Name	Value	Description
15-12	Reserved	0	Reserved
11-10	CBD	0-3h	Action when the counter equals the active CMPB register and the counter is decrementing.
		0	Do nothing (action disabled)
		1h	Clear: force EPWMxB output low.
		2h	Set: force EPWMxB output high.
		3h	Toggle EPWMxB output: low output signal will be forced high, and a high signal will be forced low.
9-8	CBU	0-3h	Action when the counter equals the active CMPB register and the counter is incrementing.
		0	Do nothing (action disabled)
		1h	Clear: force EPWMxB output low.
		2h	Set: force EPWMxB output high.
		3h	Toggle EPWMxB output: low output signal will be forced high, and a high signal will be forced low.
7-6	CAD	0-3h	Action when the counter equals the active CMPA register and the counter is decrementing.
		0	Do nothing (action disabled)
		1h	Clear: force EPWMxB output low.
		2h	Set: force EPWMxB output high.
		3h	Toggle EPWMxB output: low output signal will be forced high, and a high signal will be forced low.
5-4	CAU	0-3h	Action when the counter equals the active CMPA register and the counter is incrementing.
		0	Do nothing (action disabled)
		1h	Clear: force EPWMxB output low.
		2h	Set: force EPWMxB output high.
		3h	Toggle EPWMxB output: low output signal will be forced high, and a high signal will be forced low.
3-2	PRD	0-3h	Action when the counter equals the period. Note: By definition, in count up-down mode when the counter equals period the direction is defined as 0 or counting down.
		0	Do nothing (action disabled)
		1h	Clear: force EPWMxB output low.
		2h	Set: force EPWMxB output high.
		3h	Toggle EPWMxB output: low output signal will be forced high, and a high signal will be forced low.
1-0	ZRO	0-3h	Action when counter equals zero. Note: By definition, in count up-down mode when the counter equals 0 the direction is defined as 1 or counting up.
		0	Do nothing (action disabled)
		1h	Clear: force EPWMxB output low.
		2h	Set: force EPWMxB output high.
		3h	Toggle EPWMxB output: low output signal will be forced high, and a high signal will be forced low.

Copyright © 2013–2016, Texas Instruments Incorporated

14.4.3.3 Action-Qualifier Software Force Register (AQSFRC)

The action-qualifier software force register (AQSFRC) is shown in Figure 14-75 and described in Table 14-64.

Figure 14-75. Action-Qualifier Software Force Register (AQSFRC)

15 8	7 6	5	4 3	2	1 0
Reserved	RLDCSF	OTSFB	ACTSFB	OTSFA	ACTSFA
R-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 14-64. Action-Qualifier Software Force Register (AQSFRC) Field Descriptions

Bit	Field	Value	Description
15-8	Reserved	0	Reserved
7-6	RLDCSF	0-3h	AQCSFRC Active Register Reload From Shadow Options
		0	Load on event counter equals zero
		1h	Load on event counter equals period
		2h	Load on event counter equals zero or counter equals period
		3h	Load immediately (the active register is directly accessed by the CPU and is not loaded from the shadow register).
5	OTSFB		One-Time Software Forced Event on Output B
		0	Writing a 0 (zero) has no effect. Always reads back a 0
			This bit is auto cleared once a write to this register is complete, that is, a forced event is initiated.)
			This is a one-shot forced event. It can be overridden by another subsequent event on output B.
		1	Initiates a single s/w forced event
4-3	ACTSFB	0-3h	Action when One-Time Software Force B Is invoked
		0	Does nothing (action disabled)
		1h	Clear (low)
		2h	Set (high)
		3h	Toggle (Low -> High, High -> Low)
			Note: This action is not qualified by counter direction (CNT_dir)
2	OTSFA		One-Time Software Forced Event on Output A
		0	Writing a 0 (zero) has no effect. Always reads back a 0.
			This bit is auto cleared once a write to this register is complete (that is, a forced event is initiated).
		1	Initiates a single software forced event
1-0	ACTSFA	0-3h	Action When One-Time Software Force A Is Invoked
		0	Does nothing (action disabled)
		1h	Clear (low)
		2h	Set (high)
		3h	Toggle (Low → High, High → Low)
			Note: This action is not qualified by counter direction (CNT_dir)

Copyright © 2013–2016, Texas Instruments Incorporated

14.4.3.4 Action-Qualifier Continuous Software Force Register (AQCSFRC)

The action-qualifier continuous software force register (AQCSFRC) is shown in Figure 14-76 and described in Table 14-65.

Figure 14-76. Action-Qualifier Continuous Software Force Register (AQCSFRC)

15	4	3	2	1	0
Reserved		CSFB		CSFA	
R-0		R/W-0		R/W-0	

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 14-65. Action-Qualifier Continuous Software Force Register (AQCSFRC) Field Descriptions

Bits	Name	Value	Description
15-4	Reserved	0	Reserved
3-2	CSFB	0-3h	Continuous Software Force on Output B
			In immediate mode, a continuous force takes effect on the next TBCLK edge.
			In shadow mode, a continuous force takes effect on the next TBCLK edge after a shadow load into the active register. To configure shadow mode, use AQSFRC[RLDCSF].
		0	Forcing disabled, that is, has no effect
		1h	Forces a continuous low on output B
		2h	Forces a continuous high on output B
		3h	Software forcing is disabled and has no effect
1-0	CSFA	0-3h	Continuous Software Force on Output A
			In immediate mode, a continuous force takes effect on the next TBCLK edge.
			In shadow mode, a continuous force takes effect on the next TBCLK edge after a shadow load into the active register.
		0	Forcing disabled, that is, has no effect
		1h	Forces a continuous low on output A
		2h	Forces a continuous high on output A
		3h	Software forcing is disabled and has no effect

14.4.4 Dead-Band Generator Submodule Registers

Table 14-66 lists the memory-mapped registers for the dead-band generator submodule. See your device-specific data manual for the memory address of these registers. All other register offset addresses not listed in Table 14-66 should be considered as reserved locations and the register contents should not be modified.

Table 14-66. Dead-Band Generator Submodule Registers

Offset	Acronym	Register Description	Section
1Eh	DBCTL	Dead-Band Generator Control Register	Section 14.4.4.1
20h	DBRED	Dead-Band Generator Rising Edge Delay Register	Section 14.4.4.2
22h	DBFED	Dead-Band Generator Falling Edge Delay Register	Section 14.4.4.3

Copyright © 2013–2016, Texas Instruments Incorporated

14.4.4.1 Dead-Band Generator Control Register (DBCTL)

The dead-band generator control register (DBCTL) is shown in Figure 14-77 and described in Table 14-67.

Figure 14-77. Dead-Band Generator Control Register (DBCTL)

15 6	5 4	3 2	1 0
Reserved	IN_MODE	POLSEL	OUT_MODE
R-0	R/W-0	R/W-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 14-67. Dead-Band Generator Control Register (DBCTL) Field Descriptions

Bits	Name	Value	Description
15-6	Reserved	0	Reserved
5-4	IN_MODE	0-3h	Dead Band Input Mode Control. Bit 5 controls the S5 switch and bit 4 controls the S4 switch shown in Figure 14-29. This allows you to select the input source to the falling-edge and rising-edge delay. To produce classical dead-band waveforms, the default is EPWMxA In is the source for both falling and rising-edge delays.
		0	EPWMxA In (from the action-qualifier) is the source for both falling-edge and rising-edge delay.
		1h	EPWMxB In (from the action-qualifier) is the source for rising-edge delayed signal. EPWMxA In (from the action-qualifier) is the source for falling-edge delayed signal.
		2h	EPWMxA In (from the action-qualifier) is the source for rising-edge delayed signal. EPWMxB In (from the action-qualifier) is the source for falling-edge delayed signal.
		3h	EPWMxB In (from the action-qualifier) is the source for both rising-edge delay and falling-edge delayed signal.
3-2	POLSEL	0-3h	Polarity Select Control. Bit 3 controls the S3 switch and bit 2 controls the S2 switch shown in Figure 14-29. This allows you to selectively invert one of the delayed signals before it is sent out of the dead-band submodule. The following descriptions correspond to classical upper/lower switch control as found in one leg of a digital motor control inverter. These assume that DBCTL[OUT_MODE] = 1,1 and DBCTL[IN_MODE] = 0,0. Other enhanced modes are also possible, but not regarded as typical usage modes.
		0	Active high (AH) mode. Neither EPWMxA nor EPWMxB is inverted (default).
		1h	Active low complementary (ALC) mode. EPWMxA is inverted.
		2h	Active high complementary (AHC). EPWMxB is inverted.
		3h	Active low (AL) mode. Both EPWMxA and EPWMxB are inverted.
1-0	OUT_MODE	0-3h	Dead-band Output Mode Control. Bit 1 controls the S1 switch and bit 0 controls the S0 switch shown in Figure 14-29. This allows you to selectively enable or bypass the dead-band generation for the falling-edge and rising-edge delay.
		0	Dead-band generation is bypassed for both output signals. In this mode, both the EPWMxA and EPWMxB output signals from the action-qualifier are passed directly to the PWM-chopper submodule. In this mode, the POLSEL and IN_MODE bits have no effect.
		1h	Disable rising-edge delay. The EPWMxA signal from the action-qualifier is passed straight through to the EPWMxA input of the PWM-chopper submodule. The falling-edge delayed signal is seen on output EPWMxB. The input signal for the delay is determined by DBCTL[IN_MODE].
		2h	Disable falling-edge delay. The EPWMxB signal from the action-qualifier is passed straight through to the EPWMxB input of the PWM-chopper submodule. The rising-edge delayed signal is seen on output EPWMxA. The input signal for the delay is determined by DBCTL[IN_MODE].
		3h	Dead-band is fully enabled for both rising-edge delay on output EPWMxA and falling-edge delay on output EPWMxB. The input signal for the delay is determined by DBCTL[IN_MODE].

Copyright © 2013–2016, Texas Instruments Incorporated

14.4.4.2 Dead-Band Generator Rising Edge Delay Register (DBRED)

The dead-band generator rising edge delay register (DBRED) is shown in Figure 14-78 and described in Table 14-68.

Figure 14-78. Dead-Band Generator Rising Edge Delay Register (DBRED)

15	10	9	0
Reserved		DEL	
R-0		R/W-0	

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 14-68. Dead-Band Generator Rising Edge Delay Register (DBRED) Field Descriptions

Bits	Name	Value	Description
15-10	Reserved	0	Reserved
9-0	DEL	0-3FFh	Rising Edge Delay Count. 10-bit counter.

14.4.4.3 Dead-Band Generator Falling Edge Delay Register (DBFED)

The dead-band generator falling edge delay register (DBFED) is shown in Figure 14-79 and described in Table 14-69.

Figure 14-79. Dead-Band Generator Falling Edge Delay Register (DBFED)

15	10	9	0
Reserved		DEL	
R-0		R/W-0	

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 14-69. Dead-Band Generator Falling Edge Delay Register (DBFED) Field Descriptions

Bits	Name	Value	Description
15-10	Reserved	0	Reserved
9-0	DEL	0-3FFh	Falling Edge Delay Count. 10-bit counter

Copyright © 2013–2016, Texas Instruments Incorporated

14.4.5 PWM-Chopper Submodule Register

The PWM-chopper control register (PCCTL) is shown in Figure 14-80 and described in Table 14-70.

Figure 14-80. PWM-Chopper Control Register (PCCTL)

15 11	10 8	7 5	4 1	0
Reserved	CHPDUTY	CHPFREQ	OSHTWTH	CHPEN
R-0	R/W-0	R/W-0	R/W-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; *-n* = value after reset

Table 14-70. PWM-Chopper Control Register (PCCTL) Bit Descriptions

Bits	Name	Value	Description
15-11	Reserved	0	Reserved
10-8	CHPDUTY	0-7h	Chopping Clock Duty Cycle
		0	Duty = 1/8 (12.5%)
		1h	Duty = 2/8 (25.0%)
		2h	Duty = 3/8 (37.5%)
		3h	Duty = 4/8 (50.0%)
		4h	Duty = 5/8 (62.5%)
		5h	Duty = 6/8 (75.0%)
		6h	Duty = 7/8 (87.5%)
		7h	Reserved
7-5	CHPFREQ	0-7h	Chopping Clock Frequency
		0	Divide by 1 (no prescale)
		1h	Divide by 2
		2h	Divide by 3
		3h-7h	Divide by 4 to divide by 8
4-1	OSHTWTH	0-Fh	One-Shot Pulse Width
		0	1 × SYSCLKOUT/8 wide
		1h	2 × SYSCLKOUT/8 wide
		2h	3 × SYSCLKOUT/8 wide
		3h-Fh	4 × SYSCLKOUT/8 wide to 16 × SYSCLKOUT/8 wide
0	CHPEN		PWM-chopping Enable
		0	Disable (bypass) PWM chopping function
		1	Enable chopping function

Copyright © 2013–2016, Texas Instruments Incorporated

14.4.6 Trip-Zone Submodule Registers

Table 14-71 lists the memory-mapped registers for the trip-zone submodule. See your device-specific data manual for the memory address of these registers. All other register offset addresses not listed in Table 14-71 should be considered as reserved locations and the register contents should not be modified.

Table 14-71. Trip-Zone Submodule Registers

Offset	Acronym	Register Description	Section
24h	TZSEL	Trip-Zone Select Register	Section 14.4.6.1
28h	TZCTL	Trip-Zone Control Register	Section 14.4.6.2
2Ah	TZEINT	Trip-Zone Enable Interrupt Register	Section 14.4.6.3
2Ch	TZFLG	Trip-Zone Flag Register	Section 14.4.6.4
2Eh	TZCLR	Trip-Zone Clear Register	Section 14.4.6.5
30h	TZFRC	Trip-Zone Force Register	Section 14.4.6.6

14.4.6.1 Trip-Zone Select Register (TZSEL)

The trip-zone select register (TZSEL) is shown in Figure 14-81 and described in Table 14-72.

Figure 14-81. Trip-Zone Select Register (TZSEL)

15	9	8	7	1	0
Reserved/OSHT*n* (1)		OSHT1	Reserved/CBC*n* (1)		CBC1
R/W-0		R/W-0	R/W-0		R/W-0

LEGEND: R/W = Read/Write; -*n* = value after reset

(1) Number of register bits depends on how many trip-zone pins are available in the device. See your device-specific data manual.

Table 14-72. Trip-Zone Submodule Select Register (TZSEL) Field Descriptions

Bits	Name	Value	Description
15-8	OSHT*n*		Trip-zone *n* ($\overline{\text{TZn}}$) select. One-Shot (OSHT) trip-zone enable/disable. When any of the enabled pins go low, a one-shot trip event occurs for this ePWM module. When the event occurs, the action defined in the TZCTL register (Section 14.4.6.2) is taken on the EPWMxA and EPWMxB outputs. The one-shot trip condition remains latched until you clear the condition via the TZCLR register (Section 14.4.6.5).
		0	Disable $\overline{\text{TZn}}$ as a one-shot trip source for this ePWM module.
		1	Enable $\overline{\text{TZn}}$ as a one-shot trip source for this ePWM module.
7-0	CBC*n*		Trip-zone *n* ($\overline{\text{TZn}}$) select. Cycle-by-Cycle (CBC) trip-zone enable/disable. When any of the enabled pins go low, a cycle-by-cycle trip event occurs for this ePWM module. When the event occurs, the action defined in the TZCTL register (Section 14.4.6.2) is taken on the EPWMxA and EPWMxB outputs. A cycle-by-cycle trip condition is automatically cleared when the time-base counter reaches zero.
		0	Disable $\overline{\text{TZn}}$ as a CBC trip source for this ePWM module.
		1	Enable $\overline{\text{TZn}}$ as a CBC trip source for this ePWM module.

Copyright © 2013–2016, Texas Instruments Incorporated

14.4.6.2 Trip-Zone Control Register (TZCTL)

The trip-zone control register (TZCTL) is shown in Figure 14-82 and described in Table 14-73.

Figure 14-82. Trip-Zone Control Register (TZCTL)

15	4	3	2	1	0
Reserved		TZB		TZA	
R-0		R/W-0		R/W-0	

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 14-73. Trip-Zone Control Register (TZCTL) Field Descriptions

Bits	Name	Value	Description
15–4	Reserved	0	Reserved
3–2	TZB	0-3h	When a trip event occurs the following action is taken on output EPWMxB. Which trip-zone pins can cause an event is defined in the TZSEL register (Section 14.4.6.1).
		0	High impedance (EPWMxB = High-impedance state)
		1h	Force EPWMxB to a high state
		2h	Force EPWMxB to a low state
		3h	Do nothing, no action is taken on EPWMxB.
1–0	TZA	0-3h	When a trip event occurs the following action is taken on output EPWMxA. Which trip-zone pins can cause an event is defined in the TZSEL register (Section 14.4.6.1).
		0	High impedance (EPWMxA = High-impedance state)
		1h	Force EPWMxA to a high state
		2h	Force EPWMxA to a low state
		3h	Do nothing, no action is taken on EPWMxA.

14.4.6.3 Trip-Zone Enable Interrupt Register (TZEINT)

The trip-zone enable interrupt register (TZEINT) is shown in Figure 14-83 and described in Table 14-74.

Figure 14-83. Trip-Zone Enable Interrupt Register (TZEINT)

15	3	2	1	0
Reserved		OST	CBC	Rsvd
R-0		R/W-0	R/W-0	R-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 14-74. Trip-Zone Enable Interrupt Register (TZEINT) Field Descriptions

Bits	Name	Value	Description
15-3	Reserved	0	Reserved
2	OST		Trip-zone One-Shot Interrupt Enable
		0	Disable one-shot interrupt generation
		1	Enable Interrupt generation; a one-shot trip event will cause a EPWMxTZINT interrupt.
1	CBC		Trip-zone Cycle-by-Cycle Interrupt Enable
		0	Disable cycle-by-cycle interrupt generation.
		1	Enable interrupt generation; a cycle-by-cycle trip event will cause an EPWMxTZINT interrupt.
0	Reserved	0	Reserved

Copyright © 2013–2016, Texas Instruments Incorporated

14.4.6.4 Trip-Zone Flag Register (TZFLG)

The trip-zone flag register (TZFLG) is shown in Figure 14-84 and described in Table 14-75.

Figure 14-84. Trip-Zone Flag Register (TZFLG)

15	3	2	1	0
Reserved		OST	CBC	INT
R-0		R-0	R-0	R-0

LEGEND: R/W = Read/Write; R = Read only; *-n* = value after reset

Table 14-75. Trip-Zone Flag Register (TZFLG) Field Descriptions

Bits	Name	Value	Description
15-3	Reserved	0	Reserved
2	OST		Latched Status Flag for A One-Shot Trip Event.
		0	No one-shot trip event has occurred.
		1	Indicates a trip event has occurred on a pin selected as a one-shot trip source. This bit is cleared by writing the appropriate value to the TZCLR register (Section 14.4.6.5).
1	CBC		Latched Status Flag for Cycle-By-Cycle Trip Event
		0	No cycle-by-cycle trip event has occurred.
		1	Indicates a trip event has occurred on a pin selected as a cycle-by-cycle trip source. The TZFLG[CBC] bit will remain set until it is manually cleared by the user. If the cycle-by-cycle trip event is still present when the CBC bit is cleared, then CBC will be immediately set again. The specified condition on the pins is automatically cleared when the ePWM time-base counter reaches zero (TBCNT = 0000h) if the trip condition is no longer present. The condition on the pins is only cleared when the TBCNT = 0000h no matter where in the cycle the CBC flag is cleared. This bit is cleared by writing the appropriate value to the TZCLR register (Section 14.4.6.5).
0	INT		Latched Trip Interrupt Status Flag
		0	Indicates no interrupt has been generated.
		1	Indicates an EPWMxTZINT interrupt was generated because of a trip condition. No further EPWMxTZINT interrupts will be generated until this flag is cleared. If the interrupt flag is cleared when either CBC or OST is set, then another interrupt pulse will be generated. Clearing all flag bits will prevent further interrupts. This bit is cleared by writing the appropriate value to the TZCLR register (Section 14.4.6.5).

Copyright © 2013–2016, Texas Instruments Incorporated

14.4.6.5 Trip-Zone Clear Register (TZCLR)

The trip-zone clear register (TZCLR) is shown in Figure 14-85 and described in Table 14-76.

Figure 14-85. Trip-Zone Clear Register (TZCLR)

15	3	2	1	0
Reserved		OST	CBC	INT
R-0		R/W-0	R/W-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 14-76. Trip-Zone Clear Register (TZCLR) Field Descriptions

Bits	Name	Value	Description
15-3	Reserved	0	Reserved
2	OST		Clear Flag for One-Shot Trip (OST) Latch
		0	Has no effect. Always reads back a 0.
		1	Clears this Trip (set) condition.
1	CBC		Clear Flag for Cycle-By-Cycle (CBC) Trip Latch
		0	Has no effect. Always reads back a 0.
		1	Clears this Trip (set) condition.
0	INT		Global Interrupt Clear Flag
		0	Has no effect. Always reads back a 0.
		1	Clears the trip-interrupt flag for this ePWM module (TZFLG[INT]).
			NOTE: No further EPWMxTZINT interrupts will be generated until the flag is cleared. If the TZFLG[INT] bit is cleared and any of the other flag bits are set, then another interrupt pulse will be generated. Clearing all flag bits will prevent further interrupts.

14.4.6.6 Trip-Zone Force Register (TZFRC)

The trip-zone force register (TZFRC) is shown in Figure 14-86 and described in Table 14-77.

Figure 14-86. Trip-Zone Force Register (TZFRC)

15	3	2	1	0
Reserved		OST	CBC	Rsvd
R-0		R/W-0	R/W-0	R-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 14-77. Trip-Zone Force Register (TZFRC) Field Descriptions

Bits	Name	Value	Description
15-3	Reserved	0	Reserved
2	OST		Force a One-Shot Trip Event via Software
		0	Writing of 0 is ignored. Always reads back a 0.
		1	Forces a one-shot trip event and sets the TZFLG[OST] bit.
1	CBC		Force a Cycle-by-Cycle Trip Event via Software
		0	Writing of 0 is ignored. Always reads back a 0.
		1	Forces a cycle-by-cycle trip event and sets the TZFLG[CBC] bit.
0	Reserved	0	Reserved

Copyright © 2013–2016, Texas Instruments Incorporated

14.4.7 Event-Trigger Submodule Registers

Table 14-78 lists the memory-mapped registers for the event-trigger submodule. See your device-specific data manual for the memory address of these registers. All other register offset addresses not listed in Table 14-78 should be considered as reserved locations and the register contents should not be modified.

Table 14-78. Event-Trigger Submodule Registers

Offset	Acronym	Register Description	Section
32h	ETSEL	Event-Trigger Selection Register	Section 14.4.7.1
34h	ETPS	Event-Trigger Prescale Register	Section 14.4.7.2
36h	ETFLG	Event-Trigger Flag Register	Section 14.4.7.3
38h	ETCLR	Event-Trigger Clear Register	Section 14.4.7.4
3Ah	ETFRC	Event-Trigger Force Register	Section 14.4.7.5

14.4.7.1 Event-Trigger Selection Register (ETSEL)

The event-trigger selection register (ETSEL) is shown in Figure 14-87 and described in Table 14-79.

Figure 14-87. Event-Trigger Selection Register (ETSEL)

15	4	3	2	0
Reserved		INTEN	INTSEL	
R-0		R/W-0	R/W-0	

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 14-79. Event-Trigger Selection Register (ETSEL) Field Descriptions

Bits	Name	Value	Description
15-4	Reserved	0	Reserved
3	INTEN		Enable ePWM Interrupt (EPWMx_INT) Generation
		0	Disable EPWMx_INT generation
		1	Enable EPWMx_INT generation
2-0	INTSEL	0-7h	ePWM Interrupt (EPWMx_INT) Selection Options
		0	Reserved
		1h	Enable event time-base counter equal to zero. (TBCNT = 0000h)
		2h	Enable event time-base counter equal to period (TBCNT = TBPRD)
		3h	Reserved
		4h	Enable event time-base counter equal to CMPA when the timer is incrementing.
		5h	Enable event time-base counter equal to CMPA when the timer is decrementing.
		6h	Enable event: time-base counter equal to CMPB when the timer is incrementing.
		7h	Enable event: time-base counter equal to CMPB when the timer is decrementing.

Copyright © 2013–2016, Texas Instruments Incorporated

14.4.7.2 Event-Trigger Prescale Register (ETPS)

The event-trigger prescale register (ETPS) is shown in Figure 14-88 and described in Table 14-80.

Figure 14-88. Event-Trigger Prescale Register (ETPS)

15	4	3	2	1	0
Reserved		INTCNT		INTPRD	
R-0		R-0		R/W-0	

LEGEND: R/W = Read/Write; R = Read only; *-n* = value after reset

Table 14-80. Event-Trigger Prescale Register (ETPS) Field Descriptions

Bits	Name	Value	Description
15-4	Reserved	0	Reserved
3-2	INTCNT	0-3h	ePWM Interrupt Event (EPWMx_INT) Counter Register. These bits indicate how many selected ETSEL[INTSEL] events have occurred. These bits are automatically cleared when an interrupt pulse is generated. If interrupts are disabled, ETSEL[INT] = 0 or the interrupt flag is set, ETFLG[INT] = 1, the counter will stop counting events when it reaches the period value ETPS[INTCNT] = ETPS[INTPRD].
		0	No events have occurred.
		1h	1 event has occurred.
		2h	2 events have occurred.
		3h	3 events have occurred.
1-0	INTPRD	0-3h	ePWM Interrupt (EPWMx_INT) Period Select. These bits determine how many selected ETSEL[INTSEL] events need to occur before an interrupt is generated. To be generated, the interrupt must be enabled (ETSEL[INT] = 1). If the interrupt status flag is set from a previous interrupt (ETFLG[INT] = 1) then no interrupt will be generated until the flag is cleared via the ETCLR[INT] bit. This allows for one interrupt to be pending while another is still being serviced. Once the interrupt is generated, the ETPS[INTCNT] bits will automatically be cleared. Writing a INTPRD value that is the same as the current counter value will trigger an interrupt if it is enabled and the status flag is clear. Writing a INTPRD value that is less than the current counter value will result in an undefined state. If a counter event occurs at the same instant as a new zero or non-zero INTPRD value is written, the counter is incremented.
		0	Disable the interrupt event counter. No interrupt will be generated and ETFRC[INT] is ignored.
		1h	Generate an interrupt on the first event INTCNT = 01 (first event)
		2h	Generate interrupt on ETPS[INTCNT] = 1,0 (second event)
		3h	Generate interrupt on ETPS[INTCNT] = 1,1 (third event)

Copyright © 2013–2016, Texas Instruments Incorporated

14.4.7.3 Event-Trigger Flag Register (ETFLG)

The event-trigger flag register (ETFLG) is shown in Figure 14-89 and described in Table 14-81.

Figure 14-89. Event-Trigger Flag Register (ETFLG)

15	1	0
Reserved		INT
R-0		R-0

LEGEND: R = Read only; -*n* = value after reset

Table 14-81. Event-Trigger Flag Register (ETFLG) Field Descriptions

Bits	Name	Value	Description
15-1	Reserved	0	Reserved
0	INT		Latched ePWM Interrupt (EPWMx_INT) Status Flag
		0	Indicates no event occurred
		1	Indicates that an ePWMx interrupt (EWPMx_INT) was generated. No further interrupts will be generated until the flag bit is cleared. Up to one interrupt can be pending while the ETFLG[INT] bit is still set. If an interrupt is pending, it will not be generated until after the ETFLG[INT] bit is cleared. Refer to Figure 14-42.

14.4.7.4 Event-Trigger Clear Register (ETCLR)

The event-trigger clear register (ETCLR) is shown in Figure 14-90 and described in Table 14-82.

Figure 14-90. Event-Trigger Clear Register (ETCLR)

15	1	0
Reserved		INT
R-0		R-0

LEGEND: R = Read only; -*n* = value after reset

Table 14-82. Event-Trigger Clear Register (ETCLR) Field Descriptions

Bits	Name	Value	Description
15-1	Reserved	0	Reserved
0	INT		ePWM Interrupt (EPWMx_INT) Flag Clear Bit
		0	Writing a 0 has no effect. Always reads back a 0.
		1	Clears the ETFLG[INT] flag bit and enable further interrupts pulses to be generated.

Copyright © 2013–2016, Texas Instruments Incorporated

14.4.7.5 Event-Trigger Force Register (ETFRC)

The event-trigger force register (ETFRC) is shown in Figure 14-91 and described in Table 14-83.

Figure 14-91. Event-Trigger Force Register (ETFRC)

15	1	0
Reserved		INT
R-0		R-0

LEGEND: R = Read only; -*n* = value after reset

Table 14-83. Event-Trigger Force Register (ETFRC) Field Descriptions

Bits	Name	Value	Description
15-1	Reserved	0	Reserved
0	INT		INT Force Bit. The interrupt will only be generated if the event is enabled in the ETSEL register. The INT flag bit will be set regardless.
		0	Writing 0 to this bit will be ignored. Always reads back a 0.
		1	Generates an interrupt on $\overline{\text{EPWMxINT}}$ and set the INT flag bit. This bit is used for test purposes.

14.4.8 High-Resolution PWM Submodule Registers

Table 14-84 lists the memory-mapped registers for the high-resolution PWM submodule. See your device-specific data manual for the memory address of these registers. All other register offset addresses not listed in Table 14-84 should be considered as reserved locations and the register contents should not be modified.

Table 14-84. High-Resolution PWM Submodule Registers

Offset	Acronym	Register Description	Section
4h	TBPHSHR	Time-Base Phase High-Resolution Register	Section 14.4.8.1
10h	CMPAHR	Counter-Compare A High-Resolution Register	Section 14.4.8.2
1040h	HRCNFG	HRPWM Configuration Register	Section 14.4.8.3

Copyright © 2013–2016, Texas Instruments Incorporated

14.4.8.1 Time-Base Phase High-Resolution Register (TBPHSHR)

The time-base phase high-resolution register (TBPHSHR) is shown in Figure 14-92 and described in Table 14-85.

Figure 14-92. Time-Base Phase High-Resolution Register (TBPHSHR)

15	8	7	0
TBPHSH		Reserved	
R/W-0		R/W-0	

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 14-85. Time-Base Phase High-Resolution Register (TBPHSHR) Field Descriptions

Bit	Field	Value	Description
15-8	TBPHSH	0-FFh	Time-base phase high-resolution bits
7-0	Reserved	0	Reserved

14.4.8.2 Counter-Compare A High-Resolution Register (CMPAHR)

The counter-compare A high-resolution register (CMPAHR) is shown in Figure 14-93 and described in Table 14-86.

Figure 14-93. Counter-Compare A High-Resolution Register (CMPAHR)

15	8	7	0
CMPAHR		Reserved	
R/W-0		R/W-0	

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 14-86. Counter-Compare A High-Resolution Register (CMPAHR) Field Descriptions

Bit	Field	Value	Description
15-8	CMPAHR	1-FFh	Compare A High-Resolution register bits for MEP step control. A minimum value of 1h is needed to enable HRPWM capabilities. Valid MEP range of operation 1-255h.
7-0	Reserved	0	Reserved

Copyright © 2013–2016, Texas Instruments Incorporated

14.4.8.3 HRPWM Configuration Register (HRCNFG)

The HRPWM configuration register (HRCNFG) is shown in Figure 14-94 and described in Table 14-87.

Figure 14-94. HRPWM Configuration Register (HRCNFG)

15	4	3	2	1	0
Reserved		HRLOAD	CTLMODE	EDGMODE	
R-0		R/W-0	R/W-0	R/W-0	

LEGEND: R/W = Read/Write; R = Read only; *-n* = value after reset

Table 14-87. HRPWM Configuration Register (HRCNFG) Field Descriptions

Bit	Field	Value	Description
15-4	Reserved	0	ReserveD
3	HRLOAD		Shadow mode bit: Selects the time event that loads the CMPAHR shadow value into the active register:
		0	CTR = 0 (counter equals zero)
		1	CTR = PRD (counter equal period)
			Note: Load mode selection is valid only if CTLMODE = 0 has been selected. You should select this event to match the selection of the CMPA load mode (CMPCTL[LOADMODE] bits) in the EPWM module as follows:
			0 Load on CTR = 0: Time-base counter equal to zero (TBCNT = 0000h)
			1h Load on CTR = PRD: Time-base counter equal to period (TBCNT = TBPRD)
			2h Load on either CTR = 0 or CTR = PRD (should not be used with HRPWM)
			3h Freeze (no loads possible – should not be used with HRPWM)
2	CTLMODE		Control Mode Bits: Selects the register (CMP or TBPHS) that controls the MEP:
		0	CMPAHR(8) Register controls the edge position (this is duty control mode). (default on reset)
		1	TBPHSHR(8) Register controls the edge position (this is phase control mode).
1-0	EDGMODE	0-3h	Edge Mode Bits: Selects the edge of the PWM that is controlled by the micro-edge position (MEP) logic:
		0	HRPWM capability is disabled (default on reset)
		1h	MEP control of rising edge
		2h	MEP control of falling edge
		3h	MEP control of both edges

Copyright © 2013–2016, Texas Instruments Incorporated

Enhanced Quadrature Encoder Pulse (eQEP) Module

The enhanced quadrature encoder pulse (eQEP) module is used for direct interface with a linear or rotary incremental encoder to get position, direction, and speed information from a rotating machine for use in a high-performance motion and position-control system. This chapter describes the eQEP.

Topic **Page**

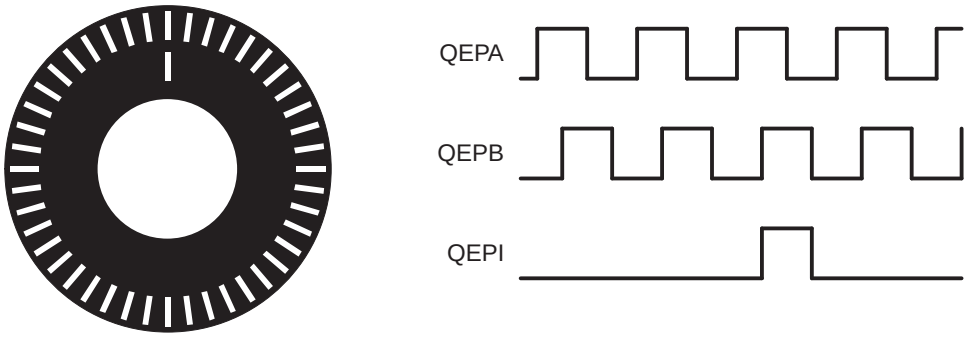
15.1 Introduction **337**
15.2 Architecture **340**
15.3 eQEP Registers **358**

Copyright © 2013–2016, Texas Instruments Incorporated

15.1 Introduction

A single track of slots patterns the periphery of an incremental encoder disk, as shown in Figure 15-1. These slots create an alternating pattern of dark and light lines. The disk count is defined as the number of dark/light line pairs that occur per revolution (lines per revolution). As a rule, a second track is added to generate a signal that occurs once per revolution (index signal: QEPI), which can be used to indicate an absolute position. Encoder manufacturers identify the index pulse using different terms such as index, marker, home position, and zero reference.

Figure 15-1. Optical Encoder Disk

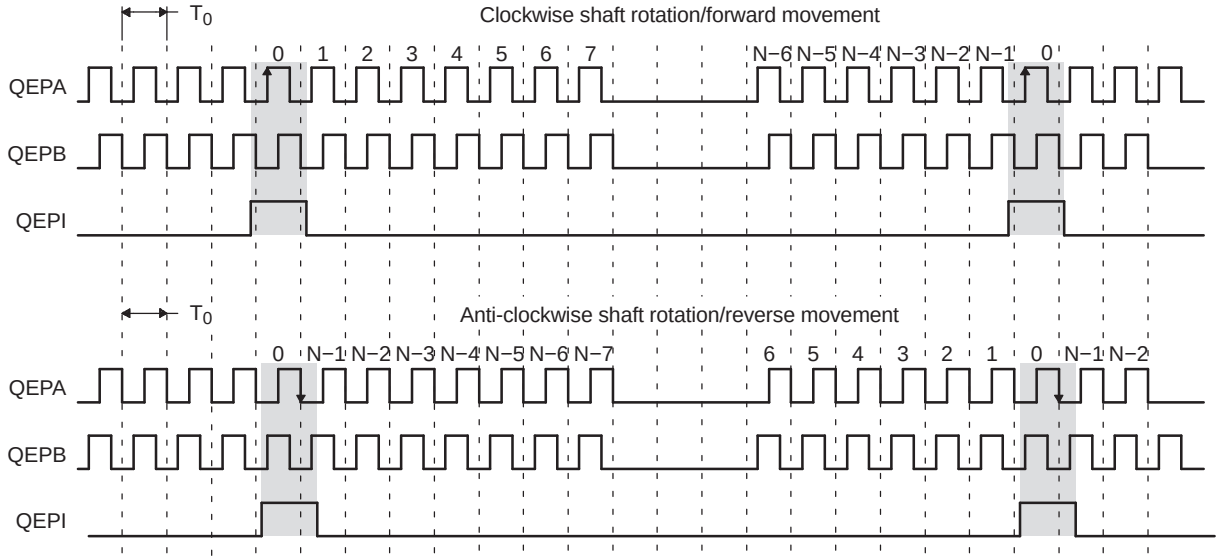


To derive direction information, the lines on the disk are read out by two different photo-elements that "look" at the disk pattern with a mechanical shift of 1/4 the pitch of a line pair between them. This shift is realized with a reticle or mask that restricts the view of the photo-element to the desired part of the disk lines. As the disk rotates, the two photo-elements generate signals that are shifted 90 degrees out of phase from each other. These are commonly called the quadrature QEPA and QEPB signals. The clockwise direction for most encoders is defined as the QEPA channel going positive before the QEPB channel and vise versa as shown in Figure 15-2.

The encoder wheel typically makes one revolution for every revolution of the motor or the wheel may be at a geared rotation ratio with respect to the motor. Therefore, the frequency of the digital signal coming from the QEPA and QEPB outputs varies proportionally with the velocity of the motor. For example, a 2000-line encoder directly coupled to a motor running at 5000 revolutions per minute (rpm) results in a frequency of 166.6 KHz, so by measuring the frequency of either the QEPA or QEPB output, the processor can determine the velocity of the motor.

Copyright © 2013–2016, Texas Instruments Incorporated

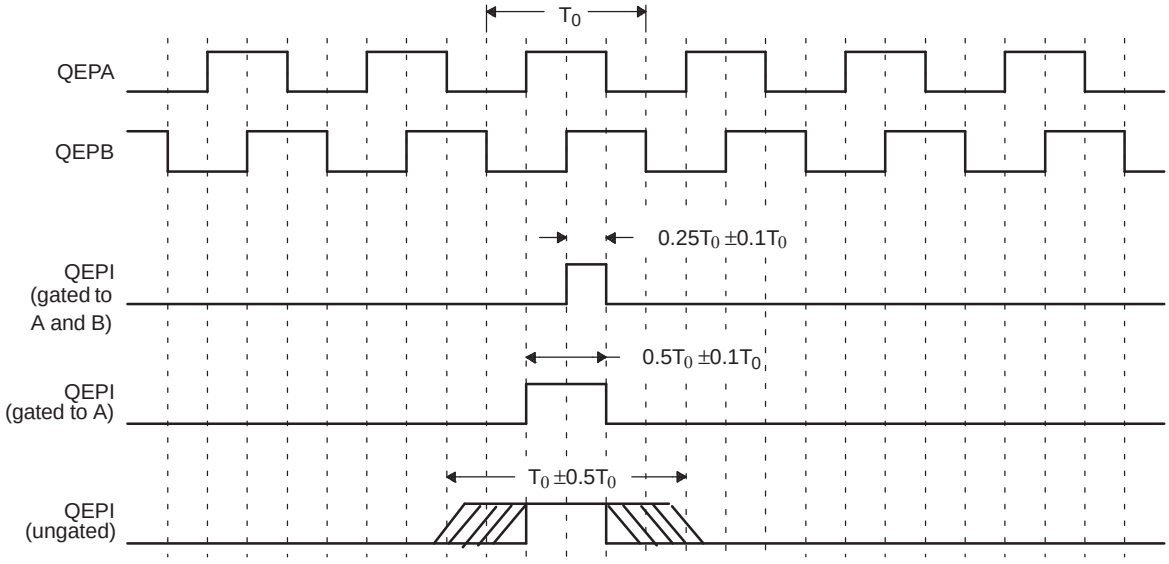
Figure 15-2. QEP Encoder Output Signal for Forward/Reverse Movement



Legend: N = lines per revolution

Quadrature encoders from different manufacturers come with two forms of index pulse (gated index pulse or ungated index pulse) as shown in Figure 15-3. A nonstandard form of index pulse is ungated. In the ungated configuration, the index edges are not necessarily coincident with A and B signals. The gated index pulse is aligned to any of the four quadrature edges and width of the index pulse and can be equal to a quarter, half, or full period of the quadrature signal.

Figure 15-3. Index Pulse Example



Copyright © 2013–2016, Texas Instruments Incorporated

Some typical applications of shaft encoders include robotics and even computer input in the form of a mouse. Inside your mouse you can see where the mouse ball spins a pair of axles (a left/right, and an up/down axle). These axles are connected to optical shaft encoders that effectively tell the computer how fast and in what direction the mouse is moving.

General Issues: Estimating velocity from a digital position sensor is a cost-effective strategy in motor control. Two different first order approximations for velocity may be written as:

$$v(k) \approx \frac{x(k) - x(k-1)}{T} = \frac{\Delta X}{T} \quad (1)$$

$$v(k) \approx \frac{X}{t(k) - t(k-1)} = \frac{X}{\Delta T} \quad (2)$$

where

v(k): Velocity at time instant k

x(k): Position at time instant k

x(k-1): Position at time instant k - 1

T: Fixed unit time or inverse of velocity calculation rate

ΔX: Incremental position movement in unit time

t(k): Time instant "k"

t(k-1): Time instant "k - 1"

X: Fixed unit position

ΔT: Incremental time elapsed for unit position movement.

Equation 1 is the conventional approach to velocity estimation and it requires a time base to provide unit time event for velocity calculation. Unit time is basically the inverse of the velocity calculation rate.

The encoder count (position) is read once during each unit time event. The quantity [x(k) - x(k-1)] is formed by subtracting the previous reading from the current reading. Then the velocity estimate is computed by multiplying by the known constant 1/T (where T is the constant time between unit time events and is known in advance).

Estimation based on Equation 1 has an inherent accuracy limit directly related to the resolution of the position sensor and the unit time period T. For example, consider a 500-line per revolution quadrature encoder with a velocity calculation rate of 400 Hz. When used for position the quadrature encoder gives a four-fold increase in resolution, in this case, 2000 counts per revolution. The minimum rotation that can be detected is therefore 0.0005 revolutions, which gives a velocity resolution of 12 rpm when sampled at 400 Hz. While this resolution may be satisfactory at moderate or high speeds, for example, 1% error at 1200 rpm, it would clearly prove inadequate at low speeds. In fact, at speeds below 12 rpm, the speed estimate would erroneously be zero much of the time.

At low speed, Equation 2 provides a more accurate approach. It requires a position sensor that outputs a fixed interval pulse train, such as the aforementioned quadrature encoder. The width of each pulse is defined by motor speed for a given sensor resolution. Equation 2 can be used to calculate motor speed by measuring the elapsed time between successive quadrature pulse edges. However, this method suffers from the opposite limitation, as does Equation 1. A combination of relatively large motor speeds and high sensor resolution makes the time interval ΔT small, and thus more greatly influenced by the timer resolution. This can introduce considerable error into high-speed estimates.

For systems with a large speed range (that is, speed estimation is needed at both low and high speeds), one approach is to use Equation 2 at low speed and have the DSP software switch over to Equation 1 when the motor speed rises above some specified threshold.

Copyright © 2013–2016, Texas Instruments Incorporated

15.2 Architecture

This section provides the eQEP inputs and functional description.

NOTE: Multiple identical eQEP modules can be contained in a system. The number of modules is device-dependent and is based on target application needs. In this document, the letter x within a signal or module name is used to indicate a generic eQEP instance on a device.

15.2.1 EQEP Inputs

The eQEP inputs include two pins for quadrature-clock mode or direction-count mode, an index (or 0 marker), and a strobe input.

- QEPA/XCLK and QEPB/XDIR: These two pins can be used in quadrature-clock mode or direction-count mode.
  - Quadrature-clock Mode: The eQEP encoders provide two square wave signals (A and B) 90 electrical degrees out of phase whose phase relationship is used to determine the direction of rotation of the input shaft and number of eQEP pulses from the index position to derive the relative position information. For forward or clockwise rotation, QEPA signal leads QEPB signal and vice versa. The quadrature decoder uses these two inputs to generate quadrature-clock and direction signals.
  - Direction-count Mode: In direction-count mode, direction and clock signals are provided directly from the external source. Some position encoders have this type of output instead of quadrature output. The QEPA pin provides the clock input and the QEPB pin provides the direction input.
- QEPI: Index or Zero Marker: The eQEP encoder uses an index signal to assign an absolute start position from which position information is incrementally encoded using quadrature pulses. This pin is connected to the index output of the eQEP encoder to optionally reset the position counter for each revolution. This signal can be used to initialize or latch the position counter on the occurrence of a desired event on the index pin.
- QEPS: Strobe Input: This general-purpose strobe signal can initialize or latch the position counter on the occurrence of a desired event on the strobe pin. This signal is typically connected to a sensor or limit switch to notify that the motor has reached a defined position.

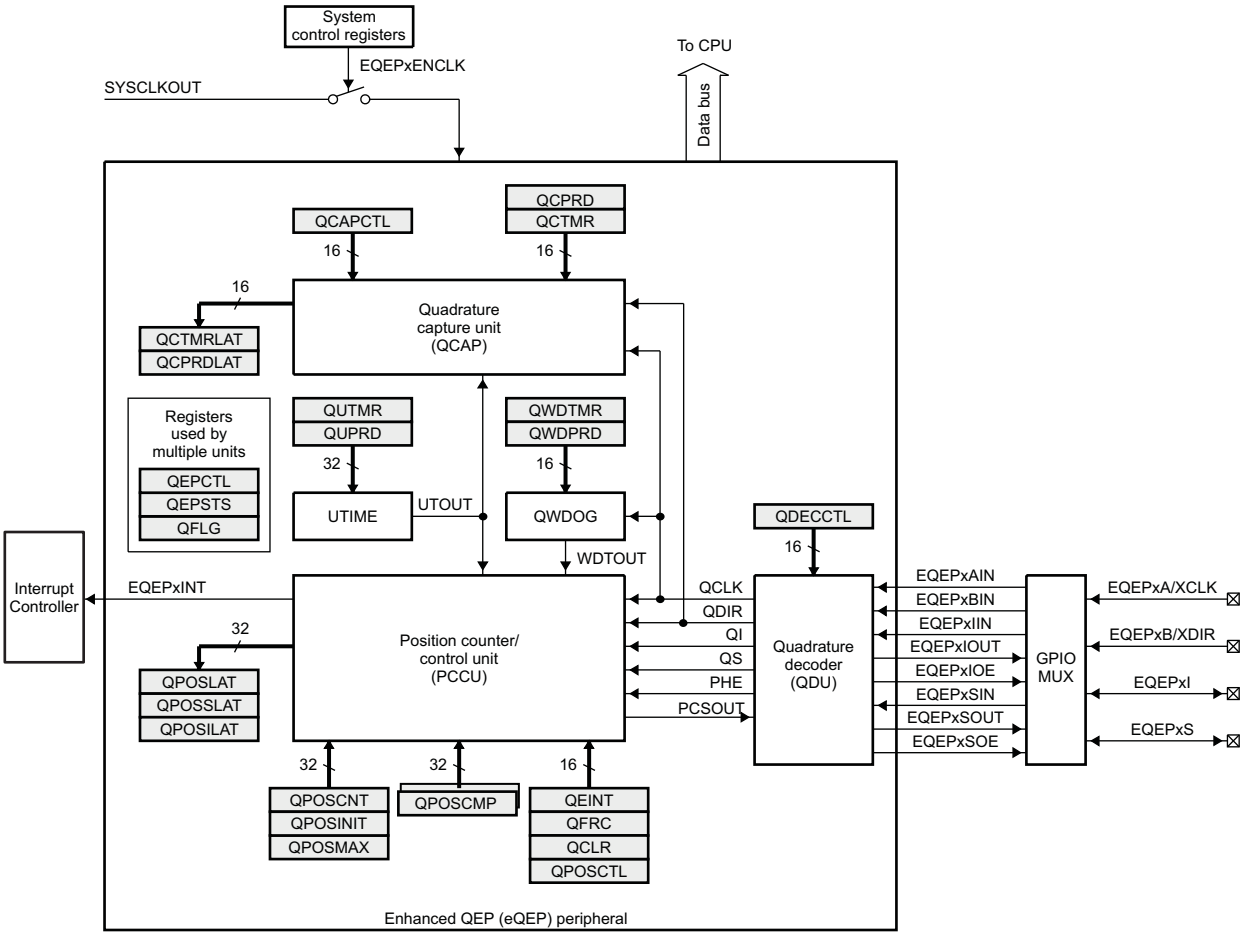
15.2.2 Functional Description

The eQEP peripheral contains the following major functional units (as shown in Figure 15-4):

- Programmable input qualification for each pin (part of the GPIO MUX)
- Quadrature decoder unit (QDU)
- Position counter and control unit for position measurement (PCCU)
- Quadrature edge-capture unit for low-speed measurement (QCAP)
- Unit time base for speed/frequency measurement (UTIME)
- Watchdog timer for detecting stalls (QWDOG)

Copyright © 2013–2016, Texas Instruments Incorporated

Figure 15-4. Functional Block Diagram of the eQEP Peripheral

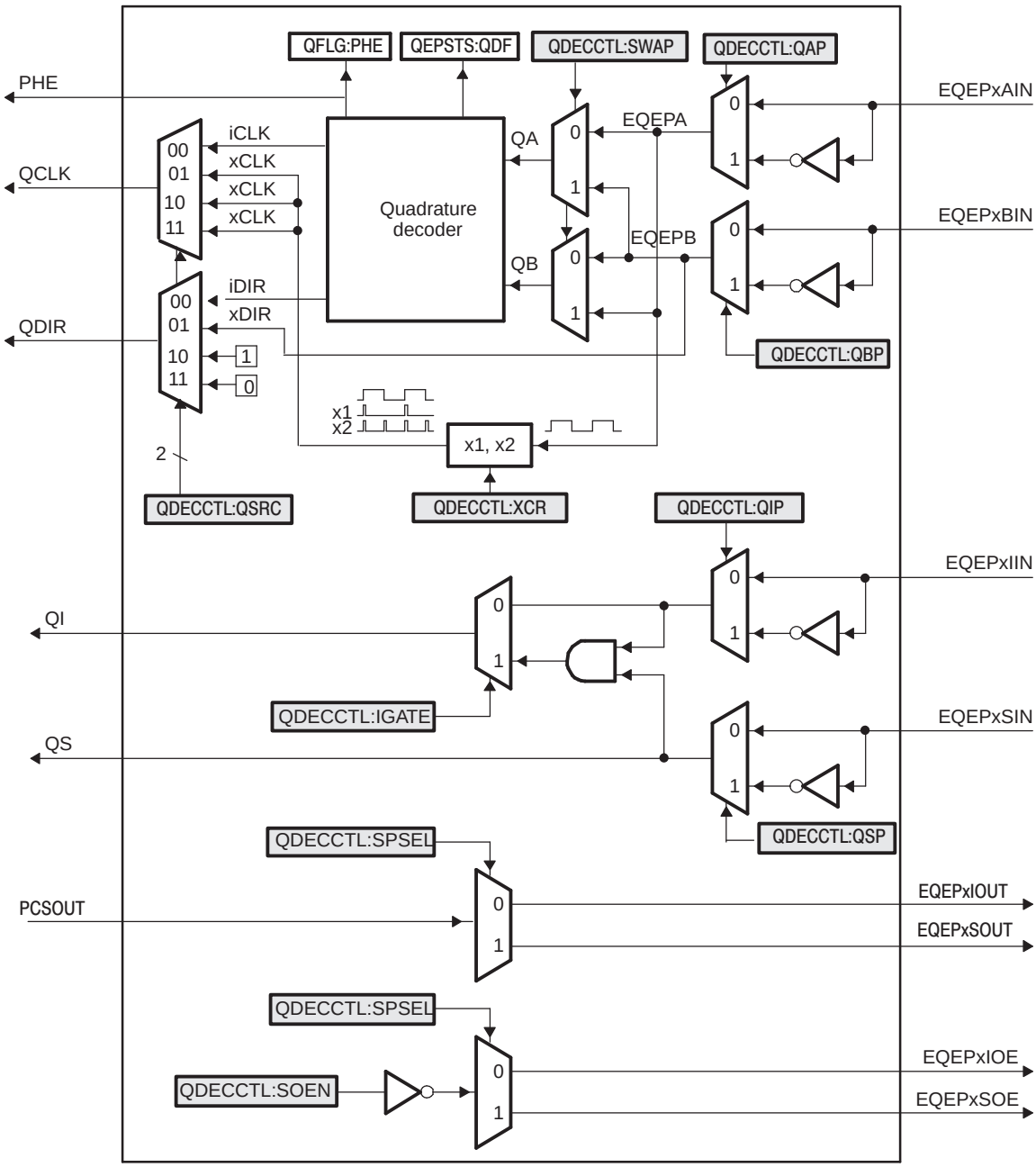


Copyright © 2013–2016, Texas Instruments Incorporated

15.2.3 Quadrature Decoder Unit (QDU)

Figure 15-5 shows a functional block diagram of the QDU.

Figure 15-5. Functional Block Diagram of Decoder Unit



Copyright © 2013–2016, Texas Instruments Incorporated

15.2.3.1 Position Counter Input Modes

Clock and direction input to position counter is selected using the QSRC bit in the eQEP decoder control register (QDECCTL), based on interface input requirement as follows:

- Quadrature-count mode
- Direction-count mode
- UP-count mode
- DOWN-count mode

15.2.3.1.1 Quadrature Count Mode

The quadrature decoder generates the direction and clock to the position counter in quadrature count mode.

Direction Decoding— The direction decoding logic of the eQEP circuit determines which one of the sequences (QEPA, QEPB) is the leading sequence and accordingly updates the direction information in the QDF bit in the eQEP status register (QEPSTS). Table 15-1 and Figure 15-6 show the direction decoding logic in truth table and state machine form. Both edges of the QEPA and QEPB signals are sensed to generate count pulses for the position counter. Therefore, the frequency of the clock generated by the eQEP logic is four times that of each input sequence. Figure 15-7 shows the direction decoding and clock generation from the eQEP input signals.

Phase Error Flag— In normal operating conditions, quadrature inputs QEPA and QEPB will be 90 degrees out of phase. The phase error flag (PHE) is set in the QFLG register when edge transition is detected simultaneously on the QEPA and QEPB signals to optionally generate interrupts. State transitions marked by dashed lines in Figure 15-6 are invalid transitions that generate a phase error.

Count Multiplication— The eQEP position counter provides 4x times the resolution of an input clock by generating a quadrature-clock (QCLK) on the rising/falling edges of both eQEP input clocks (QEPA and QEPB) as shown in Figure 15-7.

Reverse Count— In normal quadrature count operation, QEPA input is fed to the QA input of the quadrature decoder and the QEPB input is fed to the QB input of the quadrature decoder. Reverse counting is enabled by setting the SWAP bit in the eQEP decoder control register (QDECCTL). This will swap the input to the quadrature decoder thereby reversing the counting direction.

Table 15-1. Quadrature Decoder Truth Table

Previous Edge	Present Edge	QDIR	QPOSCNT
QA↑	QB↑	UP	Increment
	QB↓	DOWN	Decrement
	QA↓	TOGGLE	Increment or Decrement
QA↓	QB↓	UP	Increment
	QB↑	DOWN	Decrement
	QA↑	TOGGLE	Increment or Decrement
QB↑	QA↑	DOWN	Increment
	QA↓	UP	Decrement
	QB↓	TOGGLE	Increment or Decrement
QB↓	QA↓	DOWN	Increment
	QA↑	UP	Decrement
	QB↑	TOGGLE	Increment or Decrement

Copyright © 2013–2016, Texas Instruments Incorporated

Figure 15-6. Quadrature Decoder State Machine

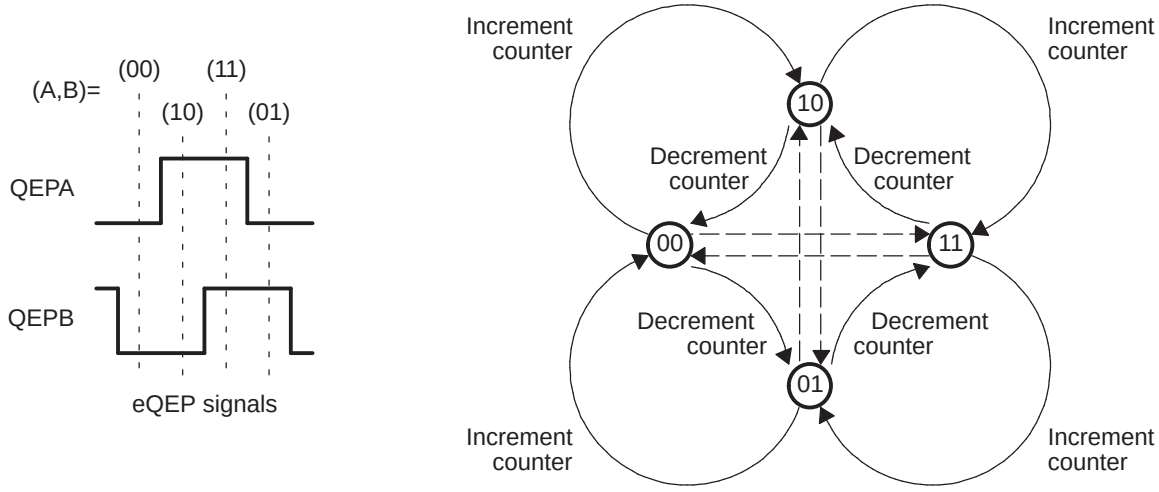
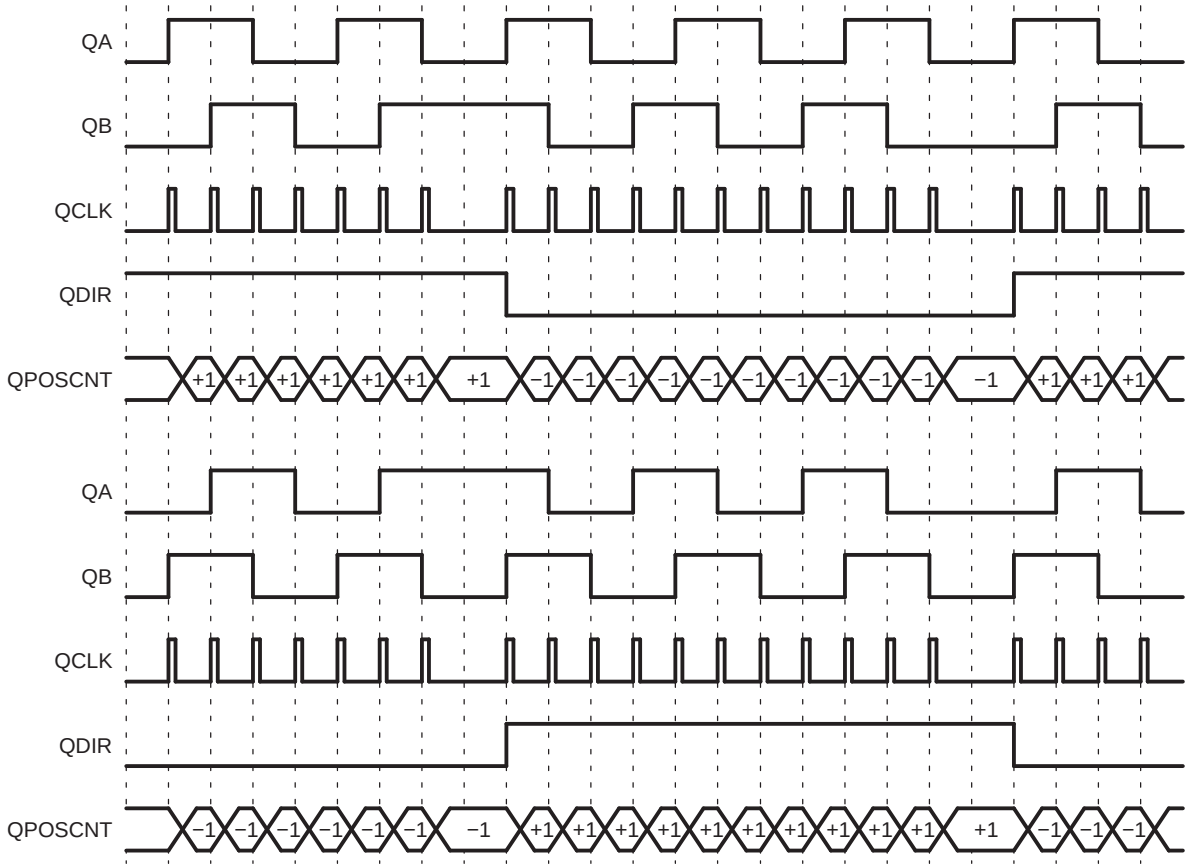


Figure 15-7. Quadrature-clock and Direction Decoding



Copyright © 2013–2016, Texas Instruments Incorporated

15.2.3.1.2 Direction-count Mode

Some position encoders provide direction and clock outputs, instead of quadrature outputs. In such cases, direction-count mode can be used. QEPA input will provide the clock for position counter and the QEPB input will have the direction information. The position counter is incremented on every rising edge of a QEPA input when the direction input is high and decremented when the direction input is low.

15.2.3.1.3 Up-Count Mode

The counter direction signal is hard-wired for up count and the position counter is used to measure the frequency of the QEPA input. Setting of the XCR bit in the eQEP decoder control register (QDECCTL) enables clock generation to the position counter on both edges of the QEPA input, thereby increasing the measurement resolution by 2× factor.

15.2.3.1.4 Down-Count Mode

The counter direction signal is hardwired for a down count and the position counter is used to measure the frequency of the QEPA input. Setting of the XCR bit in the eQEP decoder control register (QDECCTL) enables clock generation to the position counter on both edges of a QEPA input, thereby increasing the measurement resolution by 2× factor.

15.2.3.2 eQEP Input Polarity Selection

Each eQEP input can be inverted using the in the eQEP decoder control register (QDECCTL[8:5]) control bits. As an example, setting of the QIP bit in QDECCTL inverts the index input.

15.2.3.3 Position-Compare Sync Output

The eQEP peripheral includes a position-compare unit that is used to generate the position-compare sync signal on compare match between the position counter register (QPOSCNT) and the position-compare register (QPOSCMP). This sync signal can be output using an index pin or strobe pin of the EQEP peripheral.

Setting the SOEN bit in the eQEP decoder control register (QDECCTL) enables the position-compare sync output and the SPSEL bit in QDECCTL selects either an eQEP index pin or an eQEP strobe pin.

15.2.4 Position Counter and Control Unit (PCCU)

The position counter and control unit provides two configuration registers (QEPCTL and QPOSCTL) for setting up position counter operational modes, position counter initialization/latch modes and position-compare logic for sync signal generation.

15.2.4.1 Position Counter Operating Modes

Position counter data may be captured in different manners. In some systems, the position counter is accumulated continuously for multiple revolutions and the position counter value provides the position information with respect to the known reference. An example of this is the quadrature encoder mounted on the motor controlling the print head in the printer. Here the position counter is reset by moving the print head to the home position and then position counter provides absolute position information with respect to home position.

In other systems, the position counter is reset on every revolution using index pulse and position counter provides rotor angle with respect to index pulse position.

Position counter can be configured to operate in following four modes

- Position Counter Reset on Index Event
- Position Counter Reset on Maximum Position
- Position Counter Reset on the first Index Event
- Position Counter Reset on Unit Time Out Event (Frequency Measurement)

Copyright © 2013–2016, Texas Instruments Incorporated

In all the above operating modes, position counter is reset to 0 on overflow and to QPOSMAX register value on underflow. Overflow occurs when the position counter counts up after QPOSMAX value. Underflow occurs when position counter counts down after "0". Interrupt flag is set to indicate overflow/underflow in QFLG register.

15.2.4.1.1 Position Counter Reset on Index Event (QEPCTL[PCRM] = 00)

If the index event occurs during the forward movement, then position counter is reset to 0 on the next eQEP clock. If the index event occurs during the reverse movement, then the position counter is reset to the value in the QPOSMAX register on the next eQEP clock.

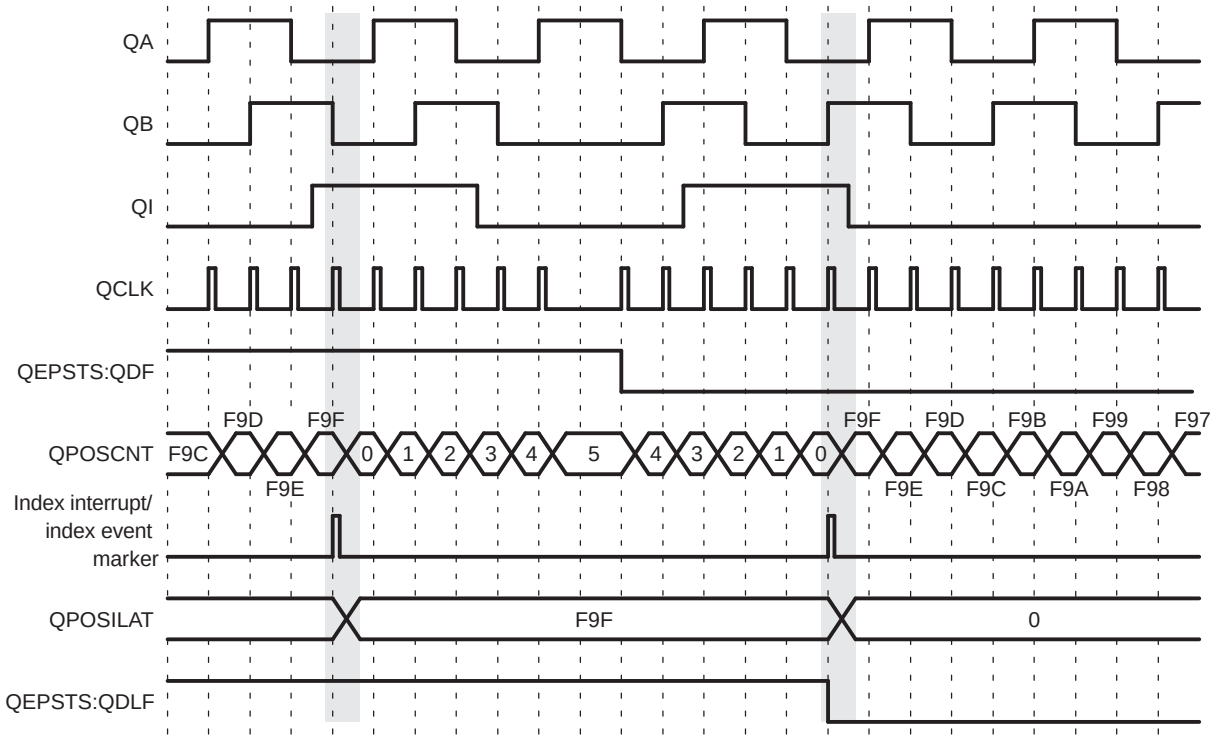
First index marker is defined as the quadrature edge following the first index edge. The eQEP peripheral records the occurrence of the first index marker (QEPSTS[FIMF]) and direction on the first index event marker (QEPSTS[FIDF]) in QEPSTS registers, it also remembers the quadrature edge on the first index marker so that same relative quadrature transition is used for index event reset operation.

For example, if the first reset operation occurs on the falling edge of QEPB during the forward direction, then all the subsequent reset must be aligned with the falling edge of QEPB for the forward rotation and on the rising edge of QEPB for the reverse rotation as shown in Figure 15-8.

The position-counter value is latched to the QPOSILAT register and direction information is recorded in the QEPSTS[QDLF] bit on every index event marker. The position-counter error flag (QEPSTS[PCEF]) and error interrupt flag (QFLG[PCE]) are set if the latched value is not equal to 0 or QPOSMAX. The position-counter error flag (QEPSTS[PCEF]) is updated on every index event marker and an interrupt flag (QFLG[PCE]) will be set on error that can be cleared only through software.

The index event latch configuration QEPCTL[IEL] bits are ignored in this mode and position counter error flag/interrupt flag are generated only in index event reset mode.

Figure 15-8. Position Counter Reset by Index Pulse for 1000 Line Encoder (QPOSMAX = 3999 or F9Fh)



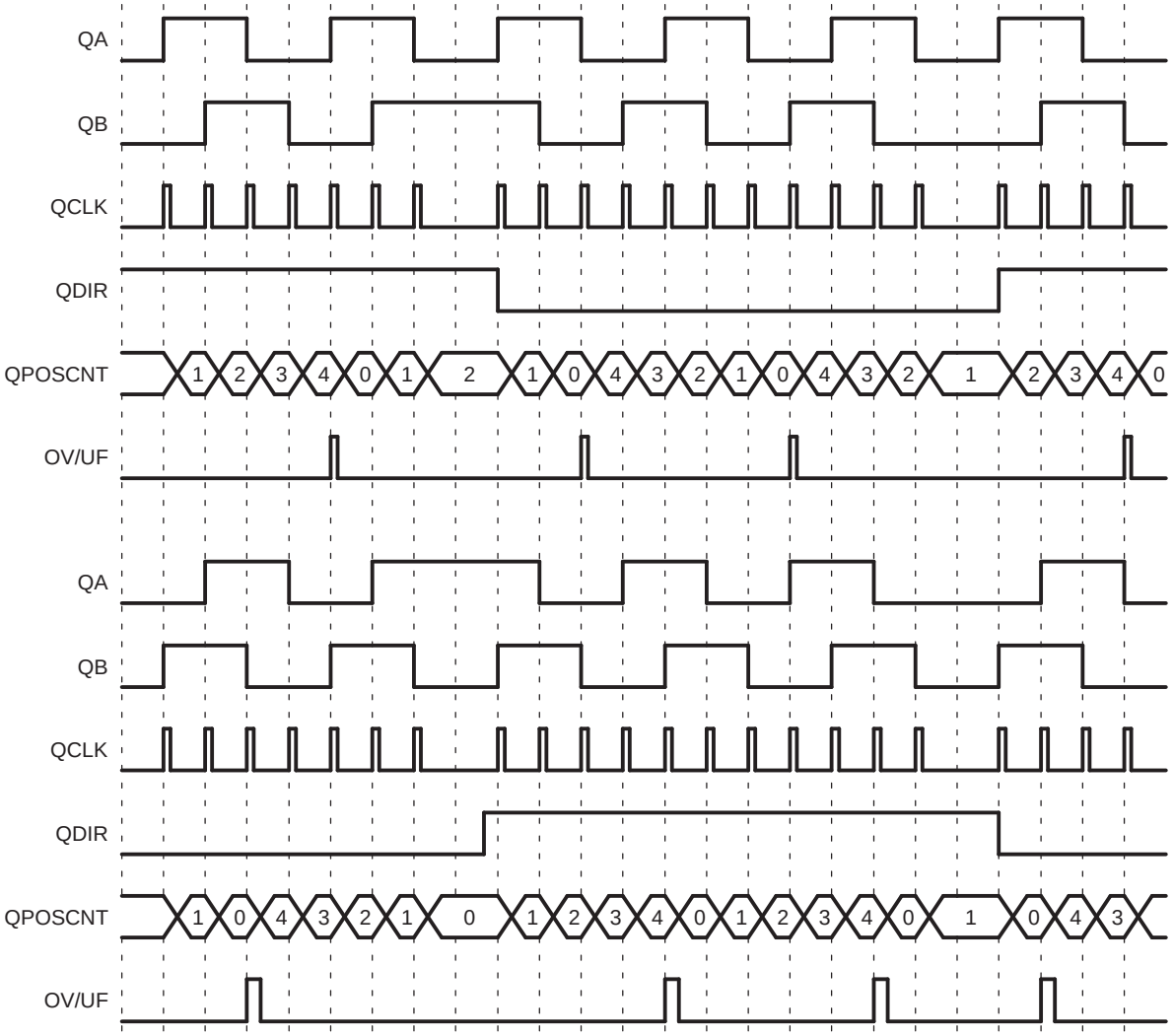
Copyright © 2013–2016, Texas Instruments Incorporated

15.2.4.1.2 Position Counter Reset on Maximum Position (QEPCTL[PCRM]=01)

If the position counter is equal to QPOSMAX, then the position counter is reset to 0 on the next eQEP clock for forward movement and position counter overflow flag is set. If the position counter is equal to ZERO, then the position counter is reset to QPOSMAX on the next QEP clock for reverse movement and position counter underflow flag is set. Figure 15-9 shows the position counter reset operation in this mode.

First index marker is defined as the quadrature edge following the first index edge. The eQEP peripheral records the occurrence of the first index marker (QEPSTS[FIMF]) and direction on the first index event marker (QEPSTS[FIDF]) in the QEPSTS registers; it also remembers the quadrature edge on the first index marker so that the same relative quadrature transition is used for the software index marker (QEPCTL[IEL]=11).

Figure 15-9. Position Counter Underflow/Overflow (QPOSMAX = 4)



Copyright © 2013–2016, Texas Instruments Incorporated

15.2.4.1.3 Position Counter Reset on the First Index Event (QEPCTL[PCRM] = 10)

If the index event occurs during forward movement, then the position counter is reset to 0 on the next eQEP clock. If the index event occurs during the reverse movement, then the position counter is reset to the value in the QPOSMAX register on the next eQEP clock. Note that this is done only on the first occurrence and subsequently the position counter value is not reset on an index event; rather, it is reset based on maximum position as described in Section 15.2.4.1.2.

First index marker is defined as the quadrature edge following the first index edge. The eQEP peripheral records the occurrence of the first index marker (QEPSTS[FIMF]) and direction on the first index event marker (QEPSTS[FIDF]) in QEPSTS registers. It also remembers the quadrature edge on the first index marker so that same relative quadrature transition is used for software index marker (QEPCTL[IEL]=11).

15.2.4.1.4 Position Counter Reset on Unit Time out Event (QEPCTL[PCRM] = 11)

In this mode, the QPOSCNT value is latched to the QPOSLAT register and then the QPOSCNT is reset (to 0 or QPOSMAX, depending on the direction mode selected by QDECCTL[QSRC] bits on a unit time event). This is useful for frequency measurement.

15.2.4.2 Position Counter Latch

The eQEP index and strobe input can be configured to latch the position counter (QPOSCNT) into QPOSILAT and QPOSSLAT, respectively, on occurrence of a definite event on these pins.

15.2.4.2.1 Index Event Latch

In some applications, it may not be desirable to reset the position counter on every index event and instead it may be required to operate the position counter in full 32-bit mode (QEPCTL[PCRM] = 01 and QEPCTL[PCRM] = 10 modes).

In such cases, the eQEP position counter can be configured to latch on the following events and direction information is recorded in the QEPSTS[QDLF] bit on every index event marker.

- Latch on Rising edge (QEPCTL[IEL] = 01)
- Latch on Falling edge (QEPCTL[IEL] = 10)
- Latch on Index Event Marker (QEPCTL[IEL] = 11)

This is particularly useful as an error checking mechanism to check if the position counter accumulated the correct number of counts between index events. As an example, the 1000-line encoder must count 4000 times when moving in the same direction between the index events.

The index event latch interrupt flag (QFLG[IEL]) is set when the position counter is latched to the QPOSILAT register. The index event latch configuration bits (QEPCTZ[IEL]) are ignored when QEPCTL[PCRM] = 00.

Latch on Rising Edge (QEPCTL[IEL] = 01)— The position counter value (QPOSCNT) is latched to the QPOSILAT register on every rising edge of an index input.

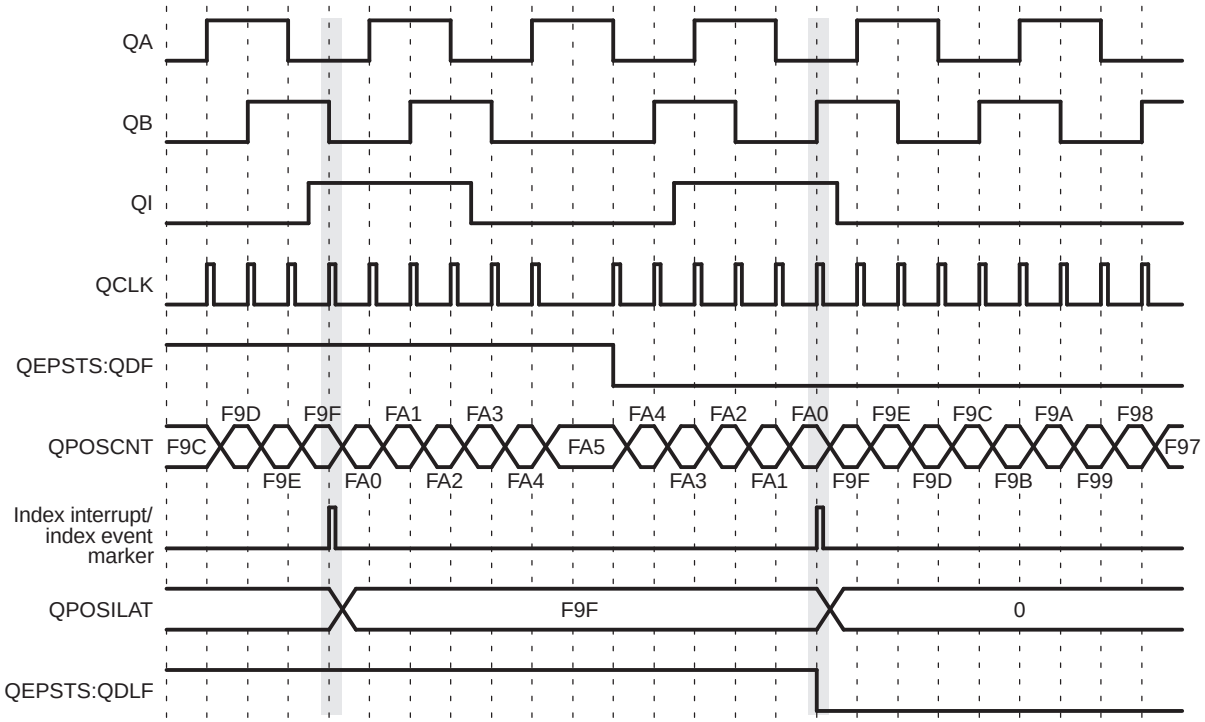
Latch on Falling Edge (QEPCTL[IEL] = 10)— The position counter value (QPOSCNT) is latched to the QPOSILAT register on every falling edge of index input.

Latch on Index Event Marker/Software Index Marker (QEPCTL[IEL] = 11)— The first index marker is defined as the quadrature edge following the first index edge. The eQEP peripheral records the occurrence of the first index marker (QEPSTS[FIMF]) and direction on the first index event marker (QEPSTS[FIDF]) in the QEPSTS registers. It also remembers the quadrature edge on the first index marker so that same relative quadrature transition is used for latching the position counter (QEPCTL[IEL] = 11).

Figure 15-10 shows the position counter latch using an index event marker.

Copyright © 2013–2016, Texas Instruments Incorporated

Figure 15-10. Software Index Marker for 1000-line Encoder (QEPCTL[IEL] = 1)



Copyright © 2013–2016, Texas Instruments Incorporated

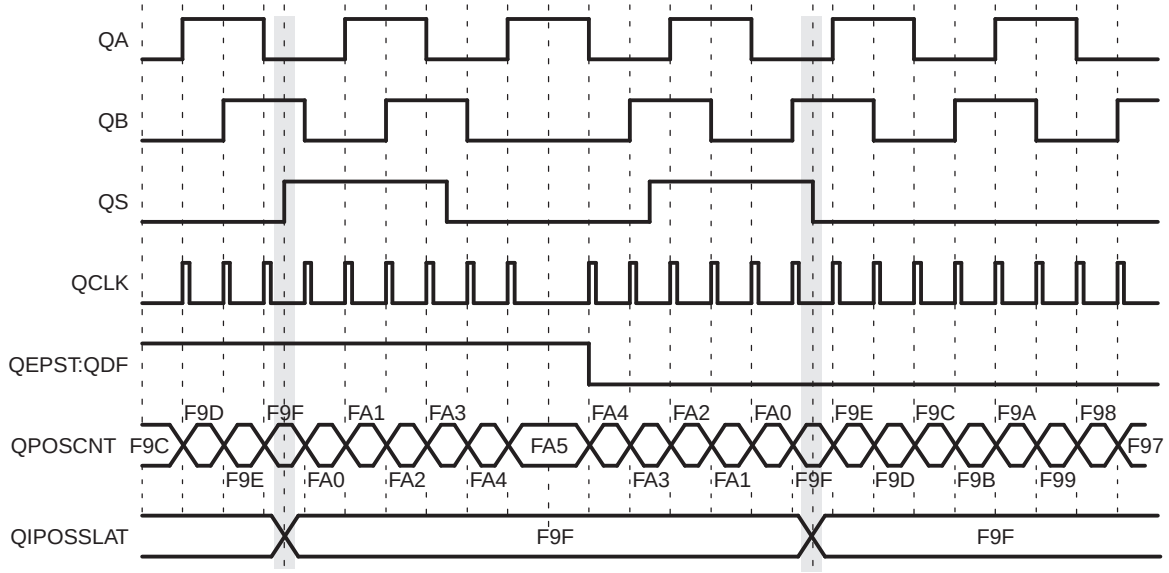
15.2.4.2.2 Strobe Event Latch

The position-counter value is latched to the QPOSSLAT register on the rising edge of the strobe input by clearing the QEPCTL[SEL] bit.

If the QEPCTL[SEL] bit is set, then the position counter value is latched to the QPOSSLAT register on the rising edge of the strobe input for forward direction and on the falling edge of the strobe input for reverse direction as shown in Figure 15-11.

The strobe event latch interrupt flag (QFLG[SEL]) is set when the position counter is latched to the QPOSSLAT register.

Figure 15-11. Strobe Event Latch (QEPCTL[SEL] = 1)



Copyright © 2013–2016, Texas Instruments Incorporated

15.2.4.3 Position Counter Initialization

The position counter can be initialized using following events:

- Index event
- Strobe event
- Software initialization

Index Event Initialization (IEI)— The QEPI index input can be used to trigger the initialization of the position counter at the rising or falling edge of the index input.

If the QEPCTL[IEI] bits are 10, then the position counter (QPOSCNT) is initialized with a value in the QPOSINIT register on the rising edge of strobe input for forward direction and on the falling edge of strobe input for reverse direction.

The index event initialization interrupt flag (QFLG[IEI]) is set when the position counter is initialized with a value in the QPOSINIT register.

Strobe Event Initialization (SEI)— If the QEPCTL[SEI] bits are 10, then the position counter is initialized with a value in the QPOSINIT register on the rising edge of strobe input.

If the QEPCTL[SEL] bits are 11, then the position counter (QPOSCNT) is initialized with a value in the QPOSINIT register on the rising edge of strobe input for forward direction and on the falling edge of strobe input for reverse direction.

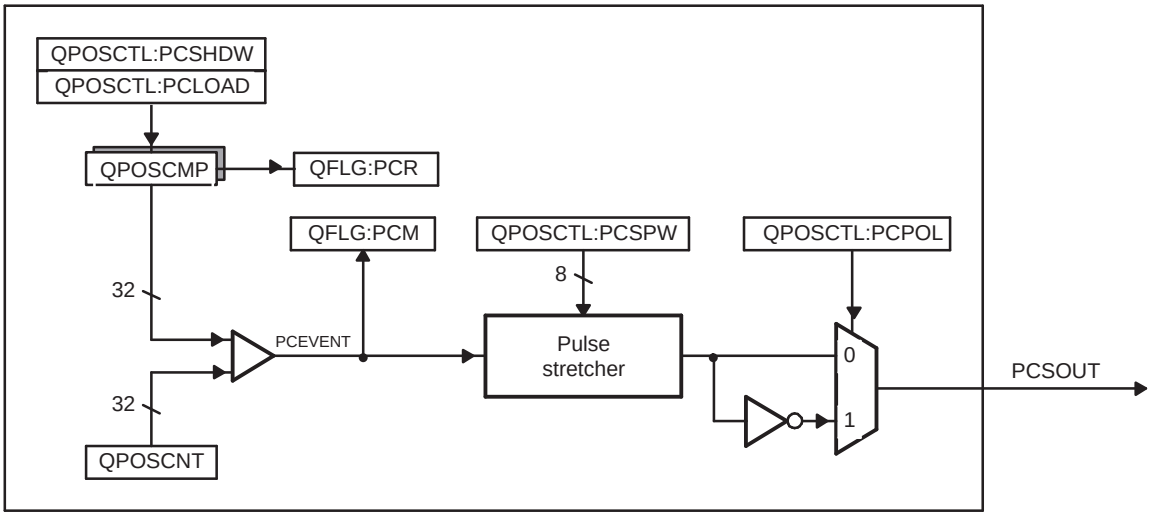
The strobe event initialization interrupt flag (QFLG[SEI]) is set when the position counter is initialized with a value in the QPOSINIT register.

Software Initialization (SWI)— The position counter can be initialized in software by writing a 1 to the QEPCTL[SWI] bit, which will automatically be cleared after initialization.

15.2.4.4 eQEP Position-compare Unit

The eQEP peripheral includes a position-compare unit that is used to generate a sync output and/or interrupt on a position-compare match. Figure 15-12 shows a diagram. The position-compare (QPOSCMP) register is shadowed and shadow mode can be enabled or disabled using the QPOSCTL[PSSHDW] bit. If the shadow mode is not enabled, the CPU writes directly to the active position compare register.

Figure 15-12. eQEP Position-compare Unit



In shadow mode, you can configure the position-compare unit (QPOSCTL[PCLOAD]) to load the shadow register value into the active register on the following events and to generate the position-compare ready (QFLG[PCR]) interrupt after loading.

- Load on compare match
- Load on position-counter zero event

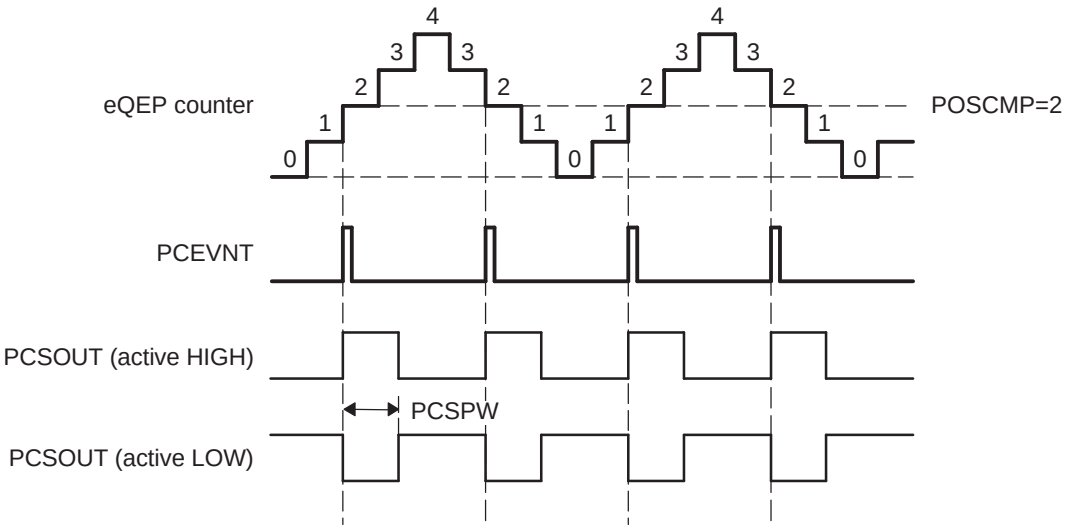
Copyright © 2013–2016, Texas Instruments Incorporated

The position-compare match (QFLG[PCM]) is set when the position-counter value (QPOSCNT) matches with the active position-compare register (QPOSCMP) and the position-compare sync output of the programmable pulse width is generated on compare match to trigger an external device.

For example, if QPOSCMP = 2, the position-compare unit generates a position-compare event on 1 to 2 transitions of the eQEP position counter for forward counting direction and on 3 to 2 transitions of the eQEP position counter for reverse counting direction (see Figure 15-13).

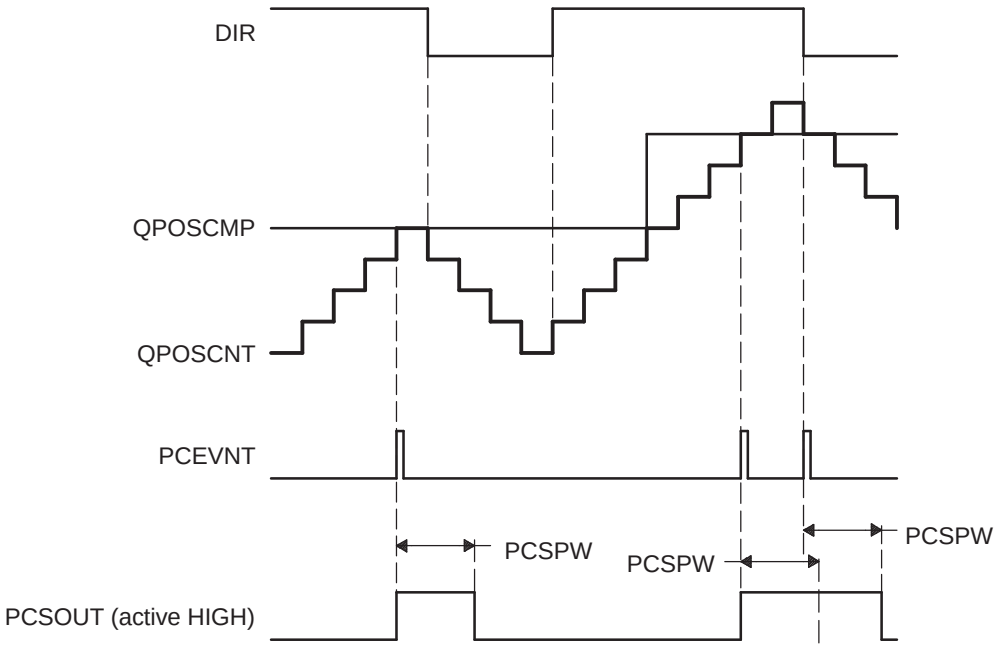
Figure 15-35 shows the layout of the eQEP Position-Compare Control Register (QPOSCTL) and Table 15-17 describes the QPOSCTL bit fields.

Figure 15-13. eQEP Position-compare Event Generation Points



The pulse stretcher logic in the position-compare unit generates a programmable position-compare sync pulse output on the position-compare match. In the event of a new position-compare match while a previous position-compare pulse is still active, then the pulse stretcher generates a pulse of specified duration from the new position-compare event as shown in Figure 15-14.

Figure 15-14. eQEP Position-compare Sync Output Pulse Stretcher



Copyright © 2013–2016, Texas Instruments Incorporated

15.2.5 eQEP Edge Capture Unit

The eQEP peripheral includes an integrated edge capture unit to measure the elapsed time between the unit position events as shown in Figure 15-15. This feature is typically used for low speed measurement using the following equation:

$$v(k) = \frac{X}{t(k) - t(k-1)} = \frac{X}{\Delta T} \quad (3)$$

where,

- X - Unit position is defined by integer multiple of quadrature edges (see Figure 15-16)
- ΔT - Elapsed time between unit position events
- v(k) - Velocity at time instant "k"

The eQEP capture timer (QCTMR) runs from prescaled SYSCLKOUT and the prescaler is programmed by the QCAPCTL[CCPS] bits. The capture timer (QCTMR) value is latched into the capture period register (QCPRD) on every unit position event and then the capture timer is reset, a flag is set in QEPSTS[UPEVNT] to indicate that new value is latched into the QCPRD register. Software can check this status flag before reading the period register for low speed measurement and clear the flag by writing 1.

Time measurement (ΔT) between unit position events will be correct if the following conditions are met:

- No more than 65,535 counts have occurred between unit position events.
- No direction change between unit position events.

The capture unit sets the eQEP overflow error flag (QEPSTS[COEF]) in the event of capture timer overflow between unit position events. If a direction change occurs between the unit position events, then an error flag is set in the status register (QEPSTS[CDEF]).

Capture Timer (QCTMR) and Capture period register (QCPRD) can be configured to latch on following events.

- CPU read of QPOSCNT register
- Unit time-out event

If the QEPCTL[QCLM] bit is cleared, then the capture timer and capture period values are latched into the QCTMRLAT and QCPRDLAT registers, respectively, when the CPU reads the position counter (QPOSCNT).

If the QEPCTL[QCLM] bit is set, then the position counter, capture timer, and capture period values are latched into the QPOSLAT, QCTMRLAT and QCPRDLAT registers, respectively, on unit time out.

Figure 15-17 shows the capture unit operation along with the position counter.

NOTE: The QCAPCTL register should not be modified dynamically (such as switching CAPCLK prescaling mode from QCLK/4 to QCLK/8). The capture unit must be disabled before changing the prescaler.

Copyright © 2013–2016, Texas Instruments Incorporated

Figure 15-15. eQEP Edge Capture Unit

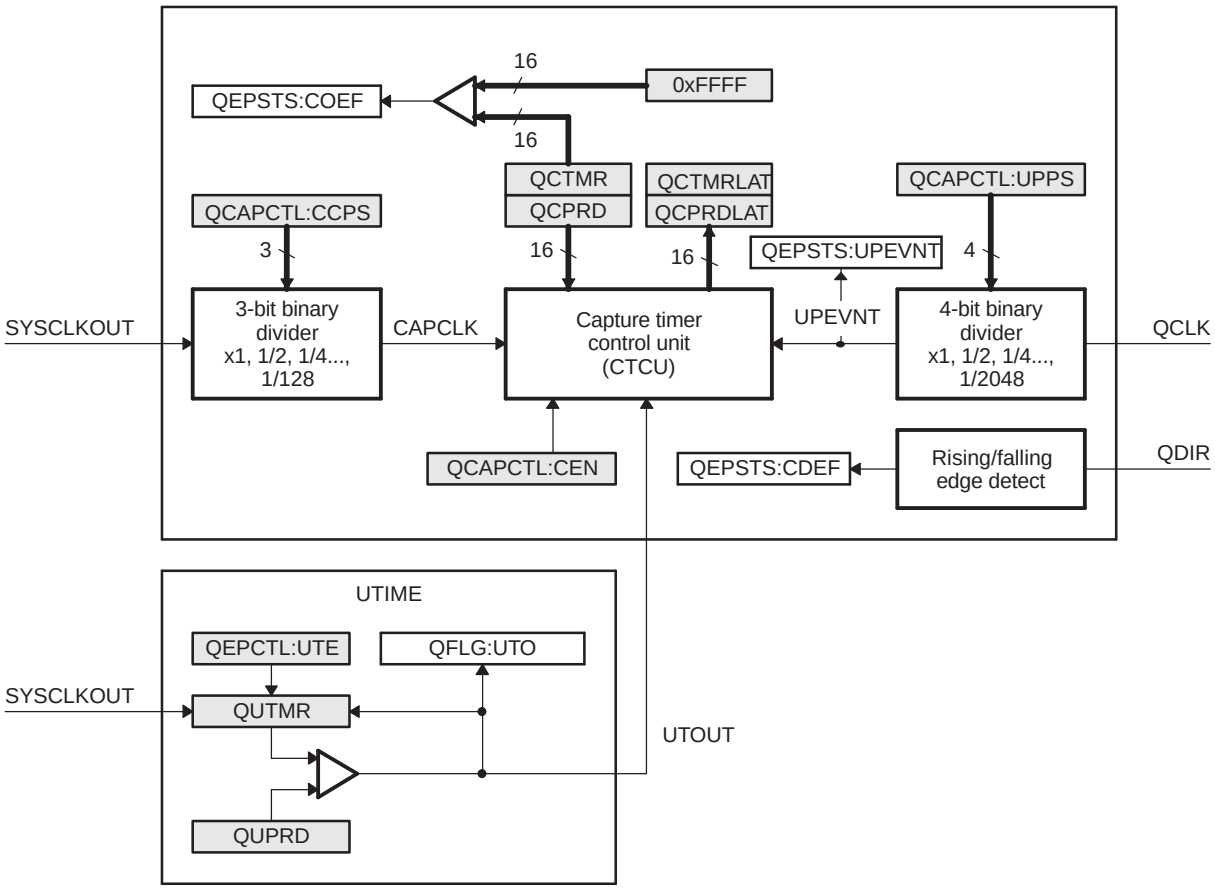
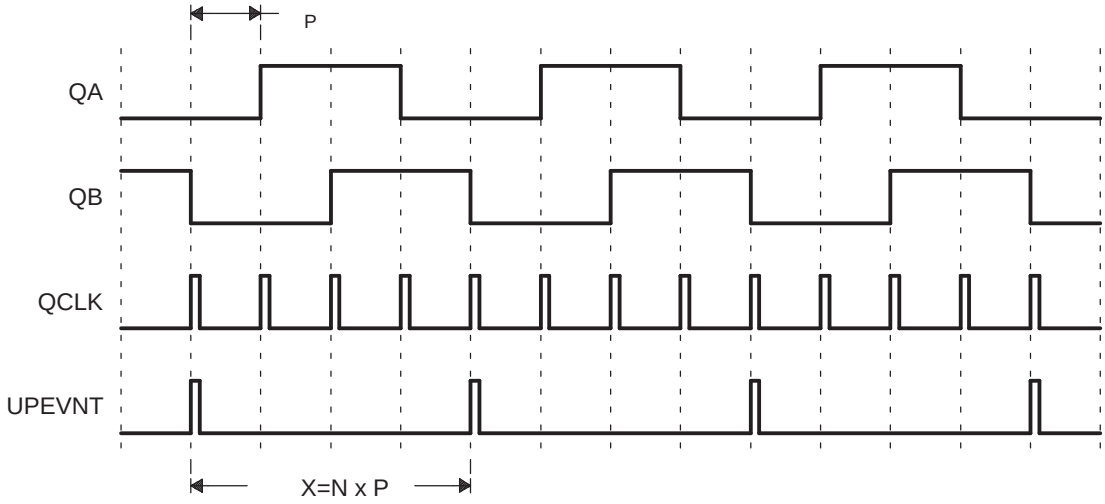


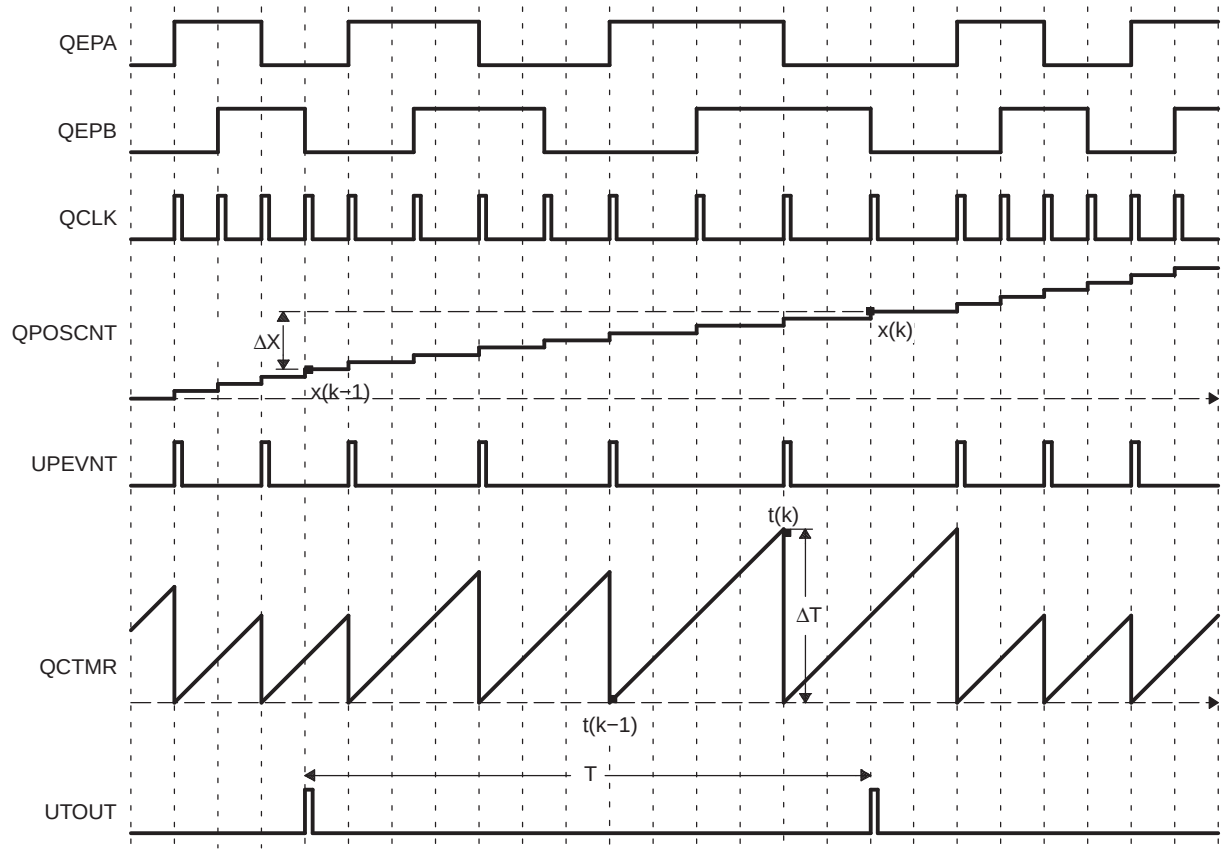
Figure 15-16. Unit Position Event for Low Speed Measurement (QCAPCTL[UPPS] = 0010)



N - Number of quadrature periods selected using QCAPCTL[UPPS] bits

Copyright © 2013–2016, Texas Instruments Incorporated

Figure 15-17. eQEP Edge Capture Unit - Timing Details



Velocity Calculation Equations:

$$v(k) = \frac{x(k) - x(k-1)}{T} = \frac{\Delta X}{T} \quad (4)$$

where

v(k): Velocity at time instant k

x(k): Position at time instant k

x(k-1): Position at time instant k - 1

T: Fixed unit time or inverse of velocity calculation rate

ΔX: Incremental position movement in unit time

X: Fixed unit position

ΔT: Incremental time elapsed for unit position movement

t(k): Time instant "k"

t(k-1): Time instant "k - 1"

Unit time (T) and unit period (X) are configured using the QUPRD and QCAPCTL[UPPS] registers. Incremental position output and incremental time output is available in the QPOSLAT and QCPRDLAT registers.

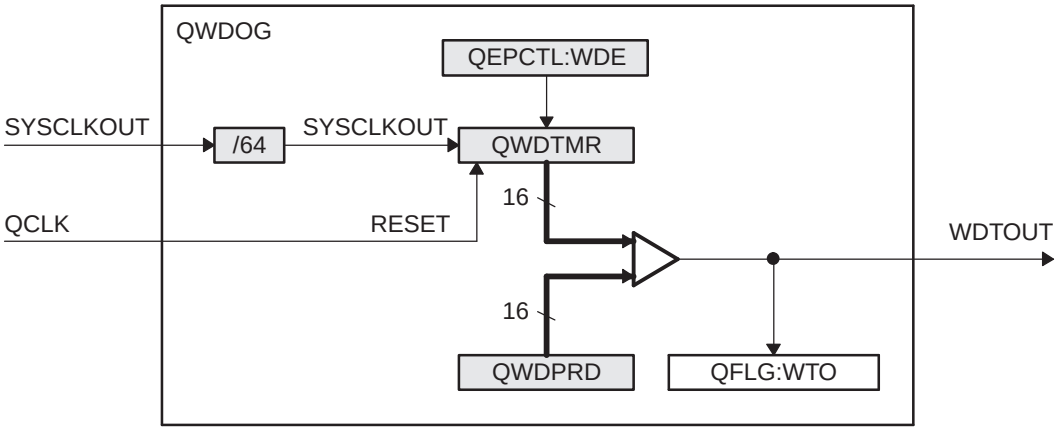
Copyright © 2013–2016, Texas Instruments Incorporated

Parameter	Relevant Register to Configure or Read the Information
T	Unit Period Register (QUPRD)
ΔX	Incremental Position = QPOSLAT(k) - QPOSLAT(K - 1)
X	Fixed unit position defined by sensor resolution and ZCAPCTL[UPPS] bits
ΔT	Capture Period Latch (QCPRDLAT)

15.2.6 eQEP Watchdog

The eQEP peripheral contains a 16-bit watchdog timer that monitors the quadrature-clock to indicate proper operation of the motion-control system. The eQEP watchdog timer is clocked from SYSCLKOUT/64 and the quadrate clock event (pulse) resets the watchdog timer. If no quadrature-clock event is detected until a period match (QWDPRD = QWDTMR), then the watchdog timer will time out and the watchdog interrupt flag will be set (QFLG[WTO]). The time-out value is programmable through the watchdog period register (QWDPRD).

Figure 15-18. eQEP Watchdog Timer



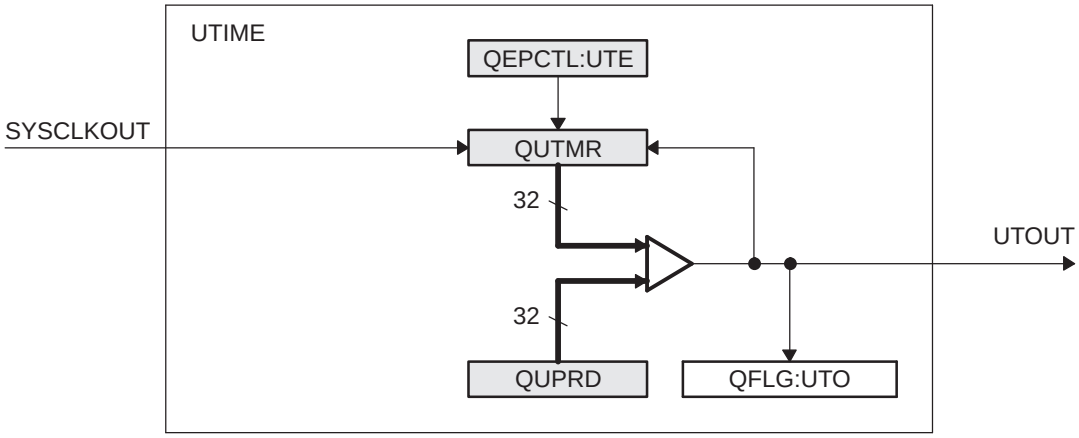
Copyright © 2013–2016, Texas Instruments Incorporated

15.2.7 Unit Timer Base

The eQEP peripheral includes a 32-bit timer (QUTMR) that is clocked by SYSCLKOUT to generate periodic interrupts for velocity calculations. The unit time out interrupt is set (QFLG[UTO]) when the unit timer (QUTMR) matches the unit period register (QUPRD).

The eQEP peripheral can be configured to latch the position counter, capture timer, and capture period values on a unit time out event so that latched values are used for velocity calculation as described in Section Section 15.2.5.

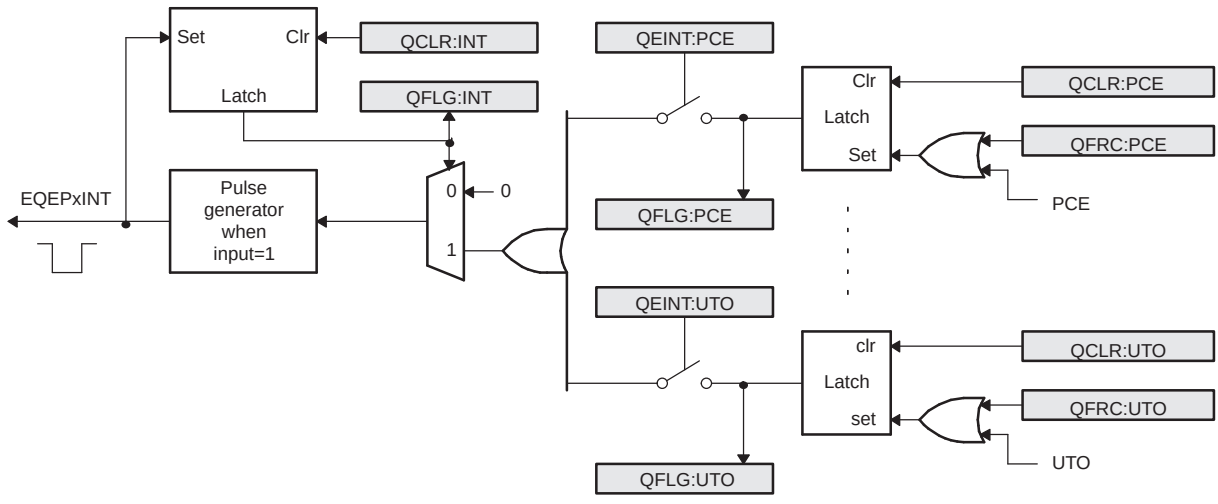
Figure 15-19. eQEP Unit Time Base



15.2.8 eQEP Interrupt Structure

Figure 15-20 shows how the interrupt mechanism works in the EQEP module.

Figure 15-20. EQEP Interrupt Generation



Eleven interrupt events (PCE, PHE, QDC, WTO, PCU, PCO, PCR, PCM, SEL, IEL, and UTO) can be generated. The interrupt control register (QEINT) is used to enable/disable individual interrupt event sources. The interrupt flag register (QFLG) indicates if any interrupt event has been latched and contains the global interrupt flag bit (INT). An interrupt pulse is generated only to the interrupt controller if any of the interrupt events is enabled, the flag bit is 1 and the INT flag bit is 0. The interrupt service routine will need to clear the global interrupt flag bit and the serviced event, via the interrupt clear register (QCLR), before any other interrupt pulses are generated. You can force an interrupt event by way of the interrupt force register (QFRC), which is useful for test purposes.

Copyright © 2013–2016, Texas Instruments Incorporated

15.3 eQEP Registers

Table 15-2 lists the registers with their memory locations, sizes, and reset values.

Table 15-2. eQEP Registers

Offset	Acronym	Register Description	Size(×16)/ #shadow	Section
0h	QPOSCNT	eQEP Position Counter Register	2/0	Section 15.3.1
4h	QPOSINIT	eQEP Position Counter Initialization Register	2/0	Section 15.3.2
8h	QPOSMAX	eQEP Maximum Position Count Register	2/0	Section 15.3.3
Ch	QPOSCMP	eQEP Position-Compare Register	2/1	Section 15.3.4
10h	QPOSILAT	eQEP Index Position Latch Register	2/0	Section 15.3.5
14h	QPOSSLAT	eQEP Strobe Position Latch Register	2/0	Section 15.3.6
18h	QPOSLAT	eQEP Position Counter Latch Register	2/0	Section 15.3.7
1Ch	QUTMR	eQEP Unit Timer Register	2/0	Section 15.3.8
20h	QUPRD	eQEP Unit Period Register	2/0	Section 15.3.9
24h	QWDTMR	eQEP Watchdog Timer Register	1/0	Section 15.3.10
26h	QWDPRD	eQEP Watchdog Period Register	1/0	Section 15.3.11
28h	QDECCTL	eQEP Decoder Control Register	1/0	Section 15.3.12
2Ah	QEPCTL	eQEP Control Register	1/0	Section 15.3.13
2Ch	QCAPCTL	eQEP Capture Control Register	1/0	Section 15.3.14
2Eh	QPOSCTL	eQEP Position-Compare Control Register	1/0	Section 15.3.15
30h	QEINT	eQEP Interrupt Enable Register	1/0	Section 15.3.16
32h	QFLG	eQEP Interrupt Flag Register	1/0	Section 15.3.17
34h	QCLR	eQEP Interrupt Clear Register	1/0	Section 15.3.18
36h	QFRC	eQEP Interrupt Force Register	1/0	Section 15.3.19
38h	QEPSTS	eQEP Status Register	1/0	Section 15.3.20
3Ah	QCTMR	eQEP Capture Timer Register	1/0	Section 15.3.21
3Ch	QCPRD	eQEP Capture Period Register	1/0	Section 15.3.22
3Eh	QCTMRLAT	eQEP Capture Timer Latch Register	1/0	Section 15.3.23
40h	QCPRDLAT	eQEP Capture Period Latch Register	1/0	Section 15.3.24
5Ch	REVID	eQEP Revision ID Register	2/0	Section 15.3.25

Copyright © 2013–2016, Texas Instruments Incorporated

15.3.1 eQEP Position Counter Register (QPOSCNT)

Figure 15-21. eQEP Position Counter Register (QPOSCNT)

31	0
QPOSCNT	
R/W-0	

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 15-3. eQEP Position Counter Register (QPOSCNT) Field Descriptions

Bits	Name	Value	Description
31-0	QPOSCNT	0-FFFF FFFFh	This 32-bit position counter register counts up/down on every eQEP pulse based on direction input. This counter acts as a position integrator whose count value is proportional to position from a give reference point.

15.3.2 eQEP Position Counter Initialization Register (QPOSINIT)

Figure 15-22. eQEP Position Counter Initialization Register (QPOSINIT)

31	0
QPOSINIT	
R/W-0	

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 15-4. eQEP Position Counter Initialization Register (QPOSINIT) Field Descriptions

Bits	Name	Value	Description
31-0	QPOSINIT	0-FFFF FFFFh	This register contains the position value that is used to initialize the position counter based on external strobe or index event. The position counter can be initialized through software.

15.3.3 eQEP Maximum Position Count Register (QPOSMAX)

Figure 15-23. eQEP Maximum Position Count Register (QPOSMAX)

31	0
QPOSMAX	
R/W-0	

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 15-5. eQEP Maximum Position Count Register (QPOSMAX) Field Descriptions

Bits	Name	Value	Description
31-0	QPOSMAX	0-FFFF FFFFh	This register contains the maximum position counter value.

Copyright © 2013–2016, Texas Instruments Incorporated

15.3.4 eQEP Position-Compare Register (QPOSCMP)

Figure 15-24. eQEP Position-Compare Register (QPOSCMP)

31	0
QPOSCMP	
R/W-0	

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 15-6. eQEP Position-Compare Register (QPOSCMP) Field Descriptions

Bits	Name	Value	Description
31-0	QPOSCMP	0-FFFF FFFFh	The position-compare value in this register is compared with the position counter (QPOSCNT) to generate sync output and/or interrupt on compare match.

15.3.5 eQEP Index Position Latch Register (QPOSILAT)

Figure 15-25. eQEP Index Position Latch Register (QPOSILAT)

31	0
QPOSILAT	
R-0	

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 15-7. eQEP Index Position Latch Register (QPOSILAT) Field Descriptions

Bits	Name	Value	Description
31-0	QPOSILAT	0-FFFF FFFFh	The position-counter value is latched into this register on an index event as defined by the QEPCTL[IEL] bits.

15.3.6 eQEP Strobe Position Latch Register (QPOSSLAT)

Figure 15-26. eQEP Strobe Position Latch Register (QPOSSLAT)

31	0
QPOSSLAT	
R-0	

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 15-8. eQEP Strobe Position Latch Register (QPOSSLAT) Field Descriptions

Bits	Name	Value	Description
31-0	QPOSSLAT	0-FFFF FFFFh	The position-counter value is latched into this register on strobe event as defined by the QEPCTL[SEL] bits.

Copyright © 2013–2016, Texas Instruments Incorporated

15.3.7 eQEP Position Counter Latch Register (QPOSLAT)

Figure 15-27. eQEP Position Counter Latch Register (QPOSLAT)

31	0
QPOSLAT	
R-0	

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 15-9. eQEP Position Counter Latch Register (QPOSLAT) Field Descriptions

Bits	Name	Value	Description
31-0	QPOSLAT	0-FFFF FFFFh	The position-counter value is latched into this register on unit time out event.

15.3.8 eQEP Unit Timer Register (QUTMR)

Figure 15-28. eQEP Unit Timer Register (QUTMR)

31	0
QUTMR	
R/W-0	

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 15-10. eQEP Unit Timer Register (QUTMR) Field Descriptions

Bits	Name	Value	Description
31-0	QUTMR	0-FFFF FFFFh	This register acts as time base for unit time event generation. When this timer value matches with unit time period value, unit time event is generated.

15.3.9 eQEP Unit Period Register (QUPRD)

Figure 15-29. eQEP Unit Period Register (QUPRD)

31	0
QUPRD	
R/W-0	

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 15-11. eQEP Unit Period Register (QUPRD) Field Descriptions

Bits	Name	Value	Description
31-0	QUPRD	0-FFFF FFFFh	This register contains the period count for unit timer to generate periodic unit time events to latch the eQEP position information at periodic interval and optionally to generate interrupt.

Copyright © 2013–2016, Texas Instruments Incorporated

15.3.10 eQEP Watchdog Timer Register (QWDTMR)

Figure 15-30. eQEP Watchdog Timer Register (QWDTMR)

15	0
QWDTMR	
R/W-0	

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 15-12. eQEP Watchdog Timer Register (QWDTMR) Field Descriptions

Bits	Name	Value	Description
15-0	QWDTMR	0-FFFF FFFFh	This register acts as time base for watch dog to detect motor stalls. When this timer value matches with watch dog period value, watch dog timeout interrupt is generated. This register is reset upon edge transition in quadrature-clock indicating the motion.

15.3.11 eQEP Watchdog Period Register (QWDPRD)

Figure 15-31. eQEP Watchdog Period Register (QWDPRD)

15	0
QWDPRD	
R/W-0	

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 15-13. eQEP Watchdog Period Register (QWDPRD) Field Description

Bits	Name	Value	Description
15-0	QWDPRD	0-FFFFh	This register contains the time-out count for the eQEP peripheral watch dog timer. When the watchdog timer value matches the watchdog period value, a watchdog timeout interrupt is generated.

Copyright © 2013–2016, Texas Instruments Incorporated

15.3.12 QEP Decoder Control Register (QDECCTL)

Figure 15-32. QEP Decoder Control Register (QDECCTL)

15	14	13	12	11	10	9	8
QSRC		SOEN	SPSEL	XCR	SWAP	IGATE	QAP
R/W-0		R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0

7	6	5	4	0
QBP	QIP	QSP	Reserved	
R/W-0	R/W-0	R/W-0	R-0	

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 15-14. eQEP Decoder Control Register (QDECCTL) Field Descriptions

Bits	Name	Value	Description
15-14	QSRC	0-3h	Position-counter source selection
		0	Quadrature count mode (QCLK = iCLK, QDIR = iDIR)
		1h	Direction-count mode (QCLK = xCLK, QDIR = xDIR)
		2h	UP count mode for frequency measurement (QCLK = xCLK, QDIR = 1)
		3h	DOWN count mode for frequency measurement (QCLK = xCLK, QDIR = 0)
13	SOEN		Sync output-enable
		0	Disable position-compare sync output
		1	Enable position-compare sync output
12	SPSEL		Sync output pin selection
		0	Index pin is used for sync output
		1	Strobe pin is used for sync output
11	XCR		External clock rate
		0	2× resolution: Count the rising/falling edge
		1	1× resolution: Count the rising edge only
10	SWAP		Swap quadrature clock inputs. This swaps the input to the quadrature decoder, reversing the counting direction.
		0	Quadrature-clock inputs are not swapped
		1	Quadrature-clock inputs are swapped
9	IGATE		Index pulse gating option
		0	Disable gating of Index pulse
		1	Gate the index pin with strobe
8	QAP		QEPA input polarity
		0	No effect
		1	Negates QEPA input
7	QBP		QEPB input polarity
		0	No effect
		1	Negates QEPB input
6	QIP		QEPI input polarity
		0	No effect
		1	Negates QEPI input
5	QSP		QEPS input polarity
		0	No effect
		1	Negates QEPS input
4-0	Reserved	0	Always write as 0

15.3.13 eQEP Control Register (QEPCTL)

Copyright © 2013–2016, Texas Instruments Incorporated

Figure 15-33. eQEP Control Register (QEPCTL)

15	14	13	12	11	10	9	8
FREE, SOFT		PCRM		SEI		IEI	
R/W-0		R/W-0		R/W-0		R/W-0	

7	6	5	4	3	2	1	0
SWI	SEL	IEL		PHEN	QCLM	UTE	WDE
R/W-0	R/W-0	R/W-0		R/W-0	R/W-0	R/W-0	R/W-0

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 15-15. eQEP Control Register (QEPCTL) Field Descriptions

Bits	Name	Value	Description
15-14	FREE, SOFT	0-3h	Emulation Control Bits
			QPOSCNT behavior:
		0	Position counter stops immediately on emulation suspend.
		1h	Position counter continues to count until the rollover.
		2h-3h	Position counter is unaffected by emulation suspend.
			QWDTMR behavior:
		0	Watchdog counter stops immediately.
		1	Watchdog counter counts until WD period match roll over.
		2h-3h	Watchdog counter is unaffected by emulation suspend.
			QUTMR behavior:
		0	Unit timer stops immediately.
		1h	Unit timer counts until period rollover.
		2h-3h	Unit timer is unaffected by emulation suspend.
			QCTMR behavior:
		0	Capture Timer stops immediately.
		1h	Capture Timer counts until next unit period event.
		2h-3h	Capture Timer is unaffected by emulation suspend.
13-12	PCRM	0-3h	Position counter reset mode
		0	Position counter reset on an index event
		1h	Position counter reset on the maximum position
		2h	Position counter reset on the first index event
		3h	Position counter reset on a unit time event
11-10	SEI	0-3h	Strobe event initialization of position counter
		0	Does nothing (action disabled)
		1h	Does nothing (action disabled)
		2h	Initializes the position counter on rising edge of the QEPS signal
		3h	**Clockwise Direction:** Initializes the position counter on the rising edge of QEPS strobe
			Counter Clockwise Direction: Initializes the position counter on the falling edge of QEPS strobe
9-8	IEI	0-3h	Index event initialization of position counter
		0	Do nothing (action disabled)
		1h	Do nothing (action disabled)
		2h	Initializes the position counter on the rising edge of the QEPI signal (QPOSCNT = QPOSINIT)
		3h	Initializes the position counter on the falling edge of QEPI signal (QPOSCNT = QPOSINIT)
7	SWI		Software initialization of position counter
		0	Do nothing (action disabled)
		1	Initialize position counter, this bit is cleared automatically

Copyright © 2013–2016, Texas Instruments Incorporated

Table 15-15. eQEP Control Register (QEPCTL) Field Descriptions (continued)

Bits	Name	Value	Description
6	SEL		Strobe event latch of position counter
		0	The position counter is latched on the rising edge of QEPS strobe (QPOSSLAT = POSCCNT). Latching on the falling edge can be done by inverting the strobe input using the QSP bit in the QDECCTL register.
		1	**Clockwise Direction:** Position counter is latched on rising edge of QEPS strobe
			Counter Clockwise Direction: Position counter is latched on falling edge of QEPS strobe
5-4	IEL	0-3h	Index event latch of position counter (software index marker)
		0	Reserved
		1h	Latches position counter on rising edge of the index signal
		2h	Latches position counter on falling edge of the index signal
		3h	Software index marker. Latches the position counter and quadrature direction flag on index event marker. The position counter is latched to the QPOSILAT register and the direction flag is latched in the QEPSTS[QDLF] bit. This mode is useful for software index marking.
3	PHEN		Quadrature position counter enable/software reset
		0	Reset the eQEP peripheral internal operating flags/read-only registers. Control/configuration registers are not disturbed by a software reset.
		1	eQEP position counter is enabled
2	QCLM		eQEP capture latch mode
		0	Latch on position counter read by CPU. Capture timer and capture period values are latched into QCTMRLAT and QCPRDLAT registers when CPU reads the QPOSCNT register.
		1	Latch on unit time out. Position counter, capture timer and capture period values are latched into QPOSLAT, QCTMRLAT and QCPRDLAT registers on unit time out.
1	UTE		eQEP unit timer enable
		0	Disable eQEP unit timer
		1	Enable unit timer
0	WDE		eQEP watchdog enable
		0	Disable the eQEP watchdog timer
		1	Enable the eQEP watchdog timer

Copyright © 2013–2016, Texas Instruments Incorporated

15.3.14 eQEP Capture Control Register (QCAPCTL)

Figure 15-34. eQEP Capture Control Register (QCAPCTL)

15	14	8
CEN	Reserved	
R/W-0	R-0	

7	6	4	3	0
Reserved	CCPS		UPPS	
R-0	R/W-0		R/W-0	

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 15-16. eQEP Capture Control Register (QCAPCTL) Field Descriptions

Bits	Name	Value	Description
15	CEN		Enable eQEP capture
		0	eQEP capture unit is disabled
		1	eQEP capture unit is enabled
14-7	Reserved	0	Always write as 0
6-4	CCPS	0-7h	eQEP capture timer clock prescaler
		0	CAPCLK = SYSCLKOUT/1
		1h	CAPCLK = SYSCLKOUT/2
		2h	CAPCLK = SYSCLKOUT/4
		3h	CAPCLK = SYSCLKOUT/8
		4h	CAPCLK = SYSCLKOUT/16
		5h	CAPCLK = SYSCLKOUT/32
		6h	CAPCLK = SYSCLKOUT/64
		7h	CAPCLK = SYSCLKOUT/128
3-0	UPPS	0-Fh	Unit position event prescaler
		0	UPEVNT = QCLK/1
		1h	UPEVNT = QCLK/2
		2h	UPEVNT = QCLK/4
		3h	UPEVNT = QCLK/8
		4h	UPEVNT = QCLK/16
		5h	UPEVNT = QCLK/32
		6h	UPEVNT = QCLK/64
		7h	UPEVNT = QCLK/128
		8h	UPEVNT = QCLK/256
		9h	UPEVNT = QCLK/512
		Ah	UPEVNT = QCLK/1024
		Bh	UPEVNT = QCLK/2048
		Ch-Fh	Reserved

Copyright © 2013–2016, Texas Instruments Incorporated

15.3.15 eQEP Position-Compare Control Register (QPOSCTL)

Figure 15-35. eQEP Position-Compare Control Register (QPOSCTL)

15	14	13	12	11	8
PCSHDW	PCLOAD	PCPOL	PCE	PCSPW	
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	

7	0
PCSPW	
R/W-0	

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 15-17. eQEP Position-Compare Control Register (QPOSCTL) Field Descriptions

Bit	Name	Value	Description
15	PCSHDW		Position-compare shadow enable
		0	Shadow disabled, load Immediate
		1	Shadow enabled
14	PCLOAD		Position-compare shadow load mode
		0	Load on QPOSCNT = 0
		1	Load when QPOSCNT = QPOSCMP
13	PCPOL		Polarity of sync output
		0	Active HIGH pulse output
		1	Active LOW pulse output
12	PCE		Position-compare enable/disable
		0	Disable position compare unit
		1	Enable position compare unit
11-0	PCSPW	0-FFFh	Select-position-compare sync output pulse width
		0	1 × 4 × SYSCLKOUT cycles
		1h	2 × 4 × SYSCLKOUT cycles
		2h-FFFh	3 × 4 × SYSCLKOUT cycles to 4096 × 4 × SYSCLKOUT cycles

Copyright © 2013–2016, Texas Instruments Incorporated

15.3.16 eQEP Interrupt Enable Register (QEINT)

Figure 15-36. eQEP Interrupt Enable Register (QEINT)

15	12	11	10	9	8
Reserved		UTO	IEL	SEL	PCM
R-0		R/W-0	R/W-0	R/W-0	R/W-0

7	6	5	4	3	2	1	0
PCR	PCO	PCU	WTO	QDC	PHE	PCE	Reserved
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 15-18. eQEP Interrupt Enable Register (QEINT) Field Descriptions

Bits	Name	Value	Description
15-12	Reserved	0	Always write as 0
11	UTO		Unit time out interrupt enable
		0	Interrupt is disabled
		1	Interrupt is enabled
10	IEL		Index event latch interrupt enable
		0	Interrupt is disabled
		1	Interrupt is enabled
9	SEL		Strobe event latch interrupt enable
		0	Interrupt is disabled
		1	Interrupt is enabled
8	PCM		Position-compare match interrupt enable
		0	Interrupt is disabled
		1	Interrupt is enabled
7	PCR		Position-compare ready interrupt enable
		0	Interrupt is disabled
		1	Interrupt is enabled
6	PCO		Position counter overflow interrupt enable
		0	Interrupt is disabled
		1	Interrupt is enabled
5	PCU		Position counter underflow interrupt enable
		0	Interrupt is disabled
		1	Interrupt is enabled
4	WTO		Watchdog time out interrupt enable
		0	Interrupt is disabled
		1	Interrupt is enabled
3	QDC		Quadrature direction change interrupt enable
		0	Interrupt is disabled
		1	Interrupt is enabled
2	PHE		Quadrature phase error interrupt enable
		0	Interrupt is disabled
		1	Interrupt is enabled
1	PCE		Position counter error interrupt enable
		0	Interrupt is disabled
		1	Interrupt is enabled
0	Reserved	0	Reserved

Copyright © 2013–2016, Texas Instruments Incorporated

15.3.17 eQEP Interrupt Flag Register (QFLG)

Figure 15-37. eQEP Interrupt Flag Register (QFLG)

15	12	11	10	9	8
Reserved		UTO	IEL	SEL	PCM
R-0		R-0	R-0	R-0	R-0

7	6	5	4	3	2	1	0
PCR	PCO	PCU	WTO	QDC	PHE	PCE	INT
R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0

LEGEND: R = Read only; -*n* = value after reset

Table 15-19. eQEP Interrupt Flag Register (QFLG) Field Descriptions

Bits	Name	Value	Description
15-12	Reserved	0	Always write as 0
11	UTO		Unit time out interrupt flag
		0	No interrupt generated
		1	Set by eQEP unit timer period match
10	IEL		Index event latch interrupt flag
		0	No interrupt generated
		1	This bit is set after latching the QPOSCNT to QPOSILAT
9	SEL		Strobe event latch interrupt flag
		0	No interrupt generated
		1	This bit is set after latching the QPOSCNT to QPOSSLAT
8	PCM		eQEP compare match event interrupt flag
		0	No interrupt generated
		1	This bit is set on position-compare match
7	PCR		Position-compare ready interrupt flag
		0	No interrupt generated
		1	This bit is set after transferring the shadow register value to the active position compare register.
6	PCO		Position counter overflow interrupt flag
		0	No interrupt generated
		1	This bit is set on position counter overflow.
5	PCU		Position counter underflow interrupt flag
		0	No interrupt generated
		1	This bit is set on position counter underflow.
4	WTO		Watchdog timeout interrupt flag
		0	No interrupt generated
		1	Set by watch dog timeout
3	QDC		Quadrature direction change interrupt flag
		0	No interrupt generated
		1	This bit is set during change of direction
2	PHE		Quadrature phase error interrupt flag
		0	No interrupt generated
		1	Set on simultaneous transition of QEPA and QEPB
1	PCE		Position counter error interrupt flag
		0	No interrupt generated
		1	Position counter error

Copyright © 2013–2016, Texas Instruments Incorporated

Table 15-19. eQEP Interrupt Flag Register (QFLG) Field Descriptions (continued)

Bits	Name	Value	Description
0	INT		Global interrupt status flag
		0	No interrupt generated
		1	Interrupt was generated

15.3.18 eQEP Interrupt Clear Register (QCLR)

Figure 15-38. eQEP Interrupt Clear Register (QCLR)

15			12	11	10	9	8
Reserved				UTO	IEL	SEL	PCM
R-0				R/W-0	R/W-0	R/W-0	R/W-0
7	6	5	4	3	2	1	0
PCR	PCO	PCU	WTO	QDC	PHE	PCE	INT
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 15-20. eQEP Interrupt Clear Register (QCLR) Field Descriptions

Bit	Field	Value	Description
15-12	Reserved	0	Always write as 0s
11	UTO		Clear unit time out interrupt flag
		0	No effect
		1	Clears the interrupt flag
10	IEL		Clear index event latch interrupt flag
		0	No effect
		1	Clears the interrupt flag
9	SEL		Clear strobe event latch interrupt flag
		0	No effect
		1	Clears the interrupt flag
8	PCM		Clear eQEP compare match event interrupt flag
		0	No effect
		1	Clears the interrupt flag
7	PCR		Clear position-compare ready interrupt flag
		0	No effect
		1	Clears the interrupt flag
6	PCO		Clear position counter overflow interrupt flag
		0	No effect
		1	Clears the interrupt flag
5	PCU		Clear position counter underflow interrupt flag
		0	No effect
		1	Clears the interrupt flag
4	WTO		Clear watchdog timeout interrupt flag
		0	No effect
		1	Clears the interrupt flag
3	QDC		Clear quadrature direction change interrupt flag
		0	No effect
		1	Clears the interrupt flag

Copyright © 2013–2016, Texas Instruments Incorporated

Table 15-20. eQEP Interrupt Clear Register (QCLR) Field Descriptions (continued)

Bit	Field	Value	Description
2	PHE		Clear quadrature phase error interrupt flag
		0	No effect
		1	Clears the interrupt flag
1	PCE		Clear position counter error interrupt flag
		0	No effect
		1	Clears the interrupt flag
0	INT		Global interrupt clear flag
		0	No effect
		1	Clears the interrupt flag and enables further interrupts to be generated if an event flags is set to 1.

Copyright © 2013–2016, Texas Instruments Incorporated

15.3.19 eQEP Interrupt Force Register (QFRC)

Figure 15-39. eQEP Interrupt Force Register (QFRC)

15			12	11	10	9	8
Reserved				UTO	IEL	SEL	PCM
R-0				R/W-0	R/W-0	R/W-0	R/W-0
7	6	5	4	3	2	1	0
PCR	PCO	PCU	WTO	QDC	PHE	PCE	Reserved
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 15-21. eQEP Interrupt Force Register (QFRC) Field Descriptions

Bit	Field	Value	Description
15-12	Reserved	0	Always write as 0s
11	UTO		Force unit time out interrupt
		0	No effect
		1	Force the interrupt
10	IEL		Force index event latch interrupt
		0	No effect
		1	Force the interrupt
9	SEL		Force strobe event latch interrupt
		0	No effect
		1	Force the interrupt
8	PCM		Force position-compare match interrupt
		0	No effect
		1	Force the interrupt
7	PCR		Force position-compare ready interrupt
		0	No effect
		1	Force the interrupt
6	PCO		Force position counter overflow interrupt
		0	No effect
		1	Force the interrupt
5	PCU		Force position counter underflow interrupt
		0	No effect
		1	Force the interrupt
4	WTO		Force watchdog time out interrupt
		0	No effect
		1	Force the interrupt
3	QDC		Force quadrature direction change interrupt
		0	No effect
		1	Force the interrupt
2	PHE		Force quadrature phase error interrupt
		0	No effect
		1	Force the interrupt
1	PCE		Force position counter error interrupt
		0	No effect
		1	Force the interrupt
0	Reserved	0	Always write as 0

Copyright © 2013–2016, Texas Instruments Incorporated

15.3.20 eQEP Status Register (QEPSTS)

Figure 15-40. eQEP Status Register (QEPSTS)

15							8
Reserved							
R-0							

7	6	5	4	3	2	1	0
UPEVNT	FIDF	QDF	QDLF	COEF	CDEF	FIMF	PCEF
R-0	R-0	R-0	R-0	R/W-1	R/W-1	R/W-1	R-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 15-22. eQEP Status Register (QEPSTS) Field Descriptions

Bit	Field	Value	Description
15-8	Reserved	0	Always write as 0
7	UPEVNT		Unit position event flag
		0	No unit position event detected
		1	Unit position event detected. Write 1 to clear.
6	FDF		Direction on the first index marker. Status of the direction is latched on the first index event marker.
		0	Counter-clockwise rotation (or reverse movement) on the first index event
		1	Clockwise rotation (or forward movement) on the first index event
5	QDF		Quadrature direction flag
		0	Counter-clockwise rotation (or reverse movement)
		1	Clockwise rotation (or forward movement)
4	QDLF		eQEP direction latch flag. Status of direction is latched on every index event marker.
		0	Counter-clockwise rotation (or reverse movement) on index event marker
		1	Clockwise rotation (or forward movement) on index event marker
3	COEF		Capture overflow error flag
		0	Sticky bit, cleared by writing 1
		1	Overflow occurred in eQEP Capture timer (QEPCTMR)
2	CDEF		Capture direction error flag
		0	Sticky bit, cleared by writing 1
		1	Direction change occurred between the capture position event.
1	FIMF		First index marker flag
		0	Sticky bit, cleared by writing 1
		1	Set by first occurrence of index pulse
0	PCEF		Position counter error flag. This bit is not sticky and it is updated for every index event.
		0	No error occurred during the last index transition.
		1	Position counter error

Copyright © 2013–2016, Texas Instruments Incorporated

15.3.21 eQEP Capture Timer Register (QCTMR)

Figure 15-41. eQEP Capture Timer Register (QCTMR)

15	0
QCTMR	
R/W	

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 15-23. eQEP Capture Time Register (QCTMR) Field Descriptions

Bits	Name	Value	Description
15-0	QCTMR	0-FFFFh	This register provides time base for edge capture unit.

15.3.22 eQEP Capture Period Register (QCPRD)

Figure 15-42. eQEP Capture Period Register (QCPRD)

15	0
QCPRD	
R/W	

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 15-24. eQEP Capture Period Register (QCPRD) Field Descriptions

Bits	Name	Value	Description
15-0	QCPRD	0-FFFFh	This register holds the period count value between the last successive eQEP position events

15.3.23 eQEP Capture Timer Latch Register (QCTMRLAT)

Figure 15-43. eQEP Capture Timer Latch Register (QCTMRLAT)

15	0
QCTMRLAT	
R	

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 15-25. eQEP Capture Timer Latch Register (QCTMRLAT) Field Descriptions

Bits	Name	Value	Description
15-0	QCTMRLAT	0-FFFFh	The eQEP capture timer value can be latched into this register on two events viz., unit timeout event, reading the eQEP position counter.

Copyright © 2013–2016, Texas Instruments Incorporated

15.3.24 eQEP Capture Period Latch Register (QCPRDLAT)

Figure 15-44. eQEP Capture Period Latch Register (QCPRDLAT)

15	0
QCPRDLAT	
R/W	

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 15-26. eQEP Capture Period Latch Register (QCPRDLAT) Field Descriptions

Bits	Name	Value	Description
15-0	QCPRDLAT	0-FFFFh	eQEP capture period value can be latched into this register on two events viz., unit timeout event, reading the eQEP position counter.

15.3.25 eQEP Revision ID Register (REVID)

Figure 15-45. eQEP Revision ID Register (REVID)

31	0
REV	
R-44D3 1103h	

LEGEND: R = Read only; -*n* = value after reset

Table 15-27. eQEP Revision ID Register (REVID) Field Descriptions

Bits	Name	Value	Description
31-0	REV	44D3 1103h	eQEP revision ID

Copyright © 2013–2016, Texas Instruments Incorporated

Enhanced Direct Memory Access (EDMA3) Controller

The enhanced direct memory access (EDMA3) controller is a high-performance, multichannel, multithreaded DMA controller that allows you to program a wide variety of transfer geometries and transfer sequences. This chapter describes the features and operations of the EDMA3 controller.

Section 16.1 provides a brief overview, features, and terminology. Section 16.2 provides the architecture details and common operations of the EDMA3 channel controller (EDMA3CC) and the EDMA3 transfer controller (EDMA3TC). Section 16.3 contains examples and common usage scenarios. Section 16.4 describes the memory-mapped registers associated with the EDMA3 controller.

Topic **Page**

16.1 Introduction **377**
16.2 Architecture **381**
16.3 Transfer Examples **423**
16.4 Registers **440**
16.5 Tips **507**
16.6 Setting Up a Transfer **509**

Copyright © 2013–2016, Texas Instruments Incorporated

16.1 Introduction

16.1.1 Overview

The enhanced direct memory access (EDMA3) controller's primary purpose is to service user-programmed data transfers between two memory-mapped slave endpoints on the device. Typical usage includes, but is not limited to:

- Servicing software driven paging transfers (for example, from external memory to internal device memory
- Servicing event driven peripherals, such as a serial port
- Performing sorting or subframe extraction of various data structures
- Offloading data transfers from the main device CPU(s) or DSP(s) (See your device-specific data manual for specific peripherals that are accessible via EDMA3. See the section on SCR connectivity in your device-specific data manual for EDMA3 connectivity.)

The EDMA3 has a different architecture from the previous EDMA2 controller on the TMS320C621x/C671x DSPs and TMS320C64x DSPs.

The EDMA3 controller consists of two principal blocks:

- EDMA3 channel controller: EDMA3CC
- EDMA3 transfer controller(s): EDMA3TC*n*

The EDMA3 channel controller serves as the user interface for the EDMA3 controller. The EDMA3CC includes parameter RAM (PaRAM), channel control registers, and interrupt control registers. The EDMA3CC serves to prioritize incoming software requests or events from peripherals, and submits transfer requests (TR) to the EDMA3 transfer controller.

The EDMA3 transfer controllers are responsible for data movement. The transfer request packets (TRP) submitted by the EDMA3CC contains the transfer context, based on which the transfer controller issues read/write commands to the source and destination addresses programmed for a given transfer.

16.1.2 Features

The EDMA3 channel controller (EDMA3CC) has the following features:

- Fully orthogonal transfer description
  - 3 transfer dimensions
  - A-synchronized transfers: 1 dimension serviced per event
  - AB-synchronized transfers: 2 dimensions serviced per event
  - Independent indexes on source and destination
  - Chaining feature allows 3-D transfer based on single event
- Flexible transfer definition
  - Increment or constant addressing modes
  - Linking mechanism allows automatic PaRAM set update. Useful for ping-pong type transfers, auto-reload transfers.
  - Chaining allows multiple transfers to execute with a single event
- Interrupt generation for:
  - Transfer completion
  - Error conditions (illegal addresses, illegal modes, exceeding queue threshold)
- Debug visibility
  - Queue watermarking
  - Error and status recording to facilitate debug
  - Missed event detection
- 128 parameter RAM (PaRAM) entries
- 4 shadow regions

Copyright © 2013–2016, Texas Instruments Incorporated

- 32 DMA channels
  - Event triggered transfers (transfers initiated by system/peripheral events)
  - Manual transfers (CPU(s) initiated DMA transfers)
  - Chained transfers (completion of transfer on one channel triggers a transfer on a “chained” channel)
- 8 QDMA channels
  - QDMA channels are triggered automatically upon writing to a parameter RAM (PaRAM) set entry
  - Supports linking and chaining features (similar to DMA channels)
  - Support for programmable QDMA channel to PaRAM mapping (any PaRAM entry can be used as a QDMA channel)
  - Optimized for use in conjunction to the IDMA controller (internal DMA in DSP subsystem)
- 2 event queues
- 16 event entries per event queue

The EDMA3 transfer controller (EDMA3TC) has the following features:

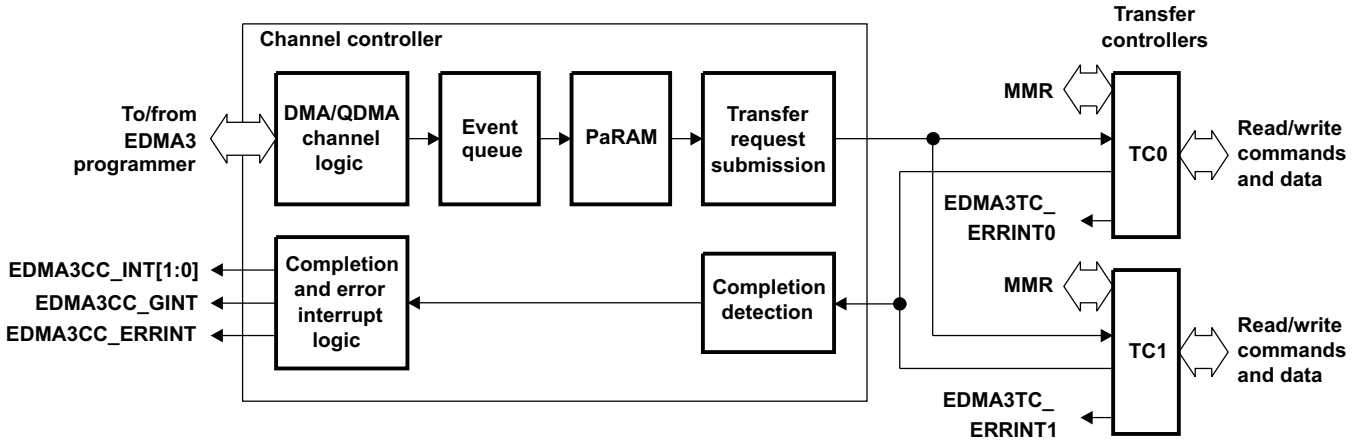
- Supports 2-dimensional transfers with independent indexes on source and destination (EDMA3CC manages the 3rd dimension)
- More then one transfer controller allows concurrent transfers
- Programmable priority level for each transfer controller relative to each other and other masters in the system.
- Support for increment or constant addressing mode transfers
- Error conditions with interrupt support
- Supports more then one in-flight transfer requests
- Debug/status visibility
- 64-bit wide read and write ports
- Little-endian mode
- Transfer controller(s):
  - FIFIOSIZE = 128 bytes
  - BUSWIDTH (Read/Write Controllers) = 8 byte
  - DSTREGDEPTH = 4
  - DBS (default) = 16 bytes. The default burst size (DBS) is programmable, and can be configured for 16-, 32-, or 64-bytes burst size. See the Chip Configuration 0 Register (CFGCHIP0) in the *System Configuration (SYSCFG) Module* chapter for details to change the default burst size value.

Copyright © 2013–2016, Texas Instruments Incorporated

16.1.3 Functional Block Diagram

Figure 16-1 shows a block diagram of the EDMA3 controller.

Figure 16-1. EDMA3 Controller Block Diagram



16.1.4 Terminology Used in This Document

The following are some terms used in this chapter.

Term	Meaning
A-synchronized transfer	A transfer type where 1 dimension is serviced per synchronization event.
AB-synchronized transfer	A transfer type where 2 dimensions are serviced per synchronization event.
Chaining	A trigger mechanism in which a transfer can be initiated at the completion of another transfer or subtransfer.
CPU(s)	The main processing engine or engines on a device. Typically a DSP or general-purpose processor. (See your device-specific data manual to learn more about the CPU on your system.)
DMA channel	A channel that can be triggered by external, manual, and chained events. All DMA channels exist in the EDMA3CC.
Dummy set or Dummy PaRAM set	A PaRAM set for which at least one of the count fields is equal to 0 and at least one of the count fields is nonzero. A null PaRAM set has all the count set fields cleared.
Dummy transfer	A dummy set results in the EDMA3CC performing a dummy transfer. This is not an error condition. A null set results in an error condition.
EDMA3 channel controller (EDMA3CC)	The user-programmable portion of the EDMA3. The EDMA3CC contains the parameter RAM (PaRAM) , event processing logic, DMA/QDMA channels, event queues, etc. The EDMA3CC services events (external, manual, chained, QDMA) and is responsible for submitting transfer requests to the transfer controllers (EDMA3TC), which perform the actual transfer.
EDMA3 programmer	Any entity on the chip that has read/write access to the EDMA3 registers and can program an EDMA3 transfer.
EDMA3 transfer controller(s) (EDMA3TC)	Transfer controllers are the transfer engine for the EDMA3. Performs the read/writes as dictated by the transfer requests submitted by the EDMA3CC.

Copyright © 2013–2016, Texas Instruments Incorporated

Term	Meaning
Enhanced direct memory access (EDMA3) controller	Consists of the EDMA3 channel controller (EDMA3CC) and EDMA3 transfer controller(s) (EDMA3TC). Is referred to as EDMA3 in this document.
Link parameter set	A PaRAM set that is used for linking.
Linking	The mechanism of reloading a PaRAM set with new transfer characteristics on completion of the current transfer.
Memory-mapped slave	All on-chip memories, off-chip memories, and slave peripherals. These typically rely on the EDMA3 (or other master peripheral) to perform transfers to and from them.
Master peripherals	All peripherals that are capable of initiating read and write transfers to the peripherals system and may not solely rely on the EDMA3 for their data transfers.
Null set or Null PaRAM set	A PaRAM set that has all count fields cleared (except for the link field). A dummy PaRAM set has at least one of the count fields nonzero.
Null transfer	A trigger event for a null PaRAM set results in the EDMA3CC performing a null transfer. This is an error condition. A dummy transfer is not an error condition.
QDMA channel	One of the 8 channels that can be triggered when writing to the trigger word (TRWORD) of a PaRAM set. All QDMA channels exist in the EDMA3CC.
Parameter RAM (PaRAM)	Programmable RAM that stores PaRAM sets used by DMA channels, QDMA channels, and linking.
Parameter RAM (PaRAM) set	A 32-byte EDMA3 channel transfer definition. Each parameter set consists of 8 words (4-bytes each), which store the context for a DMA/QDMA/link transfer. A PaRAM set includes source address, destination address, counts, indexes, options, etc.
Parameter RAM (PaRAM) set entry	One of the 4-byte components of the parameter set.
Slave end points	All on-chip memories, off-chip memories, and slave peripherals. These rely on the EDMA3 to perform transfers to and from them.
Transfer request (TR)	A command for data movement that is issued from the EDMA3CC to the EDMA3TC. A TR includes source and destination addresses, counts, indexes, options, etc.
Trigger event	Action that causes the EDMA3CC to service the PaRAM set and submit a transfer request to the EDMA3TC. Trigger events for DMA channels include manual triggered (CPU triggered), external event triggered, and chain triggered. Trigger events for QDMA channels include autotriggered and link triggered.
Trigger word	For QDMA channels, the trigger word specifies the PaRAM set entry that when written results in a QDMA trigger event. The trigger word is programmed via the QDMA channel *n* mapping register (QCHMAP*n*) and can point to any PaRAM set entry.
TR synchronization (sync) event	See Trigger event.

Copyright © 2013–2016, Texas Instruments Incorporated

16.2 Architecture

This section discusses the architecture of the EDMA3 controller.

16.2.1 Functional Overview

This section provides an overview of the EDMA3 channel controller (EDMA3CC) and EDMA3 transfer controller (EDMA3TC).

16.2.1.1 EDMA3 Channel Controller (EDMA3CC)

Figure 16-2 shows a functional block diagram of the EDMA3 channel controller (EDMA3CC).

The main blocks of the EDMA3CC are:

- DMA/QDMA Channel Logic: This block consists of logic that captures external system or peripheral events that can be used to initiate event triggered transfers, it also includes registers that allow configuring the DMA/QDMA channels (queue mapping, PaRAM entry mapping). It includes all the registers for different trigger type (manual, external events, chained and auto triggered) for enabling/disabling events, and monitor event status.
- Parameter RAM (PaRAM): Maintains parameter set entries for channel and reload parameter sets. The PaRAM needs to be written with the transfer context for the desired channels and link parameter sets.
- Event queues: These form the interface between the event detection logic and the transfer request submission logic.
- Transfer Request Submission Logic: This logic processes PaRAM sets based on a trigger event submitted to the event queue and submits a transfer request (TR) to the transfer controller associated with the event queue.
- Completion detection: The completion detect block detects completion of transfers by the EDMA3 transfer controller (EDMA3TC) and/or slave peripherals. Completion of transfers can optionally be used to chain trigger new transfers or to assert interrupts. The logic includes the interrupt processing registers for enabling/disabling interrupt (to be sent to the CPU), interrupt status/clearing registers.

Additionally there are:

- Region registers: Region registers allow DMA resources (DMA channels and interrupts) to be assigned to unique regions, which can be owned by unique EDMA programmers (a use model for hetero/multi core devices) or by unique tasks/threads (a use model for single core devices).
- Debug registers: Debug registers allow debug visibility by providing registers to read the queue status, channel controller status (what logic within the CC is active), and missed event status.

The EDMA3CC includes two channel types: DMA channels and QDMA channels.

Each channel is associated with a given event queue/transfer controller and with a given PaRAM set. The main difference between a DMA channel and QDMA channel is how the transfers are triggered by the system. See Section 16.2.4.

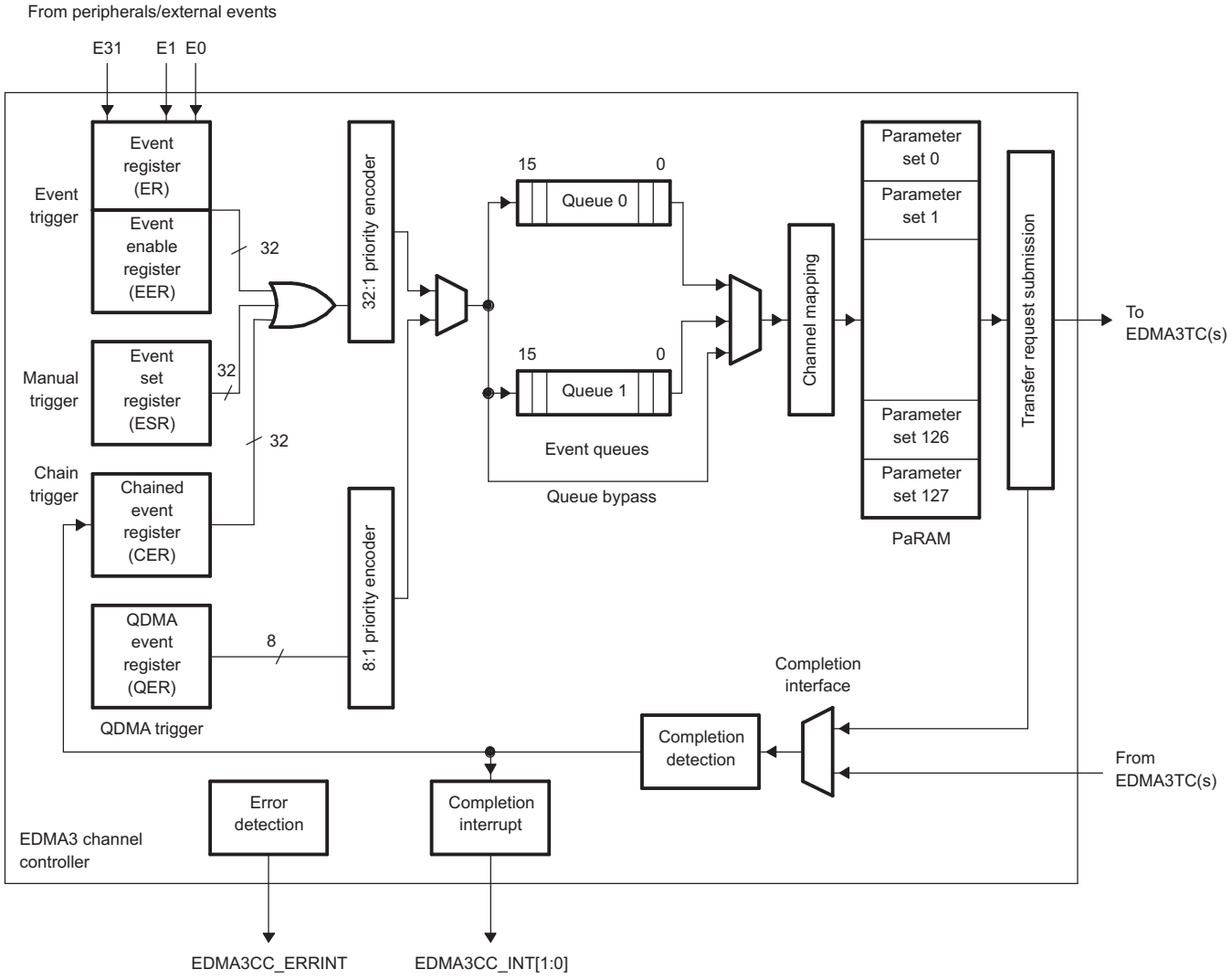
A trigger event is needed to initiate a transfer. For DMA channels, a trigger event may be due to an external event, manual write to the event set register, or chained event. QDMA channels are autotriggered when a write is performed to the user-programmed trigger word. All such trigger events are logged into appropriate registers upon recognition. See DMA channel registers (Section 16.4.2.5) and QDMA channel registers (Section 16.4.2.7).

Once a trigger event is recognized, the event type/channel is queued in the appropriate EDMA3CC event queue. The assignment of each DMA/QDMA channel to event queue is programmable. Each queue is 16 deep, so up to 16 events may be queued (on a single queue) in the EDMA3CC at an instant in time. Additional pending events mapped to a full queue are queued when event queue space becomes available. See Section 16.2.10.

If events on different channels are detected simultaneously, the events are queued based on fixed priority arbitration scheme with the DMA channels being higher priority than the QDMA channels. Among the two groups of channels, the lowest-numbered channel is the highest priority.

Copyright © 2013–2016, Texas Instruments Incorporated

Figure 16-2. EDMA3 Channel Controller (EDMA3CC) Block Diagram



Each event in the event queue is processed in the order it was queued. On reaching the head of the queue, the PaRAM associated with that channel is read to determine the transfer details. The TR submission logic evaluates the validity of the TR and is responsible for submitting a valid transfer request (TR) to the appropriate EDMA3TC (based on the event queue to EDMA3TC association, Q0 goes to TC0, and Q1 goes to TC1, etc.). For more details, see Section 16.2.3.

The EDMA3TC receives the request and is responsible for data movement as specified in the transfer request packet (TRP) and other necessary tasks like buffering, ensuring transfers are carried out in an optimal fashion wherever possible. For more details on EDMA3TC, see Section 16.2.1.2.

You may have chosen to receive an interrupt or chain to another channel on completion of the current transfer in which case the EDMA3TC signals completion to the EDMA3CC completion detection logic when the transfer is done. You can alternately choose to trigger completion when a TR leaves the EDMA3CC boundary rather than wait for all the data transfers to complete. Based on the setting of the EDMA3CC interrupt registers, the completion interrupt generation logic is responsible for generating EDMA3CC completion interrupts to the CPU. For more details, see Section 16.2.5.

Additionally, the EDMA3CC also has an error detection logic, which causes error interrupt generation on various error conditions (like missed events, exceeding event queue thresholds, etc.). For more details on error interrupts, see Section 16.2.9.4.

Copyright © 2013–2016, Texas Instruments Incorporated

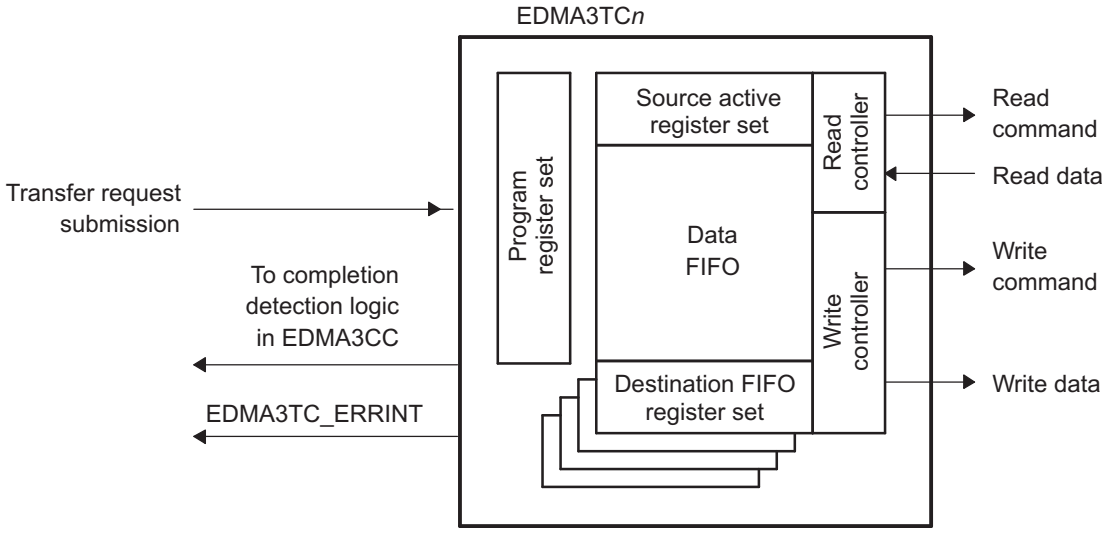
16.2.1.2 EDMA3 Transfer Controller (EDMA3TC)

Figure 16-3 shows a functional block diagram of the EDMA3 transfer controller (EDMA3TC).

The main blocks of the EDMA3TC are:

- DMA program register set: The DMA program register set stores the transfer requests received from the EDMA3 channel controller (EDMA3CC).
- DMA source active register set: The DMA source active register set stores the context for the DMA transfer request currently in progress in the read controller.
- Read controller: The read controller issues read commands to the source address.
- Destination FIFO register set: The destination (Dst) FIFO register set stores the context for the DMA transfer request(s) currently in progress or pending in the write controller.
- Write controller: The write controller issues write commands/write data to the destination address.
- Data FIFO: The data FIFO holds temporary in-flight data. The source peripheral's read data is stored in the data FIFO and subsequently written to the destination peripheral/end point by the write controller.
- Completion interface: The completion interface sends completion codes to the EDMA3CC when a transfer completes, and is used for generating interrupts and chained events (see Section 16.2.5 for details on transfer completion reporting).

Figure 16-3. EDMA3 Transfer Controller (EDMA3TC) Block Diagram



When the EDMA3TC is idle and receives its first TR, the TR is received in the DMA program register set, where it transitions to the DMA source active set and the destination FIFO register set immediately. The source active register set tracks the commands for the source side of the transfers, and the destination FIFO register set tracks commands for the destination side of the transfer. The second TR (if pending from EDMA3CC) is loaded into the DMA program set, ensuring it can start as soon as possible when the active transfer (the transfer in the source active set) is completed. As soon as the current active set is exhausted, the TR is loaded from the DMA program register set into the DMA source active register set as well as to the appropriate entry in the destination FIFO register set.

The read controller issues read commands governed by the rules of command fragmentation and optimization. These are issued only when the data FIFO has space available for the read data. The number of read commands issued depends on the TR transfer size. The TC write controller starts issuing write commands as soon as sufficient data is read in the data FIFO for the write controller to issue optimally sized write commands following the rules for command fragmentation and optimization. For details on command fragmentation and optimization, see Section 16.2.11.1.2.

Copyright © 2013–2016, Texas Instruments Incorporated

The DSTREGDEPTH parameter (fixed for a given transfer controller) determines the number of entries in the Dst FIFO register set. The number of entries determines the amount of TR pipelining possible for a given TC. The write controller can manage the write context for the number of entries in the Dst FIFO register set. This allows the read controller to go ahead and issue read commands for the subsequent TRs while the Dst FIFO register set manages the write commands and data for the previous TR. In summary, if the DSTREGDEPTH is *n*, the read controller is able to process up to *n*TRs ahead of the write controller. However, the overall TR pipelining is also subject to the amount of free space in the data FIFO.

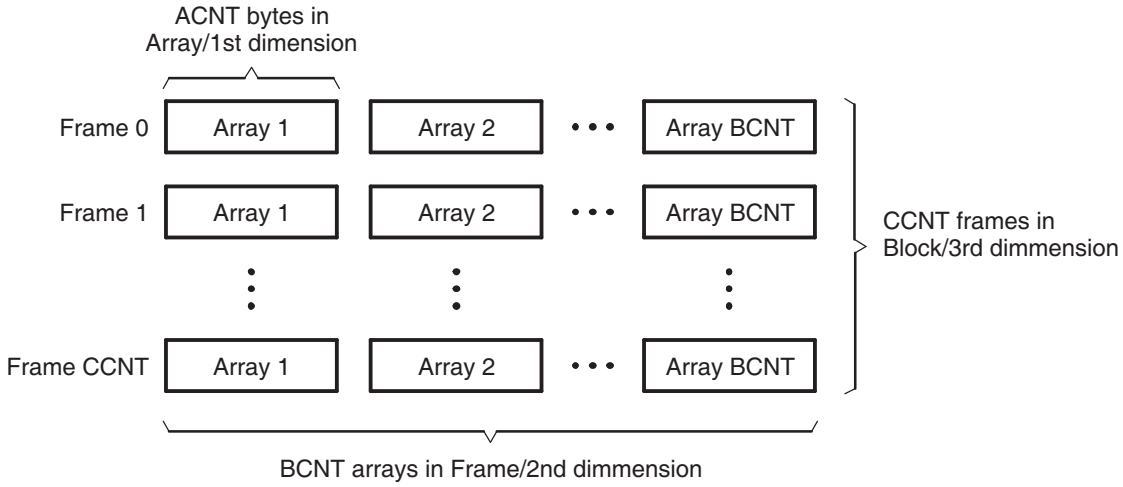
16.2.2 Types of EDMA3 Transfers

An EDMA3 transfer is always defined in terms of three dimensions. Figure 16-4 shows the three dimensions used by EDMA3 transfers. These three dimensions are defined as:

- 1st Dimension or Array (A): The 1st dimension in a transfer consists of ACNT contiguous bytes.
- 2nd Dimension or Frame (B): The 2nd dimension in a transfer consists of BCNT arrays of ACNT bytes. Each array transfer in the 2nd dimension is separated from each other by an index programmed using SRCBIDX or DSTBIDX.
- 3rd Dimension or Block (C): The 3rd dimension in a transfer consists of CCNT frames of BCNT arrays of ACNT bytes. Each transfer in the 3rd dimension is separated from the previous by an index programmed using SRCCIDX or DSTCIDX.

Note that the reference point for the index depends on the synchronization type. The amount of data transferred upon receipt of a trigger/synchronization event is controlled by the synchronization types (SYNCDIM bit in OPT). Of the three dimensions, only two synchronization types are supported: A-synchronized transfers and AB-synchronized transfers.

Figure 16-4. Definition of ACNT, BCNT, and CCNT



Copyright © 2013–2016, Texas Instruments Incorporated

16.2.2.1 A-Synchronized Transfers

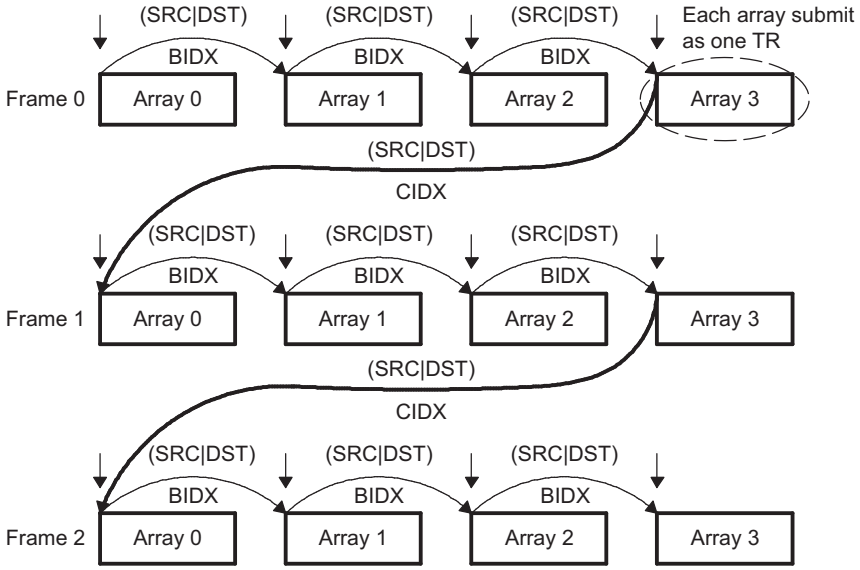
In an A-synchronized transfer, each EDMA3 sync event initiates the transfer of the 1st dimension of ACNT bytes, or one array of ACNT bytes. In other words, each event/TR packet conveys the transfer information for one array only. Thus, BCNT × CCNT events are needed to completely service a PaRAM set.

Arrays are always separated by SRCBIDX and DSTBIDX, as shown in Figure 16-5, where the start address of Array N is equal to the start address of Array N – 1 plus source (SRCBIDX) or destination (DSTBIDX).

Frames are always separated by SRCCIDX and DSTCIDX. For A-synchronized transfers, after the frame is exhausted, the address is updated by adding SRCCIDX/DSTCIDX to the beginning address of the last array in the frame. As in Figure 16-5, SRCCIDX/DSTCIDX is the difference between the start of Frame 0 Array 3 to the start of Frame 1 Array 0.

Figure 16-5 shows an A-synchronized transfer of 3 (CCNT) frames of 4 (BCNT) arrays of n (ACNT) bytes. In this example, a total of 12 sync events (BCNT × CCNT) exhaust a PaRAM set. See Section 16.2.3.6 for details on parameter set updates.

Figure 16-5. A-Synchronized Transfers (ACNT = n, BCNT = 4, CCNT = 3)



Copyright © 2013–2016, Texas Instruments Incorporated

16.2.2.2 AB-Synchronized Transfers

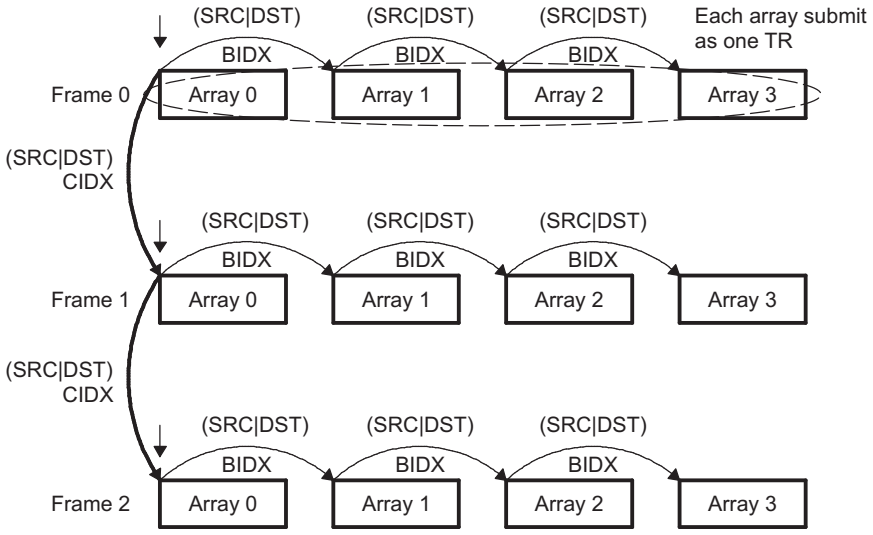
In a AB-synchronized transfer, each EDMA3 sync event initiates the transfer of 2 dimensions or one frame. In other words, each event/TR packet conveys information for one entire frame of BCNT arrays of ACNT bytes. Thus, CCNT events are needed to completely service a PaRAM set.

Arrays are always separated by SRCBIDX and DSTBIDX as shown in Figure 16-6. Frames are always separated by SRCCIDX and DSTCIDX.

Note that for AB-synchronized transfers, after a TR for the frame is submitted, the address update is to add SRCCIDX/DSTCIDX to the beginning address of the beginning array in the frame. This is different from A-synchronized transfers where the address is updated by adding SRCCIDX/DSTCIDX to the start address of the last array in the frame. See Section 16.2.3.6 for details on parameter set updates.

Figure 16-6 shows an AB-synchronized transfer of 3 (CCNT) frames of 4 (BCNT) arrays of *n* (ACNT) bytes. In this example, a total of 3 sync events (CCNT) exhaust a PaRAM set; that is, a total of 3 transfers of 4 arrays each completes the transfer.

Figure 16-6. AB-Synchronized Transfers (ACNT = n, BCNT = 4, CCNT = 3)



NOTE: ABC-synchronized transfers are not directly supported. But can be logically achieved by chaining between multiple AB-synchronized transfers.

Copyright © 2013–2016, Texas Instruments Incorporated

16.2.3 Parameter RAM (PaRAM)

The EDMA3 controller is a RAM-based architecture. The transfer context (source/destination addresses, count, indexes, etc.) for DMA or QDMA channels is programmed in a parameter RAM table within the EDMA3CC, referred to as PaRAM. The PaRAM table is segmented into multiple PaRAM sets. Each PaRAM set includes eight 4-byte PaRAM set entries (32-bytes total per PaRAM set), which includes typical DMA transfer parameters such as source address, destination address, transfer counts, indexes, options, etc. See your device-specific data manual for the addresses of the PaRAM set entries.

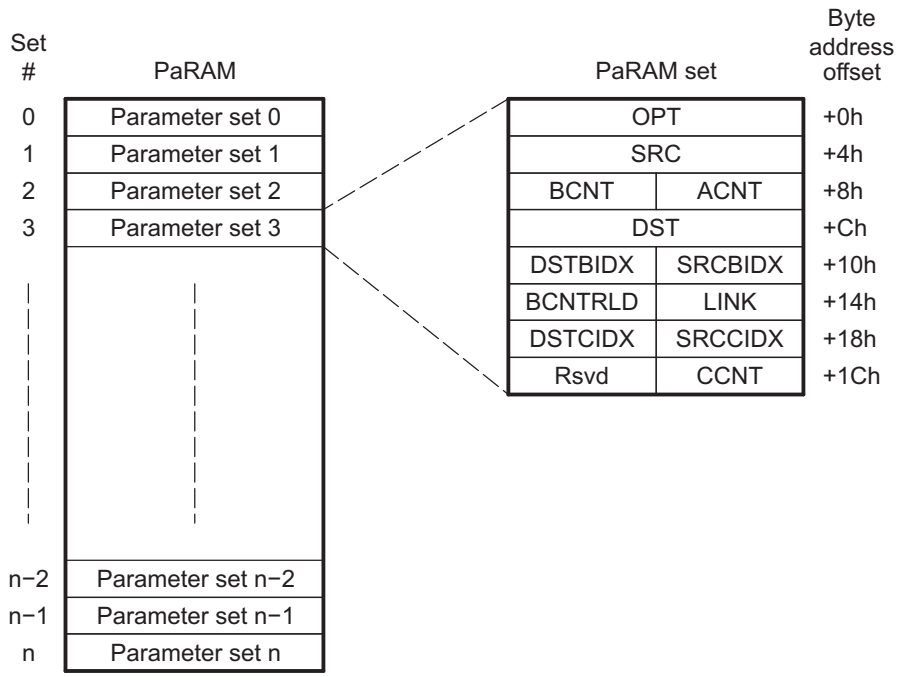
The PaRAM structure supports flexible ping-pong, circular buffering, channel chaining, and autoreloading (linking). The first *n* PaRAM sets are directly mapped to the DMA channels (where *n* is the number of DMA channels supported in the EDMA3CC for a specific device). The remaining PaRAM sets can be used for link entries or associated with QDMA channels. Additionally if the DMA channels are not used, the PaRAM sets associated with the unused DMA channels can also be used for link entries or QDMA channels.

NOTE: By default, QDMA channels are mapped to PaRAM set 0. These should be remapped before use, see Section 16.2.6.2.

16.2.3.1 PaRAM Set

Each parameter set of PaRAM is organized into eight 32-bit words or 32 bytes, as shown in Figure 16-7 and described in Table 16-1. Each PaRAM set consists of 16-bit and 32-bit parameters.

Figure 16-7. PaRAM Set



Note: *n* is the number of PaRAM sets supported in the EDMA3CC for a specific device.

Copyright © 2013–2016, Texas Instruments Incorporated

Table 16-1. EDMA3 Channel Parameter Description

Offset Address (bytes)	Acronym	Parameter	Description
0h	OPT	Channel Options	Transfer Configuration Options
4h	SRC	Channel Source Address	The byte address from which data is transferred.
8h[(1)]	ACNT	Count for 1st Dimension	Unsigned value specifying the number of contiguous bytes within an array (first dimension of the transfer). Valid values range from 1 to 65 535.
	BCNT	Count for 2nd Dimension	Unsigned value specifying the number of arrays in a frame, where an array is ACNT bytes. Valid values range from 1 to 65 535.
Ch	DST	Channel Destination Address	The byte address to which data is transferred.
10h[(1)]	SRCBIDX	Source BCNT Index	Signed value specifying the byte address offset between source arrays within a frame (2nd dimension). Valid values range from –32 768 and 32 767.
	DSTBIDX	Destination BCNT Index	Signed value specifying the byte address offset between destination arrays within a frame (2nd dimension). Valid values range from –32 768 and 32 767.
14h[(1)]	LINK	Link Address	The PaRAM address containing the PaRAM set to be linked (copied from) when the current PaRAM set is exhausted. A value of FFFFh specifies a null link.
	BCNTRLD	BCNT Reload	The count value used to reload BCNT when BCNT decrements to 0 (TR submitted for the last array in 2nd dimension). Only relevant in A-synchronized transfers.
18h[(1)]	SRCCIDX	Source CCNT Index	Signed value specifying the byte address offset between frames within a block (3rd dimension). Valid values range from –32 768 and 32 767. A-synchronized transfers: The byte address offset from the beginning of the last source array in a frame to the beginning of the first source array in the next frame. AB-synchronized transfers: The byte address offset from the beginning of the first source array in a frame to the beginning of the first source array in the next frame.
	DSTCIDX	Destination CCNT index	Signed value specifying the byte address offset between frames within a block (3rd dimension). Valid values range from –32 768 and 32 767. A-synchronized transfers: The byte address offset from the beginning of the last destination array in a frame to the beginning of the first destination array in the next frame. AB-synchronized transfers: The byte address offset from the beginning of the first destination array in a frame to the beginning of the first destination array in the next frame.
1Ch	CCNT	Count for 3rd Dimension	Unsigned value specifying the number of frames in a block, where a frame is BCNT arrays of ACNT bytes. Valid values range from 1 to 65 535.
	RSVD	Reserved	Reserved

(1) If OPT, SRC, or DST is the trigger word for a QDMA transfer then it is required to do a 32-bit access to that field. Furthermore, it is recommended to perform only 32-bit accesses on the parameter RAM for best code compatibility. For example, switching the endianness of the processor swaps addresses of the 16-bit fields, but 32-bit accesses avoid the issue entirely.

Copyright © 2013–2016, Texas Instruments Incorporated

16.2.3.2 EDMA3 Channel Parameter Set Fields

16.2.3.2.1 Channel Options Parameter (OPT)

The 32-bit channel options parameter (OPT) specifies the transfer configuration options. The channel options parameter (OPT) is described in Section 16.4.1.1.

16.2.3.2.2 Channel Source Address (SRC)

The 32-bit source address parameter specifies the starting byte address of the source. For SAM in increment mode, there are no alignment restrictions imposed by EDMA3. For SAM in constant addressing mode, you must program the source address to be aligned to a 256-bit aligned address (5 LSBs of address must be 0). The EDMA3TC will signal an error, if this rule is violated. See Section 16.2.11.2 for additional details.

16.2.3.2.3 Channel Destination Address (DST)

The 32-bit destination address parameter specifies the starting byte address of the destination. For DAM in increment mode, there are no alignment restrictions imposed by EDMA3. For DAM in constant addressing mode, you must program the destination address to be aligned to a 256-bit aligned address (5 LSBs of address must be 0). The EDMA3TC will signal an error, if this rule is violated. See Section 16.2.11.2 for additional details.

16.2.3.2.4 Count for 1st Dimension (ACNT)

ACNT represents the number of bytes within the 1st dimension of a transfer. ACNT is a 16-bit unsigned value with valid values between 0 and 65 535. Therefore, the maximum number of bytes in an array is 65 535 bytes (64K – 1 bytes). ACNT must be greater than or equal to 1 for a TR to be submitted to EDMA3TC. A transfer with ACNT equal to 0 is considered either a null or dummy transfer.

See Section 16.2.3.5 and Section 16.2.5.3 for details on dummy/null completion conditions.

16.2.3.2.5 Count for 2nd Dimension (BCNT)

BCNT is a 16-bit unsigned value that specifies the number of arrays of length ACNT. For normal operation, valid values for BCNT are between 1 and 65 535. Therefore, the maximum number of arrays in a frame is 65 535 (64K – 1 arrays). A transfer with BCNT equal to 0 is considered either a null or dummy transfer.

See Section 16.2.3.5 and Section 16.2.5.3 for details on dummy/null completion conditions.

16.2.3.2.6 Count for 3rd Dimension (CCNT)

CCNT is a 16-bit unsigned value that specifies the number of frames in a block. Valid values for CCNT are between 1 and 65 535. Therefore, the maximum number of frames in a block is 65 535 (64K – 1 frames). A transfer with CCNT equal to 0 is considered either a null or dummy transfer.

See Section 16.2.3.5 and Section 16.2.5.3 for details on dummy/null completion conditions.

16.2.3.2.7 BCNT Reload (BCNTRLD)

BCNTRLD is a 16-bit unsigned value used to reload the BCNT field once the last array in the 2nd dimension is transferred. This field is only used for A-synchronized transfers. In this case, the EDMA3CC decrements the BCNT value by 1 on each TR submission. When BCNT reaches 0, the EDMA3CC decrements CCNT and uses the BCNTRLD value to reinitialize the BCNT value.

For AB-synchronized transfers, the EDMA3CC submits the BCNT in the TR and the EDMA3TC decrements BCNT appropriately. For AB-synchronized transfers, BCNTRLD is not used.

Copyright © 2013–2016, Texas Instruments Incorporated

16.2.3.2.8 Source B Index (SRCBIDX)

SRCBIDX is a 16-bit signed value (2s complement) used for source address modification between each array in the 2nd dimension. Valid values for SRCBIDX are between –32 768 and 32 767. It provides a byte address offset from the beginning of the source array to the beginning of the next source array. It applies to both A-synchronized and AB-synchronized transfers. Some examples:

- SRCBIDX = 0000h (0): no address offset from the beginning of an array to the beginning of the next array. All arrays are fixed to the same beginning address.
- SRCBIDX = 0003h (+3): the address offset from the beginning of an array to the beginning of the next array in a frame is 3 bytes. For example, if the current array begins at address 1000h, the next array begins at 1003h.
- SRCBIDX = FFFFh (–1): the address offset from the beginning of an array to the beginning of the next array in a frame is –1 byte. For example, if the current array begins at address 5054h, the next array begins at 5053h.

16.2.3.2.9 Destination B Index (DSTBIDX)

DSTBIDX is a 16-bit signed value (2s complement) used for destination address modification between each array in the 2nd dimension. Valid values for DSTBIDX are between –32 768 and 32 767. It provides a byte address offset from the beginning of the destination array to the beginning of the next destination array within the current frame. It applies to both A-synchronized and AB-synchronized transfers. See SRCBIDX (Section 16.2.3.2.8) for examples.

16.2.3.2.10 Source C Index (SRCCIDX)

SRCCIDX is a 16-bit signed value (2s complement) used for source address modification in the 3rd dimension. Valid values for SRCCIDX are between –32 768 and 32 767. It provides a byte address offset from the beginning of the current array (pointed to by SRC address) to the beginning of the first source array in the next frame. It applies to both A-synchronized and AB-synchronized transfers. Note that when SRCCIDX is applied, the current array in an A-synchronized transfer is the last array in the frame (Figure 16-5), while the current array in an AB-synchronized transfer is the first array in the frame (Figure 16-6).

16.2.3.2.11 Destination C Index (DSTCIDX)

DSTCIDX is a 16-bit signed value (2s complement) used for destination address modification in the 3rd dimension. Valid values are between –32 768 and 32 767. It provides a byte address offset from the beginning of the current array (pointed to by DST address) to the beginning of the first destination array TR in the next frame. It applies to both A-synchronized and AB-synchronized transfers. Note that when DSTCIDX is applied, the current array in an A-synchronized transfer is the last array in the frame (Figure 16-5), while the current array in a AB-synchronized transfer is the first array in the frame (Figure 16-6).

16.2.3.2.12 Link Address (LINK)

The EDMA3CC provides a mechanism, called linking, to reload the current PaRAM set upon its natural termination (that is, after the count fields are decremented to 0) with a new PaRAM set. The 16-bit parameter LINK specifies the byte address offset in the PaRAM from which the EDMA3CC loads/reloads the next PaRAM set during linking.

You must program the link address to point to a valid aligned 32-byte PaRAM set. The 5 LSBs of the LINK field should be cleared to 0.

The EDMA3CC ignores the upper 2 bits of the LINK entry, allowing the programmer the flexibility of programming the link address as either an absolute/literal byte address or use the PaRAM-base-relative offset address. Therefore, if you make use of the literal address with a range from 4000h to 7FFFh, it will be treated as a PaRAM-base-relative value of 0000h to 3FFFh.

You should make sure to program the LINK field correctly, so that link update is requested from a PaRAM address that falls in the range of the available PaRAM addresses on the device.

Copyright © 2013–2016, Texas Instruments Incorporated

A LINK value of FFFFh is referred to as a NULL link that should cause the EDMA3CC to perform an internal write of 0 to all entries of the current PaRAM set, except for the LINK field that is set to FFFFh. Also, see Section 16.2.5 for details on terminating a transfer.

16.2.3.3 Null PaRAM Set

A null PaRAM set is defined as a PaRAM set where all count fields (ACNT, BCNT, and CCNT) are cleared to 0. If a PaRAM set associated with a channel is a NULL set, then when serviced by the EDMA3CC, the bit corresponding to the channel is set in the associated event missed register (EMR or QEMR). This bit remains set in the associated secondary event register (SER or QSER). *This implies that any future events on the same channel are ignored by the EDMA3CC and you are required to clear the bit in SER or QSER for the channel.* This is considered an error condition, since events are not expected on a channel that is configured as a null transfer. See Section 16.4.2.5.8 and Section 16.4.2.2.1 for more information on the SER and EMR registers, respectively.

16.2.3.4 Dummy PaRAM Set

A dummy PaRAM set is defined as a PaRAM set where at least one of the count fields (ACNT, BCNT, or CCNT) is cleared to 0 and at least one of the count fields is nonzero.

If a PaRAM set associated with a channel is a dummy set, then when serviced by the EDMA3CC, it will not set the bit corresponding to the channel (DMA/QDMA) in the event missed register (EMR or QEMR) and the secondary event register (SER or QSER) bit gets cleared similar to a normal transfer. Future events on that channel are serviced. A dummy transfer is a legal transfer of 0 bytes. See Section 16.4.2.5.8 and Section 16.4.2.2.1 for more information on the SER and EMR registers, respectively.

16.2.3.5 Dummy Versus Null Transfer Comparison

There are some differences in the way the EDMA3CC logic treats a dummy versus a null transfer request. A null transfer request is an error condition, but a dummy transfer is a legal transfer of 0 bytes. A null transfer causes an error bit (E*n*) in EMR to get set and the E*n* bit in SER remains set, essentially preventing any further transfers on that channel without clearing the associated error registers.

Table 16-2 summarizes the conditions and effects of null and dummy transfer requests.

Table 16-2. Dummy and Null Transfer Request

Feature	Null TR	Dummy TR
EMR/QEMR is set	Yes	No
SER/QSER remains set	Yes	No
Link update (STATIC = 0 in OPT)	Yes	Yes
QER is set	Yes	Yes
IPR and CER is set using early completion	Yes	Yes

16.2.3.6 Parameter Set Updates

When a TR is submitted for a given DMA/QDMA channel and its corresponding PaRAM set, the EDMA3CC is responsible for updating the PaRAM set in anticipation of the next trigger event. For nonfinal events, this includes address and count updates; for final events, this includes the link update.

The specific PaRAM set entries that are updated depend on the channel's synchronization type (A-synchronized or B-synchronized) and the current state of the PaRAM set. A B-update refers to the decrementing of BCNT in the case of A-synchronized transfers after the submission of successive TRs. A C-update refers to the decrementing of CCNT in the case of A-synchronized transfers after BCNT TRs for ACNT byte transfers have submitted. For AB-synchronized transfers, a C-update refers to the decrementing of CCNT after submission of every transfer request.

See Table 16-3 for details and conditions on the parameter updates. A link update occurs when the PaRAM set is exhausted, as described in Section 16.2.3.7.

Copyright © 2013–2016, Texas Instruments Incorporated

After the TR is read from the PaRAM (and is in process of being submitted to EDMA3TC), the following fields are updated if needed:

- A-synchronized: BCNT, CCNT, SRC, DST
- AB-synchronized: CCNT, SRC, DST

The following fields are not updated (except for during linking, where all fields are overwritten by the link PaRAM set):

- A-synchronized: ACNT, BCNTRLD, SRCBIDX, DSTBIDX, SRCCIDX, DSTCIDX, OPT, LINK
- AB-synchronized: ACNT, BCNT, BCNTRLD, SRCBIDX, DSTBIDX, SRCCIDX, DSTCIDX, OPT, LINK

Note that PaRAM updates only pertain to the information that is needed to properly submit the next transfer request to the EDMA3TC. Updates that occur while data is moved within a transfer request are tracked within the transfer controller, and is detailed in Section 16.2.11. For A-synchronized transfers, the EDMA3CC always submits a TRP for ACNT bytes (BCNT = 1 and CCNT = 1). For AB-synchronized transfers, the EDMA3CC always submits a TRP for ACNT bytes of BCNT arrays (CCNT = 1). The EDMA3TC is responsible for updating source and destination addresses within the array based on ACNT and FWID (in OPT). For AB-synchronized transfers, the EDMA3TC is also responsible to update source and destination addresses between arrays based on SRCBIDX and DSTBIDX.

Table 16-3 shows the details of parameter updates that occur within EDMA3CC for A-synchronized and AB-synchronized transfers.

Table 16-3. Parameter Updates in EDMA3CC (for Non-Null, Non-Dummy PaRAM Set)

	A-Synchronized Transfer			AB-Synchronized Transfer		
	B-Update	**C-Update**	**Link Update**	**B-Update**	**C-Update**	**Link Update**
Condition:	**BCNT > 1**	**BCNT == 1 && CCNT > 1**	**BCNT == 1 && CCNT == 1**	**N/A**	**CCNT > 1**	**CCNT == 1**
SRC	+= SRCBIDX	+= SRCCIDX	= Link.SRC	in EDMA3TC	+= SRCCIDX	= Link.SRC
DST	+= DSTBIDX	+= DSTCIDX	= Link.DST	in EDMA3TC	+= DSTCIDX	= Link.DST
ACNT	None	None	= Link.ACNT	None	None	= Link.ACNT
BCNT	–= 1	= BCNTRLD	= Link.BCNT	in EDMA3TC	N/A	= Link.BCNT
CCNT	None	–= 1	= Link.CCNT	in EDMA3TC	–=1	= Link.CCNT
SRCBIDX	None	None	= Link.SRCBIDX	in EDMA3TC	None	= Link.SRCBIDX
DSTBIDX	None	None	= Link.DSTBIDX	None	None	= Link.DSTBIDX
SRCCIDX	None	None	= Link.SRCBIDX	in EDMA3TC	None	= Link.SRCBIDX
DSTCIDX	None	None	= Link.DSTBIDX	None	None	= Link.DSTBIDX
LINK	None	None	= Link.LINK	None	None	= Link.LINK
BCNTRLD	None	None	= Link.BCNTRLD	None	None	= Link.BCNTRLD
OPT[1]	None	None	= LINK.OPT	None	None	= LINK.OPT

(1) In all cases, no updates occur if OPT.STATIC == 1 for the current PaRAM set.

NOTE: The EDMA3CC includes no special hardware to detect when an indexed address update calculation overflows/underflows. The address update will wrap across boundaries as programmed by the user. You should ensure that no transfer is allowed to cross internal port boundaries between peripherals. A single TR must target a single source/destination slave endpoint.

Copyright © 2013–2016, Texas Instruments Incorporated

16.2.3.7 Linking Transfers

The EDMA3CC provides a mechanism known as linking, which allows the entire PaRAM set to be reloaded from a location within the PaRAM memory map (for both DMA and QDMA channels). Linking is especially useful for maintaining ping-pong buffers, circular buffering, and repetitive/continuous transfers all with no CPU intervention. Upon completion of a transfer, the current transfer parameters are reloaded with the parameter set pointed to by the 16-bit link address field (of the current parameter set). Linking only occurs when the STATIC bit in OPT is cleared to 0.

NOTE: A transfer (DMA or QDMA) should always be linked to another useful transfer. If it is required to terminate a transfer, the transfer should be linked to a NULL set.

The link update occurs after the current PaRAM set event parameters have been exhausted. An event's parameters are exhausted when the EDMA3 channel controller has submitted all the transfers associated with the PaRAM set.

A link update occurs for null and dummy transfers depending on the state of the STATIC bit in OPT and the LINK field. In both cases (null or dummy), if the value of LINK is FFFFh then a null PaRAM set (with all 0s and LINK set to FFFFh) is written to the current PaRAM set. Similarly, if LINK is set to a value other than FFFFh then the appropriate PaRAM location pointed to by LINK is copied to the current PaRAM set.

Once the channel completion conditions are met for an event, the transfer parameters located at the link address are loaded into the current DMA or QDMA channel's associated parameter set. The EDMA3CC reads the entire PaRAM set (8 words) from the PaRAM set specified by LINK and writes all 8 words to the PaRAM set associated with the current channel. Figure 16-8 shows an example of a linked transfer.

Any PaRAM set in the PaRAM can be used as a link/reload parameter set; however, it is recommended that the PaRAM sets associated with peripheral synchronization events (see Section 16.2.6) should only be used for linking if the synchronization event isolated with the channel mapped to that PaRAM set is disabled.

If a PaRAM set location is mapped to a QDMA channel (by QCHMAP*n*), then copying the link PaRAM set onto the current QDMA channel PaRAM set is recognized as a trigger event and is latched in QER since a write to the trigger word was performed. This feature can be used to create a linked list of transfers using a single QDMA channel and multiple PaRAM sets.

Link-to-self transfers replicate the behavior of autoinitialization, which facilitates the use of circular buffering and repetitive transfers. After an EDMA3 channel exhausts its current PaRAM set, it reloads all the parameter set entries from another PaRAM set, which is initialized with values identical to the original PaRAM set. Figure 16-9 shows an example of a linked-to-self transfer. In Figure 16-9, parameter set 127 has the LINK field address pointing to the address of parameter set 127, that is, linked-to-self.

NOTE: If the STATIC bit in OPT is set for a PaRAM set, then link updates are not performed. The link updates performed internally by the EDMA3CC are atomic. This implies that when the EDMA3CC is updating a PaRAM set, accesses to PaRAM by other EDMA3 programmer's (for example, CPU configuration accesses) are not allowed. Also for QDMA, for example, if the first word of the PaRAM entry is defined as a trigger word, EDMA3CC logic assures that all 8 PaRAM words are updated before the new QDMA event can trigger the transfer for that PaRAM entry.

Copyright © 2013–2016, Texas Instruments Incorporated

16.2.3.7.1 Constant Addressing Mode Transfers/Alignment Issues

If either SAM or DAM is set to 1 (constant addressing mode), then the source or destination address must be aligned to a 256-bit aligned address, respectively, and the corresponding BIDX should be an even multiple of 32 bytes (256 bit). The EDMA3CC does not recognize errors here but the EDMA3TC asserts an error, if this is not true. See Section 16.2.11.2.

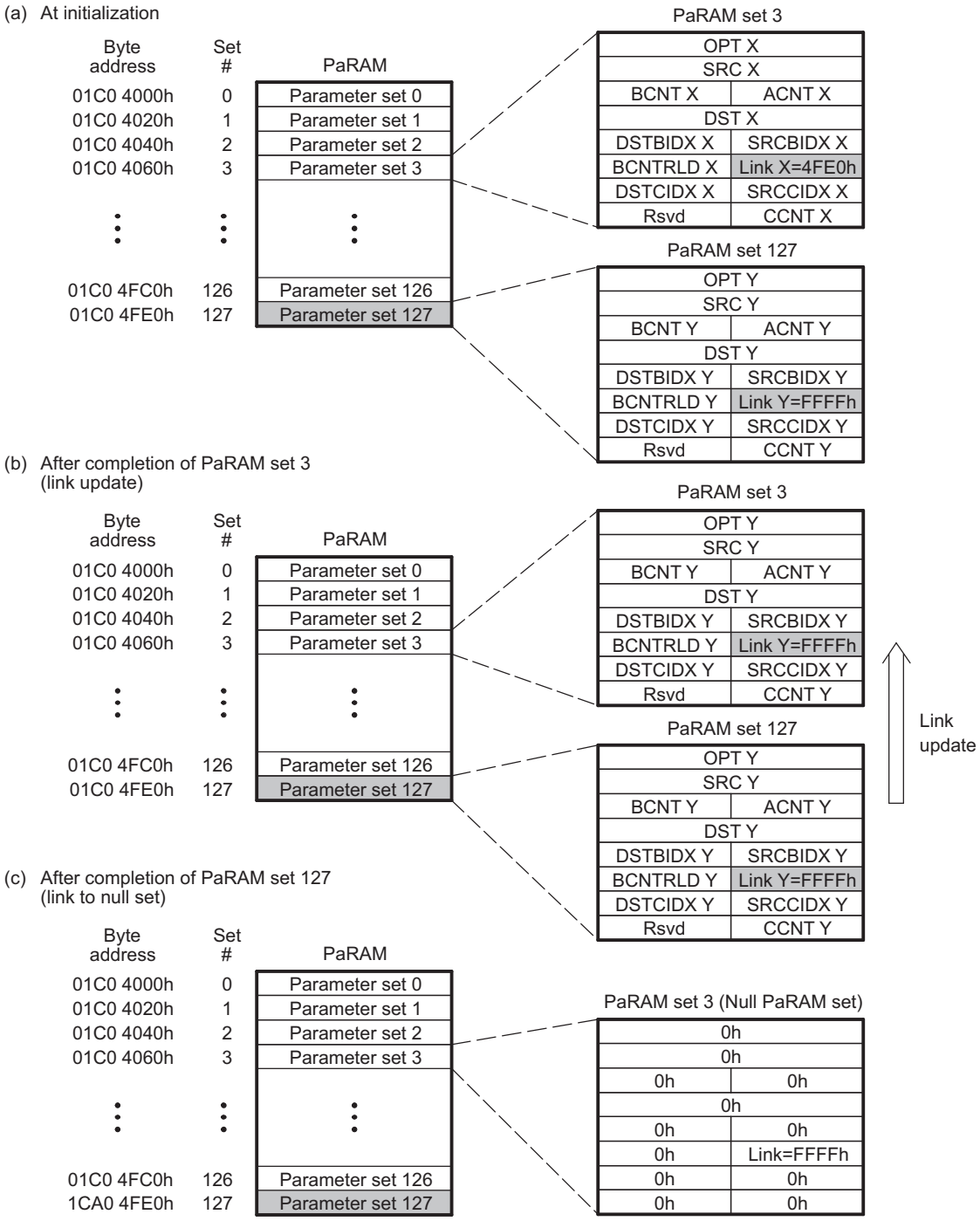
NOTE: The constant addressing (CONST) mode has limited applicability. The EDMA3 should be configured for the constant addressing mode (SAM/DAM = 1) only if the transfer source or destination (on-chip memory, off-chip memory controllers, slave peripherals) support the constant addressing mode. On the C674x/OMAP-L1x processors, no peripherals, memory, or memory controller support constant addressing mode. If the constant addressing mode is not supported, the similar logical transfer can be achieved using the increment (INCR) mode (SAM/DAM = 0) by appropriately programming the count and indices values.

16.2.3.7.2 Element Size

The EDMA3 controller does not use the concept of element-size and element-indexing. Instead, all transfers are defined in terms of all three dimensions: ACNT, BCNT, and CCNT. An element-indexed transfer is logically achieved by programming ACNT to the size of the element and BCNT to the number of elements that need to be transferred. For example, if you have 16-bit audio data and 256 audio samples that needed to be transferred to a serial port, this can be done by programming the ACNT = 2 (2 bytes) and BCNT = 256.

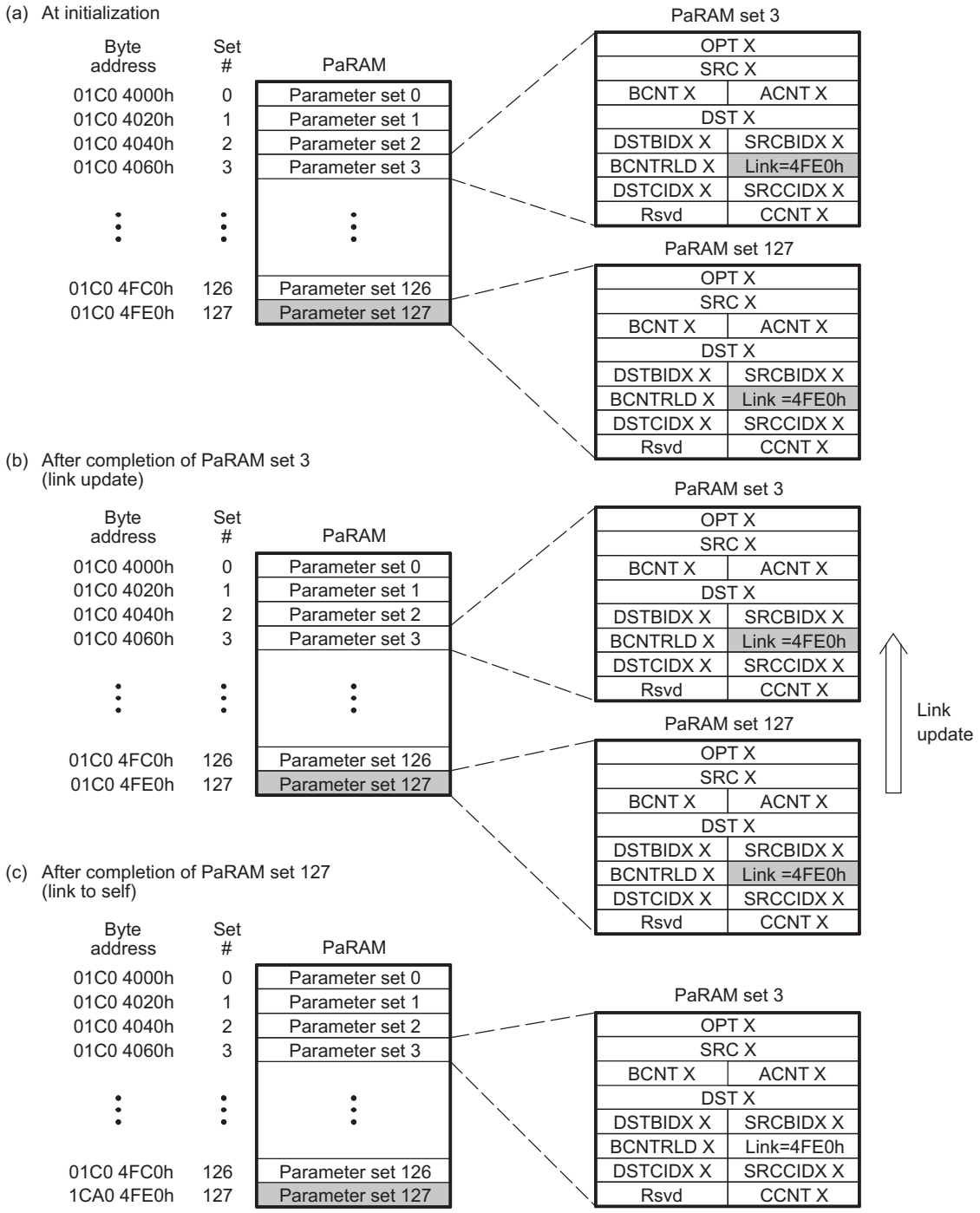
Copyright © 2013–2016, Texas Instruments Incorporated

Figure 16-8. Linked Transfer Example



Copyright © 2013–2016, Texas Instruments Incorporated

Figure 16-9. Link-to-Self Transfer Example



Copyright © 2013–2016, Texas Instruments Incorporated

16.2.4 Initiating a DMA Transfer

There are multiple ways to initiate a programmed data transfer using the EDMA3 channel controller. Transfers on DMA channels are initiated by three sources:

- **Event-triggered transfer request** (this is the more typical usage of EDMA3): Allows for a peripheral, system, or externally-generated event to trigger a transfer request.
- **Manually-triggered transfer request:** The CPU manually triggers a transfer by writing a 1 to the corresponding bit in the event set register (ESR).
- **Chain-triggered transfer request:** A transfer is triggered on the completion of another transfer or subtransfer.

Transfers on QDMA channels are initiated by two sources:

- **Autotriggered transfer request:** A transfer is triggered when the PaRAM set entry programmed trigger word is written to.
- **Link-triggered transfer requests:** When linking occurs, the transfer is triggered when the PaRAM set entry programmed trigger word is written to.

16.2.4.1 DMA Channel

16.2.4.1.1 Event-Triggered Transfer Request

When an event is asserted from a peripheral or device pins, it gets latched in the corresponding bit of the event register (ER.E*n* = 1). If the corresponding event in the event enable register (EER) is enabled (EER.E*n* = 1), then the EDMA3CC prioritizes and queues the event in the appropriate event queue. When the event reaches the head of the queue, it is evaluated for submission as a transfer request to the transfer controller.

If the PaRAM set is valid (not a NULL set), then a transfer request packet (TRP) is submitted to the EDMA3TC and the E*n* bit in ER is cleared. At this point, a new event can be safely received by the EDMA3CC.

If the PaRAM set associated with the channel is a NULL set (see Section 16.2.3.3), then no transfer request (TR) is submitted and the corresponding E*n* bit in ER is cleared and simultaneously the corresponding channel bit is set in the event miss register (EMR.E*n* = 1) to indicate that the event was discarded due to a null TR being serviced. Good programming practices should include cleaning the event missed error before retriggering the DMA channel.

When an event is received, the corresponding event bit in the event register is set (ER.E*n* = 1), regardless of the state of EER.E*n*. If the event is disabled when an external event is received (ER.E*n* = 1 and EER.E*n* = 0), the ER.E*n* bit remains set. If the event is subsequently enabled (EER.E*n* = 1), then the pending event is processed by the EDMA3CC and the TR is processed/submitted, after which the ER.E*n* bit is cleared.

If an event is being processed (prioritized or is in the event queue) and another sync event is received for the same channel prior to the original being cleared (ER.E*n* != 0), then the second event is registered as a missed event in the corresponding bit of the event missed register (EMR.E*n* = 1).

For the synchronization events associated with each of the programmable DMA channels, see your device-specific data manual to determine the event to channel mapping.

Copyright © 2013–2016, Texas Instruments Incorporated

16.2.4.1.2 Manually-Triggered Transfer Request

A DMA transfer is initiated by a write to the event set register (ESR) by the CPU (or any EDMA programmer). Writing a 1 to an event bit in the ESR results in the event being prioritized/queued in the appropriate event queue, regardless of the state of the EER.E*n* bit. When the event reaches the head of the queue, it is evaluated for submission as a transfer request to the transfer controller.

As in the event-triggered transfers, if the PaRAM set associated with the channel is valid (it is not a null set) then the TR is submitted to the associated EDMA3TC and the channel can be triggered again.

If the PaRAM set associated with the channel is a NULL set (see Section 16.2.3.3), then no transfer request (TR) is submitted and the corresponding E*n* bit in ER is cleared and simultaneously the corresponding channel bit is set in the event miss register (EMR.E*n* = 1) to indicate that the event was discarded due to a null TR being serviced. Good programming practices should include clearing the event missed error before retriggering the DMA channel.

If an event is being processed (prioritized or is in the event queue) and the same channel is manually set by a write to the corresponding channel bit of the event set register (ESR.E*n* = 1) prior to the original being cleared (ESR.E*n* = 0), then the second event is registered as a missed event in the corresponding bit of the event missed register (EMR.E*n* = 1).

16.2.4.1.3 Chain-Triggered Transfer Request

Chaining is a mechanism by which the completion of one transfer automatically sets the event for another channel. When a chained completion code is detected, the value of which is dictated by the transfer completion code (TCC[5:0] in OPT of the PaRAM set associated with the channel), it results in the corresponding bit in the chained event register (CER) to be set (CER.E[TCC] = 1).

Once a bit is set in CER, the EDMA3CC prioritizes and queues the event in the appropriate event queue. When the event reaches the head of the queue, it is evaluated for submission as a transfer request to the transfer controller.

As in the event-triggered transfers, if the PaRAM set associated with the channel is valid (it is not a null set) then the TR is submitted to the associated EDMA3TC and the channel can be triggered again.

If the PaRAM set associated with the channel is a NULL set (see Section 16.2.3.3), then no transfer request (TR) is submitted and the corresponding E*n* bit in CER is cleared and simultaneously the corresponding channel bit is set in the event miss register (EMR.E*n* = 1) to indicate that the event was discarded due to a null TR being serviced. In this case, the error condition must be cleared by you before the DMA channel can be retriggered. Good programming practices might include clearing the event missed error before retriggering the DMA channel.

If a chaining event is being processed (prioritized or queued) and another chained event is received for the same channel prior to the original being cleared (CER.E*n* != 0), then the second chained event is registered as a missed event in the corresponding channel bit of the event missed register (EMR.E*n* = 1).

NOTE: Chained event registers, event registers, and event set registers operate independently. An event (E*n*) can be triggered by any of the trigger sources (event-triggered, manually-triggered, or chain-triggered).

Copyright © 2013–2016, Texas Instruments Incorporated

16.2.4.2 QDMA Channels

16.2.4.2.1 Autotriggered and Link-Triggered Transfer Request

NOTE: If OPT, SRC, or DST is the trigger word for a QDMA transfer then it is required to do a 32-bit access to that field.

QDMA-based transfer requests are issued when a QDMA event gets latched in the QDMA event register (QER.E*n* = 1). A bit corresponding to a QDMA channel is set in the QDMA event register (QER) when the following occurs:

- A CPU (or any EDMA3 programmer) write occurs to a PaRAM address that is defined as a QDMA channel trigger word (programmed in the QDMA channel *n* mapping register (QCHMAP*n*)) for the particular QDMA channel and the QDMA channel is enabled via the QDMA event enable register (QEER.E*n* = 1).
- EDMA3CC performs a link update on a PaRAM set address that is configured as a QDMA channel (matches QCHMAP*n* settings) and the corresponding channel is enabled via the QDMA event enable register (QEER.E*n* = 1).

Once a bit is set in QER, the EDMA3CC prioritizes and queues the event in the appropriate event queue. When the event reaches the head of the queue, it is evaluated for submission as a transfer request to the transfer controller.

As in the event-triggered transfers, if the PaRAM set associated with the channel is valid (it is not a null set) then the TR is submitted to the associated EDMA3TC and the channel can be triggered again.

If a bit is already set in QER (QER.E*n* = 1) and a second QDMA event for the same QDMA channel occurs prior to the original being cleared, the second QDMA event gets captured in the QDMA event miss register (QEMR.E*n* = 1).

16.2.4.3 Comparison Between DMA and QDMA Channels

The primary difference between DMA and QDMA channels is the event/channel synchronization. QDMA events are either autotriggered or link triggered. Autotriggering allows QDMA channels to be triggered by CPU(s) with a minimum number of linear writes to PaRAM. Link triggering allows a linked list of transfers to be executed, using a single QDMA PaRAM set and multiple link PaRAM sets.

A QDMA transfer is triggered when a CPU (or other EDMA3 programmer) writes to the trigger word of the QDMA channel parameter set (autotriggered) or when the EDMA3CC performs a link update on a PaRAM set that has been mapped to a QDMA channel (link triggered). Note that for CPU triggered (manually triggered) DMA channels, in addition to writing to the PaRAM set, it is required to write to the event set register (ESR) to kick-off the transfer.

QDMA channels are typically for cases where a single event will accomplish a complete transfer since the CPU (or EDMA3 programmer) must reprogram some portion of the QDMA PaRAM set in order to retrigger the channel. In other words, QDMA transfers are programmed with BCNT = CCNT = 1 for A-synchronized transfers, and CCNT = 1 for AB-synchronized transfers.

Additionally, since linking is also supported (if STATIC = 0 in OPT) for QDMA transfers, it allows you to initiate a linked list of QDMAs, so when EDMA3CC copies over a link PaRAM set (including the write to the trigger word), the current PaRAM set mapped to the QDMA channel will automatically be recognized as a valid QDMA event and initiate another set of transfers as specified by the linked set.

Copyright © 2013–2016, Texas Instruments Incorporated

16.2.5 Completion of a DMA Transfer

A parameter set for a given channel is complete when the required number of transfer requests is submitted (based on receiving the number of synchronization events). The expected number of TRs for a non-null/non-dummy transfer is shown in Table 16-4 for both synchronization types along with state of the PaRAM set prior to the final TR being submitted. When the counts (BCNT and/or CCNT) are this value, the next TR results in a:

- Final chaining or interrupt codes to be sent by the transfer controllers (instead of intermediate).
- Link updates (linking to either null or another valid link set).

Table 16-4. Expected Number of Transfers for Non-Null Transfer

Sync Mode	Counts at time 0	Total # Transfers	Counts prior to final TR
A-synchronized	ACNT BCNT CCNT	(BCNT × CCNT) TRs of ACNT bytes each	BCNT == 1 && CCNT == 1
AB-synchronized	ACNT BCNT CCNT	CCNT TRs for ACNT × BCNT bytes each	CCNT == 1

You must program the PaRAM OPT field with a specific transfer completion code (TCC) along with the other OPT fields (TCCHEN, TCINTEN, ITCCHEN, and ITCINTEN bits) to indicate whether the completion code is to be used for generating a chained event or/and for generating an interrupt upon completion of a transfer.

The specific TCC value (6-bit binary value) programmed dictates which of the 64-bits in the chain event register (CER[TCC]) and/or interrupt pending register (IPR[TCC]) is set.

See Section 16.2.9 for details on interrupts and Section 16.2.8 for details on chaining.

You can also selectively program whether the transfer controller sends back completion codes on completion of the final transfer request (TR) of a parameter set (TCCHEN or TCINTEN), for all but the final transfer request (TR) of a parameter set (ITCCHEN or ITCINTEN), or for all TRs of a parameter set (both). See Section 16.2.8 for details on chaining (intermediate/final chaining) and Section 16.2.9 for details on intermediate/final interrupt completion.

A completion detection interface exists between the EDMA3 channel controller and transfer controller(s). This interface sends back information from the transfer controller to the channel controller to indicate that a specific transfer is completed.

All DMA/QDMA PaRAM sets must also specify a link address value. For repetitive transfers such as ping-pong buffers, the link address value should point to another predefined PaRAM set. Alternatively, a nonrepetitive transfer should set the link address value to the null link value. The null link value is defined as FFFFh. See Section 16.2.3.7 for more details.

NOTE: Any incoming events that are mapped to a null PaRAM set results in an error condition. The error condition should be cleared before the corresponding channel is used again. See Section 16.2.3.5.

There are three ways the EDMA3CC gets updated/informed about a transfer completion: normal completion, early completion, and dummy/null completion. This applies to both chained events and completion interrupt generation.

Copyright © 2013–2016, Texas Instruments Incorporated

16.2.5.1 Normal Completion

In normal completion mode (TCCMODE = 0 in OPT), the transfer or sub-transfer is considered to be complete when the EDMA3 channel controller receives the completion codes from the EDMA3 transfer controller. In this mode, the completion code to the channel controller is posted by the transfer controller after it receives a signal from the destination peripheral. Normal completion is typically used to generate an interrupt to inform the CPU that a set of data is ready for processing.

16.2.5.2 Early Completion

In early completion mode (TCCMODE = 1 in OPT), the transfer is considered to be complete when the EDMA3 channel controller submits the transfer request (TR) to the EDMA3 transfer controller. In this mode, the channel controller generates the completion code internally. Early completion is typically useful for chaining, as it allows subsequent transfers to be chained-triggered while the previous transfer is still in progress within the transfer controller, maximizing the overall throughput of the set of the transfers.

16.2.5.3 Dummy or Null Completion

This is a variation of early completion. Dummy or null completion is associated with a dummy set (Section 16.2.3.4) or null set (Section 16.2.3.3). In both cases, the EDMA3 channel controller does not submit the associated transfer request to the EDMA3 transfer controller(s). However, if the set (dummy/null) has the OPT field programmed to return completion code (intermediate/final interrupt/chaining completion), then it will set the appropriate bits in the interrupt pending register (IPR) or chained event register (CER). The internal early completion path is used by the channel controller to return the completion codes internally (that is, EDMA3CC generates the completion code).

16.2.6 Event, Channel, and PaRAM Mapping

Most of the DMA channels are tied to a specific hardware peripheral event, thus allowing transfers to be triggered by events from device peripherals or external hardware. A DMA channel typically requests a data transfer when it receives its event (apart from manually-triggered, chain-triggered, and other transfers). The amount of data transferred per synchronization event depends on the channel's configuration (ACNT, BCNT, CCNT, etc.) and the synchronization type (A-synchronized or AB-synchronized).

The association of an event to a channel is fixed. Each of the DMA channels has one specific event associated with it. For the synchronization events associated with each of the programmable DMA channels, see your device-specific data manual to determine the event to channel mapping.

If in an application, a channel does not make use of the associated synchronization event or does not have an associated synchronization event (unused), that channel can be used for manually-triggered or chained-triggered transfers, for linking/reloading, or as a QDMA channel.

Copyright © 2013–2016, Texas Instruments Incorporated

16.2.6.1 DMA Channel to PaRAM Mapping

The mapping between the DMA channel numbers and the PaRAM sets is a fixed, one-to-one mapping (see Table 16-5). In other words, channel (event) 0 is mapped to PaRAM set 0, channel (event 1) is mapped to PaRAM set 1, etc. So, for example, in order to program a transfer for event number 3, DMA channel 3 is associated with PaRAM set number 3 and you need to program this PaRAM set for configuring transfers associated with event number 3. See your device-specific data manual for the addresses of the PaRAM set entries.

Table 16-5. EDMA3 DMA Channel to PaRAM Mapping

PaRAM Set Number	Mapping
PaRAM Set 0	DMA Channel 0/Reload/QDMA
PaRAM Set 1	DMA Channel 1/Reload/QDMA
PaRAM Set 2	DMA Channel 2/Reload/QDMA
PaRAM Set 3	DMA Channel 3/Reload/QDMA
PaRAM Set 4	DMA Channel 4/Reload/QDMA
PaRAM Set 5	DMA Channel 5/Reload/QDMA
PaRAM Set 6	DMA Channel 6/Reload/QDMA
PaRAM Set 7	DMA Channel 7/Reload/QDMA
PaRAM Set 8	DMA Channel 8/Reload/QDMA
PaRAM Set 9	DMA Channel 9/Reload/QDMA
PaRAM Set 10	DMA Channel 10/Reload/QDMA
PaRAM Set 11	DMA Channel 11/Reload/QDMA
PaRAM Set 12	DMA Channel 12/Reload/QDMA
PaRAM Set 13	DMA Channel 13/Reload/QDMA
PaRAM Set 14	DMA Channel 14/Reload/QDMA
PaRAM Set 15	DMA Channel 15/Reload/QDMA
PaRAM Set 16	DMA Channel 16/Reload/QDMA
...	...
PaRAM Set 30	DMA Channel 30/Reload/QDMA
PaRAM Set 31	DMA Channel 31/Reload/QDMA
PaRAM Set 32	Reload/QDMA
PaRAM Set 33	Reload/QDMA
...	...
PaRAM Set *n* - 2	Reload/QDMA
PaRAM Set *n* - 1	Reload/QDMA
PaRAM Set *n*	Reload/QDMA

Copyright © 2013–2016, Texas Instruments Incorporated

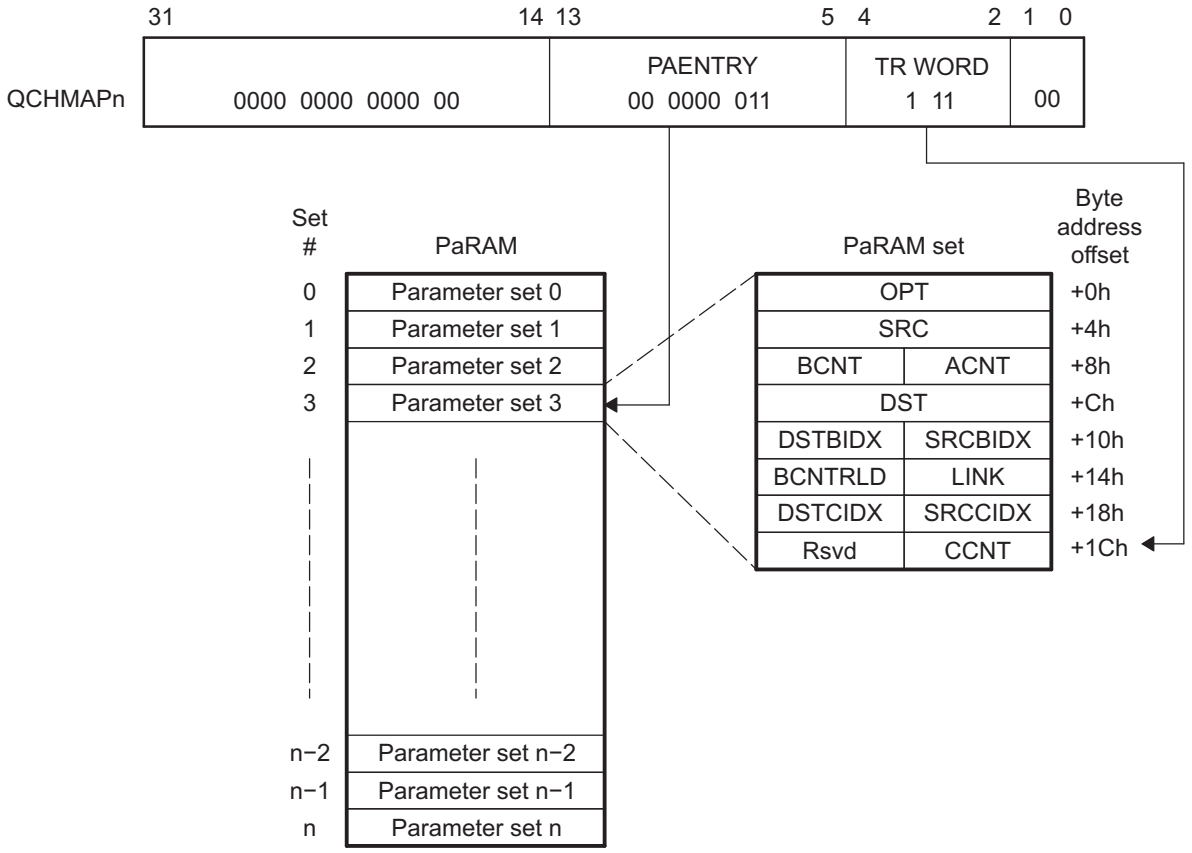
16.2.6.2 QDMA Channel to PaRAM Mapping

The mapping between the QDMA channels and the PaRAM sets is programmable .The QDMA channel *n* mapping register (QCHMAP*n*) in the EDMA3CC provides programmability for the QDMA channels to be mapped to any of the PaRAM sets in the PaRAM memory map. Figure 16-10 illustrates the use of QCHMAP.

Additionally, QCHMAP allows you to program the trigger word in the PaRAM set for the QDMA channel. A trigger word is one of the 8 words in the PaRAM set. For a QDMA transfer to occur, a valid TR synchronization event for EDMA3CC is a write to the trigger word in the PaRAM set pointed to by QCHMAP for a particular QDMA channel.

NOTE: By default, QDMA channels are mapped to PaRAM set 0. Care must be taken to appropriately remap PaRAM set 0 before it is used.

Figure 16-10. QDMA Channel to PaRAM Mapping



Note: *n* is the number of PaRAM sets supported in the EDMA3CC for a specific device.

Copyright © 2013–2016, Texas Instruments Incorporated

16.2.7 EDMA3 Channel Controller Regions

The EDMA3 channel controller (EDMA3CC) divides its address space into multiple regions. Individual channel resources can be exclusively assigned to a specific region, where each region is typically assigned to a specific EDMA programmer. This allows partitioning of EDMA channel (DMA/QDMA) resources in hetero- or multi-core devices, and devices where certain additional masters (for example, coprocessors) can also program/initiate EDMA3 transfers. The application software running on these cores/coprocessors can operate in these exclusive shadow region memory-maps, minimizing possibilities of resource conflicts.

16.2.7.1 Region Overview

The EDMA3CC memory-mapped registers are divided in three main categories:

1. Global registers
2. Global region channel registers
3. Shadow region channel registers

The global registers are located at a single/fixed location in the EDMA3CC memory map. These registers control EDMA3 resource mapping and provide debug visibility and error tracking information. See your device-specific data manual for the EDMA3CC memory map.

The channel registers (including DMA, QDMA, and interrupt registers) are accessible via the global channel region address range, or in the shadow *n* channel region address range(s). For example, the event enable register (EER) is visible in the global region register space at offset 1020h, or region addresses at offset 2020h for region 0 and at offset 2220h for region 1.

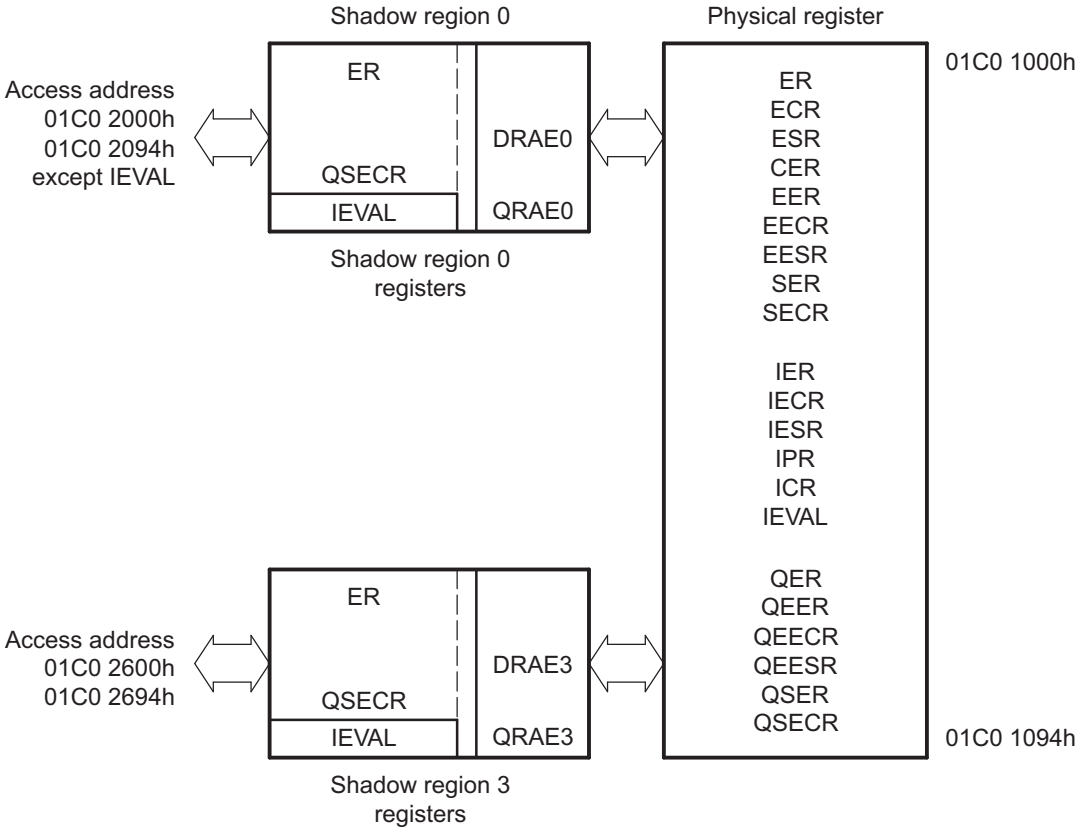
The underlying control register bits that are accessible via the shadow region address space (except for IEVAL*n*) are controlled by the DMA region access enable registers (DRAE*m*) and QDMA region access enable registers (QRAE*m*). Table 16-6 lists the registers in the shadow region memory-map. (See EDMA3CC memory-map figure for the complete global and shadow region memory-maps.) Figure 16-11 illustrates the conceptual view of the regions (where *n* is the number of shadow regions supported in the EDMA3CC for a specific device).

Table 16-6. Shadow Region Registers

DRAE*m*	**QRAE*m***
ER	QER
ECR	QEER
ESR	QEECR
CER	QEESR
EER	
EECR	
EESR	
SER	
SECR	
IER	
IECR	
IESR	
IPR	
ICR	
Register not affected by DRAE	
IEVAL	

Copyright © 2013–2016, Texas Instruments Incorporated

Figure 16-11. Shadow Region Registers



16.2.7.2 Channel Controller Shadow Regions

For each EDMA3 shadow region (and associated memory-maps) there is a set of registers associated with the shadow region that allows association of the DMA/QDMA channels and interrupt completion codes to the region. These registers are user-programmed per region to assign ownership of the DMA/QDMA channels and TCC values to a region.

- DRAE*m*: One register exists for each of the shadow regions. The number of bits in each register matches the number of DMA channels. These registers need to be programmed to assign ownership of DMA channels to the respective region. Accesses to DMA event registers and interrupt registers via the shadow region address map are filtered through DRAE. A value of 1 in the corresponding DRAE bit implies that the corresponding DMA/interrupt channel is accessible; a value of 0 in the corresponding DRAE bit forces writes to be discarded and returns a value of 0 for reads.
- QRAE*m*: One register exists for every region. The number of bits in each register matches the number of QDMA channels. These registers must be programmed to assign ownership of QDMA channels to the respective region. To enable a channel in a shadow region using shadow region 0 QEER, the respective bit in QRAE must be set or writing into QEESR will not have the desired effect.

It is typical for an application to have a unique assignment of QDMA/DMA channels (and, therefore, a given bit position) to a given region.

The use of shadow regions allows for restricted access to EDMA3 resources (DMA channels, QDMA channels, TCC, interrupts) by tasks/cores/EDMA3 programmers in a system by setting or clearing bits in the DRAE/QRAE registers. If exclusive access to any given channel/TCC code is required for a region, then only that region's DRAE/QRAE should have the associated bit set.

Copyright © 2013–2016, Texas Instruments Incorporated

Additionally, with each shadow region, there is an associated shadow region completion interrupt (EDMA3CC_INT*n* where *n* denotes the shadow region number). For multi-core/hetero-core devices, the various shadow region interrupts might be tied to the interrupt controllers for different cores. For single core devices, all shadow region interrupts would be routed to the device interrupt controller. See your device-specific data manual for the shadow region interrupt hookup to the device interrupt controller(s). The DRAE associated with each shadow region acts as a secondary interrupt enable (along with the interrupt enable register) for the respective shadow region interrupts. See Section 16.2.9 for more information on interrupts.

Example 16-1. Resource Pool Division Across Two Regions

This example illustrates a resource pool division across two regions, assuming region 0 must be allocated 16 DMA channels (0-15) and 1 QDMA channel (0), and 16 TCC codes (0-15). Region 1 needs to be allocated 16 DMA channels (16-31) and 7 QDMA channels (1-7), and 16 TCC codes (16-31). DRAE should be equal to the OR of the bits that are required for the DMA channels and the TCC codes:

```
Region 0: DRAE = 0x0000FFFF QRAE = 0x00000001 Region 1: DRAE = 0xFFFF0000 QRAE = 0x000000FE
```

16.2.8 Chaining EDMA3 Channels

The channel chaining capability for the EDMA3 allows the completion of an EDMA3 channel transfer to trigger another EDMA3 channel transfer. The purpose is to allow you the ability to chain several events through one event occurrence.

Chaining is different from linking (Section 16.2.3.7). The EDMA3 link feature reloads the current channel parameter set with the linked parameter set. The EDMA3 chaining feature does not modify or update any channel parameter set; it provides a synchronization event to the chained channel (see Section 16.2.4.1.3 for chain-triggered transfer requests).

Chaining is achieved at either final transfer completion or intermediate transfer completion, or both, of the current channel. Consider a channel *m* (DMA/QDMA) required to chain to channel *n*. Channel number *n* (0-31) needs to be programmed into the TCC field of channel *m* channel options parameter (OPT) set.

- If final transfer completion chaining (TCCHEN = 1 and ITCCHEN = 0 in channel *m* OPT) is enabled, the chain-triggered event occurs after the *last* transfer request of channel *m* is submitted (early completion) or completed (normal completion).
- If intermediate transfer completion chaining (TCCHEN = 0 and ITCCHEN = 0 in channel *m* OPT) is enabled, the chain-triggered event occurs after every *intermediate* transfer request of channel *m* is submitted (early completion) or completed (normal completion).
- If both final and intermediate transfer completion chaining (TCCHEN = 1 and ITCCHEN = 1 in channel *m* OPT) are enabled, the chain-trigger event occurs after *every* transfer request of channel *m* is submitted (early completion) or completed (normal completion).

Table 16-7 shows the number of chain event triggers occurring in different synchronized scenarios. Consider channel 31 programmed with ACNT = 3, BCNT = 4, CCNT = 5, and TCC = 30.

Table 16-7. Chain Event Triggers

	(Number of chained event triggers on channel 30)	
Options	**A-Synchronized**	**AB-Synchronized**
TCCHEN = 1, ITCCHEN = 0	1 (Last TR)	1 (Last TR)
TCCHEN = 0, ITCCHEN = 1	19 (All but the last TR)	4 (All but the last TR)
TCCHEN = 1, ITCCHEN = 1	20 (All TRs)	5 (All TRs)

16.2.9 EDMA3 Interrupts

The EDMA3 interrupts are divided into 2 categories:

- Transfer completion interrupts

Copyright © 2013–2016, Texas Instruments Incorporated

- Error interrupts

For information on the transfer completion interrupts and the error interrupts, see your device-specific data manual.

16.2.9.1 Transfer Completion Interrupts

The EDMA3CC is responsible for generating transfer completion interrupts to the CPU. The EDMA3 generates a single completion interrupt per shadow region on behalf of all DMA/QDMA channels. Various control registers and bit fields facilitate EDMA3 interrupt generation.

The transfer completion code (TCC) value is directly mapped to the bits of the interrupt pending register (IPR), as shown in Table 16-8. For example, if TCC = 00 0000b, IPR[0] is set after transfer completion, and results in an interrupt generation to the CPU if in the EDMA3CC and device interrupt controller are configured to allow a CPU interrupt. See Section 16.2.9.1.1 for details on enabling EDMA3 transfer completion interrupts.

When a completion code is returned (as a result of early or normal completion), the corresponding bit in IPR is set. For the completion code to be returned, the PaRAM set associated with the transfer must enable the transfer completion interrupt (final/intermediate) in the channel options parameter (OPT).

The transfer completion code (TCC) can be programmed to any value for a DMA/QDMA channel. There does not need to be a direct relation between the channel number and the transfer completion code value. This allows multiple channels having the same transfer completion code value to cause a CPU to execute the same interrupt service routine (ISR) for different channels.

NOTE: The TCC field in the channel options parameter (OPT) is a 6-bit field and can be programmed for any value between 0-64. For devices with 32 DMA channels, the TCC should have a value between 0 to 31 so that it sets the appropriate bits (0 to 31) in IPR (and can interrupt the CPU(s) on enabling the IER register bits (0-31)).

Table 16-8. Transfer Complete Code (TCC) to EDMA3CC Interrupt Mapping

TCC Bits in OPT (TCINTEN/ITCINTEN = 1)	IPR Bit Set
00 0000b	IPR0
00 0001b	IPR1
00 0010b	IPR2
00 0011b	IPR3
00 0100b	IPR4
…	…
…	…
01 1110b	IPR30
01 1111b	IPR31

Copyright © 2013–2016, Texas Instruments Incorporated

You can enable interrupt generation at either final transfer completion or intermediate transfer completion, or both. Consider channel *m* as an example.

- If the final transfer interrupt (TCINTEN = 1 and ITCINTEN = 0 in OPT) is enabled, the interrupt occurs after the *last* transfer request of channel *m* is either submitted or completed (depending on early or normal completion).
- If the intermediate transfer interrupt (TCINTEN = 0 and ITCINTEN = 1 in OPT) is enabled, the interrupt occurs after every *intermediate* transfer request of channel *m* is either submitted or completed (depending on early or normal completion).
- If both final and intermediate transfer completion interrupts (TCINTEN = 1 and ITCINTEN = 1 in OPT) are enabled, the interrupt occurs after *every* transfer request of channel *m* is submitted or completed (depending on early or normal completion).

Table 16-9 shows the number of interrupts occurring in different synchronized scenarios. Consider channel 31 programmed with ACNT = 3, BCNT = 4, CCNT = 5, and TCC = 30.

Table 16-9. Number of Interrupts

Options	A-Synchronized	AB-Synchronized
TCINTEN = 1, ITCINTEN = 0	1 (Last TR)	1 (Last TR)
TCINTEN = 0, ITCINTEN = 1	19 (All but the last TR)	4 (All but the last TR)
TCINTEN = 1, ITCINTEN = 1	20 (All TRs)	5 (All TRs)

16.2.9.1.1 Enabling Transfer Completion Interrupts

For the EDMA3 channel controller to assert a transfer completion to the external world, the interrupts have to be enabled in the EDMA3CC. This is in addition to setting up the TCINTEN and ITCINTEN bits in OPT of the associated PaRAM set.

The EDMA3 channel controller has interrupt enable registers (IER) and each bit location in IER serves as a primary enable for the corresponding interrupt pending register (IPR).

All the interrupt registers (IER, IESR, IECR, and IPR) are either manipulated from the global DMA channel region or by way of the DMA channel shadow regions. The shadow regions provide a view to the same set of physical registers that are in the global region.

The EDMA3 channel controller has a hierarchical completion interrupt scheme that makes use of a single set of interrupt pending register (IPR) and single set of interrupt enable registers (IER). A second level of interrupt masking is provided by the programmable DMA region access enable registers (DRAE). See Figure 16-12.

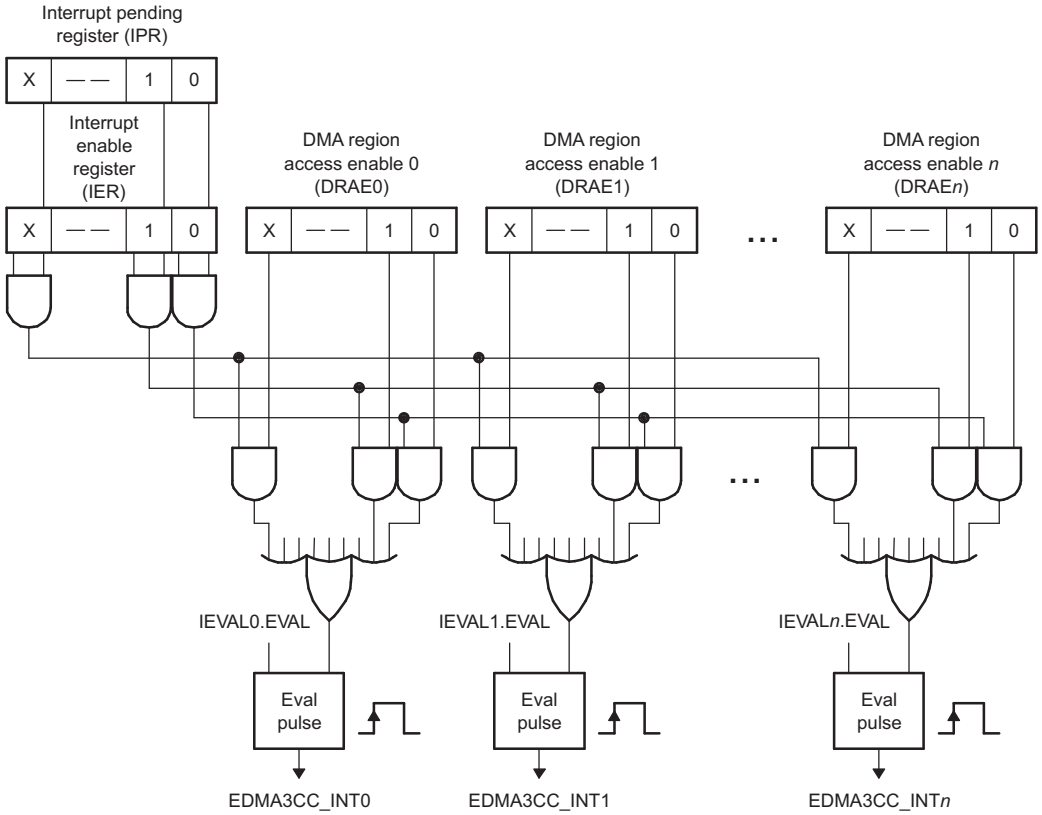
For the EDMA3CC to generate the transfer completion interrupts that are associated with each shadow region, the following conditions must be true:

- EDMA3CC_INT0: (IPR.E0 & IER.E0 & DRAE0.E0) | (IPR.E1 & IER.E1 & DRAE0.E1) | …| (IPR.E*n* & IER.E*n* & DRAE0.E*n*)
- EDMA3CC_INT1: (IPR.E0 & IER.E0 & DRAE1.E0) | (IPR.E1 & IER.E1 & DRAE1.E1) | …| (IPR.E*n* & IER.E*n* & DRAE1.E*n*)

  where *n* is the number of shadow regions supported in the EDMA3CC for a specific device.

Copyright © 2013–2016, Texas Instruments Incorporated

Figure 16-12. Interrupt Diagram



Note: *n* is the number of shadow regions supported in the EDMA3CC for a specific device.

NOTE: The DRAE for all regions is expected to be set up at system initialization and to remain static for an extended period of time. The interrupt enable registers should be used for dynamic enable/disable of individual interrupts.

Because there is no relation between the TCC value and the DMA/QDMA channel, it is possible, for example, for DMA channel 0 to have the OPT.TCC = 31 in its associated PaRAM set. This would mean that if a transfer completion interrupt is enabled (OPT.TCINTEN or OPT.ITCINTEN is set), then based on the TCC value, IPR.E31 is set up on completion. For proper channel operations and interrupt generation using the shadow region map, you must program the DRAE that is associated with the shadow region to have read/write access to both bit 0 (corresponding to channel 0) and bit 31 (corresponding to IPR.E31 bit that is set upon completion).

16.2.9.1.2 Clearing Transfer Completion Interrupts

Transfer completion interrupts that are latched to the interrupt pending register (IPR) is cleared by writing a 1 to the corresponding bit in the interrupt pending clear register (ICR). For example, a write of 1 to ICR.E0 clears a pending interrupt in IPR.E0.

If an incoming transfer completion code (TCC) gets latched to a bit in IPR, then additional bits that get set due to a subsequent transfer completion will not result in asserting the EDMA3CC completion interrupt. In order for the completion interrupt to be pulsed, the required transition is from a state where no enabled interrupts are set to a state where at least one enabled interrupt is set.

Copyright © 2013–2016, Texas Instruments Incorporated

16.2.9.2 EDMA3 Interrupt Servicing

On completion of a transfer (early or normal completion), the EDMA3 channel controller sets the appropriate bit in the interrupt pending register (IPR) as specified by the transfer completion codes. If the completion interrupts are appropriately enabled, then the CPU enters the interrupt service routine (ISR) when the completion interrupt is asserted. Since there is a single completion interrupt for all DMA/QDMA channels.

After servicing the interrupt, the ISR should clear the corresponding bit in IPR; therefore, enabling recognition of future interrupts. Only when all IPR bits are cleared, the EDMA3CC will assert additional completion interrupts.

It is possible that when one interrupt is serviced; many other transfer completions result in additional bits being set in IPR, thereby resulting in additional interrupts. It is likely that each of these bits in IPR would need different types of service; therefore, the ISR must check all pending interrupts and continue until all the posted interrupts are appropriately serviced.

Following are examples (pseudo code) for a CPU interrupt service routine for an EDMA3CC completion interrupt.

The ISR routine in Example 16-2 is more exhaustive and incurs a higher latency.

Example 16-2. Interrupt Servicing

The pseudo code:

1. Read the interrupt pending register (IPR).
2. Perform the operations needed.
3. Write to the interrupt pending clear register (ICR) to clear the corresponding IPR bit.
4. Read IPR again:

   (a) If IPR is not equal to 0, repeat from step 2 (implies occurrence of new event between step 2 to step 4).

   (b) If IPR is equal to 0, this should assure you that all enabled interrupts are inactive.

NOTE: It is possible that during step 4, an event occurs while the IPR bits are read to be 0 and the application is still in the interrupt service routine. If this happens, a new interrupt is recorded in the device interrupt controller and a new interrupt is generated as soon as the application exits the interrupt service routine.

Copyright © 2013–2016, Texas Instruments Incorporated

Example 16-3 is less rigorous, with less burden on the software in polling for set interrupt bits, but can occasionally cause a race condition, as mentioned above.

Example 16-3. Interrupt Servicing

If it is desired to leave any enabled and pending (possibly lower priority) interrupts, it is required to force the interrupt logic to reassert the interrupt pulse by setting the EVAL bit in the interrupt evaluation register (IEVAL).

The pseudo code:

1. Enter ISR.
2. Read IPR.
3. For the condition set in IPR that you desire to service:
   (a) Service interrupt as required by application.
   (b) Clear bit for serviced conditions (others may still be set, and other transfers may have resulted in returning the TCC to EDMA3CC after step 2).
4. Read IPR prior to exiting ISR:
   (a) If IPR is equal to 0, then exit ISR.
   (b) If IPR is not equal to 0, then set IEVAL so that upon exit of ISR, a new interrupt is triggered if any enabled interrupts are still pending.

The EVAL bit must not be set when IPR is read to be 0, to avoid generation of extra interrupt pulses.

NOTE: Since the DMA region access registers (DRAE) are required to enable the transfer completion region interrupts, it is assumed that there will be a unique and nonoverlapping (in most cases) assignment of the channels and interrupts among the different shadow regions. This allows the interrupt registers (IER, IESR, IECR, IPR, and ICR) in the different shadow regions to functionally operate in an independent manner and nonoverlapping. The above examples for the interrupt service routine is based on this assumption.

16.2.9.3 Interrupt Evaluation Operations

The EDMA3CC has interrupt evaluate registers (IEVAL) in each shadow region. These registers are the only registers in the DMA channel shadow region memory map that are not affected by the settings for the DMA region access enable registers (DRAE). A write of 1 to the EVAL bit in these registers associated with a particular shadow region results in pulsing the associated region interrupt, if any enabled interrupt (via IER) is still pending (IPR). This register can be used in order to assure that the interrupts are not missed by the CPU (or the EDMA3 master associated with the shadow region) if the software architecture chooses not to use all interrupts. See Example 16-3 for the use of IEVAL in the EDMA3 interrupt service routine (ISR).

Similarly an error evaluate register (EEVAL) exists in the global region. A write of 1 to the EVAL bit in EEVAL causes the pulsing of the error interrupt if any pending errors are in EMR, QEMR, or CCERR. See Section 16.2.9.4 for additional details on error interrupts.

NOTE: While using IEVAL for shadow region completion interrupts, you should make sure that the IEVAL operated upon is from that particular shadow region memory map.

Copyright © 2013–2016, Texas Instruments Incorporated

16.2.9.4 Error Interrupts

The EDMA3CC error registers provide the capability to differentiate error conditions (event missed, threshold exceed, etc.). Additionally, if the error bits are set in these registers, it results in asserting the EDMA3CC error interrupt. If EDMA3CC error interrupt is enabled in the device interrupt controller, then it allows the CPU to handle the error conditions.

The EDMA3CC has a single error interrupt (EDMA3_CC0_ERRINT) that gets asserted for all EDMA3CC error conditions. There are four conditions that cause the error interrupt to be pulsed:

- DMA missed events: for all 32 DMA channels. These get latched in the event missed registers (EMR).
- QDMA missed events: for all QDMA channels. These get latched in the QDMA event missed register (QEMR).
- Threshold exceed: for all event queues. These get latched in EDMA3CC error register (CCERR).
- TCC error: for outstanding transfer requests expected to return completion code (TCCHEN or TCINTEN bit in OPT is set to 1) exceeding the maximum limit of 31. This also gets latched in the EDMA3CC error register (CCERR).

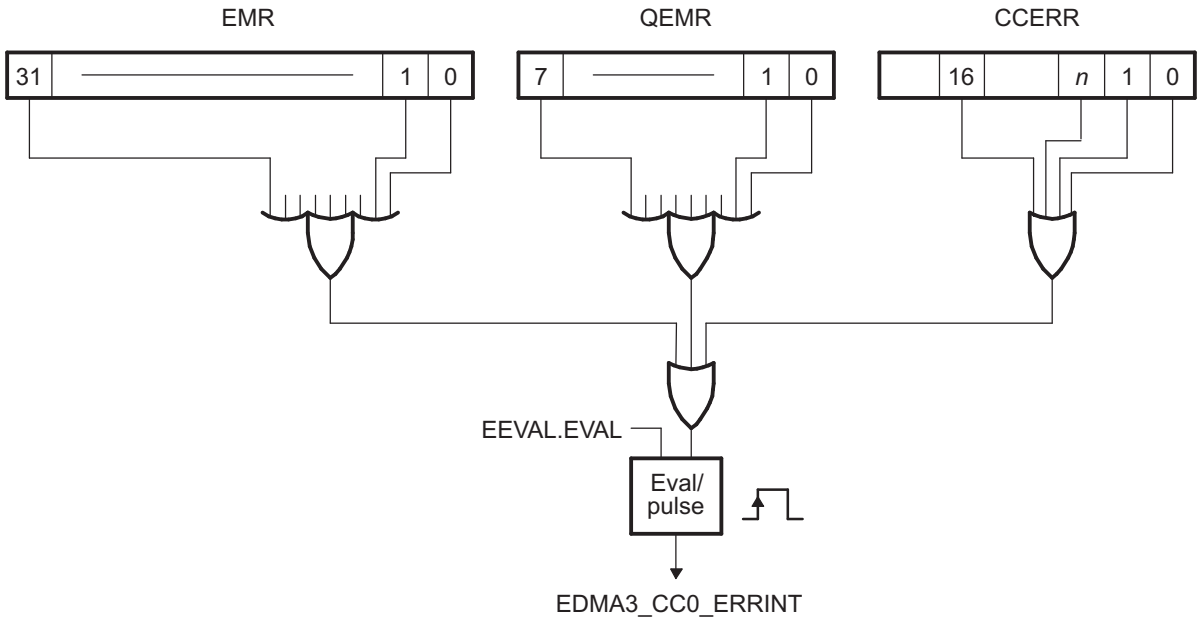
Figure 16-13 illustrates the EDMA3CC error interrupt generation operation.

If any of the bits are set in the error registers due to any error condition, the (EDMA3_CC0_ERRINT) always is asserted, as there are no enables for masking these error events. Similar to the transfer completion interrupts, the error interrupt also is pulsed only when the error interrupt condition transitions from a state where no errors are set to a state where at least one error bit is set. If additional error events are latched prior to the original error bits being cleared, the EDMA3CC does not generate additional interrupt pulses.

To reduce the burden on the software, similar to the interrupt evaluate register (IEVAL), there is an error evaluate register (EEVAL) that allows reevaluation of pending set error events/bits. This can be used so that the CPU(s) does not miss any error events.

NOTE: It is a good practice to have the error interrupt enabled in the device interrupt controller and associate an interrupt service routine with it to address the various error conditions appropriately. This puts less burden on software (polling for error status) and additionally provides a good debug mechanism for unexpected error conditions.

Figure 16-13. Error Interrupt Operation



Note: *n* is the number of queues supported in the EDMA3CC for a specific device.

Copyright © 2013–2016, Texas Instruments Incorporated

16.2.10 Event Queue(s)

Event queues are a part of the EDMA3 channel controller. Event queues form the interface between the event detection logic in the EDMA3CC and the transfer request (TR) submission logic of the EDMA3CC. Each queue is 16 entries deep, that is, a maximum of 16 queued events per event queue. If there are more than 16 events, then the events that cannot find a place in the event queue remain set in the associated event register.

The number of event queues in the EDMA3CC determines the number of transfer controllers connected to the EDMA3CC. By default, there is a one-to-one mapping between the queues and transfer controllers. Therefore, the transfer requests (TRs) associated with events in Q0 get submitted to TC0. Similarly, transfer requests associated with events in Q1 get submitted to TC1, and so on.

An event that wins prioritization against other DMA and/or QDMA pending events is placed at the end of the appropriate event queue. Each event queue is serviced in a FIFO (first in–first out) order. Once the event reaches the head of its queue and the corresponding transfer controller is ready to receive another TR, the event is dequeued and the PaRAM set corresponding to the dequeued event is processed and submitted as a transfer request packet (TRP) to the associated EDMA3 transfer controller.

A lower numbered queue has a higher dequeuing priority then a higher numbered queue. For example, Q0 has higher priority than Q1, if Q0 and Q1 both have at least one event entry and if both TC0 and TC1 can accept transfer requests, then the event in Q0 is dequeued first and its associated PaRAM set is processed and submitted as a transfer request (TR) to TC0.

All the event entries in all the event queues are software readable (not writeable) by accessing the event queue entry registers (Q*x*E*y*). Each event entry register characterizes the queued event in terms of the type of event (manual, event, chained or autotriggered) and the event number. See Section 16.4.2.4.1 for a description of the bit fields in the queue event entry registers.

16.2.10.1 DMA/QDMA Channel to Event Queue Mapping

Each DMA channel and QDMA channel is independently programmed to map to a specific queue using the DMA queue number register *n* (DMAQNUM*n*) and the QDMA channel queue number register (QDMANUM). The mapping of DMA/QDMA channels is critical to achieving the desired performance level for the EDMA and most importantly in meeting real-time deadlines.

NOTE: If an event is ready to be queued and both the event queue and the EDMA3 transfer controller associated to the event queue are empty, then the event bypasses the event queue, and goes to the PaRAM processing logic and eventually to the transfer request submission logic for submission to the EDMA3TC. In this case, the event is not logged in the event queue status registers.

Copyright © 2013–2016, Texas Instruments Incorporated

16.2.10.2 Queue RAM Debug Visibility

Each event queue has 16 entries. These 16 entries are managed in a circular FIFO manner. All event queue entries for all event queues are software readable by the event queue entry register (Q*x*E*x*). Additionally, for each queue there is a queue status register (QSTAT*n*).

These registers provide user visibility and may be helpful while debugging real-time issues (typically post-mortem), involving multiple events and event sources. The event queue entry register (Q*x*E*x*) uniquely identifies the specific event type (event-triggered, manually-triggered, chain-triggered, and QDMA events) along with the event number (for DMA/QDMA channels) that are in the queue or have been de-queued (passed through the queue). QSTAT*n* includes fields for the start pointer (STRTPTR) that provides the offset to the head entry of an event. It also includes a NUMVAL field that provides the total number of valid entries residing in the event queue at a given instance of time. The STRTPTR field may be used to index appropriately into the 16 event entries. The NUMVAL number of entries starting from STRTPTR are indicative of events still queued in the respective queue. The remaining entries may be read to determine which events have already been de-queued and submitted to the associated transfer controller.

16.2.10.3 Queue Resource Tracking

The EDMA3CC event queue includes watermarking/threshold logic that allows you to keep track of maximum usage of all event queues. This is useful for debugging real-time deadline violations that may result from head-of-line blocking on a given EDMA3 event queue.

You can program the maximum number of events that can queue up in an event queue by programming the threshold value (between 0 to 15) in the queue watermark threshold A register (QWMTHRA). The maximum queue usage is recorded actively in the watermark (WM) field of the queue status register (QSTAT*n*) that keeps getting updated based on a comparison of number of valid entries, which is also visible in the NUMVAL bit in QSTAT*n* and the maximum number of entries (WM bit in QSTAT*n*).

If the queue usage is exceeded, this status is visible in the EDMA3CC registers: the QTHRXCD*n* bit in the channel controller error register (CCERR) and the THRXCD bit in QSTAT*n*, where *n* stands for the event queue number. Any bits that are set in CCERR also generate an EDMA3CC error interrupt.

Copyright © 2013–2016, Texas Instruments Incorporated

16.2.11 EDMA3 Transfer Controller (EDMA3TC)

The EDMA3 channel controller is the user-interface of the EDMA3 and the EDMA3 transfer controller (EDMA3TC) is the data movement engine of the EDMA3. The EDMA3CC submits transfer requests (TR) to the EDMA3TC and the EDMA3TC performs the data transfers dictated by the TR.

16.2.11.1 Architecture Details

16.2.11.1.1 EDMA3TC Configuration

Each transfer controller on a device is designed differently based on considerations like performance requirements, system topology (main SCR bus width, external memory bus width), gate count, etc. The parameters that determine the TC configurations are:

- FIFOSIZE: Determines the size in bytes for the Data FIFO that is the temporary buffer for the in-flight data. The data FIFO is where the read return data read by the TC read controller from the source endpoint is stored and subsequently written out to the destination endpoint by the TC write controller.
- Default Burst Size (DBS): The DBS is the maximum number of bytes per read/write command issued by a transfer controller.
- BUSWIDTH: The width of the read and write data buses in bytes, for the TC read and write controller, respectively. This is typically equal to the bus width of the main SCR interface.
- DSTREGDEPTH: This determines the number of Destination FIFO register set. The number of Destination FIFO register set for a transfer controller, determines the maximum number of outstanding transfer requests (TR pipelining).

Of the four parameters, the FIFOSIZE, BUSWIDTH, and DSTREGDEPTH values are fixed in design for a given device. The default burst size (DBS) for each transfer controller is configurable by the chip configuration 0 register (CFGCHIP0) in the System Configuration Module.

The burst size for each transfer controlled can be programmed to be 16-, 32-, or 64-bytes. The default values for DBS are typically chosen for optimal performance in most intended-use conditions; therefore, if you decide to use a value other then the default, you should evaluate the impact on performance. Depending on the FIFOSIZE and source/destination locations the performance for the transfer can vary significantly for different burst size values.

NOTE: It is expected that the DBS value for a transfer controller is static and should be based on the application requirement. It is not recommended that the DBS value be changed on-the-fly.

16.2.11.1.2 Command Fragmentation

The TC read and write controllers in conjunction with the source and destination register sets are responsible for issuing optimally-sized reads and writes to the slave endpoints. The transfer controller read/write transaction as specified by the transfer request packet is internally broken down into smaller bursts; this determines the default burst size (DBS) for the transfer controller. See Section 16.2.11.1.1 for the DBS value of each EDMA3TC.

The EDMA3TC attempts to issue the largest possible command size as limited by the DBS value or the ACNT/BCNT value of the TR. EDMA3TC obeys the following rules:

- The read/write controllers always issue commands less than or equal to the DBS value.
- The first command of a 1D transfer is always issued so that subsequent commands align to the DBS value.

Example 16-4 shows the command fragmentation for a DBS of 32 bytes. In summary, if the ACNT value is larger than the DBS value, then the EDMA3TC breaks the ACNT array into DBS-sized commands to the source/destination addresses. Each BCNT number of arrays are then serviced in succession.

Copyright © 2013–2016, Texas Instruments Incorporated

Example 16-4. Command Fragmentation (DBS = 32)

The pseudo code:

1. ACNT = 8, BCNT = 8, SRCBIDX = 8, DSTBIDX = 10, SRCADDR = 64, DSTADDR = 191

   Read Controller: This is optimized from a 2D-transfer to a 1D-transfer such that the read side is equivalent to ACNT = 64, BCNT = 1.

   Cmd0 = 32 byte, Cmd0 = 32 byte

   Write Controller: Since DSTBIDX != ACNT, it is not optimized.

   Cmd0 = 8 byte, Cmd1 = 8 byte, Cmd2 = 8 byte, Cmd3 = 8 byte, Cmd4 = 8 byte, Cmd5 = 8 byte, Cmd6 = 8 byte, Cmd7 = 8 byte.

2. ACNT = 64, BCNT = 1, SRCADDR = 31, DSTADDR = 513

   Read Controller: Read address is not aligned.

   Cmd0 = 1 byte, (now the SRCADDR is aligned to 32 for the next command)

   Cmd1 = 32 bytes

   Cmd2 = 31 bytes

   Write Controller: The write address is also not aligned.

   Cmd0 = 31 bytes, (now the DSTADDR is aligned to 32 for the next command)

   Cmd1 = 32 bytes

   Cmd2 = 1 byte

16.2.11.1.3 TR Pipelining and Data Ordering

The transfer controller(s) can issue back-to-back transfer requests (TR). The number of outstanding TRs for a TC is limited by the number of destination FIFO register entries that is controlled by the DSTREGDEPTH parameter (fixed in design for a given transfer controller). TR pipelining refers to the ability of the TC read controller to issue read commands for a subsequent TR, while the TC write controller is still performing writes for the previous TR. Consider the case of 2 TRs (TR0 followed by TR1), because of TR pipelining, the TC read controller can start issuing the read commands for TR1 as soon as the last read command for TR0 has been issued, meanwhile the write commands and write data for TR0 are tracked by the destination FIFO registers. In summary, the TC read controller is able to process *n* TRs ahead of the write controller, where *n* is the number of destination FIFO register entries (typically 4).

TR pipelining is useful for maintaining throughput on back-to-back small TRs. It eliminates the read overhead because reads start in the background of a previous TR writes.

It should be noted that back-to-back TRs are targeted to different end points even though the read return data for the two TRs might get returned out of order (that is, read data for TR1 might come in before read data for TR0), the transfer controller issues that the write commands are issued in order (that is, write commands for TR0 will be issued before write commands for TR1).

Copyright © 2013–2016, Texas Instruments Incorporated

16.2.11.2 Error Generation

Similar to the channel controller, the transfer controllers are capable of detecting and reporting several error conditions. The TC errors are generated, under three main conditions:

- BUSERR: The TC read or write controllers detect an error signaled by the source or destination address. The additional details on the type of error is also recorded in the ERRDET register, which indicates whether it is a read error (source address errors) or write error (destination address error).
- MMRAERR: CPU accesses illegal/reserved addresses in the EDMA3CC/TC memory-map.
- TRERR: A transfer request packet is detected to be violating the constant addressing mode transfer rules (the source/destination addresses and source/destination indexes must be aligned to 32 bytes).

You can poll for the errors, as the status of the errors can be read from the ERRSTAT registers, additionally if the error bits are enabled in the ERREN register, a bit set in the ERRSTAT will cause the error condition to interrupt the CPU(s). You can decide to enable/disable either or all error types.

16.2.11.3 Debug Features

The DMA program register set, DMA source active register set, and the destination FIFO register set are used to derive a brief history of TRs serviced through the transfer controller.

Additionally, the EDMA3TC status register (TCSTAT) has dedicated bit fields to indicate the ongoing activity within different parts of the transfer controller:

- The SRCACTV bit indicates whether the source active set is active.
- The DSTACTV bit indicates the number of TRs resident in the destination register active set at a given instance.
- The PROGBUSY bit indicates whether a valid TR is present in the DMA program set.

If the TRs are in progression, caution must be used and you must realize that there is a chance that the values read from the EDMA3TC status registers will be inconsistent since the EDMA3TC may change the values of these registers due to ongoing activities.

It is recommended that you ensure no additional submission of TRs to the EDMA3TC in order to facilitate ease of debug.

16.2.11.3.1 Destination FIFO Register Pointer

The destination FIFO register pointer is implemented as a circular buffer with the start pointer being DFSTRTPTR and a buffer depth of usually 2 or 4. The EDMA3TC maintains two important status details in TCSTAT that may be used during advanced debugging, if necessary. The DFSTRTPTR is a start pointer, that is, the index to the head of the destination FIFO register. The DSTACTV is a counter for the number of valid (occupied) entries. These registers may be used to get a brief history of transfers.

Examples of some register field values and their interpretation:

- DFSTRTPTR = 0 and DSTACTV = 0 implies that no TRs are stored in the destination FIFO register.
- DFSTRTPTR = 1 and DSTACTV = 2h implies that two TRs are present. The first pending TR is read from the destination FIFO register entry 1 and the second pending TR is read from the destination FIFO register entry 2.
- DFSTRTPTR = 3h and DSTACTV = 2h implies that two TRs are present. The first pending TR is read from the destination FIFO register entry 3 and the second pending TR is read from the destination FIFO register entry 0.

Copyright © 2013–2016, Texas Instruments Incorporated

16.2.12 Event Dataflow

This section summarizes the data flow of a single event, from the time the event is latched to the channel controller to the time the transfer completion code is returned. The following steps list the sequence of EDMA3CC activity:

1. Event is asserted from an external source (peripheral or external interrupt). This also is similar for a manually-triggered, chained-triggered, or QDMA-triggered event. The event is latched into the ER.E*n* (or CER.E*n*, ESR.E*n*, QER.E*n*) bit.
2. Once an event is prioritized and queued into the appropriate event queue, the SER.E*n* (or QSER.E*n*) bit is set to inform the event prioritization/processing logic to disregard this event since it is already in the queue. Alternatively, if the transfer controller and the event queue are empty, then the event bypasses the queue.
3. The EDMA3CC processing and the submission logic evaluates the appropriate PaRAM set and determines whether it is a non-null and non-dummy transfer request (TR).
4. The EDMA3CC clears the ER.E*n* (or CER.E*n*, ESR.E*n*, QER.E*n*) bit and the SER.E*n* bit as soon as it determines the TR is non-null. In the case of a null set, the SER.E*n* bit remains set. It submits the non-null/non-dummy TR to the associated transfer controller. If the TR was programmed for early completion, the EDMA3CC immediately sets the interrupt pending register (IPR.I[TCC]).
5. If the TR was programmed for normal completion, the EDMA3CC sets the interrupt pending register (IPR.I[TCC]) when the EDMA3TC informs the EDMA3CC about completion of the transfer (returns transfer completion codes).
6. The EDMA3CC programs the associated EDMA3TC*n* Program Register Set with the TR.
7. The TR is then passed to the Source Active set and the Dst FIFO Register Set, if both the register sets are available.
8. The Read Controller processes the TR by issuing read commands to the source slave endpoint. The Read Data lands in the Data FIFO of the EDMA3TC*n*.
9. As soon as sufficient data is available, the Write Controller begins processing the TR by issuing write commands to the destination slave endpoint.
10. This continues until the TR completes and on receiving the acknowledgement signal from the destination slave end point, the EDMA3TC*n* then signals completion status to the EDMA3CC.

Copyright © 2013–2016, Texas Instruments Incorporated

16.2.13 EDMA3 Prioritization

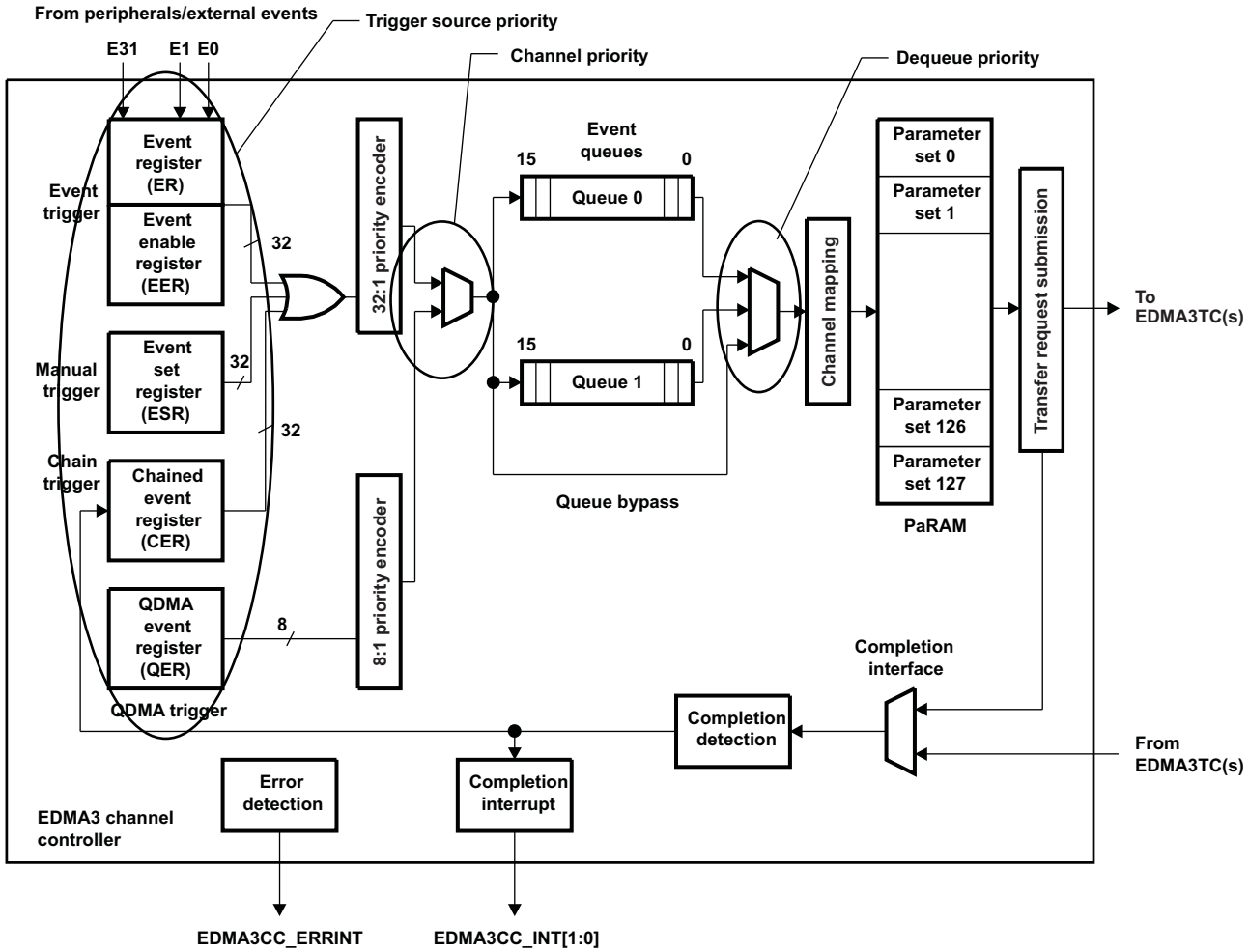
The EDMA3 controller has many implementation rules to deal with concurrent events/channels, transfers, etc. The following subsections detail various arbitration details whenever there might be occurrence of concurrent activity. Figure 16-14 shows the different places EDMA3 priorities come into play.

16.2.13.1 Channel Priority

The DMA event register (ER) captures all external/peripheral events connected to the EDMA3CC; likewise, the QDMA event register (QER) captures QDMA events for all QDMA channels; therefore, it is possible for events to occur simultaneously on the DMA/QDMA event inputs. For events arriving simultaneously, the event associated with the lowest channel number is prioritized for submission to the event queues (for DMA events, channel 0 has the highest priority and channel 31 has the lowest priority; similarly, for QDMA events, channel 0 has the highest priority and channel 7 has the lowest priority). This mechanism only sorts simultaneous events for submission to the event queues.

If a DMA and QDMA event occurs simultaneously, the DMA event always has prioritization against the QDMA event for submission to the event queues.

Figure 16-14. EDMA3 Prioritization



Copyright © 2013–2016, Texas Instruments Incorporated

16.2.13.2 Trigger Source Priority

If a DMA channel is associated with more than one trigger source (event trigger, manual trigger, and chain trigger), and if multiple events are set simultaneously for the same channel (ER.E*n* = 1, ESR.E*n* = 1, CER.E*n* = 1), then the EDMA3CC always services these events in the following priority order: event trigger (via ER) is higher priority than chain trigger (via CER) and chain trigger is higher priority than manual trigger (via ESR).

This implies that if for channel 0, both ER.E0 = 1 and CER.E0 = 1 at the same time, then the ER.E0 event is always queued before the CER.E0 event.

16.2.13.3 Dequeue Priority

The priority of the associated transfer request (TR) is further mitigated by which event queue is being used for event submission (dictated by DMAQNUM*n* and QDMAQNUM). For submission of a TR to the transfer controller, events need to be dequeued from the event queues. A lower numbered queue has a higher dequeuing priority then a higher numbered queue. For example, if there are events in Q0 and Q1 and the respective transfer controllers (TC0 and TC1) are ready to receive the next TR from the EDMA3CC, then the transfer requests associated with events in Q0 will get submitted to TC0 prior to any transfer requests associated with events in Q1 getting submitted to TC1.

NOTE: At any given time, if there are outstanding events in multiple queues, when the transfer controller associated with the lower numbered (higher priority) queue is busy processing earlier transfer requests and the transfer controller associated with the higher numbered (lower priority) queue is idle, then the event in the higher numbered (lower priority) queue will dequeue first.

16.2.13.4 Master (Transfer Controller) Priority

All master peripherals on the device have a programmable priority level. When multiple masters are trying to access common shared resources (slave memory or peripherals), this priority value allows the system interconnect to arbitrate requests from different masters based on their priority. This priority assignment is determined in the Master Priority Registers (MSTPRI0-MSTPRI2) in the System Configuration Module (see the *System Configuration (SYSCFG) Module* chapter), where each master has an allocated priority value (power on reset default value), which can be re programmed based on the applications prioritization requirements. The priority value can be configured between 0 to 7, with 0 being the highest priority and 7 being the lowest priority.

Each transfer controller on the device is also a master peripheral. The priority of the transfer requests (read/write commands) issued by the individual EDMA3TC read/write ports in the system can be programmed via these registers.

The dequeue priority has a relatively secondary effect as compared to this Master priority; therefore, it is important to program the priority of each transfer controller with respect to each other and also with respect to other masters in the system.

NOTE: On previous architectures, the EDMA3TC priority was controlled by the QUEPRI register in the EDMA3CC memory-map. However for this device, the priority control for the transfer controllers is controlled by the chip-level registers in the System Configuration Module.

Copyright © 2013–2016, Texas Instruments Incorporated

16.2.14 EDMA3CC and EDMA3TC Performance and System Considerations

16.2.14.1 System Priority Considerations

The main switched central resource (SCR) (see your device-specific data manual) arbitrates bus requests from all the masters (CPU, master peripherals, and the EDMA3 transfer controllers) to the shared slave resources (peripherals and memories). The priorities of transfer requests (read and write commands) from the EDMA3 transfer controllers with respect to each other and the other masters within the system is configured as explained in Section 16.2.13.4.

It is recommended that this priority be altered based on system level considerations. For example, peripherals servicing audio/video/display threads that typically have real-time deadlines should be programmed as highest priority requestors in the systems, where as, peripherals responsible for doing bulk/block/paging transfers with no real-time deadlines, should be programmed as a lower system priority.

The default priority for all transfer controllers is the same, 0 or highest priority relative to other masters; therefore, it is recommended that a TC servicing audio data requests from serial ports should be configured at a higher priority as compared to TC service memory to memory (paging/bulk) transfer requests.

16.2.14.2 TC Transfer Optimization Considerations

The transfer controller can internally optimize the way it issues read commands and write commands for a given transfer under certain conditions. For 2D transfers (that is, BCNT arrays of ACNT bytes), if the ACNT value is less than or equal to the DBS value, then the transfer controller will try to optimize the TR into a 1D transfer in order to maximize efficiency. The optimization only takes place if the EDMA3TC recognizes that the 2D transfer is organized as a single dimension (SAM/DAM = 0, increment mode), SRC/DST BIDX = ACNT, the ACNT value is a power of 2, and the BCNT value is less than or equal to 1023. If these conditions are met, then instead of issuing ACNT bytes worth read and/or write commands, the TC will try to optimize the bus usage by issuing commands as if ACNT' = ACNT × BCNT and BCNT = 1.

Table 16-10 summarizes the conditions in which the optimizations are performed.

Table 16-10. Read/Write Command Optimization Rules

ACNT ≤ DBS	ACNT is power of 2	BIDX = ACNT	BCNT ≤ 1023	SAM/DAM = 0 (Increment)	Description
Yes	Yes	Yes	Yes	Yes	Optimized
Yes	No	x	x	Yes	Not Optimized
Yes	x	No	x	Yes	Not Optimized
No	x	x	x	Yes	Not Optimized
x	x	x	x	No	Not Optimized

Consider a case in which it is needed to transfer 4096 bytes where the data is arranged linearly in both the source and destination locations (SAM/DAM = 0, SRC/DST BIDX = ACNT): Scenario A programs the ACNT = 4, BCNT = 1024 , AB-synchronized transfer; and Scenario B programs the ACNT = 64, BCNT = 64. Scenario B will yield a much optimized transfer and higher throughput, as the transfer meets all the optimization rules, which would result in TC internally treating it as a transfer with an ACNT' = 4096 (ACNT × BCNT). The TC will optimally size, default burst size worth read and write commands. In the case of Scenario B, since one of the optimization rules is not met (BCNT value is greater then 1023), the TC will end up issuing several ACNT byte (4 byte) size commands to complete the transfers, which will result in inefficient usage of the read/write buses.

Copyright © 2013–2016, Texas Instruments Incorporated

16.2.14.3 Throttling the Read Command Rate in a Transfer Controller

By default, the transfer controller issues reads as fast as possible. In some cases, the reads issued by the EDMA3TCC could fill the available command buffering for a slave, delaying other (potentially higher priority) masters from successfully submitting commands to that slave. The rate at which read commands are issued by the EDMA3TC is controlled by the read command rate register (RDRATE), and this can be used to throttle the rate at which the commands are issued from the TC read interface. RDRATE defines the number of cycles that the EDMA3TC read controller waits before issuing subsequent commands for a given TR, thus minimizing the chance of the EDMA3TC consuming all available slave resources. The RDRATE value should be set to a relatively small value (or kept at default, which implies issuing read requests as fast as possible) if the transfer controller is targeted for high-priority transfers and set to a high value if the transfer controller is targeted for low-priority transfers. In contrast, the write Interface does not have any performance turning knobs because writes always have an interval between commands as write commands are submitted along with the associated write data.

16.2.15 EDMA3 Operating Frequency (Clock Control)

The EDMA3 channel controller and transfer controller are clocked from PLL controller 0 (PLLC0). For details, see the *Phase-Locked Loop Controller (PLLC)* chapter.

16.2.16 Reset Considerations

A hardware reset resets the EDMA3 (EDMA3CC and EDMA3TC) and the EDMA3 configuration registers. The PaRAM memory contents are undefined after device reset and you should not rely on parameters to be reset to a known state. The PaRAM set must be initialized to a desired value before it is used.

16.2.17 Power Management

The EDMA3 (EDMA3CC and EDMA3TC) can be placed in reduced-power modes to conserve power during periods of low activity. The power management of the peripheral is controlled by the device Power and Sleep Controller (PSC). The PSC acts as a master controller for power management for all peripherals on the device. For detailed information on power management procedures using the PSC, see the *Power and Sleep Controller (PSC)* chapter.

The EDMA3 controller can be idled on receiving a clock stop request from the PSC. The requests to EDMA3CC and EDMA3TC are separate. In general, you should verify that there are no pending activities in the EDMA3 controller before issuing a clock stop request via PSC.

The EDMA3CC checks for the following conditions:

- No pending DMA/QDMA events
- No outstanding events in the event queues
- Transfer request processing logic is not active
- No completion requests outstanding (early or normal completion)
- No configuration bus requests in progress

The first four conditions are software readable by the channel controller status register (CCSTAT) in the EDMA3CC.

Similarly, from the EDMA3TC perspective, you should check that there are no outstanding TRs that are getting processed and essentially the read/write controller is not busy processing a TR. The activity of EDMA3TC logic is read in TCSTAT for each EDMA3TC.

It is generally recommended to first disable the EDMA3CC and then the EDMA3TC(s) to put the EDMA3 controller in reduced-power modes.

Additionally, when EDMA3 is involved in servicing a peripheral and it is required to power-down both the peripheral and the EDMA, the recommended sequence is to first disable the peripheral, then disable the DMA channel associated with the peripheral (clearing the EER bit for the channel), then disable the EDMA3CC, and finally disable the EDMA3TC(s).

Copyright © 2013–2016, Texas Instruments Incorporated

16.2.18 Emulation Considerations

During debug when using the emulator, the CPU(s) may be halted on an execute packet boundary for single-stepping, benchmarking, profiling, or other debug purposes. During an emulation halt, the EDMA3 channel controller and transfer controller operations continue. Events continue to be latched and processed and transfer requests continue to be submitted and serviced.

Since EDMA3 is involved in servicing multiple master and slave peripherals, it is not feasible to have an independent behavior of the EDMA3 for emulation halts. EDMA3 functionality would be coupled with the peripherals it is servicing, which might have different behavior during emulation halts. For example, if a multichannel buffered serial port (McBSP) is halted during an emulation access (FREE = 0 and SOFT = 0 or 1 in the McBSP registers), the McBSP stops generating the McBSP receive or transmit events (REVT or XEVT) to the EDMA. From the point of view of the McBSP, the EDMA3 is suspended, but other peripherals (for example, a timer) still assert events and will be serviced by the EDMA.

16.3 Transfer Examples

The EDMA3 channel controller performs a variety of transfers depending on the parameter configuration. The following sections provides a description and PaRAM configuration for some typical use case scenarios.

16.3.1 Block Move Example

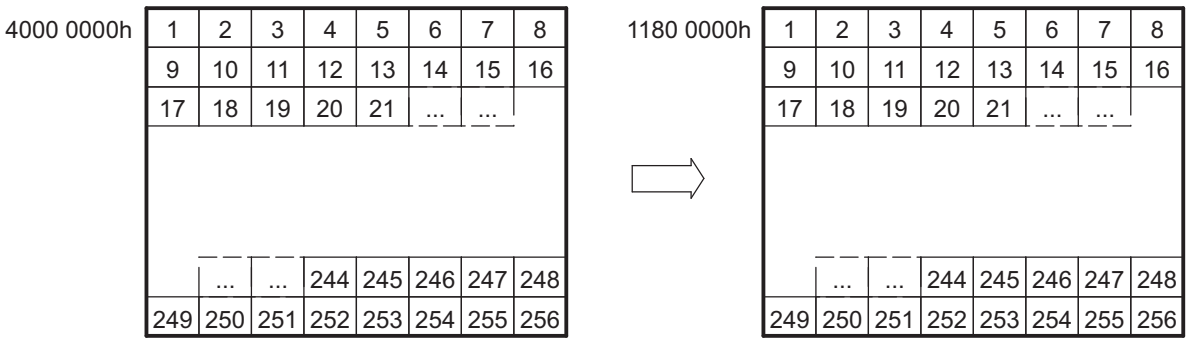
The most basic transfer performed by the EDMA3 is a block move. During device operation it is often necessary to transfer a block of data from one location to another, usually between on-chip and off-chip memory.

In this example, a section of data is to be copied from external memory to internal L2 SRAM. A data block of 256 bytes residing at address 4000 0000h (external memory) needs to be transferred to internal address 1180 0000h (L2), as shown in Figure 16-15. Figure 16-16 shows the parameters for this transfer.

The source address for the transfer is set to the start of the data block in external memory, and the destination address is set to the start of the data block in L2. If the data block is less than 64K bytes, the PaRAM configuration in Figure 16-16 holds true with the synchronization type set to A-synchronized and indexes cleared to 0. If the amount of data is greater than 64K bytes, BCNT and the B-indexes need to be set appropriately with the synchronization type set to AB-synchronized. The STATIC bit in OPT is set to prevent linking.

This transfer example may also be set up using QDMA. For successive transfer submissions, of a similar nature, the number of cycles used to submit the transfer are fewer depending on the number of changing transfer parameters. You may program the QDMA trigger word to be the highest numbered offset in the PaRAM set that undergoes change.

Figure 16-15. Block Move Example



Copyright © 2013–2016, Texas Instruments Incorporated

Figure 16-16. Block Move Example PaRAM Configuration

(a) EDMA Parameters

Parameter Contents		Parameter	
0010 0008h		Channel Options Parameter (OPT)	
4000 0000h		Channel Source Address (SRC)	
0001h	0100h	Count for 2nd Dimension (BCNT)	Count for 1st Dimension (ACNT)
1180 0000h		Channel Destination Address (DST)	
0000h	0000h	Destination BCNT Index (DSTBIDX)	Source BCNT Index (SRCBIDX)
0000h	FFFFh	BCNT Reload (BCNTRLD)	Link Address (LINK)
0000h	0000h	Destination CCNT Index (DSTCIDX)	Source CCNT Index (SRCCIDX)
0000h	0001h	Reserved	Count for 3rd Dimension (CCNT)

(b) Channel Options Parameter (OPT) Content

31	30 28	27 24	23	22	21	20	19 18	17 16
0	000	0000	0	0	0	1	00	00
PRIV	Reserved	PRIVID	ITCCHEN	TCCHEN	ITCINTEN	TCINTEN	Reserved	TCC

15 12	11	10 8	7 4	3	2	1	0
0000	0	000	0000	1	0	0	0
TCC	TCCMOD	FWID	Reserved	STATIC	SYNCDIM	DAM	SAM

Copyright © 2013–2016, Texas Instruments Incorporated

16.3.2 Subframe Extraction Example

The EDMA3 can efficiently extract a small frame of data from a larger frame of data. By performing a 2D-to-1D transfer, the EDMA3 retrieves a portion of data for the CPU to process. In this example, a 640 × 480-pixel frame of video data is stored in external memory, SDRAM. Each pixel is represented by a 16-bit halfword. The CPU extracts a 16 × 12-pixel subframe of the image for processing. To facilitate more efficient processing time by the CPU, the EDMA3 places the subframe in internal L2 SRAM. Figure 16-17 shows the transfer of a subframe from external memory to L2. Figure 16-18 shows the parameters for this transfer.

The same PaRAM set options are used for QDMA channels, as well as DMA channels. The STATIC bit in OPT is set to 1 to prevent linking. For successive transfers, only changed parameters need to be programmed before triggering the channel.

Figure 16-17. Subframe Extraction Example

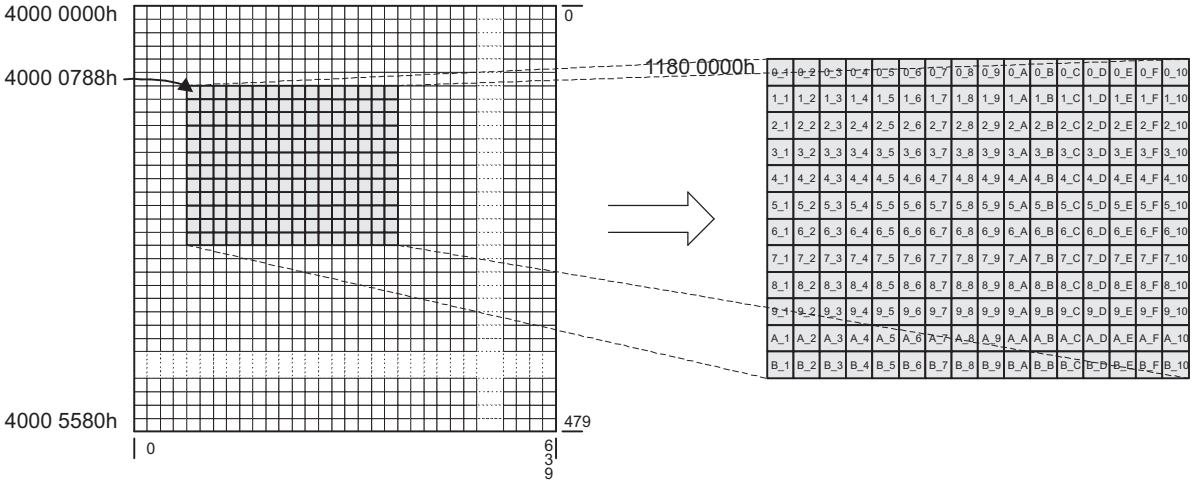


Figure 16-18. Subframe Extraction Example PaRAM Configuration

(a) EDMA Parameters

Parameter Contents	
0010 000Ch	
4000 0788h	
000Ch	0020h
1180 0000h	
0020h	0500h
0000h	FFFFh
0000h	0000h
0000h	0001h

Parameter	
Channel Options Parameter (OPT)	
Channel Source Address (SRC)	
Count for 2nd Dimension (BCNT)	Count for 1st Dimension (ACNT)
Channel Destination Address (DST)	
Destination BCNT Index (DSTBIDX)	Source BCNT Index (SRCBIDX)
BCNT Reload (BCNTRLD)	Link Address (LINK)
Destination CCNT Index (DSTCIDX)	Source CCNT Index (SRCCIDX)
Reserved	Count for 3rd Dimension (CCNT)

(b) Channel Options Parameter (OPT) Content

31	30 28	27 24	23	22	21	20	19 18	17 16
0	000	0000	0	0	0	1	00	00
PRIV	Reserved	PRIVID	ITCCHEN	TCCHEN	ITCINTEN	TCINTEN	Reserved	TCC

15 12	11	10 8	7 4	3	2	1	0
0000	0	000	0000	1	1	0	0
TCC	TCCMOD	FWID	Reserved	STATIC	SYNCDIM	DAM	SAM

Copyright © 2013–2016, Texas Instruments Incorporated

16.3.3 Data Sorting Example

Many applications require the use of multiple data arrays; it is often desirable to have the arrays arranged such that the first elements of each array are adjacent, the second elements are adjacent, and so on. Often this is not how the data is presented to the device. Either data is transferred via a peripheral with the data arrays arriving one after the other or the arrays are located in memory with each array occupying a portion of contiguous memory spaces. For these instances, the EDMA3 can reorganize the data into the desired format. Figure 16-19 shows the data sorting.

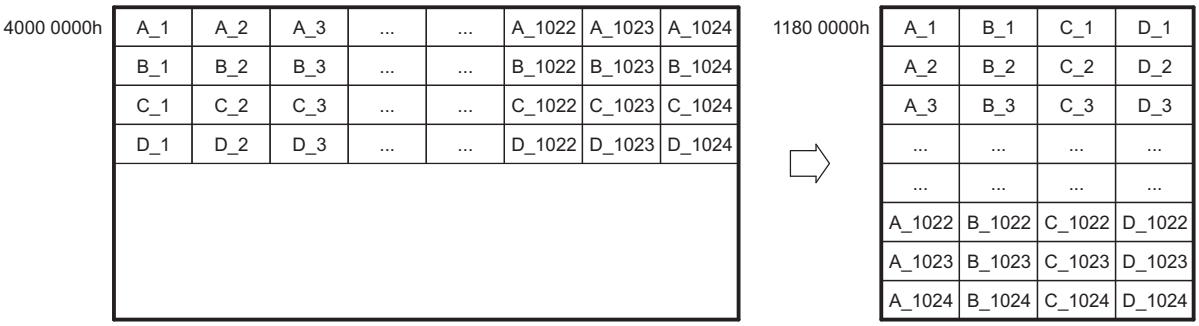
In order to determine the parameter entry values, the following need to be considered:

- ACNT – Program this to be the size in bytes of an array.
- BCNT – Program this to be the number of arrays in a frame.
- CCNT – Program this to be the number of frames.
- SRCBIDX – Program this to be the size of the array or ACNT.
- DSTBIDX = CCNT × ACNT
- SRCCIDX = ACNT × BCNT
- DSTCIDX = ACNT

The synchronization type needs to be AB-synchronized and the STATIC bit is 0 to allow updates to the parameter set. It is advised to use normal DMA channels for sorting.

It is not possible to sort this with a single trigger event. Instead, the channel can be programmed to be chained to itself. After BCNT arrays get sorted, intermediate chaining could be used to trigger the channel again causing the transfer of the next BCNT arrays and so on. Figure 16-20 shows the parameter set programming for this transfer, assuming channel 0 and an array size of 4 bytes.

Figure 16-19. Data Sorting Example



Copyright © 2013–2016, Texas Instruments Incorporated

Figure 16-20. Data Sorting Example PaRAM Configuration

(a) EDMA Parameters

Parameter Contents	Parameter
0090 0004h	Channel Options Parameter (OPT)
4000 0000h	Channel Source Address (SRC)
0400h | 0004h	Count for 2nd Dimension (BCNT) | Count for 1st Dimension (ACNT)
1180 0000h	Channel Destination Address (DST)
0010h | 0004h	Destination BCNT Index (DSTBIDX) | Source BCNT Index (SRCBIDX)
0000h | FFFFh	BCNT Reload (BCNTRLD) | Link Address (LINK)
0004h | 1000h	Destination CCNT Index (DSTCIDX) | Source CCNT Index (SRCCIDX)
0000h | 0004h	Reserved | Count for 3rd Dimension (CCNT)

(b) Channel Options Parameter (OPT) Content

31	30 28	27 24	23	22	21	20	19 18	17 16
0	000	0000	1	0	0	1	00	00
PRIV	Reserved	PRIVID	ITCCHEN	TCCHEN	ITCINTEN	TCINTEN	Reserved	TCC

15 12	11	10 8	7 4	3	2	1	0
0000	0	000	0000	0	1	0	0
TCC	TCCMOD	FWID	Reserved	STATIC	SYNCDIM	DAM	SAM

Copyright © 2013–2016, Texas Instruments Incorporated

16.3.4 Peripheral Servicing Example

NOTE: Examples in this section are sample examples. The peripherals, channels, and addresses used in these examples may not apply to your specific device. See your device-specific data manual for supported peripherals.

The EDMA3 channel controller also services peripherals in the background of CPU operation, without requiring any CPU intervention. Through proper initialization of the DMA channels, they can be configured to continuously service on-chip and off-chip peripherals throughout the device operation. Each event available to the EDMA3 has its own dedicated channel, and all channels operate simultaneously. The only requirements are to use the proper channel for a particular transfer and to enable the channel event in the event enable register (EER). When programming a DMA channel to service a peripheral, it is necessary to know how data is to be presented to the CPU. Data is always provided with some kind of synchronization event as either one element per event (nonbursting) or multiple elements per event (bursting).

16.3.4.1 Nonbursting Peripherals

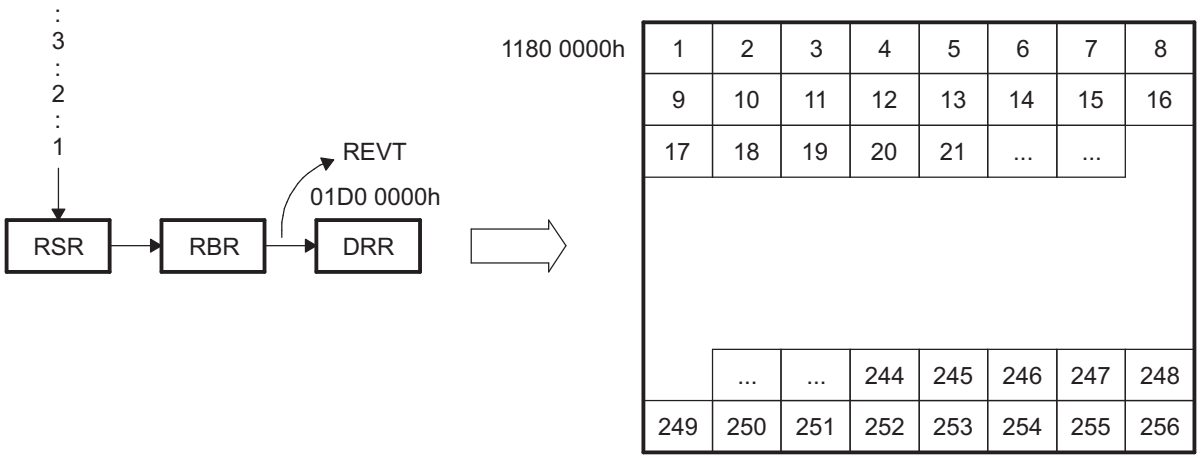
Nonbursting peripherals include the on-chip multichannel buffered serial port (McBSP) and many external devices, such as codecs. Regardless of the peripheral, the DMA channel configuration is the same.

The McBSP transmit and receive data streams are treated independently by the EDMA3. The transmit and receive data streams can have completely different counts, data sizes, and formats. Figure 16-21 shows servicing incoming McBSP data.

To transfer the incoming data stream to its proper location in L2 memory, the DMA channel must be set up for a 1D-to-1D transfer with A-synchronization. Since an event (REVT) is generated for every word as it arrives, it is necessary to have the EDMA3 issue the transfer request for each element individually. Figure 16-22 shows the parameters for this transfer. The source address of the DMA channel is set to the data receive register (DRR) address for the McBSP, and the destination address is set to the start of the data block in L2. Since the address of DRR is fixed, the source B index is cleared to 0 (no modification) and the destination B index is set to 01b (increment).

Based on the premise that serial data is typically a high priority, the DMA channel should be programmed to be on queue 0.

Figure 16-21. Servicing Incoming McBSP Data Example



Copyright © 2013–2016, Texas Instruments Incorporated

Figure 16-22. Servicing Incoming McBSP Data Example PaRAM

(a) EDMA Parameters

Parameter Contents	
0010 0000h	
01D0 0000h	
0100h	0001h
1180 0000h	
0001h	0000h
0000h	FFFFh
0000h	0000h
0000h	0004h

Parameter	
Channel Options Parameter (OPT)	
Channel Source Address (SRC)	
Count for 2nd Dimension (BCNT)	Count for 1st Dimension (ACNT)
Channel Destination Address (DST)	
Destination BCNT Index (DSTBIDX)	Source BCNT Index (SRCBIDX)
BCNT Reload (BCNTRLD)	Link Address (LINK)
Destination CCNT Index (DSTCIDX)	Source CCNT Index (SRCCIDX)
Reserved	Count for 3rd Dimension (CCNT)

(b) Channel Options Parameter (OPT) Content

31	30 28	27 24	23	22	21	20	19 18	17 16
0	000	0000	0	0	0	1	00	00
PRIV	Reserved	PRIVID	ITCCHEN	TCCHEN	ITCINTEN	TCINTEN	Reserved	TCC

15 12	11	10 8	7 4	3	2	1	0
0000	0	000	0000	0	0	0	0
TCC	TCCMOD	FWID	Reserved	STATIC	SYNCDIM	DAM	SAM

Copyright © 2013–2016, Texas Instruments Incorporated

16.3.4.2 Bursting Peripherals

Higher bandwidth applications require that multiple data elements be presented to the CPU for every synchronization event. This frame of data can either be from multiple sources that are working simultaneously or from a single high-throughput peripheral that streams data to/from the CPU. In this example, a port is receiving a video frame from a camera and presenting it to the CPU one array at a time. The video image is 640 × 480 pixels, with each pixel represented by a 16-bit element. The image is to be stored in external memory. Figure 16-23 shows this example.

To transfer data from an external peripheral to an external buffer one array at a time based on EVT*n* , channel *n* must be configured. Due to the nature of the data (a video frame made up of arrays of pixels) the destination is essentially a 2D entity. Figure 16-24 shows the parameters to service the incoming data with a 1D-to-2D transfer using AB-synchronization. The source address is set to the location of the video framer peripheral, and the destination address is set to the start of the data buffer. Since the input address is static, the SRCBIDX is 0 (no modification to the source address). The destination is made up of arrays of contiguous, linear elements; therefore, the DSTBIDX is set to pixel size, 2 bytes. ANCT is equal to the pixel size, 2 bytes. BCNT is set to the number of pixels in an array, 640. CCNT is equal to the total number of arrays in the block, 480. SRCCIDX is 0 since the source address undergoes no increment. The DSTCIDX is equal to the difference between the starting addresses of each array. Since a pixel is 16 bits (2 bytes), DSTCIDX is equal to 640 × 2.

Figure 16-23. Servicing Peripheral Burst Example

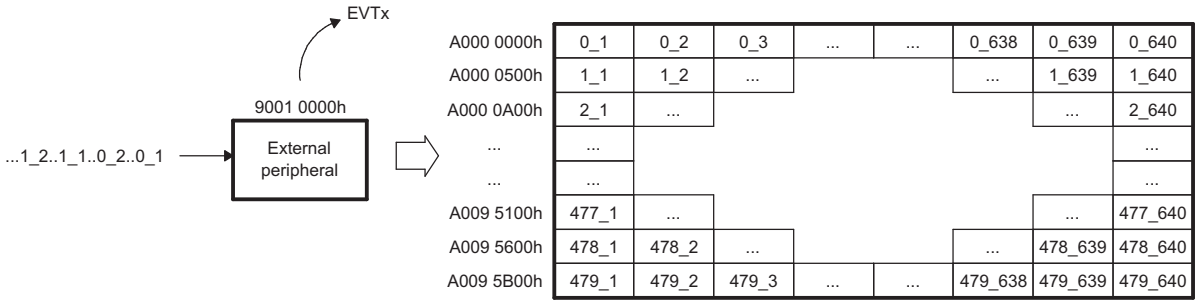


Figure 16-24. Servicing Peripheral Burst Example PaRAM

(a) EDMA Parameters

Parameter Contents	
0010 0004h	
Channel Source Address	
0280h	0002h
4000 0000h	
0002h	0000h
0000h	FFFFh
0500h	0000h
0000h	01E0h

Parameter	
Channel Options Parameter (OPT)	
Channel Source Address (SRC)	
Count for 2nd Dimension (BCNT)	Count for 1st Dimension (ACNT)
Channel Destination Address (DST)	
Destination BCNT Index (DSTBIDX)	Source BCNT Index (SRCBIDX)
BCNT Reload (BCNTRLD)	Link Address (LINK)
Destination CCNT Index (DSTCIDX)	Source CCNT Index (SRCCIDX)
Reserved	Count for 3rd Dimension (CCNT)

(b) Channel Options Parameter (OPT) Content

31	30 28	27 24	23	22	21	20	19 18	17 16
0	000	0000	0	0	0	1	00	00
PRIV	Reserved	PRIVID	ITCCHEN	TCCHEN	ITCINTEN	TCINTEN	Reserved	TCC

15 12	11	10 8	7 4	3	2	1	0
0000	0	000	0000	0	1	0	0
TCC	TCCMOD	FWID	Reserved	STATIC	SYNCDIM	DAM	SAM

Copyright © 2013–2016, Texas Instruments Incorporated

16.3.4.3 Continuous Operation

Configuring a DMA channel to receive a single frame of data is useful, and is applicable to some systems. A majority of the time, however, data is going to be continuously transmitted and received throughout the entire operation of the CPU. In this case, it is necessary to implement some form of linking such that the DMA channels continuously reload the necessary parameter sets. In this example, the multichannel buffered serial port (McBSP) is configured to transmit and receive data on a array. To simplify the example, only two channels are active for both transmit and receive data streams. Each channel receives packets of 128 elements. The packets are transferred from the serial port to L2 memory and from L2 memory to the serial port, as shown in Figure 16-25.

The McBSP generates REVT for every element received and generates XEVT for every element transmitted. To service the data streams, the DMA channels associated with the McBSP must be set up for 1D-to-1D transfers with A-synchronization.

Figure 16-26 shows the parameters for the parameter entries for the channel for these transfers. In order to service the McBSP continuously throughout CPU operation, the channels must be linked to a duplicate PaRAM set in the PaRAM. After all frames have been transferred, the DMA channels reload and continue. Figure 16-27 shows the reload parameters for the channel.

16.3.4.3.1 Receive Channel

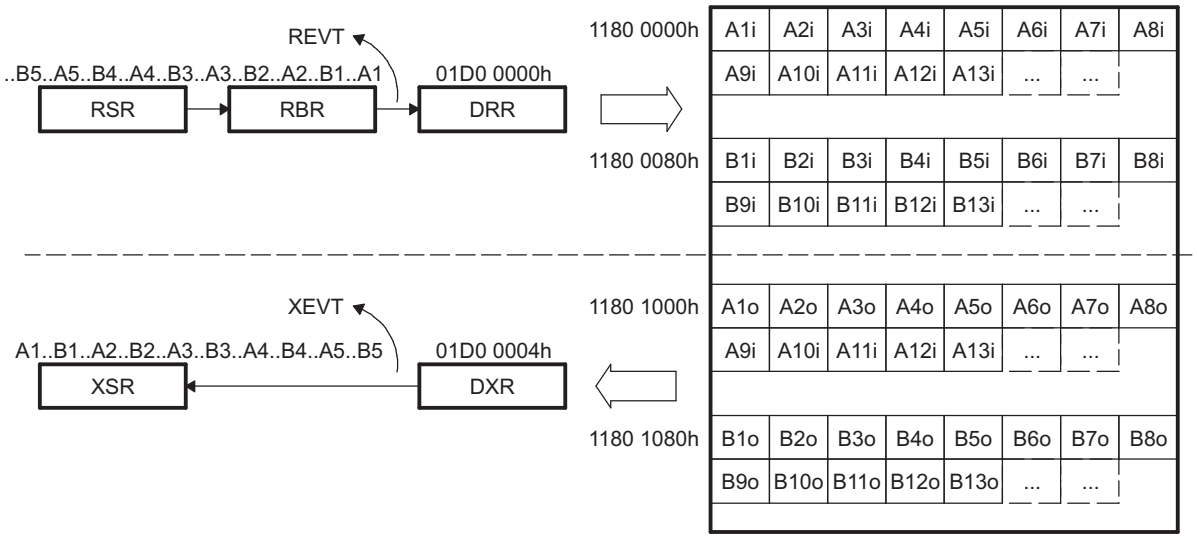
DMA channel 3 services the incoming data stream of the McBSP. The source address is set to that of the data receiver register (DRR), and the destination address is set to the first element of the data block. Since there are two data channels being serviced, A and B, they are to be located separately within the L2 SRAM.

In order to facilitate continuous operation, a copy of the PaRAM set for the channel is placed in PaRAM set 64. The LINK option is set and the link address is provided in the PaRAM set. Upon exhausting the channel 3 parameter set, the parameters located at the link address are loaded into the channel 3 parameter set and operation continues. This function continues throughout device operation until halted by the CPU.

16.3.4.3.2 Transmit Channel

DMA channel 2 services the outgoing data stream of the McBSP. In this case the destination address needs no update, hence, the parameter set changes accordingly. Linking is also used to allow continuous operation by the DMA channel, with duplicate PaRAM set entries at PaRAM set 65.

Figure 16-25. Servicing Continuous McBSP Data Example



Copyright © 2013–2016, Texas Instruments Incorporated

Figure 16-26. Servicing Continuous McBSP Data Example PaRAM

(a) EDMA Parameters for Receive Channel (PaRAM Set 3) being Linked to PaRAM Set 64

Parameter Contents	Parameter
0010 0000h	Channel Options Parameter (OPT)
01D0 0000h	Channel Source Address (SRC)
0080h | 0001h	Count for 2nd Dimension (BCNT) | Count for 1st Dimension (ACNT)
1180 0000h	Channel Destination Address (DST)
0001h | 0000h	Destination BCNT Index (DSTBIDX) | Source BCNT Index (SRCBIDX)
0080h | 4800h	BCNT Reload (BCNTRLD) | Link Address (LINK)
0000h | 0000h	Destination CCNT Index (DSTCIDX) | Source CCNT Index (SRCCIDX)
0000h | 0001h	Reserved | Count for 3rd Dimension (CCNT)

(b) Channel Options Parameter (OPT) Content for Receive Channel (PaRAM Set 3)

31	30 28	27 24	23	22	21	20	19 18	17 16
0	000	0000	0	0	0	1	00	00
PRIV	Reserved	PRIVID	ITCCHEN	TCCHEN	ITCINTEN	TCINTEN	Reserved	TCC

15 12	11	10 8	7 4	3	2	1	0
0000	0	000	0000	0	0	0	0
TCC	TCCMOD	FWID	Reserved	STATIC	SYNCDIM	DAM	SAM

(c) EDMA Parameters for Transmit Channel (PaRAM Set 2) being Linked to PaRAM Set 65

Parameter Contents	Parameter
0010 1000h	Channel Options Parameter (OPT)
1180 1000h	Channel Source Address (SRC)
0080h | 0001h	Count for 2nd Dimension (BCNT) | Count for 1st Dimension (ACNT)
01D0 0004h	Channel Destination Address (DST)
0000h | 0001h	Destination BCNT Index (DSTBIDX) | Source BCNT Index (SRCBIDX)
0080h | 4820h	BCNT Reload (BCNTRLD) | Link Address (LINK)
0000h | 0000h	Destination CCNT Index (DSTCIDX) | Source CCNT Index (SRCCIDX)
0000h | 0001h	Reserved | Count for 3rd Dimension (CCNT)

(d) Channel Options Parameter (OPT) Content for Transmit Channel (PaRAM Set 2)

31	30 28	27 24	23	22	21	20	19 18	17 16
0	000	0000	0	0	0	1	00	00
PRIV	Reserved	PRIVID	ITCCHEN	TCCHEN	ITCINTEN	TCINTEN	Reserved	TCC

15 12	11	10 8	7 4	3	2	1	0
0001	0	000	0000	0	0	0	0
TCC	TCCMOD	FWID	Reserved	STATIC	SYNCDIM	DAM	SAM

Figure 16-27. Servicing Continuous McBSP Data Example Reload PaRAM

(a) EDMA Reload Parameters (PaRAM Set 64) for Receive Channel

Copyright © 2013–2016, Texas Instruments Incorporated

Parameter Contents	
0010 0000h	
01D0 0000h	
0080h	0001h
1180 0000h	
0001h	0000h
0080h	4800h
0000h	0000h
0000h	0001h

Parameter	
Channel Options Parameter (OPT)	
Channel Source Address (SRC)	
Count for 2nd Dimension (BCNT)	Count for 1st Dimension (ACNT)
Channel Destination Address (DST)	
Destination BCNT Index (DSTBIDX)	Source BCNT Index (SRCBIDX)
BCNT Reload (BCNTRLD)	Link Address (LINK)
Destination CCNT Index (DSTCIDX)	Source CCNT Index (SRCCIDX)
Reserved	Count for 3rd Dimension (CCNT)

(b) Channel Options Parameter (OPT) Content for Receive Channel (PaRAM Set 64)

31	30 28	27 24	23	22	21	20	19 18	17 16
0	000	0000	0	0	0	1	00	00
PRIV	Reserved	PRIVID	ITCCHEN	TCCHEN	ITCINTEN	TCINTEN	Reserved	TCC

15 12	11	10 8	7 4	3	2	1	0
0000	0	000	0000	0	0	0	0
TCC	TCCMOD	FWID	Reserved	STATIC	SYNCDIM	DAM	SAM

(c) EDMA Reload Parameters (PaRAM Set 65) for Transmit Channel

Parameter Contents	
0010 1000h	
1180 1000h	
0080h	0001h
01D0 0004h	
0000h	0001h
0080h	4820h
0000h	0000h
0000h	0001h

Parameter	
Channel Options Parameter (OPT)	
Channel Source Address (SRC)	
Count for 2nd Dimension (BCNT)	Count for 1st Dimension (ACNT)
Channel Destination Address (DST)	
Destination BCNT Index (DSTBIDX)	Source BCNT Index (SRCBIDX)
BCNT Reload (BCNTRLD)	Link Address (LINK)
Destination CCNT Index (DSTCIDX)	Source CCNT Index (SRCCIDX)
Reserved	Count for 3rd Dimension (CCNT)

(d) Channel Options Parameter (OPT) Content for Transmit Channel (PaRAM Set 65)

31	30 28	27 24	23	22	21	20	19 18	17 16
0	000	0000	0	0	0	1	00	00
PRIV	Reserved	PRIVID	ITCCHEN	TCCHEN	ITCINTEN	TCINTEN	Reserved	TCC

15 12	11	10 8	7 4	3	2	1	0
0001	0	000	0000	0	0	0	0
TCC	TCCMOD	FWID	Reserved	STATIC	SYNCDIM	DAM	SAM

Copyright © 2013–2016, Texas Instruments Incorporated

16.3.4.4 Ping-Pong Buffering

Although the previous configuration allows the EDMA3 to service a peripheral continuously, it presents a number of restrictions to the CPU. Since the input and output buffers are continuously being filled/emptied, the CPU must match the pace of the EDMA3 very closely in order to process the data. The EDMA3 receive data must always be placed in memory before the CPU accesses it, and the CPU must provide the output data before the EDMA3 transfers it. Though not impossible, this is an unnecessary challenge. It is particularly difficult in a 2-level cache scheme.

Ping-pong buffering is a simple technique that allows the CPU activity to be distanced from the EDMA3 activity. This means that there are multiple (usually two) sets of data buffers for all incoming and outgoing data streams. While the EDMA3 transfers the data into and out of the ping buffers, the CPU manipulates the data in the pong buffers. When both CPU and EDMA3 activity completes, they switch. The EDMA3 then writes over the old input data and transfers the new output data. Figure 16-28 shows the ping-pong scheme for this example.

To change the continuous operation example, such that a ping-pong buffering scheme is used, the DMA channels need only a moderate change. Instead of one link parameter set, there are two; one for transferring data to/from the ping buffers and one for transferring data to/from the pong buffers. As soon as one transfer completes, the channel loads the PaRAM set for the other and the data transfers continue. Figure 16-29 shows the DMA channel configuration required.

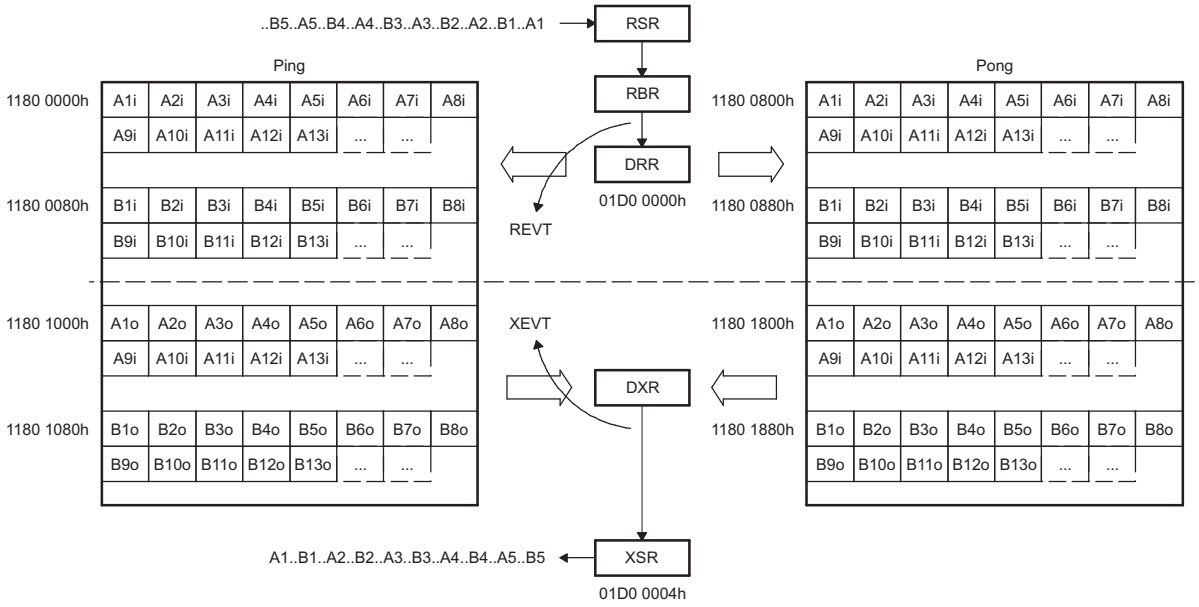
Each channel has two link parameter sets, ping and pong. The DMA channel is initially loaded with the ping parameters (Figure 16-29). The link address for the ping set is set to the PaRAM offset of the pong parameter set (Figure 16-30). The link address for the pong set is set to the PaRAM offset of the ping parameter set (Figure 16-31). The channel options, count values, and index values are all identical between the ping and pong parameters for each channel. The only differences are the link address provided and the address of the data buffer.

Copyright © 2013–2016, Texas Instruments Incorporated

16.3.4.4.1 Synchronization with the CPU

In order to utilize the ping-pong buffering technique, the system must signal the CPU when to begin to access the new data set. After the CPU finishes processing an input buffer (ping), it waits for the EDMA3 to complete before switching to the alternate (pong) buffer. In this example, both channels provide their channel numbers as their report word and set the TCINTEN bit to 1 to generate an interrupt after completion. When channel 3 fills an input buffer, the E3 bit in the interrupt pending register (IPR) is set to 1; when channel 2 empties an output buffer, the E2 bit in IPR is set to 1. The CPU must manually clear these bits. With the channel parameters set, the CPU polls IPR to determine when to switch. The EDMA3 and CPU could alternatively be configured such that the channel completion interrupts the CPU. By doing this, the CPU could service a background task while waiting for the EDMA3 to complete.

Figure 16-28. Ping-Pong Buffering for McBSP Data Example



Copyright © 2013–2016, Texas Instruments Incorporated

Figure 16-29. Ping-Pong Buffering for McBSP Example PaRAM

(a) EDMA Parameters for Channel 3 (Using PaRAM Set 3 Linked to Pong Set 64)

Parameter Contents	Parameter
0010 3000h	Channel Options Parameter (OPT)
01D0 0000h	Channel Source Address (SRC)
0080h / 0001h	Count for 2nd Dimension (BCNT) / Count for 1st Dimension (ACNT)
1180 0000h	Channel Destination Address (DST)
0001h / 0000h	Destination BCNT Index (DSTBIDX) / Source BCNT Index (SRCBIDX)
0080h / 4800h	BCNT Reload (BCNTRLD) / Link Address (LINK)
0000h / 0000h	Destination CCNT Index (DSTCIDX) / Source CCNT Index (SRCCIDX)
0000h / 0001h	Reserved / Count for 3rd Dimension (CCNT)

(b) Channel Options Parameter (OPT) Content for Channel 3

31	30–28	27–24	23	22	21	20	19–18	17–16
0	000	0000	0	0	0	1	00	00
PRIV	Reserved	PRIVID	ITCCHEN	TCCHEN	ITCINTEN	TCINTEN	Reserved	TCC

15–12	11	10–8	7–4	3	2	1	0
0011	0	000	0000	0	0	0	0
TCC	TCCMOD	FWID	Reserved	STATIC	SYNCDIM	DAM	SAM

(c) EDMA Parameters for Channel 2 (Using PaRAM Set 2 Linked to Pong Set 65)

Parameter Contents	Parameter
0010 2000h	Channel Options Parameter (OPT)
1180 1000h	Channel Source Address (SRC)
0080h / 0001h	Count for 2nd Dimension (BCNT) / Count for 1st Dimension (ACNT)
01D0 0004h	Channel Destination Address (DST)
0000h / 0001h	Destination BCNT Index (DSTBIDX) / Source BCNT Index (SRCBIDX)
0080h / 4840h	BCNT Reload (BCNTRLD) / Link Address (LINK)
0000h / 0000h	Destination CCNT Index (DSTCIDX) / Source CCNT Index (SRCCIDX)
0000h / 0001h	Reserved / Count for 3rd Dimension (CCNT)

(d) Channel Options Parameter (OPT) Content for Channel 2

31	30–28	27–24	23	22	21	20	19–18	17–16
0	000	0000	0	0	0	1	00	00
PRIV	Reserved	PRIVID	ITCCHEN	TCCHEN	ITCINTEN	TCINTEN	Reserved	TCC

15–12	11	10–8	7–4	3	2	1	0
0010	0	000	0000	0	0	0	0
TCC	TCCMOD	FWID	Reserved	STATIC	SYNCDIM	DAM	SAM

Figure 16-30. Ping-Pong Buffering for McBSP Example Pong PaRAM

(a) EDMA Pong Parameters for Channel 3 at Set 64 Linked to Set 65

Copyright © 2013–2016, Texas Instruments Incorporated

Parameter Contents	
0010 D000h	
01D0 0000h	
0080h	0001h
1180 0800h	
0001h	0000h
0080h	4820h
0000h	0000h
0000h	0001h

Parameter	
Channel Options Parameter (OPT)	
Channel Source Address (SRC)	
Count for 2nd Dimension (BCNT)	Count for 1st Dimension (ACNT)
Channel Destination Address (DST)	
Destination BCNT Index (DSTBIDX)	Source BCNT Index (SRCBIDX)
BCNT Reload (BCNTRLD)	Link Address (LINK)
Destination CCNT Index (DSTCIDX)	Source CCNT Index (SRCCIDX)
Reserved	Count for 3rd Dimension (CCNT)

(b) EDMA Pong Parameters for Channel 2 at Set 66 Linked to Set 67

Parameter Contents	
0010 C000h	
1180 1800h	
0080h	0001h
01D0 0004h	
0000h	0001h
0080h	4860h
0000h	0000h
0000h	0001h

Parameter	
Channel Options Parameter (OPT)	
Channel Source Address (SRC)	
Count for 2nd Dimension (BCNT)	Count for 1st Dimension (ACNT)
Channel Destination Address (DST)	
Destination BCNT Index (DSTBIDX)	Source BCNT Index (SRCBIDX)
BCNT Reload (BCNTRLD)	Link Address (LINK)
Destination CCNT Index (DSTCIDX)	Source CCNT Index (SRCCIDX)
Reserved	Count for 3rd Dimension (CCNT)

Figure 16-31. Ping-Pong Buffering for McBSP Example Ping PaRAM

(a) EDMA Ping Parameters for Channel 3 at Set 65 Linked to Set 64

Parameter Contents	
0010 D000h	
01D0 0000h	
0080h	0001h
1180 0000h	
0001h	0000h
0080h	4800h
0000h	0000h
0000h	0001h

Parameter	
Channel Options Parameter (OPT)	
Channel Source Address (SRC)	
Count for 2nd Dimension (BCNT)	Count for 1st Dimension (ACNT)
Channel Destination Address (DST)	
Destination BCNT Index (DSTBIDX)	Source BCNT Index (SRCBIDX)
BCNT Reload (BCNTRLD)	Link Address (LINK)
Destination CCNT Index (DSTCIDX)	Source CCNT Index (SRCCIDX)
Reserved	Count for 3rd Dimension (CCNT)

(b) EDMA Ping Parameters for Channel 2 at Set 67 Linked to Set 66

Parameter Contents	
0010 C000h	
1180 1000h	
0080h	0001h
01D0 0004h	
0000h	0001h
0080h	4840h
0000h	0000h
0000h	0001h

Parameter	
Channel Options Parameter (OPT)	
Channel Source Address (SRC)	
Count for 2nd Dimension (BCNT)	Count for 1st Dimension (ACNT)
Channel Destination Address (DST)	
Destination BCNT Index (DSTBIDX)	Source BCNT Index (SRCBIDX)
BCNT Reload (BCNTRLD)	Link Address (LINK)
Destination CCNT Index (DSTCIDX)	Source CCNT Index (SRCCIDX)
Reserved	Count for 3rd Dimension (CCNT)

Copyright © 2013–2016, Texas Instruments Incorporated

16.3.4.5 Transfer Chaining Examples

The following examples explain the intermediate transfer complete chaining function.

16.3.4.5.1 Servicing Input/Output FIFOs with a Single Event

Many systems require the use of a pair of external FIFOs that must be serviced at the same rate. One FIFO buffers data input, and the other buffers data output. The EDMA3 channels that service these FIFOs can be set up for AB-synchronized transfers. While each FIFO is serviced with a different set of parameters, both can be signaled from a single event. For example, an external interrupt pin can be tied to the status flags of one of the FIFOs. When this event arrives, the EDMA3 needs to perform servicing for both the input and output streams. Without the intermediate transfer complete chaining feature this would require two events, and thus two external interrupt pins. The intermediate transfer complete chaining feature allows the use of a single external event (for example, a GPIO event). Figure 16-32 shows the EDMA3 setup and illustration for this example.

A GPIO event (in this case, GPINT0) triggers an array transfer. Upon completion of each intermediate array transfer of channel 16, intermediate transfer complete chaining sets the E31 bit (specified by TCC of 31) in the chained event register (CER) and provides a synchronization event to channel 31. Upon completion of the last array transfer of channel 16, transfer complete chaining—not intermediate transfer complete chaining—sets the E31 bit in CER (specified by TCCMODE:TCC) and provides a synchronization event to channel 31. The completion of channel 31 sets the I31 bit (specified by TCCMODE:TCC) in the interrupt pending register (IPR), which can generate an interrupt to the CPU, if the I31 bit in the interrupt enable register (IER) is set to 1.

16.3.4.5.2 Breaking Up Large Transfers with Intermediate Chaining

Another feature of intermediate transfer chaining (ITCCHEN) is for breaking up large transfers. A large transfer may lock out other transfers of the same priority level for the duration of the transfer. For example, a large transfer on queue 0 from the internal memory to the external memory using the EMIF may starve other EDMA3 transfers on the same queue. In addition, this large high-priority transfer may prevent the EMIF for a long duration to service other lower priority transfers. When a large transfer is considered to be high priority, it should be split into multiple smaller transfers. Figure 16-33 shows the EDMA3 setup and illustration of an example single large block transfer.

The intermediate transfer chaining enable (ITCCHEN) provides a method to break up a large transfer into smaller transfers. For example, to move a single large block of memory (16K bytes), the EDMA3 performs an A-synchronized transfer. The element count is set to a reasonable value, where reasonable derives from the amount of time it would take to move this smaller amount of data. Assume 1K byte is a reasonable small transfer in this example. The EDMA3 is set up to transfer 16 arrays of 1K byte elements, for a total of 16K byte elements. The TCC field in the channel options parameter (OPT) is set to the same value as the channel number and ITCCHEN are set. In this example, DMA channel 25 is used and TCC is also set to 25. The TCINTEN may also be set to trigger interrupt 25 when the last 1K byte array is transferred. The CPU starts the EDMA3 transfer by writing to the appropriate bit of the event set register (ESR.E25). The EDMA3 transfers the first 1K byte array. Upon completion of the first array, intermediate transfer complete code chaining generates a synchronization event to channel 25, a value specified by the TCC field. This intermediate transfer completion chaining event causes DMA channel 25 to transfer the next 1K byte array. This process continues until the transfer parameters are exhausted, at which point the EDMA3 has completed the 16K byte transfer. This method breaks up a large transfer into smaller packets, thus providing natural time slices in the transfer such that other events may be processed. Figure 16-34 shows the EDMA3 setup and illustration of the broken up smaller packet transfers.

Copyright © 2013–2016, Texas Instruments Incorporated

Figure 16-32. Intermediate Transfer Completion Chaining Example

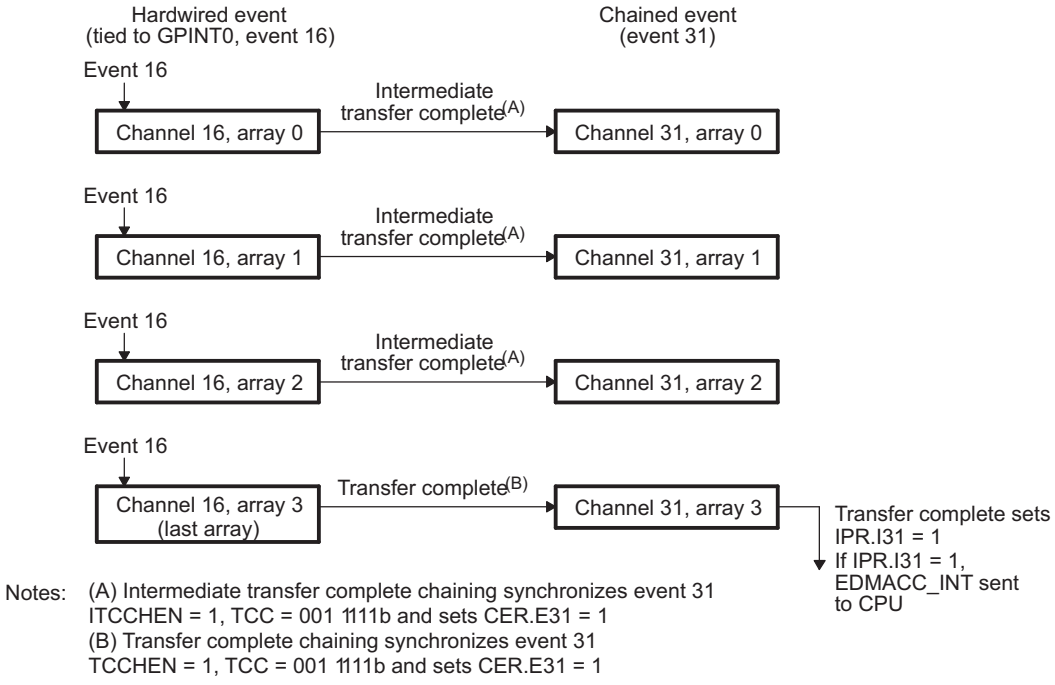
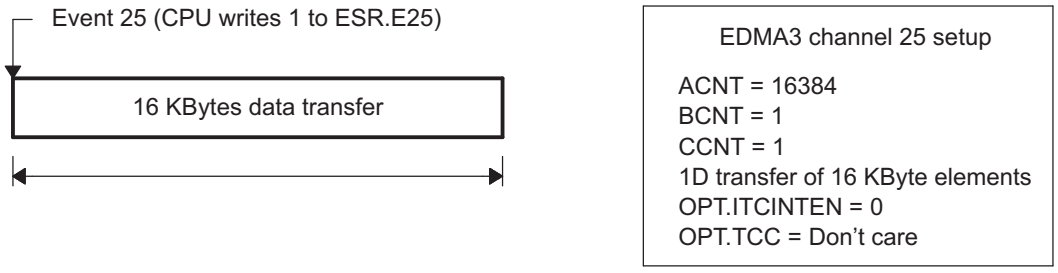
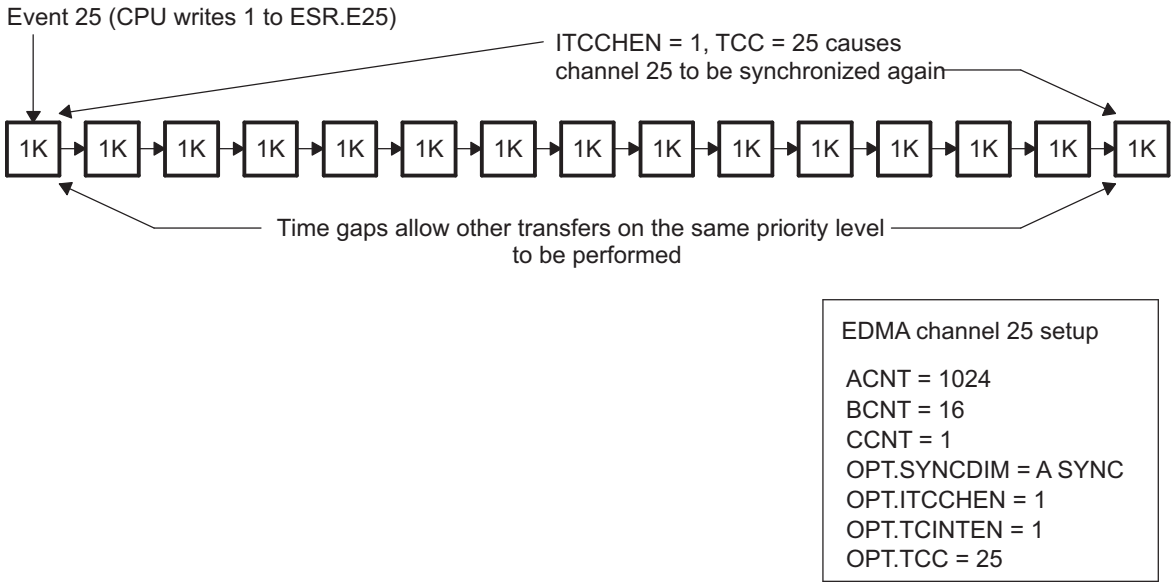


Figure 16-33. Single Large Block Transfer Example



Copyright © 2013–2016, Texas Instruments Incorporated

Figure 16-34. Smaller Packet Data Transfers Example



16.4 Registers

This section discusses the registers of the EDMA3 controller.

16.4.1 Parameter RAM (PaRAM) Entries

Table 16-11 lists the parameter RAM (PaRAM) entries for the EDMA3 channel controller (EDMA3CC). See your device-specific data manual for the memory address of these registers.

Table 16-11. EDMA3 Channel Controller (EDMA3CC) Parameter RAM (PaRAM) Entries

Offset	Acronym	Parameter	Section
0h	OPT	Channel Options	Section 16.4.1.1
4h	SRC	Channel Source Address	Section 16.4.1.2
8h	A_B_CNT	A Count/B Count	Section 16.4.1.3
Ch	DST	Channel Destination Address	Section 16.4.1.4
10h	SRC_DST_BIDX	Source B Index/Destination B Index	Section 16.4.1.5
14h	LINK_BCNTRLD	Link Address/B Count Reload	Section 16.4.1.6
18h	SRC_DST_CIDX	Source C Index/Destination C Index	Section 16.4.1.7
1Ch	CCNT	C Count	Section 16.4.1.8

Copyright © 2013–2016, Texas Instruments Incorporated

16.4.1.1 Channel Options Parameter (OPT)

The channel options parameter (OPT) is shown in Figure 16-35 and described in Table 16-12.

NOTE: The TCC field in OPT is a 6-bit field and can be programmed for any value between 0-64. For devices with 32 DMA channels, the TCC field should have a value between 0 to 31 so that it sets the appropriate bits (0 to 31) in the interrupt pending register (IPR) (and can interrupt the CPU(s) on enabling the interrupt enable register (IER) bits (0-31)).

Figure 16-35. Channel Options Parameter (OPT)

31–28	27–24	23	22	21	20	19–18	17–16
Reserved	PRIVID	ITCCHEN	TCCHEN	ITCINTEN	TCINTEN	Reserved	TCC
R-0	R-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0

15–12	11–10	9–8	7–4	3	2	1	0
TCC	TCCMOD	FWID	Reserved	STATIC	SYNCDIM	DAM	SAM
R/W-0	R/W-0	R/W-0	R-0	R/W-0	R/W-0	R/W-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 16-12. Channel Options Parameters (OPT) Field Descriptions

Bit	Field	Value	Description
31-28	Reserved	0	Reserved
27-24	PRIVID	0-Fh	Privilege identification for the external host/CPU/DMA that programmed this PaRAM set. This value is set with the EDMA3 master's privilege identification value when any part of the PaRAM set is written.
23	ITCCHEN		Intermediate transfer completion chaining enable.
		0	Intermediate transfer complete chaining is disabled.
		1	Intermediate transfer complete chaining is enabled.
			When enabled, the chained event register (CER) bit is set on every intermediate chained transfer completion (upon completion of every intermediate TR in the PaRAM set, except the final TR in the PaRAM set). The bit (position) set in CER is the TCC value specified.
22	TCCHEN		Transfer complete chaining enable.
		0	Transfer complete chaining is disabled.
		1	Transfer complete chaining is enabled.
			When enabled, the chained event register (CER) bit is set on final chained transfer completion (upon completion of the final TR in the PaRAM set). The bit (position) set in CER is the TCC value specified.
21	ITCINTEN		Intermediate transfer completion interrupt enable.
		0	Intermediate transfer complete interrupt is disabled.
		1	Intermediate transfer complete interrupt is enabled.
			When enabled, the interrupt pending register (IPR) bit is set on every intermediate transfer completion (upon completion of every intermediate TR in the PaRAM set, except the final TR in the PaRAM set). The bit (position) set in IPR is the TCC value specified. In order to generate a completion interrupt to the CPU, the corresponding IER[TCC] bit must be set to 1.
20	TCINTEN		Transfer complete interrupt enable.
		0	Transfer complete interrupt is disabled.
		1	Transfer complete interrupt is enabled.
			When enabled, the interrupt pending register (IPR) bit is set on transfer completion (upon completion of the final TR in the PaRAM set). The bit (position) set in IPR is the TCC value specified. In order to generate a completion interrupt to the CPU, the corresponding IER[TCC] bit must be set to 1.
19	Reserved	0	Reserved. Always write 0 to this bit.
18	Reserved	0	Reserved

Copyright © 2013–2016, Texas Instruments Incorporated

Table 16-12. Channel Options Parameters (OPT) Field Descriptions (continued)

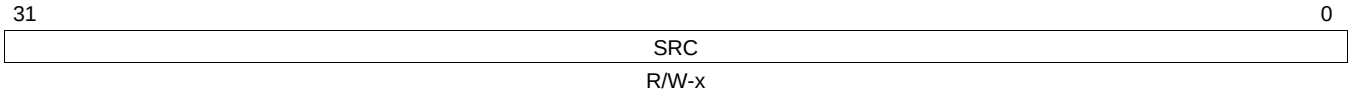
Bit	Field	Value	Description
17-12	TCC	0-3Fh	Transfer complete code. This 6-bit code is used to set the relevant bit in chaining enable register (CER[TCC]) for chaining or in interrupt pending register (IPR[TCC]) for interrupts.
		0-1Fh	Valid values
		20h-3Fh	Reserved
11	TCCMODE		Transfer complete code mode. Indicates the point at which a transfer is considered completed for chaining and interrupt generation.
		0	Normal completion: A transfer is considered completed after the data has been transferred.
		1	Early completion: A transfer is considered completed after the EDMA3CC submits a TR to the EDMA3TC. TC may still be transferring data when interrupt/chain is triggered.
10-8	FWID	0-7h	FIFO Width. Applies if either SAM or DAM is set to constant addressing mode.
		0	FIFO width is 8-bit.
		1h	FIFO width is 16-bit.
		2h	FIFO width is 32-bit.
		3h	FIFO width is 64-bit.
		4h	FIFO width is 128-bit.
		5h	FIFO width is 256-bit.
		6h-7h	Reserved
7-4	Reserved	0	Reserved
3	STATIC		Static PaRAM set.
		0	PaRAM set is not static. PaRAM set is updated or linked after TR is submitted. A value of 0 should be used for DMA channels and for nonfinal transfers in a linked list of QDMA transfers.
		1	PaRAM set is static. PaRAM set is not updated or linked after TR is submitted. A value of 1 should be used for isolated QDMA transfers or for the final transfer in a linked list of QDMA transfers.
2	SYNCDIM		Transfer synchronization dimension.
		0	A-synchronized. Each event triggers the transfer of a single array of ACNT bytes.
		1	AB-synchronized. Each event triggers the transfer of BCNT arrays of ACNT bytes.
1	DAM		Destination address mode.
		0	Increment (INCR) mode. Destination addressing within an array increments. Destination is not a FIFO.
		1	Constant addressing (CONST) mode. Destination addressing within an array wraps around upon reaching FIFO width. Note: The constant addressing (CONST) mode has limited applicability. The EDMA3 should be configured for the constant addressing mode (SAM/DAM = 1) only if the transfer source or destination (on-chip memory, off-chip memory controllers, slave peripherals) support the constant addressing mode. On the C674x/OMAP-L1x processors, no peripherals, memory, or memory controller support constant addressing mode. If the constant addressing mode is not supported, the similar logical transfer can be achieved using the increment (INCR) mode (SAM/DAM = 0) by appropriately programming the count and indices values.
0	SAM		Source address mode.
		0	Increment (INCR) mode. Source addressing within an array increments. Source is not a FIFO.
		1	Constant addressing (CONST) mode. Source addressing within an array wraps around upon reaching FIFO width. Note: The constant addressing (CONST) mode has limited applicability. The EDMA3 should be configured for the constant addressing mode (SAM/DAM = 1) only if the transfer source or destination (on-chip memory, off-chip memory controllers, slave peripherals) support the constant addressing mode. On the C674x/OMAP-L1x processors, no peripherals, memory, or memory controller support constant addressing mode. If the constant addressing mode is not supported, the similar logical transfer can be achieved using the increment (INCR) mode (SAM/DAM = 0) by appropriately programming the count and indices values.

Copyright © 2013–2016, Texas Instruments Incorporated

16.4.1.2 Channel Source Address Parameter (SRC)

The channel source address parameter (SRC) specifies the starting byte address of the source. The SRC is shown in Figure 16-36 and described in Table 16-13.

Figure 16-36. Channel Source Address Parameter (SRC)



LEGEND: R = Read only; -*n* = value after reset

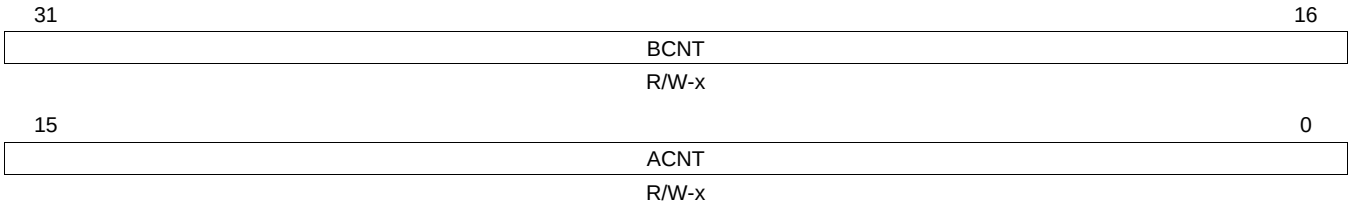
Table 16-13. Channel Source Address Parameter (SRC) Field Descriptions

Bit	Field	Value	Description
31-0	SRC	0-FFFF FFFFh	Source address. Specifies the starting byte address of the source.

16.4.1.3 A Count/B Count Parameter (A_B_CNT)

The A count/B count parameter (A_B_CNT) specifies the number of bytes within the 1st dimension of a transfer and the number of arrays of length ACNT. The A_B_CNT is shown in Figure 16-37 and described in Table 16-14.

Figure 16-37. A Count/B Count Parameter (A_B_CNT)



LEGEND: R/W = Read/Write; -*n* = value after reset; -x = value is indeterminate after reset

Table 16-14. A Count/B Count Parameter (A_B_CNT) Field Descriptions

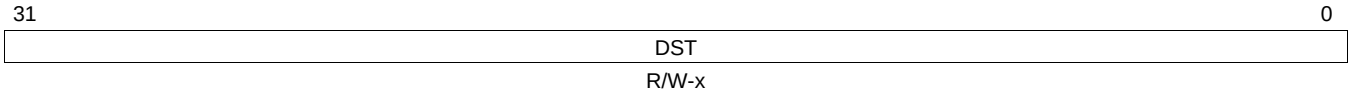
Bit	Field	Value	Description
31-16	BCNT	0-FFFFh	B count. Unsigned value specifying the number of arrays in a frame, where an array is ACNT bytes. Valid values range from 1 to 65 535.
15-0	ACNT	0-FFFFh	A count for 1st Dimension. Unsigned value specifying the number of contiguous bytes within an array (first dimension of the transfer). Valid values range from 1 to 65 535.

Copyright © 2013–2016, Texas Instruments Incorporated

16.4.1.4 Channel Destination Address Parameter (DST)

The channel destination address parameter (DST) specifies the starting byte address of the source. The DST is shown in Figure 16-38 and described in Table 16-15.

Figure 16-38. Channel Destination Address Parameter (DST)



LEGEND: R = Read only; -*n* = value after reset

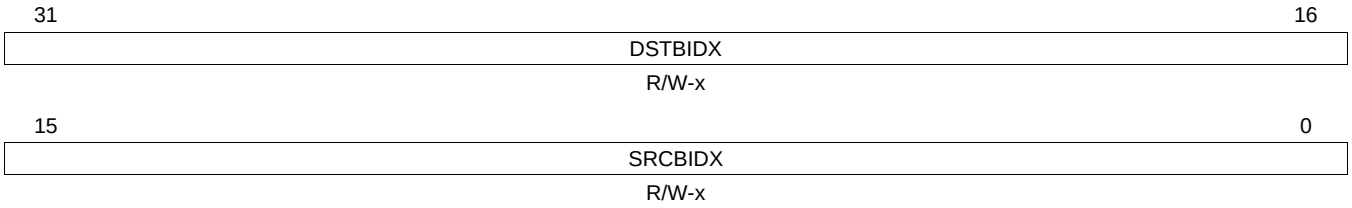
Table 16-15. Channel Destination Address Parameter (DST) Field Descriptions

Bit	Field	Value	Description
31-0	DST	0-FFFF FFFFh	Destination address. Specifies the starting byte address of the destination where data is transferred.

16.4.1.5 Source B Index/Destination B Index Parameter (SRC_DST_BIDX)

The source B index/destination B index parameter (SRC_DST_BIDX) specifies the value (2s complement) used for source address modification between each array in the 2nd dimension and the value (2s complement) used for destination address modification between each array in the 2nd dimension. The SRC_DST_BIDX is shown in Figure 16-39 and described in Table 16-16.

Figure 16-39. Source B Index/Destination B Index Parameter (SRC_DST_BIDX)



LEGEND: R/W = Read/Write; -*n* = value after reset; -x = value is indeterminate after reset

Table 16-16. Source B Index/Destination B Index Parameter (SRC_DST_BIDX) Field Descriptions

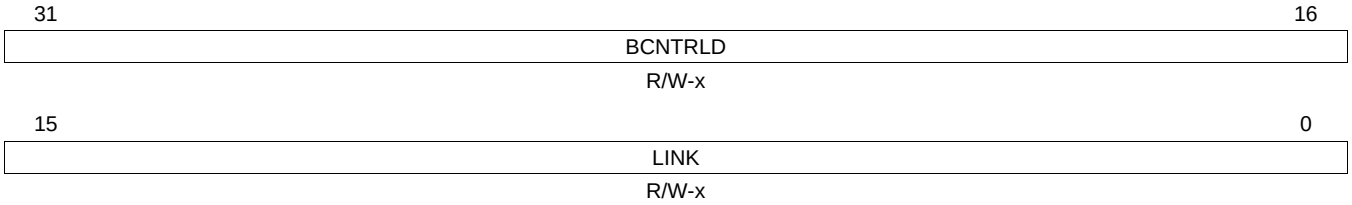
Bit	Field	Value	Description
31-16	DSTBIDX	0-FFFFh	Destination B index. Signed value specifying the byte address offset between destination arrays within a frame (2nd dimension). Valid values range from –32 768 and 32 767.
15-0	SRCBIDX	0-FFFFh	Source B index. Signed value specifying the byte address offset between source arrays within a frame (2nd dimension). Valid values range from –32 768 and 32 767.

Copyright © 2013–2016, Texas Instruments Incorporated

16.4.1.6 Link Address/B Count Reload Parameter (LINK_BCNTRLD)

The link address/B count reload parameter (LINK_BCNTRLD) specifies the byte address offset in the PaRAM from which the EDMA3CC loads/reloads the next PaRAM set during linking and the value used to reload the BCNT field in the A count/B count parameter (A_B_CNT) once the last array in the 2nd dimension is transferred. The LINK_BCNTRLD is shown in Figure 16-40 and described in Table 16-17.

Figure 16-40. Link Address/B Count Reload Parameter (LINK_BCNTRLD)



LEGEND: R/W = Read/Write; -*n* = value after reset; -x = value is indeterminate after reset

Table 16-17. Link Address/B Count Reload Parameter (LINK_BCNTRLD) Field Descriptions

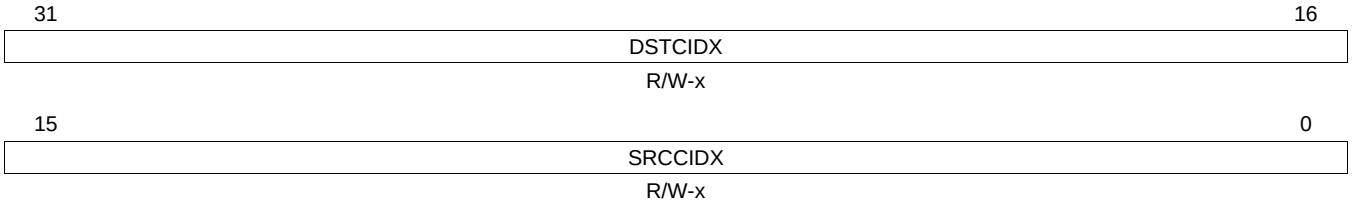
Bit	Field	Value	Description
31-16	BCNTRLD	0-FFFFh	B count reload. The count value used to reload BCNT in the A count/B count parameter (A_B_CNT) when BCNT decrements to 0 (TR submitted for the last array in 2nd dimension). Only relevant in A-synchronized transfers.
15-0	LINK	0-FFFFh	Link address. The PaRAM address containing the PaRAM set to be linked (copied from) when the current PaRAM set is exhausted. You must program the link address to point to a valid aligned 32-byte PaRAM set. The 5 LSBs of the LINK field should be cleared to 0. A value of FFFFh specifies a null link.

Copyright © 2013–2016, Texas Instruments Incorporated

16.4.1.7 Source C Index/Destination C Index Parameter (SRC_DST_CIDX)

The source C index/destination C index parameter (SRC_DST_CIDX) specifies the value (2s complement) used for source address modification between each array in the 3rd dimension and the value (2s complement) used for destination address modification between each array in the 3rd dimension. The SRC_DST_CIDX is shown in Figure 16-41 and described in Table 16-18.

Figure 16-41. Source C Index/Destination C Index Parameter (SRC_DST_CIDX)



LEGEND: R/W = Read/Write; -*n* = value after reset; -x = value is indeterminate after reset

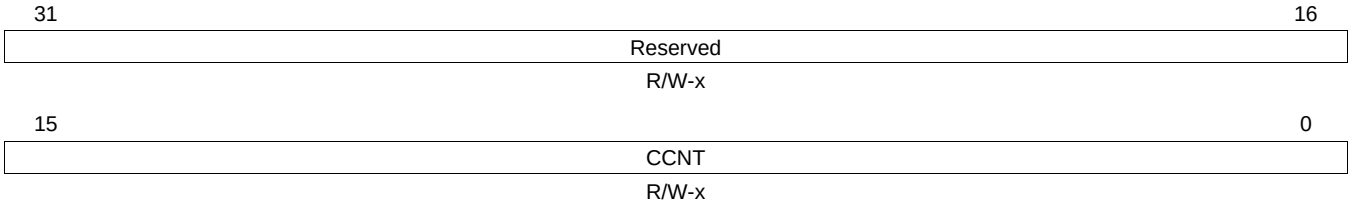
Table 16-18. Source C Index/Destination C Index Parameter (SRC_DST_CIDX) Field Descriptions

Bit	Field	Value	Description
31-16	DSTCIDX	0-FFFFh	Destination C index. Signed value specifying the byte address offset between frames within a block (3rd dimension). Valid values range from –32 768 and 32 767.
15-0	SRCCIDX	0-FFFFh	Source C index. Signed value specifying the byte address offset between frames within a block (3rd dimension). Valid values range from –32 768 and 32 767.

16.4.1.8 C Count Parameter (CCNT)

The C count parameter (CCNT) specifies the number of frames in a block. The CCNT is shown in Figure 16-42 and described in Table 16-19.

Figure 16-42. C Count Parameter (CCNT)



LEGEND: R/W = Read/Write; -*n* = value after reset; -x = value is indeterminate after reset

Table 16-19. C Count Parameter (CCNT) Field Descriptions

Bit	Field	Value	Description
31-16	Reserved	0	Reserved
15-0	CCNT	0-FFFFh	C counter. Unsigned value specifying the number of frames in a block, where a frame is BCNT arrays of ACNT bytes. Valid values range from 1 to 65 535.

Copyright © 2013–2016, Texas Instruments Incorporated

16.4.2 EDMA3 Channel Controller (EDMA3CC) Registers

Table 16-20 lists the memory-mapped registers for the EDMA3 channel controller (EDMA3CC). See your device-specific data manual for the memory address of these registers and for the shadow region addresses. All other register offset addresses not listed in Table 16-20 should be considered as reserved locations and the register contents should not be modified.

Table 16-20. EDMA3 Channel Controller (EDMA3CC) Registers

Offset	Acronym	Register Description	Section
0h	REVID	Revision Identification Register	Section 16.4.2.1.1
4h	CCCFG	EDMA3CC Configuration Register	Section 16.4.2.1.2
		Global Registers	
200h	QCHMAP0	QDMA Channel 0 Mapping Register	Section 16.4.2.1.3
204h	QCHMAP1	QDMA Channel 1 Mapping Register	Section 16.4.2.1.3
208h	QCHMAP2	QDMA Channel 2 Mapping Register	Section 16.4.2.1.3
20Ch	QCHMAP3	QDMA Channel 3 Mapping Register	Section 16.4.2.1.3
210h	QCHMAP4	QDMA Channel 4 Mapping Register	Section 16.4.2.1.3
214h	QCHMAP5	QDMA Channel 5 Mapping Register	Section 16.4.2.1.3
218h	QCHMAP6	QDMA Channel 6 Mapping Register	Section 16.4.2.1.3
21Ch	QCHMAP7	QDMA Channel 7 Mapping Register	Section 16.4.2.1.3
240h	DMAQNUM0	DMA Channel Queue Number Register 0	Section 16.4.2.1.4
244h	DMAQNUM1	DMA Channel Queue Number Register 1	Section 16.4.2.1.4
248h	DMAQNUM2	DMA Channel Queue Number Register 2	Section 16.4.2.1.4
24Ch	DMAQNUM3	DMA Channel Queue Number Register 3	Section 16.4.2.1.4
260h	QDMAQNUM	QDMA Channel Queue Number Register	Section 16.4.2.1.5
284h	QUEPRI	Queue Priority Register(1)	Section 16.4.2.1.6
300h	EMR	Event Missed Register	Section 16.4.2.2.1
308h	EMCR	Event Missed Clear Register	Section 16.4.2.2.2
310h	QEMR	QDMA Event Missed Register	Section 16.4.2.2.3
314h	QEMCR	QDMA Event Missed Clear Register	Section 16.4.2.2.4
318h	CCERR	EDMA3CC Error Register	Section 16.4.2.2.5
31Ch	CCERRCLR	EDMA3CC Error Clear Register	Section 16.4.2.2.6
320h	EEVAL	Error Evaluate Register	Section 16.4.2.2.7
340h	DRAE0	DMA Region Access Enable Register for Region 0	Section 16.4.2.3.1
348h	DRAE1	DMA Region Access Enable Register for Region 1	Section 16.4.2.3.1
350h	DRAE2	DMA Region Access Enable Register for Region 2	Section 16.4.2.3.1
358h	DRAE3	DMA Region Access Enable Register for Region 3	Section 16.4.2.3.1
380h	QRAE0	QDMA Region Access Enable Register for Region 0	Section 16.4.2.3.2
384h	QRAE1	QDMA Region Access Enable Register for Region 1	Section 16.4.2.3.2
388h	QRAE2	QDMA Region Access Enable Register for Region 2	Section 16.4.2.3.2
38Ch	QRAE3	QDMA Region Access Enable Register for Region 3	Section 16.4.2.3.2
400h-43Ch	Q0E0-Q0E15	Event Queue Entry Registers Q0E0-Q0E15	Section 16.4.2.4.1
440h-47Ch	Q1E0-Q1E15	Event Queue Entry Registers Q1E0-Q1E15	Section 16.4.2.4.1
600h	QSTAT0	Queue 0 Status Register	Section 16.4.2.4.2
604h	QSTAT1	Queue 1 Status Register	Section 16.4.2.4.2
620h	QWMTHRA	Queue Watermark Threshold A Register	Section 16.4.2.4.3
640h	CCSTAT	EDMA3CC Status Register	Section 16.4.2.4.4

(1) On previous architectures, the EDMA3TC priority was controlled by the queue priority register (QUEPRI) in the EDMA3CC memory-map. However for this device, the priority control for the transfer controllers is controlled by the chip-level registers in the System Configuration Module. You should use the chip-level registers and not QUEPRI to configure the TC priority.

Copyright © 2013–2016, Texas Instruments Incorporated

Table 16-20. EDMA3 Channel Controller (EDMA3CC) Registers (continued)

Offset	Acronym	Register Description	Section
		Global Channel Registers	
1000h	ER	Event Register	Section 16.4.2.5.1
1008h	ECR	Event Clear Register	Section 16.4.2.5.2
1010h	ESR	Event Set Register	Section 16.4.2.5.3
1018h	CER	Chained Event Register	Section 16.4.2.5.4
1020h	EER	Event Enable Register	Section 16.4.2.5.5
1028h	EECR	Event Enable Clear Register	Section 16.4.2.5.6
1030h	EESR	Event Enable Set Register	Section 16.4.2.5.7
1038h	SER	Secondary Event Register	Section 16.4.2.5.8
1040h	SECR	Secondary Event Clear Register	Section 16.4.2.5.9
1050h	IER	Interrupt Enable Register	Section 16.4.2.6.1
1058h	IECR	Interrupt Enable Clear Register	Section 16.4.2.6.2
1060h	IESR	Interrupt Enable Set Register	Section 16.4.2.6.3
1068h	IPR	Interrupt Pending Register	Section 16.4.2.6.4
1070h	ICR	Interrupt Clear Register	Section 16.4.2.6.5
1078h	IEVAL	Interrupt Evaluate Register	Section 16.4.2.6.6
1080h	QER	QDMA Event Register	Section 16.4.2.7.1
1084h	QEER	QDMA Event Enable Register	Section 16.4.2.7.2
1088h	QEECR	QDMA Event Enable Clear Register	Section 16.4.2.7.3
108Ch	QEESR	QDMA Event Enable Set Register	Section 16.4.2.7.4
1090h	QSER	QDMA Secondary Event Register	Section 16.4.2.7.5
1094h	QSECR	QDMA Secondary Event Clear Register	Section 16.4.2.7.6
		Shadow Region 0 Channel Registers	
2000h	ER	Event Register	—
2008h	ECR	Event Clear Register	—
2010h	ESR	Event Set Register	—
2018h	CER	Chained Event Register	—
2020h	EER	Event Enable Register	—
2028h	EECR	Event Enable Clear Register	—
2030h	EESR	Event Enable Set Register	—
2038h	SER	Secondary Event Register	—
2040h	SECR	Secondary Event Clear Register	—
2050h	IER	Interrupt Enable Register	—
2058h	IECR	Interrupt Enable Clear Register	—
2060h	IESR	Interrupt Enable Set Register	—
2068h	IPR	Interrupt Pending Register	—
2070h	ICR	Interrupt Clear Register	—
2078h	IEVAL	Interrupt Evaluate Register	—
2080h	QER	QDMA Event Register	—
2084h	QEER	QDMA Event Enable Register	—
2088h	QEECR	QDMA Event Enable Clear Register	—
208Ch	QEESR	QDMA Event Enable Set Register	—
2090h	QSER	QDMA Secondary Event Register	—
2094h	QSECR	QDMA Secondary Event Clear Register	—

Copyright © 2013–2016, Texas Instruments Incorporated

Table 16-20. EDMA3 Channel Controller (EDMA3CC) Registers (continued)

Offset	Acronym	Register Description	Section
		Shadow Region 1 Channel Registers	
2200h	ER	Event Register	—
2208h	ECR	Event Clear Register	—
2210h	ESR	Event Set Register	—
2218h	CER	Chained Event Register	—
2220h	EER	Event Enable Register	—
2228h	EECR	Event Enable Clear Register	—
2230h	EESR	Event Enable Set Register	—
2238h	SER	Secondary Event Register	—
2240h	SECR	Secondary Event Clear Register	—
2250h	IER	Interrupt Enable Register	—
2258h	IECR	Interrupt Enable Clear Register	—
2260h	IESR	Interrupt Enable Set Register	—
2268h	IPR	Interrupt Pending Register	—
2270h	ICR	Interrupt Clear Register	—
2278h	IEVAL	Interrupt Evaluate Register	—
2280h	QER	QDMA Event Register	—
2284h	QEER	QDMA Event Enable Register	—
2288h	QEECR	QDMA Event Enable Clear Register	—
228Ch	QEESR	QDMA Event Enable Set Register	—
2290h	QSER	QDMA Secondary Event Register	—
2294h	QSECR	QDMA Secondary Event Clear Register	—
4000h-4FFFh	—	Parameter RAM (PaRAM)	—

Copyright © 2013–2016, Texas Instruments Incorporated

16.4.2.1 Global Registers

16.4.2.1.1 Revision Identification Register (REVID)

The revision identification register (REVID) uniquely identifies the EDMA3CC and the specific revision of the EDMA3CC. The REVID is shown in Figure 16-43 and described in Table 16-21.

Figure 16-43. Revision ID Register (REVID)

31	0
REV	
R-4001 5300h	

LEGEND: R = Read only; -*n* = value after reset

Table 16-21. Revision ID Register (REVID) Field Descriptions

Bit	Field	Value	Description
31-0	REV		Peripheral identifier.
		4001 5300h	Uniquely identifies the EDMA3CC and the specific revision of the EDMA3CC.

16.4.2.1.2 EDMA3CC Configuration Register (CCCFG)

The EDMA3CC configuration register (CCCFG) provides the features/resources for the EDMA3CC in a particular device. The CCCFG is shown in Figure 16-44 and described in Table 16-22.

Figure 16-44. EDMA3CC Configuration Register (CCCFG)

31	26	25	24
Reserved		MP_EXIST	CHMAP_EXIST
R-x		R-0	R-0

23	22	21	20	19	18	16
Reserved		NUM_REGN		Reserved	NUM_EVQUE	
R-0		R-2h		R-x	R-1h	

15	14	12	11	10	8
Reserved	NUM_PAENTRY		Reserved	NUM_INTCH	
R-x	R-3h		R-x	R-3h	

7	6	4	3	2	0
Reserved	NUM_QDMACH		Reserved	NUM_DMACH	
R-x	R-4h		R-x	R-4h	

LEGEND: R = Read only; -*n* = value after reset; -x = value is indeterminate after reset

Copyright © 2013–2016, Texas Instruments Incorporated

Table 16-22. EDMA3CC Configuration Register (CCCFG) Field Descriptions

Bit	Field	Value	Description
31-26	Reserved	0-3Fh	Reserved
25	MP_EXIST		Memory protection existence.
		0	No memory protection.
		1	Reserved
24	CHMAP_EXIST		Channel mapping existence.
		0	No channel mapping. This implies that there is fixed association for a channel number to a parameter entry number or, in other words, PaRAM entry *n* corresponds to channel *n*.
		1	Reserved
23-22	Reserved	0	Reserved
21-20	NUM_REGN	0-3h	Number of shadow regions.
		0-1h	Reserved
		2h	4 regions
		3h	Reserved
19	Reserved	0	Reserved
18-16	NUM_EVQUE	0-7h	Number of queues/number of transfer controllers.
		0	Reserved
		1h	2 event queues
		2h	2 transfer controllers
		3h-7h	Reserved
15	Reserved	0	Reserved
14-12	NUM_PAENTRY	0-7h	Number of PaRAM sets.
		0-2h	Reserved
		3h	128 PaRAM sets
		4h-7h	Reserved
11	Reserved	0	Reserved
10-8	NUM_INTCH	0-7h	Number of interrupt channels.
		0-2h	Reserved
		3h	32 interrupt channels
		4h-7h	Reserved
7	Reserved	0	Reserved
6-4	NUM_QDMACH	0-7h	Number of QDMA channels.
		0-3h	Reserved
		4h	8 QDMA channels
		5h-7h	Reserved
3	Reserved	0	Reserved
2-0	NUM_DMACH	0-7h	Number of DMA channels.
		0-3h	Reserved
		4h	32 DMA channels
		5h-7h	Reserved

Copyright © 2013–2016, Texas Instruments Incorporated

16.4.2.1.3 QDMA Channel *n* Mapping Register (QCHMAP*n*)

Each QDMA channel in EDMA3CC can be associated with any PaRAM set available on the device. Furthermore, the specific trigger word (0-7) of the PaRAM set can be programmed. The PaRAM set association and trigger word for every QDMA channel register is configurable using the QDMA channel *n* mapping register (QCHMAP*n*). The QCHMAP*n* is shown in Figure 16-45 and described in Table 16-23.

NOTE: At reset the QDMA channel mapping registers for all QDMA channels point to the PaRAM set 0. Prior to using any QDMA channel, QCHMAP*n* should be programmed appropriately to point to a different PaRAM set.

Figure 16-45. QDMA Channel *n* Mapping Register (QCHMAP*n*)

31	16
Reserved	
R-0	

15	14	13	5	4	2	1	0
Reserved		PAENTRY		TRWORD		Reserved	
R-0		R/W-0		R/W-0		R-0	

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 16-23. QDMA Channel *n* Mapping Register (QCHMAP*n*) Field Descriptions

Bit	Field	Value	Description
31-14	Reserved	0	Reserved
13-5	PAENTRY	0-1FFh	PAENTRY points to the PaRAM set number for QDMA channel *n*.
		0-7Fh	PaRAM set number 0 through 127
		80h-1FFh	Reserved
4-2	TRWORD	0-7h	Points to the specific PaRAM entry or the trigger word in the PaRAM set pointed to by PAENTRY. A write to the trigger word results in a QDMA event being recognized.
1-0	Reserved	0	Reserved

Copyright © 2013–2016, Texas Instruments Incorporated

16.4.2.1.4 DMA Channel Queue Number Register *n* (DMAQNUM*n*)

The DMA channel queue number register *n* (DMAQNUM*n*) allows programmability of each of the 32 DMA channels in the EDMA3CC to submit its associated synchronization event to any event queue in the EDMA3CC. At reset, all channels point to event queue 0. The DMAQNUM*n* is shown in Figure 16-46 and described in Table 16-24. Table 16-25 shows the channels and their corresponding bits in DMAQNUM*n*.

NOTE: Since the event queues in EDMA3CC have a fixed association to the transfer controllers, that is, Q0 TRs are submitted to TC0 and Q1 TRs are submitted to TC1, by programming DMAQNUM*n* for a particular DMA channel also dictates which transfer controller is utilized for the data movement (or which EDMA3TC receives the TR request).

Figure 16-46. DMA Channel Queue Number Register *n* (DMAQNUM*n*)

31	30 28	27	26 24	23	22 20	19	18 16
Rsvd	E*n*	Rsvd	E*n*	Rsvd	E*n*	Rsvd	E*n*
R-0	R/W-0	R-0	R/W-0	R-0	R/W-0	R-0	R/W-0

15	14 12	11	10 8	7	6 4	3	2 0
Rsvd	E*n*	Rsvd	E*n*	Rsvd	E*n*	Rsvd	E*n*
R-0	R/W-0	R-0	R/W-0	R-0	R/W-0	R-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 16-24. DMA Channel Queue Number Register *n* (DMAQNUM*n*) Field Descriptions

Bit	Field	Value	Description
31-0	E*n*	0-7h	DMA queue number. Contains the event queue number to be used for the corresponding DMA channel. Programming DMAQNUM*n* for an event queue number to a value more then the number of queues available in the EDMA3CC results in undefined behavior.
		0	Event *n* is queued on Q0.
		1h	Event *n* is queued on Q1.
		2h-7h	Reserved

Table 16-25. Bits in DMAQNUM*n*

	DMAQNUM*n*			
E*n* bit	**0**	**1**	**2**	**3**
0-2	E0	E8	E16	E24
4-6	E1	E9	E17	E25
8-10	E2	E10	E18	E26
12-14	E3	E11	E19	E27
16-18	E4	E12	E20	E28
20-22	E5	E13	E21	E29
24-26	E6	E14	E22	E30
28-30	E7	E15	E23	E31

Copyright © 2013–2016, Texas Instruments Incorporated

16.4.2.1.5 QDMA Channel Queue Number Register (QDMAQNUM)

The QDMA channel queue number register (QDMAQNUM) is used to program all the QDMA channels in the EDMA3CC to submit the associated QDMA event to any of the event queues in the EDMA3CC. The QDMAQNUM is shown in Figure 16-47 and described in Table 16-26.

Figure 16-47. QDMA Channel Queue Number Register (QDMAQNUM)

31	30 28	27	26 24	23	22 20	19	18 16
Rsvd	E7	Rsvd	E6	Rsvd	E5	Rsvd	E4
R-0	R/W-0	R-0	R/W-0	R-0	R/W-0	R-0	R/W-0

15	14 12	11	10 8	7	6 4	3	2 0
Rsvd	E3	Rsvd	E2	Rsvd	E1	Rsvd	E0
R-0	R/W-0	R-0	R/W-0	R-0	R/W-0	R-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 16-26. QDMA Channel Queue Number Register (QDMAQNUM) Field Descriptions

Bit	Field	Value	Description
31-0	E*n*	0-7h	QDMA queue number. Contains the event queue number to be used for the corresponding QDMA channel.
		0	Event *n* is queued on Q0.
		1h	Event *n* is queued on Q1.
		2h-7h	Reserved

16.4.2.1.6 Queue Priority Register (QUEPRI)

On previous architectures, the EDMA3TC priority was controlled by the queue priority register (QUEPRI) in the EDMA3CC memory-map. However for this device, the priority control for the transfer controllers is controlled by the chip-level registers in the System Configuration Module. You should use the chip-level registers and not QUEPRI to configure the TC priority.

Copyright © 2013–2016, Texas Instruments Incorporated

16.4.2.2 Error Registers

The EDMA3CC contains a set of registers that provide information on missed DMA and/or QDMA events, and instances when event queue thresholds are exceeded. If any of the bits in these registers is set, it results in the EDMA3CC generating an error interrupt.

16.4.2.2.1 Event Missed Registers (EMR)

For a particular DMA channel, if a second event is received prior to the first event getting cleared/serviced, the bit corresponding to that channel is set/asserted in the event missed register (EMR). All trigger types are treated individually, that is, manual triggered (ESR), chain triggered (CER), and event triggered (ER) are all treated separately. The EMR bit for a channel is also set if an event on that channel encounters a NULL entry (or a NULL TR is serviced). If any EMR bit is set (and all errors, including bits in other error registers (QEMR, CCERR) were previously cleared), the EDMA3CC generates an error interrupt. See Section 16.2.9.4 for details on EDMA3CC error interrupt generation.

The EMR is shown in Figure 16-48 and described in Table 16-27.

Figure 16-48. Event Missed Register (EMR)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
E31	E30	E29	E28	E27	E26	E25	E24	E23	E22	E21	E20	E19	E18	E17	E16
R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
E15	E14	E13	E12	E11	E10	E9	E8	E7	E6	E5	E4	E3	E2	E1	E0
R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0

LEGEND: R = Read only; -*n* = value after reset

Table 16-27. Event Missed Register (EMR) Field Descriptions

Bit	Field	Value	Description
31-0	E*n*		Channel 0-31 event missed. E*n* is cleared by writing a 1 to the corresponding bit in the event missed clear register (EMCR).
		0	No missed event.
		1	Missed event occurred.

Copyright © 2013–2016, Texas Instruments Incorporated

16.4.2.2.2 Event Missed Clear Registers (EMCR)

Once a missed event is posted in the event missed register (EMR), the bit remains set and you need to clear the set bit(s). This is done by way of CPU writes to the event missed clear register (EMCR). Writing a 1 to any of the bits clears the corresponding missed event (bit) in EMR; writing a 0 has no effect.

The EMCR is shown in Figure 16-49 and described in Table 16-28.

Figure 16-49. Event Missed Clear Register (EMCR)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
E31	E30	E29	E28	E27	E26	E25	E24	E23	E22	E21	E20	E19	E18	E17	E16
W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
E15	E14	E13	E12	E11	E10	E9	E8	E7	E6	E5	E4	E3	E2	E1	E0
W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0

LEGEND: W = Write only; -*n* = value after reset

Table 16-28. Event Missed Clear Register (EMCR) Field Descriptions

Bit	Field	Value	Description
31-0	E*n*		Event missed 0-31 clear. All error bits must be cleared before additional error interrupts will be asserted by the EDMA3CC.
		0	No effect.
		1	Corresponding missed event bit in the event missed register (EMR) is cleared (E*n* = 0).

Copyright © 2013–2016, Texas Instruments Incorporated

16.4.2.2.3 QDMA Event Missed Register (QEMR)

For a particular QDMA channel, if two QDMA events are detected without the first event getting cleared/serviced, the bit corresponding to that channel is set/asserted in the QDMA event missed register (QEMR). The QEMR bits for a channel are also set if a QDMA event on the channel encounters a NULL entry (or a NULL TR is serviced). If any QEMR bit is set (and all errors, including bits in other error registers (EMR or CCERR) were previously cleared), the EDMA3CC generates an error interrupt. See Section 16.2.9.4 for details on EDMA3CC error interrupt generation.

The QEMR is shown in Figure 16-50 and described in Table 16-29.

Figure 16-50. QDMA Event Missed Register (QEMR)

31	16
Reserved	
R-0	

15	8	7	6	5	4	3	2	1	0
Reserved		E7	E6	E5	E4	E3	E2	E1	E0
R-0		R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0

LEGEND: R = Read only; -*n* = value after reset

Table 16-29. QDMA Event Missed Register (QEMR) Field Descriptions

Bit	Field	Value	Description
31-8	Reserved	0	Reserved
7-0	E*n*		Channel 0-7 QDMA event missed. E*n* is cleared by writing a 1 to the corresponding bit in the QDMA event missed clear register (QEMCR).
		0	No missed event.
		1	Missed event occurred.

Copyright © 2013–2016, Texas Instruments Incorporated

16.4.2.2.4 QDMA Event Missed Clear Register (QEMCR)

Once a missed event is posted in the QDMA event missed registers (QEMR), the bit remains set and you need to clear the set bit(s). This is done by way of CPU writes to the QDMA event missed clear registers (QEMCR). Writing a 1 to any of the bits clears the corresponding missed event (bit) in QEMR; writing a 0 has no effect.

The QEMCR is shown in Figure 16-51 and described in Table 16-30.

Figure 16-51. QDMA Event Missed Clear Register (QEMCR)

31	16
Reserved	
R-0	

15	8	7	6	5	4	3	2	1	0
Reserved		E7	E6	E5	E4	E3	E2	E1	E0
R-0		W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0

LEGEND: W = Write only; -*n* = value after reset

Table 16-30. QDMA Event Missed Clear Register (QEMCR) Field Descriptions

Bit	Field	Value	Description
31-8	Reserved	0	Reserved
7-0	E*n*		QDMA event missed clear. All error bits must be cleared before additional error interrupts will be asserted by the EDMA3CC.
		0	No effect.
		1	Corresponding missed event bit in the QDMA event missed register (QEMR) is cleared (E*n* = 0).

Copyright © 2013–2016, Texas Instruments Incorporated

16.4.2.2.5 EDMA3CC Error Register (CCERR)

The EDMA3CC error register (CCERR) indicates whether or not at any instant of time the number of events queued up in any of the event queues exceeds or equals the threshold/watermark value that is set in the queue watermark threshold register (QWMTHRA). Additionally, CCERR also indicates if when the number of outstanding TRs that have been programmed to return transfer completion code (TRs that have the TCINTEN or TCCHEN bit in OPT set to 1) to the EDMA3CC has exceeded the maximum allowed value of 31. If any bit in CCERR is set (and all errors, including bits in other error registers (EMR or QEMR) were previously cleared), the EDMA3CC generates an error interrupt. See Section 16.2.9.4 for details on EDMA3CC error interrupt generation. Once the error bits are set in CCERR, they can only be cleared by writing to the corresponding bits in the EDMA3CC error clear register (CCERRCLR).

The CCERR is shown in Figure 16-52 and described in Table 16-31.

Figure 16-52. EDMA3CC Error Register (CCERR)

31	17	16
Reserved		TCCERR
R-0		R-0

15	2	1	0
Reserved		QTHRXCD1	QTHRXCD0
R-0		R-0	R-0

LEGEND: R = Read only; -*n* = value after reset

Table 16-31. EDMA3CC Error Register (CCERR) Field Descriptions

Bit	Field	Value	Description
31-17	Reserved	0	Reserved
16	TCCERR		Transfer completion code error. TCCERR is cleared by writing a 1 to the corresponding bit in the EDMA3CC error clear register (CCERRCLR).
		0	Total number of allowed TCCs outstanding has not been reached.
		1	Total number of allowed TCCs has been reached.
15-2	Reserved	0	Reserved
1	QTHRXCD1		Queue threshold error for queue 1. QTHRXCD1 is cleared by writing a 1 to the corresponding bit in the EDMA3CC error clear register (CCERRCLR).
		0	Watermark/threshold has not been exceeded.
		1	Watermark/threshold has been exceeded.
0	QTHRXCD0		Queue threshold error for queue 0. QTHRXCD0 is cleared by writing a 1 to the corresponding bit in the EDMA3CC error clear register (CCERRCLR).
		0	Watermark/threshold has not been exceeded.
		1	Watermark/threshold has been exceeded.

Copyright © 2013–2016, Texas Instruments Incorporated

16.4.2.2.6 EDMA3CC Error Clear Register (CCERRCLR)

The EDMA3CC error clear register (CCERRCLR) is used to clear any error bits that are set in the EDMA3CC error register (CCERR). In addition, CCERRCLR also clears the values of some bit fields in the queue status registers (QSTAT*n*) associated with a particular event queue. Writing a 1 to any of the bits clears the corresponding bit in CCERR; writing a 0 has no effect.

The CCERRCLR is shown in Figure 16-53 and described in Table 16-32.

Figure 16-53. EDMA3CC Error Clear Register (CCERRCLR)

31	17	16
Reserved		TCCERR
W-0		W-0

15	2	1	0
Reserved		QTHRXCD1	QTHRXCD0
W-0		W-0	W-0

LEGEND: W= Write only; -*n* = value after reset

Table 16-32. EDMA3CC Error Clear Register (CCERRCLR) Field Descriptions

Bit	Field	Value	Description
31-17	Reserved	0	Reserved
16	TCCERR		Transfer completion code error clear.
		0	No effect.
		1	Clears the TCCERR bit in the EDMA3CC error register (CCERR).
15-2	Reserved	0	Reserved
1	QTHRXCD1		Queue threshold error clear for queue 1.
		0	No effect.
		1	Clears the QTHRXCD1 bit in the EDMA3CC error register (CCERR) and the WM and THRXCD bits in the queue status register 1 (QSTAT1).
0	QTHRXCD0		Queue threshold error clear for queue 0.
		0	No effect.
		1	Clears the QTHRXCD0 bit in the EDMA3CC error register (CCERR) and the WM and THRXCD bits in the queue status register 0 (QSTAT0).

Copyright © 2013–2016, Texas Instruments Incorporated

16.4.2.2.7 Error Evaluate Register (EEVAL)

The EDMA3CC error interrupt is asserted whenever an error bit is set in any of the error registers (EMR, QEMR, and CCERR). For subsequent error bits that get set, the EDMA3CC error interrupt is reasserted only when transitioning from an “all the error bits cleared” to “at least one error bit is set”. Alternatively, a CPU write of 1 to the EVAL bit in the error evaluate register (EEVAL) results in reasserting the EDMA3CC error interrupt, if there are any outstanding error bits set due to subsequent error conditions. Writes of 0 have no effect.

The EEVAL is shown in Figure 16-54 and described in Table 16-33.

Figure 16-54. Error Evaluate Register (EEVAL)

31	16
Reserved	
R-0	

15	2	1	0
Reserved		Rsvd	EVAL
R-0		R/W-0	W-0

LEGEND: R/W = Read/Write; R = Read only; W = Write only; -*n* = value after reset

Table 16-33. Error Evaluate Register (EEVAL) Field Descriptions

Bit	Field	Value	Description
31-2	Reserved	0	Reserved
1	Reserved	0	Reserved. Always write 0 to this bit; writes of 1 to this bit are not supported and attempts to do so may result in undefined behavior.
0	EVAL		Error interrupt evaluate.
		0	No effect.
		1	EDMA3CC error interrupt will be pulsed if any errors have not been cleared in any of the error registers (EMR, QEMR, or CCERR).

Copyright © 2013–2016, Texas Instruments Incorporated

16.4.2.3 Region Access Enable Registers

The region access enable register group consists of the DMA access enable registers (DRAE*m*) and the QDMA access enable registers (QRAE*m*). Where *m* is the number of shadow regions in the EDMA3CC memory-map for a device. You can configure these registers to assign ownership of DMA/QDMA channels to a particular shadow region.

16.4.2.3.1 DMA Region Access Enable for Region m (DRAEm)

The DMA region access enable registers for shadow region *m* (DRAE*m*) is programmed to allow or disallow read/write accesses on a bit-by-bit bases for all DMA registers in the shadow region *m* view of the DMA channel registers. See the EDMA3CC register memory-map for a list of all the DMA channel and interrupt registers mapped in the shadow region view. Additionally, the DRAE*m* configuration determines completion of which DMA channels will result in assertion of the shadow region *m* DMA completion interrupt (see Section 16.2.9).

The DRAE*m* is shown in Figure 16-55 and described in Table 16-34.

Figure 16-55. DMA Region Access Enable Register for Region *m* (DRAE*m*)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
E31	E30	E29	E28	E27	E26	E25	E24	E23	E22	E21	E20	E19	E18	E17	E16
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
E15	E14	E13	E12	E11	E10	E9	E8	E7	E6	E5	E4	E3	E2	E1	E0
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0

LEGEND: R/W = Read/Write; -*n* = value after reset

Table 16-34. DMA Region Access Enable Register for Region *m* (DRAE*m*) Field Descriptions

Bit	Field	Value	Description
31-0	E*n*		DMA region access enable for bit *n*/channel *n* in region *m*.
		0	Accesses via region *m* address space to bit *n* in any DMA channel register are not allowed. Reads return 0 on bit *n* and writes do not modify the state of bit *n*. Enabled interrupt bits for bit *n* do not contribute to the generation of a transfer completion interrupt for shadow region *m*.
		1	Accesses via region *m* address space to bit *n* in any DMA channel register are allowed. Reads return the value from bit *n* and writes modify the state of bit *n*. Enabled interrupt bits for bit *n* contribute to the generation of a transfer completion interrupt for shadow region *m*.

Copyright © 2013–2016, Texas Instruments Incorporated

16.4.2.3.2 QDMA Region Access Enable Registers (QRAE*m*)

The QDMA region access enable registers for shadow region *m* (QRAE*m*) is programmed to allow or disallow read/write accesses on a bit-by-bit bases for all QDMA registers in the shadow region *m* view of the QDMA registers. This includes all 8-bit QDMA registers.

The QRAE*m* is shown in Figure 16-56 and described in Table 16-35.

Figure 16-56. QDMA Region Access Enable for Region *m* (QRAE*m*)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
R-0															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								E7	E6	E5	E4	E3	E2	E1	E0
R-0								R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 16-35. QDMA Region Access Enable for Region *m* (QRAE*m*) Field Descriptions

Bit	Field	Value	Description
31-8	Reserved	0	Reserved
7-0	E*n*		QDMA region access enable for bit *n*/QDMA channel *n* in region *m*.
		0	Accesses via region *m* address space to bit *n* in any QDMA channel register are not allowed. Reads return 0 on bit *n* and writes do not modify the state of bit *n*.
		1	Accesses via region *m* address space to bit *n* in any QDMA channel register are allowed. Reads return the value from bit *n* and writes modify the state of bit *n*.

Copyright © 2013–2016, Texas Instruments Incorporated

16.4.2.4 Status/Debug Visibility Registers

The following set of registers provide visibility into the event queues and a TR lifecycle. These are useful for system debug as they provide in-depth visibility for the events queued up in the event queue and also provide information on what parts of the EDMA3CC logic are active once the event has been received by the EDMA3CC.

16.4.2.4.1 Event Queue Entry Registers (QxEy)

The event queue entry registers (Q*x*E*y*) exist for all 16 queue entries (the maximum allowed queue entries) for all event queues (Q0 and Q1) in the EDMA3CC: Q0E0 to Q0E15 and Q1E0 to Q1E15. Each register details the event number (ENUM) and the event type (ETYPE). For example, if the value in Q1E4 is read as 0000 004Fh, this means the 4th entry in queue 1 is a manually-triggered event on DMA channel 15.

The Q*x*E*y* is shown in Figure 16-57 and described in Table 16-36.

Figure 16-57. Event Queue Entry Registers (Q*x*E*y*)

31	16
Reserved	
R-0	

15	8	7	6	5	4	0
Reserved		ETYPE		Rsvd	ENUM	
R-0		R-x		R-0	R-x	

LEGEND: R = Read only; -*n* = value after reset; -x = value is indeterminate after reset

Table 16-36. Event Queue Entry Registers (Q*x*E*y*) Field Descriptions

Bit	Field	Value	Description
31-8	Reserved	0	Reserved
7-6	ETYPE	0-3h	Event entry *y* in queue *x*. Specifies the specific event type for the given entry in the event queue.
		0	Event triggered via ER
		1h	Manual triggered via ESR
		2h	Chain triggered via CER
		3h	Autotriggered via QER
5	Reserved	0	Reserved
4-0	ENUM	0-1Fh	Event entry *y* in queue *x*. Event number:
		0-7h	QDMA channel number (0 to 7)
		0-1Fh	DMA channel/event number (0 to 31)

Copyright © 2013–2016, Texas Instruments Incorporated

16.4.2.4.2 Queue *n* Status Registers (QSTAT*n*)

The queue *n* status register (QSTAT*n*) is shown in Figure 16-58 and described in Table 16-37.

Figure 16-58. Queue *n* Status Register (QSTAT*n*)

31 – 25	24	23 – 21	20 – 16
Reserved	THRXCD	Reserved	WM
R-0	R-0	R-0	R-0

15 – 13	12 – 8	7 – 4	3 – 0
Reserved	NUMVAL	Reserved	STRTPTR
R-0	R-0	R-0	R-0

LEGEND: R = Read only; -*n* = value after reset

Table 16-37. Queue *n* Status Register (QSTAT*n*) Field Descriptions

Bit	Field	Value	Description
31-25	Reserved	0	Reserved
24	THRXCD		Threshold exceeded. THRXCD is cleared by writing a 1 to the corresponding QTHRXCD*n* bit in the EDMA3CC error clear register (CCERRCLR).
		0	Threshold specified by the Q*n* bit in the queue watermark threshold A register (QWMTHRA) has not been exceeded.
		1	Threshold specified by the Q*n* bit in the queue watermark threshold A register (QWMTHRA) has been exceeded.
23-21	Reserved	0	Reserved
20-16	WM	0-1Fh	Watermark for maximum queue usage. Watermark tracks the most entries that have been in queue *n* since reset or since the last time that the watermark (WM) bit was cleared. WM is cleared by writing a 1 to the corresponding QTHRXCD*n* bit in the EDMA3CC error clear register (CCERRCLR).
		0-10h	Legal values are 0 (empty) to 10h (full).
		11h-1Fh	Reserved
15-13	Reserved	0	Reserved
12-8	NUMVAL	0-1Fh	Number of valid entries in queue *n*. The total number of entries residing in the queue manager FIFO at a given instant. Always enabled.
		0-10h	Legal values are 0 (empty) to 10h (full).
		11h-1Fh	Reserved
7-4	Reserved	0	Reserved
3-0	STRTPTR	0-Fh	Start pointer. The offset to the head entry of queue *n*, in units of entries. Always enabled. Legal values are 0 (0th entry) to Fh (15th entry).

Copyright © 2013–2016, Texas Instruments Incorporated

16.4.2.4.3 Queue Watermark Threshold A Register (QWMTHRA)

The queue watermark threshold A register (QWMTHRA) is shown in Figure 16-59 and described in Table 16-38.

Figure 16-59. Queue Watermark Threshold A Register (QWMTHRA)

31	16
Reserved	
R-0	

15	13	12	8	7	5	4	0
Reserved		Q1		Reserved		Q0	
R-0		R/W-10h		R-0		R/W-10h	

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 16-38. Queue Watermark Threshold A Register (QWMTHRA) Field Descriptions

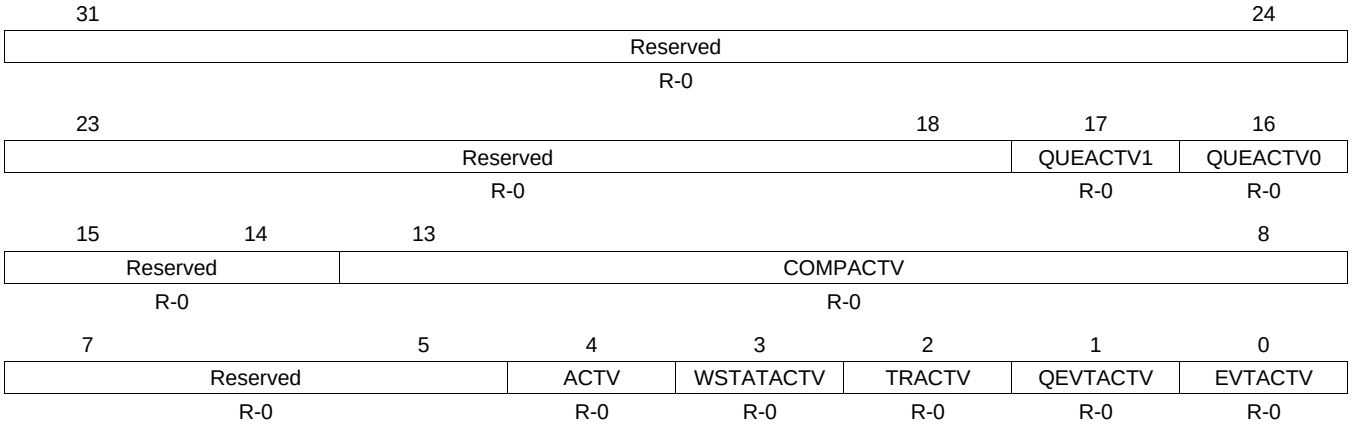
Bit	Field	Value	Description
31-13	Reserved	0	Reserved
12-8	Q1	0-1Fh	Queue threshold for queue 1 value. The QTHRXCD1 bit in the EDMA3CC error register (CCERR) and the THRXCD bit in the queue status register 1 (QSTAT1) are set when the number of events in queue 1 at an instant in time (visible via the NUMVAL bit in QSTAT1) equals or exceeds the value specified by Q1.
		0-10h	The default is 16 (maximum allowed).
		11h	Disables the threshold errors.
		12h-1Fh	Reserved
7-5	Reserved	0	Reserved
4-0	Q0	0-1Fh	Queue threshold for queue 0 value. The QTHRXCD0 bit in the EDMA3CC error register (CCERR) and the THRXCD bit in the queue status register 0 (QSTAT0) are set when the number of events in queue 0 at an instant in time (visible via the NUMVAL bit in QSTAT0) equals or exceeds the value specified by Q0.
		0-10h	The default is 16 (maximum allowed).
		11h	Disables the threshold errors.
		12h-1Fh	Reserved

Copyright © 2013–2016, Texas Instruments Incorporated

16.4.2.4.4 EDMA3CC Status Register (CCSTAT)

The EDMA3CC status register (CCSTAT) has a number of status bits that reflect which parts of the EDMA3CC logic is active at any given instant of time. The CCSTAT is shown in Figure 16-60 and described in Table 16-39.

Figure 16-60. EDMA3CC Status Register (CCSTAT)



LEGEND: R = Read only; -*n* = value after reset

Table 16-39. EDMA3CC Status Register (CCSTAT) Field Descriptions

Bit	Field	Value	Description
31-18	Reserved	0	Reserved
17	QUEACTV1		Queue 1 active.
		0	No events are queued in queue 1.
		1	At least one TR is queued in queue 1.
16	QUEACTV0		Queue 0 active.
		0	No events are queued in queue 0.
		1	At least one TR is queued in queue 0.
15-14	Reserved	0	Reserved
13-8	COMPACTV	0-3Fh	Completion request active. The COMPACTV field reflects the count for the number of completion requests submitted to the transfer controllers. This count increments every time a TR is submitted and is programmed to report completion (the TCINTEN or TCCCHEN bits in OPT in the parameter entry associated with the TR are set to 1). The counter decrements for every valid TCC received back from the transfer controllers. If at any time the count reaches a value of 63, the EDMA3CC will not service any new TRs until the count is less then 63 (or return a transfer completion code from a transfer controller, which would decrement the count).
		0	No completion requests outstanding.
		1h-3Fh	Total of 1 completion request to 63 completion requests are outstanding.
7-5	Reserved	0	Reserved
4	ACTV		Channel controller active. Channel controller active is a logical-OR of each of the *ACTV bits. The ACTV bit remains high through the life of a TR.
		0	Channel is idle.
		1	Channel is busy.
3	WSTATACTV		Write status interface active.
		0	Write status req is idle and write status fifo is idle.
		1	Either the write status request is active or additional write status responses are pending in the write status fifo.
2	TRACTV		Transfer request active.
		0	Transfer request processing/submission logic is inactive.
		1	Transfer request processing/submission logic is active.

Copyright © 2013–2016, Texas Instruments Incorporated

Table 16-39. EDMA3CC Status Register (CCSTAT) Field Descriptions (continued)

Bit	Field	Value	Description
1	QEVTACTV		QDMA event active.
		0	No enabled QDMA events are active within the EDMA3CC.
		1	At least one enabled QDMA event (QER) is active within the EDMA3CC.
0	EVTACTV		DMA event active.
		0	No enabled DMA events are active within the EDMA3CC.
		1	At least one enabled DMA event (ER and EER, ESR, CER) is active within the EDMA3CC.

Copyright © 2013–2016, Texas Instruments Incorporated

16.4.2.5 DMA Channel Registers

The following registers pertain to the 32 DMA channels. The 32 DMA channels consist of registers (with the exception of DMAQNUM*n*) that each have 32 bits and the bit position of each register matches the DMA channel number.

The DMA channel registers are accessible via read/writes to the global address range. They are also accessible via read/writes to the shadow address range. The read/write ability to the registers in the shadow region is controlled by the DMA region access registers (DRAE*m*). These registers are described in Section 16.4.2.3.1 and the details for shadow region/global region usage is explained in Section 16.2.7.

16.4.2.5.1 Event Register (ER)

All external events are captured in the event register (ER). The events are latched even when the events are not enabled. If the event bit corresponding to the latched event is enabled (EER.E*n* = 1), then the event is evaluated by the EDMA3CC logic for an associated transfer request submission to the transfer controllers. The event register bits are automatically cleared (ER.E*n* = 0) once the corresponding events are prioritized and serviced. If ER.E*n* are already set and another event is received on the same channel/event, then the corresponding event is latched in the event miss register (EMR.E*n*), provided that the event was enabled (EER.E*n* = 1).

Event *n* can be cleared by the CPU writing a 1 to corresponding event bit in the event clear register (ECR). The setting of an event is a higher priority relative to clear operations (via hardware or software). If set and clear conditions occur concurrently, the set condition wins. If the event was previously set, then EMR would be set since an event is lost. If the event was previously clear, then the event remains set and is prioritized for submission to the event queues.

The ER is shown in Figure 16-61 and described in Table 16-40.

Figure 16-61. Event Register (ER)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
E31	E30	E29	E28	E27	E26	E25	E24	E23	E22	E21	E20	19	E18	E17	E16
R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
E15	E14	E13	E12	E11	E10	E9	E8	E7	E6	E5	E4	E3	E2	E1	E0
R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0

LEGEND: R = Read only; -*n* = value after reset

Table 16-40. Event Register (ER) Field Descriptions

Bit	Field	Value	Description
31-0	E*n*		Event 0-31. Events 0-31 are captured by the EDMA3CC and are latched into ER. The events are set (E*n* = 1) even when events are disabled (E*n* = 0 in the event enable register, EER).
		0	EDMA3CC event is not asserted.
		1	EDMA3CC event is asserted. Corresponding DMA event is prioritized versus other pending DMA/QDMA events for submission to the EDMA3TC.

Copyright © 2013–2016, Texas Instruments Incorporated

16.4.2.5.2 Event Clear Register (ECR)

Once an event has been posted in the event register (ER), the event is cleared in two ways. If the event is enabled in the event enable register (EER) and the EDMA3CC submits a transfer request for the event to the EDMA3TC, it clears the corresponding event bit in the event register. If the event is disabled in the event enable register (EER), the CPU can clear the event by way of the event clear register (ECR).

Writing a 1 to any of the bits clears the corresponding event; writing a 0 has no effect. Once an event bit is set in the event register, it remains set until EDMA3CC submits a transfer request for that event or the CPU clears the event by setting the corresponding bit in ECR.

The ECR is shown in Figure 16-62 and described in Table 16-41.

Figure 16-62. Event Clear Register (ECR)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
E31	E30	E29	E28	E27	E26	E25	E24	E23	E22	E21	E20	E19	E18	E17	E16
W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
E15	E14	E13	E12	E11	E10	E9	E8	E7	E6	E5	E4	E3	E2	E1	E0
W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0

LEGEND: W = Write only; -*n* = value after reset

Table 16-41. Event Clear Register (ECR) Field Descriptions

Bit	Field	Value	Description
31-0	E*n*		Event clear for event 0-31. Any of the event bits in ECR is set to 1 to clear the event (E*n*) in the event register (ER). A write of 0 has no effect.
		0	No effect.
		1	EDMA3CC event is cleared in the event register (ER).

Copyright © 2013–2016, Texas Instruments Incorporated

16.4.2.5.3 Event Set Register (ESR)

The event set register (ESR) allows the CPU (or EDMA programmers) to manually set events to initiate DMA transfer requests. CPU writes of 1 to any event set register (E*n*) bits set the corresponding bits in the registers. The set event is evaluated by the EDMA3CC logic for an associated transfer request submission to the transfer controllers. Writing a 0 has no effect.

The event set register operates independent of the event register (ER), and a write of 1 is always considered a valid event regardless of whether the event is enabled (the corresponding event bits are set or cleared in EER.E*n*).

Once the event is set in the event set register, it cannot be cleared by CPU writes, in other words, the event clear register (ECR) has no effect on the state of ESR. The bits will only be cleared once the transfer request corresponding to the event has been submitted to the transfer controller. The setting of an event is a higher priority relative to clear operations (via hardware). If set and clear conditions occur concurrently, the set condition wins. If the event was previously set, then EMR would be set since an event is lost. If the event was previously clear, then the event remains set and is prioritized for submission to the event queues.

Manually-triggered transfers via writes to ESR allow the CPU to submit DMA requests in the system, these are relevant for memory-to-memory transfer scenarios. If the ESR.E*n* bit is already set and another CPU write of 1 is attempted to the same bit, then the corresponding event is latched in the event missed registers (EMR.E*n* = 1).

The ESR is shown in Figure 16-63 and described in Table 16-42.

Figure 16-63. Event Set Register (ESR)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
E31	E30	E29	E28	E27	E26	E25	E24	E23	E22	E21	E20	E19	E18	E17	E16
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
E15	E14	E13	E12	E11	E10	E9	E8	E7	E6	E5	E4	E3	E2	E1	E0
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0

LEGEND: R/W = Read/Write; -*n* = value after reset

Table 16-42. Event Set Register (ESR) Field Descriptions

Bit	Field	Value	Description
31-0	E*n*		Event set for event 0-31.
		0	No effect.
		1	Corresponding DMA event is prioritized versus other pending DMA/QDMA events for submission to the EDMA3TC.

Copyright © 2013–2016, Texas Instruments Incorporated

16.4.2.5.4 Chained Event Register (CER)

When the OPTIONS parameter for a PaRAM entry is programmed to returned a chained completion code (ITCCHEN = 1 and/or TCCHEN = 1), then the value dictated by the TCC[5:0] (also programmed in OPT) forces the corresponding event bit to be set in the chained event register (CER). The set chained event is evaluated by the EDMA3CC logic for an associated transfer request submission to the transfer controllers. This results in a chained-triggered transfer.

The chained event registers do not have any enables. The generation of a chained event is essentially enabled by the PaRAM entry that has been configured for intermediate and/or final chaining on transfer completion. The E*n* bit is set (regardless of the state of EER.E*n*) when a chained completion code is returned from one of the transfer controllers or is generated by the EDMA3CC via the early completion path. The bits in the chained event register are cleared when the corresponding events are prioritized and serviced.

If the E*n* bit is already set and another chaining completion code is return for the same event, then the corresponding event is latched in the event missed register (EMR.E*n* = 1). The setting of an event is a higher priority relative to clear operations (via hardware). If set and clear conditions occur concurrently, the set condition wins. If the event was previously set, then EMR would be set since an event is lost. If the event was previously clear, then the event remains set and is prioritized for submission to the event queues.

The CER is shown in Figure 16-64 and described in Table 16-43.

Figure 16-64. Chained Event Register (CER)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
E31	E30	E29	E28	E27	E26	E25	E24	E23	E22	E21	E20	E19	E18	E17	E16
R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
E15	E14	E13	E12	E11	E10	E9	E8	E7	E6	E5	E4	E3	E2	E1	E0
R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0

LEGEND: R = Read only; -*n* = value after reset

Table 16-43. Chained Event Register (CER) Field Descriptions

Bit	Field	Value	Description
31-0	E*n*		Chained event for event 0-31.
		0	No effect.
		1	Corresponding DMA event is prioritized versus other pending DMA/QDMA events for submission to the EDMA3TC.

Copyright © 2013–2016, Texas Instruments Incorporated

16.4.2.5.5 Event Enable Register (EER)

The EDMA3CC provides the option of selectively enabling/disabling each event in the event register (ER) by using the event enable register (EER). If an event bit in EER is set to 1 (using the event enable set register, EESR), it will enable that corresponding event. Alternatively, if an event bit in EER is cleared (using the event enable clear register, EECR), it will disable the corresponding event.

The event register latches all events that are captured by EDMA3CC, even if the events are disabled (although EDMA3CC does not process it). Enabling an event with a pending event already set in the event register enables the EDMA3CC to process the already set event like any other new event. The EER settings do not have any effect on chained events (CER.E*n* = 1) and manually set events (ESR.E*n* = 1).

The EER is shown in Figure 16-65 and described in Table 16-44.

Figure 16-65. Event Enable Register (EER)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
E31	E30	E29	E28	E27	E26	E25	E24	E23	E22	E21	E20	E19	E18	E17	E16
R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
E15	E14	E13	E12	E11	E10	E9	E8	E7	E6	E5	E4	E3	E2	E1	E0
R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0

LEGEND: R = Read only; -*n* = value after reset

Table 16-44. Event Enable Register (EER) Field Descriptions

Bit	Field	Value	Description
31-0	E*n*		Event enable for events 0-31.
		0	Event is not enabled. An external event latched in the event register (ER) is not evaluated by the EDMA3CC.
		1	Event is enabled. An external event latched in the event register (ER) is evaluated by the EDMA3CC.

Copyright © 2013–2016, Texas Instruments Incorporated

16.4.2.5.6 Event Enable Clear Register (EECR)

The event enable register (EER) cannot be modified by directly writing to it. The intent is to ease the software burden for the case where multiple tasks are attempting to simultaneously modify these registers. The event enable clear register (EECR) is used to disable events. Writes of 1 to the bits in EECR clear the corresponding event bits in EER; writes of 0 have no effect.

The EECR is shown in Figure 16-66 and described in Table 16-45.

Figure 16-66. Event Enable Clear Register (EECR)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
E31	E30	E29	E28	E27	E26	E25	E24	E23	E22	E21	E20	E19	E18	E17	E16
W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
E15	E14	E13	E12	E11	E10	E9	E8	E7	E6	E5	E4	E3	E2	E1	E0
W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0

LEGEND: W = Write only; -*n* = value after reset

Table 16-45. Event Enable Clear Register (EECR) Field Descriptions

Bit	Field	Value	Description
31-0	E*n*		Event enable clear for events 0-31.
		0	No effect.
		1	Event is disabled. Corresponding bit in the event enable register (EER) is cleared (E*n* = 0).

16.4.2.5.7 Event Enable Set Register (EESR)

The event enable register (EER) cannot be modified by directly writing to it. The intent is to ease the software burden for the case where multiple tasks are attempting to simultaneously modify these registers. The event enable set register (EESR) is used to enable events. Writes of 1 to the bits in EESR set the corresponding event bits in EER; writes of 0 have no effect.

The EESR is shown in Figure 16-67 and described in Table 16-46.

Figure 16-67. Event Enable Set Register (EESR)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
E31	E30	E29	E28	E27	E26	E25	E24	E23	E22	E21	E20	E19	E18	E17	E16
W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
E15	E14	E13	E12	E11	E10	E9	E8	E7	E6	E5	E4	E3	E2	E1	E0
W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0

LEGEND: W = Write only; -*n* = value after reset

Table 16-46. Event Enable Set Register (EESR) Field Descriptions

Bit	Field	Value	Description
31-0	E*n*		Event enable set for events 0-31.
		0	No effect.
		1	Event is enabled. Corresponding bit in the event enable register (EER) is set (E*n* = 1).

Copyright © 2013–2016, Texas Instruments Incorporated

16.4.2.5.8 Secondary Event Register (SER)

The secondary event register (SER) provides information on the state of a DMA channel or event (0 through 31). If the EDMA3CC receives a TR synchronization due to a manual-trigger, event-trigger, or chained-trigger source (ESR.E*n* = 1, ER.E*n* = 1, or CER.E*n* = 1), which results in the setting of a corresponding event bit in SER (SER.E*n* = 1), it implies that the corresponding DMA event is in the queue.

Once a bit corresponding to an event is set in SER, the EDMA3CC does not prioritize additional events on the same DMA channel. Depending on the condition that leads to the setting of the SER bits, either the EDMA3CC hardware or the software (using SECR) needs to clear the SER bits for the EDMA3CC to evaluate subsequent events and perform subsequent transfers on the same channel. Based on whether the associated TR is valid, or it is a null or dummy TR, the implications on the state of SER and the required user action in order to submit another DMA transfer might be different.

The SER is shown in Figure 16-68 and described in Table 16-47.

Figure 16-68. Secondary Event Register (SER)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
E31	E30	E29	E28	E27	E26	E25	E24	E23	E22	E21	E20	E19	E18	E17	E16
R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
E15	E14	E13	E12	E11	E10	E9	E8	E7	E6	E5	E4	E3	E2	E1	E0
R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0

LEGEND: R = Read only; -*n* = value after reset

Table 16-47. Secondary Event Register (SER) Field Descriptions

Bit	Field	Value	Description
31-0	E*n*		Secondary event register. The secondary event register is used to provide information on the state of an event.
		0	Event is not currently stored in the event queue.
		1	Event is currently stored in the event queue. Event arbiter will not prioritize additional events.

16.4.2.5.9 Secondary Event Clear Register (SECR)

The secondary event clear register (SECR) clears the status of the secondary event registers (SER). CPU writes of 1 clear the corresponding set bits in SER. Writes of 0 have no effect.

The SECR is shown in Figure 16-69 and described in Table 16-48.

Figure 16-69. Secondary Event Clear Register (SECR)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
E31	E30	E29	E28	E27	E26	E25	E24	E23	E22	E21	E20	E19	E18	E17	E16
W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
E15	E14	E13	E12	E11	E10	E9	E8	E7	E6	E5	E4	E3	E2	E1	E0
W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0

LEGEND: W = Write only; -*n* = value after reset

Table 16-48. Secondary Event Clear Register (SECR) Field Descriptions

Bit	Field	Value	Description
31-0	E*n*		Secondary event clear register
		0	No effect.
		1	Corresponding bit in the secondary event register (SER) is cleared (E*n* = 0).

Copyright © 2013–2016, Texas Instruments Incorporated

16.4.2.6 Interrupt Registers

All DMA/QDMA channels can be set to assert an EDMA3CC completion interrupt to the CPU on transfer completion, by appropriately configuring the PaRAM entry associated with the channels. The following registers are used for the transfer completion interrupt reporting/generating by the EDMA3CC. See Section 16.2.9 for more details on EDMA3CC completion interrupt generation.

16.4.2.6.1 Interrupt Enable Registers (IER)

Interrupt enable register (IER) is used to enable/disable the transfer completion interrupt generation by the EDMA3CC for all DMA/QDMA channels. The IER cannot be written to directly. To set any interrupt bit in IER, a 1 must be written to the corresponding interrupt bit in the interrupt enable set registers (IESR). Similarly, to clear any interrupt bit in IER, a 1 must be written to the corresponding interrupt bit in the interrupt enable clear register (IECR).

The IER is shown in Figure 16-70 and described in Table 16-49.

Figure 16-70. Interrupt Enable Register (IER)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
I31	I30	I29	I28	I27I	I26	I25	I24	I23	I22	I21	I20	I19	I18	I17	I16
R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
I15	I14	I13	I12	I11	I10	I9	I8	I7	I6	I5	I4	I3	I2	I1	I0
R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0

LEGEND: R = Read only; -*n* = value after reset

Table 16-49. Interrupt Enable Register (IER) Field Descriptions

Bit	Field	Value	Description
31-0	E*n*		Interrupt enable for channels 0-31.
		0	Interrupt is not enabled.
		1	Interrupt is enabled.

Copyright © 2013–2016, Texas Instruments Incorporated

16.4.2.6.2 Interrupt Enable Clear Register (IECR)

The interrupt enable clear register (IECR) is used to clear interrupts. Writes of 1 to the bits in IECR clear the corresponding interrupt bits in the interrupt enable registers (IER); writes of 0 have no effect.

The IECR is shown in Figure 16-71 and described in Table 16-50.

Figure 16-71. Interrupt Enable Clear Register (IECR)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
I31	I30	I29	I28	I27	I26	I25	I24	I23	I22	I21	I20	I19	I18	I17	16
W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
I15	I14	I13	I12	I11	I10	I9	I8	I7	I6	I5	I4	I3	I2	I1	I0
W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0

LEGEND: W = Write only; -*n* = value after reset

Table 16-50. Interrupt Enable Clear Register (IECR) Field Descriptions

Bit	Field	Value	Description
31-0	E*n*		Interrupt enable clear for channels 0-31.
		0	No effect
		1	Corresponding bit in the interrupt enable register (IER) is cleared (I*n* = 0).

16.4.2.6.3 Interrupt Enable Set Register (IESR)

The interrupt enable set register (IESR) is used to enable interrupts. Writes of 1 to the bits in IESR set the corresponding interrupt bits in the interrupt enable registers (IER); writes of 0 have no effect.

The IESR is shown in Figure 16-72 and described in Table 16-51.

Figure 16-72. Interrupt Enable Set Register (IESR)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
I31	I30	I29	I28	I27	I26	I25	I24	I23	I22	I21	I20	I19	II8	I17	I16
W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
I15	I14	I13	I12	I11	I10	I9	I8	I7	I6	I5	I4	I3	I2	I1	I0
W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0

LEGEND: W = Write only; -*n* = value after reset

Table 16-51. Interrupt Enable Set Register (IESR) Field Descriptions

Bit	Field	Value	Description
31-0	E*n*		Interrupt enable set for channels 0-31.
		0	No effect.
		1	Corresponding bit in the interrupt enable register (IER) is set (I*n* = 1).

Copyright © 2013–2016, Texas Instruments Incorporated

16.4.2.6.4 Interrupt Pending Register (IPR)

If the TCINTEN and/or ITCINTEN bit in the channel option parameter (OPT) is set to 1 in the PaRAM entry associated with the channel (DMA or QDMA), then the EDMA3TC (for normal completion) or the EDMA3CC (for early completion) returns a completion code on transfer or intermediate transfer completion. The value of the returned completion code is equal to the TCC bit in OPT for the PaRAM entry associated with the channel.

When an interrupt transfer completion code with TCC = *n* is detected by the EDMA3CC, then the corresponding bit is set in the interrupt pending register (IPR.I*n*, if *n* = 0 to 31). Note that once a bit is set in the interrupt pending registers, it remains set; it is your responsibility to clear these bits. The bits set in IPR are cleared by writing a 1 to the corresponding bits in the interrupt clear registers (ICR).

The IPR is shown in Figure 16-73 and described in Table 16-52.

Figure 16-73. Interrupt Pending Register (IPR)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
I31	I30	I29	I28	I27	I26	I25	I24	I23	I22	I21	I20	I19	I18	I17	I16
R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
I15	I14	I13	I12	I11	I10	I9	I8	I7	I6	I5	I4	I3	I2	I1	I0
R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0

LEGEND: R = Read only; -*n* = value after reset

Table 16-52. Interrupt Pending Register (IPR) Field Descriptions

Bit	Field	Value	Description
31-0	I*n*		Interrupt pending for TCC = 0-31.
		0	Interrupt transfer completion code is not detected or was cleared.
		1	Interrupt transfer completion code is detected (I*n* = 1, *n* = EDMA3TC[5:0]).

Copyright © 2013–2016, Texas Instruments Incorporated

16.4.2.6.5 Interrupt Clear Register (ICR)

The bits in the interrupt pending register (IPR) are cleared by writing a 1 to the corresponding bits in the interrupt clear register (ICR); writes of 0 have no effect. All set bits in IPR must be cleared to allow EDMA3CC to assert additional transfer completion interrupts.

The ICR is shown in Figure 16-74 and described in Table 16-53.

Figure 16-74. Interrupt Clear Register (ICR)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
I31	I30	I29	I28	I27	I26	I25	I24	I23	I22	I21	I20	I19	I18	I17	I16
W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
I15	I14	I13	I12	I11	I10	I9	I8	I7	I6	I5	I4	I3	I2	I1	I0
W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0

LEGEND: W = Write only; -*n* = value after reset

Table 16-53. Interrupt Clear Register (ICR) Field Descriptions

Bit	Field	Value	Description
31-0	I*n*		Interrupt clear register for TCC = 0-31.
		0	No effect.
		1	Corresponding bit in the interrupt pending register (IPR) is cleared (I*n* = 0).

Copyright © 2013–2016, Texas Instruments Incorporated

16.4.2.6.6 Interrupt Evaluate Register (IEVAL)

The interrupt evaluate register (IEVAL) is the only register that physically exists in both the global region and the shadow regions. In other words, the read/write accessibility for the shadow region IEVAL is not affected by the DMA/QDMA region access registers (DRAE*m* and QRAE*m*). IEVAL is needed for robust ISR operations to ensure that interrupts are not missed by the CPU.

The IEVAL is shown in Figure 16-75 and described in Table 16-54.

Figure 16-75. Interrupt Evaluate Register (IEVAL)

31	16
Reserved	
R-0	

15	2	1	0
Reserved		Rsvd	EVAL
R-0		W-0	W-0

LEGEND: R = Read only; W = Write only; -*n* = value after reset

Table 16-54. Interrupt Evaluate Register (IEVAL) Field Descriptions

Bit	Field	Value	Description
31-2	Reserved	0	Reserved
1	Reserved	0	Reserved. Always write 0 to this bit; writes of 1 to this bit are not supported and attempts to do so may result in undefined behavior.
0	EVAL		Interrupt evaluate.
		0	No effect.
		1	Causes EDMA3CC completion interrupt to be pulsed, if any enabled (IER*n* = 1) interrupts are still pending (IPR*n* = 1). The EDMA3CC completion region interrupt that is pulsed depends on which IEVAL is being exercised. For example, writing to the EVAL bit in IEVAL0 pulses the region 0 completion interrupt, but writing to the EVAL bit in IEVAL1 pulses the region 1 completion interrupt.

Copyright © 2013–2016, Texas Instruments Incorporated

16.4.2.7 QDMA Channel Registers

The following registers pertain to the 8 QDMA channels. The 8 QDMA channels consist of registers (with the exception of QDMAQNUM) that each have 8 bits and the bit position of each register matches the QDMA channel number.

The QDMA channel registers are accessible via read/writes to the global address range. They are also accessible via read/writes to the shadow address range. The read/write ability to the registers in the shadow region is controlled by the QDMA region access registers (QRAE*m*). These registers are described in Section 16.4.2.3.2 and the details for shadow region/global region usage is explained in Section 16.2.7.

16.4.2.7.1 QDMA Event Register (QER)

The QDMA event register (QER) channel *n* bit is set (E*n* = 1) when the CPU or any EDMA programmer (including EDMA3) performs a write to the trigger word (using the QDMA channel *n* mapping register (QCHMAP*n*)) in the PaRAM entry associated with QDMA channel *n* (which is also programmed using QCHMAP*n*). The E*n* bit is also set when the EDMA3CC performs a link update on a PaRAM address that matches the QCHMAP*n* settings. The QDMA event is latched only if the QDMA event enable register (QEER) channel *n* bit is also enabled (QEER.E*n* = 1). Once a bit is set in QER, then the corresponding QDMA event (auto-trigger) is evaluated by the EDMA3CC logic for an associated transfer request submission to the transfer controllers.

The setting of an event is a higher priority relative to clear operations (via hardware). If set and clear conditions occur concurrently, the set condition wins. If the event was previously set, then the QDMA event missed register (QEMR) would be set because an event is lost. If the event was previously clear, then the event remains set and is prioritized for submission to the event queues.

The set bits in QER are only cleared when the transfer request associated with the corresponding channels has been processed by the EDMA3CC and submitted to the transfer controller. If the E*n* bit is already set and a QDMA event for the same QDMA channel occurs prior to the original being cleared, then the second missed event is latched in QEMR (E*n* =1).

The QER is shown in Figure 16-76 and described in Table 16-55.

Figure 16-76. QDMA Event Register (QER)

31	16
Reserved	
R-0	

15	8	7	6	5	4	3	2	1	0
Reserved		E7	E6	E5	E4	E3	E2	E1	E0
R-0		R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0

LEGEND: R = Read only; -*n* = value after reset

Table 16-55. QDMA Event Register (QER) Field Descriptions

Bit	Field	Value	Description
31-8	Reserved	0	Reserved
7-0	E*n*		QDMA event for channels 0-7.
		0	No effect.
		1	Corresponding QDMA event is prioritized versus other pending DMA/QDMA events for submission to the EDMA3TC.

Copyright © 2013–2016, Texas Instruments Incorporated

16.4.2.7.2 QDMA Event Enable Register (QEER)

The EDMA3CC provides the option of selectively enabling/disabling each channel in the QDMA event register (QER) by using the QDMA event enable register (QEER). If any of the event bits in QEER is set to 1 (using the QDMA event enable set register, QEESR), it will enable that corresponding event. Alternatively, if any event bit in QEER is cleared (using the QDMA event enable clear register, QEECR), it will disable the corresponding QDMA channel. The QDMA event register will not latch any event for a QDMA channel, if it is not enabled via QEER.

The QEER is shown in Figure 16-77 and described in Table 16-56.

Figure 16-77. QDMA Event Enable Register (QEER)

31	16
Reserved	
R-0	

15	8	7	6	5	4	3	2	1	0
Reserved		E7	E6	E5	E4	E3	E2	E1	E0
R-0		R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0

LEGEND: R = Read only; -*n* = value after reset

Table 16-56. QDMA Event Enable Register (QEER) Field Descriptions

Bit	Field	Value	Description
31-8	Reserved	0	Reserved
7-0	E*n*		QDMA event enable for channels 0-7.
		0	QDMA channel *n* is not enabled. QDMA event will not be recognized and will not latch in the QDMA event register (QER).
		1	QDMA channel *n* is enabled. QDMA events will be recognized and will get latched in the QDMA event register (QER).

Copyright © 2013–2016, Texas Instruments Incorporated

16.4.2.7.3 QDMA Event Enable Clear Register (QEECR)

The QDMA event enable register (QEER) cannot be modified by directly writing to the register, in order to ease the software burden when multiple tasks are attempting to simultaneously modify these registers. The QDMA event enable clear register (QEECR) is used to disable events. Writes of 1 to the bits in QEECR clear the corresponding QDMA channel bits in QEER; writes of 0 have no effect.

The QEECR is shown in Figure 16-78 and described in Table 16-57.

Figure 16-78. QDMA Event Enable Clear Register (QEECR)

31	16
Reserved	
R-0	

15	8	7	6	5	4	3	2	1	0
Reserved		E7	E6	E5	E4	E3	E2	E1	E0
R-0		W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0

LEGEND: R = Read only; W = Write only; -*n* = value after reset

Table 16-57. QDMA Event Enable Clear Register (QEECR) Field Descriptions

Bit	Field	Value	Description
31-8	Reserved	0	Reserved
7-0	E*n*		QDMA event enable clear for channels 0-7.
		0	No effect.
		1	QDMA event is disabled. Corresponding bit in the QDMA event enable register (QEER) is cleared (E*n* = 0).

16.4.2.7.4 QDMA Event Enable Set Register (QEESR)

The QDMA event enable register (QEER) cannot be modified by directly writing to the register, in order to ease the software burden when multiple tasks are attempting to simultaneously modify these registers. The QDMA event enable set register (QEESR) is used to enable events. Writes of 1 to the bits in QEESR set the corresponding QDMA channel bits in QEER; writes of 0 have no effect.

The QEESR is shown in Figure 16-79 and described in Table 16-58.

Figure 16-79. QDMA Event Enable Set Register (QEESR)

31	16
Reserved	
R-0	

15	8	7	6	5	4	3	2	1	0
Reserved		E7	E6	E5	E4	E3	E2	E1	E0
R-0		W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0

LEGEND: R = Read only; W = Write only; -*n* = value after reset

Table 16-58. QDMA Event Enable Set Register (QEESR) Field Descriptions

Bit	Field	Value	Description
31-8	Reserved	0	Reserved
7-0	E*n*		QDMA event enable set for channels 0-7.
		0	No effect.
		1	QDMA event is enabled. Corresponding bit in the QDMA event enable register (QEER) is set (E*n* = 1).

Copyright © 2013–2016, Texas Instruments Incorporated

16.4.2.7.5 QDMA Secondary Event Register (QSER)

The QDMA secondary event register (QSER) provides information on the state of a QDMA event. If at any time a bit corresponding to a QDMA channel is set in QSER, that implies that the corresponding QDMA event is in the queue. Once a bit corresponding to a QDMA channel is set in QSER, the EDMA3CC does not prioritize additional events on the same QDMA channel. Depending on the condition that lead to the setting of the QSER bits, either the EDMA3CC hardware or the software (using QSECR) needs to clear the QSER bits for the EDMA3CC to evaluate subsequent QDMA events on the channel. Based on whether the associated TR is valid, or it is a null or dummy TR, the implications on the state of QSER and the required user action in order to submit another QDMA transfer might be different.

The QSER is shown in Figure 16-80 and described in Table 16-59.

Figure 16-80. QDMA Secondary Event Register (QSER)

31	16
Reserved	
R-0	

15	8	7	6	5	4	3	2	1	0
Reserved		E7	E6	E5	E4	E3	E2	E1	E0
R-0		R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0

LEGEND: R = Read only; -*n* = value after reset

Table 16-59. QDMA Secondary Event Register (QSER) Field Descriptions

Bit	Field	Value	Description
31-8	Reserved	0	Reserved
7-0	E*n*		QDMA secondary event register for channels 0-7.
		0	QDMA event is not currently stored in the event queue.
		1	QDMA event is currently stored in event queue. EDMA3CC will not prioritize additional events.

Copyright © 2013–2016, Texas Instruments Incorporated

16.4.2.7.6 QDMA Secondary Event Clear Register (QSECR)

The QDMA secondary event clear register (QSECR) clears the status of the QDMA secondary event register (QSER) and the QDMA event register (QER). CPU writes of 1 clear the corresponding set bits in QSER and QER. Writes of 0 have no effect. Note that this differs from the secondary event clear register (SECR) operation, which only clears the secondary event register (SER) bits and does not affect the event registers.

The QSECR is shown in Figure 16-81 and described in Table 16-60.

Figure 16-81. QDMA Secondary Event Clear Register (QSECR)

31	16
Reserved	
R-0	

15	8	7	6	5	4	3	2	1	0
Reserved		E7	E6	E5	E4	E3	E2	E1	E0
R-0		W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0

LEGEND: R = Read only; W = Write only; -*n* = value after reset

Table 16-60. QDMA Secondary Event Clear Register (QSECR) Field Descriptions

Bit	Field	Value	Description
31-8	Reserved	0	Reserved
7-0	E*n*		QDMA secondary event clear register for channels 0-7.
		0	No effect.
		1	Corresponding bit in the QDMA secondary event register (QSER) and the QDMA event register (QER) is cleared (E*n* = 0).

Copyright © 2013–2016, Texas Instruments Incorporated

16.4.3 EDMA3 Transfer Controller (EDMA3TC) Registers

Table 16-61 lists the memory-mapped registers for the EDMA3 transfer controller (EDMA3TC). See your device-specific data manual for the memory address of these registers. All other register offset addresses not listed in Table 16-61 should be considered as reserved locations and the register contents should not be modified.

Table 16-61. EDMA3 Transfer Controller (EDMA3TC) Registers

Offset	Acronym	Register Description	Section
0h	REVID	Revision Identification Register	Section 16.4.3.1
4h	TCCFG	EDMA3TC Configuration Register	Section 16.4.3.2
100h	TCSTAT	EDMA3TC Channel Status Register	Section 16.4.3.3
120h	ERRSTAT	Error Status Register	Section 16.4.3.4.1
124h	ERREN	Error Enable Register	Section 16.4.3.4.2
128h	ERRCLR	Error Clear Register	Section 16.4.3.4.3
12Ch	ERRDET	Error Details Register	Section 16.4.3.4.4
130h	ERRCMD	Error Interrupt Command Register	Section 16.4.3.4.5
140h	RDRATE	Read Command Rate Register	Section 16.4.3.5
240h	SAOPT	Source Active Options Register	Section 16.4.3.6.1
244h	SASRC	Source Active Source Address Register	Section 16.4.3.6.2
248h	SACNT	Source Active Count Register	Section 16.4.3.6.3
24Ch	SADST	Source Active Destination Address Register	Section 16.4.3.6.4
250h	SABIDX	Source Active B-Index Register	Section 16.4.3.6.5
254h	SAMPPRXY	Source Active Memory Protection Proxy Register	Section 16.4.3.6.6
258h	SACNTRLD	Source Active Count Reload Register	Section 16.4.3.6.7
25Ch	SASRCBREF	Source Active Source Address B-Reference Register	Section 16.4.3.6.8
260h	SADSTBREF	Source Active Destination Address B-Reference Register	Section 16.4.3.6.9
280h	DFCNTRLD	Destination FIFO Set Count Reload Register	Section 16.4.3.6.10
284h	DFSRCBREF	Destination FIFO Set Source Address B-Reference Register	Section 16.4.3.6.11
288h	DFDSTBREF	Destination FIFO Set Destination Address B-Reference Register	Section 16.4.3.6.12
300h	DFOPT0	Destination FIFO Options Register 0	Section 16.4.3.6.13
304h	DFSRC0	Destination FIFO Source Address Register 0	Section 16.4.3.6.14
308h	DFCNT0	Destination FIFO Count Register 0	Section 16.4.3.6.15
30Ch	DFDST0	Destination FIFO Destination Address Register 0	Section 16.4.3.6.16
310h	DFBIDX0	Destination FIFO B-Index Register 0	Section 16.4.3.6.17
314h	DFMPPRXY0	Destination FIFO Memory Protection Proxy Register 0	Section 16.4.3.6.18
340h	DFOPT1	Destination FIFO Options Register 1	Section 16.4.3.6.13
344h	DFSRC1	Destination FIFO Source Address Register 1	Section 16.4.3.6.14
348h	DFCNT1	Destination FIFO Count Register 1	Section 16.4.3.6.15
34Ch	DFDST1	Destination FIFO Destination Address Register 1	Section 16.4.3.6.16
350h	DFBIDX1	Destination FIFO B-Index Register 1	Section 16.4.3.6.17
354h	DFMPPRXY1	Destination FIFO Memory Protection Proxy Register 1	Section 16.4.3.6.18
380h	DFOPT2	Destination FIFO Options Register 2	Section 16.4.3.6.13
384h	DFSRC2	Destination FIFO Source Address Register 2	Section 16.4.3.6.14
388h	DFCNT2	Destination FIFO Count Register 2	Section 16.4.3.6.15
38Ch	DFDST2	Destination FIFO Destination Address Register 2	Section 16.4.3.6.16
390h	DFBIDX2	Destination FIFO B-Index Register 2	Section 16.4.3.6.17
394h	DFMPPRXY2	Destination FIFO Memory Protection Proxy Register 2	Section 16.4.3.6.18
3C0h	DFOPT3	Destination FIFO Options Register 3	Section 16.4.3.6.13
3C4h	DFSRC3	Destination FIFO Source Address Register 3	Section 16.4.3.6.14
3C8h	DFCNT3	Destination FIFO Count Register 3	Section 16.4.3.6.15

Copyright © 2013–2016, Texas Instruments Incorporated

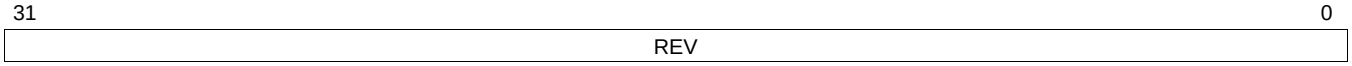
Table 16-61. EDMA3 Transfer Controller (EDMA3TC) Registers (continued)

Offset	Acronym	Register Description	Section
3CCh	DFDST3	Destination FIFO Destination Address Register 3	Section 16.4.3.6.16
3D0h	DFBIDX3	Destination FIFO B-Index Register 3	Section 16.4.3.6.17
3D4h	DFMPPRXY3	Destination FIFO Memory Protection Proxy Register 3	Section 16.4.3.6.18

16.4.3.1 Revision Identification Register (REVID)

The revision identification register (REVID) is a constant register that uniquely identifies the EDMA3TC and specific revision of the EDMA3TC. The REVID is shown in Figure 16-82 and described in Table 16-62.

Figure 16-82. Revision ID Register (REVID)



R-4000 3B00h

LEGEND: R = Read only; -*n* = value after reset

Table 16-62. Revision ID Register (REVID) Field Descriptions

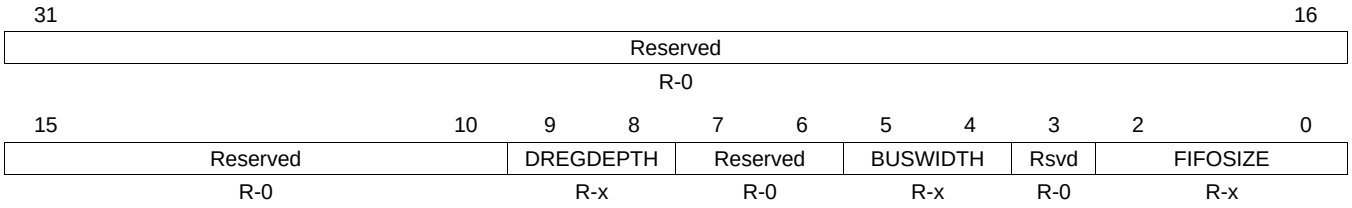
Bit	Field	Value	Description
31-0	REV		Peripheral identifier.
		4000 3B00h	Uniquely identifies the EDMA3TC and the specific revision of the EDMA3TC.

Copyright © 2013–2016, Texas Instruments Incorporated

16.4.3.2 EDMA3TC Configuration Register (TCCFG)

The EDMA3TC configuration register (TCCFG) is shown in Figure 16-83 and described in Table 16-63.

Figure 16-83. EDMA3TC Configuration Register (TCCFG)



LEGEND: R = Read only; -*n* = value after reset; -x = value is indeterminate after reset

Table 16-63. EDMA3TC Configuration Register (TCCFG) Field Descriptions

Bit	Field	Value	Description
31-10	Reserved	0	Reserved
9-8	DREGDEPTH	0-3h	Destination register FIFO depth parameterization.
		0	1 entry
		1h	2 entry
		2h	4 entry (for EDMA3TC0 and EDMA3TC1)
		3h	Reserved
7-6	Reserved	0	Reserved
5-4	BUSWIDTH	0-3h	Bus width parameterization.
		0	32-bit
		1h	64-bit (for EDMA3TC0 and EDMA3TC1)
		2h-3h	Reserved
3	Reserved	0	Reserved
2-0	FIFOSIZE	0-7h	FIFO size.
		0	32-byte FIFO
		1h	64-byte FIFO
		2h	128-byte FIFO (for EDMA3TC0 and EDMA3TC1)
		3h	256-byte FIFO
		4h-7h	Reserved

Copyright © 2013–2016, Texas Instruments Incorporated

16.4.3.3 EDMA3TC Channel Status Register (TCSTAT)

The EDMA3TC channel status register (TCSTAT) is shown in Figure 16-84 and described in Table 16-64.

Figure 16-84. EDMA3TC Channel Status Register (TCSTAT)

31	16
Reserved	
R-0	

15	13	12	11	10	8
Reserved		DFSTRTPTR		Reserved	
R-0		R-0		R-0	

7	6	4	3	2	1	0
Reserved	DSTACTV		Reserved	WSACTV	SRCACTV	PROGBUSY
R-0	R-0		R-0	R-0	R-0	R-0

LEGEND: R = Read only; -*n* = value after reset

Table 16-64. EDMA3TC Channel Status Register (TCSTAT) Field Descriptions

Bit	Field	Value	Description
31-13	Reserved	0	Reserved
12-11	DFSTRTPTR	0-3h	Destination FIFO start pointer. The offset to the head entry of the destination register FIFO, in units of *entries*.
10-7	Reserved	0	Reserved
6-4	DSTACTV	0-7h	Destination active state. Specifies the number of transfer requests (TRs) that are resident in the destination register FIFO at a given instant. This bit field can be primarily used for advanced debugging.
		0	Destination FIFO is empty.
		1h	Destination FIFO contains 1 TR.
		2h	Destination FIFO contains 2 TR.
		3h	Destination FIFO contains 3 TR.
		4h	Destination FIFO contains 4 TR. (Full if DSTREGDEPTH == 4)
			If the destination register FIFO is empty, then any TR written to Prog Set immediately transitions to the destination register FIFO. If the destination register FIFO is not empty and not full, then any TR written to Prog Set immediately transitions to the destination register FIFO set if the source active state (SRCACTV) bit is set to idle.
			If the destination register FIFO is full, then TRs cannot transition to the destination register FIFO. The destination register FIFO becomes not full when the TR at the head of the destination register FIFO is completed.
		5h-7h	Reserved
3	Reserved	0	Reserved
2	WSACTV		Write status active.
		0	Write status is not pending. Write status has been received for all previously issued write commands.
		1	Write status is pending. Write status has not been received for all previously issued write commands.
1	SRCACTV		Source active state.
		0	Source active set is idle and is available for programming by the EDMA3CC. Source active register set contains a previously processed transfer request.
		1	Source active set is busy servicing a transfer request.
0	PROGBUSY		Program register set busy.
		0	Program set idle and is available for programming by the EDMA3CC.
		1	Program set busy.

Copyright © 2013–2016, Texas Instruments Incorporated

16.4.3.4 Error Registers

16.4.3.4.1 Error Status Register (ERRSTAT)

The error status register (ERRSTAT) is shown in Figure 16-85 and described in Table 16-65.

Figure 16-85. Error Status Register (ERRSTAT)

31	16
Reserved	
R-0	

15	4	3	2	1	0
Reserved		MMRAERR	TRERR	Reserved	BUSERR
R-0		R-0	R-0	R-0	R-0

LEGEND: R = Read only; -*n* = value after reset

Table 16-65. Error Status Register (ERRSTAT) Field Descriptions

Bit	Field	Value	Description
31-4	Reserved	0	Reserved
3	MMRAERR		MMR address error.
		0	MMR address error is not detected.
		1	User attempted to read or write to an invalid address in configuration memory map.
2	TRERR		Transfer request (TR) error event.
		0	Transfer request (TR) error is not detected.
		1	Transfer request (TR) detected that violates constant addressing mode transfer (SAM or DAM is set to 1) alignment rules or has ACNT or BCNT == 0.
1	Reserved	0	Reserved
0	BUSERR		Bus error event.
		0	Bus error is not detected.
		1	EDMA3TC has detected an error at source or destination address. Error information can be read from the error details register (ERRDET).

Copyright © 2013–2016, Texas Instruments Incorporated

16.4.3.4.2 Error Enable Register (ERREN)

The error enable register (ERREN) is shown in Figure 16-86 and described in Table 16-66. When any of the enable bits in ERREN is set, a bit set in the corresponding error status register (ERRSTAT) causes an assertion of the EDMA3TC interrupt.

Figure 16-86. Error Enable Register (ERREN)

31	16
Reserved	
R-0	

15	4	3	2	1	0
Reserved		MMRAERR	TRERR	Reserved	BUSERR
R-0		R/W-0	R/W-0	R/W-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 16-66. Error Enable Register (ERREN) Field Descriptions

Bit	Field	Value	Description
31-4	Reserved	0	Reserved
3	MMRAERR		Interrupt enable for MMR address error (MMRAERR).
		0	MMRAERR is disabled.
		1	MMRAERR is enabled and contributes to the state of EDMA3TC error interrupt generation
2	TRERR		Interrupt enable for transfer request error (TRERR).
		0	TRERR is disabled.
		1	TRERR is enabled and contributes to the state of EDMA3TC error interrupt generation.
1	Reserved	0	Reserved. Always write 0 to this bit; writes of 1 to this bit are not supported and attempts to do so may result in undefined behavior.
0	BUSERR		Interrupt enable for bus error (BUSERR).
		0	BUSERR is disabled.
		1	BUSERR is enabled and contributes to the state of EDMA3TC error interrupt generation.

Copyright © 2013–2016, Texas Instruments Incorporated

16.4.3.4.3 Error Clear Register (ERRCLR)

The error clear register (ERRCLR) is shown in Figure 16-87 and described in Table 16-67.

Figure 16-87. Error Clear Register (ERRCLR)

31	16
Reserved	
R-0	

15	4	3	2	1	0
Reserved		MMRAERR	TRERR	Reserved	BUSERR
R-0		W-0	W-0	R-0	W-0

LEGEND: R = Read only; W = Write only; -*n* = value after reset

Table 16-67. Error Clear Register (ERRCLR) Field Descriptions

Bit	Field	Value	Description
31-4	Reserved	0	Reserved
3	MMRAERR		Interrupt enable clear for the MMR address error (MMRAERR) bit in the error status register (ERRSTAT).
		0	No effect.
		1	Clears the MMRAERR bit in the error status register (ERRSTAT) but does not clear the error details register (ERRDET).
2	TRERR		Interrupt enable clear for the transfer request error (TRERR) bit in the error status register (ERRSTAT).
		0	No effect.
		1	Clears the TRERR bit in the error status register (ERRSTAT) but does not clear the error details register (ERRDET).
1	Reserved	0	Reserved
0	BUSERR		Interrupt clear for the bus error (BUSERR) bit in the error status register (ERRSTAT).
		0	No effect.
		1	Clears the BUSERR bit in the error status register (ERRSTAT) and clears the error details register (ERRDET).

Copyright © 2013–2016, Texas Instruments Incorporated

16.4.3.4.4 Error Details Register (ERRDET)

The error details register (ERRDET) is shown in Figure 16-88 and described in Table 16-68.

Figure 16-88. Error Details Register (ERRDET)

31	18	17	16
Reserved		TCCHEN	TCINTEN
R-0		R-0	R-0

15	14	13	8	7	4	3	0
Reserved		TCC		Reserved		STAT	
R-0		R-0		R-0		R-0	

LEGEND: R = Read only; -*n* = value after reset

Table 16-68. Error Details Register (ERRDET) Field Descriptions

Bit	Field	Value	Description
31-8	Reserved	0	Reserved
17	TCCHEN	0-1	Transfer completion chaining enable. Contains the TCCHEN value in the channel options parameter (OPT) programmed by the channel controller for the read or write transaction that resulted in an error.
16	TCINTEN	0-1	Transfer completion interrupt enable. Contains the TCINTEN value in the channel options parameter (OPT) programmed by the channel controller for the read or write transaction that resulted in an error.
15-14	Reserved	0	Reserved
13 - 8	TCC	0-3Fh	Transfer complete code. Contains the TCC value in the channel options parameter (OPT) programmed by the channel controller for the read or write transaction that resulted in an error.
7-4	Reserved	0	Reserved
3-0	STAT	0-Fh	Transaction status. Stores the nonzero status/error code that was detected on the read status or write status bus. If read status and write status are returned on the same cycle, then the EDMA3TC chooses nonzero version. If both are nonzero, then the write status is treated as higher priority.
		0	No error
		1h	Read addressing error
		2h	Read privilege error
		3h	Read timeout error
		4h	Read data error
		5h-6h	Reserved
		7h	Read exclusive operation error
		8h	Reserved
		9h	Write addressing error
		Ah	Write privilege error
		Bh	Write timeout error
		Ch	Write data error
		Dh-Eh	Reserved
		Fh	Write exclusive operation error

Copyright © 2013–2016, Texas Instruments Incorporated

16.4.3.4.5 Error Interrupt Command Register (ERRCMD)

The error interrupt command register (ERRCMD) is shown in Figure 16-89 and described in Table 16-69.

Figure 16-89. Error Interrupt Command Register (ERRCMD)

31	16
Reserved	
R-0	

15	1	0
Reserved		EVAL
R-0		W-0

LEGEND: R = Read only; W = Write only; -*n* = value after reset

Table 16-69. Error Interrupt Command Register (ERRCMD) Field Descriptions

Bit	Field	Value	Description
31-1	Reserved	0	Reserved
0	EVAL		Error evaluate.
		0	No effect.
		1	EDMA3TC error line is pulsed if any of the error status register (ERRSTAT) bits are set to 1.

Copyright © 2013–2016, Texas Instruments Incorporated

16.4.3.5 Read Command Rate Register (RDRATE)

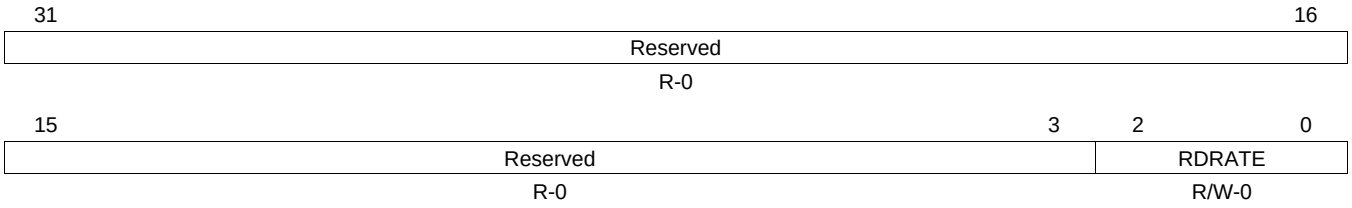
The EDMA3 transfer controller issues Read commands at a rate controlled by the Read command rate register (RDRATE). The RDRATE defines the number of idle cycles that the Read controller must wait before issuing subsequent commands. This applies both to commands within a transfer request packet (TRP) and for commands that are issued for different transfer requests (TRs). For instance, if RDRATE is set to 4 cycles between reads, there are 32 inactive cycles between reads.

RDRATE allows flexibility in transfer controller access requests to an endpoint. For an application, RDRATE can be manipulated to slow down the access rate, so that the endpoint may service requests from other masters during the inactive EDMA3TC cycles.

The RDRATE is shown in Figure 16-90 and described in Table 16-70.

NOTE: It is expected that the RDRATE value for a transfer controller is static, as it is decided based on the application requirement. It is not recommended to change this setting on the go.

Figure 16-90. Read Command Rate Register (RDRATE)



LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 16-70. Read Command Rate Register (RDRATE) Field Descriptions

Bit	Field	Value	Description
31-3	Reserved	0	Reserved
2-0	RDRATE	0-7h	Read rate. Controls the number of cycles between Read commands. This is a global setting that applies to all TRs for this EDMA3TC.
		0	Reads issued as fast as possible.
		1h	4 EDMA3TC cycles between reads.
		2h	8 EDMA3TC cycles between reads.
		3h	16 EDMA3TC cycles between reads.
		4h	32 EDMA3TC cycles between reads.
		5h-7h	Reserved

Copyright © 2013–2016, Texas Instruments Incorporated

16.4.3.6 EDMA3TC Channel Registers

The EDMA3TC channel registers are split into three parts: the programming registers, the source active registers, and the destination FIFO registers. This section describes the registers and their functions. The program register set is programmed by the channel controller and is for internal use. The source active registers and the destination FIFO registers are read-only and are provided to facilitate advanced debug capabilities. The number of destination FIFO register sets depends on the destination FIFO depth. Both TC0 and TC1 have a destination FIFO depth of 4, and there are four sets of destination FIFO registers.

16.4.3.6.1 Source Active Options Register (SAOPT)

The source active options register (SAOPT) is shown in Figure 16-91 and described in Table 16-71.

Figure 16-91. Source Active Options Register (SAOPT)

31	23	22	21	20	19	18	17	16
Reserved		TCCHEN	Rsvd	TCINTEN	Reserved		TCC	
R-0		R/W-0	R-0	R/W-0	R-0		R/W-0	

15	12	11	10	8	7	6	4	3	2	1	0
TCC		Rsvd	FWID		Rsvd	PRI[1]		Reserved		DAM	SAM
R/W-0		R-0	R/W-0		R-0	R/W-0		R-0		R/W-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

(1) On previous architectures, the EDMA3TC priority was controlled by the queue priority register (QUEPRI) in the EDMA3CC memory-map. However for this device, the priority control for the transfer controllers is controlled by the chip-level registers in the System Configuration Module. You should use the chip-level registers and not QUEPRI to configure the TC priority.

Table 16-71. Source Active Options Register (SAOPT) Field Descriptions

Bit	Field	Value	Description
31-23	Reserved	0	Reserved
22	TCCHEN		Transfer complete chaining enable.
		0	Transfer complete chaining is disabled.
		1	Transfer complete chaining is enabled.
21	Reserved	0	Reserved
20	TCINTEN		Transfer complete interrupt enable.
		0	Transfer complete interrupt is disabled.
		1	Transfer complete interrupt is enabled.
19-18	Reserved	0	Reserved
17-12	TCC	0-3Fh	Transfer complete code. This 6-bit code is used to set the relevant bit in CER or IPR of the EDMA3CC.
11	Reserved	0	Reserved
10-8	FWID	0-7h	FIFO width. Applies if either SAM or DAM is set to constant addressing mode.
		0	FIFO width is 8 bits.
		1h	FIFO width is 16 bits.
		2h	FIFO width is 32 bits.
		3h	FIFO width is 64 bits.
		4h	FIFO width is 128 bits.
		5h-7h	Reserved
7	Reserved	0	Reserved
6-4	PRI	0-7h	Transfer priority. Reflects the values programmed in the queue priority register (QUEPRI) in the EDMA3CC.
		0	Priority 0 - Highest priority
		1h-6h	Priority 1 to priority 6
		7h	Priority 7 - Lowest priority
3-2	Reserved	0	Reserved

Copyright © 2013–2016, Texas Instruments Incorporated

Table 16-71. Source Active Options Register (SAOPT) Field Descriptions (continued)

Bit	Field	Value	Description
1	DAM		Destination address mode within an array.
		0	Increment (INCR) mode. Destination addressing within an array increments.
		1	Constant addressing (CONST) mode. Destination addressing within an array wraps around upon reaching FIFO width.
0	SAM		Source address mode within an array.
		0	Increment (INCR) mode. Source addressing within an array increments.
		1	Constant addressing (CONST) mode. Source addressing within an array wraps around upon reaching FIFO width.

16.4.3.6.2 Source Active Source Address Register (SASRC)

The source active source address register (SASRC) is shown in Figure 16-92 and described in Table 16-72.

Figure 16-92. Source Active Source Address Register (SASRC)

31	0
SADDR	
R-0	

LEGEND: R = Read only; -*n* = value after reset

Table 16-72. Source Active Source Address Register (SASRC) Field Descriptions

Bit	Field	Value	Description
31-0	SADDR	0-FFFF FFFFh	Source address for program register set. EDMA3TC updates value according to source addressing mode (SAM bit in the source active options register, SAOPT) .

16.4.3.6.3 Source Active Count Register (SACNT)

The source active count register (SACNT) is shown in Figure 16-93 and described in Table 16-73.

Figure 16-93. Source Active Count Register (SACNT)

31	16
BCNT	
R-0	
15	0
ACNT	
R-0	

LEGEND: R = Read only; -*n* = value after reset

Table 16-73. Source Active Count Register (SACNT) Field Descriptions

Bit	Field	Value	Description
31-16	BCNT	0-FFFFh	B dimension count. Number of arrays to be transferred, where each array is ACNT in length. It is decremented after each Read command appropriately. Represents the amount of data remaining to be Read. It should be 0 when transfer request (TR) is complete.
15-0	ACNT	0-FFFFh	A dimension count. Number of bytes to be transferred in first dimension. It is decremented after each Read command appropriately. Represents the amount of data remaining to be Read. It should be 0 when transfer request (TR) is complete.

Copyright © 2013–2016, Texas Instruments Incorporated

16.4.3.6.4 Source Active Destination Address Register (SADST)

The source active destination address register (SADST) is shown in Figure 16-94 and described in Table 16-74.

Figure 16-94. Source Active Destination Address Register (SADST)

31	0
DADDR	
R-0	

LEGEND: R = Read only; -*n* = value after reset

Table 16-74. Source Active Destination Address Register (SADST) Field Descriptions

Bit	Field	Value	Description
31-0	DADDR	0	Always reads as 0

16.4.3.6.5 Source Active B-Index Register (SABIDX)

The source active B-index register (SABIDX) is shown in Figure 16-95 and described in Table 16-75.

Figure 16-95. Source Active B-Index Register (SABIDX)

31	16
DSTBIDX	
R-0	
15	0
SRCBIDX	
R-0	

LEGEND: R = Read only; -*n* = value after reset

Table 16-75. Source Active B-Index Register (SABIDX) Field Descriptions

Bit	Field	Value	Description
31-16	DSTBIDX	0	B-Index offset between destination arrays. Represents the offset in bytes between the starting address of each destination. Always reads as 0.
15-0	SRCBIDX	0-FFFFh	B-Index offset between source arrays. Represents the offset in bytes between the starting address of each source array.

Copyright © 2013–2016, Texas Instruments Incorporated

16.4.3.6.6 Source Active Memory Protection Proxy Register (SAMPPRXY)

The source active memory protection proxy register (SAMPPRXY) is shown in Figure 16-96 and described in Table 16-76.

Figure 16-96. Source Active Memory Protection Proxy Register (SAMPPRXY)

31	16
Reserved	
R-0	

15	9	8	7	4	3	0
Reserved		PRIV	Reserved		PRIVID	
R-0		R-0	R-0		R-0	

LEGEND: R = Read only; -*n* = value after reset

Table 16-76. Source Active Memory Protection Proxy Register (SAMPPRXY) Field Descriptions

Bit	Field	Value	Description
31-9	Reserved	0	Reserved
8	PRIV		Privilege level. The privilege level used by the host to set up the parameter entry in the channel controller. This field is set up when the associated TR is submitted to the EDMA3TC. The privilege ID is used while issuing Read and write command to the target endpoints so that the target endpoints can perform memory protection checks based on the PRIV of the host that set up the DMA transaction.
		0	User-level privilege
		1	Supervisor-level privilege
7-4	Reserved	0	Reserved
3-0	PRIVID	0-Fh	Privilege ID. This contains the privilege ID of the host that set up the parameter entry in the channel controller. This field is set up when the associated TR is submitted to the EDMA3TC. This PRIVID value is used while issuing Read and write commands to the target endpoints so that the target endpoints can perform memory protection checks based on the PRIVID of the host that set up the DMA transaction.
		0	For any other master that sets up the PaRAM entry.
		1	If DSP sets up the PaRAM entry.

Copyright © 2013–2016, Texas Instruments Incorporated

16.4.3.6.7 Source Active Count Reload Register (SACNTRLD)

The source active count reload register (SACNTRLD) is shown in Figure 16-97 and described in Table 16-77.

Figure 16-97. Source Active Count Reload Register (SACNTRLD)

31	16
Reserved	
R-0	

15	0
ACNTRLD	
R-0	

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 16-77. Source Active Count Reload Register (SACNTRLD) Field Descriptions

Bit	Field	Value	Description
31-16	Reserved	0	Reserved
15-0	ACNTRLD	0-FFFFh	A-count reload value. Represents the originally programmed value of ACNT. The reload value is used to reinitialize ACNT after each array is serviced.

16.4.3.6.8 Source Active Source Address B-Reference Register (SASRCBREF)

The source active source address B-reference register (SASRCBREF) is shown in Figure 16-98 and described in Table 16-78.

Figure 16-98. Source Active Source Address B-Reference Register (SASRCBREF)

31	0
SADDRBREF	
R-0	

LEGEND: R = Read only; -*n* = value after reset

Table 16-78. Source Active Source Address B-Reference Register (SASRCBREF) Field Descriptions

Bit	Field	Value	Description
31-0	SADDRBREF	0-FFFF FFFFh	Source address B-reference. Represents the starting address for the array currently being Read.

Copyright © 2013–2016, Texas Instruments Incorporated

16.4.3.6.9 Source Active Destination Address B-Reference Register (SADSTBREF)

The source active destination address B-reference register (SADSTBREF) is shown in Figure 16-99 and described in Table 16-79.

Figure 16-99. Source Active Destination Address B-Reference Register (SADSTBREF)

31	0
DADDRBREF	
R-0	

LEGEND: R = Read only; -*n* = value after reset

Table 16-79. Source Active Destination Address B-Reference Register (SADSTBREF) Field Descriptions

Bit	Field	Value	Description
31-0	DADDRBREF	0	Always reads as 0

16.4.3.6.10 Destination FIFO Set Count Reload Register (DFCNTRLD)

The destination FIFO set count reload register (DFCNTRLD) is shown in Figure 16-100 and described in Table 16-80.

Figure 16-100. Destination FIFO Set Count Reload Register (DFCNTRLD)

31	16
Reserved	
R-0	

15	0
ACNTRLD	
R-0	

LEGEND: R = Read only; -*n* = value after reset

Table 16-80. Destination FIFO Set Count Reload Register (DFCNTRLD) Field Descriptions

Bit	Field	Value	Description
31-16	Reserved	0	Reserved
15-0	ACNTRLD	0-FFFFh	A-count reload value. Represents the originally programmed value of ACNT. The reload value is used to reinitialize ACNT after each array is serviced.

Copyright © 2013–2016, Texas Instruments Incorporated

16.4.3.6.11 Destination FIFO Set Source Address B-Reference Register (DFSRCBREF)

The destination FIFO set source address B-reference register (DFSRCBREF) is shown in Figure 16-101 and described in Table 16-81.

Figure 16-101. Destination FIFO Set Source Address B-Reference Register (DFSRCBREF)

31	0
SADDRBREF	
R-0	

LEGEND: R = Read only; -*n* = value after reset

Table 16-81. Destination FIFO Set Source Address B-Reference Register (DFSRCBREF) Field Descriptions

Bit	Field	Value	Description
31-0	SADDRBREF	0	Not applicable. Always Read as 0.

16.4.3.6.12 Destination FIFO Set Destination Address B-Reference (DFDSTBREF)

The destination FIFO set destination address B-reference register (DFDSTBREF) is shown in Figure 16-102 and described in Table 16-82.

Figure 16-102. Destination FIFO Set Destination Address B-Reference Register (DFDSTBREF)

31	0
DADDRBREF	
R-0	

LEGEND: R = Read only; -*n* = value after reset

Table 16-82. Destination FIFO Set Destination Address B-Reference Register (DFDSTBREF) Field Descriptions

Bit	Field	Value	Description
31-0	DADDRBREF	0-FFFF FFFFh	Destination address reference for the destination FIFO register set. Represents the starting address for the array currently being written.

Copyright © 2013–2016, Texas Instruments Incorporated

16.4.3.6.13 Destination FIFO Options Register *n* (DFOPT*n*)

The destination FIFO options register *n* (DFOPT*n*) is shown in Figure 16-103 and described in Table 16-83.

Figure 16-103. Destination FIFO Options Register *n* (DFOPT*n*)

31	23	22	21	20	19	18	17	16
Reserved		TCCHEN	Rsvd	TCINTEN	Reserved		TCC	
R-0		R/W-0	R-0	R/W-0	R-0		R/W-0	

15	12	11	10	8	7	6	4	3	2	1	0
TCC		Rsvd	FWID		Rsvd	PRI		Reserved		DAM	SAM
R/W-0		R-0	R/W-0		R-0	R/W-0		R-0		R/W-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 16-83. Destination FIFO Options Register *n* (DFOPT*n*) Field Descriptions

Bit	Field	Value	Description
31-23	Reserved	0	Reserved
22	TCCHEN		Transfer complete chaining enable.
		0	Transfer complete chaining is disabled.
		1	Transfer complete chaining is enabled.
21	Reserved	0	Reserved
20	TCINTEN		Transfer complete interrupt enable.
		0	Transfer complete interrupt is disabled.
		1	Transfer complete interrupt is enabled.
19-18	Reserved	0	Reserved
17-12	TCC	0-3Fh	Transfer complete code. This 6-bit code is used to set the relevant bit in CER or IPR of the EDMA3CC.
11	Reserved	0	Reserved
10-8	FWID	0-7h	FIFO width. Applies if either SAM or DAM is set to constant addressing mode.
		0	FIFO width is 8 bits.
		1h	FIFO width is 16 bits.
		2h	FIFO width is 32 bits.
		3h	FIFO width is 64 bits.
		4h	FIFO width is 128 bits.
		5h-7h	Reserved
7	Reserved	0	Reserved
6-4	PRI	0-7h	Transfer priority.
		0	Priority 0 - Highest priority
		1h-6h	Priority 1 to priority 6
		7h	Priority 7 - Lowest priority
3-2	Reserved	0	Reserved
1	DAM		Destination address mode within an array.
		0	Increment (INCR) mode. Destination addressing within an array increments.
		1	Constant addressing (CONST) mode. Destination addressing within an array wraps around upon reaching FIFO width.
0	SAM		Source address mode within an array.
		0	Increment (INCR) mode. Source addressing within an array increments.
		1	Constant addressing (CONST) mode. Source addressing within an array wraps around upon reaching FIFO width.

Copyright © 2013–2016, Texas Instruments Incorporated

16.4.3.6.14 Destination FIFO Source Address Register *n* (DFSRC*n*)

The destination FIFO source address register *n* (DFSRC*n*) is shown in Figure 16-104 and described in Table 16-84.

Figure 16-104. Destination FIFO Source Address Register *n* (DFSRC*n*)

31	0
SADDR	
R-0	

LEGEND: R = Read only; -*n* = value after reset

Table 16-84. Destination FIFO Source Address Register *n* (DFSRC*n*) Field Descriptions

Bit	Field	Value	Description
31-0	SADDR	0	Always Read as 0.

16.4.3.6.15 Destination FIFO Count Register *n* (DFCNT*n*)

The destination FIFO count register *n* (DFCNT*n*) is shown in Figure 16-105 and described in Table 16-85.

Figure 16-105. Destination FIFO Count Register *n* (DFCNT*n*)

31	16
BCNT	
R-0	
15	0
ACNT	
R-0	

LEGEND: R = Read only; -*n* = value after reset

Table 16-85. Destination FIFO Count Register *n* (DFCNT*n*) Field Descriptions

Bit	Field	Value	Description
31-16	BCNT	0-FFFFh	B-dimension count. Number of arrays to be transferred, where each array is ACNT in length. Count/count remaining for destination register set. Represents the amount of data remaining to be written.
15-0	ACNT	0-FFFFh	A-dimension count. Number of bytes to be transferred in first dimension count/count remaining for destination register set. Represents the amount of data remaining to be written.

Copyright © 2013–2016, Texas Instruments Incorporated

16.4.3.6.16 Destination FIFO Destination Address Register *n* (DFDST*n*)

The destination FIFO destination address register *n* (DFDST*n*) is shown in Figure 16-106 and described in Table 16-86.

Figure 16-106. Destination FIFO Destination Address Register *n* (DFDST*n*)

31	0
DADDR	
R-0	

LEGEND: R = Read only; -*n* = value after reset

Table 16-86. Destination FIFO Destination Address Register *n* (DFDST*n*) Field Descriptions

Bit	Field	Value	Description
31-0	DADDR	0	Destination address for the destination FIFO register set. When a transfer request (TR) is complete, the final value should be the address of the last write command issued.

16.4.3.6.17 Destination FIFO B-Index Register *n* (DFBIDX*n*)

The destination FIFO B-index register *n* (DFBIDX*n*) is shown in Figure 16-107 and described in Table 16-87.

Figure 16-107. Destination FIFO B-Index Register *n* (DFBIDX*n*)

31	16
DSTBIDX	
R-0	
15	0
SRCBIDX	
R-0	

LEGEND: R = Read only; -*n* = value after reset

Table 16-87. Destination FIFO B-Index Register *n* (DFBIDX*n*) Field Descriptions

Bit	Field	Value	Description
31-16	DSTBIDX	0-FFFFh	B-Index offset between destination arrays. Represents the offset in bytes between the starting address of each destination.
15-0	SRCBIDX	0	B-Index offset between source arrays. Represents the offset in bytes between the starting address of each source array. Always Read as 0.

Copyright © 2013–2016, Texas Instruments Incorporated

16.4.3.6.18 Destination FIFO Memory Protection Proxy Register *n* (DFMPPRXY*n*)

The destination FIFO memory protection proxy register *n* (DFMPPRXY*n*) is shown in Figure 16-108 and described in Table 16-88.

Figure 16-108. Destination FIFO Memory Protection Proxy Register *n* (DFMPPRXY*n*)

31	16
Reserved	
R-0	

15	9	8	7	4	3	0
Reserved		PRIV	Reserved		PRIVID	
R-0		R-0	R-0		R-0	

LEGEND: R = Read only; -*n* = value after reset

Table 16-88. Destination FIFO Memory Protection Proxy Register *n* (DFMPPRXY*n*) Field Descriptions

Bit	Field	Value	Description
31-9	Reserved	0	Reserved
8	PRIV		Privilege level. This contains the privilege level used by the EDMA programmer to set up the parameter entry in the channel controller. This field is set up when the associated TR is submitted to the EDMA3TC. The privilege ID is used while issuing Read and write command to the target endpoints so that the target endpoints can perform memory protection checks based on the PRIV of the host that set up the DMA transaction.
		0	User-level privilege
		1	Supervisor-level privilege
7-4	Reserved	0	Reserved
3-0	PRIVID	0-Fh	Privilege ID. This contains the Privilege ID of the EDMA programmer that set up the parameter entry in the channel controller. This field is set up when the associated TR is submitted to the EDMA3TC. This PRIVID value is used while issuing Read and write commands to the target endpoints so that the target endpoints can perform memory protection checks based on the PRIVID of the host that set up the DMA transaction.
		0	For any other master that sets up the PaRAM entry
		1	If DSP sets up the PaRAM entry

Copyright © 2013–2016, Texas Instruments Incorporated

16.5 Tips

16.5.1 Debug Checklist

This section lists some tips to keep in mind while debugging applications using the EDMA3. Table 16-89 provides some common issues and their probable causes and resolutions.

Table 16-89. Debug List

Issue	Description/Solution
The transfer associated with the channel does not happen. The channel does not get serviced.	The EDMA3 channel controller (EDMA3CC) may not service a transfer request, even though the associated PaRAM set is programmed appropriately. Check for the following: 1) Verify that events are enabled, that is, if an external/peripheral event is latched in the event register (ER), make sure that the event is enabled in the event enable register (EER). Similarly for QDMA channels, make sure that QDMA events are appropriately enabled in the QDMA event enable register (QEER). 2) Verify that the DMA or QDMA secondary event register (SER) bits corresponding to the particular event or channel are not set.
The secondary event register bits are set, not allowing additional transfers to occur on a channel.	It is possible that a trigger event was received when the parameter set associated with the channel/event was a NULL set for a previous transfer on the channel. This is typical in two cases: 1) QDMA channels: Typically if the parameter set is nonstatic and expected to be terminated by a NULL set (OPT.STATIC = 0, LINK = FFFFh), the parameter set is updated with a NULL set after submission of the last TR. Because QDMA channels are autotriggered, this update caused the generation of an event. An event generated for a NULL set causes an error condition and results in setting the bits corresponding to the QDMA channel in QEMR and QSER. This will disable further prioritization of the channel. 2) DMA channels used in a continuous mode: The peripheral may be set up to continuously generate infinite events (for instance, in case of the McBSP, every time the data shifts out from DXR, it generates an XEVT). The parameter set may be programmed to expect only a finite number of events and to be terminated by a NULL link. After the expected number of events, the parameter set is reloaded with a NULL parameter set. Because the peripheral will generate additional events, an error condition is set in SER.E*n* and EMR.E*n*, preventing further event prioritization. You must ensure that the number of events received is limited to the expected number of events for which the parameter set is programmed, or you must ensure that bits corresponding to a particular channel or event are not set in the secondary event registers (SER/QSER) and the event missed registers (EMR/QEMR) before trying to perform subsequent transfers for the event/channel.
Completion interrupts are not asserted, or no further interrupts are received after the first completion interrupt.	You must ensure the following: 1) The interrupt generation is enabled in the OPT of the associated PaRAM set (TCINTEN = 1 and/or ITCINTEN = 1). 2) The interrupts are enabled in the EDMA3 channel controller (EDMA3CC), via the interrupt enable register (IER). 3) The corresponding interrupts are enabled in the device interrupt controller. 4) The set interrupts are cleared in the interrupt pending register (IPR) before exiting the transfer completion interrupt service routine (ISR). See Section 16.2.9.1.2 for details on writing EDMA3 ISRs. 5) If working with shadow region interrupts, make sure that the DMA region access enable registers (DRAE) are set up properly, because DRAE act as secondary enables for shadow region completion interrupts, along with IER. If working with shadow region interrupts, make sure that the bits corresponding to the transfer completion code (TCC) value are also enabled in DRAE. For instance, if the PaRAM set associated with channel 0 returns a completion code of 31 (OPT.TCC = 31), make sure that DRAE.E31 is also set for a shadow region completion interrupt because the interrupt pending register bit set will be IPR.I31.

Copyright © 2013–2016, Texas Instruments Incorporated

16.5.2 Miscellaneous Programming/Debug Tips

1. For several registers, the setting and clearing of bits needs to be done via separate dedicated registers. For example, the event register (ER) bits can only be cleared by writing a 1 to the corresponding bits in the event clear register (ECR). Similarly, the event enable register (EER) bits can only be set with writes of 1 to the corresponding bits in the event enable set registers (EESR) and can only be cleared with writes of 1 to the corresponding bits in the event enable clear register (EECR).
2. Writes to the shadow region memory maps are governed by region access enable registers (DRAE/QRAE). If the appropriate channels are not enabled in these registers, read/write access to the shadow region memory map is not enabled.
3. When working with shadow region completion interrupts, ensure that the DMA region access enable registers (DRAE) for every region are set in a mutually exclusive way (unless it is a requirement for an application). If there is an overlap in the allocated channels and transfer completion codes (setting of interrupt pending register bits) in the region resource allocation, it results in multiple shadow region completion interrupts. For example, if DRAE0.E0 and DRAE1.E0 are both set, then on completion of a transfer that returns a TCC = 0, they will generate both shadow region 0 and 1 completion interrupts.
4. While programming a non-dummy parameter set, ensure the CCNT is not left to zero.
5. Enable the EDMA3CC error interrupt in the device controller and attach an interrupt service routine (ISR) to ensure that error conditions are not missed in an application and are appropriately addressed with the ISR.
6. Depending on the application, you may want to break large transfers into smaller transfers and use self-chaining to prevent starvation of other events in an event queue.
7. In applications where a large transfer is broken into sets of small transfers using chaining or other methods, you might choose to use the early chaining option to reduce the time between the sets of transfers and increase the throughput. However, keep in mind that with early completion, all data might have not been received at the end point when completion is reported because the EDMA3CC internally signals completion when the TR is submitted to the EDMA3TC, potentially before any data has been transferred.
8. The event queue entries can be observed to determine the last few events if there is a system failure (provided the entries were not bypassed).
9. In order to put the EDMA3CC and EDMA3TC in power-down modes, you should ensure that there is no activity with the EDMA3CC and EDMA3TC. The EDMA3CC status register (CCSTAT) and the EDMA3TC channel status register (TCSTAT) should be used.

Copyright © 2013–2016, Texas Instruments Incorporated

16.6 Setting Up a Transfer

The following list provides a quick guide for the typical steps involved in setting up a transfer.

1. Initiating a DMA/QDMA channel:
   (a) Determine the type of channel (QDMA or DMA) to be used.
   (b) If using a QDMA channel, program the QDMA channel *n* mapping register (QCHMAP*n*) with the parameter set number to which the channel maps and the trigger word.
   (c) If the channel is being used in the context of a shadow region, ensure the DMA region access enable register (DRAE) for the region is properly set up to allow read/write accesses to bits in the event register and interrupt register in the shadow region memory-map. The subsequent steps in this process should be done using the respective shadow region registers. (Shadow region descriptions and usage are provided in Section 16.2.7.1.)
   (d) Determine the type of triggering used.
      (i) If external events are used for triggering (DMA channels), enable the respective event in EER by writing into EESR.
      (ii) If a QDMA channel is used, enable the channel in QEER by writing into QEESR.
   (e) Queue setup.
      (i) If a QDMA channel is used, set up QDMAQNUM to map the channel to the respective event queue.
      (ii) If a DMA channel is used, set up DMAQNUM to map the event to the respective event queue.
2. Parameter set setup: Program the PaRAM set number associated with the channel. Note that if it is a QDMA channel, the PaRAM entry that is configured as trigger word is written last. Alternatively, enable the QDMA channel just before the write to the trigger word.
   See Section 16.3 for parameter set field setups for different types of transfers. See the sections on chaining (Section 16.2.8) and interrupt completion (Section 16.2.9) on how to set up final/intermediate completion chaining and/or interrupts.
3. Interrupt setup:
   (a) If working in the context of a shadow region, ensure the relevant bits in DRAE are set.
   (b) Enable the interrupt in IER by writing into IESR.
   (c) Ensure that the EDMA3CC completion interrupt is enabled properly in the device interrupt controller.
   (d) Set up the interrupt controller properly to receive the expected EDMA3 interrupt.
4. Initiate transfer (this step is highly dependent on the event trigger source):
   (a) If the source is an external event coming from a peripheral, the peripheral will be enabled to start generating relevant EDMA3 events that can be latched to the ER transfer.
   (b) For QDMA events, writes to the trigger word will initiate the transfer.
   (c) Manually-triggered transfers will be initiated by writes to the event set register (ESR).
   (d) Chained-trigger events initiate when a previous transfer returns a transfer completion code equal to the chained channel number.
5. Wait for completion:
   (a) If the interrupts are enabled as mentioned in step 3, then the EDMA3CC generates a completion interrupt to the CPU whenever transfer completion results in setting the corresponding bits in the interrupt pending register (IPR). The set bits must be cleared in IPR by writing to the corresponding bit in ICR.
   (b) If polling for completion (interrupts not enabled in the device controller), then the application code can wait on the expected bits to be set in IPR. Again, the set bits in IPR must be manually cleared by writing to ICR before the next set of transfers is performed for the same transfer completion code values.

Copyright © 2013–2016, Texas Instruments Incorporated

General-Purpose Input/Output (GPIO)

The GPIO peripheral provides dedicated general-purpose pins that can be configured as either inputs or outputs. When configured as an output, you can write to an internal register to control the state driven on the output pin. When configured as an input, you can detect the state of the input by reading the state of an internal register. This chapter describes the GPIO.

Topic **Page**

17.1 Introduction **511**
17.2 Architecture **512**
17.3 Registers **520**

Copyright © 2013–2016, Texas Instruments Incorporated

17.1 Introduction

17.1.1 Purpose of the Peripheral

Most system-on-chip (SoC) devices require some general-purpose input/output (GPIO) functionality in order to interact with other components in the system using low-speed interface pins. The control and use of the GPIO capability on this device is grouped together in the GPIO peripheral and is described in the following sections.

17.1.2 Features

The GPIO peripheral consists of the following features.

- Output set/clear functionality through separate data set and clear registers allows multiple software processes to control GPIO signals without critical section protection.
- Set/clear functionality through writing to a single output data register is also supported.
- Separate input/output registers
  - Output register can be read to reflect output drive status.
  - Input register can be read to reflect pin status.
- All GPIO signals can be used as interrupt sources with configurable edge detection.
- All GPIO signals can be used to generate events to the EDMA.

17.1.3 Functional Block Diagram

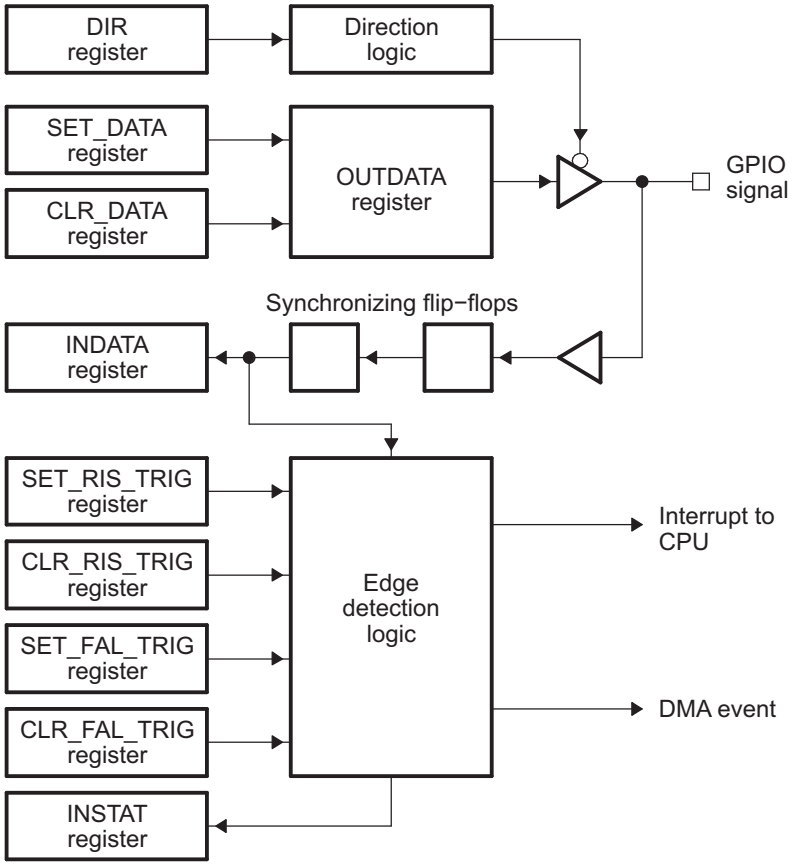
Figure 17-1 shows a block diagram of the GPIO peripheral.

17.1.4 Industry Standard(s) Compliance Statement

The GPIO peripheral connects to external devices. While it is possible that the software implements some standard connectivity protocol over GPIO, the GPIO peripheral itself is not compliant with any such standards.

Copyright © 2013–2016, Texas Instruments Incorporated

Figure 17-1. GPIO Block Diagram



17.2 Architecture

The following sections describe the GPIO peripheral.

17.2.1 Clock Control

The input clock to the GPIO peripheral is indicated in the device datasheet. The maximum operating speed of the GPIO peripheral is limited by system-level latencies. More specifically, how quickly the GPIO registers can be written to or read from.

17.2.2 Signal Descriptions

The number of GPIO signals supported will vary between devices. For information on the number of signals supported and the package pinout of each GPIO signal, see your device-specific data manual.

17.2.3 Pin Multiplexing

Extensive pin multiplexing is used to accommodate the largest number of peripheral functions in the smallest possible package. Pin multiplexing is controlled using a combination of hardware configuration at device reset and software programmable register settings. Refer to the device-specific data manual to determine how pin multiplexing affects the GPIO module.

17.2.4 Endianness Considerations

The GPIO operation is independent of endianness; therefore, there are no endianness considerations for the GPIO module.

Copyright © 2013–2016, Texas Instruments Incorporated

17.2.5 GPIO Register Structure

The GPIO signals are grouped by banks of 16 signals per bank. Each bank of GPIO signals has several registers with various control fields for each GPIO signal. Each 32-bit GPIO control register controls a pair of GPIO banks.

The register names for each bank of control registers (or pair of banks of GPIO bits) are all of the form *register_nameXY*, where *X* and *Y* are the two banks of GPIO bits controlled, such as 01, 23, 45, etc. The register fields associated with each GPIO are all of the form B*k*P*j*, where *k* is the GPIO bank and *j* is the pin number within the GPIO bank. For example, for GP2[5], which is located in GPIO bank 2, the control register names are of the form *register_name*23, and the register field associated with GP2[5] is GP2P5.

Table 17-1 shows the banks and register control bit information associated with each GPIO pin for up to 144 supportable pins. The table is not indicative of how many GPIO pins are supported on a device; it is only a reference for what register and field mappings look like for the first 144 supportable GPIO pins. For devices with less than 144 GPIO pins, assume that the extraneous fields and registers listed in the table are Reserved with no function. For devices with more than 144 GPIO pins, additional control registers and fields should be appended using the same numbering scheme in the table. Detailed information regarding the specific register names for each bank and the contents and function of these registers is presented in Section 17.3.

Table 17-1. GPIO Register Bits and Banks Associated With GPIO Signals

GPIO Pin Number	GPIO Signal Name	Bank Number	Control Registers	Register Bit	Register Field
1	GP0[0]	0	*register_name*01	Bit 0	GP0P0
2	GP0[1]	0	*register_name*01	Bit 1	GP0P1
3	GP0[2]	0	*register_name*01	Bit 2	GP0P2
4	GP0[3]	0	*register_name*01	Bit 3	GP0P3
5	GP0[4]	0	*register_name*01	Bit 4	GP0P4
6	GP0[5]	0	*register_name*01	Bit 5	GP0P5
7	GP0[6]	0	*register_name*01	Bit 6	GP0P6
8	GP0[7]	0	*register_name*01	Bit 7	GP0P7
9	GP0[8]	0	*register_name*01	Bit 8	GP0P8
10	GP0[9]	0	*register_name*01	Bit 9	GP0P9
11	GP0[10]	0	*register_name*01	Bit 10	GP0P10
12	GP0[11]	0	*register_name*01	Bit 11	GP0P11
13	GP0[12]	0	*register_name*01	Bit 12	GP0P12
14	GP0[13]	0	*register_name*01	Bit 13	GP0P13
15	GP0[14]	0	*register_name*01	Bit 14	GP0P14
16	GP0[15]	0	*register_name*01	Bit 15	GP0P15
17	GP1[0]	1	*register_name*01	Bit 16	GP1P0
18	GP1[1]	1	*register_name*01	Bit 17	GP1P1
19	GP1[2]	1	*register_name*01	Bit 18	GP1P2
20	GP1[3]	1	*register_name*01	Bit 19	GP1P3
21	GP1[4]	1	*register_name*01	Bit 20	GP1P4
22	GP1[5]	1	*register_name*01	Bit 21	GP1P5
23	GP1[6]	1	*register_name*01	Bit 22	GP1P6
24	GP1[7]	1	*register_name*01	Bit 23	GP1P7
25	GP1[8]	1	*register_name*01	Bit 24	GP1P8
26	GP1[9]	1	*register_name*01	Bit 25	GP1P9
27	GP1[10]	1	*register_name*01	Bit 26	GP1P10
28	GP1[11]	1	*register_name*01	Bit 27	GP1P11
29	GP1[12]	1	*register_name*01	Bit 28	GP1P12
30	GP1[13]	1	*register_name*01	Bit 29	GP1P13

Copyright © 2013–2016, Texas Instruments Incorporated

Table 17-1. GPIO Register Bits and Banks Associated With GPIO Signals (continued)

GPIO Pin Number	GPIO Signal Name	Bank Number	Control Registers	Register Bit	Register Field
31	GP1[14]	1	*register_name*01	Bit 30	GP1P14
32	GP1[15]	1	*register_name*01	Bit 31	GP1P15
33	GP2[0]	2	*register_name*23	Bit 0	GP2P0
34	GP2[1]	2	*register_name*23	Bit 1	GP2P1
35	GP2[2]	2	*register_name*23	Bit 2	GP2P2
36	GP2[3]	2	*register_name*23	Bit 3	GP2P3
37	GP2[4]	2	*register_name*23	Bit 4	GP2P4
38	GP2[5]	2	*register_name*23	Bit 5	GP2P5
39	GP2[6]	2	*register_name*23	Bit 6	GP2P6
40	GP2[7]	2	*register_name*23	Bit 7	GP2P7
41	GP2[8]	2	*register_name*23	Bit 8	GP2P8
42	GP2[9]	2	*register_name*23	Bit 9	GP2P9
43	GP2[10]	2	*register_name*23	Bit 10	GP2P10
44	GP2[11]	2	*register_name*23	Bit 11	GP2P11
45	GP2[12]	2	*register_name*23	Bit 12	GP2P12
46	GP2[13]	2	*register_name*23	Bit 13	GP2P13
47	GP2[14]	2	*register_name*23	Bit 14	GP2P14
48	GP2[15]	2	*register_name*23	Bit 15	GP2P15
49	GP3[0]	3	*register_name*23	Bit 16	GP3P0
50	GP3[1]	3	*register_name*23	Bit 17	GP3P1
51	GP3[2]	3	*register_name*23	Bit 18	GP3P2
52	GP3[3]	3	*register_name*23	Bit 19	GP3P3
53	GP3[4]	3	*register_name*23	Bit 20	GP3P4
54	GP3[5]	3	*register_name*23	Bit 21	GP3P5
55	GP3[6]	3	*register_name*23	Bit 22	GP3P6
56	GP3[7]	3	*register_name*23	Bit 23	GP3P7
57	GP3[8]	3	*register_name*23	Bit 24	GP3P8
58	GP3[9]	3	*register_name*23	Bit 25	GP3P9
59	GP3[10]	3	*register_name*23	Bit 26	GP3P10
60	GP3[11]	3	*register_name*23	Bit 27	GP3P11
61	GP3[12]	3	*register_name*23	Bit 28	GP3P12
62	GP3[13]	3	*register_name*23	Bit 29	GP3P13
63	GP3[14]	3	*register_name*23	Bit 30	GP3P14
64	GP3[15]	3	*register_name*23	Bit 31	GP3P15
65	GP4[0]	4	*register_name*45	Bit 0	GP4P0
66	GP4[1]	4	*register_name*45	Bit 1	GP4P1
67	GP4[2]	4	*register_name*45	Bit 2	GP4P2
68	GP4[3]	4	*register_name*45	Bit 3	GP4P3
69	GP4[4]	4	*register_name*45	Bit 4	GP4P4
70	GP4[5]	4	*register_name*45	Bit 5	GP4P5
71	GP4[6]	4	*register_name*45	Bit 6	GP4P6
72	GP4[7]	4	*register_name*45	Bit 7	GP4P7
73	GP4[8]	4	*register_name*45	Bit 8	GP4P8
74	GP4[9]	4	*register_name*45	Bit 9	GP4P9
75	GP4[10]	4	*register_name*45	Bit 10	GP4P10
76	GP4[11]	4	*register_name*45	Bit 11	GP4P11
77	GP4[12]	4	*register_name*45	Bit 12	GP4P12

Copyright © 2013–2016, Texas Instruments Incorporated

Table 17-1. GPIO Register Bits and Banks Associated With GPIO Signals (continued)

GPIO Pin Number	GPIO Signal Name	Bank Number	Control Registers	Register Bit	Register Field
78	GP4[13]	4	*register_name*45	Bit 13	GP4P13
79	GP4[14]	4	*register_name*45	Bit 14	GP4P14
80	GP4[15]	4	*register_name*45	Bit 15	GP4P15
81	GP5[0]	5	*register_name*45	Bit 16	GP5P0
82	GP5[1]	5	*register_name*45	Bit 17	GP5P1
83	GP5[2]	5	*register_name*45	Bit 18	GP5P2
84	GP5[3]	5	*register_name*45	Bit 19	GP5P3
85	GP5[4]	5	*register_name*45	Bit 20	GP5P4
86	GP5[5]	5	*register_name*45	Bit 21	GP5P5
87	GP5[6]	5	*register_name*45	Bit 22	GP5P6
88	GP5[7]	5	*register_name*45	Bit 23	GP5P7
89	GP5[8]	5	*register_name*45	Bit 24	GP5P8
90	GP5[9]	5	*register_name*45	Bit 25	GP5P9
91	GP5[10]	5	*register_name*45	Bit 26	GP5P10
92	GP5[11]	5	*register_name*45	Bit 27	GP5P11
93	GP5[12]	5	*register_name*45	Bit 28	GP5P12
94	GP5[13]	5	*register_name*45	Bit 29	GP5P13
95	GP5[14]	5	*register_name*45	Bit 30	GP5P14
96	GP5[15]	5	*register_name*45	Bit 31	GP5P15
97	GP6[0]	6	*register_name*67	Bit 0	GP6P0
98	GP6[1]	6	*register_name*67	Bit 1	GP6P1
99	GP6[2]	6	*register_name*67	Bit 2	GP6P2
100	GP6[3]	6	*register_name*67	Bit 3	GP6P3
101	GP6[4]	6	*register_name*67	Bit 4	GP6P4
102	GP6[5]	6	*register_name*67	Bit 5	GP6P5
103	GP6[6]	6	*register_name*67	Bit 6	GP6P6
104	GP6[7]	6	*register_name*67	Bit 7	GP6P7
105	GP6[8]	6	*register_name*67	Bit 8	GP6P8
106	GP6[9]	6	*register_name*67	Bit 9	GP6P9
107	GP6[10]	6	*register_name*67	Bit 10	GP6P10
108	GP6[11]	6	*register_name*67	Bit 11	GP6P11
109	GP6[12]	6	*register_name*67	Bit 12	GP6P12
110	GP6[13]	6	*register_name*67	Bit 13	GP6P13
111	GP6[14]	6	*register_name*67	Bit 14	GP6P14
112	GP6[15]	6	*register_name*67	Bit 15	GP6P15
113	GP7[0]	7	*register_name*67	Bit 16	GP7P0
114	GP7[1]	7	*register_name*67	Bit 17	GP7P1
115	GP7[2]	7	*register_name*67	Bit 18	GP7P2
116	GP7[3]	7	*register_name*67	Bit 19	GP7P3
117	GP7[4]	7	*register_name*67	Bit 20	GP7P4
118	GP7[5]	7	*register_name*67	Bit 21	GP7P5
119	GP7[6]	7	*register_name*67	Bit 22	GP7P6
120	GP7[7]	7	*register_name*67	Bit 23	GP7P7
121	GP7[8]	7	*register_name*67	Bit 24	GP7P8
122	GP7[9]	7	*register_name*67	Bit 25	GP7P9
123	GP7[10]	7	*register_name*67	Bit 26	GP7P10
124	GP7[11]	7	*register_name*67	Bit 27	GP7P11

Copyright © 2013–2016, Texas Instruments Incorporated

Table 17-1. GPIO Register Bits and Banks Associated With GPIO Signals (continued)

GPIO Pin Number	GPIO Signal Name	Bank Number	Control Registers	Register Bit	Register Field
125	GP7[12]	7	*register_name*67	Bit 28	GP7P12
126	GP7[13]	7	*register_name*67	Bit 29	GP7P13
127	GP7[14]	7	*register_name*67	Bit 30	GP7P14
128	GP7[15]	7	*register_name*67	Bit 31	GP7P15
129	GP8[0]	8	*register_name*8	Bit 0	GP8P0
130	GP8[1]	8	*register_name*8	Bit 1	GP8P1
131	GP8[2]	8	*register_name*8	Bit 2	GP8P2
132	GP8[3]	8	*register_name*8	Bit 3	GP8P3
133	GP8[4]	8	*register_name*8	Bit 4	GP8P4
134	GP8[5]	8	*register_name*8	Bit 5	GP8P5
135	GP8[6]	8	*register_name*8	Bit 6	GP8P6
136	GP8[7]	8	*register_name*8	Bit 7	GP8P7
137	GP8[8]	8	*register_name*8	Bit 8	GP8P8
138	GP8[9]	8	*register_name*8	Bit 9	GP8P9
139	GP8[10]	8	*register_name*8	Bit 10	GP8P10
140	GP8[11]	8	*register_name*8	Bit 11	GP8P11
141	GP8[12]	8	*register_name*8	Bit 12	GP8P12
142	GP8[13]	8	*register_name*8	Bit 13	GP8P13
143	GP8[14]	8	*register_name*8	Bit 14	GP8P14
144	GP8[15]	8	*register_name*8	Bit 15	GP8P15

17.2.6 Using a GPIO Signal as an Output

GPIO signals are configured to operate as inputs or outputs by writing the appropriate value to the GPIO direction register (DIR). This section describes using the GPIO signal as an output signal.

17.2.6.1 Configuring a GPIO Output Signal

To configure a given GPIO signal as an output, clear the bit in DIR that is associated with the desired GPIO signal. For detailed information on DIR, see Section 17.3.

17.2.6.2 Controlling the GPIO Output Signal State

There are three registers that control the output state driven on a GPIO signal configured as an output:

1. GPIO set data register (SET_DATA) controls driving GPIO signals high.
2. GPIO clear data register (CLR_DATA) controls driving GPIO signals low.
3. GPIO output data register (OUT_DATA) contains the current state of the output signals.

Reading SET_DATA, CLR_DATA, and OUT_DATA returns the output state, not necessarily the actual signal state (since some signals may be configured as inputs). The actual signal state is read using the GPIO input data register (IN_DATA) associated with the desired GPIO signal. IN_DATA contains the actual logic state on the external signal.

For detailed information on these registers, see Section 17.3.

Copyright © 2013–2016, Texas Instruments Incorporated

17.2.6.2.1 Driving a GPIO Output Signal High

To drive a GPIO signal high, use one of the following methods:

- Write a logic 1 to the bit in SET_DATA associated with the desired GPIO signal(s) to be driven high. Bit positions in SET_DATA containing logic 0 do not affect the state of the associated output signals.
- Modify the bit in OUT_DATA associated with the desired GPIO signal by using a read-modify-write operation. The logic states driven on the GPIO output signals match the logic values written to all bits in OUT_DATA.

For GPIO signals configured as inputs, the values written to the associated SET_DATA, CLR_DATA, and OUT_DATA bits have no effect.

17.2.6.2.2 Driving a GPIO Output Signal Low

To drive a GPIO signal low, use one of the following methods:

- Write a logic 1 to the bit in CLR_DATA associated with the desired GPIO signal(s) to be driven low. Bit positions in CLR_DATA containing logic 0 do not affect the state of the associated output signals.
- Modify the bit in OUT_DATA associated with the desired GPIO signal by using a read-modify-write operation. The logic states driven on the GPIO output signals match the logic values written to all bits in OUT_DATA.

For GPIO signals configured as inputs, the values written to the associated SET_DATA, CLR_DATA, and OUT_DATA bits have no effect.

17.2.7 Using a GPIO Signal as an Input

GPIO signals are configured to operate as inputs or outputs by writing the appropriate value to the GPIO direction register (DIR). This section describes using the GPIO signal as an input signal.

17.2.7.1 Configuring a GPIO Input Signal

To configure a given GPIO signal as an input, set the bit in DIR that is associated with the desired GPIO signal. For detailed information on DIR, see Section 17.3.

17.2.7.2 Reading a GPIO Input Signal

The current state of the GPIO signals is read using the GPIO input data register (IN_DATA).

- For GPIO signals configured as inputs, reading IN_DATA returns the state of the input signal synchronized to the GPIO peripheral clock.
- For GPIO signals configured as outputs, reading IN_DATA returns the output value being driven by the device.

Some signals may utilize open-drain output buffers for wired-logic operations. For open-drain GPIO signals, reading IN_DATA returns the wired-logic value on the signal (which will not be driven by the device alone). Information on any signals using open-drain outputs is available in your device-specific data manual.

To use GPIO input signals as interrupt sources, see Section 17.2.10.

17.2.8 Reset Considerations

The GPIO peripheral has two reset sources: software reset and hardware reset.

17.2.8.1 Software Reset Considerations

A software reset (such as a reset initiated through the emulator) does not modify the configuration and state of the GPIO signals. A reset invoked via the Power and Sleep Controller (PSC) (GPIO clock disable, PSC reset, followed by GPIO clock enable) will result in the default configuration register settings. For details on the PSC, see the *Power and Sleep Controller (PSC)* chapter.

Copyright © 2013–2016, Texas Instruments Incorporated

17.2.8.2 Hardware Reset Considerations

A hardware reset does reset the GPIO configuration and data registers to their default states; therefore, affecting the configuration and state of the GPIO signals.

17.2.9 Initialization

The following steps are required to configure the GPIO module after a hardware reset:

1. Perform the necessary device pin multiplexing setup (see your device-specific data manual).
2. Program the Power and Sleep Controller (PSC) to enable the GPIO module. For details on the PSC, see the *Power and Sleep Controller (PSC)* chapter.
3. Program the direction, data, and interrupt control registers to set the configuration of the desired GPIO pins (described in this chapter).

The GPIO module is now ready to perform data transactions.

17.2.10 Interrupt Support

The GPIO peripheral can send an interrupt event to the CPU.

17.2.10.1 Interrupt Events and Requests

All GPIO signals can be configured to generate interrupts. The device supports interrupts from single GPIO signals, interrupts from banks of GPIO signals, or both.

Note that the GPIO interrupts may also be used to provide synchronization events to the DMA controller.

17.2.10.2 Enabling GPIO Interrupt Events

GPIO interrupt events are enabled in banks of 16 by setting the appropriate bit(s) in the GPIO interrupt per-bank enable register (BINTEN). For example, to enable bank 0 interrupts (events from GP0[15-0]), set bit 0 in BINTEN; to enable bank 3 interrupts (events from GP3[15-0]), set bit 3 in BINTEN.

For detailed information on BINTEN, see Section 17.3.

17.2.10.3 Configuring GPIO Interrupt Edge Triggering

Each GPIO interrupt source can be configured to generate an interrupt on the GPIO signal rising edge, falling edge, both edges, or neither edge (no event). The edge detection is synchronized to the GPIO peripheral module clock.

The following four registers control the configuration of the GPIO interrupt edge detection:

1. The GPIO set rising edge interrupt register (SET_RIS_TRIG) enables GPIO interrupts on the occurrence of a rising edge on the GPIO signal.
2. The GPIO clear rising edge interrupt register (CLR_RIS_TRIG) disables GPIO interrupts on the occurrence of a rising edge on the GPIO signal.
3. The GPIO set falling edge interrupt register (SET_FAL_TRIG) enables GPIO interrupts on the occurrence of a falling edge on the GPIO signal.
4. The GPIO clear falling edge interrupt register (CLR_FAL_TRIG) disables GPIO interrupts on the occurrence of a falling edge on the GPIO signal.

To configure a GPIO interrupt to occur only on rising edges of the GPIO signal:

- Write a logic 1 to the associated bit in SET_RIS_TRIG.
- Write a logic 1 to the associated bit in CLR_FAL_TRIG.

To configure a GPIO interrupt to occur only on falling edges of the GPIO signal:

- Write a logic 1 to the associated bit in SET_FAL_TRIG.
- Write a logic 1 to the associated bit in CLR_RIS_TRIG.

Copyright © 2013–2016, Texas Instruments Incorporated

To configure a GPIO interrupt to occur on both the rising and falling edges of the GPIO signal:

- Write a logic 1 to the associated bit in SET_RIS_TRIG.
- Write a logic 1 to the associated bit in SET_FAL_TRIG.

To disable a specific GPIO interrupt:

- Write a logic 1 to the associated bit in CLR_RIS_TRIG.
- Write a logic 1 to the associated bit in CLR_FAL_TRIG.

For detailed information on these registers, see Section 17.3.

Note that the direction of the GPIO signal does not have to be an input for the interrupt event generation to work. When a GPIO signal is configured as an output, the software can change the GPIO signal state and, in turn, generate an interrupt. This can be useful for debugging interrupt signal connectivity.

17.2.10.4 GPIO Interrupt Status

The status of GPIO interrupt events can be monitored by reading the GPIO interrupt status register (INTSTAT). Pending GPIO interrupts are indicated with a logic 1 in the associated bit position; interrupts that are not pending are indicated with a logic 0.

For individual GPIO interrupts that are directly routed to the DSP subsystem, the interrupt status can be read by reading the associated interrupt flag in the CPU. For the GPIO bank interrupts, INTSTAT can be used to determine which GPIO interrupt occurred. It is the responsibility of software to ensure that all pending GPIO interrupts are appropriately serviced.

Pending GPIO interrupt flags can be cleared by writing a logic 1 to the associated bit position in INTSTAT.

For detailed information on INTSTAT, see Section 17.3.

17.2.10.5 Interrupt Multiplexing

GPIO interrupts may be multiplexed with other interrupt functions on the device.

17.2.11 EDMA Event Support

The GPIO peripheral may provide synchronization events to the DMA controller.

17.2.12 Power Management

The GPIO peripheral can be placed in reduced-power modes to conserve power during periods of low activity. The power management of the GPIO peripheral is controlled by the processor Power and Sleep Controller (PSC). The PSC acts as a master controller for power management for all of the peripherals on the device. For detailed information on power management procedures using the PSC, see the *Power and Sleep Controller (PSC)* chapter.

When the GPIO peripheral is placed in a low-power state by the PSC, the interrupt generation capability is suspended until the GPIO peripheral is removed from the low-power state. While in the low-power state, the GPIO signals configured as outputs are maintained at their state prior to the GPIO peripheral entering the low-power state.

17.2.13 Emulation Considerations

The GPIO peripheral is not affected by emulation suspend events (such as halts and breakpoints).

Copyright © 2013–2016, Texas Instruments Incorporated

17.3 Registers

Table 17-2 lists the memory-mapped registers for the general-purpose input/output (GPIO). The table enumerates the registers required to support 144 GPIO pins, however not all devices will support 144 GPIO pins. For devices with less than 144 GPIO pins, assume that the extraneous fields and registers are Reserved and serve no function. For devices with more than 144 GPIO pins, append registers and fields as necessary using the address offset scheme in the table. See your device-specific data manual for the number of GPIO pins supported and the base memory address for these registers.

Table 17-2. GPIO Registers

Offset	Acronym	Register Description	Section
0h	REVID	Revision ID Register	Section 17.3.1
8h	BINTEN	GPIO Interrupt Per-Bank Enable Register	Section 17.3.2
		GPIO Banks 0 and 1	
10h	DIR01	GPIO Banks 0 and 1 Direction Register	Section 17.3.3
14h	OUT_DATA01	GPIO Banks 0 and 1 Output Data Register	Section 17.3.4
18h	SET_DATA01	GPIO Banks 0 and 1 Set Data Register	Section 17.3.5
1Ch	CLR_DATA01	GPIO Banks 0 and 1 Clear Data Register	Section 17.3.6
20h	IN_DATA01	GPIO Banks 0 and 1 Input Data Register	Section 17.3.7
24h	SET_RIS_TRIG01	GPIO Banks 0 and 1 Set Rising Edge Interrupt Register	Section 17.3.8
28h	CLR_RIS_TRIG01	GPIO Banks 0 and 1 Clear Rising Edge Interrupt Register	Section 17.3.9
2Ch	SET_FAL_TRIG01	GPIO Banks 0 and 1 Set Falling Edge Interrupt Register	Section 17.3.10
30h	CLR_FAL_TRIG01	GPIO Banks 0 and 1 Clear Falling Edge Interrupt Register	Section 17.3.11
34h	INTSTAT01	GPIO Banks 0 and 1 Interrupt Status Register	Section 17.3.12
		GPIO Banks 2 and 3	
38h	DIR23	GPIO Banks 2 and 3 Direction Register	Section 17.3.3
3Ch	OUT_DATA23	GPIO Banks 2 and 3 Output Data Register	Section 17.3.4
40h	SET_DATA23	GPIO Banks 2 and 3 Set Data Register	Section 17.3.5
44h	CLR_DATA23	GPIO Banks 2 and 3 Clear Data Register	Section 17.3.6
48h	IN_DATA23	GPIO Banks 2 and 3 Input Data Register	Section 17.3.7
4Ch	SET_RIS_TRIG23	GPIO Banks 2 and 3 Set Rising Edge Interrupt Register	Section 17.3.8
50h	CLR_RIS_TRIG23	GPIO Banks 2 and 3 Clear Rising Edge Interrupt Register	Section 17.3.9
54h	SET_FAL_TRIG23	GPIO Banks 2 and 3 Set Falling Edge Interrupt Register	Section 17.3.10
58h	CLR_FAL_TRIG23	GPIO Banks 2 and 3 Clear Falling Edge Interrupt Register	Section 17.3.11
5Ch	INTSTAT23	GPIO Banks 2 and 3 Interrupt Status Register	Section 17.3.12
		GPIO Banks 4 and 5	
60h	DIR45	GPIO Banks 4 and 5 Direction Register	Section 17.3.3
64h	OUT_DATA45	GPIO Banks 4 and 5 Output Data Register	Section 17.3.4
68h	SET_DATA45	GPIO Banks 4 and 5 Set Data Register	Section 17.3.5
6Ch	CLR_DATA45	GPIO Banks 4 and 5 Clear Data Register	Section 17.3.6
70h	IN_DATA45	GPIO Banks 4 and 5 Input Data Register	Section 17.3.7
74h	SET_RIS_TRIG45	GPIO Banks 4 and 5 Set Rising Edge Interrupt Register	Section 17.3.8
78h	CLR_RIS_TRIG45	GPIO Banks 4 and 5 Clear Rising Edge Interrupt Register	Section 17.3.9
7Ch	SET_FAL_TRIG45	GPIO Banks 4 and 5 Set Falling Edge Interrupt Register	Section 17.3.10
80h	CLR_FAL_TRIG45	GPIO Banks 4 and 5 Clear Falling Edge Interrupt Register	Section 17.3.11
84h	INTSTAT45	GPIO Banks 4 and 5 Interrupt Status Register	Section 17.3.12

Copyright © 2013–2016, Texas Instruments Incorporated

Table 17-2. GPIO Registers (continued)

Offset	Acronym	Register Description	Section
		GPIO Banks 6 and 7	
88h	DIR67	GPIO Banks 6 and 7 Direction Register	Section 17.3.3
8Ch	OUT_DATA67	GPIO Banks 6 and 7 Output Data Register	Section 17.3.4
90h	SET_DATA67	GPIO Banks 6 and 7 Set Data Register	Section 17.3.5
94h	CLR_DATA67	GPIO Banks 6 and 7 Clear Data Register	Section 17.3.6
98h	IN_DATA67	GPIO Banks 6 and 7 Input Data Register	Section 17.3.7
9Ch	SET_RIS_TRIG67	GPIO Banks 6 and 7 Set Rising Edge Interrupt Register	Section 17.3.8
A0h	CLR_RIS_TRIG67	GPIO Banks 6 and 7 Clear Rising Edge Interrupt Register	Section 17.3.9
A4h	SET_FAL_TRIG67	GPIO Banks 6 and 7 Set Falling Edge Interrupt Register	Section 17.3.10
A8h	CLR_FAL_TRIG67	GPIO Banks 6 and 7 Clear Falling Edge Interrupt Register	Section 17.3.11
ACh	INTSTAT67	GPIO Banks 6 and 7 Interrupt Status Register	Section 17.3.12
		GPIO Bank 8	
B0h	DIR8	GPIO Bank 8 Direction Register	Section 17.3.3
B4h	OUT_DATA8	GPIO Bank 8 Output Data Register	Section 17.3.4
B8h	SET_DATA8	GPIO Bank 8 Set Data Register	Section 17.3.5
BCh	CLR_DATA8	GPIO Bank 8 Clear Data Register	Section 17.3.6
C0h	IN_DATA8	GPIO Bank 8 Input Data Register	Section 17.3.7
C4h	SET_RIS_TRIG8	GPIO Bank 8 Set Rising Edge Interrupt Register	Section 17.3.8
C8h	CLR_RIS_TRIG8	GPIO Bank 8 Clear Rising Edge Interrupt Register	Section 17.3.9
CCh	SET_FAL_TRIG8	GPIO Bank 8 Set Falling Edge Interrupt Register	Section 17.3.10
D0h	CLR_FAL_TRIG8	GPIO Bank 8 Clear Falling Edge Interrupt Register	Section 17.3.11
D4h	INTSTAT8	GPIO Bank 8 Interrupt Status Register	Section 17.3.12

17.3.1 Revision ID Register (REVID)

The revision ID register (REVID) contains the peripheral version information. REVID is shown in Figure 17-2 and described in Table 17-3.

Figure 17-2. Revision ID Register (REVID)

31	0
REV	
R-4483 0105h	

LEGEND: R = Read only; -*n* = value after reset

Table 17-3. Revision ID Register (REVID) Field Descriptions

Bit	Field	Value	Description
31-0	REV	4483 0105h	Peripheral Revision

Copyright © 2013–2016, Texas Instruments Incorporated

17.3.2 GPIO Interrupt Per-Bank Enable Register (BINTEN)

The GPIO interrupt per-bank enable register (BINTEN) is shown in Figure 17-3 and described in Table 17-4. For information on which GPIO signals are associated with each bank, see Table 17-1. Note that the bits in BINTEN control both the interrupt and EDMA events.

Figure 17-3. GPIO Interrupt Per-Bank Enable Register (BINTEN)

31	16
Reserved	
R-0	

15	9	8	7	6	5	4	3	2	1	0
Reserved		EN8	EN7	EN6	EN5	EN4	EN3	EN2	EN1	EN0
R-0		R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 17-4. GPIO Interrupt Per-Bank Enable Register (BINTEN) Field Descriptions

Bit	Field	Value	Description
31-9	Reserved	0	Reserved
8	EN8		Bank 8 interrupt enable is used to disable or enable the bank 8 interrupts (events from GP8[15-0]).
		0	Bank 8 interrupts are disabled.
		1	Bank 8 interrupts are enabled.
7	EN7		Bank 7 interrupt enable is used to disable or enable the bank 7 interrupts (events from GP7[15-0]).
		0	Bank 7 interrupts are disabled.
		1	Bank 7 interrupts are enabled.
6	EN6		Bank 6 interrupt enable is used to disable or enable the bank 6 interrupts (events from GP6[15-0]).
		0	Bank 6 interrupts are disabled.
		1	Bank 6 interrupts are enabled.
5	EN5		Bank 5 interrupt enable is used to disable or enable the bank 5 interrupts (events from GP5[15-0]).
		0	Bank 5 interrupts are disabled.
		1	Bank 5 interrupts are enabled.
4	EN4		Bank 4 interrupt enable is used to disable or enable the bank 4 interrupts (events from GP4[15-0]).
		0	Bank 4 interrupts are disabled.
		1	Bank 4 interrupts are enabled.
3	EN3		Bank 3 interrupt enable is used to disable or enable the bank 3 interrupts (events from GP3[15-0])
		0	Bank 3 interrupts are disabled.
		1	Bank 3 interrupts are enabled.
2	EN2		Bank 2 interrupt enable is used to disable or enable the bank 2 interrupts (events from GP2[15-0]).
		0	Bank 2 interrupts are disabled.
		1	Bank 2 interrupts are enabled.
1	EN1		Bank 1 interrupt enable is used to disable or enable the bank 1 interrupts (events from GP1[15-0]).
		0	Bank 1 interrupts are disabled.
		1	Bank 1 interrupts are enabled.
0	EN0		Bank 0 interrupt enable is used to disable or enable the bank 0 interrupts (events from GP0[15-0]).
		0	Bank 0 interrupts are disabled.
		1	Bank 0 interrupts are enabled.

Copyright © 2013–2016, Texas Instruments Incorporated

17.3.3 GPIO Direction Registers (DIR*n*)

The GPIO direction register (DIR*n*) determines if GPIO pin *j* in GPIO bank *k* is an input or an output. Each of the GPIO banks may have up to 16 GPIO pins. By default, all the GPIO pins are configured as inputs (bit value = 1). The GPIO direction register (DIR01) is shown in Figure 17-4, DIR23 is shown in Figure 17-5, DIR45 is shown in Figure 17-6, DIR67 is shown in Figure 17-7, DIR8 is shown in Figure 17-8, and described in Table 17-5. See Table 17-1 to determine the DIR*n* bit associated with each GPIO bank and pin number.

Figure 17-4. GPIO Banks 0 and 1 Direction Register (DIR01)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
GP1P15	GP1P14	GP1P13	GP1P12	GP1P11	GP1P10	GP1P9	GP1P8	GP1P7	GP1P6	GP1P5	GP1P4	GP1P3	GP1P2	GP1P1	GP1P0
R/W-1															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GP0P15	GP0P14	GP0P13	GP0P12	GP0P11	GP0P10	GP0P9	GP0P8	GP0P7	GP0P6	GP0P5	GP0P4	GP0P3	GP0P2	GP0P1	GP0P0
R/W-1															

LEGEND: R/W = Read/Write; -*n* = value after reset

Figure 17-5. GPIO Banks 2 and 3 Direction Register (DIR23)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
GP3P15	GP3P14	GP3P13	GP3P12	GP3P11	GP3P10	GP3P9	GP3P8	GP3P7	GP3P6	GP3P5	GP3P4	GP3P3	GP3P2	GP3P1	GP3P0
R/W-1															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GP2P15	GP2P14	GP2P13	GP2P12	GP2P11	GP2P10	GP2P9	GP2P8	GP2P7	GP2P6	GP2P5	GP2P4	GP2P3	GP2P2	GP2P1	GP2P0
R/W-1															

LEGEND: R/W = Read/Write; -*n* = value after reset

Figure 17-6. GPIO Banks 4 and 5 Direction Register (DIR45)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
GP5P15	GP5P14	GP5P13	GP5P12	GP5P11	GP5P10	GP5P9	GP5P8	GP5P7	GP5P6	GP5P5	GP5P4	GP5P3	GP5P2	GP5P1	GP5P0
R/W-1															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GP4P15	GP4P14	GP4P13	GP4P12	GP4P11	GP4P10	GP4P9	GP4P8	GP4P7	GP4P6	GP4P5	GP4P4	GP4P3	GP4P2	GP4P1	GP4P0
R/W-1															

LEGEND: R/W = Read/Write; -*n* = value after reset

Figure 17-7. GPIO Banks 6 and 7 Direction Register (DIR67)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
GP7P15	GP7P14	GP7P13	GP7P12	GP7P11	GP7P10	GP7P9	GP7P8	GP7P7	GP7P6	GP7P5	GP7P4	GP7P3	GP7P2	GP7P1	GP7P0
R/W-1															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GP6P15	GP6P14	GP6P13	GP6P12	GP6P11	GP6P10	GP6P9	GP6P8	GP6P7	GP6P6	GP6P5	GP6P4	GP6P3	GP6P2	GP6P1	GP6P0
R/W-1															

LEGEND: R/W = Read/Write; -*n* = value after reset

Copyright © 2013–2016, Texas Instruments Incorporated

Figure 17-8. GPIO Bank 8 Direction Register (DIR8)

31															16
Reserved															
R/W-1															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GP8P15	GP8P14	GP8P13	GP8P12	GP8P11	GP8P10	GP8P9	GP8P8	GP8P7	GP8P6	GP8P5	GP8P4	GP8P3	GP8P2	GP8P1	GP8P0
R/W-1															

LEGEND: R/W = Read/Write; -*n* = value after reset

Table 17-5. GPIO Direction Register (DIR*n*) Field Descriptions

Bit	Field	Value	Description
31-0	GP*k*P*j*		Direction of pin GP*k*[*j*]. The GP*k*P*j* bit is used to control the direction (output = 0, input = 1) of pin *j* in GPIO bank*k*.
		0	GP*k*[*j*] is an output.
		1	GP*k*[*j*] is an input.

Copyright © 2013–2016, Texas Instruments Incorporated

17.3.4 GPIO Output Data Registers (OUT_DATA*n*)

The GPIO output data register (OUT_DATA*n*) determines the value driven on the corresponding GPIO pin *j* in GPIO bank *k*, if the pin is configured as an output (DIR*n* = 0). Writes do not affect pins not configured as GPIO outputs. The bits in OUT_DATA*n* are set or cleared by writing directly to this register. A read of OUT_DATA*n* returns the value of the register not the value at the pin (that might be configured as an input). The GPIO output data register (OUT_DATA01) is shown in Figure 17-9, OUT_DATA23 is shown in Figure 17-10, OUT_DATA45 is shown in Figure 17-11, OUT_DATA67 is shown in Figure 17-12, OUT_DATA8 is shown in Figure 17-13, and described in Table 17-6. See Table 17-1 to determine the OUT_DATA*n* bit associated with each GPIO bank and pin number.

Figure 17-9. GPIO Banks 0 and 1 Output Data Register (OUT_DATA01)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
GP1P15	GP1P14	GP1P13	GP1P12	GP1P11	GP1P10	GP1P9	GP1P8	GP1P7	GP1P6	GP1P5	GP1P4	GP1P3	GP1P2	GP1P1	GP1P0
R/W-0															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GP0P15	GP0P14	GP0P13	GP0P12	GP0P11	GP0P10	GP0P9	GP0P8	GP0P7	GP0P6	GP0P5	GP0P4	GP0P3	GP0P2	GP0P1	GP0P0
R/W-0															

LEGEND: R/W = Read/Write; -*n* = value after reset

Figure 17-10. GPIO Banks 2 and 3 Output Data Register (OUT_DATA23)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
GP3P15	GP3P14	GP3P13	GP3P12	GP3P11	GP3P10	GP3P9	GP3P8	GP3P7	GP3P6	GP3P5	GP3P4	GP3P3	GP3P2	GP3P1	GP3P0
R/W-0															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GP2P15	GP2P14	GP2P13	GP2P12	GP2P11	GP2P10	GP2P9	GP2P8	GP2P7	GP2P6	GP2P5	GP2P4	GP2P3	GP2P2	GP2P1	GP2P0
R/W-0															

LEGEND: R/W = Read/Write; -*n* = value after reset

Figure 17-11. GPIO Banks 4 and 5 Output Data Register (OUT_DATA45)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
GP5P15	GP5P14	GP5P13	GP5P12	GP5P11	GP5P10	GP5P9	GP5P8	GP5P7	GP5P6	GP5P5	GP5P4	GP5P3	GP5P2	GP5P1	GP5P0
R/W-0															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GP4P15	GP4P14	GP4P13	GP4P12	GP4P11	GP4P10	GP4P9	GP4P8	GP4P7	GP4P6	GP4P5	GP4P4	GP4P3	GP4P2	GP4P1	GP4P0
R/W-0															

LEGEND: R/W = Read/Write; -*n* = value after reset

Figure 17-12. GPIO Banks 6 and 7 Output Data Register (OUT_DATA67)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
GP7P15	GP7P14	GP7P13	GP7P12	GP7P11	GP7P10	GP7P9	GP7P8	GP7P7	GP7P6	GP7P5	GP7P4	GP7P3	GP7P2	GP7P1	GP7P0
R/W-0															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GP6P15	GP6P14	GP6P13	GP6P12	GP6P11	GP6P10	GP6P9	GP6P8	GP6P7	GP6P6	GP6P5	GP6P4	GP6P3	GP6P2	GP6P1	GP6P0
R/W-0															

LEGEND: R/W = Read/Write; -*n* = value after reset

Copyright © 2013–2016, Texas Instruments Incorporated

Figure 17-13. GPIO Bank 8 Output Data Register (OUT_DATA8)

31															16
Reserved															
R/W-0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GP8P15	GP8P14	GP8P13	GP8P12	GP8P11	GP8P10	GP8P9	GP8P8	GP8P7	GP8P6	GP8P5	GP8P4	GP8P3	GP8P2	GP8P1	GP8P0
R/W-0															

LEGEND: R/W = Read/Write; -*n* = value after reset

Table 17-6. GPIO Output Data Register (OUT_DATA*n*) Field Descriptions

Bit	Field	Value	Description
31-0	GP*k*P*j*		Output drive state of GP*k*[*j*]. The GP*k*P*j* bit is used to drive the output (low = 0, high = 1) of pin *j* in GPIO bank*k*. The GP*k*P*j* bit is ignored when GP*k*[*j*] is configured as an input.
		0	GP*k*[*j*] is driven low.
		1	GP*k*[*j*] is driven high.

Copyright © 2013–2016, Texas Instruments Incorporated

17.3.5 GPIO Set Data Registers (SET_DATA*n*)

The GPIO set data register (SET_DATA*n*) controls driving high of the corresponding GPIO pin *j* in GPIO bank *k*, if the pin is configured as an output (DIR*n* = 0). Writes do not affect pins not configured as GPIO outputs. Writing a 1 to a specific bit in SET_DATA*n* sets the corresponding GPIO pin *j* in GPIO bank *k*. A read of the B*k*P*j* bit returns the output drive state of the corresponding pin GPIO*k*[*j*]. The GPIO set data register (SET_DATA01) is shown in Figure 17-14, SET_DATA23 is shown in Figure 17-15, SET_DATA45 is shown in Figure 17-16, SET_DATA67 is shown in Figure 17-17, SET_DATA8 is shown in Figure 17-18, and described in Table 17-7. See Table 17-1 to determine the SET_DATA*n* bit associated with each GPIO bank and pin number.

Figure 17-14. GPIO Banks 0 and 1 Set Data Register (SET_DATA01)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
GP1P15	GP1P14	GP1P13	GP1P12	GP1P11	GP1P10	GP1P9	GP1P8	GP1P7	GP1P6	GP1P5	GP1P4	GP1P3	GP1P2	GP1P1	GP1P0
R/W-0															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GP0P15	GP0P14	GP0P13	GP0P12	GP0P11	GP0P10	GP0P9	GP0P8	GP0P7	GP0P6	GP0P5	GP0P4	GP0P3	GP0P2	GP0P1	GP0P0
R/W-0															

LEGEND: R/W = Read/Write; -*n* = value after reset

Figure 17-15. GPIO Banks 2 and 3 Set Data Register (SET_DATA23)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
GP3P15	GP3P14	GP3P13	GP3P12	GP3P11	GP3P10	GP3P9	GP3P8	GP3P7	GP3P6	GP3P5	GP3P4	GP3P3	GP3P2	GP3P1	GP3P0
R/W-0															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GP2P15	GP2P14	GP2P13	GP2P12	GP2P11	GP2P10	GP2P9	GP2P8	GP2P7	GP2P6	GP2P5	GP2P4	GP2P3	GP2P2	GP2P1	GP2P0
R/W-0															

LEGEND: R/W = Read/Write; -*n* = value after reset

Figure 17-16. GPIO Banks 4 and 5 Set Data Register (SET_DATA45)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
GP5P15	GP5P14	GP5P13	GP5P12	GP5P11	GP5P10	GP5P9	GP5P8	GP5P7	GP5P6	GP5P5	GP5P4	GP5P3	GP5P2	GP5P1	GP5P0
R/W-0															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GP4P15	GP4P14	GP4P13	GP4P12	GP4P11	GP4P10	GP4P9	GP4P8	GP4P7	GP4P6	GP4P5	GP4P4	GP4P3	GP4P2	GP4P1	GP4P0
R/W-0															

LEGEND: R/W = Read/Write; -*n* = value after reset

Figure 17-17. GPIO Banks 6 and 7 Set Data Register (SET_DATA67)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
GP7P15	GP7P14	GP7P13	GP7P12	GP7P11	GP7P10	GP7P9	GP7P8	GP7P7	GP7P6	GP7P5	GP7P4	GP7P3	GP7P2	GP7P1	GP7P0
R/W-0															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GP6P15	GP6P14	GP6P13	GP6P12	GP6P11	GP6P10	GP6P9	GP6P8	GP6P7	GP6P6	GP6P5	GP6P4	GP6P3	GP6P2	GP6P1	GP6P0
R/W-0															

LEGEND: R/W = Read/Write; -*n* = value after reset

Copyright © 2013–2016, Texas Instruments Incorporated

Figure 17-18. GPIO Bank 8 Set Data Register (SET_DATA8)

31															16
Reserved															
R/W-0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GP8P15	GP8P14	GP8P13	GP8P12	GP8P11	GP8P10	GP8P9	GP8P8	GP8P7	GP8P6	GP8P5	GP8P4	GP8P3	GP8P2	GP8P1	GP8P0
R/W-0															

LEGEND: R/W = Read/Write; -*n* = value after reset

Table 17-7. GPIO Set Data Register (SET_DATA*n*) Field Descriptions

Bit	Field	Value	Description
31-0	GP*k*P*j*		Set the output drive state of GP*k*[*j*] to logic high. The GP*k*P*j* bit is used to drive the output high on pin *j* in GPIO bank*k*. The GP*k*P*j* bit is ignored when GP*k*[*j*] is configured as an input. Reading the GP*k*P*j* bit returns the output drive state of GP*k*[*j*].
		0	No effect.
		1	GP*k*[*j*] is set to output logic high.

Copyright © 2013–2016, Texas Instruments Incorporated

17.3.6 GPIO Clear Data Registers (CLR_DATA*n*)

The GPIO clear data register (CLR_DATA*n*) controls clearing low of the corresponding GPIO pin *j* in GPIO bank *k*, if the pin is configured as an output (DIR*n* = 0). Writes do not affect pins not configured as GPIO outputs. Writing a 1 to a specific bit in CLR_DATA*n* resets the corresponding GPIO pin *j* in GPIO bank *k*. A read of the B*k*P*j* bit returns the output drive state of the corresponding pin GPIO*k*[*j*]. The GPIO clear data register (CLR_DATA01) is shown in Figure 17-19, CLR_DATA23 is shown in Figure 17-20, CLR_DATA45 is shown in Figure 17-21, CLR_DATA67 is shown in Figure 17-22, CLR_DATA8 is shown in Figure 17-23, and described in Table 17-8. See Table 17-1 to determine the CLR_DATA*n* bit associated with each GPIO bank and pin number.

Figure 17-19. GPIO Banks 0 and 1 Clear Data Register (CLR_DATA01)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
GP1P15	GP1P14	GP1P13	GP1P12	GP1P11	GP1P10	GP1P9	GP1P8	GP1P7	GP1P6	GP1P5	GP1P4	GP1P3	GP1P2	GP1P1	GP1P0
R/W-0															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GP0P15	GP0P14	GP0P13	GP0P12	GP0P11	GP0P10	GP0P9	GP0P8	GP0P7	GP0P6	GP0P5	GP0P4	GP0P3	GP0P2	GP0P1	GP0P0
R/W-0															

LEGEND: R/W = Read/Write; -*n* = value after reset

Figure 17-20. GPIO Banks 2 and 3 Clear Data Register (CLR_DATA23)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
GP3P15	GP3P14	GP3P13	GP3P12	GP3P11	GP3P10	GP3P9	GP3P8	GP3P7	GP3P6	GP3P5	GP3P4	GP3P3	GP3P2	GP3P1	GP3P0
R/W-0															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GP2P15	GP2P14	GP2P13	GP2P12	GP2P11	GP2P10	GP2P9	GP2P8	GP2P7	GP2P6	GP2P5	GP2P4	GP2P3	GP2P2	GP2P1	GP2P0
R/W-0															

LEGEND: R/W = Read/Write; -*n* = value after reset

Figure 17-21. GPIO Banks 4 and 5 Clear Data Register (CLR_DATA45)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
GP5P15	GP5P14	GP5P13	GP5P12	GP5P11	GP5P10	GP5P9	GP5P8	GP5P7	GP5P6	GP5P5	GP5P4	GP5P3	GP5P2	GP5P1	GP5P0
R/W-0															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GP4P15	GP4P14	GP4P13	GP4P12	GP4P11	GP4P10	GP4P9	GP4P8	GP4P7	GP4P6	GP4P5	GP4P4	GP4P3	GP4P2	GP4P1	GP4P0
R/W-0															

LEGEND: R/W = Read/Write; -*n* = value after reset

Figure 17-22. GPIO Banks 6 and 7 Clear Data Register (CLR_DATA67)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
GP7P15	GP7P14	GP7P13	GP7P12	GP7P11	GP7P10	GP7P9	GP7P8	GP7P7	GP7P6	GP7P5	GP7P4	GP7P3	GP7P2	GP7P1	GP7P0
R/W-0															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GP6P15	GP6P14	GP6P13	GP6P12	GP6P11	GP6P10	GP6P9	GP6P8	GP6P7	GP6P6	GP6P5	GP6P4	GP6P3	GP6P2	GP6P1	GP6P0
R/W-0															

LEGEND: R/W = Read/Write; -*n* = value after reset

Copyright © 2013–2016, Texas Instruments Incorporated

Figure 17-23. GPIO Bank 8 Clear Data Register (CLR_DATA8)

31															16
Reserved															
R/W-0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GP8P15	GP8P14	GP8P13	GP8P12	GP8P11	GP8P10	GP8P9	GP8P8	GP8P7	GP8P6	GP8P5	GP8P4	GP8P3	GP8P2	GP8P1	GP8P0
R/W-0															

LEGEND: R/W = Read/Write; -*n* = value after reset

Table 17-8. GPIO Clear Data Register (CLR_DATA*n*) Field Descriptions

Bit	Field	Value	Description
31-0	GP*k*P*j*		Clear the output drive state of GP*k*[*j*] to logic low. The GP*k*P*j* bit is used to drive the output low on pin *j* in GPIO bank*k*. The GP*k*P*j* bit is ignored when GP*k*[*j*] is configured as an input. Reading the GP*k*P*j* bit returns the output drive state of GP*k*[*j*].
		0	No effect.
		1	GP*k*[*j*] is set to output logic low.

Copyright © 2013–2016, Texas Instruments Incorporated

17.3.7 GPIO Input Data Registers (IN_DATA*n*)

The current state of the GPIO signals is read using the GPIO input data register (IN_DATA*n*).

- For GPIO signals configured as inputs, reading IN_DATA*n* returns the state of the input signal synchronized to the GPIO peripheral clock.
- For GPIO signals configured as outputs, reading IN_DATA*n* returns the output value being driven by the device.

The GPIO input data register (IN_DATA01) is shown in Figure 17-24, IN_DATA23 is shown in Figure 17-25, IN_DATA45 is shown in Figure 17-26, IN_DATA67 is shown in Figure 17-27, IN_DATA8 is shown in Figure 17-28, and described in Table 17-9. See Table 17-1 to determine the IN_DATA*n* bit associated with each GPIO bank and pin number.

Figure 17-24. GPIO Banks 0 and 1 Input Data Register (IN_DATA01)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
GP1P15	GP1P14	GP1P13	GP1P12	GP1P11	GP1P10	GP1P9	GP1P8	GP1P7	GP1P6	GP1P5	GP1P4	GP1P3	GP1P2	GP1P1	GP1P0
R-0															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GP0P15	GP0P14	GP0P13	GP0P12	GP0P11	GP0P10	GP0P9	GP0P8	GP0P7	GP0P6	GP0P5	GP0P4	GP0P3	GP0P2	GP0P1	GP0P0
R-0															

LEGEND: R = Read only; -*n* = value after reset

Figure 17-25. GPIO Banks 2 and 3 Input Data Register (IN_DATA23)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
GP3P15	GP3P14	GP3P13	GP3P12	GP3P11	GP3P10	GP3P9	GP3P8	GP3P7	GP3P6	GP3P5	GP3P4	GP3P3	GP3P2	GP3P1	GP3P0
R-0															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GP2P15	GP2P14	GP2P13	GP2P12	GP2P11	GP2P10	GP2P9	GP2P8	GP2P7	GP2P6	GP2P5	GP2P4	GP2P3	GP2P2	GP2P1	GP2P0
R-0															

LEGEND: R = Read only; -*n* = value after reset

Figure 17-26. GPIO Banks 4 and 5 Input Data Register (IN_DATA45)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
GP5P15	GP5P14	GP5P13	GP5P12	GP5P11	GP5P10	GP5P9	GP5P8	GP5P7	GP5P6	GP5P5	GP5P4	GP5P3	GP5P2	GP5P1	GP5P0
R-0															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GP4P15	GP4P14	GP4P13	GP4P12	GP4P11	GP4P10	GP4P9	GP4P8	GP4P7	GP4P6	GP4P5	GP4P4	GP4P3	GP4P2	GP4P1	GP4P0
R-0															

LEGEND: R = Read only; -*n* = value after reset

Figure 17-27. GPIO Banks 6 and 7 Input Data Register (IN_DATA67)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
GP7P15	GP7P14	GP7P13	GP7P12	GP7P11	GP7P10	GP7P9	GP7P8	GP7P7	GP7P6	GP7P5	GP7P4	GP7P3	GP7P2	GP7P1	GP7P0
R-0															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GP6P15	GP6P14	GP6P13	GP6P12	GP6P11	GP6P10	GP6P9	GP6P8	GP6P7	GP6P6	GP6P5	GP6P4	GP6P3	GP6P2	GP6P1	GP6P0
R-0															

LEGEND: R = Read only; -*n* = value after reset

Copyright © 2013–2016, Texas Instruments Incorporated

Figure 17-28. GPIO Bank 8 Input Data Register (IN_DATA8)

31															16
Reserved															
R-0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GP8P15	GP8P14	GP8P13	GP8P12	GP8P11	GP8P10	GP8P9	GP8P8	GP8P7	GP8P6	GP8P5	GP8P4	GP8P3	GP8P2	GP8P1	GP8P0
R-0															

LEGEND: R = Read only; -*n* = value after reset

Table 17-9. GPIO Input Data Register (IN_DATA*n*) Field Descriptions

Bit	Field	Value	Description
31-0	GP*k*P*j*		Status of pin GP*k*[*j*]. Reading the GP*k*P*j* bit returns the state of pin *j* in GPIO bank *k*.
		0	GP*k*[*j*] is logic low.
		1	GP*k*[*j*] is logic high.

Copyright © 2013–2016, Texas Instruments Incorporated

17.3.8 GPIO Set Rising Edge Interrupt Registers (SET_RIS_TRIG*n*)

The GPIO set rising edge trigger interrupt register (SET_RIS_TRIG*n*) enables a rising edge trigger on the GPIO pin to generate a GPIO interrupt. The GPIO set rising edge interrupt register (SET_RIS_TRIG01) is shown in Figure 17-29, SET_RIS_TRIG23 is shown in Figure 17-30, SET_RIS_TRIG45 is shown in Figure 17-31, SET_RIS_TRIG67 is shown in Figure 17-32, SET_RIS_TRIG8 is shown in Figure 17-33, and described in Table 17-10. See Table 17-1 to determine the SET_RIS_TRIG*n* bit associated with each GPIO bank and pin number.

Figure 17-29. GPIO Banks 0 and 1 Set Rise Trigger Register (SET_RIS_TRIG01)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
GP1P15	GP1P14	GP1P13	GP1P12	GP1P11	GP1P10	GP1P9	GP1P8	GP1P7	GP1P6	GP1P5	GP1P4	GP1P3	GP1P2	GP1P1	GP1P0
R/W-0															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GP0P15	GP0P14	GP0P13	GP0P12	GP0P11	GP0P10	GP0P9	GP0P8	GP0P7	GP0P6	GP0P5	GP0P4	GP0P3	GP0P2	GP0P1	GP0P0
R/W-0															

LEGEND: R/W = Read/Write; -*n* = value after reset

Figure 17-30. GPIO Banks 2 and 3 Set Rise Trigger Register (SET_RIS_TRIG23)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
GP3P15	GP3P14	GP3P13	GP3P12	GP3P11	GP3P10	GP3P9	GP3P8	GP3P7	GP3P6	GP3P5	GP3P4	GP3P3	GP3P2	GP3P1	GP3P0
R/W-0															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GP2P15	GP2P14	GP2P13	GP2P12	GP2P11	GP2P10	GP2P9	GP2P8	GP2P7	GP2P6	GP2P5	GP2P4	GP2P3	GP2P2	GP2P1	GP2P0
R/W-0															

LEGEND: R/W = Read/Write; -*n* = value after reset

Figure 17-31. GPIO Banks 4 and 5 Set Rise Trigger Register (SET_RIS_TRIG45)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
GP5P15	GP5P14	GP5P13	GP5P12	GP5P11	GP5P10	GP5P9	GP5P8	GP5P7	GP5P6	GP5P5	GP5P4	GP5P3	GP5P2	GP5P1	GP5P0
R/W-0															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GP4P15	GP4P14	GP4P13	GP4P12	GP4P11	GP4P10	GP4P9	GP4P8	GP4P7	GP4P6	GP4P5	GP4P4	GP4P3	GP4P2	GP4P1	GP4P0
R/W-0															

LEGEND: R/W = Read/Write; -*n* = value after reset

Figure 17-32. GPIO Banks 6 and 7 Set Rise Trigger Register (SET_RIS_TRIG67)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
GP7P15	GP7P14	GP7P13	GP7P12	GP7P11	GP7P10	GP7P9	GP7P8	GP7P7	GP7P6	GP7P5	GP7P4	GP7P3	GP7P2	GP7P1	GP7P0
R/W-0															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GP6P15	GP6P14	GP6P13	GP6P12	GP6P11	GP6P10	GP6P9	GP6P8	GP6P7	GP6P6	GP6P5	GP6P4	GP6P3	GP6P2	GP6P1	GP6P0
R/W-0															

LEGEND: R/W = Read/Write; -*n* = value after reset

Copyright © 2013–2016, Texas Instruments Incorporated

Figure 17-33. GPIO Bank 8 Set Rise Trigger Register (SET_RIS_TRIG8)

31															16
Reserved															
R/W-0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GP8P15	GP8P14	GP8P13	GP8P12	GP8P11	GP8P10	GP8P9	GP8P8	GP8P7	GP8P6	GP8P5	GP8P4	GP8P3	GP8P2	GP8P1	GP8P0
R/W-0															

LEGEND: R/W = Read/Write; -*n* = value after reset

Table 17-10. GPIO Set Rising Edge Trigger Interrupt Register (SET_RIS_TRIG*n*) Field Descriptions

Bit	Field	Value	Description
31-0	GP*j*P*k*		Enable rising edge trigger interrupt detection on GP*k*[*j*]. Reading the GP*k*P*j* bit in either SET_RIS_TRIG*n* or CLR_RIS_TRIG*n* always returns an indication of whether the rising edge interrupt generation function is enabled for pin GP*k*[*j*]. Therefore, this bit will be one in both registers if the function is enabled, and zero in both registers if the function is disabled.
		0	No effect.
		1	Interrupt is caused by a low-to-high transition on GP*k*[*j*].

Copyright © 2013–2016, Texas Instruments Incorporated

17.3.9 GPIO Clear Rising Edge Interrupt Registers (CLR_RIS_TRIG*n*)

The GPIO clear rising edge trigger interrupt register (CLR_RIS_TRIG*n*) disables the rising edge trigger on the GPIO pin to generate a GPIO interrupt. The GPIO clear rising edge interrupt register (CLR_RIS_TRIG01) is shown in Figure 17-34, CLR_RIS_TRIG23 is shown in Figure 17-35, CLR_RIS_TRIG45 is shown in Figure 17-36, CLR_RIS_TRIG67 is shown in Figure 17-37, CLR_RIS_TRIG8 is shown in Figure 17-38, and described in Table 17-11. See Table 17-1 to determine the CLR_RIS_TRIG*n* bit associated with each GPIO bank and pin number.

Figure 17-34. GPIO Banks 0 and 1 Clear Rise Trigger Register (CLR_RIS_TRIG01)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
GP1P15	GP1P14	GP1P13	GP1P12	GP1P11	GP1P10	GP1P9	GP1P8	GP1P7	GP1P6	GP1P5	GP1P4	GP1P3	GP1P2	GP1P1	GP1P0
R/W-0															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GP0P15	GP0P14	GP0P13	GP0P12	GP0P11	GP0P10	GP0P9	GP0P8	GP0P7	GP0P6	GP0P5	GP0P4	GP0P3	GP0P2	GP0P1	GP0P0
R/W-0															

LEGEND: R/W = Read/Write; -*n* = value after reset

Figure 17-35. GPIO Banks 2 and 3 Clear Rise Trigger Register (CLR_RIS_TRIG23)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
GP3P15	GP3P14	GP3P13	GP3P12	GP3P11	GP3P10	GP3P9	GP3P8	GP3P7	GP3P6	GP3P5	GP3P4	GP3P3	GP3P2	GP3P1	GP3P0
R/W-0															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GP2P15	GP2P14	GP2P13	GP2P12	GP2P11	GP2P10	GP2P9	GP2P8	GP2P7	GP2P6	GP2P5	GP2P4	GP2P3	GP2P2	GP2P1	GP2P0
R/W-0															

LEGEND: R/W = Read/Write; -*n* = value after reset

Figure 17-36. GPIO Banks 4 and 5 Clear Rise Trigger Register (CLR_RIS_TRIG45)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
GP5P15	GP5P14	GP5P13	GP5P12	GP5P11	GP5P10	GP5P9	GP5P8	GP5P7	GP5P6	GP5P5	GP5P4	GP5P3	GP5P2	GP5P1	GP5P0
R/W-0															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GP4P15	GP4P14	GP4P13	GP4P12	GP4P11	GP4P10	GP4P9	GP4P8	GP4P7	GP4P6	GP4P5	GP4P4	GP4P3	GP4P2	GP4P1	GP4P0
R/W-0															

LEGEND: R/W = Read/Write; -*n* = value after reset

Figure 17-37. GPIO Banks 6 and 7 Clear Rise Trigger Register (CLR_RIS_TRIG67)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
GP7P15	GP7P14	GP7P13	GP7P12	GP7P11	GP7P10	GP7P9	GP7P8	GP7P7	GP7P6	GP7P5	GP7P4	GP7P3	GP7P2	GP7P1	GP7P0
R/W-0															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GP6P15	GP6P14	GP6P13	GP6P12	GP6P11	GP6P10	GP6P9	GP6P8	GP6P7	GP6P6	GP6P5	GP6P4	GP6P3	GP6P2	GP6P1	GP6P0
R/W-0															

LEGEND: R/W = Read/Write; -*n* = value after reset

Copyright © 2013–2016, Texas Instruments Incorporated

Figure 17-38. GPIO Bank 8 Clear Rise Trigger Register (CLR_RIS_TRIG8)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
R/W-0															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GP8P15	GP8P14	GP8P13	GP8P12	GP8P11	GP8P10	GP8P9	GP8P8	GP8P7	GP8P6	GP8P5	GP8P4	GP8P3	GP8P2	GP8P1	GP8P0
R/W-0															

LEGEND: R/W = Read/Write; -*n* = value after reset

Table 17-11. GPIO Clear Rising Edge Interrupt Register (CLR_RIS_TRIG*n*) Field Descriptions

Bit	Field	Value	Description
31-0	GP*k*P*j*		Disable rising edge interrupt detection on GP*k*[*j*]. Reading the GP*k*P*j* bit in either SET_RIS_TRIG*n* or CLR_RIS_TRIG*n* always returns an indication of whether the rising edge interrupt generation function is enabled for GP*k*[*j*]. Therefore, this bit will be one in both registers if the function is enabled, and zero in both registers if the function is disabled.
		0	No effect.
		1	No interrupt is caused by a low-to-high transition on GP*k*[*j*].

Copyright © 2013–2016, Texas Instruments Incorporated

17.3.10 GPIO Set Falling Edge Interrupt Registers (SET_FAL_TRIG*n*)

The GPIO set falling edge trigger interrupt register (SET_FAL_TRIG*n*) enables a falling edge trigger on the GPIO pin to generate a GPIO interrupt. The GPIO set falling edge interrupt register (SET_FAL_TRIG01) is shown in Figure 17-39, SET_FAL_TRIG23 is shown in Figure 17-40, SET_FAL_TRIG45 is shown in Figure 17-41, SET_FAL_TRIG67 is shown in Figure 17-42, SET_FAL_TRIG8 is shown in Figure 17-43, and described in Table 17-12. See Table 17-1 to determine the SET_FAL_TRIG*n* bit associated with each GPIO bank and pin number.

Figure 17-39. GPIO Banks 0 and 1 Set Rise Trigger Register (SET_FAL_TRIG01)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
GP1P15	GP1P14	GP1P13	GP1P12	GP1P11	GP1P10	GP1P9	GP1P8	GP1P7	GP1P6	GP1P5	GP1P4	GP1P3	GP1P2	GP1P1	GP1P0
R/W-0															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GP0P15	GP0P14	GP0P13	GP0P12	GP0P11	GP0P10	GP0P9	GP0P8	GP0P7	GP0P6	GP0P5	GP0P4	GP0P3	GP0P2	GP0P1	GP0P0
R/W-0															

LEGEND: R/W = Read/Write; -*n* = value after reset

Figure 17-40. GPIO Banks 2 and 3 Set Rise Trigger Register (SET_FAL_TRIG23)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
GP3P15	GP3P14	GP3P13	GP3P12	GP3P11	GP3P10	GP3P9	GP3P8	GP3P7	GP3P6	GP3P5	GP3P4	GP3P3	GP3P2	GP3P1	GP3P0
R/W-0															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GP2P15	GP2P14	GP2P13	GP2P12	GP2P11	GP2P10	GP2P9	GP2P8	GP2P7	GP2P6	GP2P5	GP2P4	GP2P3	GP2P2	GP2P1	GP2P0
R/W-0															

LEGEND: R/W = Read/Write; -*n* = value after reset

Figure 17-41. GPIO Banks 4 and 5 Set Rise Trigger Register (SET_FAL_TRIG45)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
GP5P15	GP5P14	GP5P13	GP5P12	GP5P11	GP5P10	GP5P9	GP5P8	GP5P7	GP5P6	GP5P5	GP5P4	GP5P3	GP5P2	GP5P1	GP5P0
R/W-0															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GP4P15	GP4P14	GP4P13	GP4P12	GP4P11	GP4P10	GP4P9	GP4P8	GP4P7	GP4P6	GP4P5	GP4P4	GP4P3	GP4P2	GP4P1	GP4P0
R/W-0															

LEGEND: R/W = Read/Write; -*n* = value after reset

Figure 17-42. GPIO Banks 6 and 7 Set Rise Trigger Register (SET_FAL_TRIG67)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
GP7P15	GP7P14	GP7P13	GP7P12	GP7P11	GP7P10	GP7P9	GP7P8	GP7P7	GP7P6	GP7P5	GP7P4	GP7P3	GP7P2	GP7P1	GP7P0
R/W-0															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GP6P15	GP6P14	GP6P13	GP6P12	GP6P11	GP6P10	GP6P9	GP6P8	GP6P7	GP6P6	GP6P5	GP6P4	GP6P3	GP6P2	GP6P1	GP6P0
R/W-0															

LEGEND: R/W = Read/Write; -*n* = value after reset

Copyright © 2013–2016, Texas Instruments Incorporated

Figure 17-43. GPIO Bank 8 Set Rise Trigger Register (SET_FAL_TRIG8)

31															16
Reserved															
R/W-0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GP8P15	GP8P14	GP8P13	GP8P12	GP8P11	GP8P10	GP8P9	GP8P8	GP8P7	GP8P6	GP8P5	GP8P4	GP8P3	GP8P2	GP8P1	GP8P0
R/W-0															

LEGEND: R/W = Read/Write; -*n* = value after reset

Table 17-12. GPIO Set Falling Edge Trigger Interrupt Register (SET_FAL_TRIG*n*) Field Descriptions

Bit	Field	Value	Description
31-0	GP*k*P*j*		Enable falling edge trigger interrupt detection on GP*k*[*j*]. Reading the GP*k*P*j* bit in either SET_FAL_TRIG*n* or CLR_FAL_TRIG*n* always returns an indication of whether the falling edge interrupt generation function is enabled for pin GP*k*[*j*]. Therefore, this bit will be one in both registers if the function is enabled, and zero in both registers if the function is disabled.
		0	No effect.
		1	Interrupt is caused by a high-to-low transition on GP*k*[*j*].

Copyright © 2013–2016, Texas Instruments Incorporated

17.3.11 GPIO Clear Falling Edge Interrupt Registers (CLR_FAL_TRIG*n*)

The GPIO clear falling edge trigger interrupt register (CLR_FAL_TRIG*n*) disables the falling edge trigger on the GPIO pin to generate a GPIO interrupt. The GPIO clear falling edge interrupt register (CLR_FAL_TRIG01) is shown in Figure 17-44, CLR_FAL_TRIG23 is shown in Figure 17-45, CLR_FAL_TRIG45 is shown in Figure 17-46, CLR_FAL_TRIG67 is shown in Figure 17-47, CLR_FAL_TRIG8 is shown in Figure 17-48, and described in Table 17-13. See Table 17-1 to determine the CLR_FAL_TRIG*n* bit associated with each GPIO bank and pin number.

Figure 17-44. GPIO Banks 0 and 1 Clear Rise Trigger Register (CLR_FAL_TRIG01)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
GP1P15	GP1P14	GP1P13	GP1P12	GP1P11	GP1P10	GP1P9	GP1P8	GP1P7	GP1P6	GP1P5	GP1P4	GP1P3	GP1P2	GP1P1	GP1P0
R/W-0															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GP0P15	GP0P14	GP0P13	GP0P12	GP0P11	GP0P10	GP0P9	GP0P8	GP0P7	GP0P6	GP0P5	GP0P4	GP0P3	GP0P2	GP0P1	GP0P0
R/W-0															

LEGEND: R/W = Read/Write; -*n* = value after reset

Figure 17-45. GPIO Banks 2 and 3 Clear Rise Trigger Register (CLR_FAL_TRIG23)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
GP3P15	GP3P14	GP3P13	GP3P12	GP3P11	GP3P10	GP3P9	GP3P8	GP3P7	GP3P6	GP3P5	GP3P4	GP3P3	GP3P2	GP3P1	GP3P0
R/W-0															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GP2P15	GP2P14	GP2P13	GP2P12	GP2P11	GP2P10	GP2P9	GP2P8	GP2P7	GP2P6	GP2P5	GP2P4	GP2P3	GP2P2	GP2P1	GP2P0
R/W-0															

LEGEND: R/W = Read/Write; -*n* = value after reset

Figure 17-46. GPIO Banks 4 and 5 Clear Rise Trigger Register (CLR_FAL_TRIG45)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
GP5P15	GP5P14	GP5P13	GP5P12	GP5P11	GP5P10	GP5P9	GP5P8	GP5P7	GP5P6	GP5P5	GP5P4	GP5P3	GP5P2	GP5P1	GP5P0
R/W-0															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GP4P15	GP4P14	GP4P13	GP4P12	GP4P11	GP4P10	GP4P9	GP4P8	GP4P7	GP4P6	GP4P5	GP4P4	GP4P3	GP4P2	GP4P1	GP4P0
R/W-0															

LEGEND: R/W = Read/Write; -*n* = value after reset

Figure 17-47. GPIO Banks 6 and 7 Clear Rise Trigger Register (CLR_FAL_TRIG67)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
GP7P15	GP7P14	GP7P13	GP7P12	GP7P11	GP7P10	GP7P9	GP7P8	GP7P7	GP7P6	GP7P5	GP7P4	GP7P3	GP7P2	GP7P1	GP7P0
R/W-0															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GP6P15	GP6P14	GP6P13	GP6P12	GP6P11	GP6P10	GP6P9	GP6P8	GP6P7	GP6P6	GP6P5	GP6P4	GP6P3	GP6P2	GP6P1	GP6P0
R/W-0															

LEGEND: R/W = Read/Write; -*n* = value after reset

Copyright © 2013–2016, Texas Instruments Incorporated

Figure 17-48. GPIO Bank 8 Clear Rise Trigger Register (CLR_FAL_TRIG8)

31															16
Reserved															
R/W-0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GP8P15	GP8P14	GP8P13	GP8P12	GP8P11	GP8P10	GP8P9	GP8P8	GP8P7	GP8P6	GP8P5	GP8P4	GP8P3	GP8P2	GP8P1	GP8P0
R/W-0															

LEGEND: R/W = Read/Write; -*n* = value after reset

Table 17-13. GPIO Clear Falling Edge Interrupt Register (CLR_FAL_TRIG*n*) Field Descriptions

Bit	Field	Value	Description
31-0	GP*k*P*j*		Disable falling edge interrupt detection on GP*k*[*j*]. Reading the GP*k*P*j* bit in either SET_FAL_TRIG*n* or CLR_FAL_TRIG*n* always returns an indication of whether the falling edge interrupt generation function is enabled for GP*k*[*j*]. Therefore, this bit will be one in both registers if the function is enabled, and zero in both registers if the function is disabled.
		0	No effect.
		1	No interrupt is caused by a high-to-low transition on GP*k*[*j*].

Copyright © 2013–2016, Texas Instruments Incorporated

17.3.12 GPIO Interrupt Status Registers (INTSTAT*n*)

The status of GPIO interrupt events can be monitored by reading the GPIO interrupt status register (INTSTAT*n*). In the associated bit position, pending GPIO interrupts are indicated with a logic 1 and GPIO interrupts that are not pending are indicated with a logic 0. The GPIO interrupt status register (INTSTAT01) is shown in Figure 17-49, INTSTAT23 is shown in Figure 17-50, INTSTAT45 is shown in Figure 17-51, INTSTAT67 is shown in Figure 17-52, INTSTAT8 is shown in Figure 17-53, and described in Table 17-14. See Table 17-1 to determine the INTSTAT*n* bit associated with each GPIO bank and pin number.

Figure 17-49. GPIO Banks 0 and 1 Interrupt Status Register (INTSTAT01)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
GP1P15	GP1P14	GP1P13	GP1P12	GP1P11	GP1P10	GP1P9	GP1P8	GP1P7	GP1P6	GP1P5	GP1P4	GP1P3	GP1P2	GP1P1	GP1P0
R/W1C-0															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GP0P15	GP0P14	GP0P13	GP0P12	GP0P11	GP0P10	GP0P9	GP0P8	GP0P7	GP0P6	GP0P5	GP0P4	GP0P3	GP0P2	GP0P1	GP0P0
R/W1C-0															

LEGEND: R/W = Read/Write; W1C = Write 1 to clear bit (writing 0 has no effect); -*n* = value after reset

Figure 17-50. GPIO Banks 2 and 3 Interrupt Status Register (INTSTAT23)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
GP3P15	GP3P14	GP3P13	GP3P12	GP3P11	GP3P10	GP3P9	GP3P8	GP3P7	GP3P6	GP3P5	GP3P4	GP3P3	GP3P2	GP3P1	GP3P0
R/W1C-0															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GP2P15	GP2P14	GP2P13	GP2P12	GP2P11	GP2P10	GP2P9	GP2P8	GP2P7	GP2P6	GP2P5	GP2P4	GP2P3	GP2P2	GP2P1	GP2P0
R/W1C-0															

LEGEND: R/W = Read/Write; W1C = Write 1 to clear bit (writing 0 has no effect); -*n* = value after reset

Figure 17-51. GPIO Banks 4 and 5 Interrupt Status Register (INTSTAT45)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
GP5P15	GP5P14	GP5P13	GP5P12	GP5P11	GP5P10	GP5P9	GP5P8	GP5P7	GP5P6	GP5P5	GP5P4	GP5P3	GP5P2	GP5P1	GP5P0
R/W1C-0															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GP4P15	GP4P14	GP4P13	GP4P12	GP4P11	GP4P10	GP4P9	GP4P8	GP4P7	GP4P6	GP4P5	GP4P4	GP4P3	GP4P2	GP4P1	GP4P0
R/W1C-0															

LEGEND: R/W = Read/Write; W1C = Write 1 to clear bit (writing 0 has no effect); -*n* = value after reset

Figure 17-52. GPIO Banks 6 and 7 Interrupt Status Register (INTSTAT67)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
GP7P15	GP7P14	GP7P13	GP7P12	GP7P11	GP7P10	GP7P9	GP7P8	GP7P7	GP7P6	GP7P5	GP7P4	GP7P3	GP7P2	GP7P1	GP7P0
R/W1C-0															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GP6P15	GP6P14	GP6P13	GP6P12	GP6P11	GP6P10	GP6P9	GP6P8	GP6P7	GP6P6	GP6P5	GP6P4	GP6P3	GP6P2	GP6P1	GP6P0
R/W1C-0															

LEGEND: R/W = Read/Write; W1C = Write 1 to clear bit (writing 0 has no effect); -*n* = value after reset

Copyright © 2013–2016, Texas Instruments Incorporated

Figure 17-53. GPIO Bank 8 Interrupt Status Register (INTSTAT8)

31															16
Reserved															
R/W1C-0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GP8P15	GP8P14	GP8P13	GP8P12	GP8P11	GP8P10	GP8P9	GP8P8	GP8P7	GP8P6	GP8P5	GP8P4	GP8P3	GP8P2	GP8P1	GP8P0
R/W1C-0															

LEGEND: R/W = Read/Write; W1C = Write 1 to clear bit (writing 0 has no effect); -*n* = value after reset

Table 17-14. GPIO Interrupt Status Register (INTSTAT*n*) Field Descriptions

Bit	Field	Value	Description
31-0	GP*k*P*j*		Interrupt status of GP*k*[*j*]. The GP*k*P*j* bit is used to monitor pending GPIO interrupts on pin *j* of GPIO bank *k*. Write a 1 to the GP*k*P*j* bit to clear the status bit; a write of 0 has no effect.
		0	No pending interrupt on GP*k*[*j*].
		1	Pending interrupt on GP*k*[*j*].

Copyright © 2013–2016, Texas Instruments Incorporated

Inter-Integrated Circuit (I2C) Module

This chapter describes the inter-integrated circuit (I2C) peripheral. The scope of this chapter assumes that you are familiar with the Philips Semiconductors Inter-IC bus (I2C-bus) specification version 2.1.

Topic **Page**

18.1 Introduction **544**
18.2 Architecture **546**
18.3 Registers **558**

Copyright © 2013–2016, Texas Instruments Incorporated

18.1 Introduction

18.1.1 Purpose of the Peripheral

The I2C peripheral provides an interface between the SoC and other devices that are compliant with the I2C-bus specification and connected by way of an I2C-bus. External components that are attached to this two-wire serial bus can transmit and receive data that is up to eight bits wide both to and from the SoC through the I2C peripheral.

18.1.2 Features

The I2C peripheral has the following features:

- Compliance with the Philips Semiconductors I2C-bus specification (version 2.1):
  - Support for byte format transfer
  - 7-bit and 10-bit addressing modes
  - General call
  - START byte mode
  - Support for multiple master-transmitters and slave-receivers mode
  - Support for multiple slave-transmitters and master-receivers mode
  - Combined master transmit/receive and receive/transmit mode
  - I2C data transfer rate of from 10 kbps up to 400 kbps (Philips I2C rate)
- 2-bit to 8-bit format transfer
- Free data format mode
- One read DMA event and one write DMA event that the DMA can use
- Seven interrupts that the CPU can use
- Peripheral enable/disable capability

18.1.2.1 Features Not Supported

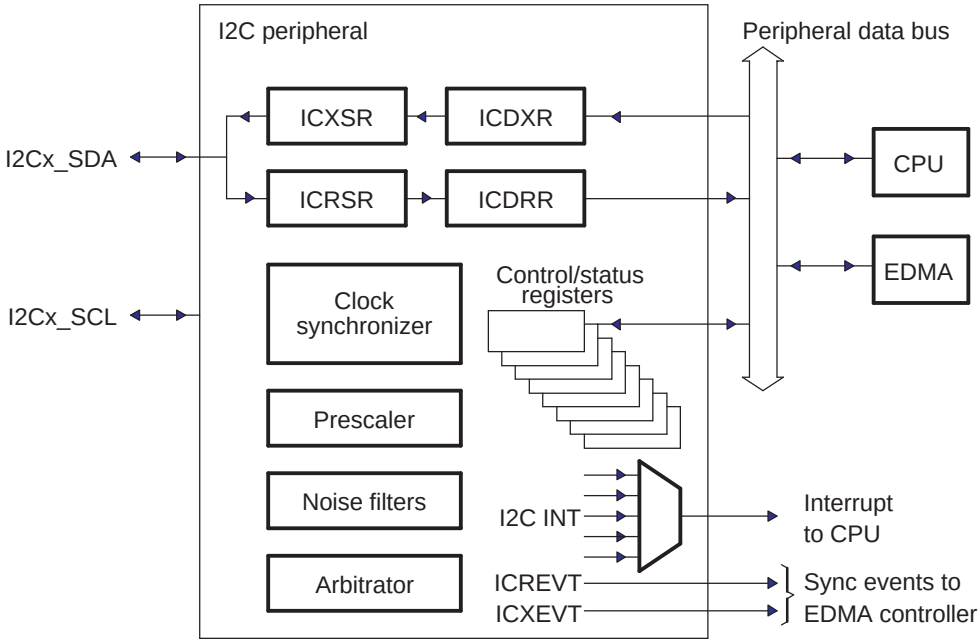
- High-speed mode
- CBUS-compatibility mode
- The combined format in 10-bit addressing mode (the I2C sends the slave address the second byte every time it sends the slave address the first byte).

Copyright © 2013–2016, Texas Instruments Incorporated

18.1.3 Functional Block Diagram

A block diagram of the I2C peripheral is shown in Figure 19-1. Refer to Section 18.2 for detailed information about the architecture of the I2C peripheral.

Figure 18-1. I2C Peripheral Block Diagram



18.1.4 Industry Standard(s) Compliance Statement

The I2C peripheral is compliant with the Philips Semiconductors Inter-IC bus (I2C-bus) specification version 2.1.

Copyright © 2013–2016, Texas Instruments Incorporated

18.2 Architecture

The I2C peripheral consists of the following primary blocks:

- A serial interface: one data pin (I2Cx_SDA) and one clock pin (I2Cx_SCL)
- Data registers to temporarily hold receive data and transmit data traveling between the I2Cx_SDA pin and the CPU or the EDMA controller
- Control and status registers
- A peripheral data bus interface to enable the CPU and the EDMA controller to access the I2C peripheral registers
- A clock synchronizer to synchronize the I2C input clock (from the processor clock generator) and the clock on the I2Cx_SCL pin, and to synchronize data transfers with masters of different clock speeds
- A prescaler to divide down the input clock that is driven to the I2C peripheral
- A noise filter on each of the two pins, I2Cx_SDA and I2Cx_SCL
- An arbitrator to handle arbitration between the I2C peripheral (when it is a master) and another master
- Interrupt generation logic, so that an interrupt can be sent to the CPU
- EDMA event generation logic, so that activity in the EDMA controller can be synchronized to data reception and data transmission in the I2C peripheral

Figure 19-1 shows the four registers used for transmission and reception. The CPU or the EDMA controller writes data for transmission to ICDXR and reads received data from ICDRR. When the I2C peripheral is configured as a transmitter, data written to ICDXR is copied to ICXSR and shifted out on the I2Cx_SDA pin one bit a time. When the I2C peripheral is configured as a receiver, received data is shifted into ICRSR and then copied to ICDRR.

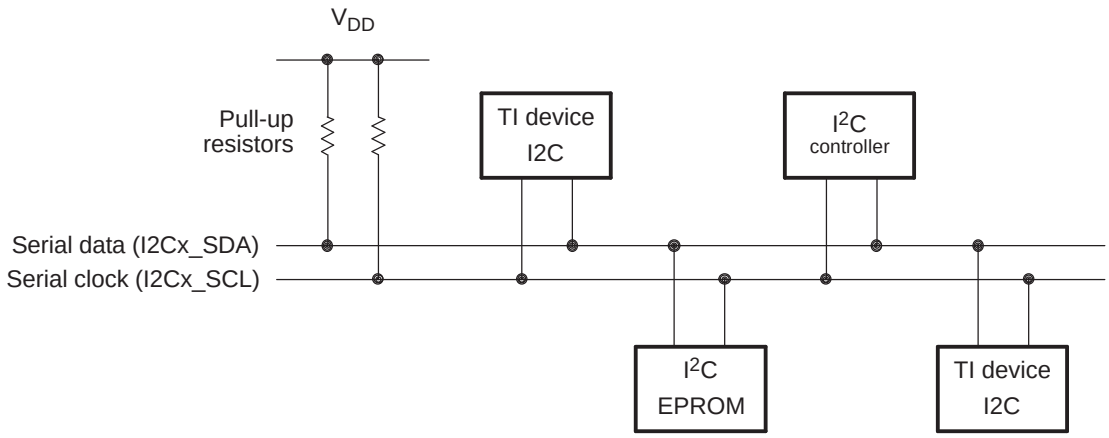
18.2.1 Bus Structure

Figure 19-1 shows how the I2C peripheral is connected to the I2C bus. The I2C bus is a multi-master bus that supports a multi-master mode. This allows more than one device capable of controlling the bus that is connected to it. A unique address recognizes each I2C device. Each I2C device can operate as either transmitter or receiver, depending on the function of the device. Devices that are connected to the I2C bus can be considered a master or slave when performing data transfers, in addition to being a transmitter or receiver.

NOTE: A master device is the device that initiates a data transfer on the bus and generates the clock signals to permit that transfer. Any device that is addressed by this master is considered a slave during this transfer.

An example of multiple I2C modules that are connected for a two-way transfer from one device to other devices is shown in Figure 18-2.

Figure 18-2. Multiple I2C Modules Connected



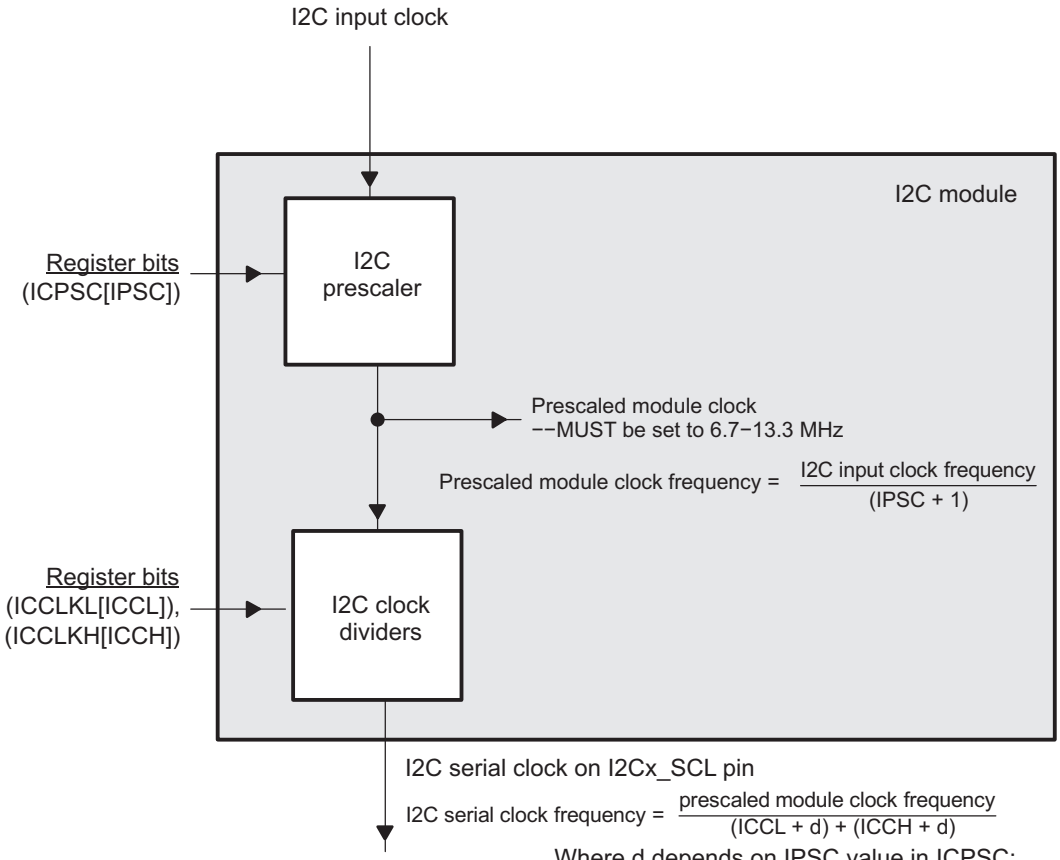
Copyright © 2013–2016, Texas Instruments Incorporated

18.2.2 Clock Generation

As shown in Figure 18-3, I2C input clock is fed to the I2C module. A programmable prescaler (IPSC bit in ICPSC) in the I2C module divides down the I2C input clock to produce a prescaled module clock. The prescaled module clock must be operated within the range of 6.7 to 13.3 MHz. The I2C clock dividers divide-down the high (ICCH bit in ICCLKH) and low portions (ICCL bit in ICCLKL) of the prescaled module clock signal to produce the I2C serial clock, which appears on the I2Cx_SCL pin when the I2C module is configured to be a master on the I2C bus.

The prescaler (IPSC bit in ICPSC) must only be initialized while the I2C module is in the reset state (IRS = 0 in ICMDR). The prescaled frequency only takes effect when the IRS bit in ICMDR is changed to 1. Changing the IPSC bit in ICPSC while IRS = 1 in ICMDR has no effect. Likewise, you must configure the I2C clock dividers (ICCH bit in ICCLKH and ICCL bit in ICCLKL) while the I2C module is still in reset (IRS = 0 in ICMDR).

Figure 18-3. Clocking Diagram for the I2C Peripheral



IPSC value	d
0	7
1	6
2h−FFh	5

CAUTION

Prescaled Module Clock Frequency Range:

The I2C module must be operated with a prescaled module clock frequency of 6.7 to 13.3 MHz. The I2C prescaler register (ICPSC) must be configured to this frequency range.

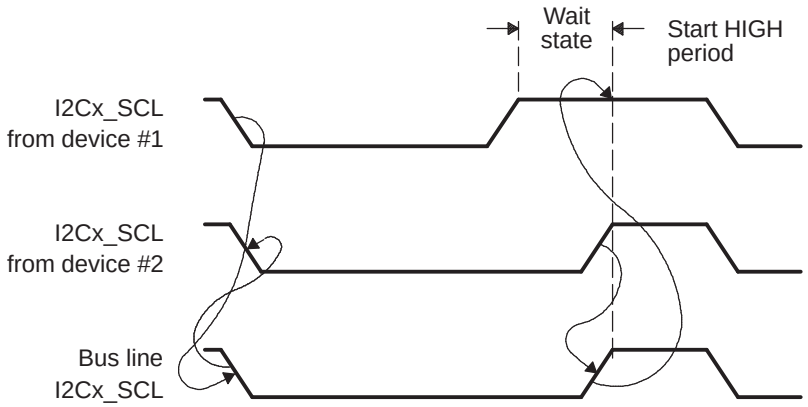
Copyright © 2013–2016, Texas Instruments Incorporated

18.2.3 Clock Synchronization

Only one master device generates the clock signal (I2Cx_SCL) under normal conditions. However, there are two or more masters during the arbitration procedure; and, you must synchronize the clock so that you can compare the data output. Figure 18-4 illustrates the clock synchronization. The wired-AND property of I2Cx_SCL means that a device that first generates a low period on I2Cx_SCL (device #1) overrules the other devices. At this high-to-low transition, the clock generators of the other devices are forced to start their own low period. The I2Cx_SCL is held low by the device with the longest low period. The other devices that finish their low periods must wait for I2Cx_SCL to be released before starting their high periods. A synchronized signal on I2Cx_SCL is obtained, where the slowest device determines the length of the low period and the fastest device determines the length of the high period.

If a device pulls down the clock line for a longer time, the result is that all clock generators must enter the wait state. This way, a slave slows down a fast master and the slow device creates enough time to store a received data word or to prepare a data word that you are going to transmit.

Figure 18-4. Synchronization of Two I2C Clock Generators During Arbitration



18.2.4 Signal Descriptions

The I2C peripheral has a serial data pin (I2Cx_SDA) and a serial clock pin (I2Cx_SCL) for data communication, as shown in Figure 19-1. These two pins carry information between the device and other devices that are connected to the I2C-bus. The I2Cx_SDA and I2Cx_SCL pins both are bi-directional. They each must be connected to a positive supply voltage using a pull-up resistor. When the bus is free, both pins are high. The driver of these two pins has an open-drain configuration to perform the required wired-AND function.

See your device-specific data manual for additional timing and electrical specifications for these pins.

18.2.4.1 Input and Output Voltage Levels

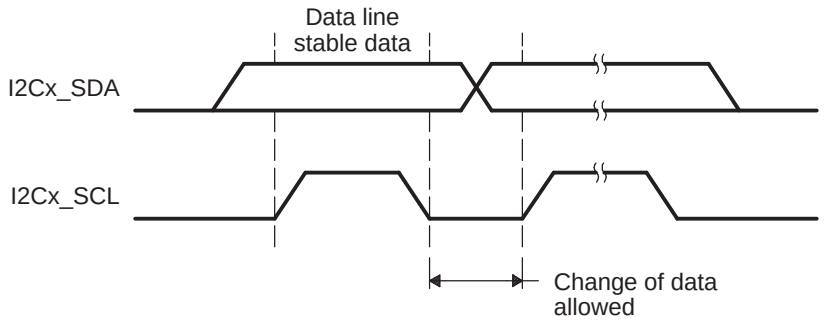
The master device generates one clock pulse for each data bit that is transferred. Due to a variety of different technology devices that can be connected to the I2C-bus, the levels of logic 0 (low) and logic 1 (high) are not fixed and depend on the associated power supply level. See your device-specific data manual for more information.

Copyright © 2013–2016, Texas Instruments Incorporated

18.2.4.2 Data Validity

The data on I2Cx_SDA must be stable during the high period of the clock (see Figure 18-5). The high or low state of the data line, I2Cx_SDA, can change only when the clock signal on I2Cx_SCL is low.

Figure 18-5. Bit Transfer on the I2C-Bus



18.2.5 START and STOP Conditions

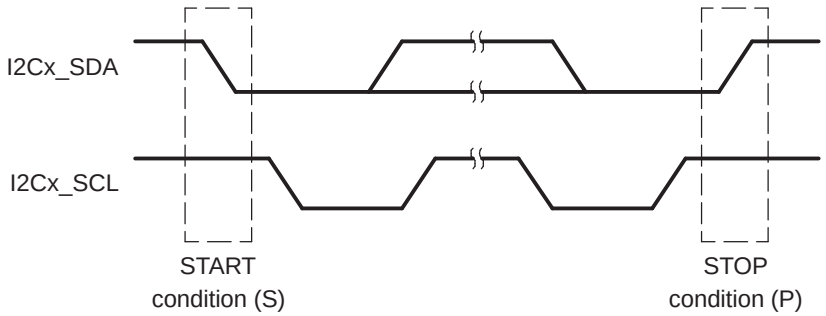
The I2C peripheral can generate START and STOP conditions when the peripheral is configured to be a master on the I2C-bus, as shown in Figure 18-6:

- The START condition is defined as a high-to-low transition on the I2Cx_SDA line while I2Cx_SCL is high. A master drives this condition to indicate the start of a data transfer.
- The STOP condition is defined as a low-to-high transition on the I2Cx_SDA line while I2Cx_SCL is high. A master drives this condition to indicate the end of a data transfer.

The I2C-bus is considered busy after a START condition and before a subsequent STOP condition. The bus busy (BB) bit of ICSTR is 1. The bus is considered free between a STOP condition and the next START condition. The BB is 0.

The master mode (MST) bit and the START condition (STT) bit in ICMDR must both be 1 for the I2C peripheral to start a data transfer with a START condition. The STOP condition (STP) bit must be set to 1 for the I2C peripheral to end a data transfer with a STOP condition. A repeated START condition generates when BB is set to 1 and STT is also set to 1. See Section 18.3.9 for a description of ICMDR (including the MST, STT, and STP bits).

Figure 18-6. I2C Peripheral START and STOP Conditions



Copyright © 2013–2016, Texas Instruments Incorporated

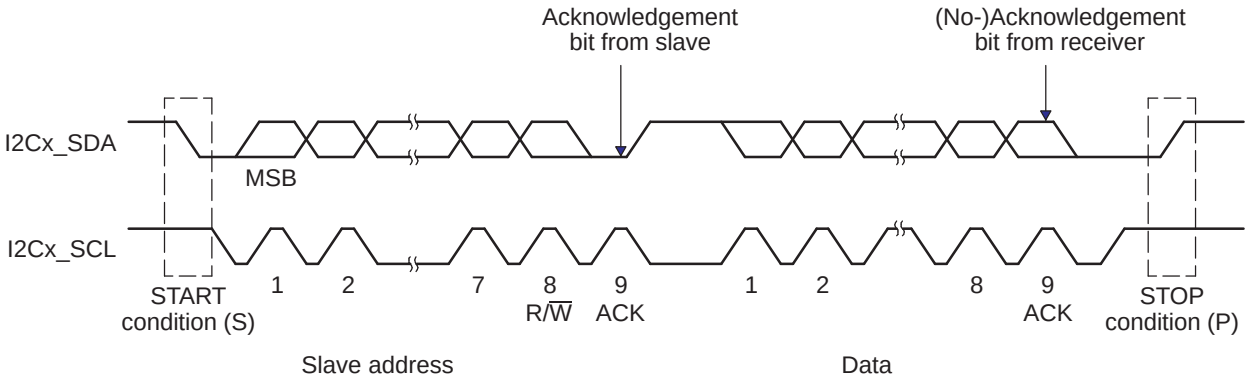
18.2.6 Serial Data Formats

Figure 18-7 shows an example of a data transfer on the I2C-bus. The I2C peripheral supports 1-bit to 8-bit data values. Figure 18-7 is shown in an 8-bit data format (BC = 000 in ICMDR). Each bit put on the I2Cx_SDA line is equivalent to one pulse on the I2Cx_SCL line. The data is always transferred with the most-significant bit (MSB) first. The number of data values that can be transmitted or received is unrestricted; however, the transmitters and receivers must agree on the number of data values being transferred.

The I2C peripheral supports the following data formats:

- 7-bit addressing mode
- 10-bit addressing mode
- Free data format mode

Figure 18-7. I2C Peripheral Data Transfer



18.2.6.1 7-Bit Addressing Format

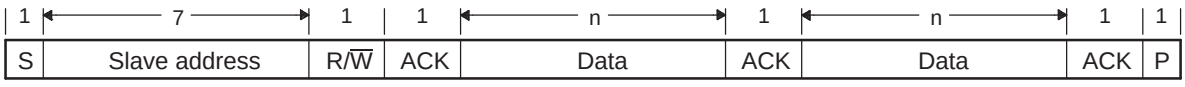
In the 7-bit addressing format (Figure 18-8), the first byte after a START condition (S) consists of a 7-bit slave address followed by a R/$\overline{W}$ bit. The R/$\overline{W}$ bit determines the direction of the data.

- R/$\overline{W}$ = 0: The master writes (transmits) data to the addressed slave.
- R/$\overline{W}$ = 1: The master reads (receives) data from the slave.

An extra clock cycle dedicated for acknowledgment (ACK) is inserted after the R/$\overline{W}$ bit. If the slave inserts the ACK bit, *n* bits of data from the transmitter (master or slave, depending on the R/$\overline{W}$ bit) follow it. *n* is a number from 1 to 8 that the bit count (BC) bits of ICMDR determine. The receiver inserts an ACK bit after the data bits have been transferred.

Write a 0 to the expanded address enable (XA) bit of ICMDR to select the 7-bit addressing format.

Figure 18-8. I2C Peripheral 7-Bit Addressing Format (FDF = 0, XA = 0 in ICMDR)



n = The number of data bits (from 1 to 8) specified by the bit count (BC) field of ICM DR.

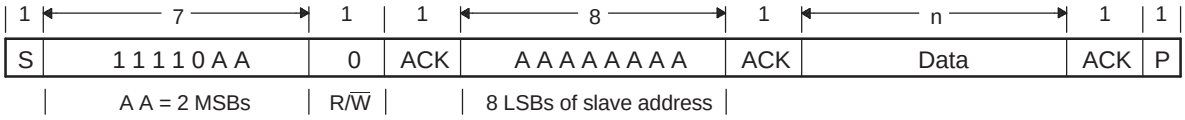
Copyright © 2013–2016, Texas Instruments Incorporated

18.2.6.2 10-Bit Addressing Format

The 10-bit addressing format (Figure 18-9) is like the 7-bit addressing format, but the master sends the slave address in two separate byte transfers. The first byte consists of 11110b, the two MSBs of the 10-bit slave address, and R/$\overline{W}$ = 0 (write). The second byte is the remaining 8 bits of the 10-bit slave address. The slave must send acknowledgment (ACK) after each of the two byte transfers. Once the master has written the second byte to the slave, the master can either write data or use a repeated START condition to change the data direction. (For more information about using 10-bit addressing, see the Philips Semiconductors I2C-bus specification.)

Write 1 to the XA bit of ICMDR to select the 10-bit addressing format.

Figure 18-9. I2C Peripheral 10-Bit Addressing Format With Master-Transmitter Writing to Slave-Receiver (FDF = 0, XA = 1 in ICMDR)



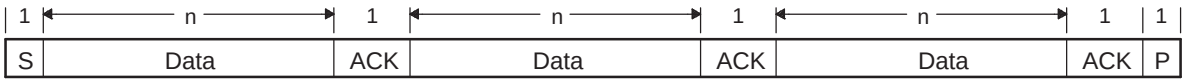
n = The number of data bits (from 1 to 8) specified by the bit count (BC) field of ICMDR.

18.2.6.3 Free Data Format

In the free data format (Figure 18-10), the first bits after a START condition (S) are a data word. An ACK bit is inserted after each data word. The data word can be from 1 to 8 bits, depending on the bit count (BC) bits of ICMDR. No address or data-direction bit is sent. Therefore, the transmitter and the receiver must both support the free data format, and the direction of the data must be constant throughout the transfer.

To select the free data format, write 1 to the free data format (FDF) bit of ICMDR.

Figure 18-10. I2C Peripheral Free Data Format (FDF = 1 in ICMDR)

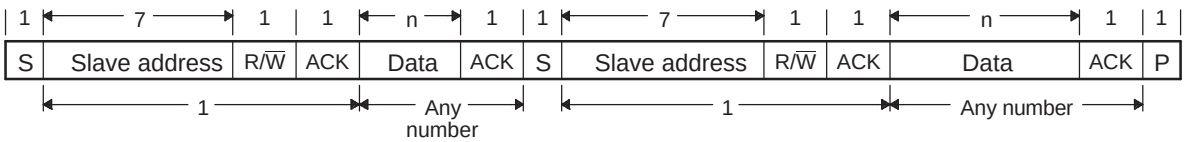


n = The number of data bits (from 1 to 8) specified by the bit count (BC) field of ICMDR.

18.2.6.4 Using a Repeated START Condition

The repeated START condition can be used with the 7-bit addressing, 10-bit addressing, and free data formats. The 7-bit addressing format using a repeated START condition (S) is shown in Figure 18-11. At the end of each data word, the master can drive another START condition. Using this capability, a master can transmit/receive any number of data words before driving a STOP condition. The length of a data word can be from 1 to 8 bits and is selected with the bit count (BC) bits of ICMDR.

Figure 18-11. I2C Peripheral 7-Bit Addressing Format With Repeated START Condition (FDF = 0, XA = 0 in ICMDR)



n = The number of data bits (from 1 to 8) specified by the bit count (BC) field of ICMDR.

Copyright © 2013–2016, Texas Instruments Incorporated

18.2.7 Operating Modes

The I2C peripheral has four basic operating modes to support data transfers as a master and as a slave. See Table 18-1 for the names and descriptions of the modes.

If the I2C peripheral is a master, it begins as a master-transmitter and, typically, transmits an address for a particular slave. When giving data to the slave, the I2C peripheral must remain a master-transmitter. In order to receive data from a slave, the I2C peripheral must be changed to the master-receiver mode.

If the I2C peripheral is a slave, it begins as a slave-receiver and, typically, sends acknowledgment when it recognizes its slave address from a master. If the master will be sending data to the I2C peripheral, the peripheral must remain a slave-receiver. If the master has requested data from the I2C peripheral, the peripheral must be changed to the slave-transmitter mode.

Table 18-1. Operating Modes of the I2C Peripheral

Operating Mode	Description
Slave-receiver mode	The I2C peripheral is a slave and receives data from a master. All slave modules begin in this mode. In this mode, serial data bits received on I2Cx_SDA are shifted in with the clock pulses that are generated by the master. As a slave, the I2C peripheral does not generate the clock signal, but it can hold I2Cx_SCL low while the intervention of the processor is required (RSFULL = 1 in ICSTR) after data has been received.
Slave-transmitter mode	The I2C peripheral is a slave and transmits data to a master. This mode can only be entered from the slave-receiver mode; the I2C peripheral must first receive a command from the master. When you are using any of the 7-bit/10-bit addressing formats, the I2C peripheral enters its slave-transmitter mode if the slave address is the same as its own address (in ICOAR) and the master has transmitted R/$\overline{W}$ = 1. As a slave-transmitter, the I2C peripheral then shifts the serial data out on I2Cx_SDA with the clock pulses that are generated by the master. While a slave, the I2C peripheral does not generate the clock signal, but it can hold I2Cx_SCL low while the intervention of the processor is required (XSMT = 0 in ICSTR) after data has been transmitted.
Master-receiver mode	The I2C peripheral is a master and receives data from a slave. This mode can only be entered from the master-transmitter mode; the I2C peripheral must first transmit a command to the slave. When you are using any of the 7-bit/10-bit addressing formats, the I2C peripheral enters its master-receiver mode after transmitting the slave address and R/$\overline{W}$ = 1. Serial data bits on I2Cx_SDA are shifted into the I2C peripheral with the clock pulses generated by the I2C peripheral on I2Cx_SCL. The clock pulses are inhibited and I2Cx_SCL is held low when the intervention of the processor is required (RSFULL = 1 in ICSTR) after data has been received.
Master-transmitter mode	The I2C peripheral is a master and transmits control information and data to a slave. All master modules begin in this mode. In this mode, data assembled in any of the 7-bit/10-bit addressing formats is shifted out on I2Cx_SDA. The bit shifting is synchronized with the clock pulses generated by the I2C peripheral on I2Cx_SCL. The clock pulses are inhibited and I2Cx_SCL is held low when the intervention of the processor is required (XSMT = 0 in ICSTR) after data has been transmitted.

Copyright © 2013–2016, Texas Instruments Incorporated

18.2.8 NACK Bit Generation

When the I2C peripheral is a receiver (master or slave), it can acknowledge or ignore bits sent by the transmitter. To ignore any new bits, the I2C peripheral must send a no-acknowledge (NACK) bit during the acknowledge cycle on the bus. Table 18-2 summarizes the various ways the I2C peripheral sends a NACK bit.

Table 18-2. Ways to Generate a NACK Bit

I2C Peripheral Condition	NACK Bit Generation	
	Basic	**Optional**
Slave-receiver mode	• Disable data transfers (STT = 0 in ICSTR). • Allow an overrun condition (RSFULL = 1 in ICSTR). • Reset the peripheral (IRS = 0 in ICMDR) .	Set the NACKMOD bit of ICMDR before the rising edge of the last data bit you intend to receive.
Master-receiver mode AND Repeat mode (RM = 1 in ICMDR)	• Generate a STOP condition (STOP = 1 in ICMDR). • Reset the peripheral (IRS = 0 in ICMDR).	Set the NACKMOD bit of ICMDR before the rising edge of the last data bit you intend to receive.
Master-receiver mode AND Nonrepeat mode (RM = 0 in ICMDR)	• If STP = 1 in ICMDR, allow the internal data counter to count down to 0 and force a STOP condition. • If STP = 0, make STP = 1 to generate a STOP condition. • Reset the peripheral (IRS = 0 in ICMDR).	Set the NACKMOD bit of ICMDR before the rising edge of the last data bit you intend to receive.

Copyright © 2013–2016, Texas Instruments Incorporated

18.2.9 Arbitration

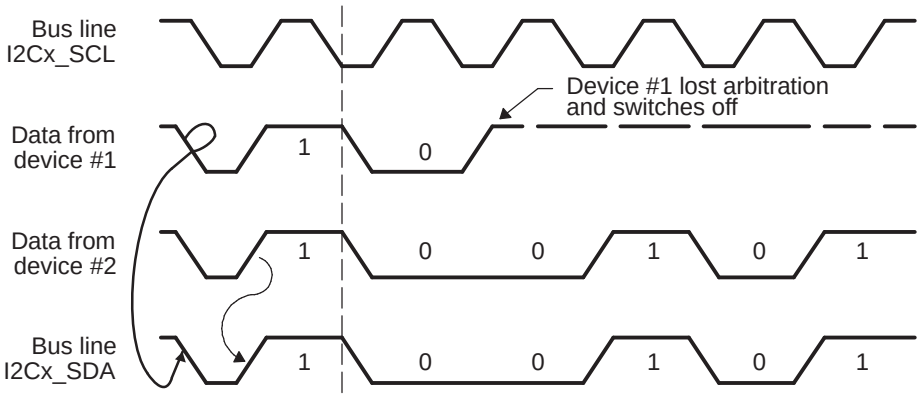
If two or more master-transmitters simultaneously start a transmission on the same bus, an arbitration procedure is invoked. The arbitration procedure uses the data presented on the serial data bus (I2Cx_SDA) by the competing transmitters. Figure 18-12 illustrates the arbitration procedure between two devices. The first master-transmitter, which drives I2Cx_SDA high, is overruled by another master-transmitter that drives I2Cx_SDA low. The arbitration procedure gives priority to the device that transmits the serial data stream with the lowest binary value. Should two or more devices send identical first bytes, arbitration continues on the subsequent bytes.

If the I2C peripheral is the losing master, it switches to the slave-receiver mode, sets the arbitration lost (AL) flag, and generates the arbitration-lost interrupt.

If during a serial transfer the arbitration procedure is still in progress when a repeated START condition or a STOP condition is transmitted to I2Cx_SDA, the master-transmitters involved must send the repeated START condition or the STOP condition at the same position in the format frame. Arbitration is not allowed between:

- A repeated START condition and a data bit
- A STOP condition and a data bit
- A repeated START condition and a STOP condition

Figure 18-12. Arbitration Procedure Between Two Master-Transmitters



Copyright © 2013–2016, Texas Instruments Incorporated

18.2.10 Reset Considerations

The I2C peripheral has two reset sources: software reset and hardware reset.

18.2.10.1 Software Reset Considerations

To reset the I2C peripheral, write 0 to the I2C reset (IRS) bit in the I2C mode register (ICMDR). All status bits in the I2C interrupt status register (ICSTR) are forced to their default values, and the I2C peripheral remains disabled until IRS is changed to 1. The I2Cx_SDA and I2Cx_SCL pins are in the high-impedance state.

NOTE: If the IRS bit is cleared to 0 during a transfer, this can cause the I2C bus to hang (I2Cx_SDA and I2Cx_SCL are in the high-impedance state).

18.2.10.2 Hardware Reset Considerations

When a hardware reset occurs, all the registers of the I2C peripheral are set to their default values and the I2C peripheral remains disabled until the I2C reset (IRS) bit in the I2C mode register (ICMDR) is changed to 1.

NOTE: The IRS bit must be cleared to 0 while you configure/reconfigure the I2C peripheral. Forcing IRS to 0 can be used to save power and to clear error conditions.

18.2.11 Initialization

Proper I2C initialization is required prior to starting communication with other I2C device(s). Unless a fully fledged driver is in place, you need to determine the required I2C configuration needed (for example, Master Receiver, etc.) and configure the I2C controller with the desired settings. Enabling the I2C clock should be the first task. Then the I2C controller is placed in reset. You now are ready to configure the I2C controller. Once configuration is done, you need to enable the I2C controller by releasing the controller from reset. Prior to starting communication, you need to make sure that all status bits are cleared and no pending interrupts exist. Once the bus is determined to be available (the bus is not busy), the I2C is ready to proceed with the desired communication.

Copyright © 2013–2016, Texas Instruments Incorporated

18.2.11.1 Configuring the I2C in Master Receiver Mode and Servicing Receive Data via CPU

The following initialization procedure is for the I2C controller configured in Master Receiver mode. The CPU is used to move data from the I2C receive register to CPU memory (memory accessible by the CPU).

1. Enable I2C clock from the Power and Sleep Controller, if it is driven by the Power and Sleep Controller (see the *Power and Sleep Controller (PSC)* chapter).
2. Place I2C in reset (clear IRS = 0 in ICMDR).
3. Configure ICMDR:
   - Configure I2C as Master (MST = 1).
   - Indicate the I2C configuration to be used; for example, Data Receiver (TRX = 0)
   - Indicate 7-bit addressing is to be used (XA = 0).
   - Disable repeat mode (RM = 0).
   - Disable loopback mode (DLB = 0).
   - Disable free data format (FDF = 0).
   - Optional: Disable start byte mode if addressing a fully fledged I2C device (STB = 0).
   - Set number of bits to transfer to be 8 bits (BC = 0).
4. Configure Slave Address: the I2C device this I2C master would be addressing (ICSAR = 7BIT ADDRESS).
5. Configure the peripheral clock operation frequency (ICPSC). This value should be selected in such a way that the frequency is between 6.7 and 13.3 MHz.
6. Configure I2C master clock frequency:
   - Configure the low-time divider value (ICCLKL).
   - Configure the high-time divider value (ICCLKH).
7. Make sure the interrupt status register (ICSTR) is cleared:
   - Read ICSTR and write it back (write 1 to clear) ICSTR = ICSTR
   - Read ICIVR until it is 0.
8. Take I2C controller out of reset: enable I2C controller (set IRS bit = 1 in ICMDR).
9. Wait until bus busy bit is cleared (BB = 0 in ICSTR).
10. Generate a START event, followed by Slave Address, etc. (set STT = 1 in ICMDR).
11. Wait until data is received (ICRRDY = 1 in ICSTR).
12. Read data:
    - If ICRRDY = 1 in ICSTR, then read ICDRR.
    - Perform the previous two steps until receiving one byte short of the entire byte expecting to receive.
13. Configure the I2C controller not to generate an ACK on the next/final byte reception: set NACKMOD bit for the I2C to generate a NACK on the last byte received (set NACKMOD = 1 in ICMDR).
14. End transfer/release bus when transfer is done. Generate a STOP event (set STP = 1 in ICMDR).

18.2.12 Interrupt Support

The I2C peripheral is capable of interrupting the CPU. The CPU can determine which I2C events caused the interrupt by reading the I2C interrupt vector register (ICIVR). ICIVR contains a binary-coded interrupt vector type to indicate which interrupt has occurred. Reading ICIVR clears the interrupt flag; if other interrupts are pending, a new interrupt is generated. If there is more than one pending interrupt flag, reading ICIVR clears the highest-priority interrupt flag.

Copyright © 2013–2016, Texas Instruments Incorporated

18.2.12.1 Interrupt Events and Requests

The I2C peripheral can generate the interrupts described in Table 18-3. Each interrupt has a flag bit in the I2C interrupt status register (ICSTR) and a mask bit in the interrupt mask register (ICIMR). When one of the specified events occurs, its flag bit is set. If the corresponding mask bit is 0, the interrupt request is blocked; if the mask bit is 1, the request is forwarded to the CPU as an I2C interrupt.

Table 18-3. Descriptions of the I2C Interrupt Events

I2C Interrupt	Initiating Event
Arbitration-lost interrupt (AL)	Generated when the I2C arbitration procedure is lost or illegal START/STOP conditions occur
No-acknowledge interrupt (NACK)	Generated when the master I2C does not receive any acknowledge from the receiver
Registers-ready-for-access interrupt (ARDY)	Generated by the I2C when the previously programmed address, data and command have been performed and the status bits have been updated. This interrupt is used to let the controlling processor know that the I2C registers are ready to be accessed.
Receive interrupt/status (ICRINT and ICRRDY)	Generated when the received data in the receive-shift register (ICRSR) has been copied into the ICDRR. The ICRRDY bit can also be polled by the CPU to read the received data in the ICDRR.
Transmit interrupt/status (ICXINT and ICXRDY)	Generated when the transmitted data has been copied from ICDXR to the transmit-shift register (ICXSR) and shifted out on the I2Cx_SDA pin. This bit can also polled by the CPU to write the next transmitted data into the ICDXR.
Stop-Condition-Detection interrupt (SCD)	Generated when a STOP condition has been detected
Address-as-Slave interrupt (AAS)	Generated when the I2C has recognized its own slave address or an address of all (8) zeros.

18.2.13 DMA Events Generated by the I2C Peripheral

For the EDMA controller to handle transmit and receive data, the I2C peripheral generates the following two EDMA events. Activity in EDMA channels can be synchronized to these events.

- Receive event (ICREVT): When receive data has been copied from the receive shift register (ICRSR) to the data receive register (ICDRR), the I2C peripheral sends an REVT signal to the EDMA controller. In response, the EDMA controller can read the data from ICDRR.
- Transmit event (ICXEVT): When transmit data has been copied from the data transmit register (ICDXR) to the transmit shift register (ICXSR), the I2C peripheral sends an XEVT signal to the EDMA controller. In response, the EDMA controller can write the next transmit data value to ICDXR.

18.2.14 Power Management

The I2C peripheral can be placed in reduced-power modes to conserve power during periods of low activity. The power management of the I2C peripheral is controlled by the processor Power and Sleep Controller (PSC). The PSC acts as a master controller for power management for all of the peripherals on the device. For detailed information on power management procedures using the PSC, see the *Power and Sleep Controller (PSC)* chapter.

18.2.15 Emulation Considerations

The response of the I2C events to emulation suspend events (such as halts and breakpoints) is controlled by the FREE bit in the I2C mode register (ICMDR). The I2C peripheral either stops exchanging data (FREE = 0) or continues to run (FREE = 1) when an emulation suspend event occurs. How the I2C peripheral terminates data transactions is affected by whether the I2C peripheral is acting as a master or a slave. For more information, see the description of the FREE bit in ICMDR (see Section 18.3.9).

Copyright © 2013–2016, Texas Instruments Incorporated

18.3 Registers

Table 20-8 lists the memory-mapped registers for the inter-integrated circuit (I2C) peripheral. See your device-specific data manual for the memory address of these registers. All other register offset addresses not listed in Table 20-8 should be considered as reserved locations and the register contents should not be modified.

Table 18-4. Inter-Integrated Circuit (I2C) Registers

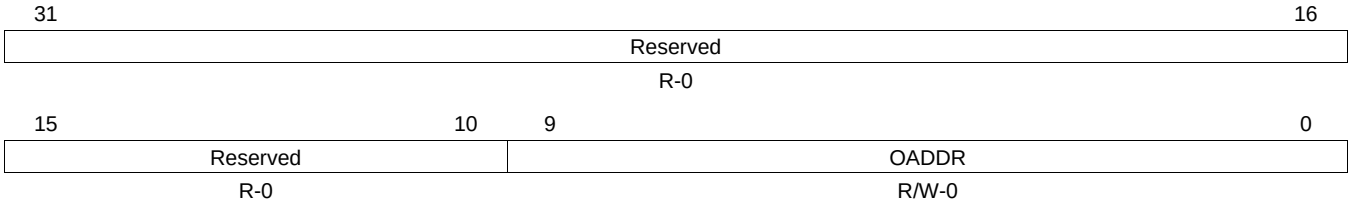
Offset	Acronym	Register Description	Section
0h	ICOAR	I2C Own Address Register	Section 18.3.1
4h	ICIMR	I2C Interrupt Mask Register	Section 18.3.2
8h	ICSTR	I2C Interrupt Status Register	Section 18.3.3
Ch	ICCLKL	I2C Clock Low-Time Divider Register	Section 18.3.4
10h	ICCLKH	I2C Clock High-Time Divider Register	Section 18.3.4
14h	ICCNT	I2C Data Count Register	Section 18.3.5
18h	ICDRR	I2C Data Receive Register	Section 18.3.6
1Ch	ICSAR	I2C Slave Address Register	Section 18.3.7
20h	ICDXR	I2C Data Transmit Register	Section 18.3.8
24h	ICMDR	I2C Mode Register	Section 18.3.9
28h	ICIVR	I2C Interrupt Vector Register	Section 18.3.10
2Ch	ICEMDR	I2C Extended Mode Register	Section 18.3.11
30h	ICPSC	I2C Prescaler Register	Section 18.3.12
34h	REVID1	I2C Revision Identification Register 1	Section 18.3.13
38h	REVID2	I2C Revision Identification Register 2	Section 18.3.13
3Ch	ICDMAC	I2C DMA Control Register	Section 18.3.15
48h	ICPFUNC	I2C Pin Function Register	Section 18.3.16
4Ch	ICPDIR	I2C Pin Direction Register	Section 18.3.17
50h	ICPDIN	I2C Pin Data In Register	Section 18.3.18
54h	ICPDOUT	I2C Pin Data Out Register	Section 18.3.19
58h	ICPDSET	I2C Pin Data Set Register	Section 18.3.20
5Ch	ICPDCLR	I2C Pin Data Clear Register	Section 18.3.21

Copyright © 2013–2016, Texas Instruments Incorporated

18.3.1 I2C Own Address Register (ICOAR)

The I2C own address register (ICOAR) is used to specify its own slave address, which distinguishes it from other slaves connected to the I2C-bus. If the 7-bit addressing mode is selected (XA = 0 in ICMDR), only bits 6-0 are used; bits 9-7 are ignored. ICOAR is shown in Figure 18-13 and described in Table 18-5.

Figure 18-13. I2C Own Address Register (ICOAR)



LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 18-5. I2C Own Address Register (ICOAR) Field Descriptions

Bit	Field	Value	Description
31-10	Reserved	0	These reserved bit locations are always read as zeros. A value written to this field has no effect.
9-0	OADDR	0-3FFh	Own slave address. Provides the slave address of the I2C. In 7-bit addressing mode (XA = 0 in ICMDR): bits 6-0 provide the 7-bit slave address of the I2C. Bits 9-7 are ignored. In 10-bit addressing mode (XA = 1 in ICMDR): bits 9-0 provide the 10-bit slave address of the I2C.

Copyright © 2013–2016, Texas Instruments Incorporated

18.3.2 I2C Interrupt Mask Register (ICIMR)

The I2C interrupt mask register (ICIMR) is used to individually enable or disable I2C interrupt requests. ICIMR is shown in Figure 18-14 and described Table 18-6.

Figure 18-14. I2C Interrupt Mask Register (ICIMR)

31							8
Reserved							
R-0							
7	6	5	4	3	2	1	0
Reserved	AAS	SCD	ICXRDY	ICRRDY	ARDY	NACK	AL
R-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 18-6. I2C Interrupt Mask Register (ICIMR) Field Descriptions

Bit	Field	Value	Description
31-7	Reserved	0	These reserved bit locations are always read as zeros. A value written to this field has no effect.
6	AAS		Address-as-slave interrupt enable bit.
		0	Interrupt request is disabled.
		1	Interrupt request is enabled.
5	SCD		Stop condition detected interrupt enable bit.
		0	Interrupt request is disabled.
		1	Interrupt request is enabled.
4	ICXRDY		Transmit-data-ready interrupt enable bit.
		0	Interrupt request is disabled.
		1	Interrupt request is enabled.
3	ICRRDY		Receive-data-ready interrupt enable bit.
		0	Interrupt request is disabled.
		1	Interrupt request is enabled.
2	ARDY		Register-access-ready interrupt enable bit.
		0	Interrupt request is disabled.
		1	Interrupt request is enabled.
1	NACK		No-acknowledgment interrupt enable bit.
		0	Interrupt request is disabled.
		1	Interrupt request is enabled.
0	AL		Arbitration-lost interrupt enable bit
		0	Interrupt request is disabled.
		1	Interrupt request is enabled.

Copyright © 2013–2016, Texas Instruments Incorporated

18.3.3 I2C Interrupt Status Register (ICSTR)

The I2C interrupt status register (ICSTR) is used to determine which interrupt has occurred and to read status information. ICSTR is shown in Figure 18-15 and described in Table 18-7.

Figure 18-15. I2C Interrupt Status Register (ICSTR)

31							16
Reserved							
R-0							
15	14	13	12	11	10	9	8
Reserved	SDIR	NACKSNT	BB	RSFULL	XSMT	AAS	AD0
R-0	R/W1C-0	R/W1C-0	R/W1C-0	R-0	R-1	R-0	R-0
7	6	5	4	3	2	1	0
Reserved		SCD	ICXRDY	ICRRDY	ARDY	NACK	AL
R-0		R/W1C-0	R/W1C-1	R/W1C-0	R/W1C-0	R/W1C-0	R/W1C-0

LEGEND: R/W = Read/Write; R = Read only; W1C = Write 1 to clear (writing 0 has no effect); -*n* = value after reset

Table 18-7. I2C Interrupt Status Register (ICSTR) Field Descriptions

Bit	Field	Value	Description
31-15	Reserved	0	These reserved bit locations are always read as zeros. A value written to this field has no effect.
14	SDIR		Slave direction bit. In digital-loopback mode (DLB), the SDIR bit is cleared to 0.
		0	I2C is acting as a master-transmitter/receiver or a slave-receiver. SDIR is cleared by one of the following events: • A STOP or a START condition. • SDIR is manually cleared. To clear this bit, write a 1 to it.
		1	I2C is acting as a slave-transmitter.
13	NACKSNT		No-acknowledgment sent bit. NACKSNT bit is used when the I2C is in the receiver mode. One instance in which NACKSNT is affected is when the NACK mode is used (see the description for NACKMOD in Section 18.3.9).
		0	NACK is not sent. NACKSNT is cleared by one of the following events: • It is manually cleared. To clear this bit, write a 1 to it. • The I2C is reset (either when 0 is written to the IRS bit of ICMDR or when the processor is reset).
		1	NACK is sent. A no-acknowledge bit was sent during the acknowledge cycle on the I2C-bus.
12	BB		Bus busy bit. BB bit indicates whether the I2C-bus is busy or is free for another data transfer. In the master mode, BB is controlled by the software.
		0	Bus is free. BB is cleared by one of the following events: • The I2C receives or transmits a STOP bit (bus free). • BB is manually cleared. To clear this bit, write a 1 to it. • The I2C is reset (either when 0 is written to the IRS bit of ICMDR or when the processor is reset).
		1	Bus is busy. When the STT bit in ICMDR is set to 1, a restart condition is generated. BB is set by one of the following events: • The I2C has received or transmitted a START bit on the bus. • I2Cx_SCL is in a low state and the IRS bit in ICMDR is 0.
11	RSFULL		Receive shift register full bit. RSFULL indicates an overrun condition during reception. Overrun occurs when the receive shift register (ICRSR) is full with new data but the previous data has not been read from the data receive register (ICDRR). The new data will not be copied to ICDRR until the previous data is read. As new bits arrive from the I2Cx_SDA pin, they overwrite the bits in ICRSR.
		0	No overrun is detected. RSFULL is cleared by one of the following events: • ICDRR is read. • The I2C is reset (either when 0 is written to the IRS bit of ICMDR or when the processor is reset).
		1	Overrun is detected.

Copyright © 2013–2016, Texas Instruments Incorporated

Table 18-7. I2C Interrupt Status Register (ICSTR) Field Descriptions (continued)

Bit	Field	Value	Description
10	XSMT		Transmit shift register empty bit. XSMT indicates that the transmitter has experienced underflow. Underflow occurs when the transmit shift register (ICXSR) is empty but the data transmit register (ICDXR) has not been loaded since the last ICDXR-to-ICXSR transfer. The next ICDXR-to-ICXSR transfer will not occur until new data is in ICDXR. If new data is not transferred in time, the previous data may be re-transmitted on the I2Cx_SDA pin.
		0	Underflow is detected.
		1	No underflow is detected. XSMT is set by one of the following events: • Data is written to ICDXR. • The I2C is reset (either when 0 is written to the IRS bit of ICMDR or when the processor is reset).
9	AAS		Addressed-as-slave bit.
		0	The AAS bit has been cleared by a repeated START condition or by a STOP condition.
		1	AAS is set by one of the following events: • I2C has recognized its own slave address or an address of all zeros (general call). • The first data word has been received in the free data format (FDF = 1 in ICMDR).
8	AD0		Address 0 bit.
		0	AD0 has been cleared by a START or STOP condition.
		1	An address of all zeros (general call) is detected.
7-6	Reserved	0	These reserved bit locations are always read as zeros. A value written to this field has no effect.
5	SCD		Stop condition detected bit. SCD indicates when a STOP condition has been detected on the I2C bus. The STOP condition could be generated by the I2C or by another I2C device connected to the bus.
		0	No STOP condition has been detected. SCD is cleared by one of the following events: • By reading the INTCODE bits in ICIVR as 110b. • SCD is manually cleared. To clear this bit, write a 1 to it.
		1	A STOP condition has been detected.
4	ICXRDY		Transmit-data-ready interrupt flag bit. ICXRDY indicates that the data transmit register (ICDXR) is ready to accept new data because the previous data has been copied from ICDXR to the transmit shift register (ICXSR). The CPU can poll ICXRDY or use the XRDY interrupt request.
		0	ICDXR is not ready. ICXRDY is cleared by one of the following events: • Data is written to ICDXR. • ICXRDY is manually cleared. To clear this bit, write a 1 to it.
		1	ICDXR is ready. Data has been copied from ICDXR to ICXSR. ICXRDY is forced to 1 when the I2C is reset.
3	ICRRDY		Receive-data-ready interrupt flag bit. ICRRDY indicates that the data receive register (ICDRR) is ready to be read because data has been copied from the receive shift register (ICRSR) to ICDRR. The CPU can poll ICRRDY or use the RRDY interrupt request.
		0	ICDRR is not ready. ICRRDY is cleared by one of the following events: • ICDRR is read. • ICRRDY is manually cleared. To clear this bit, write a 1 to it. • The I2C is reset (either when 0 is written to the IRS bit of ICMDR or when the processor is reset).
		1	ICDRR is ready. Data has been copied from ICRSR to ICDRR.
2	ARDY		Register-access-ready interrupt flag bit (only applicable when the I2C is in the master mode). ARDY indicates that the I2C registers are ready to be accessed because the previously programmed address, data, and command values have been used. The CPU can poll ARDY or use the ARDY interrupt request.
		0	The registers are not ready to be accessed. ARDY is cleared by one of the following events: • The I2C starts using the current register contents. • ARDY is manually cleared. To clear this bit, write a 1 to it. • The I2C is reset (either when 0 is written to the IRS bit of ICMDR or when the processor is reset).
		1	The registers are ready to be accessed. This bit is set after the slave address appears on the I2C bus. • In the nonrepeat mode (RM = 0 in ICMDR): If STP = 0 in ICMDR, ARDY is set when the internal data counter counts down to 0. If STP = 1, ARDY is not affected (instead, the I2C generates a STOP condition when the counter reaches 0). • In the repeat mode (RM = 1): ARDY is set at the end of each data word transmitted from ICDXR.

Copyright © 2013–2016, Texas Instruments Incorporated

Table 18-7. I2C Interrupt Status Register (ICSTR) Field Descriptions (continued)

Bit	Field	Value	Description
1	NACK		No-acknowledgment interrupt flag bit. NACK applies when the I2C is a transmitter (master or slave). NACK indicates whether the I2C has detected an acknowledge bit (ACK) or a no-acknowledge bit (NACK) from the receiver. The CPU can poll NACK or use the NACK interrupt request.
		0	ACK received/NACK is not received. NACK is cleared by one of the following events: • An acknowledge bit (ACK) has been sent by the receiver. • NACK is manually cleared. To clear this bit, write a 1 to it. • The CPU reads the interrupt vector register (ICIVR) when the register contains the code for a NACK interrupt. • The I2C is reset (either when 0 is written to the IRS bit of ICMDR or when the processor is reset).
		1	NACK bit is received. The hardware detects that a no-acknowledge (NACK) bit has been received. **Note:** While the I2C performs a general call transfer, NACK is 1, even if one or more slaves send acknowledgment.
0	AL		Arbitration-lost interrupt flag bit (only applicable when the I2C is a master-transmitter). AL primarily indicates when the I2C has lost an arbitration contest with another master-transmitter. The CPU can poll AL or use the AL interrupt request.
		0	Arbitration is not lost. AL is cleared by one of the following events: • AL is manually cleared. To clear this bit, write a 1 to it. • The CPU reads the interrupt vector register (ICIVR) when the register contains the code for an AL interrupt. • The I2C is reset (either when 0 is written to the IRS bit of ICMDR or when the processor is reset).
		1	Arbitration is lost. AL is set by one of the following events: • The I2C senses that it has lost an arbitration with two or more competing transmitters that started a transmission almost simultaneously. • The I2C attempts to start a transfer while the BB (bus busy) bit is set to 1. When AL is set to 1, the MST and STP bits of ICMDR are cleared, and the I2C becomes a slave-receiver.

Copyright © 2013–2016, Texas Instruments Incorporated

18.3.4 I2C Clock Divider Registers (ICCLKL and ICCLKH)

When the I2C is a master, the prescaled module clock is divided down for use as the I2C serial clock on the I2Cx_SCL pin. The shape of the I2C serial clock depends on two divide-down values, ICCL and ICCH. For detailed information on how these values are programmed, see Section 18.2.2.

18.3.4.1 I2C Clock Low-Time Divider Register (ICCLKL)

For each I2C serial clock cycle, ICCL in the I2C clock low-time divider register (ICCLKL) determines the amount of time the signal is low. ICCLKL must be configured while the I2C is still in reset (IRS = 0 in ICMDR). ICCLKL is shown in Figure 18-16 and described in Table 18-8.

Figure 18-16. I2C Clock Low-Time Divider Register (ICCLKL)

31	16
Reserved	
R-0	
15	**0**
ICCL	
R/W-0	

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 18-8. I2C Clock Low-Time Divider Register (ICCLKL) Field Descriptions

Bit	Field	Value	Description
31-16	Reserved	0	These reserved bit locations are always read as zeros. A value written to this field has no effect.
15-0	ICCL	0-FFFFh	Clock low-time divide-down value of 1-65536. The period of the module clock is multiplied by (ICCL + d) to produce the low-time duration of the I2C serial on the I2Cx_SCL pin.

18.3.4.2 I2C Clock High-Time Divider Register (ICCLKH)

For each I2C serial clock cycle, ICCH in the I2C clock high-time divider register (ICCLKH) determines the amount of time the signal is high. ICCLKH must be configured while the I2C is still in reset (IRS = 0 in ICMDR). ICCLKH is shown in Figure 18-17 and described in Table 18-9.

Figure 18-17. I2C Clock High-Time Divider Register (ICCLKH)

31	16
Reserved	
R-0	
15	**0**
ICCH	
R/W-0	

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 18-9. I2C Clock High-Time Divider Register (ICCLKH) Field Descriptions

Bit	Field	Value	Description
31-16	Reserved	0	These reserved bit locations are always read as zeros. A value written to this field has no effect.
15-0	ICCH	0-FFFFh	Clock high-time divide-down value of 1-65536. The period of the module clock is multiplied by (ICCH + d) to produce the high-time duration of the I2C serial on the I2Cx_SCL pin.

Copyright © 2013–2016, Texas Instruments Incorporated

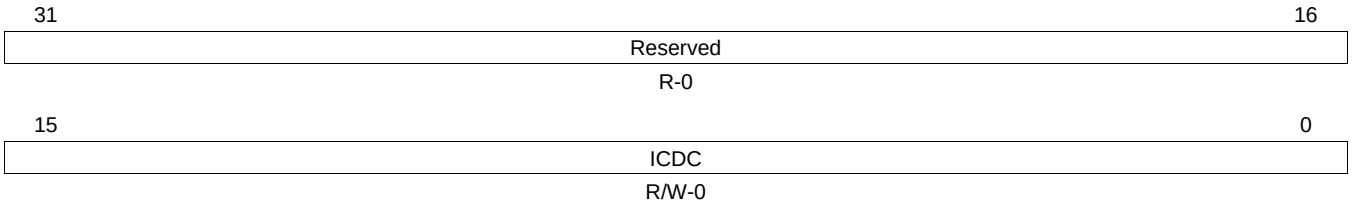
18.3.5 I2C Data Count Register (ICCNT)

The I2C data count register (ICCNT) is used to indicate how many data words to transfer when the I2C is configured as a master-transmitter-receiver (MST = 1 and TRX = 1/0 in ICMDR) and the repeat mode is off (RM = 0 in ICMDR). In the repeat mode (RM = 1), ICCNT is not used.

The value written to ICCNT is copied to an internal data counter. The internal data counter is decremented by 1 for each data word transferred (ICCNT remains unchanged). If a STOP condition is requested (STP = 1 in ICMDR), the I2C terminates the transfer with a STOP condition when the countdown is complete (that is, when the last data word has been transferred).

ICCNT is shown in Figure 18-18 and described in Table 18-10.

Figure 18-18. I2C Data Count Register (ICCNT)



LEGEND: R/W = Read/Write; R = Read only; *-n* = value after reset

Table 18-10. I2C Data Count Register (ICCNT) Field Descriptions

Bit	Field	Value	Description
31-16	Reserved	0	These reserved bit locations are always read as zeros. A value written to this field has no effect.
15-0	ICDC	0-FFFFh	Data count value. When RM = 0 in ICMDR, ICDC indicates the number of data words to transfer in the nonrepeat mode. When RM = 1 in ICMDR, the value in ICCNT is a don't care. If STP = 1 in ICMDR, a STOP condition is generated when the internal data counter counts down to 0.
		0	The start value loaded to the internal data counter is 65536.
		1h-FFFFh	The start value loaded to internal data counter is 1-65535.

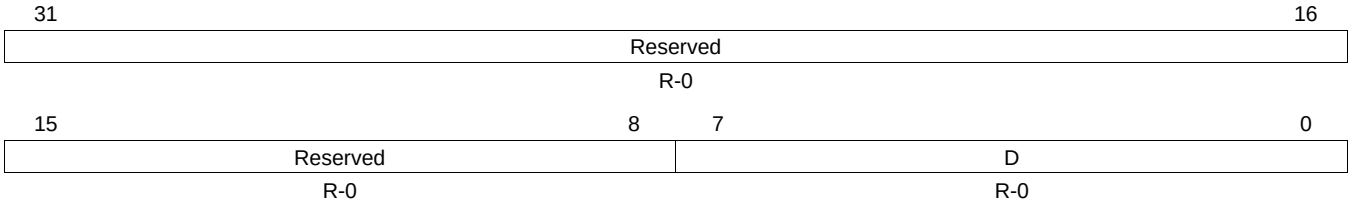
Copyright © 2013–2016, Texas Instruments Incorporated

18.3.6 I2C Data Receive Register (ICDRR)

The I2C data receive register (ICDRR) is used to read the receive data. The ICDRR can receive a data value of up to 8 bits; data values with fewer than 8 bits are right-aligned in the D bits and the remaining D bits are undefined. The number of data bits is selected by the bit count bits (BC) of ICMDR. The I2C receive shift register (ICRSR) shifts in the received data from the I2Cx_SDA pin. Once data is complete, the I2C copies the contents of ICRSR into ICDRR. The CPU and the EDMA controller cannot access ICRSR.

ICDRR is shown in Figure 18-19 and described in Table 18-11.

Figure 18-19. I2C Data Receive Register (ICDRR)



LEGEND: R = Read only; -*n* = value after reset

Table 18-11. I2C Data Receive Register (ICDRR) Field Descriptions

Bit	Field	Value	Description
31-8	Reserved	0	These reserved bit locations are always read as zeros. A value written to this field has no effect.
7-0	D	0-FFh	Receive data.

Copyright © 2013–2016, Texas Instruments Incorporated

18.3.7 I2C Slave Address Register (ICSAR)

The I2C slave address register (ICSAR) contains a 7-bit or 10-bit slave address. When the I2C is not using the free data format (FDF = 0 in ICMDR), it uses this address to initiate data transfers with a slave or slaves. When the address is nonzero, the address is for a particular slave. When the address is 0, the address is a general call to all slaves. If the 7-bit addressing mode is selected (XA = 0 in ICMDR), only bits 6-0 of ICSAR are used; bits 9-7 are ignored.

ICSAR is shown in Figure 18-20 and described in Table 18-12.

Figure 18-20. I2C Slave Address Register (ICSAR)

31	16
Reserved	
R-0	

15	10	9	0
Reserved		SADDR	
R-0		R/W-3FFh	

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 18-12. I2C Slave Address Register (ICSAR) Field Descriptions

Bit	Field	Value	Description
31-10	Reserved	0	These reserved bit locations are always read as zeros. A value written to this field has no effect.
9-0	SADDR	0-3FFh	Slave address. Provides the slave address of the I2C. In 7-bit addressing mode (XA = 0 in ICMDR): bits 6-0 provide the 7-bit slave address that the I2C transmits when it is in the master-transmitter mode. Bits 9-7 are ignored. In 10-bit addressing mode (XA = 1 in ICMDR): Bits 9-0 provide the 10-bit slave address that the I2C transmits when it is in the master-transmitter mode.

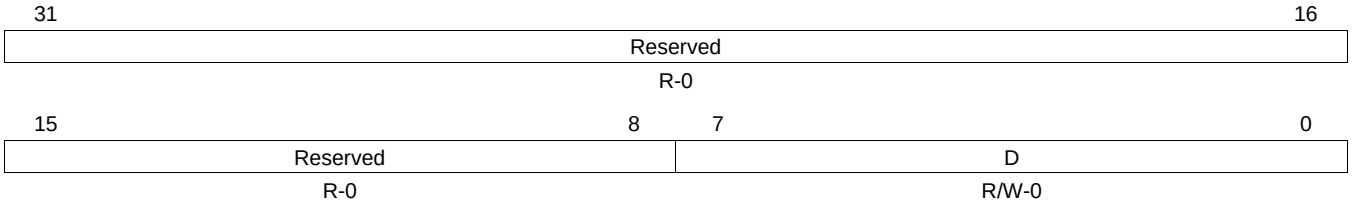
Copyright © 2013–2016, Texas Instruments Incorporated

18.3.8 I2C Data Transmit Register (ICDXR)

The CPU or EDMA writes transmit data to the I2C data transmit register (ICDXR). The ICDXR can accept a data value of up to 8 bits. When writing a data value with fewer than 8 bits, the written data must be right-aligned in the D bits. The number of data bits is selected by the bit count bits (BC) of ICMDR. Once data is written to ICDXR, the I2C copies the contents of ICDXR into the I2C transmit shift register (ICXSR). The ICXSR shifts out the transmit data from the I2Cx_SDA pin. The CPU and the EDMA controller cannot access ICXSR.

ICDXR is shown in Figure 18-21 and described in Table 18-13.

Figure 18-21. I2C Data Transmit Register (ICDXR)



LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 18-13. I2C Data Transmit Register (ICDXR) Field Descriptions

Bit	Field	Value	Description
31-8	Reserved	0	These reserved bit locations are always read as zeros. A value written to this field has no effect.
7-0	D	0-FFh	Transmit data.

Copyright © 2013–2016, Texas Instruments Incorporated

18.3.9 I2C Mode Register (ICMDR)

The I2C mode register (ICMDR) contains the control bits of the I2C. ICMDR is shown in shown in Figure 18-22 and described in Table 18-14.

Figure 18-22. I2C Mode Register (ICMDR)

31							16
Reserved							
R-0							
15	14	13	12	11	10	9	8
NACKMOD	FREE	STT	Reserved	STP	MST	TRX	XA
R/W-0	R/W-0	R/W-0	R-0	R/W-0	R/W-0	R/W-0	R/W-0
7	6	5	4	3	2		0
RM	DLB	IRS	STB	FDF	BC		
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0		

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 18-14. I2C Mode Register (ICMDR) Field Descriptions

Bit	Field	Value	Description
31-16	Reserved	0	These reserved bit locations are always read as zeros. A value written to this field has no effect.
15	NACKMOD		No-acknowledge (NACK) mode bit (only applicable when the I2C is a receiver).
		0	In slave-receiver mode: The I2C sends an acknowledge (ACK) bit to the transmitter during the each acknowledge cycle on the bus. The I2C only sends a no-acknowledge (NACK) bit if you set the NACKMOD bit. In master-receiver mode: The I2C sends an ACK bit during each acknowledge cycle until the internal data counter counts down to 0. When the counter reaches 0, the I2C sends a NACK bit to the transmitter. To have a NACK bit sent earlier, you must set the NACKMOD bit.
		1	In either slave-receiver or master-receiver mode: The I2C sends a NACK bit to the transmitter during the next acknowledge cycle on the bus. Once the NACK bit has been sent, NACKMOD is cleared. To send a NACK bit in the next acknowledge cycle, you must set NACKMOD before the rising edge of the last data bit.
14	FREE		This emulation mode bit is used to determine the state of the I2C when a breakpoint is encountered in the high-level language debugger.
		0	When I2C is master: If I2Cx_SCL is low when the breakpoint occurs, the I2C stops immediately and keeps driving I2Cx_SCL low, whether the I2C is the transmitter or the receiver. If I2Cx_SCL is high, the I2C waits until I2Cx_SCL becomes low and then stops. When I2C is slave: A breakpoint forces the I2C to stop when the current transmission/reception is complete.
		1	The I2C runs free; that is, it continues to operate when a breakpoint occurs.
13	STT		START condition bit (only applicable when the I2C is a master). The RM, STT, and STP bits determine when the I2C starts and stops data transmissions (see Table 18-15). Note that the STT and STP bits can be used to terminate the repeat mode.
		0	In master mode, STT is automatically cleared after the START condition has been generated. In slave mode, if STT is 0, the I2C does not monitor the bus for commands from a master. As a result, the I2C performs no data transfers.
		1	In master mode, setting STT to 1 causes the I2C to generate a START condition on the I2C-bus. In slave mode, if STT is 1, the I2C monitors the bus and transmits/receives data in response to commands from a master.
12	Reserved	0	These reserved bit locations are always read as zeros. A value written to this field has no effect.
11	STP		STOP condition bit (only applicable when the I2C is a master). The RM, STT, and STP bits determine when the I2C starts and stops data transmissions (see Table 18-15). Note that the STT and STP bits can be used to terminate the repeat mode.
		0	STP is automatically cleared after the STOP condition has been generated.
		1	STP has been set to generate a STOP condition when the internal data counter of the I2C counts down to 0.

Copyright © 2013–2016, Texas Instruments Incorporated

Table 18-14. I2C Mode Register (ICMDR) Field Descriptions (continued)

Bit	Field	Value	Description
10	MST		Master mode bit. MST determines whether the I2C is in the slave mode or the master mode. MST is automatically changed from 1 to 0 when the I2C master generates a STOP condition. See Table 18-16.
		0	Slave mode. The I2C is a slave and receives the serial clock from the master.
		1	Master mode. The I2C is a master and generates the serial clock on the I2Cx_SCL pin.
9	TRX		Transmitter mode bit. When relevant, TRX selects whether the I2C is in the transmitter mode or the receiver mode. Table 18-16 summarizes when TRX is used and when it is a don't care.
		0	Receiver mode. The I2C is a receiver and receives data on the I2Cx_SDA pin.
		1	Transmitter mode. The I2C is a transmitter and transmits data on the I2Cx_SDA pin.
8	XA		Expanded address enable bit.
		0	7-bit addressing mode (normal address mode). The I2C transmits 7-bit slave addresses (from bits 6-0 of ICSAR), and its own slave address has 7 bits (bits 6-0 of ICOAR).
		1	10-bit addressing mode (expanded address mode). The I2C transmits 10-bit slave addresses (from bits 9-0 of ICSAR), and its own slave address has 10 bits (bits 9-0 of ICOAR).
7	RM		Repeat mode bit (only applicable when the I2C is a master). The RM, STT, and STP bits determine when the I2C starts and stops data transmissions (see Table 18-15). If the I2C is configured in slave mode, the RM bit is don't care.
		0	Nonrepeat mode. The value in the data count register (ICCNT) determines how many data words are received/transmitted by the I2C.
		1	Repeat mode. Data words are continuously received/transmitted by the I2C until the STP bit is manually set to 1, regardless of the value in ICCNT.
6	DLB		Digital loopback mode bit (only applicable when the I2C is a master-transmitter). This bit disables or enables the digital loopback mode of the I2C. The effects of this bit are shown in Figure 18-23. Note that DLB mode in the free data format mode (DLB = 1 and FDF = 1) is not supported.
		0	Digital loopback mode is disabled.
		1	Digital loopback mode is enabled. In this mode, the MST bit must be set to 1 and data transmitted out of ICDXR is received in ICDRR after n clock cycles by an internal path, where: n = ((I2C input clock frequency/prescaled module clock frequency) × 8) The transmit clock is also the receive clock. The address transmitted on the I2Cx_SDA pin is the address in ICOAR.
5	IRS		I2C reset bit. Note that if IRS is reset during a transfer, it can cause the I2C bus to hang (I2Cx_SDA and I2Cx_SCL are in a high-impedance state).
		0	The I2C is in reset/disabled. When this bit is cleared to 0, all status bits (in ICSTR) are set to their default values.
		1	The I2C is enabled.
4	STB		START byte mode bit (only applicable when the I2C is a master). As described in version 2.1 of the Philips I2C-bus specification, the START byte can be used to help a slave that needs extra time to detect a START condition. When the I2C is a slave, the I2C ignores a START byte from a master, regardless of the value of the STB bit.
		0	The I2C is not in the START byte mode.
		1	The I2C is in the START byte mode. When you set the START condition bit (STT), the I2C begins the transfer with more than just a START condition. Specifically, it generates: 1. A START condition 2. A START byte (0000 0001b) 3. A dummy acknowledge clock pulse 4. A repeated START condition The I2C sends the slave address that is in ICSAR.
3	FDF		Free data format mode bit. Note that DLB mode in the free data format mode (DLB = 1 and FDF = 1) is not supported. See Table 18-16.
		0	Free data format mode is disabled. Transfers use the 7-/10-bit addressing format selected by the XA bit.
		1	Free data format mode is enabled.

Copyright © 2013–2016, Texas Instruments Incorporated

Table 18-14. I2C Mode Register (ICMDR) Field Descriptions (continued)

Bit	Field	Value	Description
2-0	BC	0-7h	Bit count bits. BC defines the number of bits (1 to 8) in the next data word that is to be received or transmitted by the I2C. The number of bits selected with BC must match the data size of the other device. Note that when BC = 0, a data word has 8 bits. If the bit count is less than 8, receive data is right aligned in the D bits of ICDRR and the remaining D bits are undefined. Also, transmit data written to ICDXR must be right aligned.
		0	8 bits per data word
		1h	1 bit per data word
		2h	2 bits per data word
		3h	3 bits per data word
		4h	4 bits per data word
		5h	5 bits per data word
		6h	6 bits per data word
		7h	7 bits per data word

Table 18-15. Master-Transmitter/Receiver Bus Activity Defined by RM, STT, and STP Bits

ICMDR Bit				
RM	STT	STP	Bus Activity[(1)]	Description
0	0	0	None	No activity
0	0	1	P	STOP condition
0	1	0	S-A-D..(*n*)..D	START condition, slave address, *n* data words (*n* = value in ICCNT)
0	1	1	S-A-D..(*n*)..D-P	START condition, slave address, *n* data words, STOP condition (*n* = value in ICCNT)
1	0	0	None	No activity
1	0	1	P	STOP condition
1	1	0	S-A-D-D-D..	Repeat mode transfer: START condition, slave address, continuous data transfers until STOP condition or next START condition
1	1	1	None	Reserved bit combination (No activity

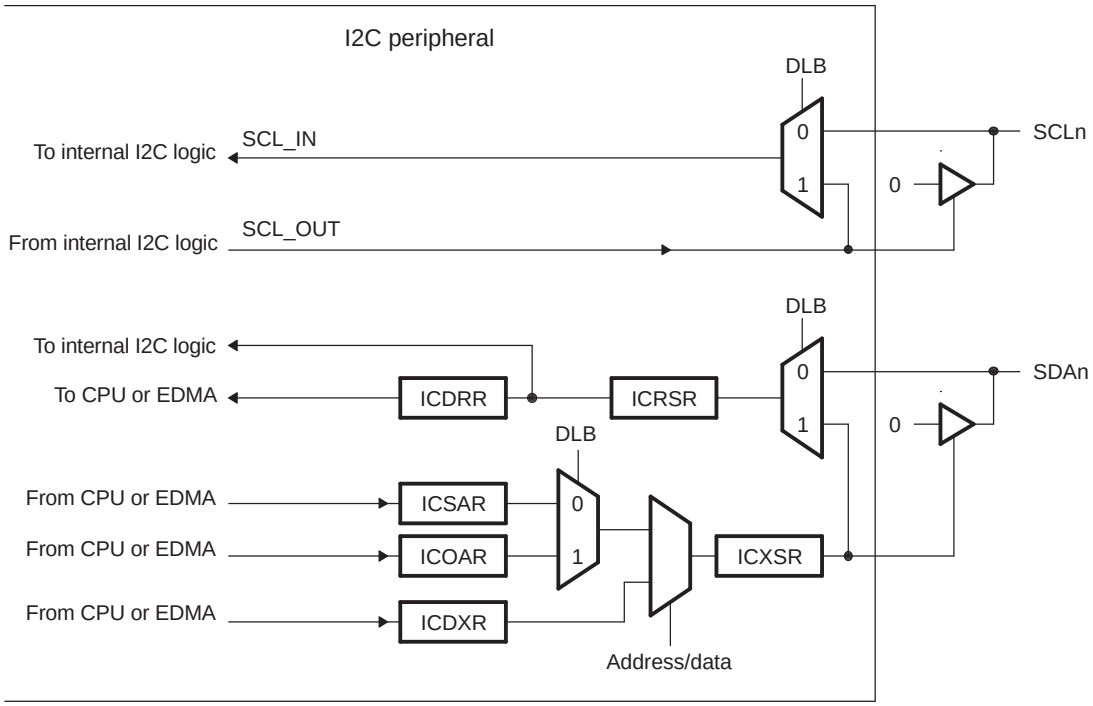
(1) A = Address; D = Data word; P = STOP condition; S = START condition

Table 18-16. How the MST and FDF Bits Affect the Role of TRX Bit

ICMDR Bit			
MST	FDF	I2C State	Function of TRX Bit
0	0	In slave mode but not free data format mode	TRX is a don't care. Depending on the command from the master, the I2C responds as a receiver or a transmitter.
0	1	In slave mode and free data format mode	The free data format mode requires that the transmitter and receiver be fixed. TRX identifies the role of the I2C: TRX = 0: The I2C is a receiver. TRX = 1: The I2C is a transmitter.
1	0	In master mode but not free data format mode	TRX identifies the role of the I2C: TRX = 0: The I2C is a receiver. TRX = 1: The I2C is a transmitter.
1	1	In master mode and free data format mode	The free data format mode requires that the transmitter and receiver be fixed. TRX identifies the role of the I2C: TRX = 0: The I2C is a receiver. TRX = 1: The I2C is a transmitter.

Copyright © 2013–2016, Texas Instruments Incorporated

Figure 18-23. Block Diagram Showing the Effects of the Digital Loopback Mode (DLB) Bit



Copyright © 2013–2016, Texas Instruments Incorporated

18.3.10 I2C Interrupt Vector Register (ICIVR)

The I2C interrupt vector register (ICIVR) is used by the CPU to determine which event generated the I2C interrupt. Reading ICIVR clears the interrupt flag; if other interrupts are pending, a new interrupt is generated. If there are more than one interrupt flag, reading ICIVR clears the highest priority interrupt flag. Note that you must read (clear) ICIVR before doing another start; otherwise, ICIVR could contain an incorrect (old interrupt flags) value.

ICIVR is shown in Figure 18-24 and described in Table 18-17.

Figure 18-24. I2C Interrupt Vector Register (ICIVR)

31	16
Reserved	
R-0	

15	2	0
Reserved	INTCODE	
R-0	R-0	

LEGEND: R= Read only; *-n* = value after reset

Table 18-17. I2C Interrupt Vector Register (ICIVR) Field Descriptions

Bit	Field	Value	Description
31-3	Reserved	0	These reserved bit locations are always read as zeros. A value written to this field has no effect.
2-0	INTCODE	0-7h	Interrupt code bits. The binary code in INTCODE indicates which event generated an I2C interrupt.
		0	None
		1h	Arbitration-lost interrupt (AL). Highest priority if multiple I2C interrupts are pending.
		2h	No-acknowledgment interrupt (NACK)
		3h	Register-access-ready interrupt (ARDY)
		4h	Receive-data-ready interrupt (ICRRDY)
		5h	Transmit-data-ready interrupt (ICXRDY)
		6h	Stop condition detected interrupt (SCD)
		7h	Address-as-slave interrupt (AAS). Lowest priority if multiple I2C interrupts are pending.

Copyright © 2013–2016, Texas Instruments Incorporated

18.3.11 I2C Extended Mode Register (ICEMDR)

The I2C extended mode register (ICEMDR) is used to indicate which condition generates a transmit data ready interrupt. ICEMDR is shown in Figure 18-25 and described in Table 18-18.

Figure 18-25. I2C Extended Mode Register (ICEMDR)

31	16
Reserved	
R-0	

15	1	0
Reserved	IGNACK	BCM
R-0	R/W-0	R/W-1

LEGEND: R/W = Read/Write; R= Read only; -*n* = value after reset

Table 18-18. I2C Extended Mode Register (ICEMDR) Field Descriptions

Bit	Field	Value	Description
31-2	Reserved	0	These reserved bit locations are always read as zeros. A value written to this field has no effect.
1	IGNACK		Ignore NACK mode.
		0	Master transmitter operates normally, that is, it discontinues the data transfer and sets the ARDY and NACK bits in ICSTR when receiving a NACK from the slave.
		1	Master transmitter ignores a NACK from the slave.
0	BCM		Backward compatibility mode bit. Determines which condition generates a transmit data ready interrupt.
			The BCM bit only has an effect when the I2C is operating as a slave-transmitter.
		0	The transmit data ready interrupt is generated when the master requests more data by sending an acknowledge signal after the transmission of the last data.
		1	The transmit data ready interrupt is generated when the data in ICDXR is copied to ICXSR.

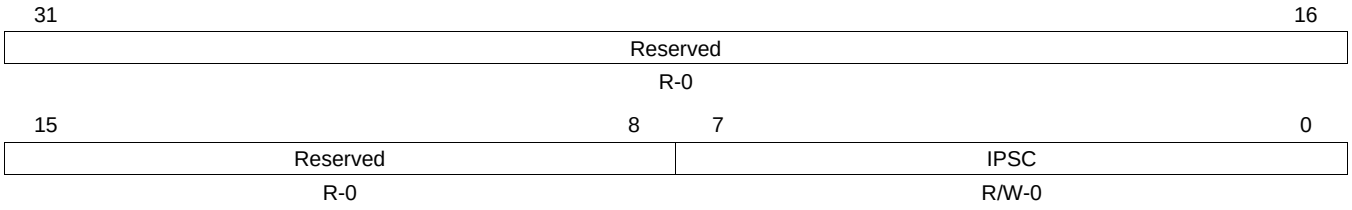
Copyright © 2013–2016, Texas Instruments Incorporated

18.3.12 I2C Prescaler Register (ICPSC)

The I2C prescaler register (ICPSC) is used for dividing down the I2C input clock to obtain the desired prescaled module clock for the operation of the I2C. The IPSC bits must be initialized while the I2C is in reset (IRS = 0 in ICMDR). The prescaled frequency takes effect only when the IRS bit is changed to 1. Changing the IPSC value while IRS = 1 has no effect.

ICPSC is shown in Figure 18-26 and described in Table 18-19.

Figure 18-26. I2C Prescaler Register (ICPSC)



LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 18-19. I2C Prescaler Register (ICPSC) Field Descriptions

Bit	Field	Value	Description
31-8	Reserved	0	These reserved bit locations are always read as zeros. A value written to this field has no effect.
7-0	IPSC	0-FFh	I2C prescaler divide-down value. IPSC determines how much the I2C input clock is divided to create the I2C prescaled module clock: I2C clock frequency = I2C input clock frequency/(IPSC + 1) **Note:** IPSC must be initialized while the I2C is in reset (IRS = 0 in ICMDR).

Copyright © 2013–2016, Texas Instruments Incorporated

18.3.13 I2C Revision Identification Register (REVID1)

The I2C revision identification register (REVID1) contains identification data for the peripheral. REVID1 is shown in Figure 18-27 and described in Table 18-20.

Figure 18-27. I2C Revision Identification Register 1 (REVID1)

31	0
REVID1	
R-4415h	

LEGEND: R = Read only; -*n* = value after reset

Table 18-20. I2C Revision Identification Register 1 (REVID1) Field Descriptions

Bit	Field	Value	Description
31-0	REVID1	4415h	Peripheral Identification Number

18.3.14 I2C Revision Identification Register (REVID2)

The I2C revision identification register (REVID2) contains identification data for the peripheral. REVID2 is shown in Figure 18-28 and described in Table 18-21.

Figure 18-28. I2C Revision Identification Register 2 (REVID2)

31	0
REVID2	
R-6h	

LEGEND: R = Read only; -*n* = value after reset

Table 18-21. I2C Revision Identification Register 2 (REVID2) Field Descriptions

Bit	Field	Value	Description
31-0	REVID2	6h	Peripheral Identification Number

Copyright © 2013–2016, Texas Instruments Incorporated

18.3.15 I2C DMA Control Register (ICDMAC)

The I2C DMA control register (ICDMAC) is used to control the transmit DMA event and receive DMA event pin to the system . ICDMAC is shown in Figure 18-29 and described in Table 18-22.

Figure 18-29. I2C DMA Control Register (ICDMAC)

31	16
Reserved	
R-0	

15	1	0
Reserved	TXDMAEN	RXDMAEN
R-0	R/W-1	R/W-1

LEGEND: R/W = Read/Write; R= Read only; -*n* = value after reset

Table 18-22. I2C DMA Control Register (ICDMAC) Field Descriptions

Bit	Field	Value	Description
31-2	Reserved	0	These reserved bit locations are always read as zeros. A value written to this field has no effect.
1	TXDMAEN		Transmit DMA enable. This bit controls the transmit DMA event pin to the system. Always set this bit to 1.
		0	DMA transmit event is disabled.
		1	DMA transmit event is enabled.
0	RXDMAEN		Receive DMA enable . This bit controls the receive DMA event pin to the system. Always set this bit to 1.
		0	DMA receive event is disabled.
		1	DMA receive event is enabled.

Copyright © 2013–2016, Texas Instruments Incorporated

18.3.16 I2C Pin Function Register (ICPFUNC)

The I2C pin function register (ICPFUNC) is used to configure the external I2C pins (I2Cx_SDA and I2Cx_SCL) as a I2C peripheral pin or a GPIO pin. ICPFUNC is shown in Figure 18-30 and described in Table 18-23.

Figure 18-30. I2C Pin Function Register (ICPFUNC)

31	16
Reserved	
R-0	

15	1	0
Reserved		PFUNC0
R-0		R/W-0

LEGEND: R/W = Read/Write; R= Read only; -*n* = value after reset

Table 18-23. I2C Pin Function Register (ICPFUNC) Field Descriptions

Bit	Field	Value	Description
31-1	Reserved	0	These reserved bit locations are always read as zeros. A value written to this field has no effect.
0	PFUNC0		Controls the function of the I2Cx_SCL and I2Cx_SDA pins.
		0	Pins function as I2Cx_SCL and I2Cx_SDA.
		1	Pins function as GPIO.
			Note: No hardware protection is required to disable the I2C function when the PFUNC0 bit and the IRS bit in the I2C mode register (ICMDR) are both set to 1. When PFUNC0 = 1 (GPIO mode), the sub-module that controls the I2C function receives the value 1 for I2Cx_SCL and I2Cx_SDA. The IRS bit can be set to 1 regardless of PFUNC0, and the I2C function works whenever the IRS bit is 1. You are expected to hold I2C in reset via the IRS bit when changing to/from GPIO mode via the PFUNC0 bit.

Copyright © 2013–2016, Texas Instruments Incorporated

18.3.17 I2C Pin Direction Register (ICPDIR)

The I2C pin direction register (ICPDIR) is used to configure each GPIO pin as either an input or an output. ICPDIR is shown in Figure 18-31 and described in Table 18-24.

Figure 18-31. I2C Pin Direction Register (ICPDIR)

31	16
Reserved	
R-0	

15	2	1	0
Reserved		PDIR1	PDIR0
R-0		R/W-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 18-24. I2C Pin Direction Register (ICPDIR) Field Descriptions

Bit	Field	Value	Description
31-2	Reserved	0	These reserved bit locations are always read as zeros. A value written to this field has no effect.
1	PDIR1		Controls the direction of the I2Cx_SDA pin when configured as GPIO.
		0	I2Cx_SDA pin functions as input.
		1	I2Cx_SDA pin functions as output.
0	PDIR0		Controls the direction of the I2Cx_SCL pin when configured as GPIO.
		0	I2Cx_SCL pin functions as input.
		1	I2Cx_SCL pin functions as output.

Copyright © 2013–2016, Texas Instruments Incorporated

18.3.18 I2C Pin Data In Register (ICPDIN)

The I2C pin data in register (ICPDIN) holds the I/O state of each of the I2C pins (I2Cx_SDA and I2Cx_SCL); and should return the value from the pin's input buffer (with appropriate synchronization/DFT considerations). However, this register allows the actual value of the pin to be read regardless of the state of PFUNC or PDIR bits . ICPDIN is shown in Figure 18-32 and described in Table 18-25.

Figure 18-32. I2C Pin Data In Register (ICPDIN)

31	16
Reserved	
R-0	

15	2	1	0
Reserved		PDIN1	PDIN0
R-0		R/W-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 18-25. I2C Pin Data In Register (ICPDIN) Field Descriptions

Bit	Field	Value	Description
31-2	Reserved	0	These reserved bit locations are always read as zeros. A value written to this field has no effect.
1	PDIN1		Indicates the logic level present on the I2Cx_SDA pin.
			During reads:
		0	Logic-low present at I2Cx_SDA pin, regardless of PFUNC bit setting.
		1	Logic-high present at I2Cx_SDA pin, regardless of PFUNC bit setting.
			During writes: Writes have no effect.
0	PDIN0		Indicates the logic level present on the I2Cx_SCL pin.
			During reads:
		0	Logic-low present at I2Cx_SCL pin, regardless of PFUNC bit setting.
		1	Logic-high present at I2Cx_SCL pin, regardless of PFUNC bit setting.
			During writes: Writes have no effect.

Copyright © 2013–2016, Texas Instruments Incorporated

18.3.19 I2C Pin Data Out Register (ICPDOUT)

The I2C pin data out register (ICPDOUT) has one bit for each of the GPIO pins. This bit holds a value for data out at all times, and may be read back at all times. The value held by this register is not affected by writing to the PDIR and PFUNC bits. However, the data value in this register is driven out onto the GPIO pin only if the PFUNC0 bit in ICPFUNC is set to 1 (I2Cx_SDA and I2Cx_SCL function as GPIO) and also the corresponding bit in ICPDIR is set to 1 (output).

ICPDOUT is shown in Figure 18-33 and described in Table 18-26.

Figure 18-33. I2C Pin Data Out Register (ICPDOUT)

31	16
Reserved	
R-0	

15	2	1	0
Reserved		PDOUT1	PDOUT0
R-0		R/W-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 18-26. I2C Pin Data Out Register (ICPDOUT) Field Descriptions

Bit	Field	Value	Description
31-2	Reserved	0	These reserved bit locations are always read as zeros. A value written to this field has no effect.
1	PDOUT1		Controls the level driven on the I2Cx_SDA pin when configured as GPIO output. Note: If I2Cx_SDA is connected to an open-drain buffer at the chip level, the I2C cannot drive I2Cx_SDA to high. **During reads:** Reads return register values, not GPIO pin levels.
			During writes:
		0	I2Cx_SDA pin is driven low.
		1	I2Cx_SDA pin is driven high.
0	PDOUT0		Controls the level driven on the I2Cx_SCL pin when configured as GPIO output. Note: If I2Cx_SCL is connected to an open-drain buffer at the chip level, the I2C cannot drive I2Cx_SCL to high. **During reads:** Reads return register values, not GPIO pin levels.
			During writes:
		0	I2Cx_SCL pin is driven low.
		1	I2Cx_SCL pin is driven high.

Copyright © 2013–2016, Texas Instruments Incorporated

18.3.20 I2C Pin Data Set Register (ICPDSET)

The I2C pin data set register (ICPDSET) is an alias of the I2C pin data out register (ICPDOUT). Writing a 1 to a bit in ICPDSET sets the corresponding bit in ICPDOUT to a 1, while writing a 0 keeps the bit unchanged. ICPDSET is shown in Figure 18-34 and described in Table 18-27.

Figure 18-34. I2C Pin Data Set Register (ICPDSET)

31	16
Reserved	
R-0	

15	2	1	0
Reserved		PDSET1	PDSET0
R-0		R/W-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; *-n* = value after reset

Table 18-27. I2C Pin Data Set Register (ICPDSET) Field Descriptions

Bit	Field	Value	Description
31-2	Reserved	0	These reserved bit locations are always read as zeros. A value written to this field has no effect.
1	PDSET1		Used to set the PDOUT1 bit in the I2C pin data out register (ICPDOUT) that corresponds to the I2Cx_SDA GPIO pin. **During reads:** Reads return indeterminate values.
			During writes:
		0	No effect
		1	PDOUT1 bit is set to logic high.
0	PDSET0		Used to set the PDOUT0 bit in the I2C pin data out register (ICPDOUT) that corresponds to the I2Cx_SCL GPIO pin. **During reads:** Reads return indeterminate values.
			During writes:
		0	No effect
		1	PDOUT0 bit is set to logic high.

Copyright © 2013–2016, Texas Instruments Incorporated

18.3.21 I2C Pin Data Clear Register (ICPDCLR)

The I2C pin data clear register (ICPDCLR) is an alias of the I2C pin data out register (ICPDOUT). Writing a 1 to a bit in ICPDCLR clears the corresponding bit in ICPDOUT to a 0, while writing a 0 keeps the bit unchanged. ICPDCLR is shown in Figure 18-35 and described in Table 18-28.

Figure 18-35. I2C Pin Data Clear Register (ICPDCLR)

31	16
Reserved	
R-0	

15	2	1	0
Reserved		PDCLR1	PDCLR0
R-0		R/W-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; *-n* = value after reset

Table 18-28. I2C Pin Data Clear Register (ICPDCLR) Field Descriptions

Bit	Field	Value	Description
31-2	Reserved	0	These reserved bit locations are always read as zeros. A value written to this field has no effect.
1	PDCLR1		Used to clear the PDOUT1 bit in the I2C pin data out register (ICPDOUT) that corresponds to the I2Cx_SDA GPIO pin. **During reads:** Reads return indeterminate values.
			During writes:
		0	No effect
		1	PDOUT1 bit is cleared to logic low.
0	PDCLR0		Used to clear the PDOUT0 bit in the I2C pin data out register (ICPDOUT) that corresponds to the I2Cx_SCL GPIO pin. **During reads:** Reads return indeterminate values.
			During writes:
		0	No effect
		1	PDOUT0 bit is cleared to logic low.

Copyright © 2013–2016, Texas Instruments Incorporated

Multichannel Audio Serial Port (McASP)

This chapter describes the multichannel audio serial port (McASP). See your device-specific data manual to determine how many McASPs are available on your device.

Topic **Page**

19.1 Registers **639**

Copyright © 2013–2016, Texas Instruments Incorporated

Introduction

Purpose of the Peripheral

The multichannel audio serial port (McASP) functions as a general-purpose audio serial port optimized for the needs of multichannel audio applications. The McASP is useful for time-division multiplexed (TDM) stream, Inter-IC Sound (I2S) protocols, and intercomponent digital audio interface transmission (DIT).

The McASP consists of transmit and receive sections that may operate synchronized, or completely independently with separate master clocks, bit clocks, and frame syncs, and using different transmit modes with different bit-stream formats. The McASP module also includes up to 16 serializers that can be individually enabled to either transmit or receive. In addition, all of the McASP pins can be configured as general-purpose input/output (GPIO) pins.

19.0.22 Features

Features of the McASP include:

- Two independent clock generator modules for transmit and receive
  - Clocking flexibility allows the McASP to receive and transmit at different rates. For example, the McASP can receive data at 48 kHz but output up-sampled data at 96 kHz or 192 kHz.
- Independent transmit and receive modules, each includes:
  - Programmable clock and frame sync generator
  - TDM streams from 2 to 32, and 384 time slots
  - Support for time slot sizes of 8, 12, 16, 20, 24, 28, and 32 bits
  - Data formatter for bit manipulation
- Up to 16 individually assignable serial data pins:
  - McASP0 can have up to 16 serial data pins
  - McASP1 can have up to 12 serial data pins
  - McASP2 can have up to 4 serial data pins
- Glueless connection to audio analog-to-digital converters (ADC), digital-to-analog converters (DAC), codec, digital audio interface receiver (DIR), and S/PDIF transmit physical layer components
- Wide variety of Inter-IC Sound (I2S) and similar bit-stream formats
- Integrated digital audio interface transmitter (DIT) supports (McASP2 only):
  - S/PDIF, IEC60958-1, AES-3 formats
  - Up to 4 transmit pins
  - Enhanced channel status/user data RAM
- 384-slot TDM with external digital audio interface receiver (DIR) device
  - For DIR reception, an external DIR receiver integrated circuit should be used with I2S output format and connected to the McASP receive section.
- Extensive error checking and recovery:
  - Transmit underruns and receiver overruns due to the system not meeting real-time requirements
  - Early or late frame sync in TDM mode
  - Out-of-range high-frequency master clock for both transmit and receive
  - External error signal coming into the AMUTEIN input
  - DMA error due to incorrect programming
- McASP Audio FIFO (AFIFO):
  - Provides additional data buffering
  - Provides added tolerance to variations in host/DMA controller response times
  - May be used as a DMA event pacer
  - Independent Read FIFO and Write FIFO
  - 256 bytes of RAM for each FIFO (read and write)

Copyright © 2013–2016, Texas Instruments Incorporated

    - 256 bytes = four 32-bit words per serializer in the case of 16 data pins
    - 256 bytes = 64 32-bit words in the case of one data pin
  - Option to bypass Write FIFO and/or Read FIFO independently

19.0.23 Protocols Supported

The McASP supports a wide variety of protocols.

- Transmit section supports
  - Wide variety of I2S and similar bit-stream formats
  - TDM streams from 2 to 32 time slots
  - S/PDIF, IEC60958-1, AES-3 formats
- Receive section supports
  - Wide variety of I2S and similar bit-stream formats
  - TDM streams from 2 to 32 time slots
  - TDM stream of 384 time slots specifically designed for easy interface to external digital interface receiver (DIR) device transmitting DIR frames to McASP using the I2S protocol (one time slot for each DIR subframe)

The transmit and receive sections may each be individually programmed to support the following options on the basic serial protocol:

- Programmable clock and frame sync polarity (rising or falling edge): ACLKR/X, AHCLKR/X, and AFSR/X
- Slot length (number of bits per time slot): 8, 12, 16, 20, 24, 28, 32 bits supported
- Word length (bits per word): 8, 12, 16, 20, 24, 28, 32 bits; always less than or equal to the time slot length
- First-bit data delay: 0, 1, 2 bit clocks
- Left/right alignment of word inside slot
- Bit order: MSB first or LSB first
- Bit mask/pad/rotate function
  - Automatically aligns data for DSP internally in either Q31 or integer formats
  - Automatically masks nonsignificant bits (sets to 0, 1, or extends value of another bit)

In DIT mode (McASP2 only), additional features of the transmitter are:

- Transmit-only mode 384 time slots (subframe) per frame
- Bi-phase encoded 3.3 V output
- Support for consumer and professional applications
- Channel status RAM (384 bits)
- User data RAM (384 bits)
- Separate valid bit (V) for subframe A, B

In I2S mode, the transmit and receive sections can support simultaneous transfers on up to all serial data pins operating as 192 kHz stereo channels.

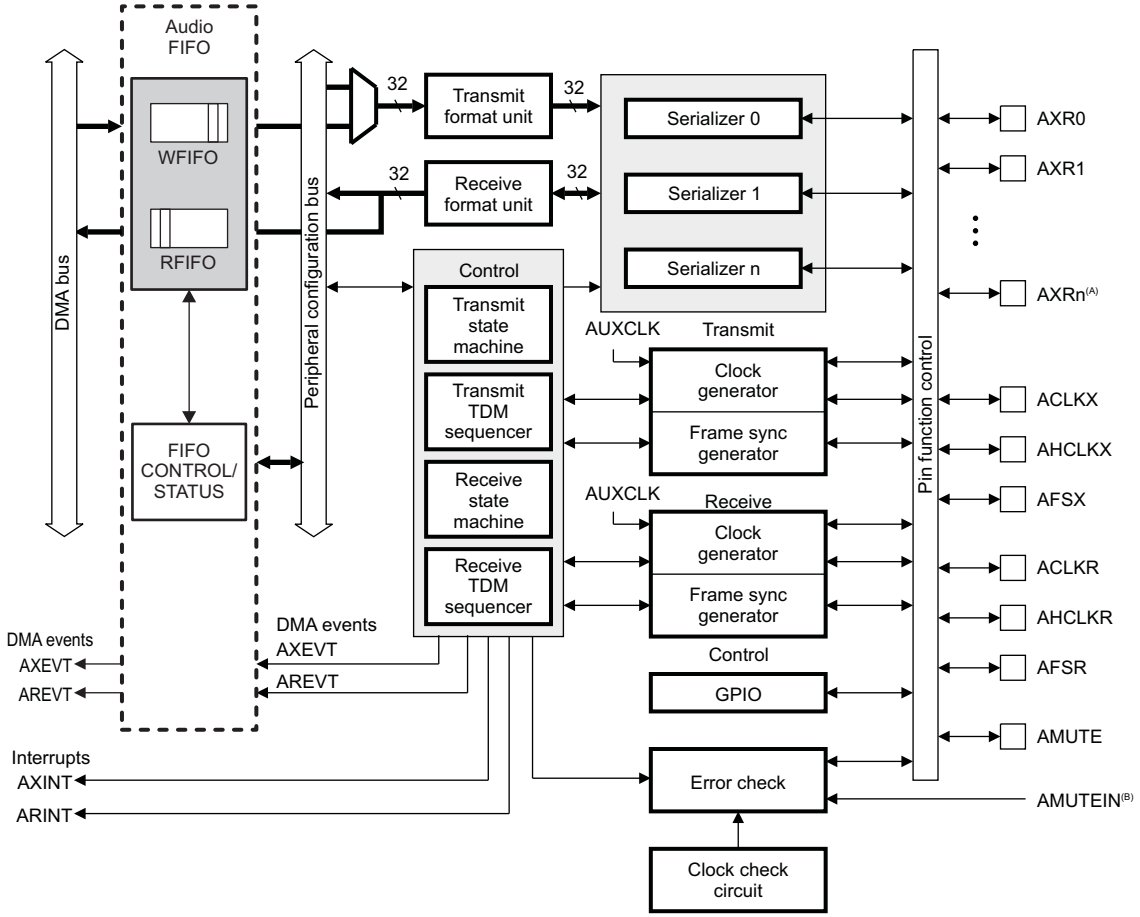
In DIT mode, the transmitter can support a 192 kHz frame rate (stereo) on up to 2 serial data pins simultaneously (note that the internal bit clock for DIT runs two times faster than the equivalent bit clock for I2S mode, due to the need to generate Biphase Mark Encoded Data).

Copyright © 2013–2016, Texas Instruments Incorporated

19.0.24 Functional Block Diagram

A block diagram of the McASP is shown in Figure 19-1. The McASP has independent receive/transmit clock generators and frame sync generators.

Figure 19-1. McASP Block Diagram



A McASP0 has up to 16 serial data pins, *n* = 15; McASP1 has up to 12 serial data pins, *n* = 11; McASP2 has up to 4 serial data pins, *n* = 3.

B One of the DSP's external pins, see your device-specific data manual.

Copyright © 2013–2016, Texas Instruments Incorporated

19.0.24.1 System Level Connections

Figure 19-2 through Figure 19-5 show examples of McASP usage in digital audio encoder/decoder systems.

Figure 19-2. McASP to Parallel 2-Channel DACs

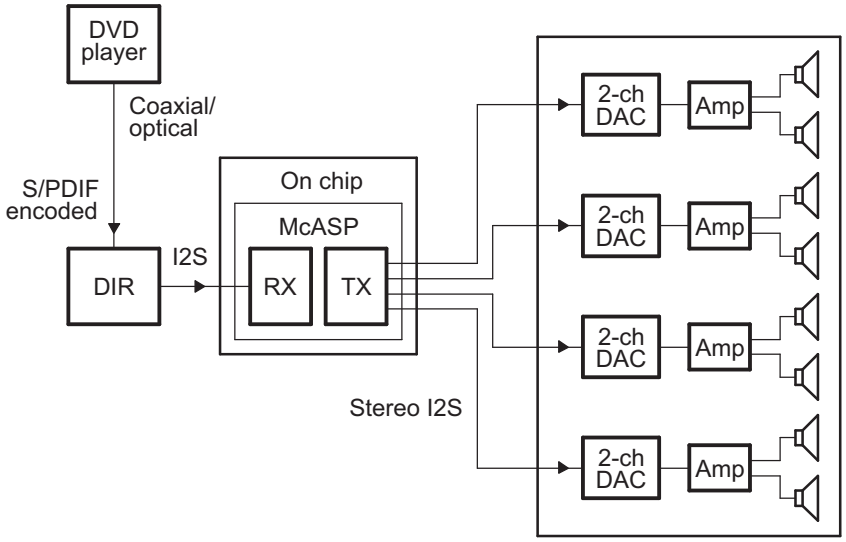
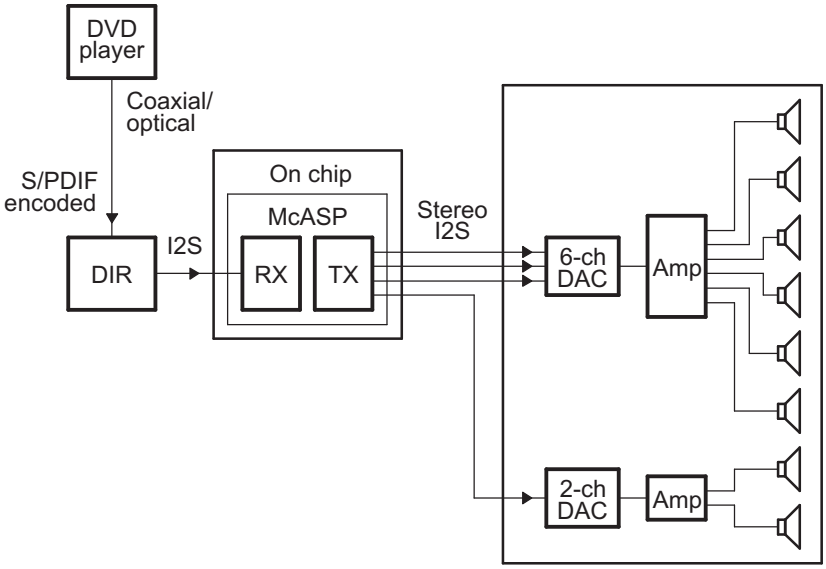


Figure 19-3. McASP to 6-Channel DAC and 2-Channel DAC



Copyright © 2013–2016, Texas Instruments Incorporated

Figure 19-4. McASP to Digital Amplifier

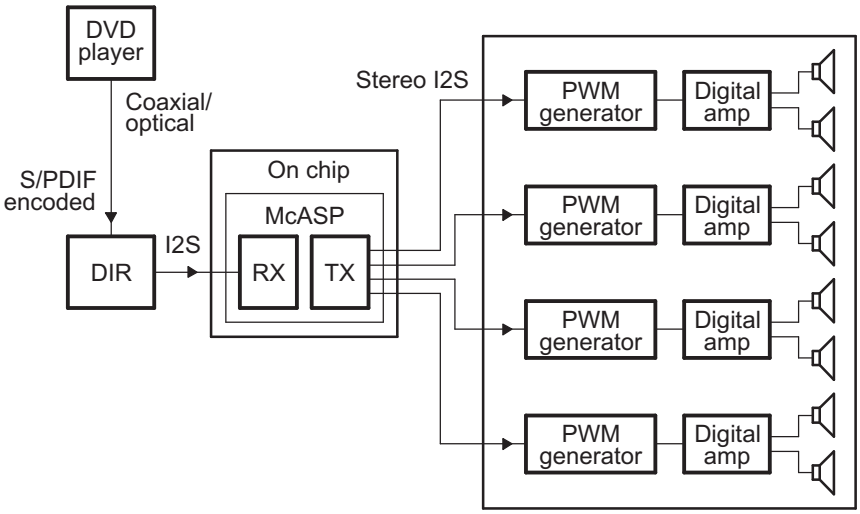
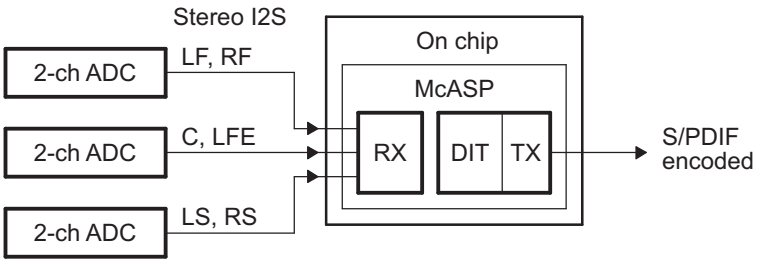


Figure 19-5. McASP as Digital Audio Encoder



Copyright © 2013–2016, Texas Instruments Incorporated

Industry Standard Compliance Statement

The McASP supports the following industry standard interfaces.

19.0.24.1 TDM Format

The McASP transmitter and receiver support the multichannel, synchronous time-division-multiplexed (TDM) format via the TDM transfer mode. Within this transfer mode, a wide variety of serial data formats are supported, including formats compatible with devices using the Inter-IC Sound (I2S) protocol. This section briefly discusses the TDM format and the I2S protocol.

19.0.24.1.1 TDM Format

The TDM format is typically used when communicating between integrated circuit devices on the same printed circuit board or on another printed circuit board within the same piece of equipment. For example, the TDM format is used to transfer data between the DSP and one or more analog-to-digital converter (ADC), digital-to-analog converter (DAC), or S/PDIF receiver (DIR) devices.

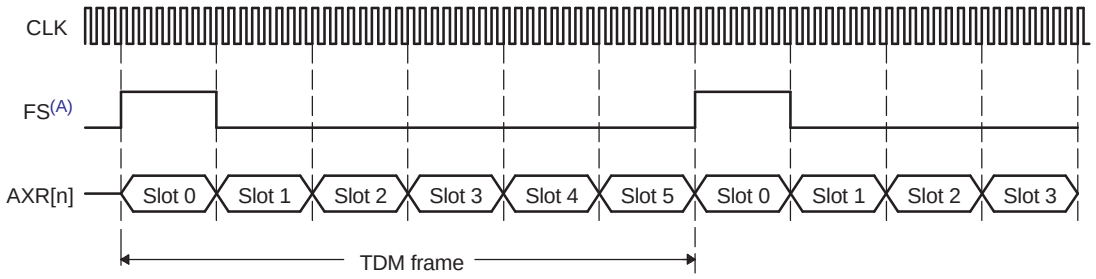
The TDM format consists of three components in a basic synchronous serial transfer: the clock, the data, and the frame sync. In a TDM transfer, all data bits (AXR[n]) are synchronous to the serial clock (ACLKX or ACLKR). The data bits are grouped into words and slots (as defined in Section 19.0.25). The "slots" are also commonly referred to as "time slots" or "channels" in TDM terminology. A frame consists of multiple slots (or channels). Each TDM frame is defined by the frame sync signal (AFSX or AFSR). Data transfer is continuous and periodic, since the TDM format is most commonly used to communicate with data converters that operate at a fixed sample rate.

There are no delays between slots. The last bit of slot N is followed immediately on the next serial clock cycle with the first bit of slot N + 1, and the last bit of the last slot is followed immediately on the next serial clock with the first bit of the first slot. However, the frame sync may be offset from the first bit of the first slot with a 0, 1, or 2-cycle delay.

It is required that the transmitter and receiver in the system agree on the number of bits per slot, since the determination of a slot boundary is not made by the frame sync signal (although the frame sync marks the beginning of slot 0 and the beginning of a new frame).

Figure 19-6 shows the TDM format. Figure 19-7 shows the different bit delays from the frame sync.

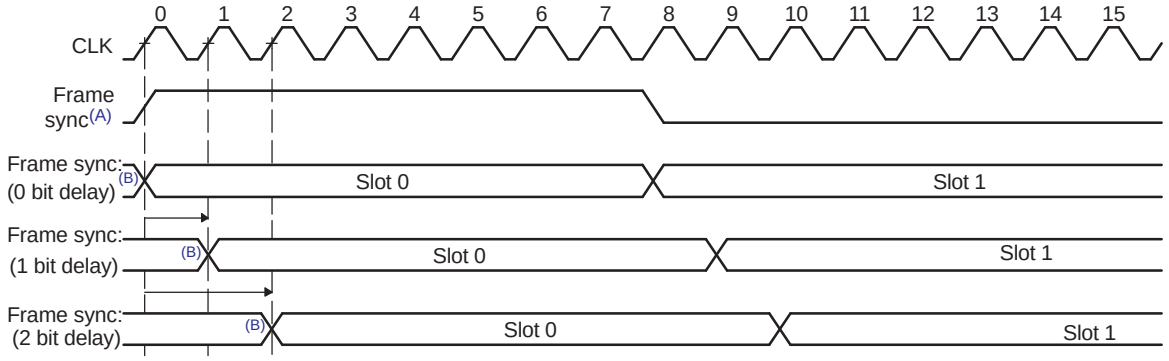
Figure 19-6. TDM Format–6 Channel TDM Example



A FS duration of slot is shown. FS duration of single bit is also supported.

Copyright © 2013–2016, Texas Instruments Incorporated

Figure 19-7. TDM Format Bit Delays from Frame Sync



A FS duration of slot is shown. FS duration of single bit is also supported.

B Last bit of last slot of previous frame. No gap between this bit and the first bit of slot 0 is allowed.

In a typical audio system, one frame of data is transferred during each data converter sample period f_s. To support multiple channels, the choices are to either include more time slots per frame (thus operating with a higher bit clock rate), or to use additional data pins to transfer the same number of channels (thus operating with a slower bit clock rate).

For example, a particular six channel DAC may be designed to transfer over a single serial data pin AXR[n] as shown in Figure 19-6. In this case the serial clock must run fast enough to transfer a total of 6 channels within each frame period. Alternatively, a similar six channel DAC may be designed to use three serial data pins AXR[0,1,2], transferring two channels of data on each pin during each sample period (Figure 19-8). In the latter case, if the sample period remains the same, the serial clock can run three times slower than the former case. The McASP is flexible enough to support either type of DAC.

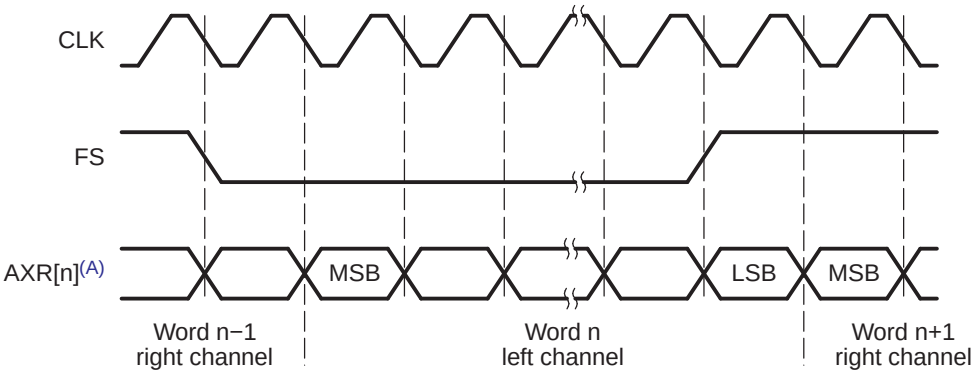
19.0.24.1.2 Inter-IC Sound (I2S) Format

The Inter-IC Sound (I2S) format is used extensively in audio interfaces. The TDM transfer mode of the McASP supports the I2S format when configured to 2 slots per frame.

I2S format is specifically designed to transfer a stereo channel (left and right) over a single data pin AXR[n]. "Slots" are also commonly referred to as "channels". The frame width duration in the I2S format is the same as the slot size. The frame signal is also referred to as "word select" in the I2S format. Figure 19-8 shows the I2S protocol.

The McASP supports transfer of multiple stereo channels over multiple AXR[n] pins.

Figure 19-8. Inter-IC Sound (I2S) Format



A 1 to 16 data pins may be supported.

Copyright © 2013–2016, Texas Instruments Incorporated

19.0.24.2 S/PDIF Coding Format

The McASP transmitter supports the S/PDIF format with 3.3V biphase-mark encoded output. The S/PDIF format is supported by the digital audio interface transmit (DIT) transfer mode of the McASP. This section briefly discusses the S/PDIF coding format.

19.0.24.2.1 Biphase-Mark Code (BMC)

In S/PDIF format, the digital signal is coded using the biphase-mark code (BMC). The clock, frame, and data are embedded in only one signal—the data pin AXR[n]. In the BMC system, each data bit is encoded into two logical states (00, 01, 10, or 11) at the pin. These two logical states form a cell. The duration of the cell, which equals to the duration of the data bit, is called a time interval. A logical 1 is represented by two transitions of the signal within a time interval, which corresponds to a cell with logical states 01 or 10. A logical 0 is represented by one transition within a time interval, which corresponds to a cell with logical states 00 or 11. In addition, the logical level at the start of a cell is inverted from the level at the end of the previous cell. Figure 19-9 and Table 19-1 show how data is encoded to the BMC format.

As shown in Figure 19-9, the frequency of the clock is twice the unencoded data bit rate. In addition, the clock is always programmed to 128 × f_s, where f_s is the sample rate (see Section 19.0.24.2.3 for details on how this clock rate is derived based on the S/PDIF format). The device receiving in S/PDIF format can recover the clock and frame information from the BMC signal.

Figure 19-9. Biphase-Mark Code (BMC)

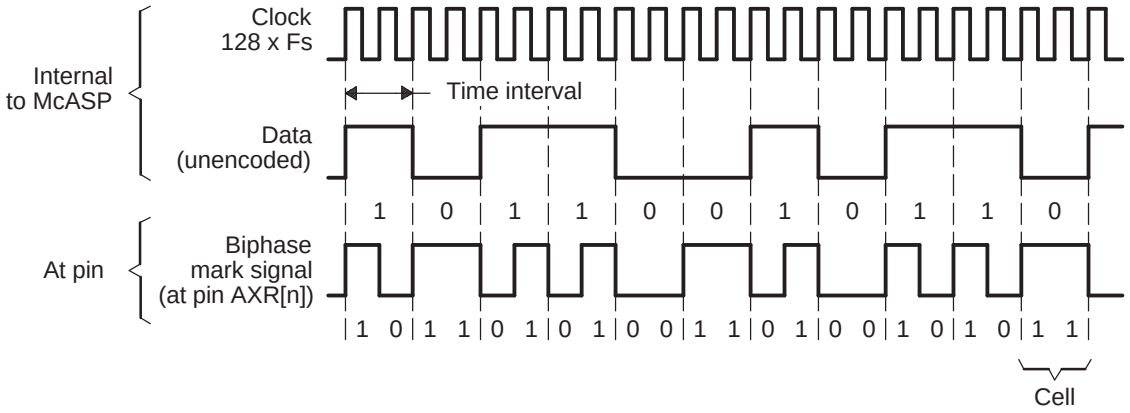


Table 19-1. Biphase-Mark Encoder

Data (Unencoded)	Previous State at Pin AXR[n]	BMC-Encoded Cell Output at AXR[n]
0	0	11
0	1	00
1	0	10
1	1	01

Copyright © 2013–2016, Texas Instruments Incorporated

19.0.24.2.2 Subframe Format

Every audio sample transmitted in a subframe consists of 32 S/PDIF time intervals (or cells), numbered from 0 to 31. Figure 19-10 shows a subframe.

- **Time intervals 0-3** carry one of the three permitted preambles to signify the type of audio sample in the current subframe. The preamble is *not* encoded in BMC format, and therefore the preamble code can contain more than two consecutive 0 or 1 logical states in a row. See Table 19-2.
- **Time intervals 4-27** carry the audio sample word in linear 2s-complement representation. The most-significant bit (MSB) is carried by time interval 27. When a 24-bit coding range is used, the least-significant bit (LSB) is in time interval 4. When a 20-bit coding range is used, time intervals 8-27 carry the audio sample word with the LSB in time interval 8. Time intervals 4-7 may be used for other applications and are designated auxiliary sample bits.
- If the source provides fewer bits than the interface allows (either 20 or 24), the unused LSBs are set to logical 0. For a nonlinear PCM audio application or a data application, the main data field may carry any other information.
- **Time interval 28** carries the validity bit (V) associated with the main data field in the subframe.
- **Time interval 29** carries the user data channel (U) associated with the main data field in the subframe.
- **Time interval 30** carries the channel status information (C) associated with the main data field in the subframe. The channel status indicates if the data in the subframe is digital audio or some other type of data.
- **Time interval 31** carries a parity bit (P) such that time intervals 4-31 carry an even number of 1s and an even number of 0s (even parity). As shown in Table 19-2, the preambles (time intervals 0-3) are also defined with even parity.

Figure 19-10. S/PDIF Subframe Format

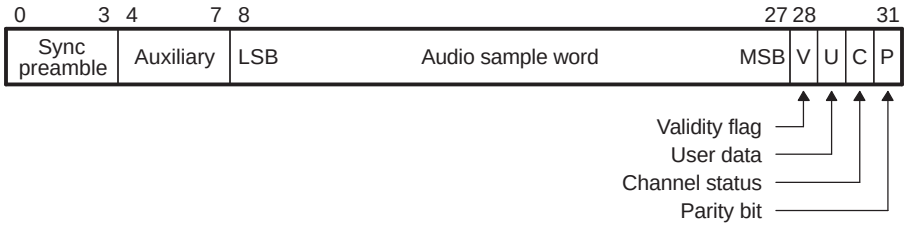


Table 19-2. Preamble Codes

Preamble Code[1]	Previous Logical State	Logical States on pin AXR[n][2]	Description
B (or Z)	0	1110 1000	Start of a block and subframe 1
M (or X)	0	1110 0010	Subframe 1
W (or Y)	0	1110 0100	Subframe 2

(1) Historically, preamble codes are referred to as B, M, W. For use in professional applications, preambles are referred to as Z, X, Y, respectively.

(2) The preamble is not BMC encoded. Each logical state is synchronized to the serial clock. These 8 logical states make up time slots (cells) 0 to 3 in the S/PDIF stream.

As shown in Table 19-2, the McASP DIT only generates one polarity of preambles and it assumes the previous logical state to be 0. This is because the McASP assures an even-polarity encoding scheme when transmitting in DIT mode. If an underrun condition occurs, the DIT resynchronizes to the correct logic level on the AXR[n] pin before continuing with the next transmission.

Copyright © 2013–2016, Texas Instruments Incorporated

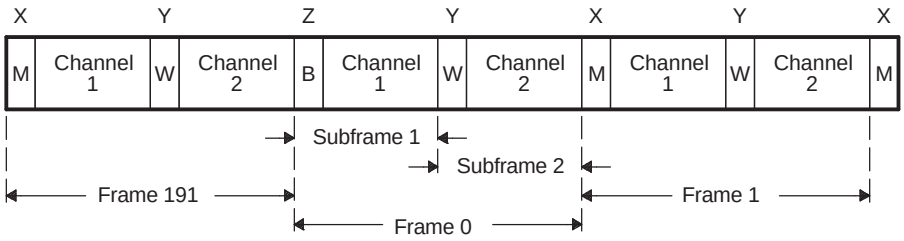
19.0.24.2.3 Frame Format

An S/PDIF frame is composed of two subframes (Figure 19-11). For linear coded audio applications, the rate of frame transmission normally corresponds exactly to the source sampling frequency f_s. The S/PDIF format clock rate is therefore 128 × f_s (128 = 32 cells/subframe × 2 clocks/cell × 2 subframes/sample). For example, for an S/PDIF stream at a 192 kHz sampling frequency, the serial clock is 128 × 192 kHz = 24.58 MHz.

In 2-channel operation mode, the samples taken from both channels are transmitted by time multiplexing in consecutive subframes. Both subframes contain valid data. The first subframe (**left** or **A** channel in stereophonic operation and **primary** channel in monophonic operation) normally starts with preamble M. However, the preamble of the first subframe changes to preamble B once every 192 frames to identify the start of the block structure used to organize the channel status information. The second subframe (**right** or **B** channel in stereophonic operation and **secondary** channel in monophonic operation) always starts with preamble W.

In single-channel operation mode in a professional application, the frame format is the same as in the 2-channel mode. Data is carried in the first subframe and may be duplicated in the second subframe. If the second subframe is not carrying duplicate data, time slot 28 (validity bit) is set to logical 1.

Figure 19-11. S/PDIF Frame Format



Copyright © 2013–2016, Texas Instruments Incorporated

19.0.25 Definition of Terms

The serial bit stream transmitted or received by the McASP is a long sequence of 1s and 0s, either output or input on one of the audio transmit/receive pins (AXR[n]). However, the sequence has a hierarchical organization that can be described in terms of frames of data, slots, words, and bits.

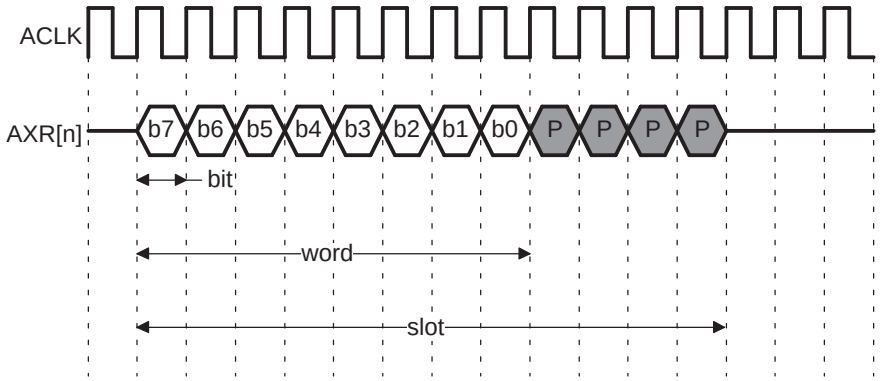
A basic synchronous serial interface consists of three important components: clock, frame sync, and data. Figure 19-12 shows two of the three basic components—the clock (ACLK) and the data (AXR[n]). Figure 19-12 does not specify whether the clock is for transmit (ACLKX) or receive (ACLKR) because the definitions of terms apply to both receive and transmit interfaces. In operation, the transmitter uses ACLKX as the serial clock, and the receiver uses ACLKR as the serial clock. Optionally, the receiver can use ACLKX as the serial clock when the transmitter and receiver of the McASP are configured to operate synchronously.

Bit A bit is the smallest entity in the serial data stream. The beginning and end of each bit is marked by an edge of the serial clock. The duration of a bit is a serial clock period. A 1 is represented by a logic high on the AXR[n] pin for the entire duration of the bit. A 0 is represented by a logic low on the AXR[n] pin for the entire duration of the bit.

Word A word is a group of bits that make up the data being transferred between the DSP and the external device. Figure 19-12 shows an 8-bit word.

Slot A slot consists of the bits that make up the word, and may consist of additional bits used to pad the word to a convenient number of bits for the interface between the DSP and the external device. In Figure 19-12, the audio data consists of only 8 bits of useful data (8-bit word), but it is padded with 4 zeros (12-bit slot) to satisfy the desired protocol in interfacing to an external device. Within a slot, the bits may be shifted in/out of the McASP on the AXR[n] pin either MSB or LSB first. When the word size is smaller than the slot size, the word may be aligned to the left (beginning) of the slot or to the right (end) of the slot. The additional bits in the slot not belonging to the word may be padded with 0, 1, or with one of the bits (the MSB or the LSB typically) from the data word. These options are shown in Figure 19-13.

Figure 19-12. Definition of Bit, Word, and Slot



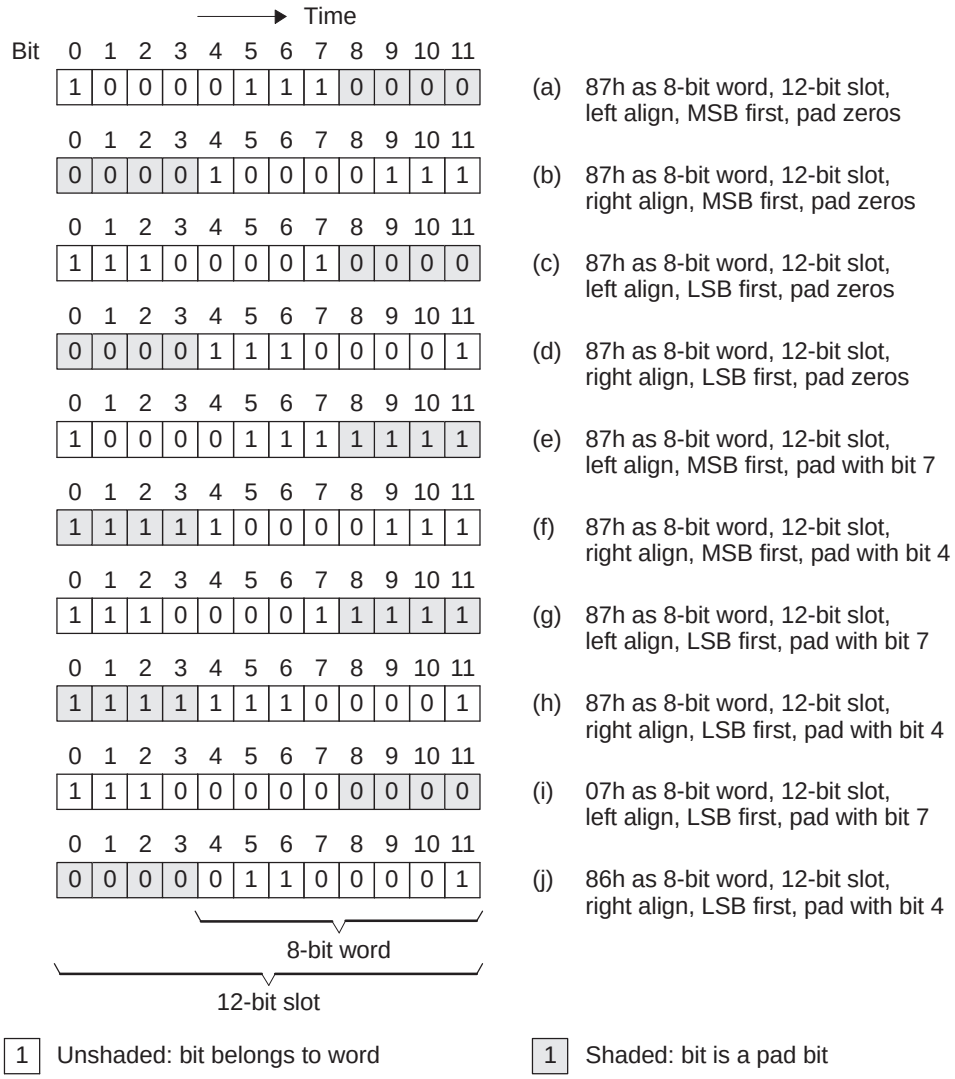
(1) b7:b0 - bits. Bits b7 to b0 form a word.

(2) P - pad bits. Bits b7 to b0, together with the four pad bits, form a slot.

(3) In this example, the data is transmitted MSB first, left aligned.

Copyright © 2013–2016, Texas Instruments Incorporated

Figure 19-13. Bit Order and Word Alignment Within a Slot Examples



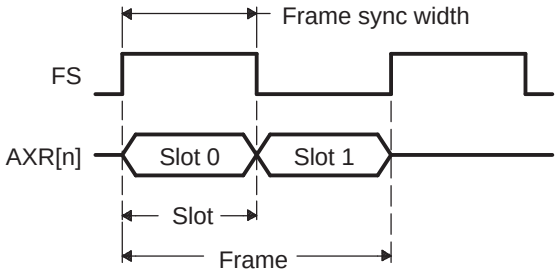
Copyright © 2013–2016, Texas Instruments Incorporated

The third basic element of a synchronous serial interface is the frame synchronization signal, also referred to as frame sync in this chapter.

Frame A frame contains one or multiple slots, as determined by the desired protocol. Figure 19-14 shows an example frame of data and the frame definitions. Figure 19-14 does not specify whether the frame sync (FS) is for transmit (AFSX) or receive (AFSR) because the definitions of terms apply to both receive and transmit interfaces. In operation, the transmitter uses AFSX and the receiver uses AFSR. Optionally, the receiver can use AFSX as the frame sync when the transmitter and receiver of the McASP are configured to operate synchronously.

This section only shows the generic definition of the frame sync. See Section 19.0.24.1, Section 19.0.24.2, and Section 19.0.27.2.1 for details on the frame sync formats required for the different transfer modes and protocols (burst mode, TDM mode and I2S format, DIT mode and S/PDIF format).

Figure 19-14. Definition of Frame and Frame Sync Width



(1) In this example, there are two slots in a frame, and FS duration of slot length is shown.

Other terms used throughout this chapter:

TDM	Time-division multiplexed. See Section 19.0.24.1 for details on the TDM protocol.
DIR	Digital audio interface receive. The McASP does not natively support receiving in the S/PDIF format. The McASP supports I2S format output by an external DIR device.
DIT	Digital audio interface transmit. The McASP supports transmitting in S/PDIF format on up to all data pins configured as outputs.
I2S	Inter-IC Sound protocol, commonly used on audio interfaces. The McASP supports the I2S protocol as part of the TDM mode (when configured as a 2-slot frame).
Slot or Time Slot	For TDM format, the term time slot is interchangeable with the term slot defined in this section. For DIT format, a McASP time slot corresponds to a DIT subframe.

Copyright © 2013–2016, Texas Instruments Incorporated

Architecture

19.0.26 Overview

Figure 19-1 shows the major blocks of the McASP. The McASP has independent receive/transmit clock generators and frame sync generators, error-checking logic, and up to 16 serial data pins. See your device-specific data manual for the number of data pins available on your device.

All the McASP pins on the device may be individually programmed as general-purpose I/O (GPIO) if they are not used for serial port functions.

The McASP includes the following pins:

- Serializers
  - Data pins AXR[n]: Up to sixteen per McASP
- Transmit clock generator:
  - AHCLKX: McASP transmit high-frequency master clock
  - ACLKX: McASP transmit bit clock
- Transmit Frame Sync Generator
  - AFSX: McASP transmit frame sync or left/right clock (LRCLK)
- Receive clock generator:
  - AHCLKR: McASP receive high-frequency master clock
  - ACLKR: McASP receive bit clock
- Receive Frame Sync Generator
  - AFSR: McASP receive frame sync or left/right clock (LRCLK)
- Mute in/out:
  - AMUTEIN: McASP mute input (from external device)
  - AMUTE: McASP mute output
  - Data pins AXR[n]

19.0.27 Clock and Frame Sync Generators

The McASP clock generators are able to produce two independent clock zones: transmit and receive clock zones. The serial clock generators may be programmed independently for the transmit section and the receive section, and may be completely asynchronous to each other. The serial clock (clock at the bit rate) may be sourced:

- **Internally** - by passing through two clock dividers off the internal clock source (AUXCLK)
- **Externally** - directly from ACLKR/X pin
- **Mixed** - an external high-frequency clock is input to the McASP on either the AHCLKX or AHCLKR pins, and divided down to produce the bit rate clock

In the internal/mixed cases, the bit rate clock is generated internally and should be driven out on the ACLKX (for transmit) or ACLKR (for receive) pins. In the internal case, an internally-generated high-frequency clock may be driven out onto the AHCLKX or AHCLKR pins to serve as a reference clock for other components in the system.

The McASP requires a minimum of a bit clock and a frame sync to operate, and provides the capability to reference these clocks from an external high-frequency master clock. In DIT mode, it is possible to use only internally-generated clocks and frame syncs.

Copyright © 2013–2016, Texas Instruments Incorporated

19.0.27.1 Transmit Clock

The transmit bit clock, ACLKX, (Figure 19-15) may be either externally sourced from the ACLKX pin or internally generated, as selected by the CLKXM bit. If internally generated (CLKXM = 1), the clock is divided down by a programmable bit clock divider (CLKXDIV) from the transmit high-frequency master clock (AHCLKX).

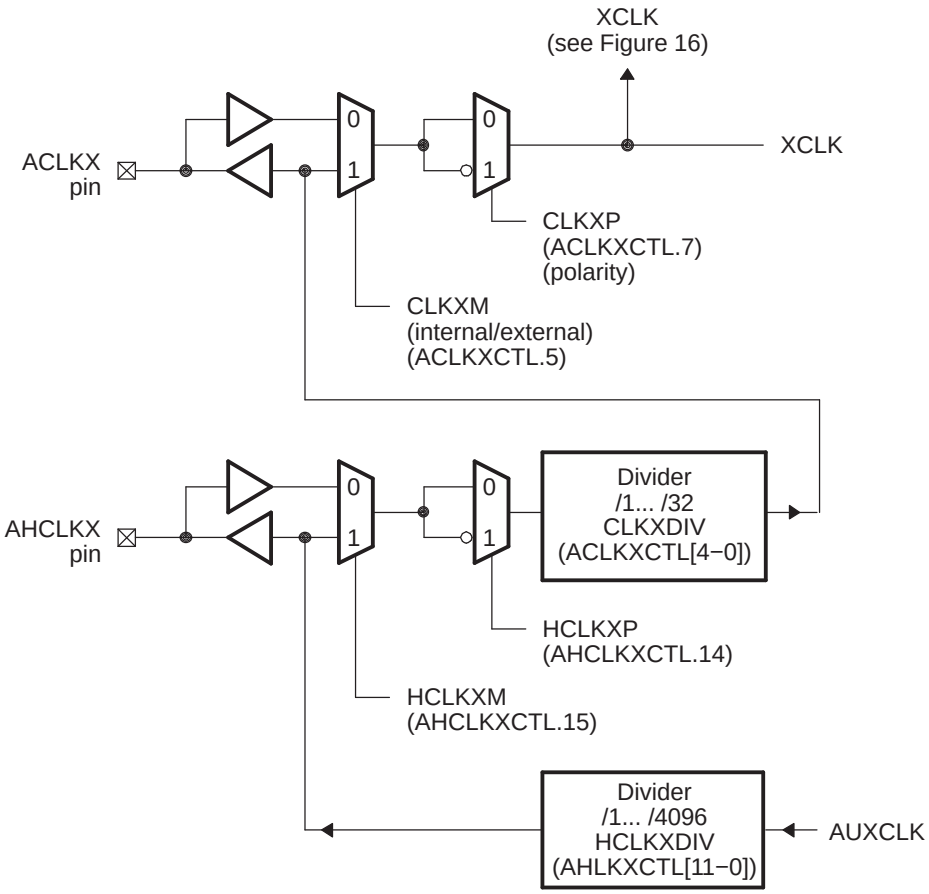
Internally, the McASP always shifts transmit data at the rising edge of the internal transmit clock, XCLK, (Figure 19-15). The CLKXP mux determines if ACLKX needs to be inverted to become XCLK. If CLKXP = 0, the CLKXP mux directly passes ACLKX to XCLK. As a result, the McASP shifts transmit data at the rising edge of ACLKX. If CLKXP = 1, the CLKX mux passes the inverted version of ACLKX to XCLK. As a result, the McASP shifts transmit data at the falling edge of ACLKX.

The transmit high-frequency master clock, AHCLKX, may be either externally sourced from the AHCLKX pin or internally generated, as selected by the HCLKXM bit. If internally generated (HCLKXM = 1), the clock is divided down by a programmable high clock divider (HCLKXDIV) from McASP internal clock source AUXCLK. The transmit high-frequency master clock may be (but is not required to be) output on the AHCLKX pin where it is available to other devices in the system.

The transmit clock configuration is controlled by the following registers:

- ACLKXCTL
- AHCLKXCTL

Figure 19-15. Transmit Clock Generator Block Diagram



Copyright © 2013–2016, Texas Instruments Incorporated

19.0.27.2 Receive Clock

The receiver has a clock generation circuit identical to (but independent of) that of the transmitter. The receive bit clock, ACLKR, (Figure 19-16) may be either externally sourced from the ACLKR pin or internally generated, as selected by the CLKRM bit. If internally generated (CLKRM = 1), the clock is divided down by a programmable divider (CLKRDIV) from the receive high-frequency master clock (AHCLKR). Regardless if ACLKR is either internally generated or externally sourced, polarity of the clock may be programmed (CLKRP) to be either rising or falling edge.

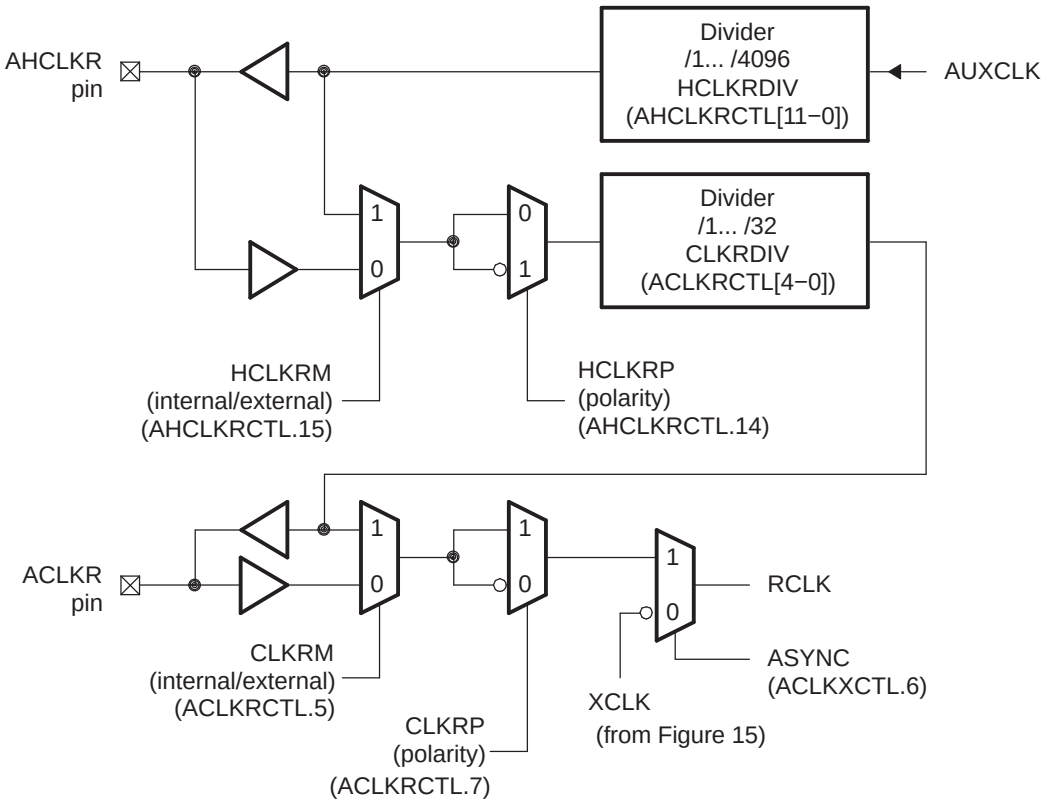
The receive high-frequency master clock, AHCLKR, may be either externally sourced from the AHCLKR pin or internally generated, as selected by the HCLKRM bit. If internally generated (HCLKRM = 1), the clock is divided down by a programmable divider (HCLKRDIV) from AUXCLK. The receive high-frequency master clock may be (but is not required to be) output on the AHCLKR pin where it is available to other devices in the system. Regardless if AHCLKR is either internally generated or externally sourced, polarity of the high-frequency clock may be programmed (HCLKRP) to be either rising or falling edge.

The receiver also has the option to operate synchronously from the ACLKX and AFSX signals. This is achieved when the ASYNC bit in the transmit clock control register (ACLKXCTL) is cleared to 0. See Section 19.0.27.1.5 for details on McASP operation when ACLKXCTL.ASYNC = 0.

The receive clock configuration is controlled by the following registers:

- ACLKRCTL
- AHCLKRCTL

Figure 19-16. Receive Clock Generator Block Diagram



Copyright © 2013–2016, Texas Instruments Incorporated

19.0.27.3 Frame Sync Generator

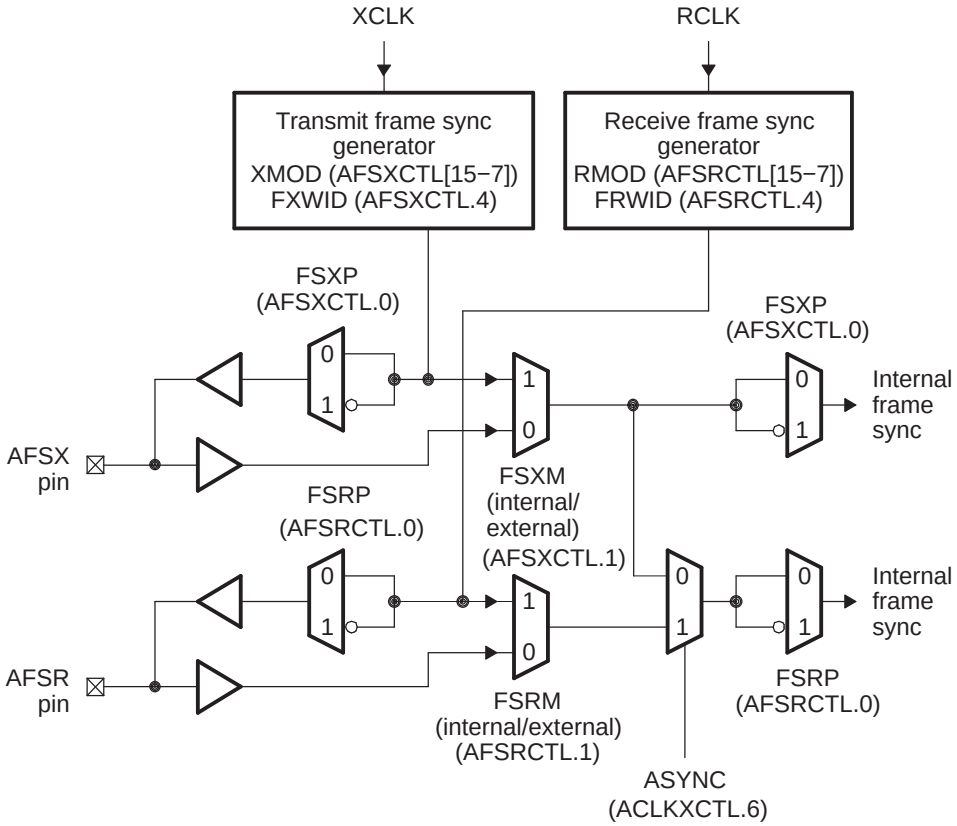
There are two different modes for frame sync: burst and TDM. A block diagram of the frame sync generator is shown in Figure 19-17. The frame sync options are programmed by the receive and transmit frame sync control registers (AFSRCTL and AFSXCTL). The options are:

- Internally-generated or externally-generated
- Frame sync polarity: rising edge or falling edge
- Frame sync width: single bit or single word
- Bit delay: 0, 1, or 2 cycles before the first data bit

The transmit frame sync pin is AFSX and the receive frame sync pin is AFSR. A typical usage for these pins is to carry the left/right clock (LRCLK) signal when transmitting and receiving stereo data.

Regardless if the AFSX/AFSR is internally generated or externally sourced, the polarity of AFSX/AFSR is determined by FSXP/FSRP, respectively, to be either rising or falling edge. If FSXP/FSRP = 0, the frame sync polarity is rising edge. If FSXP/FSRP = 1, the frame sync polarity is falling edge.

Figure 19-17. Frame Sync Generator Block Diagram



Copyright © 2013–2016, Texas Instruments Incorporated

19.0.27.4 Clocking Examples

Some examples of processes using the McASP clocking and frame flexibility are:

- Receive data from a DVD at 48 kHz, but output up-sampled or decoded audio at 96 kHz or 192 kHz. This could be accomplished by inputting a high-frequency master clock (for example, 512 × receive FS), receiving with an internally-generated bit clock ratio of divide-by-8, and transmitting with an internally-generated bit clock ratio of divide-by-4 or divide-by-2.
- Transmit/receive data based on one sample rate (for example, 44.1 kHz), and transmit/receive data at a different sample rate (for example, 48 kHz).

General Architecture

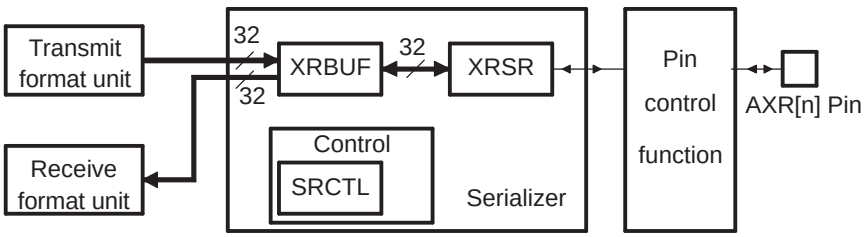
19.0.27.1 Serializers

The serializers take care of shifting serial data in and out of the McASP. Each serializer consists of a shift register (XRSR), a data buffer (XRBUF), a control register (SRCTL), and logic to support the data alignment options of the McASP. For each serializer, there is a dedicated serial data pin (AXR[n]) and a dedicated control register (SRCTL[n]). The control register allows the serializer to be configured as a transmitter, receiver, or as inactive. When configured as a transmitter the serializer shifts out data to the serial data pin AXR[n]. When configured as a receiver, the serializer shifts in data from the AXR[n] pin. The serializer is clocked from the transmit/receive section clock (ACLKX/ACLKR) if configured to transmit/receive respectively.

All serializers that are configured to transmit operate in lock-step. Similarly, all serializers that are configured to receive also operate in lock-step. This means that at most there are two zones per McASP, one for transmit and one for receive.

Figure 19-18 shows the block diagram of the serializer and its interface to other units within the McASP.

Figure 19-18. Individual Serializer and Connections Within McASP



For receive, data is shifted in through the AXR[n] pin to the shift register XRSR. Once the entire slot of data is collected in the XRSR, the data is copied to the data buffer XRBUF. The data is now ready to be read by the DSP through the RBUF register, which is an alias of the XRBUF for receive. When the DSP reads from the RBUF, the McASP passes the data from RBUF through the receive format unit and returns the formatted data to the DSP.

For transmit, the DSP services the McASP by writing data into the XBUF register, which is an alias of the XRBUF for transmit. The data automatically passes through the transmit format unit before actually reaching the XRBUF in the serializer. The data is then copied from XRBUF to XRSR, and shifted out from the AXR[n] synchronously to the serial clock.

In DIT mode, in addition to the data, the serializer shifts out other DIT-specific information accordingly (preamble, user data, etc.).

The serializer configuration is controlled by SRCTL[n].

Copyright © 2013–2016, Texas Instruments Incorporated

19.0.27.2 Format Unit

The McASP has two data formatting units, one for transmit and one for receive. These units automatically remap the data bits within the transmitted and received words between a natural format for the DSP (such as a Q31 representation) and the required format for the external serial device (such as "I2S format"). During the remapping process, the format unit also can mask off certain bits or perform sign extension.

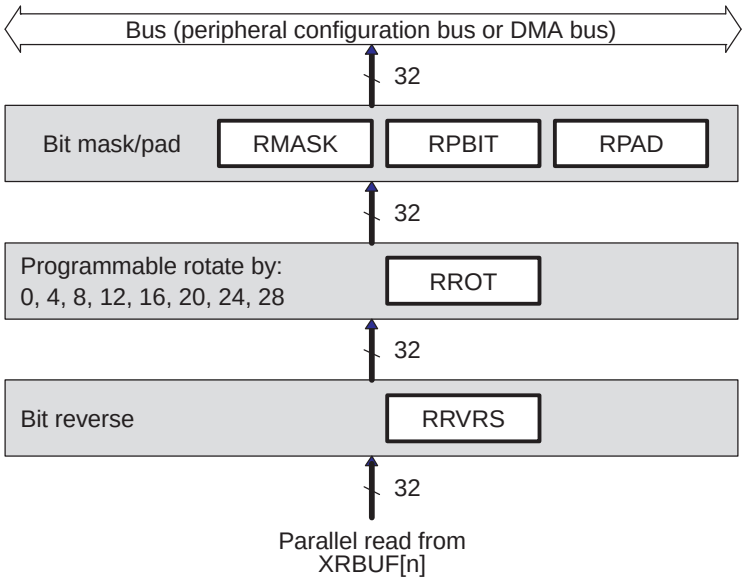
Since all transmitters share the same data formatting unit, the McASP only supports one transmit format at a time. For example, the McASP will not transmit in "I2S format" on serializer 0, while transmitting "Left Justified" on serializer 1. Likewise, the receiver section of the McASP only supports one data format at a time, and this format applies to all receiving serializers. However, the McASP can transmit in one format while receiving in a completely different format.

This formatting unit consists of three stages:

- Bit mask and pad (masks off bits, performs sign extension)
- Rotate right (aligns data within word)
- Bit reversal (selects between MSB first or LSB first)

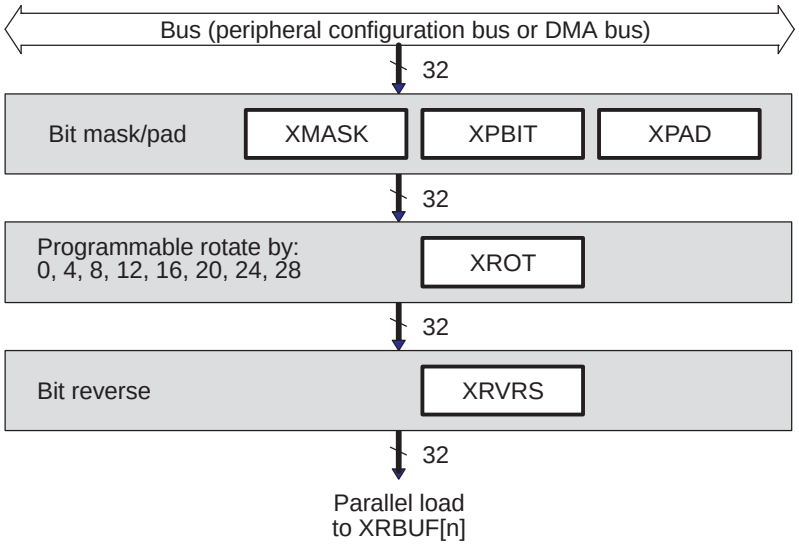
Figure 19-19 shows a block diagram of the receive formatting unit, and Figure 19-20 shows the transmit formatting unit. Note that the order in which data flows through the three stages is different between the transmit and receive formatting units.

Figure 19-19. Receive Format Unit



Copyright © 2013–2016, Texas Instruments Incorporated

Figure 19-20. Transmit Format Unit



The bit mask and pad stage includes a full 32-bit mask register, allowing selected individual bits to either pass through the stage unchanged, or be masked off. The bit mask and pad then pad the value of the masked off bits by inserting either a 0, a 1, or one of the original 32 bits as the pad value. The last option allows for sign-extension when the sign bit is selected to pad the remaining bits.

The rotate right stage performs bitwise rotation by a multiple of 4 bits (between 0 and 28 bits), programmable by the (R/X)FMT register. Note that this is a rotation process, not a shifting process, so bit 0 gets shifted back into bit 31 during the rotation.

The bit reversal stage either passes all 32 bits directly through, or swaps them. This allows for either MSB or LSB first data formats. If bit reversal is not enabled, then the McASP will naturally transmit and receive in an LSB first order.

Finally, note that the (R/X)DATDLY bits in (R/X)FMT also determine the data format. For example, the difference between I2S format and left-justified is determined by the delay between the frame sync edge and the first data bit of a given time slot. For I2S format, (R/X)DATDLY should be set to a 1-bit delay, whereas for left-justified format, it should be set to a 0-bit delay.

The combination of all the options in (R/X)FMT means that the McASP supports a wide variety of data formats, both on the serial data lines, and in the internal DSP representation.

Section 19.0.27.4 provides more detail and specific examples. The examples use internal representation in integer and Q31 notation, but other fractional notations are also possible.

19.0.27.3 State Machine

The receive and transmit sections have independent state machines. Each state machine controls the interactions between the various units in the respective section. In addition, the state machine keeps track of error conditions and serial port status.

No serial transfers can occur until the respective state machine is released from reset. See initialization sequence for details (Section 19.0.27.1).

The receive state machine is controlled by the RFMT register, and it reports the McASP status and error conditions in the RSTAT register. Similarly, the transmit state machine is controlled by the XFMT register, and it reports the McASP status and error conditions in the XSTAT register.

Copyright © 2013–2016, Texas Instruments Incorporated

19.0.27.4 TDM Sequencer

There are separate TDM sequencers for the transmit section and the receive section. Each TDM sequencer keeps track of the slot count. In addition, the TDM sequencer checks the bits of (R/X)TDM and determines if the McASP should receive/transmit in that time slot.

If the McASP should participate (transmit/receive bit is active) in the time slot, the McASP functions normally. If the McASP should not participate (transmit/receive bit is inactive) in the time slot, no transfers between the XRBUF and XRSR registers in the serializer would occur during that time slot. In addition, the serializers programmed as transmitters place their data output pins in a predetermined state (logic low, high, or high impedance) as programmed by each serializer control register (SRCTL). Refer also to Section 19.0.27.2.2 for details on how DMA event or interrupt generations are handled during inactive time slots in TDM mode.

The receive TDM sequencer is controlled by register RTDM and reports current receive slot to RSLOT. The transmit TDM sequencer is controlled by register XTDM and reports current transmit slot to XSLOT.

19.0.27.5 Clock Check Circuit

A common source of error in audio systems is a serial clock failure due to instabilities in the off-chip DIR circuit. To detect a clock error quickly, a clock-check circuit is included in the McASP for both transmit and receive clocks, since both may be sourced from off chip.

The clock check circuit can detect and recover from transmit and receive clock failures. See Section 19.0.27.6.6 for implementation and programming details.

19.0.27.6 Pin Function Control

All McASP pins except AMUTEIN are bidirectional input/output pins. In addition, these bidirectional pins function either as McASP or general-purpose I/O (GPIO) pins. The following registers control the pin functions:

- Pin function register (PFUNC): selects pin to function as McASP or GPIO
- Pin direction register (PDIR): selects pin to be input or output
- Pin data input register (PDIN): shows data input at the pin
- Pin data output register (PDOUT): data to be output at the pin if the pin is configured as GPIO output (PFUNC[n] = 1 and PDIR[n] = 1). Not applicable when the pin is configured as McASP pin (PFUNC[n] = 0).
- Pin data set register (PDSET): alias of PDOUT. Writing a 1 to PDSET[n] sets the respective PDOUT[n] to 1. Writing a 0 has no effect. Applicable only when the pin is configured as GPIO output (PFUNC[n] = 1 and PDIR[n] = 1).
- Pin data clear register (PDCLR): alias of PDOUT. Writing a 1 to PDCLR[n] clears the respective PDOUT[n] to 0. Writing a 0 has no effect. Applicable only when the pin is configured as GPIO output (PFUNC[n] = 1 and PDIR[n] = 1).

See the register descriptions in Section 19.1 for details on the mapping of each McASP pin to the register bits. Figure 19-21 shows the pin control block diagram.

19.0.27.6.1 McASP Pin Control-Transmit and Receive

You must correctly set the McASP GPIO registers PFUNC and PDIR, even when McASP pins are used for their serial port (non-GPIO) function.

Serial port functions include:

- Clock pins (ACLKX, ACLKR, AHCLKX, AHCLKR, AFSX, AFSR) used as clock inputs and outputs
- Serializer data pins (AXR[n]) used to transmit or receive
- AMUTE used as a mute output signal

When using these pins in their serial port function, you must clear PFUNC[n] to 0 for each pin, as opposed to PFUNC[n] = 1, which makes the pin a GPIO.

Copyright © 2013–2016, Texas Instruments Incorporated

Also, certain outputs require PDIR[n] = 1, such as clock pins used as clock outputs, serializer data pins used to transmit, and AMUTE used as mute output.

Clock inputs and serializers configured to receive must have PDIR[n] = 0.

PFUNC and PDIR do not control the AMUTEIN device pin, it is usually tied to a device pin (see your device-specific data manual). If used as a mute input, this pin needs to be configured as an input in the appropriate peripheral.

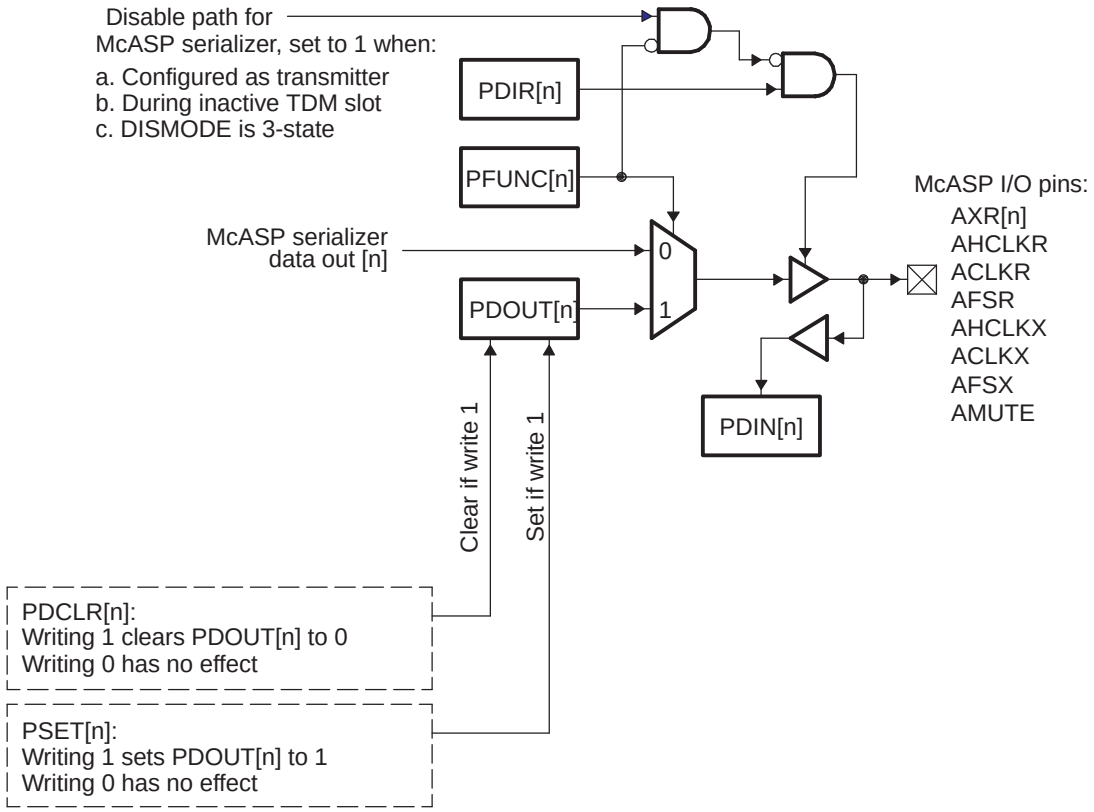
Finally, there is an important advantage to having separate control of pin direction (by PDIR), and the choice of internal versus external clocking (by CLKRM/CLKXM). Depending on the specific device and usage, you might select an external clock (CLKRM = 0), while enabling the internal clock divider, and the clock pin as an output in the PDIR register (PDIR[ACLKR] = 1). In this case, the bit clock is an output (PDIR[ACLKR] = 1) and, therefore, routed to the ACLKR pin. However, because CLKRM = 0, the bit clock is then routed back to the McASP module as an "external" clock source. This may result in less skew between the clock inside the McASP and the clock in the external device, thus producing more balanced setup and hold times for a particular system. As a result, this may allow a higher serial clock rate interface.

19.0.27.6.2 GPIO Pin Control

For GPIO operation, you must set the desired PFUNC[n] to 1 to indicate GPIO function. PDIR[n] must be configured to the desired direction. PDOUT, PDSET, PDCLR control the output value on the pin. PDIN always reflects the state at the pin, regardless of the PDIR and PFUNC setting.

Figure 19-21 and Figure 19-22 display the pin descriptions. The examples that follow (Example 19-1 through Example 19-4) show how the pins can be used as general-purpose input or output pins.

Figure 19-21. McASP I/O Pin Control Block Diagram



Copyright © 2013–2016, Texas Instruments Incorporated

Figure 19-22. McASP I/O Pin to Control Register Mapping

31	30	29	28	27	26	25	24
AFSR	AHCLKR	ACLKR	AFSX	AHCLKX	ACLKX	AMUTE	Reserved
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R-0

23	16
Reserved	
R-0	

15	14	13	12	11	10	9	8
AXR15	AXR14	AXR13	AXR12	AXR11	AXR10	AXR9	AXR8
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0

7	6	5	4	3	2	1	0
AXR7	AXR6	AXR5	AXR4	AXR3	AXR2	AXR1	AXR0
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Example 19-1. General-Purpose Input Pin

Because the PDIN register always reflects the state at the pin, you can read the PDIN register to obtain the pin input state. To explicitly set the pin as a general-purpose input pin, you can set the registers as follows:

- PDIR[n] = 0 (input)
- PFUNC[n] = 1 (GPIO function)

Example 19-2. General-Purpose Output Pin—Initialization Using PDOUT

All pins default as inputs. To initialize a pin as output, you should follow this sequence:

1. PDIR[n] = 0 (default as input)
2. PFUNC[n] = 1 (GPIO function)
3. PDOUT[n] = desired output value
4. PDIR[n] = 1 (change to output after desired value is configured in PDOUT[n])

Example 19-3. General-Purpose Output Pin—Change Data from 0 to 1 Using PDSET

If the pin is already configured as a general-purpose output pin driving a 0, and you want to change the output from 0 to 1, the recommended method is to use the PDSET register instead of the PDOUT register. This is because writing to the PDSET register only affects pin(s) in concern. To change a pin from 0 to 1:

- Set PDSET[n]. This sets the respective PDOUT[n].

Example 19-4. General-Purpose Output Pin—Change Data from 1 to 0 Using PDCLR

If the pin is already configured as a general-purpose output pin driving a 1, and you want to change the output from 1 to 0, the recommended method is to use the PDCLR register instead of the PDOUT register. This is because writing to the PDCLR register only affects pin(s) in concern. To change a pin from 1 to 0:

- Set PDCLR[n]. This clears the respective PDOUT[n].

Copyright © 2013–2016, Texas Instruments Incorporated

McASP Audio FIFO (AFIFO)

The McASP Audio FIFO (AFIFO) provides additional data buffering for the McASP. The time it takes the host CPU or DMA controller to respond to DMA requests from the McASP may vary; the additional buffering provided by the AFIFO allows greater tolerance to such variations.

For convenience, the AFIFO is treated here as a block between McASP and the host/DMA controller (see Figure 19-1). Details on configuration of the AFIFO are provided in McASP Audio FIFO (AFIFO).

Operation

This section discusses the operation of the McASP.

19.0.27.1 Setup and Initialization

This section discusses steps necessary to use the McASP module.

19.0.27.1.1 Considerations When Using a McASP

The following is a list of things to be considered for systems using a McASP:

19.0.27.1.1.1 Clocks

For each receive and transmit section:

- External or internal generated bit clock and high frequency clock?
- If internally generated, what is the bit clock speed and the high frequency clock speed?
- Clock polarity?
- External or internal generated frame sync?
- If internally generated, what is frame sync speed?
- Frame sync polarity?
- Frame sync width?
- Transmit and receive sync or asynchronous?

19.0.27.1.1.2 Data Pins

For each pin of each McASP:

- McASP or GPIO?
- Input or output?

19.0.27.1.1.3 Data Format

For each transmit and receive data:

- Internal numeric representation (integer, Q31 fraction)?
- I2S or DIT (transmit only)?
- Time slot delay (0, 1, or 2 bit)?
- Alignment (left or right)?
- Order (MSB first, LSB first)?
- Pad (if yes, pad with what value)?
- Slot size?
- Rotate?
- Mask?

Copyright © 2013–2016, Texas Instruments Incorporated

19.0.27.1.1.4 Data Transfers

- Internal: DMA or CPU?
- External: TDM or burst?
- Bus: peripheral configuration bus or DMA bus?

19.0.27.1.2 Transmit/Receive Section Initialization

You must follow the following steps to properly configure the McASP. If external clocks are used, they should be present prior to the following initialization steps.

1. Reset McASP to default values by setting GBLCTL = 0.
2. Configure McASP Audio FIFO. Recall that the Write FIFO and Read FIFO are enabled/disabled independently.
   (a) Write FIFO:
   - If the Write FIFO will not be enabled, verify that WFIFOCTL.WENA is cleared to 0 (the default value).
   - If the Write FIFO will be enabled, configure WFIFOCTL. Note that WFIFOCTL.WENA should not be set to 1 (enabled) until the other bitfields in this register are configured.

   (b) Read FIFO:
   - If the Read FIFO will not be enabled, verify that RFIFOCTL.RENA is cleared to 0 (the default value).
   - If the Read FIFO will be enabled, configure RFIFOCTL. Note that RFIFOCTL.RENA should not be set to 1 (enabled) until the other bitfields in this register are configured.
3. Configure all McASP registers except GBLCTL in the following order:
   (a) Receive registers: RMASK, RFMT, AFSRCTL, ACLKRCTL, AHCLKRCTL, RTDM, RINTCTL, RCLKCHK. If external clocks AHCLKR and/or ACLKR are used, they must be running already for proper synchronization of the GBLCTL register.
   (b) Transmit registers: XMASK, XFMT, AFSXCTL, ACLKXCTL, AHCLKXCTL, XTDM, XINTCTL, XCLKCHK. If external clocks AHCLKX and/or ACLKX are used, they must be running already for proper synchronization of the GBLCTL register.
   (c) Serializer registers: SRCTL[n].
   (d) Global registers: Registers PFUNC, PDIR, DITCTL, DLBCTL, AMUTE. Note that PDIR should only be programmed after the clocks and frames are set up in the steps above. This is because the moment a clock pin is configured as an output in PDIR, the clock pin starts toggling at the rate defined in the corresponding clock control register. Therefore you must ensure that the clock control register is configured appropriately before you set the pin to be an output. A similar argument applies to the frame sync pins.
   (e) DIT registers: For DIT mode operation, set up registers DITCSRA[n], DITCSRB[n], DITUDRA[n], and DITUDRB[n].
4. Start the respective high-frequency serial clocks AHCLKX and/or AHCLKR. This step is necessary even if external high-frequency serial clocks are used:
   (a) Take the respective internal high-frequency serial clock divider(s) out of reset by setting the RHCLKRST bit for the receiver and/or the XHCLKRST bit for the transmitter in GBLCTL. All other bits in GBLCTL should be held at 0.
   (b) Read back from GBLCTL to ensure the bit(s) to which you wrote are successfully latched in GBLCTL before you proceed.

Copyright © 2013–2016, Texas Instruments Incorporated

5. Start the respective serial clocks ACLKX and/or ACLKR. This step can be skipped if external serial clocks are used and they are running:
   (a) Take the respective internal serial clock divider(s) out of reset by setting the RCLKRST bit for the receiver and/or the XCLKRST bit for the transmitter in GBLCTL. All other bits in GBLCTL should be left at the previous state.
   (b) Read back from GBLCTL to ensure the bit(s) to which you wrote are successfully latched in GBLCTL before you proceed.
6. Setup data acquisition as required:
   (a) If DMA is used to service the McASP, set up data acquisition as desired and start the DMA in this step, before the McASP is taken out of reset.
   (b) If CPU interrupt is used to service the McASP, enable the transmit and/ or receive interrupt as required.
   (c) If CPU polling is used to service the McASP, no action is required in this step.
7. Activate serializers.
   (a) Before starting, clear the respective transmitter and receiver status registers by writing XSTAT = FFFFh and RSTAT = FFFFh.
   (b) Take the respective serializers out of reset by setting the RSRCLR bit for the receiver and/or the XSRCLR bit for the transmitter in GBLCTL. All other bits in GBLCTL should be left at the previous state.
   (c) Read back from GBLCTL to ensure the bit(s) to which you wrote are successfully latched in GBLCTL before you proceed.
8. Verify that all transmit buffers are serviced. Skip this step if the transmitter is not used. Also, skip this step if time slot 0 is selected as inactive (special cases, see Figure 19-24, second waveform). As soon as the transmit serializer is taken out of reset, XDATA in the XSTAT register is set, indicating that XBUF is empty and ready to be serviced. The XDATA status causes a DMA event AXEVT to be generated, and can cause an interrupt AXINT to be generated if it is enabled in the XINTCTL register.
   (a) If DMA is used to service the McASP, the DMA automatically services the McASP upon receiving AXEVT. Before proceeding in this step, you should verify that the XDATA bit in the XSTAT is cleared to 0, indicating that all transmit buffers are already serviced by the DMA.
   (b) If CPU interrupt is used to service the McASP, interrupt service routine is entered upon the AXINT interrupt. The interrupt service routine should service the XBUF registers. Before proceeding in this step, you should verify that the XDATA bit in XSTAT is cleared to 0, indicating that all transmit buffers are already serviced by the CPU.
   (c) If CPU polling is used to service the McASP, the XBUF registers should be written to in this step.

CAUTION

The DSP does not support the emulation suspend signal. Therefore, if a data window is open in the Code Composer Studio™ integrated development environment to observe the XRBUF locations, the emulation read from the XRBUF locations causes an undesirable side effect of clearing the RDATA bit in RSTAT. Furthermore, if you write to the XRBUF through the Code Composer Studio™ integrated development environment, the emulation write to the XRBUF locations causes the XDATA bit in XSTAT to be cleared.

9. Release state machines from reset.
   (a) Take the respective state machine(s) out of reset by setting the RSMRST bit for the receiver and/or the XSMRST bit for the transmitter in GBLCTL. All other bits in GBLCTL should be left at the previous state.
   (b) Read back from GBLCTL to ensure the bit(s) to which you wrote are successfully latched in GBLCTL before you proceed.

Copyright © 2013–2016, Texas Instruments Incorporated

10. Release frame sync generators from reset. Note that it is necessary to release the internal frame sync generators from reset, even if an external frame sync is being used, because the frame sync error detection logic is built into the frame sync generator.
    (a) Take the respective frame sync generator(s) out of reset by setting the RFRST bit for the receiver, and/or the XFRST bit for the transmitter in GBLCTL. All other bits in GBLCTL should be left at the previous state.
    (b) Read back from GBLCTL to ensure the bit(s) to which you wrote are successfully latched in GBLCTL before you proceed.
11. Upon the first frame sync signal, McASP transfers begin. The McASP synchronizes to an edge on the frame sync pin, not the level on the frame sync pin. This makes it easy to release the state machine and frame sync generators from reset.
    (a) For example, if you configure the McASP for a rising edge transmit frame sync, then you do not need to wait for a low level on the frame sync pin before releasing the McASP transmitter state machine and frame sync generators from reset.

19.0.27.1.3 Separate Transmit and Receive Initialization

In many cases, it is desirable to separately initialize the McASP transmitter and receiver. For example, you may delay the initialization of the transmitter until the type of data coming in on the receiver is recognized. Or a change in the incoming data stream on the receiver may necessitate a reinitialization of the transmitter.

In this case, you may still follow the sequence outlined in Section 19.0.27.1.2, but use it for each section (transmit, receive) individually. The GBLCTL register is aliased to RGBLCTL and XGBLCTL to facilitate separate initialization of transmit and receive sections.

Also, make sure that the initialization or reinitialization sequence follows the guidelines in Bits With Restrictions on When They May be Changed.

19.0.27.1.4 Importance of Reading Back GBLCTL

In Section 19.0.27.1.2, steps 4b, 5b, 7c, 9b, and 10b state that GBLCTL should be read back until the bits that were written are successfully latched. This is important, because the transmitter and receiver state machines run off of the respective bit clocks, which are typically about tens to hundreds of times slower than the DSP's internal bus clock. Therefore, it takes many cycles between when the DSP writes to GBLCTL (or RGBLCTL and XGBLCTL), and when the McASP actually recognizes the write operation. If you skip this step, then the McASP may never see the reset bits in the global control registers get asserted and deasserted; resulting in an uninitialized McASP.

Therefore, the logic in McASP has been implemented such that once the DSP writes GBLCTL, RGBLCTL, or XGBLCTL, the resulting write is not visible by reading back GBLCTL until the McASP has recognized the change. This typically requires two bit clocks plus two DSP bus clocks to occur.

Also, if the bit clocks can be completely stopped, any software that polls GBLCTL should be implemented with a time-out. If GBLCTL does not have a time-out, and the bit clock stops, the changes written to GBLCTL will not be reflected until the bit clock restarts.

Finally, please note that while RGBLCTL and XGBLCTL allow separate changing of the receive and transmit halves of GBLCTL, they also immediately reflect the updated value (useful for debug purposes). Only GBLCTL can be used for the read back step.

19.0.27.1.5 Synchronous Transmit and Receive Operation (ASYNC = 0)

When ASYNC = 0 in ACLKXCTL, the transmit and receive sections operate synchronously from the transmit section clock and transmit frame sync signals (Figure 19-15). The receive section may have a different (but compatible in terms of slot size) data format. Note that when ASYNC = 0, XCLK is automatically inverted to produce RCLK (note the inversion on the ASYNC multiplexer as shown in Figure 19-16).

Copyright © 2013–2016, Texas Instruments Incorporated

When ASYNC = 0, the transmit and receive sections must share some common settings, since they both use the same clock and frame sync signals:

- DITEN = 0 in DITCTL (TDM mode is enabled)
- The total number of bits per frame must be the same (that is, RSSZ × RMOD must equal XSSZ × XMOD)
- Both transmit and receive should either be specified as burst or TDM mode, but not mixed
- The settings in ACLKRCTL are irrelevant
- RCLK is an inverted version of XCLK (note the inversion on the multiplexer labeled “ASYNC” shown in Figure 19-16)
- FSXM must match FSRM
- FXWID must match FRWID

For all other settings, the transmit and receive sections may be programmed independently.

19.0.27.1.6 Asynchronous Transmit and Receive Operation (ASYNC = 1)

When ASYNC = 1 in ACLKXCTL, the transmit and receive sections operate completely independently and have separate clock and frame sync signals (Figure 19-15, Figure 19-16, and Figure 19-17). The events generated by each section come asynchronously.

19.0.27.2 Transfer Modes

19.0.27.2.1 Burst Transfer Mode

The McASP supports a burst transfer mode, which is useful for nonaudio data such as passing control information between two DSPs. Burst transfer mode uses a synchronous serial format similar to the TDM mode. The frame sync generation is not periodic or time-driven as in TDM mode, but data driven, and the frame sync is generated for each data word transferred.

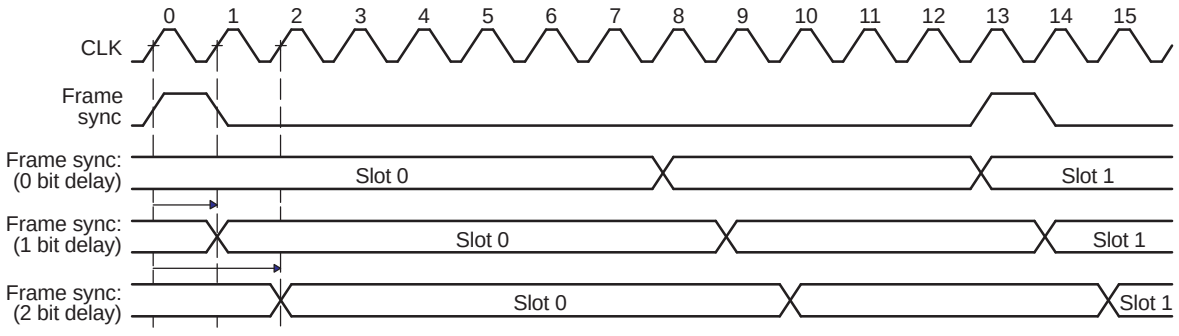
When operating in burst frame sync mode (Figure 19-23), as specified for transmit (XMOD = 0 in AFSXCTL) and receive (RMOD = 0 in AFSRCTL), one slot is shifted for each active edge of the frame sync signal that is recognized. Additional clocks after the slot and before the next frame sync edge are ignored.

In burst frame sync mode, the frame sync delay may be specified as 0, 1, or 2 serial clock cycles. This is the delay between the frame sync active edge and the start of the slot. The frame sync signal lasts for a single bit clock duration (FRWID = 0 in AFSRCTL, FXWID = 0 in AFSXCTL).

For transmit, when generating the transmit frame sync internally, the frame sync begins when the previous transmission has completed and when all the XBUF*n* (for every serializer set to operate as a transmitter) has been updated with new data.

For receive, when generating the receive frame sync internally, frame sync begins when the previous transmission has completed and when all the RBUF*n* (for every serializer set to operate as a receiver) has been read.

Figure 19-23. Burst Frame Sync Mode



Copyright © 2013–2016, Texas Instruments Incorporated

The control registers must be configured as follows for the burst transfer mode. The burst mode specific bit fields are in bold face:

- PFUNC: The clock, frame, data pins must be configured for McASP function.
- PDIR: The clock, frame, data pins must be configured to the direction desired.
- PDOUT, PDIN, PDSET, PDCLR: Not applicable. Leave at default.
- GBLCTL: Follow the initialization sequence in Section 19.0.27.1.2 to configure this register.
- AMUTE: Not applicable. Leave at default.
- DLBCTL: If loopback mode is desired, configure this register according to Section 19.0.27.7, otherwise leave this register at default.
- DITCTL: DITEN must be left at default 0 to select non-DIT mode. Leave the register at default.
- RMASK/XMASK: Mask desired bits according to Section 19.0.27.2 and Section 19.0.27.4.
- RFMT/XFMT: Program all fields according to data format desired. See Section 19.0.27.4.
- AFSRCTL/AFSXCTL: Clear **RMOD/XMOD** bits to 0 to indicate burst mode. Clear **FRWID/FXWID** bits to 0 for single bit frame sync duration. Configure other fields as desired.
- ACLKRCTL/ACLKXCTL: Program all fields according to bit clock desired. See Section 19.0.27.
- AHCLKRCTL/AHCLKXCTL: Program all fields according to high-frequency clock desired. See Section 19.0.27.
- RTDM/XTDM: Program RTDMS0/XTDMS0 to 1 to indicate one active slot only. Leave other fields at default.
- RINTCTL/XINTCTL: Program all fields according to interrupts desired.
- RCLKCHK/XCLKCHK: Not applicable. Leave at default.
- SRCTLn: Program SRMOD to inactive/transmitter/receiver as desired. DISMOD is not applicable and should be left at default.
- DITCSRA[n], DITCSRB[n], DITUDRA[n], DITUDRB[n]: Not applicable. Leave at default.

Copyright © 2013–2016, Texas Instruments Incorporated

19.0.27.2.2 Time-Division Multiplexed (TDM) Transfer Mode

The McASP time-division multiplexed (TDM) transfer mode supports the TDM format discussed in Section 19.0.24.1.

Transmitting data in the TDM transfer mode requires a minimum set of pins:

- ACLKX - transmit bit clock
- AFSX - transmit frame sync (or commonly called left/right clock)
- One or more serial data pins, AXR[n], whose serializers have been configured to transmit

The transmitter has the option to receive the ACLKX bit clock as an input, or to generate the ACLKX bit clock by dividing down the AHCLKX high-frequency master clock. The transmitter can either generate AHCLKX internally or receive AHCLKX as an input. See Section 19.0.27.1.

Similarly, to receive data in the TDM transfer mode requires a minimum set of pins:

- ACLKR - receive bit clock
- AFSR - receive frame sync (or commonly called left/right clock)
- One or more serial data pins, AXR[n], whose serializers have been configured to receive

The receiver has the option to receive the ACLKR bit clock as an input or to generate the ACLKR bit clock by dividing down the AHCLKR high-frequency master clock. The receiver can either generate AHCLKR internally or receive AHCLKR as an input. See Section 19.0.27.2 and Section 19.0.27.3.

The control registers must be configured as follows for the TDM mode. The TDM mode specific bit fields are in bold face:

- PFUNC: The clock, frame, data pins must be configured for McASP function.
- PDIR: The clock, frame, data pins must be configured to the direction desired.
- PDOUT, PDIN, PDSET, PDCLR: Not applicable. Leave at default.
- GBLCTL: Follow the initialization sequence in Section 19.0.27.1.2 to configure this register.
- AMUTE: Program all fields according to mute control desired.
- DLBCTL: If loopback mode is desired, configure this register according to Section 19.0.27.7, otherwise leave this register at default.
- DITCTL: DITEN must be left at default 0 to select TDM mode. Leave the register at default.
- RMASK/XMASK: Mask desired bits according to Section 19.0.27.2 and Section 19.0.27.4.
- RFMT/XFMT: Program all fields according to data format desired. See Section 19.0.27.4.
- AFSRCTL/AFSXCTL: Set **RMOD/XMOD** bits to 2-32 for TDM mode. Configure other fields as desired.
- ACLKRCTL/ACLKXCTL: Program all fields according to bit clock desired. See Section 19.0.27.
- AHCLKRCTL/AHCLKXCTL: Program all fields according to high-frequency clock desired. See Section 19.0.27.
- RTDM/XTDM: Program all fields according to the time slot characteristics desired.
- RINTCTL/XINTCTL: Program all fields according to interrupts desired.
- RCLKCHK/XCLKCHK: Program all fields according to clock checking desired.
- SRCTLn: Program all fields according to serializer operation desired.
- DITCSRA[n], DITCSRB[n], DITUDRA[n], DITUDRB[n]: Not applicable. Leave at default.

19.0.27.2.2.1 TDM Time Slots

TDM mode on the McASP can extend to support multiprocessor applications, with up to 32 time slots per frame. For each of the time slots, the McASP may be configured to participate or to be inactive by configuring XTDM and/or RTDM (this allows multiple DSPs to communicate on the same TDM serial bus).

The TDM sequencer (separate ones for transmit and receive) functions in this mode. The TDM sequencer counts the slots beginning with the frame sync. For each slot, the TDM sequencer checks the respective bit in either XTDM or RTDM to determine if the McASP should transmit/receive in that time slot.

Copyright © 2013–2016, Texas Instruments Incorporated

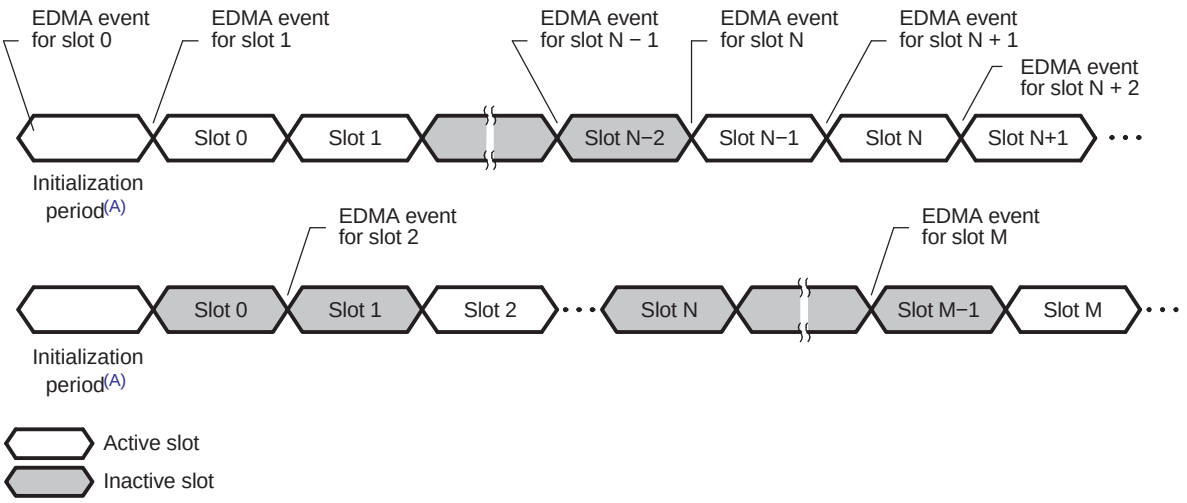
If the transmit/receive bit is active, the McASP functions normally during that time slot; otherwise, the McASP is inactive during that time slot; no update to the buffer occurs, and no event is generated. Transmit pins are automatically set to a high-impedance state, 0, or 1 during that slot, as determined by bit DISMOD in SRCTL[n].

Figure 19-24 shows when the transmit DMA event AXEVT is generated. See Section 19.0.27.3.1 for details on data ready and the initialization period indication. The transmit DMA event for an active time slot (slot N) is generated during the previous time slot (slot N - 1), regardless if the previous time slot (slot N - 1) is active or inactive.

During an active transmit time slot (slot N), if the next time slot (slot N + 1) is configured to be active, the copy from XRBUF[n] to XRSR[n] generates the DMA event for time slot N + 1. If the next time slot (slot N + 1) is configured to be inactive, then the DMA event will be delayed to time slot M - 1. In this case, slot M is the next active time slot. The DMA event for time slot M is generated during the first bit time of slot M - 1.

The receive DMA request generation does not need this capability, since the receive DMA event is generated after data is received in the buffer (looks back in time). If a time slot is disabled, then no data is copied to the buffer for that time slot and no DMA event is generated.

Figure 19-24. Transmit DMA Event (AXEVT) Generation in TDM Time Slots



A See Section 19.0.27.1.2, step 7a.

19.0.27.2.2.2 Special 384 Slot TDM Mode for Connection to External DIR

The McASP receiver also supports a 384 time slot TDM mode (DIR mode), to support S/PDIF, AES-3, IEC-60958 receiver ICs whose natural block (block corresponds to McASP frame) size is 384 samples. The advantage to using the 384 time slot TDM mode is that interrupts may be generated synchronous to the S/PDIF, AES-3, IEC-60958, such as the last slot interrupt.

The receive TDM time slot register (RTDM) should be programmed to all 1s during reception of a DIR block. Other TDM functionalities (for example, inactive slots) are not supported (only the slot counter counts the 384 subframes in a block).

To receive data in the DIR mode, the following pins are typically needed:

- ACLKR - receive bit clock.
- AFSR - receive frame sync (or commonly called left/right clock). In this mode, AFSR should be connected to a DIR which outputs a start of block signal, instead of LRCLK.
- One or more serial data pins, AXR[n], whose serializers have been configured to receive.

For this special DIR mode, the control registers can be configured just as for TDM mode, except set RMOD in AFSRCTL to 384 to receive 384 time slots.

Copyright © 2013–2016, Texas Instruments Incorporated

19.0.27.2.3 Digital Audio Interface Transmit (DIT) Transfer Mode

In addition to the TDM and burst transfer modes, which are suitable for transmitting audio data between ICs inside the same system, the digital audio interface transmit (DIT) transfer mode of the McASP also supports transmission of audio data in the S/PDIF, AES-3, or IEC-60958 format. These formats are designed to carry audio data between different systems through an optical or coaxial cable. The DIT mode only applies to serializers configured as transmitters, not receivers. Refer to Section 19.0.24.2 for a description of the S/PDIF format.

19.0.27.2.3.1 Transmit DIT Encoding

The McASP operation in DIT mode is basically identical to the 2 time slot TDM mode, but the data transmitted is output as a biphase mark encoded bit stream, with preamble, channel status, user data, validity, and parity automatically stuffed into the bit stream by the McASP. The McASP includes separate validity bits for even/odd subframes and two 384-bit RAM modules to hold channel status and user data bits.

The transmit TDM time slot register (XTDM) should be programmed to all 1s during DIT mode. TDM functionality is not supported in DIT mode, except that the TDM slot counter counts the DIT subframes.

To transmit data in the DIT mode, the following pins are typically needed:

- AHCLKX - transmit high-frequency master clock
- One or more serial data pins, AXR[n], whose serializers have been configured to transmit

AHCLKX is optional (the internal clock source may be used instead), but if used as a reference, the DSP provides a clock check circuit that continually monitors the AHCLKX input for stability.

If the McASP is configured to transmit in the DIT mode on more than one serial data pin, the bit streams on all pins will be synchronized. In addition, although they will carry unique audio data, they will carry the same channel status, user data, and validity information.

The actual 24-bit audio data must always be in bit positions 23-0 after passing through the first three stages of the transmit format unit.

For left-aligned Q31 data, the following transmit format unit settings process the data into right aligned 24-bit audio data ready for transmission:

- XROT = 010 (rotate right by 8 bits)
- XRVRS = 0 (no bit reversal, LSB first)
- XMASK = FFFF FF00h-FFFF 0000h (depending upon whether 24, 23, 22, 21, 20, 19, 18, 17, or 16 valid audio data bits are present)
- XPAD = 00 (pad extra bits with 0)

For right-aligned data, the following transmit format unit settings process the data into right aligned 24-bit audio data ready for transmission:

- XROT = 000 (rotate right by 0 bits)
- XRVRS = 0 (no bit reversal, LSB first)
- XMASK = 00FF FFFFh to 0000 FFFFh (depending upon whether 24, 23, 22, 21, 20, 19, 18, 17, or 16 valid audio data bits are present)
- XPAD = 00 (pad extra bits with 0)

Copyright © 2013–2016, Texas Instruments Incorporated

19.0.27.2.3.2 Transmit DIT Clock and Frame Sync Generation

The DIT transmitter only works in the following configuration:

- In transmit frame control register (AFSXCTL):
  - Internally-generated transmit frame sync, FSXM = 1
  - Rising-edge frame sync, FSXP = 0
  - Bit-width frame sync, FXWID = 0
  - 384-slot TDM, XMOD = 1 1000 0000b
- In transmit clock control register (ACLKXCTL), ASYNC = 1
- In transmit bitstream format register (XFMT), XSSZ = 1111 (32-bit slot size)

All combinations of AHCLKX and ACLKX are supported.

This is a summary of the register configurations required for DIT mode. The DIT mode specific bit fields are in bold face:

- PFUNC: The data pins must be configured for McASP function. If AHCLKX is used, it must also be configured for McASP function. Other pins can be configured to function as GPIO if desired.
- PDIR: The data pins must be configured as outputs. If AHCLKX is used as an input reference, it should be configured as input. If internal clock source AUXCLK is used as the reference clock, it may be output on the AHCLKX pin by configuring AHCLKX as an output.
- PDOUT, PDIN, PDSET, PDCLR: Not applicable for DIT operation. Leave at default.
- GBLCTL: Follow the initialization sequence in Section 19.0.27.1.2 to configure this register.
- AMUTE: Program all fields according to mute control desired.
- DLBCTL: Not applicable. Loopback is not supported for DIT mode. Leave at default.
- DITCTL: **DITEN** bit must be set to 1 to enable DIT mode. Configure other bits as desired.
- RMASK: Not applicable. Leave at default.
- RFMT: Not applicable. Leave at default.
- AFSRCTL: Not applicable. Leave at default.
- ACLKRCTL: Not applicable. Leave at default.
- AHCLKRCTL: Not applicable. Leave at default.
- RTDM: Not applicable. Leave at default.
- RINTCTL: Not applicable. Leave at default.
- RCLKCHK: Not applicable. Leave at default.
- **XMASK**: Mask desired bits according to the discussion in this section, depending upon left-aligned or right-aligned internal data.
- **XFMT: XDATDLY** = 0. **XRVRS** = 0. **XPAD** = 0. **XPBIT** = default (not applicable). **XSSZ** = Fh (32-bit slot). XBUSEL = configured as desired. **XROT** bit is configured according to the discussion in this section, either 0 or 8-bit rotate.
- **AFSXCTL**: Configure the bits according to the discussion in this section.
- **ACLKXCTL**: **ASYNC** = 1. Program CLKXDIV bits to obtain the bit clock rate desired. Configure CLKXP and CLKXM bits as desired, because CLKX is not actually used in the DIT protocol.
- **AHCLKXCTL**: Program all fields according to high-frequency clock desired.
- **XTDM**: Set to FFFF FFFFh for all active slots for DIT transfers.
- XINTCTL: Program all fields according to interrupts desired.
- XCLKCHK: Program all fields according to clock checking desired.
- SRCTLn: Set **SRMOD** = 1 (transmitter) for the DIT pins. DISMOD field is don't care for DIT mode.
- **DITCSRA[n], DITCSRB[n]**: Program the channel status bits as desired.
- **DITUDRA[n], DITUDRB[n]**: Program the user data bits as desired.

Copyright © 2013–2016, Texas Instruments Incorporated

19.0.27.2.3.3 DIT Channel Status and User Data Register Files

The channel status registers (DITCSRA*n* and DITCSRB*n*) and user data registers (DITUDRA*n* and DITUDRB*n*) are not double buffered. Typically the programmer uses one of the synchronizing interrupts, such as last slot, to create an event at a safe time so the register may be updated. In addition, the CPU reads the transmit TDM slot counter to determine which word of the register is being used.

It is a requirement that the software avoid writing to the word of user data and channel status that are being used to encode the current time slot; otherwise, it will be indeterminate whether the old or new data is used to encode the bitstream.

The DIT subframe format is defined in Section 19.0.24.2.2. The channel status information (C) and user data (U) are defined in these DIT control registers:

- DITCSRA0 to DITCSRA5: The 192 bits in these six registers contain the channel status information for the LEFT channel within each frame.
- DITCSRB0 to DITCSRB5: The 192 bits in these six registers contain the channel status information for the RIGHT channel within each frame.
- DITUDRA0 to DITUDRA5: The 192 bits in these six registers contain the user data information for the LEFT channel within each frame.
- DITUDRB0 to DITUDRB5: The 192 bits in these six registers contain the user data information for the RIGHT channel within each frame.

The S/PDIF block format is shown in Figure 19-11. There are 192 frames within a block (frame 0 to frame 191). Within each frame there are two subframes (subframe 1 and 2 for left and right channels, respectively). The channel status and user data information sent on each subframe is summarized in Table 19-3.

19.0.27.3 Data Transmission and Reception

The DSP services the McASP by writing data to the XBUF register(s) for transmit operations, and by reading data from the RBUF register(s) for receive operations. The McASP sets status flag and notifies the DSP whenever data is ready to be serviced. Section 19.0.27.3.1 discusses data ready status in detail.

The XBUF and RBUF registers can be accessed through one of the two peripheral ports of the device:

- The DMA port: This port is dedicated for data transfers on the device.
- The peripheral configuration port: This port is used for both data transfers and peripheral configuration control on the device.

Section 19.0.27.3.2 and Section 19.0.27.3.3 discuss how to perform transfers through the DMA bus and the peripheral configuration bus.

Either the CPU or the DMA can be used to service the McASP through any of these two peripheral ports. The CPU and DMA usages are discussed in Section 19.0.27.3.4 and Section 19.0.27.3.5.

Copyright © 2013–2016, Texas Instruments Incorporated

Table 19-3. Channel Status and User Data for Each DIT Block

Frame	Subframe	Preamble	Channel Status defined in:	User Data defined in:
Defined by DITCSRA0, DITCSRB0, DITUDRA0, DITUDRB0				
0	1 (L)	B	DITCSRA0[0]	DITUDRA0[0]
0	2 (R)	W	DITCSRB0[0]	DITUDRB0[0]
1	1 (L)	M	DITCSRA0[1]	DITUDRA0[1]
1	2 (R)	W	DITCSRB0[1]	DITUDRB0[1]
2	1 (L)	M	DITCSRA0[2]	DITUDRA0[2]
2	2 (R)	W	DITCSRB0[2]	DITUDRB0[2]
…	…	…	…	…
31	1 (L)	M	DITCSRA0[31]	DITUDRA0[31]
31	2 (R)	W	DITCSRB0[31]	DITUDRB0[31]
Defined by DITCSRA1, DITCSRB1, DITUDRA1, DITUDRB1				
32	1 (L)	M	DITCSRA1[0]	DITUDRA1[0]
32	2 (R)	W	DITCSRB1[0]	DITUDRB1[0]
…	…	…	…	…
63	1 (L)	M	DITCSRA1[31]	DITUDRA1[31]
63	2 (R)	W	DITCSRB1[31]	DITUDRB1[31]
Defined by DITCSRA2, DITCSRB2, DITUDRA2, DITUDRB2				
64	1 (L)	M	DITCSRA2[0]	DITUDRA2[0]
64	2 (R)	W	DITCSRB2[0]	DITUDRB2[0]
…	…	…	…	…
95	1 (L)	M	DITCSRA2[31]	DITUDRA2[31]
95	2 (R)	W	DITCSRB2[31]	DITUDRB2[31]
Defined by DITCSRA3, DITCSRB3, DITUDRA3, DITUDRB3				
96	1 (L)	M	DITCSRA3[0]	DITUDRA3[0]
96	2 (R)	W	DITCSRB3[0]	DITUDRB3[0]
…	…	…	…	…
127	1 (L)	M	DITCSRA3[31]	DITUDRA3[31]
127	2 (R)	W	DITCSRB3[31]	DITUDRB3[31]
Defined by DITCSRA4, DITCSRB4, DITUDRA4, DITUDRB4				
128	1 (L)	M	DITCSRA4[0]	DITUDRA4[0]
128	2 (R)	W	DITCSRB4[0]	DITUDRB4[0]
…	…	…	…	…
159	1 (L)	M	DITCSRA4[31]	DITUDRA4[31]
159	2 (R)	W	DITCSRB4[31]	DITUDRB4[31]
Defined by DITCSRA5, DITCSRB5, DITDRA5, DITUDRB5				
160	1 (L)	M	DITCSRA5[0]	DITUDRA5[0]
160	2 (R)	W	DITCSRB5[0]	DITUDRB5[0]
…	…	…	…	…
191	1 (L)	M	DITCSRA5[31]	DITUDRA5[31]
191	2 (R)	W	DITCSRB5[31]	DITUDRB5[31]

Copyright © 2013–2016, Texas Instruments Incorporated

19.0.27.3.1 Data Ready Status and Event/Interrupt Generation

19.0.27.3.1.1 Transmit Data Ready

The transmit data ready flag XDATA bit in the XSTAT register reflects the status of the XBUF register. The XDATA flag is set when data is transferred from the XRBUF[n] buffers to the XRSR[n] shift registers, indicating that the XBUF is empty and ready to accept new data from the DSP. This flag is cleared when the XDATA bit is written with a 1, or when all the serializers configured as transmitters are written by the DSP.

Whenever XDATA is set, an DMA event AXEVT is automatically generated to notify the DMA of the XBUF empty status. An interrupt AXINT is also generated if XDATA interrupt is enabled in the XINTCTL register (See Section 19.0.27.5.1 for details).

For DMA requests, the McASP does not require XSTAT to be read between DMA events. This means that even if XSTAT already has the XDATA flag set to 1 from a previous request, the next transfer triggers another DMA request.

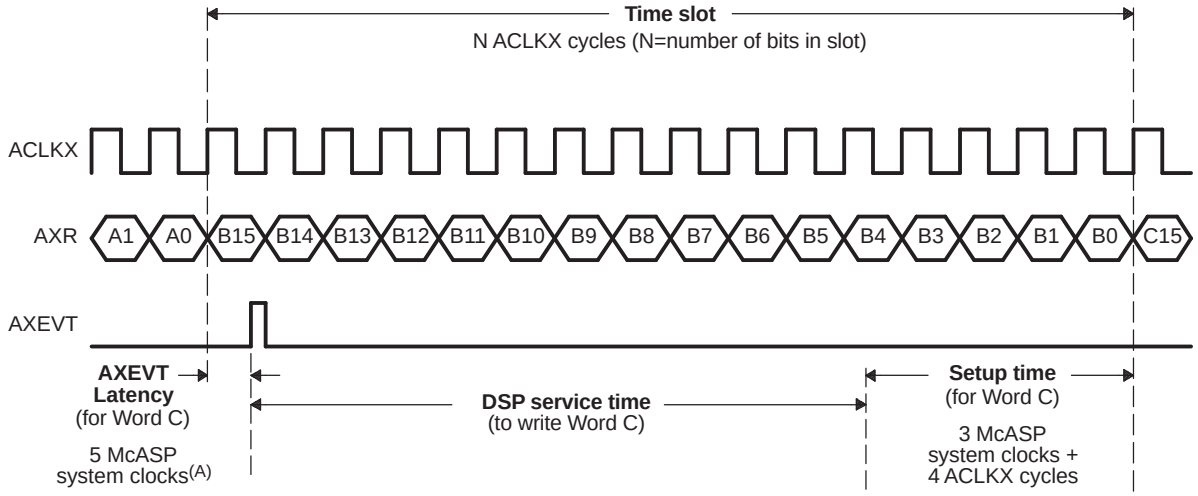
Since all serializers act in lockstep, only one DMA event is generated to indicate that all active transmit serializers are ready to be written to with new data.

Figure 19-25 shows the timing details of when AXEVT is generated at the McASP boundary. In this example, as soon as the last bit (bit A0) of Word A is transmitted, the McASP sets the XDATA flag and generates an AXEVT event. However, it takes up to 5 McASP system clocks (AXEVT Latency) before AXEVT is active at the McASP boundary. Upon AXEVT, the DSP can begin servicing the McASP by writing Word C into the XBUF (DSP Service Time). The DSP must write Word C into the XBUF no later than the setup time required by the McASP (Setup Time).

The maximum DSP Service Time (Figure 19-25) can be calculated as:

DSP Service Time = Time Slot - AXEVT Latency - Setup Time

Figure 19-25. DSP Service Time Upon Transmit DMA Event (AXEVT)



A This is not the same as AUXCLK. The DSP uses SYSCLK2 as the McASP system clock source.

Copyright © 2013–2016, Texas Instruments Incorporated

19.0.27.3.1.2 Receive Data Ready

Similarly, the receive data ready flag RDATA bit in the RSTAT reflects the status of the RBUF register. The RDATA flag is set when data is transferred from the XRSR[n] shift registers to the XRBUF[n] buffers, indicating that the RBUF contains received data and is ready to have the DSP read the data. This flag is cleared when the RDATA bit is written with a 1, or when all the serializers configured as receivers are read.

Whenever RDATA is set, an DMA event AREVT is automatically generated to notify the DMA of the RBUF ready status. An interrupt ARINT is also generated if RDATA interrupt is enabled in the RINTCTL register (See Section 19.0.27.5.2 for details).

For DMA requests, the McASP does not require RSTAT to be read between DMA events. This means that even if RSTAT already has the RDATA flag set to 1 from a previous request, the next transfer triggers another DMA request.

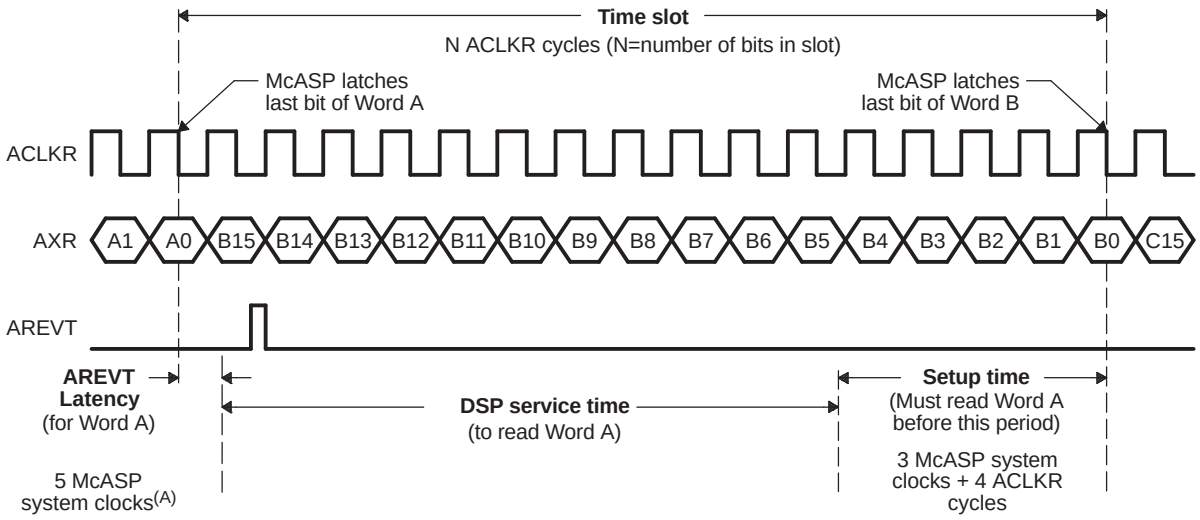
Since all serializers act in lockstep, only one DMA event is generated to indicate that all active receive serializers are ready to receive new data.

Figure 19-26 shows the timing details of when AREVT is generated at the McASP boundary. In this example, as soon as the last bit (bit A0) of Word A is received, the McASP sets the RDATA flag and generates an AREVT event. However, it takes up to 5 McASP system clocks (AREVT Latency) before AREVT is active at the McASP boundary. Upon AREVT, the DSP can begin servicing the McASP by reading Word A from the RBUF (DSP Service Time). The DSP must read Word A from the XBUF no later than the setup time required by the McASP (Setup Time).

The maximum DSP Service Time (Figure 19-26) can be calculated as:

DSP Service Time = Time Slot - AREVT Latency - Setup Time

Figure 19-26. DSP Service Time Upon Receive DMA Event (AREVT)



A This is not the same as AUXCLK. The DSP uses SYSCLK2 as the McASP system clock source.

Copyright © 2013–2016, Texas Instruments Incorporated

19.0.27.3.2 Transfers through the DMA Port

CAUTION

To perform internal transfers through the DMA port, clear XBUSEL/RBUSEL bit to 0 in the respective XFMT/RFMT registers. Failure to do so will result in software malfunction.

Typically, you will access the McASP XRBUF registers through the DMA port. To access through the DMA port, simply have the CPU or DMA access the XRBUF through its DMA port location. See your device-specific data manual for the exact memory address. Through the DMA port, the DMA/CPU can service all the serializers through a single address. The McASP automatically cycles through the appropriate serializers.

For transmit operations through the DMA port, the DMA/CPU should write to the same XBUF DMA port address to service all of the active transmit serializers. In addition, the DMA/CPU should write to the XBUF for all active transmit serializers in incremental (although not necessarily consecutive) order. For example, if serializers 0, 4, 5, and 7 are set up as active transmitters, the DMA/CPU should write to the XBUF DMA port address four times with data for serializers 0, 4, 5, and 7 upon each transmit data ready event. This exact servicing order must be followed so that data appears in the appropriate serializers.

Similarly, for receive operations through the DMA port, the DMA/CPU should read from the same RBUF DMA port address to service all of the active receive serializers. In addition, reads from the active receive serializers through the DMA port return data in incremental (although not necessarily consecutive) order. For example, if serializers 1, 2, 3, and 6 are set up as active receivers, the DMA/CPU should read from the RBUF DMA port address four times to obtain data for serializers 1, 2, 3, and 6 in this exact order, upon each receive data ready event.

When transmitting, the DMA/CPU must write data to each serializer configured as "active" and "transmit" within each time slot. Failure to do so results in a buffer underrun condition (Section 19.0.27.6.2). Similarly, when receiving, data must be read from each serializer configured as "active" and "receive" within each time slot. Failure to do results in a buffer overrun condition (Section 19.0.27.6.3).

To perform internal transfers through the DMA port, clear XBUSEL/RBUSEL bit to 0 in the respective XFMT/RFMT registers.

19.0.27.3.3 Transfers Through the Peripheral Configuration Bus

CAUTION

The DSP does not support the emulation suspend signal. Therefore, if a data window is open in the Code Composer Studio™ integrated development environment to observe the XRBUF locations, the emulation read from the XRBUF locations causes an undesirable side effect of clearing the RDATA bit in RSTAT. Furthermore, if you write to the XRBUF through the Code Composer Studio™ integrated development environment, the emulation write to the XRBUF locations causes the XDATA bit in XSTAT to be cleared.

To perform internal transfers through the peripheral configuration bus, set XBUSEL/RBUSEL bit to 1 in the respective XFMT/RFMT registers. Failure to do so will result in software malfunction.

In this method, the DMA/CPU accesses the XRBUF through the peripheral configuration bus address. The exact XRBUF address for any particular serializer is determined by adding the offset for that particular serializer to the base address for the particular McASP (found in the device-specific data manual). XRBUF for the serializers configured as transmitters is given the name XBUF*n*. For example, the XRBUF associated with transmit serializer 2 is named XBUF2. Similarly, XRBUF for the serializers configured as receivers is given the name RBUF*n*.

Copyright © 2013–2016, Texas Instruments Incorporated

Accessing the XRBUF through the DMA port is different because the CPU/DMA only needs to access one single address. When accessing through the peripheral configuration bus, the CPU/DMA must provide the exact XBUF*n* or RBUF*n* address for each access.

When transmitting, DMA/CPU must write data to each serializer configured as "active" and "transmit" within each time slot. Failure to do so results in a buffer underrun condition (Section 19.0.27.6.2). Similarly when receiving, data must be read from each serializer configured as "active" and "receive" within each time slot. Failure to do results in a buffer overrun condition (Section 19.0.27.6.3).

19.0.27.3.4 Using the CPU for McASP Servicing

The CPU can be used to service the McASP through interrupt (upon AXINT/ARINT interrupts) or through polling the XDATA bit in the XSTAT register. As discussed in Section 19.0.27.3.2 and Section 19.0.27.3.3, the CPU can access either through the DMA port or through the peripheral configuration port.

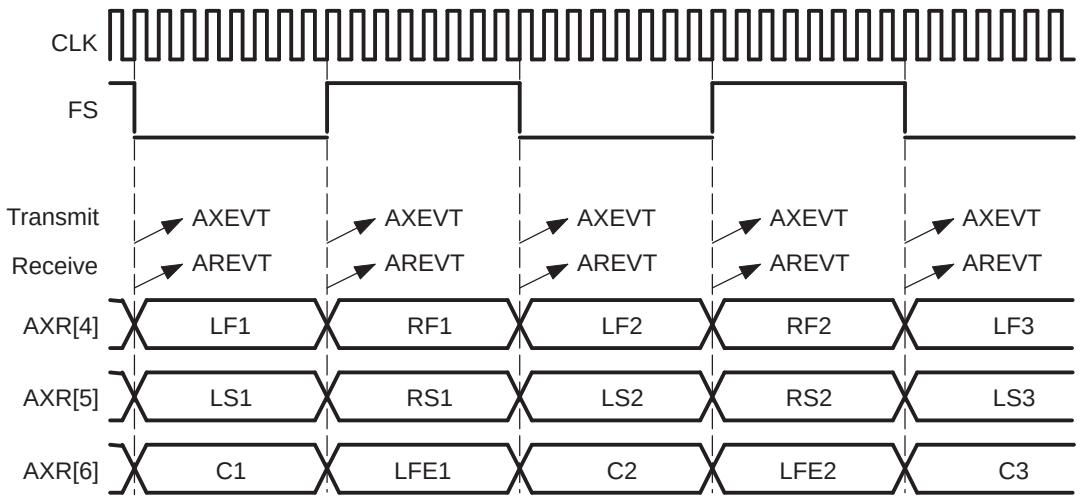
To use the CPU to service the McASP through interrupts, the XSTAT/RSTAT bit must be enabled in the respective XINTCTL/RINTCTL registers, to generate interrupts AXINT/ARINT to the CPU upon data ready.

19.0.27.3.5 Using the DMA for McASP Servicing

The most typical scenario is to use the DMA to service the McASP through the DMA port, although the DMA can also service the McASP through the peripheral configuration port. Use AXEVT/AREVT that is triggered upon each XDATA/RDATA transition from 0 to 1.

Figure 19-27 shows an example audio system with six audio channels (LF, RF, LS, RS, C, and LFE) transmitted from three AXR[n] pins on the McASP and shows when events AXEVT and AREVT are triggered.

Figure 19-27. DMA Events in an Audio Example–Two Events



In Figure 19-27, a DMA event AXEVT/AREVT is triggered on each time slot. In the example, AXEVT is triggered for each of the transmit audio channel time slot (time slot for channels LF, LS, and C; and time slot for channels RF, RS, LFE). Similarly, AREVT is triggered for each of the receive audio channel time slot. This allows for the use of a single DMA to transmit all audio channels, and a single DMA to receive all audio channels.

Note the difference between DMA event generation and the CPU interrupt generation. DMA events are generated automatically upon data ready; whereas CPU interrupt generation needs to be enabled in the XINTCTL/RINTCTL register.

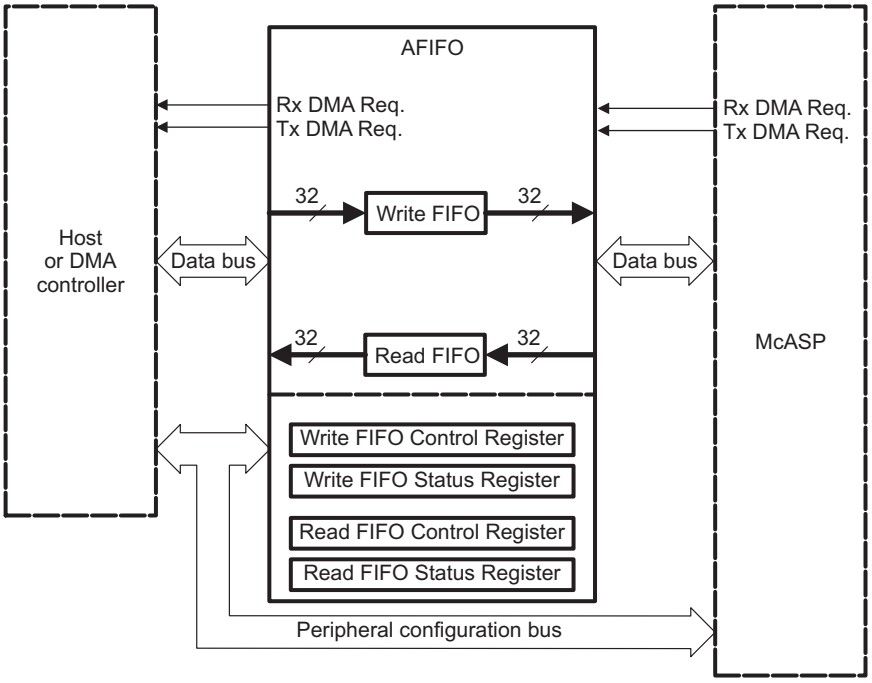
Copyright © 2013–2016, Texas Instruments Incorporated

McASP Audio FIFO (AFIFO)

The AFIFO contains two FIFOs: one Read FIFO (RFIFO), and one Write FIFO (WFIFO). To ensure backward compatibility with existing software, both the Read and Write FIFOs are disabled by default. See Figure 19-28 for a high-level block diagram of the AFIFO.

The AFIFO may be enabled/disabled and configured via the WFIFOCTL and RFIFOCTL registers. Note that if the Read or Write FIFO is to be enabled, it must be enabled prior to initializing the receive/transmit section of the McASP (see Section 19.0.27.1.2 for details).

Figure 19-28. McASP Audio FIFO (AFIFO) Block Diagram



AFIFO Data Transmission

When the Write FIFO is disabled, transmit DMA requests pass through directly from the McASP to the host/DMA controller. Whether the WFIFO is enabled or disabled, the McASP generates transmit DMA requests as needed; the AFIFO is “invisible” to the McASP.

When the Write FIFO is enabled, transmit DMA requests from the McASP are sent to the AFIFO, which in turn generates transmit DMA requests to the host/DMA controller.

If the Write FIFO is enabled, upon a transmit DMA request from the McASP, the WFIFO writes *WNUMDMA* 32-bit words to the McASP if and when there are at least *WNUMDMA* words in the Write FIFO. If there are not, the WFIFO waits until this condition has been satisfied. At that point, it writes *WNUMDMA* words to the McASP. (See description for WFIFOCTL.WNUMDMA in Section 19.1.45.)

If the host CPU writes to the Write FIFO, independent of a transmit DMA request, the WFIFO will accept host writes until full. After this point, excess data will be discarded.

Note that when the WFIFO is first enabled, it will immediately issue a transmit DMA request to the host. This is because it begins in an empty state, and is therefore ready to accept data.

Copyright © 2013–2016, Texas Instruments Incorporated

Transmit DMA Event Pacer

The AFIFO may be configured to delay making a transmit DMA request to the host until the Write FIFO has enough space for a specified number of words. In this situation, the number of transmit DMA requests to the host or DMA controller is reduced.

If the Write FIFO has space to accept *WNUMEVT* 32-bit words, it generates a transmit DMA request to the host and then waits for a response. Once *WNUMEVT* words have been written to the FIFO, it checks again to see if there is space for *WNUMEVT* 32-bit words. If there is space, it generates another transmit DMA request to the host, and so on. In this fashion, the Write FIFO will attempt to stay filled.

Note that if transmit DMA event pacing is desired, WFIFOCTL.WNUMEVT should be set to a non-zero integer multiple of the value in WFIFOCTL.WNUMDMA. If transmit DMA event pacing is not desired, then the value in WFIFOCTL.WNUMEVT should be set equal to the value in WFIFOCTL.WNUMDMA.

AFIFO Data Reception

When the Read FIFO is disabled, receive DMA requests pass through directly from McASP to the host/DMA controller. Whether the RFIFO is enabled or disabled, the McASP generates receive DMA requests as needed; the AFIFO is “invisible” to the McASP.

When the Read FIFO is enabled, receive DMA requests from the McASP are sent to the AFIFO, which in turn generates receive DMA requests to the host/DMA controller.

If the Read FIFO is enabled and the McASP makes a receive DMA request, the RFIFO reads *RNUMDMA* 32-bit words from the McASP, if and when the RFIFO has space for *RNUMDMA* words. If it does not, the RFIFO waits until this condition has been satisfied; at that point, it reads *RNUMDMA* words from the McASP. (See description for RFIFOCTL.RNUMDMA in Section 19.1.47.)

If the host CPU reads the Read FIFO, independent of a receive DMA request, and the RFIFO at that time contains less than *RNUMEVT* words, those words will be read correctly, emptying the FIFO.

Receive DMA Event Pacer

The AFIFO may be configured to delay making a receive DMA request to the host until the Read FIFO contains a specified number of words. In this situation, the number of receive DMA requests to the host or DMA controller is reduced.

If the Read FIFO contains at least *RNUMEVT* 32-bit words, it generates a receive DMA request to the host and then waits for a response. Once *RNUMEVT* 32-bit words have been read from the RFIFO, the RFIFO checks again to see if it contains at least another *RNUMEVT* words. If it does, it generates another receive DMA request to the host, and so on. In this fashion, the Read FIFO will attempt to stay empty.

Note that if receive DMA event pacing is desired, RFIFOCTL.RNUMEVT should be set to a non-zero integer multiple of the value in RFIFOCTL.RNUMDMA. If receive DMA event pacing is not desired, then the value in RFIFOCTL.RNUMEVT should be set equal to the value in RFIFOCTL.RNUMDMA.

Arbitration Between Transmit and Receive DMA Requests

If both the WFIFO and the RFIFO are enabled and a transmit DMA request and receive DMA request occur simultaneously, priority is given to the transmit DMA request. Once a transfer is in progress, it is allowed to complete.

If only the WFIFO is enabled and a transmit DMA request and receive DMA request occur simultaneously, priority is given to the transmit DMA request. Once a transfer is in progress, it is allowed to complete.

If only the RFIFO is enabled and a transmit DMA request and receive DMA request occur simultaneously, priority is given to the receive DMA request. Once a transfer is in progress, it is allowed to complete.

Copyright © 2013–2016, Texas Instruments Incorporated

19.0.27.4 Formatter

19.0.27.4.1 Transmit Bit Stream Data Alignment

The McASP transmitter supports serial formats of:

- Slot (or Time slot) size = 8, 12, 16, 20, 24, 28, 32 bits
- Word size ≤ Slot size
- Alignment: when more bits/slot than bits/words, then:
  - Left aligned = word shifted first, remaining bits are pad
  - Right aligned = pad bits are shifted first, word occupies the last bits in slot
- Order: order of bits shifted out:
  - MSB: most-significant bit of word is shifted out first, last bit is LSB
  - LSB: least-significant bit of word is shifted out last, last bit is MSB

Hardware support for these serial formats comes from the programmable options in the transmit bitstream format register (XFMT):

- XRVRS: bit reverse (1) or no bit reverse (0)
- XROT: rotate right by 0, 4, 8, 12, 16, 20, 24, or 28 bits
- XSSZ: transmit slot size of 8, 12, 16, 20, 24, 28, or 32 bits

XSSZ should always be programmed to match the slot size of the serial stream. The word size is not directly programmed into the McASP, but rather is used to determine the rotation needed in the XROT field.

Table 19-4 and Figure 19-29 show the XRVRS and XROT fields for each serial format and for both integer and Q31 fractional internal representations.

This discussion assumes that all slot size (SLOT in Table 19-4) and word size (WORD in Table 19-4) options are multiples of 4, since the transmit rotate right unit only supports rotation by multiples of 4. However, the bit mask/pad unit does allow for any number of significant digits. For example, a Q31 number may have 19 significant digits (word) and be transmitted in a 24-bit slot; this would be formatted as a word size of 20 bits and a slot size of 24 bits. However, it is possible to set the bit mask unit to only pass the 19 most-significant digits (program the mask value to FFFF E000h). The digits that are not significant can be set to a selected pad value, which can be any one of the significant digits, a fixed value of 0, or a fixed value of 1.

The transmit bit mask/pad unit operates on data as an initial step of the transmit format unit (see Figure 19-20), and the data is aligned in the same representation as it is written to the transmitter by the DSP (typically Q31 or integer).

Table 19-4. Transmit Bitstream Data Alignment

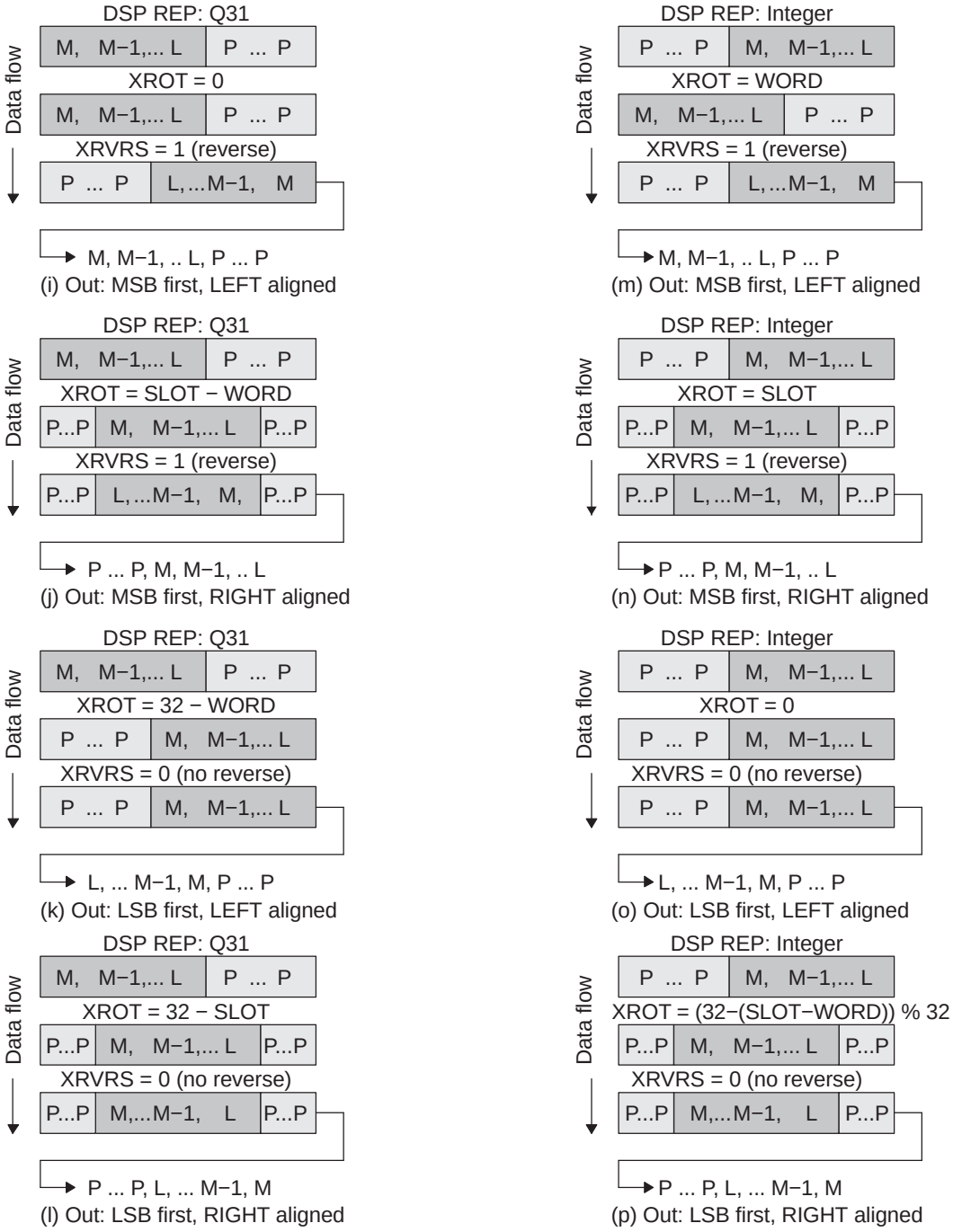
Figure 19-29	Bit Stream Order	Bit Stream Alignment	Internal Numeric Representation	XFMT Bit	
				XROT[1]	XRVRS
(a)[2]	MSB first	Left aligned	Q31 fraction	0	1
(b)	MSB first	Right aligned	Q31 fraction	SLOT - WORD	1
(c)	LSB first	Left aligned	Q31 fraction	32 - WORD	0
(d)	LSB first	Right aligned	Q31 fraction	32 - SLOT	0
(e)[2]	MSB first	Left aligned	Integer	WORD	1
(f)	MSB first	Right aligned	Integer	SLOT	1
(g)	LSB first	Left aligned	Integer	0	0
(h)	LSB first	Right aligned	Integer	(32 - (SLOT - WORD)) % 32	0

(1) WORD = Word size rounded up to the nearest multiple of 4; SLOT = slot size; % = modulo operator

(2) To transmit in I2S format, use MSB first, left aligned, and also select XDATDLY = 01 (1 bit delay)

Copyright © 2013–2016, Texas Instruments Incorporated

Figure 19-29. Data Flow Through Transmit Format Unit



Copyright © 2013–2016, Texas Instruments Incorporated

19.0.27.4.2 Receive Bit Stream Data Alignment

The McASP receiver supports serial formats of:

- Slot or time slot size = 8, 12, 16, 20, 24, 28, 32 bits
- Word size ≤ Slot size
- Alignment when more bits/slot than bits/words, then:
  - Left aligned = word shifted first, remaining bits are pad
  - Right aligned = pad bits are shifted first, word occupies the last bits in slot
- Order of bits shifted out:
  - MSB: most-significant bit of word is shifted out first, last bit is LSB
  - LSB: least-significant bit of word is shifted out last, last bit is MSB

Hardware support for these serial formats comes from the programmable options in the receive bitstream format register (RFMT):

- RRVRS: bit reverse (1) or no bit reverse (0)
- RROT: rotate right by 0, 4, 8, 12, 16, 20, 24, or 28 bits
- RSSZ: receive slot size of 8, 12, 16, 20, 24, 28, or 32 bits

RSSZ should always be programmed to match the slot size of the serial stream. The word size is not directly programmed into the McASP, but rather is used to determine the rotation needed in the RROT field.

Table 19-5 and Figure 19-30 show the RRVRS and RROT fields for each serial format and for both integer and Q31 fractional internal representations.

This discussion assumes that all slot size and word size options are multiples of 4; since the receive rotate right unit only supports rotation by multiples of 4. However, the bit mask/pad unit does allow for any number of significant digits. For example, a Q31 number may have 19 significant digits (word) and be transmitted in a 24-bit slot; this would be formatted as a word size of 20 bits and a slot size of 24 bits. However, it is possible to set the bit mask unit to only pass the 19 most-significant digits (program the mask value to FFFF E000h). The digits that are not significant can be set to a selected pad value, which can be any one of the significant digits, a fixed value of 0, or a fixed value of 1.

The receive bit mask/pad unit operates on data as the final step of the receive format unit (see Figure 19-19), and the data is aligned in the same representation as it is read from the receiver by the DSP (typically Q31 or integer).

Table 19-5. Receive Bitstream Data Alignment

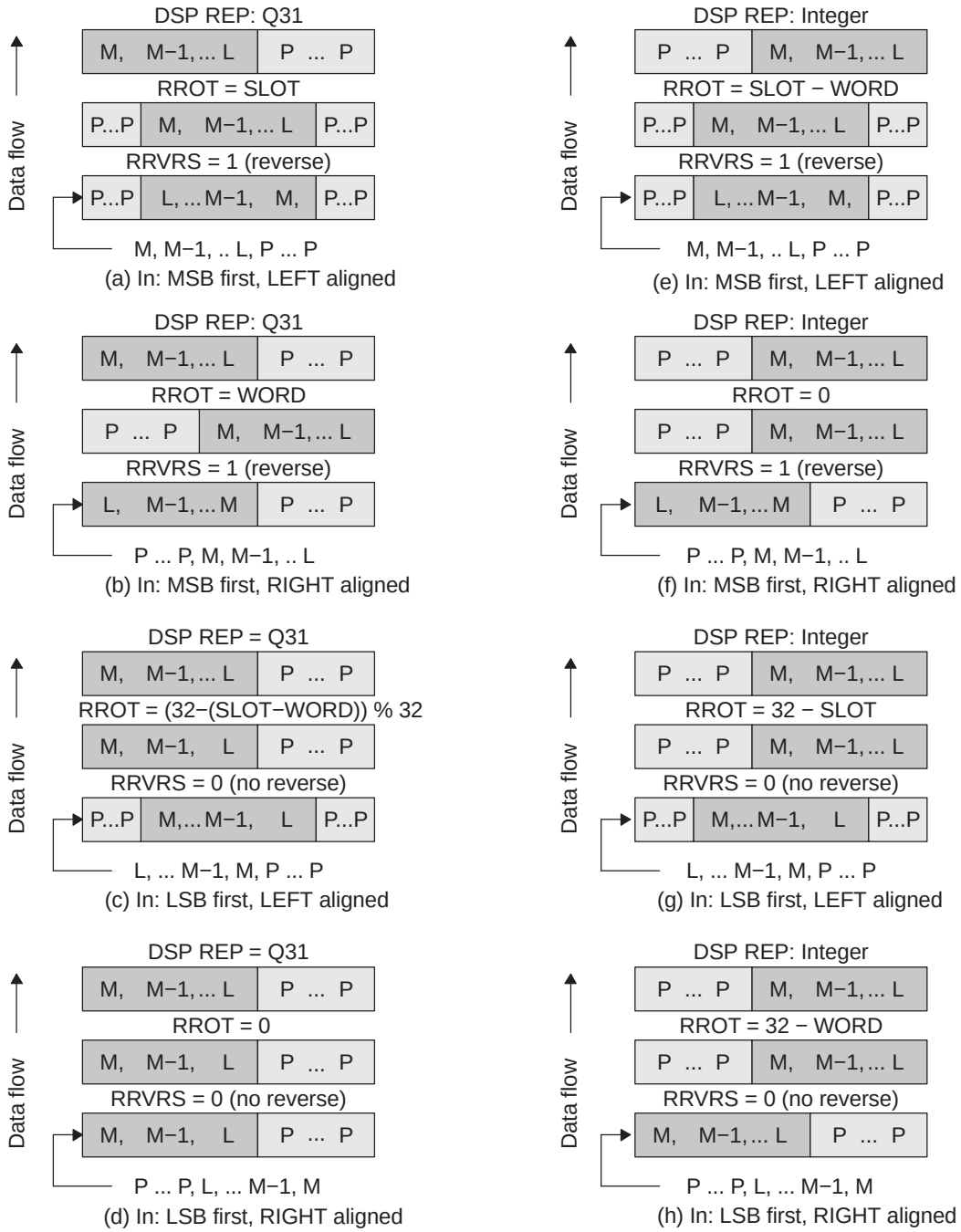
Figure 19-30	Bit Stream Order	Bit Stream Alignment	Internal Numeric Representation	RFMT Bit	
				RROT[1]	RRVRS
(a)[2]	MSB first	Left aligned	Q31 fraction	SLOT	1
(b)	MSB first	Right aligned	Q31 fraction	WORD	1
(c)	LSB first	Left aligned	Q31 fraction	(32 - (SLOT - WORD)) % 32	0
(d)	LSB first	Right aligned	Q31 fraction	0	0
(e)[2]	MSB first	Left aligned	Integer	SLOT - WORD	1
(f)	MSB first	Right aligned	Integer	0	1
(g)	LSB first	Left aligned	Integer	32 - SLOT	0
(h)	LSB first	Right aligned	Integer	32 - WORD	0

(1) WORD = Word size rounded up to the nearest multiple of 4; SLOT = slot size; % = modulo operator

(2) To transmit in I2S format, select MSB first, left aligned, and also select RDATDLY = 01 (1 bit delay)

Copyright © 2013–2016, Texas Instruments Incorporated

Figure 19-30. Data Flow Through Receive Format Unit



Copyright © 2013–2016, Texas Instruments Incorporated

19.0.27.5 Interrupts

19.0.27.5.1 Transmit Data Ready Interrupt

The transmit data ready interrupt (XDATA) is generated if XDATA is 1 in the XSTAT register and XDATA is also enabled in XINTCTL. Section 19.0.27.3.1 provides details on when XDATA is set in the XSTAT register.

A transmit start of frame interrupt (XSTAFRM) is triggered by the recognition of transmit frame sync. A transmit last slot interrupt (XLAST) is a qualified version of the data ready interrupt (XDATA). It has the same behavior as the data ready interrupt, but is further qualified by having the data requested belonging to the last slot (the slot that just ended was next-to-last TDM slot, current slot is last slot).

19.0.27.5.2 Receive Data Ready Interrupt

The receive data ready interrupt (RDATA) is generated if RDATA is 1 in the RSTAT register and RDATA is also enabled in RINTCTL. Section 19.0.27.3.2 provides details on when RDATA is set in the RSTAT register.

A receiver start of frame interrupt (RSTAFRM) is triggered by the recognition of a receiver frame sync. A receiver last slot interrupt (RLAST) is a qualified version of the data ready interrupt (RDATA). It has the same behavior as the data ready interrupt, but is further qualified by having the data in the buffer come from the last TDM time slot (the slot that just ended was last TDM slot).

19.0.27.5.3 Error Interrupts

Upon detection, the following error conditions generate interrupt flags:

- In the receive status register (RSTAT):
  - Receiver overrun (ROVRN)
  - Unexpected receive frame sync (RSYNCERR)
  - Receive clock failure (RCKFAIL)
  - Receive DMA error (RDMAERR)
- In the transmit status register (XSTAT):
  - Transmit underrun (XUNDRN)
  - Unexpected transmit frame sync (XSYNCERR)
  - Transmit clock failure (XCKFAIL)
  - Transmit DMA error (XDMAERR)

Each interrupt source also has a corresponding enable bit in the receive interrupt control register (RINTCTL) and transmit interrupt control register (XINTCTL). If the enable bit is set in RINTCTL or XINTCTL, an interrupt is requested when the interrupt flag is set in RSTAT or XSTAT. If the enable bit is not set, no interrupt request is generated. However, the interrupt flag may be polled.

19.0.27.5.4 Audio Mute (AMUTE) Function

The McASP includes an automatic audio mute function (Figure 19-31) that asserts in hardware the AMUTE device pin to a preprogrammed output state, as selected by the MUTEN bit in the audio mute control register (AMUTE). The AMUTE device pin is asserted when one of the interrupt flags is set or an external device issues an error signal on the AMUTEIN input. Typically, the AMUTEIN input is shared with a device pin.

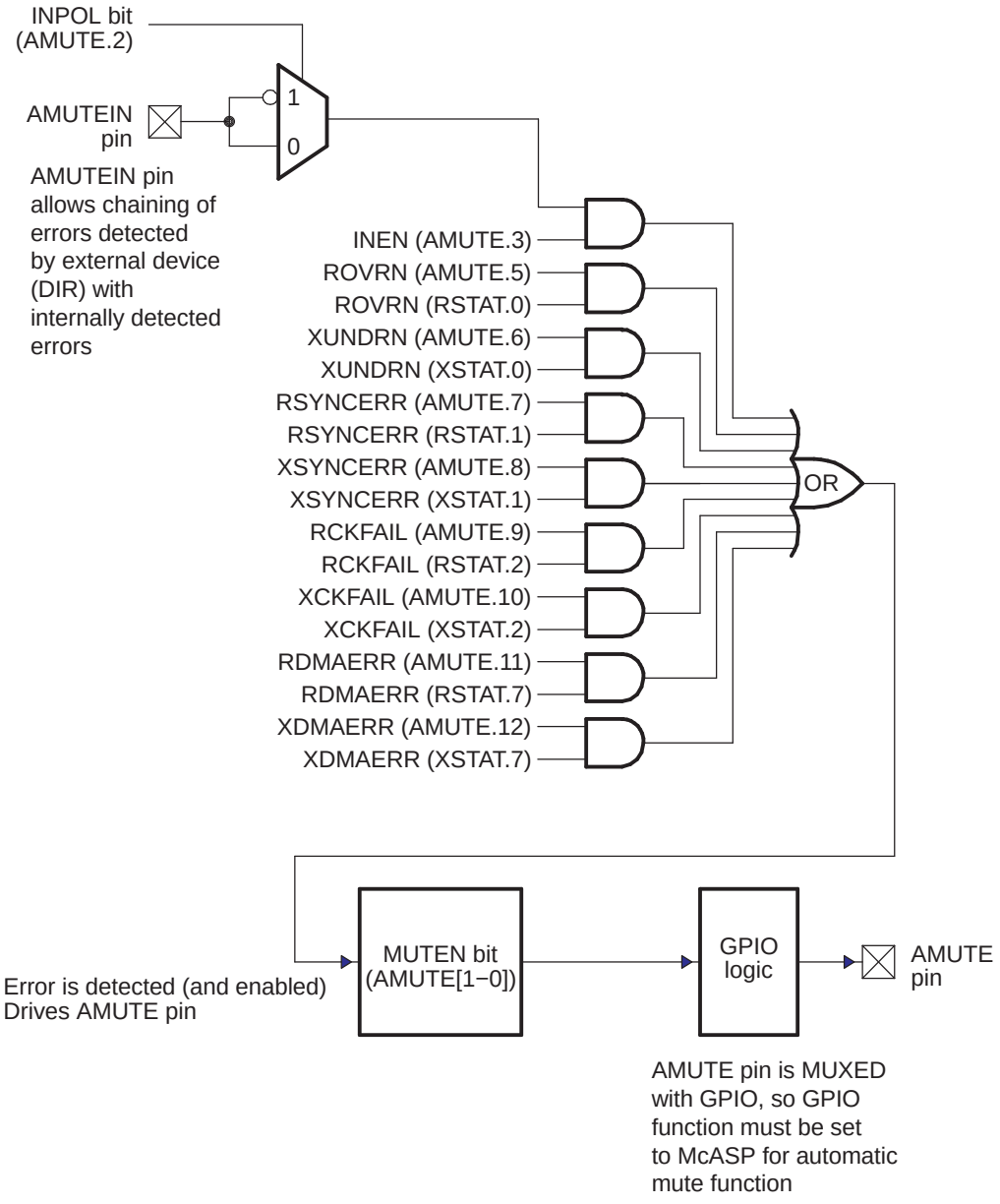
The AMUTEIN input allows the on-chip logic to consider a mute input from other devices in the system, so that all errors may be considered. The AMUTEIN input has a programmable polarity to allow it to adapt to different devices, as selected by the INPOL bit in AMUTE, and it must be enabled explicitly.

In addition to the external AMUTEIN input, the AMUTE device pin output may be asserted when one of the error interrupt flags is set and its mute function is enabled in AMUTE.

Copyright © 2013–2016, Texas Instruments Incorporated

When one or more of the errors is detected and enabled, the AMUTE device pin is driven to an active state that is selected by MUTEN in AMUTE. The active polarity of the AMUTE device pin is programmable by MUTEN (and the inactive polarity is the opposite of the active polarity). The AMUTE device pin remains driven active until software clears all the error interrupt flags that are enabled to mute, and until the AMUTEIN is inactive.

Figure 19-31. Audio Mute (AMUTE) Block Diagram



Copyright © 2013–2016, Texas Instruments Incorporated

19.0.27.5.5 Multiple Interrupts

This only applies to interrupts and not to DMA requests. The following terms are defined:

- **Active Interrupt Request:** a flag in RSTAT or XSTAT is set and the interrupt is enabled in RINTCTL or XINTCTL.
- **Outstanding Interrupt Request:** An interrupt request has been issued on one of the McASP transmit/receive interrupt ports, but that request has not yet been serviced.
- **Serviced:** The CPU writes to RSTAT or XSTAT to clear one or more of the active interrupt request flags.

The first interrupt request to become active for the transmitter with the interrupt flag set in XSTAT and the interrupt enabled in XINTCTL generates a request on the McASP transmit interrupt port AXINT.

If more than one interrupt request becomes active in the same cycle, a single interrupt request is generated on the McASP transmit interrupt port. Subsequent interrupt requests that become active while the first interrupt request is outstanding do not immediately generate a new request pulse on the McASP transmit interrupt port.

The transmit interrupt is serviced with the CPU writing to XSTAT. If any interrupt requests are active after the write, a new request is generated on the McASP transmit interrupt port.

The receiver operates in a similar way, but using RSTAT, RINTCTL, and the McASP receive interrupt port ARINT.

One outstanding interrupt request is allowed on each port, so a transmit and a receive interrupt request may both be outstanding at the same time.

19.0.27.6 Error Handling and Management

To support the design of a robust audio system, the McASP includes error-checking capability for the serial protocol, data underrun, and data overrun. In addition, the McASP includes a timer that continually measures the high-frequency master clock every 32 AHCLKX/AHCLKR clock cycles. The timer value can be read to get a measurement of the clock frequency and has a minimum and maximum range setting that can set an error flag if the master clock goes out of a specified range.

Upon the detection of any one or more errors (software selectable), or the assertion of the AMUTEIN input pin, the AMUTE output pin may be asserted to a high or low level to immediately mute the audio output. In addition, an interrupt may be generated if desired, based on any one or more of the error sources.

19.0.27.6.1 Unexpected Frame Sync Error

An unexpected frame sync occurs when:

- In burst mode, when the next active edge of the frame sync occurs early such that the current slot will not be completed by the time the next slot is scheduled to begin.
- In TDM mode, a further constraint is that the frame sync must occur exactly during the correct bit clock (not a cycle earlier or later) and only before slot 0. An unexpected frame sync occurs if this condition is not met.

When an unexpected frame sync occurs, there are two possible actions depending upon when the unexpected frame sync occurs:

1. Early: An early unexpected frame sync occurs when the McASP is in the process of completing the current frame and a new frame sync is detected (not including overlap that occurs due to a 1 or 2 bit frame sync delay). When an early unexpected frame sync occurs:
   - Error interrupt flag is set (XSYNCERR, if an unexpected transmit frame sync occurs; RSYNCERR, if an unexpected receive frame sync occurs).
   - Current frame is not resynchronized. The number of bits in the current frame is completed. The next frame sync, which occurs after the current frame is completed, will be resynchronized.

Copyright © 2013–2016, Texas Instruments Incorporated

2. Late: A late unexpected frame sync occurs when there is a gap or delay between the last bit of the previous frame and the first bit of the next frame. When a late unexpected frame sync occurs (as soon as the gap is detected):
   - Error interrupt flag is set (XSYNCERR, if an unexpected transmit frame sync occurs; RSYNCERR, if an unexpected receive frame sync occurs).
   - Resynchronization occurs upon the arrival of the next frame sync.

Late frame sync is detected the same way in both burst mode and TDM mode; however, in burst mode, late frame sync is not meaningful and its interrupt enable should not be set.

19.0.27.6.2 Buffer Underrun Error - Transmitter

A buffer underrun can only occur for serializers programmed to be transmitters. A buffer underrun occurs when the serializer is instructed by the transmit state machine to transfer data from XRBUF[n] to XRSR[n], but XRBUF[n] has not yet been written with new data since the last time the transfer occurred. When this occurs, the transmit state machine sets the XUNDRN flag.

An underrun is checked only once per time slot. The XUNDRN flag is set when an underrun condition occurs. Once set, the XUNDRN flag remains set until the DSP explicitly writes a 1 to the XUNDRN bit to clear the XUNDRN bit.

In DIT mode, a pair of BMC zeros is shifted out when an underrun occurs (four bit times at 128 × f_s). By shifting out a pair of zeros, a clock may be recovered on the receiver. To recover, reset the McASP and start again with the proper initialization.

In TDM mode, during an underrun case, a long stream of zeros are shifted out causing the DACs to mute. To recover, reset the McASP and start again with the proper initialization.

19.0.27.6.3 Buffer Overrun Error - Receiver

A buffer overrun can only occur for serializers programmed to be receivers. A buffer overrun occurs when the serializer is instructed to transfer data from XRSR[n] to XRBUF[n], but XRBUF[n] has not yet been read by either the DMA or the DSP. When this occurs, the receiver state machine sets the ROVRN flag. However, the individual serializer writes over the data in the XRBUF[n] register (destroying the previous sample) and continues shifting.

An overrun is checked only once per time slot. The ROVRN flag is set when an overrun condition occurs. It is possible that an overrun occurs on one time slot but then the DSP catches up and does not cause an overrun on the following time slots. However, once the ROVRN flag is set, it remains set until the DSP explicitly writes a 1 to the ROVRN bit to clear the ROVRN bit.

19.0.27.6.4 DMA Error - Transmitter

A transmit DMA error, as indicated by the XDMAERR flag in the XSTAT register, occurs when the DMA (or CPU) writes more words to the DMA port of the McASP than it should. For each DMA event, the DMA should write exactly as many words as there are serializers enabled as transmitters.

XDMAERR indicates that the DMA (or CPU) wrote too many words to the McASP for a given transmit DMA event. Writing too few words results in a transmit underrun error setting XUNDRN in XSTAT.

While XDMAERR occurs infrequently, an occurrence indicates a serious loss of synchronization between the McASP and the DMA or CPU. You should reinitialize both the McASP transmitter and the DMA to resynchronize them.

Copyright © 2013–2016, Texas Instruments Incorporated

19.0.27.6.5 DMA Error - Receiver

A receive DMA error, as indicated by the RDMAERR flag in the RSTAT register, occurs when the DMA (or CPU) reads more words from the DMA port of the McASP than it should. For each DMA event, the DMA should read exactly as many words as there are serializers enabled as receivers.

RDMAERR indicates that the DMA (or CPU) read too many words from the McASP for a given receive DMA event. Reading too few words results in a receiver overrun error setting ROVRN in RSTAT.

While RDMAERR occurs infrequently, an occurrence indicates a serious loss of synchronization between the McASP and the DMA or CPU. You should reinitialize both the McASP receiver and the DMA to resynchronize them.

19.0.27.6.6 Clock Failure Detection

19.0.27.6.6.1 Clock-Failure Check Startup

It is expected, initially, that the clock-failure circuits will generate an error until at least one measurement has been taken. Therefore, the clock failure interrupts, clock switch, and mute functions should not immediately be enabled, but be enabled only after a specific startup procedure. The startup procedure is:

1. For the transmit clock failure check:
   (a) Configure transmit clock failure detect logic (XMIN, XMAX, XPS) in the transmit clock check control register (XCLKCHK).
   (b) Clear transmit clock failure flag (XCKFAIL) in the transmit status register (XSTAT).
   (c) Wait until first measurement is taken (> 32 AHCLKX clock periods).
   (d) Verify no clock failure is detected.
   (e) Repeat steps b–d until clock is running and is no longer issuing clock failure errors.
   (f) After the transmit clock is measured and falls within the acceptable range, the following may be enabled:
      (i) transmit clock failure interrupt enable bit (XCKFAIL) in the transmitter interrupt control register (XINTCTL)
      (ii) transmit clock failure detect autoswitch enable bit (XCKFAILSW) in the transmit clock check control register (XCLKCHK)
      (iii) mute option (XCKFAIL) in the mute control register (AMUTE)
2. For the receive clock failure check:
   (a) Configure receive clock failure detect logic (RMIN, RMAX, RPS) in the receive clock check control register (RCLKCHK).
   (b) Clear receive clock failure flag (RCKFAIL) in the receive status register (RSTAT).
   (c) Wait until first measurement is taken (> 32 AHCLKR clock periods).
   (d) Verify no clock failure is detected.
   (e) Repeat steps b–d until clock is running and is no longer issuing clock failure errors.
   (f) After the receive clock is measured and falls within the acceptable range, the following may be enabled:
      (i) receive clock failure interrupt enable bit (RCKFAIL) in the receiver interrupt control register (RINTCTL)
      (ii) mute option (RCKFAIL) in the mute control register (AMUTE)

Copyright © 2013–2016, Texas Instruments Incorporated

19.0.27.6.6.2 Transmit Clock Failure Check and Recovery

The transmit clock failure check circuit (Figure 19-32) works off both the internal McASP system clock and the external high-frequency serial clock (AHCLKX). It continually counts the number of system clocks for every 32 high rate serial clock (AHCLKX) periods, and stores the count in XCNT of the transmit clock check control register (XCLKCHK) every 32 high rate serial clock cycles.

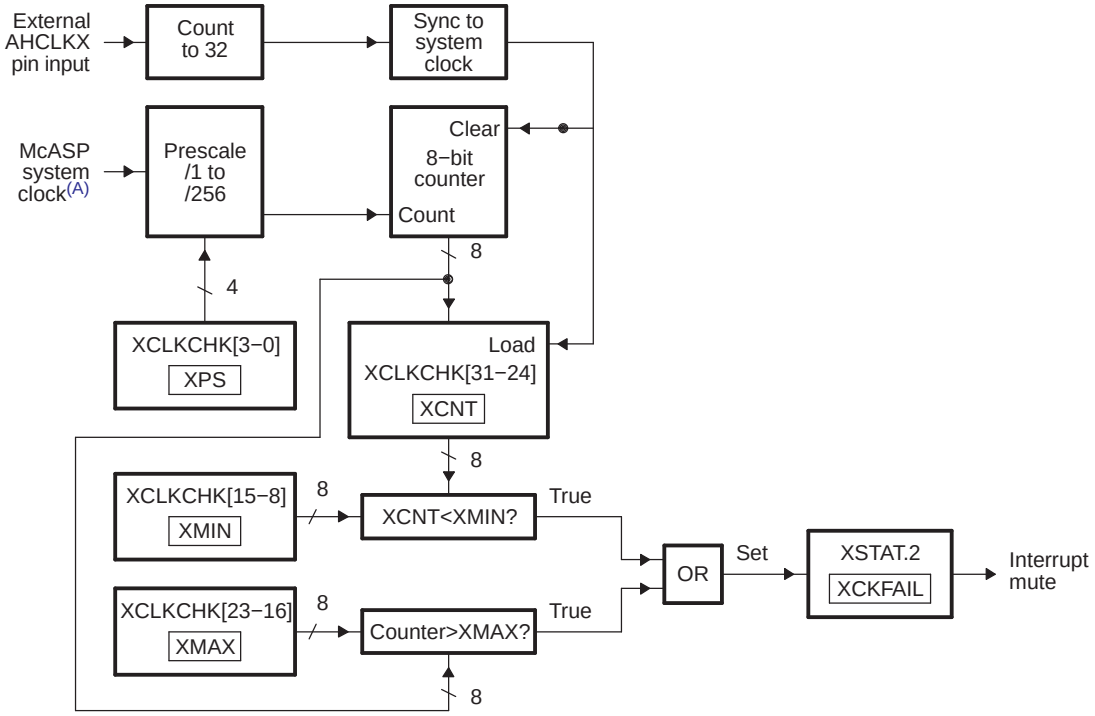
The logic compares the count against a user-defined minimum allowable boundary (XMIN), and automatically flags an interrupt (XCKFAIL in XSTST) when an out-of-range condition occurs. An out-of-range minimum condition occurs when the count is smaller than XMIN. The logic continually compares the current count (from the running system clock counter) against the maximum allowable boundary (XMAX). This is in case the external clock completely stops, so that the counter value is not copied to XCNT. An out-of-range maximum condition occurs when the count is greater than XMAX. Note that the XMIN and XMAX fields are 8-bit unsigned values, and the comparison is performed using unsigned arithmetic.

An out-of-range count may indicate either that an unstable clock was detected, or that the audio source has changed and a new sample rate is being used.

In order for the transmit clock failure check circuit to operate correctly, the high-frequency serial clock divider must be taken out of reset regardless if AHCLKX is internally generated or externally sourced.

If a clock failure is detected, the transmit clock failure flag (XCKFAIL) in XSTAT is set. This causes an interrupt, if the transmit clock failure interrupt enable bit (XCKFAIL) in XINTCTL is set.

Figure 19-32. Transmit Clock Failure Detection Circuit Block Diagram



A This is not the same as AUXCLK. The DSP uses SYSCLK2 as the McASP system clock.

Copyright © 2013–2016, Texas Instruments Incorporated

19.0.27.6.6.3 Receive Clock Failure Check and Recovery

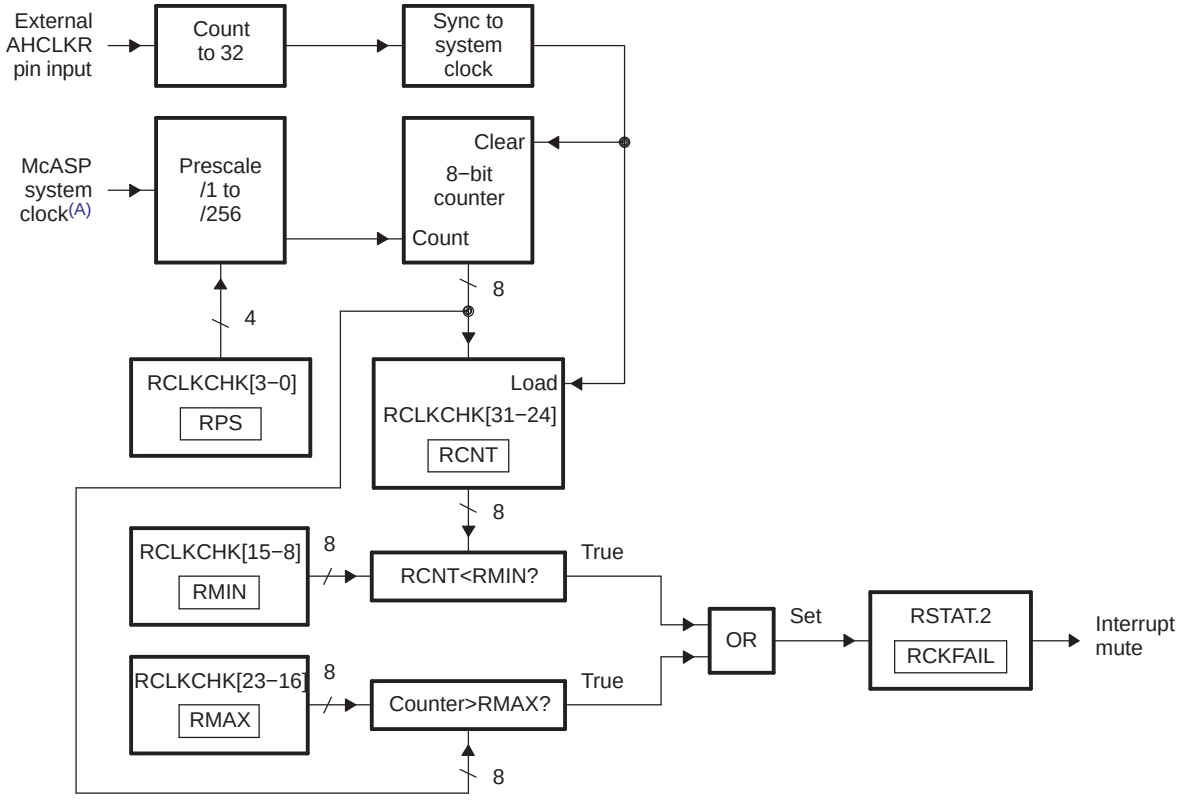
The receive clock failure check circuit (Figure 19-33) works off both the internal McASP system clock and the external high-frequency serial clock (AHCLKR). It continually counts the number of system clocks for every 32 high rate serial clock (AHCLKR) periods, and stores the count in RCNT of the receive clock check control register (RCLKCHK) every 32 high rate serial clock cycles.

The logic compares the count against a user-defined minimum allowable boundary (RMIN) and automatically flags an interrupt (RCKFAIL in RSTAT) when an out-of-range condition occurs. An out-of-range minimum condition occurs when the count is smaller than RMIN. The logic continually compares the current count (from the running system clock counter) against the maximum allowable boundary (RMAX). This is in case the external clock completely stops, so that the counter value is not copied to RCNT. An out-of-range maximum condition occurs when the count is greater than RMAX. Note that the RMIN and RMAX fields are 8-bit unsigned values, and the comparison is performed using unsigned arithmetic.

An out-of-range count may indicate either that an unstable clock was detected or that the audio source has changed and a new sample rate is being used.

In order for the receive clock failure check circuit to operate correctly, the high-frequency serial clock divider must be taken out of reset regardless if AHCLKR is internally generated or externally sourced.

Figure 19-33. Receive Clock Failure Detection Circuit Block Diagram



A This is not the same as AUXCLK. The DSP uses SYSCLK2 as the McASP system clock source.

Copyright © 2013–2016, Texas Instruments Incorporated

19.0.27.7 Loopback Modes

The McASP features a digital loopback mode (DLB) that allows testing of the McASP code in TDM mode with a single DSP device. In loopback mode, output of the transmit serializers is connected internally to the input of the receive serializers. Therefore, you can check the receive data against the transmit data to ensure that the McASP settings are correct. Digital loopback mode applies to TDM mode only (2 to 32 slots in a frame). It does not apply to DIT mode (XMOD = 180h) or burst mode (XMOD = 0).

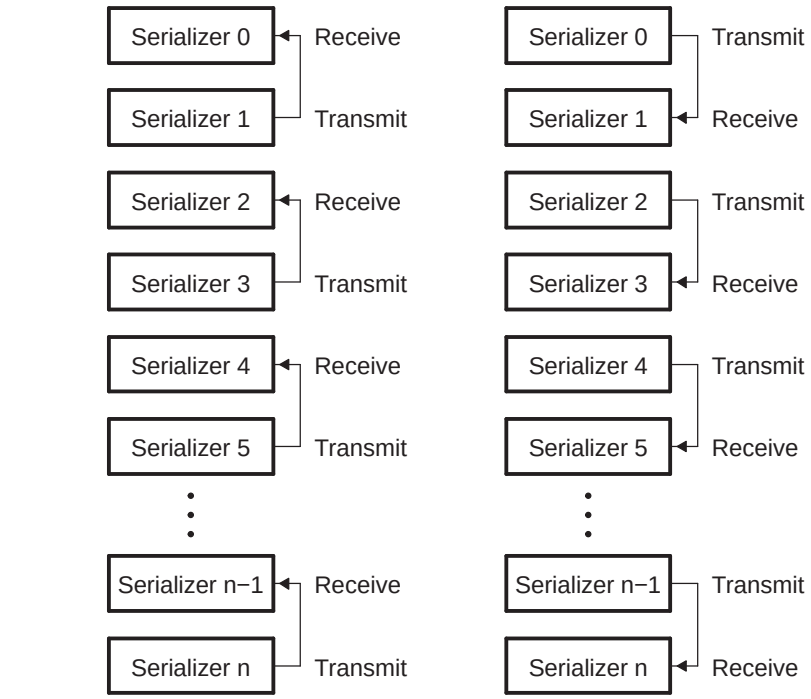
Figure 19-34 shows the basic logical connection of the serializers in loopback mode. Two types of loopback connections are possible, selected by the ORD bit in the digital loopback control register (DLBCTL) as follows:

- ORD = 0: Outputs of odd serializers are connected to inputs of even serializers. If this mode is selected, you should configure odd serializers to be transmitters and even serializers to be receivers.
- ORD = 1: Outputs of even serializers are connected to inputs of odd serializers. If this mode is selected, you should configure even serializers to be transmitters and odd serializers to be receivers.

Data can be externally visible at the I/O pin of the transmit serializer if the pin is configured as a McASP output pin by setting the corresponding PFUNC bit to 0 and PDIR bit to 1.

In loopback mode, the transmit clock and frame sync are used by both the transmit and receive sections of the McASP. The transmit and receive sections operate synchronously. This is achieved by setting the MODE bit of the DLBCTL register to 01b and the ASYNC bit of the ACLKXCTL register to 0.

Figure 19-34. Serializers in Loopback Mode



(a) DLBEN = 1 (loopback enabled) and ORD = 0 (even receive, odd transmit)

(b) DLBEN = 1 (loopback enabled) and ORD = 1 (odd receive, even transmit)

Copyright © 2013–2016, Texas Instruments Incorporated

19.0.27.7.1 Loopback Mode Configurations

This is a summary of the settings required for digital loopback mode for TDM format:

- The DLBEN bit in DLBCTL must be set to 1 to enable loopback mode.
- The MODE bits in DLBCTL must be set to 01b for both the transmit and receive sections to use the transmit clock and frame sync generator.
- The ORD bit in DLBCTL must be programmed appropriately to select odd or even serializers to be transmitters or receivers. The corresponding serializers must be configured accordingly.
- The ASYNC bit in ACLKXCTL must be cleared to 0 to ensure synchronous transmit and receive operations.
- RMOD field in AFSRCTL and XMOD field in AFSXCTL must be set to 2h to 20h to indicate TDM mode. Loopback mode does not apply to DIT or burst mode.

19.0.28 Reset Considerations

The McASP has two reset sources: software reset and hardware reset.

19.0.28.1 Software Reset Considerations

The transmitter and receiver portions of the McASP may be put in reset through the global control register (GBLCTL). Note that a valid serial clock must be supplied to the desired portion of the McASP (transmit and/or receive) in order to assert the software reset bits in GBLCTL. See Section 19.0.27.1.2 for details on how to ensure reset has occurred.

The entire McASP module may also be reset through the Power and Sleep Controller (PSC). Note that from the McASP perspective, this reset appears as a hardware reset to the entire module.

19.0.28.2 Hardware Reset Considerations

When the McASP is reset due to device reset, the entire serial port (including the transmitter and receiver state machines, and other registers) is reset.

19.0.29 EDMA Event Support

The McASP-related EDMA events are shown in Table 19-6.

Table 19-6. EDMA Events - McASP

Channel	Event Name	Event Description
0	AREVT0	McASP0 Receive Event
1	AXEVT0	McASP0 Transmit Event
2	AREVT1	McASP1 Receive Event
3	AXEVT1	McASP1 Transmit Event
4	AREVT2	McASP2 Receive Event
5	AXEVT2	McASP2 Transmit Event

19.0.30 Power Management

The McASP can be placed in reduced power modes to conserve power during periods of low activity. The power management of the peripheral is controlled by the processor Power and Sleep Controller (PSC). The PSC acts as a master controller for power management for all of the peripherals on the device. For information on power management procedures using the PSC, see the *Power and Sleep Controller (PSC)* chapter.

Copyright © 2013–2016, Texas Instruments Incorporated

19.1 Registers

Control registers for the McASP are summarized in Table 19-7. The control registers are accessed through the peripheral configuration port. The receive buffer registers (RBUF) and transmit buffer registers (XBUF) can also be accessed through the DMA port, as listed in Table 19-8. See your device-specific data manual for the memory address of these registers.

Control registers for the McASP Audio FIFO (AFIFO) are summarized in Table 19-9. Note that the AFIFO Write FIFO (WFIFO) and Read FIFO (RFIFO) have independent control and status registers. The AFIFO control registers are accessed through the peripheral configuration port. See your device-specific data manual for the memory address of these registers.

Table 19-7. McASP Registers Accessed by CPU/EDMA Through Peripheral Configuration Port

Offset	Acronym	Register Description	Section
0h	REV	Revision identification register	Section 19.1.2
10h	PFUNC	Pin function register	Section 19.1.3
14h	PDIR	Pin direction register	Section 19.1.4
18h	PDOUT	Pin data output register	Section 19.1.5
1Ch	PDIN	Read returns: Pin data input register	Section 19.1.6
1Ch	PDSET	Writes affect: Pin data set register (alternate write address: PDOUT)	Section 19.1.7
20h	PDCLR	Pin data clear register (alternate write address: PDOUT)	Section 19.1.8
44h	GBLCTL	Global control register	Section 19.1.9
48h	AMUTE	Audio mute control register	Section 19.1.10
4Ch	DLBCTL	Digital loopback control register	Section 19.1.11
50h	DITCTL	DIT mode control register	Section 19.1.12
60h	RGBLCTL	Receiver global control register: Alias of GBLCTL, only receive bits are affected - allows receiver to be reset independently from transmitter	Section 19.1.13
64h	RMASK	Receive format unit bit mask register	Section 19.1.14
68h	RFMT	Receive bit stream format register	Section 19.1.15
6Ch	AFSRCTL	Receive frame sync control register	Section 19.1.16
70h	ACLKRCTL	Receive clock control register	Section 19.1.17
74h	AHCLKRCTL	Receive high-frequency clock control register	Section 19.1.18
78h	RTDM	Receive TDM time slot 0-31 register	Section 19.1.19
7Ch	RINTCTL	Receiver interrupt control register	Section 19.1.20
80h	RSTAT	Receiver status register	Section 19.1.21
84h	RSLOT	Current receive TDM time slot register	Section 19.1.22
88h	RCLKCHK	Receive clock check control register	Section 19.1.23
8Ch	REVTCTL	Receiver DMA event control register	Section 19.1.24
A0h	XGBLCTL	Transmitter global control register. Alias of GBLCTL, only transmit bits are affected - allows transmitter to be reset independently from receiver	Section 19.1.25
A4h	XMASK	Transmit format unit bit mask register	Section 19.1.26
A8h	XFMT	Transmit bit stream format register	Section 19.1.27
ACh	AFSXCTL	Transmit frame sync control register	Section 19.1.28
B0h	ACLKXCTL	Transmit clock control register	Section 19.1.29
B4h	AHCLKXCTL	Transmit high-frequency clock control register	Section 19.1.30
B8h	XTDM	Transmit TDM time slot 0-31 register	Section 19.1.31
BCh	XINTCTL	Transmitter interrupt control register	Section 19.1.32
C0h	XSTAT	Transmitter status register	Section 19.1.33
C4h	XSLOT	Current transmit TDM time slot register	Section 19.1.34
C8h	XCLKCHK	Transmit clock check control register	Section 19.1.35
CCh	XEVTCTL	Transmitter DMA event control register	Section 19.1.36

Copyright © 2013–2016, Texas Instruments Incorporated

Table 19-7. McASP Registers Accessed by CPU/EDMA Through Peripheral Configuration Port (continued)

Offset	Acronym	Register Description	Section
100h	DITCSRA0	Left (even TDM time slot) channel status register (DIT mode) 0	Section 19.1.38
104h	DITCSRA1	Left (even TDM time slot) channel status register (DIT mode) 1	Section 19.1.38
108h	DITCSRA2	Left (even TDM time slot) channel status register (DIT mode) 2	Section 19.1.38
10Ch	DITCSRA3	Left (even TDM time slot) channel status register (DIT mode) 3	Section 19.1.38
110h	DITCSRA4	Left (even TDM time slot) channel status register (DIT mode) 4	Section 19.1.38
114h	DITCSRA5	Left (even TDM time slot) channel status register (DIT mode) 5	Section 19.1.38
118h	DITCSRB0	Right (odd TDM time slot) channel status register (DIT mode) 0	Section 19.1.39
11Ch	DITCSRB1	Right (odd TDM time slot) channel status register (DIT mode) 1	Section 19.1.39
120h	DITCSRB2	Right (odd TDM time slot) channel status register (DIT mode) 2	Section 19.1.39
124h	DITCSRB3	Right (odd TDM time slot) channel status register (DIT mode) 3	Section 19.1.39
128h	DITCSRB4	Right (odd TDM time slot) channel status register (DIT mode) 4	Section 19.1.39
12Ch	DITCSRB5	Right (odd TDM time slot) channel status register (DIT mode) 5	Section 19.1.39
130h	DITUDRA0	Left (even TDM time slot) channel user data register (DIT mode) 0	Section 19.1.40
134h	DITUDRA1	Left (even TDM time slot) channel user data register (DIT mode) 1	Section 19.1.40
138h	DITUDRA2	Left (even TDM time slot) channel user data register (DIT mode) 2	Section 19.1.40
13Ch	DITUDRA3	Left (even TDM time slot) channel user data register (DIT mode) 3	Section 19.1.40
140h	DITUDRA4	Left (even TDM time slot) channel user data register (DIT mode) 4	Section 19.1.40
144h	DITUDRA5	Left (even TDM time slot) channel user data register (DIT mode) 5	Section 19.1.40
148h	DITUDRB0	Right (odd TDM time slot) channel user data register (DIT mode) 0	Section 19.1.41
14Ch	DITUDRB1	Right (odd TDM time slot) channel user data register (DIT mode) 1	Section 19.1.41
150h	DITUDRB2	Right (odd TDM time slot) channel user data register (DIT mode) 2	Section 19.1.41
154h	DITUDRB3	Right (odd TDM time slot) channel user data register (DIT mode) 3	Section 19.1.41
158h	DITUDRB4	Right (odd TDM time slot) channel user data register (DIT mode) 4	Section 19.1.41
15Ch	DITUDRB5	Right (odd TDM time slot) channel user data register (DIT mode) 5	Section 19.1.41
180h	SRCTL0	Serializer control register 0	Section 19.1.37
184h	SRCTL1	Serializer control register 1	Section 19.1.37
188h	SRCTL2	Serializer control register 2	Section 19.1.37
18Ch	SRCTL3	Serializer control register 3	Section 19.1.37
190h	SRCTL4	Serializer control register 4	Section 19.1.37
194h	SRCTL5	Serializer control register 5	Section 19.1.37
198h	SRCTL6	Serializer control register 6	Section 19.1.37
19Ch	SRCTL7	Serializer control register 7	Section 19.1.37
1A0h	SRCTL8	Serializer control register 8	Section 19.1.37
1A4h	SRCTL9	Serializer control register 9	Section 19.1.37
1A8h	SRCTL10	Serializer control register 10	Section 19.1.37
1ACh	SRCTL11	Serializer control register 11	Section 19.1.37
1B0h	SRCTL12	Serializer control register 12	Section 19.1.37
1B4h	SRCTL13	Serializer control register 13	Section 19.1.37
1B8h	SRCTL14	Serializer control register 14	Section 19.1.37
1BCh	SRCTL15	Serializer control register 15	Section 19.1.37

Copyright © 2013–2016, Texas Instruments Incorporated

Table 19-7. McASP Registers Accessed by CPU/EDMA Through Peripheral Configuration Port (continued)

Offset	Acronym	Register Description	Section
200h	XBUF0[1]	Transmit buffer register for serializer 0	Section 19.1.42
204h	XBUF1[1]	Transmit buffer register for serializer 1	Section 19.1.42
208h	XBUF2[1]	Transmit buffer register for serializer 2	Section 19.1.42
20Ch	XBUF3[1]	Transmit buffer register for serializer 3	Section 19.1.42
210h	XBUF4[1]	Transmit buffer register for serializer 4	Section 19.1.42
214h	XBUF5[1]	Transmit buffer register for serializer 5	Section 19.1.42
218h	XBUF6[1]	Transmit buffer register for serializer 6	Section 19.1.42
21Ch	XBUF7[1]	Transmit buffer register for serializer 7	Section 19.1.42
220h	XBUF8[1]	Transmit buffer register for serializer 8	Section 19.1.42
224h	XBUF9[1]	Transmit buffer register for serializer 9	Section 19.1.42
228h	XBUF10[1]	Transmit buffer register for serializer 10	Section 19.1.42
22Ch	XBUF11[1]	Transmit buffer register for serializer 11	Section 19.1.42
230h	XBUF12[1]	Transmit buffer register for serializer 12	Section 19.1.42
234h	XBUF13[1]	Transmit buffer register for serializer 13	Section 19.1.42
238h	XBUF14[1]	Transmit buffer register for serializer 14	Section 19.1.42
23Ch	XBUF15[1]	Transmit buffer register for serializer 15	Section 19.1.42
280h	RBUF0[2]	Receive buffer register for serializer 0	Section 19.1.43
284h	RBUF1[2]	Receive buffer register for serializer 1	Section 19.1.43
288h	RBUF2[2]	Receive buffer register for serializer 2	Section 19.1.43
28Ch	RBUF3[2]	Receive buffer register for serializer 3	Section 19.1.43
290h	RBUF4[2]	Receive buffer register for serializer 4	Section 19.1.43
294h	RBUF5[2]	Receive buffer register for serializer 5	Section 19.1.43
298h	RBUF6[2]	Receive buffer register for serializer 6	Section 19.1.43
29Ch	RBUF7[2]	Receive buffer register for serializer 7	Section 19.1.43
2A0h	RBUF8[2]	Receive buffer register for serializer 8	Section 19.1.43
2A4h	RBUF9[2]	Receive buffer register for serializer 9	Section 19.1.43
2A8h	RBUF10[2]	Receive buffer register for serializer 10	Section 19.1.43
2ACh	RBUF11[2]	Receive buffer register for serializer 11	Section 19.1.43
2B0h	RBUF12[2]	Receive buffer register for serializer 12	Section 19.1.43
2B4h	RBUF13[2]	Receive buffer register for serializer 13	Section 19.1.43
2B8h	RBUF14[2]	Receive buffer register for serializer 14	Section 19.1.43
2BCh	RBUF15[2]	Receive buffer register for serializer 15	Section 19.1.43

(1) Writes to XRBUF[n] by way of XBUF*n* by the CPU/EDMA can only occur through the peripheral configuration port when XBUSEL = 1 in XFMT.

(2) Reads from XRBUF[n] by way of RBUF*n* by the CPU/EDMA can only occur through the peripheral configuration port when RBUSEL = 1 in RFMT.

Copyright © 2013–2016, Texas Instruments Incorporated

Table 19-8. McASP Registers Accessed by CPU/EDMA Through DMA Port

Offset[1]	Access	Acronym	Register Description
2000h	Read Accesses	RBUF	Receive buffer DMA port address. Cycles through receive serializers, skipping over transmit serializers and inactive serializers. Starts at the lowest serializer at the beginning of each time slot. Reads from XRBUF[n] by way of RBUF by the CPU/EDMA can only occur through the DMA port when RBUSEL = 0 in RFMT.
2000h	Write Accesses	XBUF	Transmit buffer DMA port address. Cycles through transmit serializers, skipping over receive and inactive serializers. Starts at the lowest serializer at the beginning of each time slot. Writes to XRBUF[n] by way of XBUF by the CPU/EDMA can only occur through the DMA port when XBUSEL = 0 in XFMT.

(1) RBUF and XBUF are at the same address location. Reads access RBUF and writes access XBUF.

Table 19-9. McASP AFIFO Registers Accessed Through Peripheral Configuration Port[1]

Offset	Acronym	Register Description	Section
1000h	AFIFOREV	AFIFO revision identification register	Section 19.1.44
1010h	WFIFOCTL	Write FIFO control register	Section 19.1.45
1014h	WFIFOSTS	Write FIFO status register	Section 19.1.46
1018h	RFIFOCTL	Read FIFO control register	Section 19.1.47
101Ch	RFIFOSTS	Read FIFO status register	Section 19.1.48

(1) The AFIFO cannot be used with the peripheral configuration port. Only the DMA port has access to the AFIFO.

19.1.1 Register Bit Restrictions

Some bit fields (see Bits With Restrictions on When They May be Changed) have restrictions on when they may be changed. These restrictions take the form of certain registers that must be asserted in GBLCTL. Once these registers have been asserted, the user may then, and only then, change the desired bit field.

Bits With Restrictions on When They May be Changed

To Change Register:	To Change Bit Field:	... these registers must be asserted in GBLCTL									
		HCLKRRST	RGRST	RSRCLR	RSMRST	RFRST	HCLKXRST	XGRST	XSRCLR	XSMRST	XFRST
DITCTL	DITEN									x	x
XFMT	XSSZ									x	
XFMT	XDATDLY				x					x	
RFMT	RSSZ				x						
RFMT	RDATDLY				x						
AFSXCTL	FSXP									x	x
AFSXCTL	FSXM									x	x
AFSXCTL	FXWID									x	x
AFSXCTL	XMOD									x	x
AFSRCTL	FSRP				x	x					
AFSRCTL	FSRM				x	x					
AFSRCTL	FRWID				x	x					
AFSRCTL	RMOD				x	x					
ACLKXCTL	CLKXDIV							x	x	x	x
ACLKXCTL	CLKXM								x	x	x
ACLKXCTL	ASYNC				x	x					
ACLKXCTL	CLKXP								x	x	x
ACLKRCTL	CLKRDIV		x	x	x	x					
ACLKRCTL	CLKRM			x	x	x					
ACLKRCTL	CLKRP			x	x	x					

Copyright © 2013–2016, Texas Instruments Incorporated

Bits With Restrictions on When They May be Changed (continued)

To Change Register:	To Change Bit Field:	... these registers must be asserted in GBLCTL									
		HCLKRRST	RGRST	RSRCLR	RSMRST	RFRST	HCLKXRST	XGRST	XSRCLR	XSMRST	XFRST
AHCLKXCTL	HCLKXDIV						x	x	x	x	x
AHCLKXCTL	HCLKXP						x	x	x	x	x
AHCLKXCTL	HCLKXM						x	x	x	x	x
AHCLKRCTL	HCLKRDIV	x	x	x	x	x					
AHCLKRCTL	HCLKRP	x	x	x	x	x					
AHCLKRCTL	HCLKRM	x	x	x	x	x					
DLBCTL	DLBEN			x	x	x			x	x	x
DLBCTL	ORD			x	x	x			x	x	x
DLBCTL	MODE			x	x	x			x	x	x

19.1.2 Revision Identification Register (REV)

The revision identification register (REV) contains revision data for the peripheral. The REV is shown in Figure 19-35 and described in Table 19-10.

Figure 19-35. Revision Identification Register (REV)

31	0
REV	
R-4430 0A02h	

LEGEND: R = Read only; -*n* = value after reset

Table 19-10. Revision Identification Register (REV) Field Descriptions

Bit	Field	Value	Description
31-0	REV	4430 0A02h	Identifies revision of peripheral.

Copyright © 2013–2016, Texas Instruments Incorporated

19.1.3 Pin Function Register (PFUNC)

The pin function register (PFUNC) specifies the function of AXR[n], ACLKX, AHCLKX, AFSX, ACLKR, AHCLKR, and AFSR pins as either a McASP pin or a general-purpose input/output (GPIO) pin. The PFUNC is shown in Figure 19-36 and described in Table 19-11.

CAUTION

Writing to Reserved Bits

Writing a value other than 0 to reserved bits in this register may cause improper device operation. This includes bits that are not implemented on a particular DSP.

Figure 19-36. Pin Function Register (PFUNC)

31	30	29	28	27	26	25	24
AFSR	AHCLKR	ACLKR	AFSX	AHCLKX	ACLKX	AMUTE	Reserved (A)
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R-0

23	16
Reserved (A)	
R-0	

15	14	13	12	11	10	9	8
AXR15	AXR14	AXR13	AXR12	AXR11	AXR10	AXR9	AXR8
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0

7	6	5	4	3	2	1	0
AXR7	AXR6	AXR5	AXR4	AXR3	AXR2	AXR1	AXR0
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

A If writing to this field, always write the default value for future device compatibility.

Copyright © 2013–2016, Texas Instruments Incorporated

Table 19-11. Pin Function Register (PFUNC) Field Descriptions

Bit	Field	Value	Description
31	AFSR		Determines if AFSR pin functions as McASP or GPIO.
		0	Pin functions as McASP pin.
		1	Pin functions as GPIO pin.
30	AHCLKR		Determines if AHCLKR pin functions as McASP or GPIO.
		0	Pin functions as McASP pin.
		1	Pin functions as GPIO pin.
29	ACLKR		Determines if ACLKR pin functions as McASP or GPIO.
		0	Pin functions as McASP pin.
		1	Pin functions as GPIO pin.
28	AFSX		Determines if AFSX pin functions as McASP or GPIO.
		0	Pin functions as McASP pin.
		1	Pin functions as GPIO pin.
27	AHCLKX		Determines if AHCLKX pin functions as McASP or GPIO.
		0	Pin functions as McASP pin.
		1	Pin functions as GPIO pin.
26	ACLKX		Determines if ACLKX pin functions as McASP or GPIO.
		0	Pin functions as McASP pin.
		1	Pin functions as GPIO pin.
25	AMUTE		Determines if AMUTE pin functions as McASP or GPIO.
		0	Pin functions as McASP pin.
		1	Pin functions as GPIO pin.
24-16	Reserved	0	Reserved. The reserved bit location always returns the default value. A value written to this field has no effect. If writing to this field, always write the default value for future device compatibility.
15-0	AXR[15-0]		Determines if AXR[n] pin functions as McASP or GPIO.
		0	Pin functions as McASP pin.
		1	Pin functions as GPIO pin.

Copyright © 2013–2016, Texas Instruments Incorporated

19.1.4 Pin Direction Register (PDIR)

The pin direction register (PDIR) specifies the direction of AXR[n], ACLKX, AHCLKX, AFSX, ACLKR, AHCLKR, and AFSR pins as either an input or an output pin. The PDIR is shown in Figure 19-37 and described in Table 19-12.

Regardless of the pin function register (PFUNC) setting, each PDIR bit must be set to 1 for the specified pin to be enabled as an output and each PDIR bit must be cleared to 0 for the specified pin to be an input.

For example, if the McASP is configured to use an internally-generated bit clock and the clock is to be driven out to the system, the PFUNC bit must be cleared to 0 (McASP function) and the PDIR bit must be set to 1 (an output).

When AXR[n] is configured to transmit, the PFUNC bit must be cleared to 0 (McASP function) and the PDIR bit must be set to 1 (an output). Similarly, when AXR[n] is configured to receive, the PFUNC bit must be cleared to 0 (McASP function) and the PDIR bit must be cleared to 0 (an input).

CAUTION

Writing to Reserved Bits

Writing a value other than 0 to reserved bits in this register may cause improper device operation. This includes bits that are not implemented on a particular DSP.

Figure 19-37. Pin Direction Register (PDIR)

31	30	29	28	27	26	25	24
AFSR	AHCLKR	ACLKR	AFSX	AHCLKX	ACLKX	AMUTE	Reserved (A)
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R-0

23	16
Reserved (A)	
R-0	

15	14	13	12	11	10	9	8
AXR15	AXR14	AXR13	AXR12	AXR11	AXR10	AXR9	AXR8
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0

7	6	5	4	3	2	1	0
AXR7	AXR6	AXR5	AXR4	AXR3	AXR2	AXR1	AXR0
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

A If writing to this field, always write the default value for future device compatibility.

Copyright © 2013–2016, Texas Instruments Incorporated

Table 19-12. Pin Direction Register (PDIR) Field Descriptions

Bit	Field	Value	Description
31	AFSR		Determines if AFSR pin functions as an input or output.
		0	Pin functions as input.
		1	Pin functions as output.
30	AHCLKR		Determines if AHCLKR pin functions as an input or output.
		0	Pin functions as input.
		1	Pin functions as output.
29	ACLKR		Determines if ACLKR pin functions as an input or output.
		0	Pin functions as input.
		1	Pin functions as output.
28	AFSX		Determines if AFSX pin functions as an input or output.
		0	Pin functions as input.
		1	Pin functions as output.
27	AHCLKX		Determines if AHCLKX pin functions as an input or output.
		0	Pin functions as input.
		1	Pin functions as output.
26	ACLKX		Determines if ACLKX pin functions as an input or output.
		0	Pin functions as input.
		1	Pin functions as output.
25	AMUTE		Determines if AMUTE pin functions as an input or output.
		0	Pin functions as input.
		1	Pin functions as output.
24-16	Reserved	0	Reserved. The reserved bit location always returns the default value. A value written to this field has no effect. If writing to this field, always write the default value for future device compatibility.
15-0	AXR[15-0]		Determines if AXR[n] pin functions as an input or output.
		0	Pin functions as input.
		1	Pin functions as output.

Copyright © 2013–2016, Texas Instruments Incorporated

19.1.5 Pin Data Output Register (PDOUT)

The pin data output register (PDOUT) holds a value for data out at all times, and may be read back at all times. The value held by PDOUT is not affected by writing to PDIR and PFUNC. However, the data value in PDOUT is driven out onto the McASP pin only if the corresponding bit in PFUNC is set to 1 (GPIO function) and the corresponding bit in PDIR is set to 1 (output). When reading data, returns the corresponding bit value in PDOUT[n], does not return input from I/O pin; when writing data, writes to the corresponding PDOUT[n] bit. The PDOUT is shown in Figure 19-38 and described in Table 19-13.

PDOUT has these aliases or alternate addresses:

- PDSET - when written to at this address, writing a 1 to a bit in PDSET sets the corresponding bit in PDOUT to 1; writing a 0 has no effect and keeps the bits in PDOUT unchanged.
- PDCLR - when written to at this address, writing a 1 to a bit in PDCLR clears the corresponding bit in PDOUT to 0; writing a 0 has no effect and keeps the bits in PDOUT unchanged.

There is only one set of data out bits, PDOUT[31-0]. The other registers, PDSET and PDCLR, are just different addresses for the same control bits, with different behaviors during writes.

CAUTION

Writing to Reserved Bits

Writing a value other than 0 to reserved bits in this register may cause improper device operation. This includes bits that are not implemented on a particular DSP.

Figure 19-38. Pin Data Output Register (PDOUT)

31	30	29	28	27	26	25	24
AFSR	AHCLKR	ACLKR	AFSX	AHCLKX	ACLKX	AMUTE	Reserved (A)
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R-0

23	16
Reserved (A)	
R-0	

15	14	13	12	11	10	9	8
AXR15	AXR14	AXR13	AXR12	AXR11	AXR10	AXR9	AXR8
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0

7	6	5	4	3	2	1	0
AXR7	AXR6	AXR5	AXR4	AXR3	AXR2	AXR1	AXR0
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

A If writing to this field, always write the default value for future device compatibility.

Copyright © 2013–2016, Texas Instruments Incorporated

Table 19-13. Pin Data Output Register (PDOUT) Field Descriptions

Bit	Field	Value	Description
31	AFSR		Determines drive on AFSR output pin when the corresponding PFUNC[31] and PDIR[31] bits are set to 1.
		0	Pin drives low.
		1	Pin drives high.
30	AHCLKR		Determines drive on AHCLKR output pin when the corresponding PFUNC[30] and PDIR[30] bits are set to 1.
		0	Pin drives low.
		1	Pin drives high.
29	ACLKR		Determines drive on ACLKR output pin when the corresponding PFUNC[29] and PDIR[29] bits are set to 1.
		0	Pin drives low.
		1	Pin drives high.
28	AFSX		Determines drive on AFSX output pin when the corresponding PFUNC[28] and PDIR[28] bits are set to 1.
		0	Pin drives low.
		1	Pin drives high.
27	AHCLKX		Determines drive on AHCLKX output pin when the corresponding PFUNC[27] and PDIR[27] bits are set to 1.
		0	Pin drives low.
		1	Pin drives high.
26	ACLKX		Determines drive on ACLKX output pin when the corresponding PFUNC[26] and PDIR[26] bits are set to 1.
		0	Pin drives low.
		1	Pin drives high.
25	AMUTE		Determines drive on AMUTE output pin when the corresponding PFUNC[25] and PDIR[25] bits are set to 1.
		0	Pin drives low.
		1	Pin drives high.
24-16	Reserved	0	Reserved. The reserved bit location always returns the default value. A value written to this field has no effect. If writing to this field, always write the default value for future device compatibility.
15-0	AXR[15-0]		Determines drive on AXR[n] output pin when the corresponding PFUNC[n] and PDIR[n] bits are set to 1.
		0	Pin drives low.
		1	Pin drives high.

Copyright © 2013–2016, Texas Instruments Incorporated

19.1.6 Pin Data Input Register (PDIN)

The pin data input register (PDIN) holds the I/O pin state of each of the McASP pins. PDIN allows the actual value of the pin to be read, regardless of the state of PFUNC and PDIR. The value after reset for registers 1 through 15 and 24 through 31 depends on how the pins are being driven. The PDIN is shown in Figure 19-39 and described in Table 19-14.

CAUTION

Writing to Reserved Bits

Writing a value other than 0 to reserved bits in this register may cause improper device operation. This includes bits that are not implemented on a particular DSP.

Figure 19-39. Pin Data Input Register (PDIN)

31	30	29	28	27	26	25	24
AFSR	AHCLKR	ACLKR	AFSX	AHCLKX	ACLKX	AMUTE	Reserved (A)
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R-0

23	16
Reserved (A)	
R-0	

15	14	13	12	11	10	9	8
AXR15	AXR14	AXR13	AXR12	AXR11	AXR10	AXR9	AXR8
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0

7	6	5	4	3	2	1	0
AXR7	AXR6	AXR5	AXR4	AXR3	AXR2	AXR1	AXR0
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

A If writing to this field, always write the default value for future device compatibility.

Copyright © 2013–2016, Texas Instruments Incorporated

Table 19-14. Pin Data Input Register (PDIN) Field Descriptions

Bit	Field	Value	Description
31	AFSR		Logic level on AFSR pin.
		0	Pin is logic low.
		1	Pin is logic high.
30	AHCLKR		Logic level on AHCLKR pin.
		0	Pin is logic low.
		1	Pin is logic high.
29	ACLKR		Logic level on ACLKR pin.
		0	Pin is logic low.
		1	Pin is logic high.
28	AFSX		Logic level on AFSX pin.
		0	Pin is logic low.
		1	Pin is logic high.
27	AHCLKX		Logic level on AHCLKX pin.
		0	Pin is logic low.
		1	Pin is logic high.
26	ACLKX		Logic level on ACLKX pin.
		0	Pin is logic low.
		1	Pin is logic high.
25	AMUTE		Logic level on AMUTE pin.
		0	Pin is logic low.
		1	Pin is logic high.
24-16	Reserved	0	Reserved. The reserved bit location always returns the default value. A value written to this field has no effect. If writing to this field, always write the default value for future device compatibility.
15-0	AXR[15-0]		Logic level on AXR[n] pin.
		0	Pin is logic low.
		1	Pin is logic high.

Copyright © 2013–2016, Texas Instruments Incorporated

19.1.7 Pin Data Set Register (PDSET)

The pin data set register (PDSET) is an alias of the pin data output register (PDOUT) for writes only. Writing a 1 to the PDSET bit sets the corresponding bit in PDOUT and, if PFUNC = 1 (GPIO function) and PDIR = 1 (output), drives a logic high on the pin. PDSET is useful for a multitasking system because it allows you to set to a logic high only the desired pin(s) within a system without affecting other I/O pins controlled by the same McASP. The PDSET is shown in Figure 19-40 and described in Table 19-15.

CAUTION

Writing to Reserved Bits

Writing a value other than 0 to reserved bits in this register may cause improper device operation. This includes bits that are not implemented on a particular DSP.

Figure 19-40. Pin Data Set Register (PDSET)

31	30	29	28	27	26	25	24
AFSR	AHCLKR	ACLKR	AFSX	AHCLKX	ACLKX	AMUTE	Reserved (A)
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R-0

23	16
Reserved (A)	
R-0	

15	14	13	12	11	10	9	8
AXR15	AXR14	AXR13	AXR12	AXR11	AXR10	AXR9	AXR8
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0

7	6	5	4	3	2	1	0
AXR7	AXR6	AXR5	AXR4	AXR3	AXR2	AXR1	AXR0
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

A If writing to this field, always write the default value for future device compatibility.

Copyright © 2013–2016, Texas Instruments Incorporated

Table 19-15. Pin Data Set Register (PDSET) Field Descriptions

Bit	Field	Value	Description
31	AFSR		Allows the corresponding AFSR bit in PDOUT to be set to a logic high without affecting other I/O pins controlled by the same port.
		0	No effect.
		1	PDOUT[31] bit is set to 1.
30	AHCLKR		Allows the corresponding AHCLKR bit in PDOUT to be set to a logic high without affecting other I/O pins controlled by the same port.
		0	No effect.
		1	PDOUT[30] bit is set to 1.
29	ACLKR		Allows the corresponding ACLKR bit in PDOUT to be set to a logic high without affecting other I/O pins controlled by the same port.
		0	No effect.
		1	PDOUT[29] bit is set to 1.
28	AFSX		Allows the corresponding AFSX bit in PDOUT to be set to a logic high without affecting other I/O pins controlled by the same port.
		0	No effect.
		1	PDOUT[28] bit is set to 1.
27	AHCLKX		Allows the corresponding AHCLKX bit in PDOUT to be set to a logic high without affecting other I/O pins controlled by the same port.
		0	No effect.
		1	PDOUT[27] bit is set to 1.
26	ACLKX		Allows the corresponding ACLKX bit in PDOUT to be set to a logic high without affecting other I/O pins controlled by the same port.
		0	No effect.
		1	PDOUT[26] bit is set to 1.
25	AMUTE		Allows the corresponding AMUTE bit in PDOUT to be set to a logic high without affecting other I/O pins controlled by the same port.
		0	No effect.
		1	PDOUT[25] bit is set to 1.
24-16	Reserved	0	Reserved. The reserved bit location always returns the default value. A value written to this field has no effect. If writing to this field, always write the default value for future device compatibility.
15-0	AXR[15-0]		Allows the corresponding AXR[n] bit in PDOUT to be set to a logic high without affecting other I/O pins controlled by the same port.
		0	No effect.
		1	PDOUT[n] bit is set to 1.

Copyright © 2013–2016, Texas Instruments Incorporated

19.1.8 Pin Data Clear Register (PDCLR)

The pin data clear register (PDCLR) is an alias of the pin data output register (PDOUT) for writes only. Writing a 1 to the PDCLR bit clears the corresponding bit in PDOUT and, if PFUNC = 1 (GPIO function) and PDIR = 1 (output), drives a logic low on the pin. PDCLR is useful for a multitasking system because it allows you to clear to a logic low only the desired pin(s) within a system without affecting other I/O pins controlled by the same McASP. The PDCLR is shown in Figure 19-41 and described in Table 19-16.

CAUTION

Writing to Reserved Bits

Writing a value other than 0 to reserved bits in this register may cause improper device operation. This includes bits that are not implemented on a particular DSP.

Figure 19-41. Pin Data Clear Register (PDCLR)

31	30	29	28	27	26	25	24
AFSR	AHCLKR	ACLKR	AFSX	AHCLKX	ACLKX	AMUTE	Reserved (A)
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R-0

23	16
Reserved (A)	
R-0	

15	14	13	12	11	10	9	8
AXR15	AXR14	AXR13	AXR12	AXR11	AXR10	AXR9	AXR8
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0

7	6	5	4	3	2	1	0
AXR7	AXR6	AXR5	AXR4	AXR3	AXR2	AXR1	AXR0
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

A If writing to this field, always write the default value for future device compatibility.

Copyright © 2013–2016, Texas Instruments Incorporated

Table 19-16. Pin Data Clear Register (PDCLR) Field Descriptions

Bit	Field	Value	Description
31	AFSR		Allows the corresponding AFSR bit in PDOUT to be cleared to a logic low without affecting other I/O pins controlled by the same port.
		0	No effect.
		1	PDOUT[31] bit is cleared to 0.
30	AHCLKR		Allows the corresponding AHCLKR bit in PDOUT to be cleared to a logic low without affecting other I/O pins controlled by the same port.
		0	No effect.
		1	PDOUT[30] bit is cleared to 0.
29	ACLKR		Allows the corresponding ACLKR bit in PDOUT to be cleared to a logic low without affecting other I/O pins controlled by the same port.
		0	No effect.
		1	PDOUT[29] bit is cleared to 0.
28	AFSX		Allows the corresponding AFSX bit in PDOUT to be cleared to a logic low without affecting other I/O pins controlled by the same port.
		0	No effect.
		1	PDOUT[28] bit is cleared to 0.
27	AHCLKX		Allows the corresponding AHCLKX bit in PDOUT to be cleared to a logic low without affecting other I/O pins controlled by the same port.
		0	No effect.
		1	PDOUT[27] bit is cleared to 0.
26	ACLKX		Allows the corresponding ACLKX bit in PDOUT to be cleared to a logic low without affecting other I/O pins controlled by the same port.
		0	No effect.
		1	PDOUT[26] bit is cleared to 0.
25	AMUTE		Allows the corresponding AMUTE bit in PDOUT to be cleared to a logic low without affecting other I/O pins controlled by the same port.
		0	No effect.
		1	PDOUT[25] bit is cleared to 0.
24-16	Reserved	0	Reserved. The reserved bit location always returns the default value. A value written to this field has no effect. If writing to this field, always write the default value for future device compatibility.
15-0	AXR[15-0]		Allows the corresponding AXR[n] bit in PDOUT to be cleared to a logic low without affecting other I/O pins controlled by the same port.
		0	No effect.
		1	PDOUT[n] bit is cleared to 0.

Copyright © 2013–2016, Texas Instruments Incorporated

19.1.9 Global Control Register (GBLCTL)

The global control register (GBLCTL) provides initialization of the transmit and receive sections. The GBLCTL is shown in Figure 19-42 and described in Table 19-17.

The bit fields in GBLCTL are synchronized and latched by the corresponding clocks (ACLKX for bits 12-8 and ACLKR for bits 4-0). Before GBLCTL is programmed, you must ensure that serial clocks are running. If the corresponding external serial clocks, ACLKX and ACLKR, are not yet running, you should select the internal serial clock source in AHCLKXCTL, AHCLKRCTL, ACLKXCTL, and ACLKRCTL before GBLCTL is programmed. Also, after programming any bits in GBLCTL you should not proceed until you have read back from GBLCTL and verified that the bits are latched in GBLCTL.

Figure 19-42. Global Control Register (GBLCTL)

31	16
Reserved (A)	
R-0	

15	13	12	11	10	9	8
Reserved (A)		XFRST	XSMRST	XSRCLR	XHCLKRST	XCLKRST
R-0		R/W-0	R/W-0	R/W-0	R/W-0	R/W-0

7	5	4	3	2	1	0
Reserved (A)		RFRST	RSMRST	RSRCLR	RHCLKRST	RCLKRST
R-0		R/W-0	R/W-0	R/W-0	R/W-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

A If writing to this field, always write the default value for future device compatibility.

Table 19-17. Global Control Register (GBLCTL) Field Descriptions

Bit	Field	Value	Description
31-13	Reserved	0	Reserved. The reserved bit location always returns the default value. A value written to this field has no effect. If writing to this field, always write the default value for future device compatibility.
12	XFRST		Transmit frame sync generator reset enable bit.
		0	Transmit frame sync generator is reset.
		1	Transmit frame sync generator is active. When released from reset, the transmit frame sync generator begins counting serial clocks and generating frame sync as programmed.
11	XSMRST		Transmit state machine reset enable bit.
		0	Transmit state machine is held in reset. AXR[n] pin state: If PFUNC[n] = 0 and PDIR[n] = 1; then the serializer drives the AXR[n] pin to the state specified for inactive time slot (as determined by DISMOD bits in SRCTL).
		1	Transmit state machine is released from reset. When released from reset, the transmit state machine immediately transfers data from XRBUF[n] to XRSR[n]. The transmit state machine sets the underrun flag (XUNDRN) in XSTAT, if XRBUF[n] have not been preloaded with data before reset is released. The transmit state machine also immediately begins detecting frame sync and is ready to transmit. Transmit TDM time slot begins at slot 0 after reset is released.
10	XSRCLR		Transmit serializer clear enable bit. By clearing then setting this bit, the transmit buffer is flushed to an empty state (XDATA = 1). If XSMRST = 1, XSRCLR = 1, XDATA = 1, and XBUF is not loaded with new data before the start of the next active time slot, an underrun will occur.
		0	Transmit serializers are cleared.
		1	Transmit serializers are active. When the transmit serializers are first taken out of reset (XSRCLR changes from 0 to 1), the transmit data ready bit (XDATA) in XSTAT is set to indicate XBUF is ready to be written.
9	XHCLKRST		Transmit high-frequency clock divider reset enable bit.
		0	Transmit high-frequency clock divider is held in reset.
		1	Transmit high-frequency clock divider is running.

Copyright © 2013–2016, Texas Instruments Incorporated

Table 19-17. Global Control Register (GBLCTL) Field Descriptions (continued)

Bit	Field	Value	Description
8	XCLKRST		Transmit clock divider reset enable bit.
		0	Transmit clock divider is held in reset. When the clock divider is in reset, it passes through a divide-by-1 of its input.
		1	Transmit clock divider is running.
7-5	Reserved	0	Reserved. The reserved bit location always returns the default value. A value written to this field has no effect. If writing to this field, always write the default value for future device compatibility.
4	RFRST		Receive frame sync generator reset enable bit.
		0	Receive frame sync generator is reset.
		1	Receive frame sync generator is active. When released from reset, the receive frame sync generator begins counting serial clocks and generating frame sync as programmed.
3	RSMRST		Receive state machine reset enable bit.
		0	Receive state machine is held in reset.
		1	Receive state machine is released from reset. When released from reset, the receive state machine immediately begins detecting frame sync and is ready to receive. Receive TDM time slot begins at slot 0 after reset is released.
2	RSRCLR		Receive serializer clear enable bit. By clearing then setting this bit, the receive buffer is flushed.
		0	Receive serializers are cleared.
		1	Receive serializers are active.
1	RHCLKRST		Receive high-frequency clock divider reset enable bit.
		0	Receive high-frequency clock divider is held in reset.
		1	Receive high-frequency clock divider is running.
0	RCLKRST		Receive clock divider reset enable bit.
		0	Receive clock divider is held in reset. When the clock divider is in reset, it passes through a divide-by-1 of its input.
		1	Receive clock divider is running.

Copyright © 2013–2016, Texas Instruments Incorporated

19.1.10 Audio Mute Control Register (AMUTE)

The audio mute control register (AMUTE) controls the McASP audio mute (AMUTE) output pin. The value after reset for register 4 depends on how the pins are being driven. The AMUTE is shown in Figure 19-43 and described in Table 19-18.

Figure 19-43. Audio Mute Control Register (AMUTE)

31	16
Reserved (A)	
R-0	

15	13	12	11	10	9	8
Reserved (A)		XDMAERR	RDMAERR	XCKFAIL	RCKFAIL	XSYNCERR
R-0		R/W-0	R/W-0	R/W-0	R/W-0	R/W-0

7	6	5	4	3	2	1	0
RSYNCERR	XUNDRN	ROVRN	INSTAT	INEN	INPOL	MUTEN	
R/W-0	R/W-0	R/W-0	R-0	R/W-0	R/W-0	R/W-0	

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

A If writing to this field, always write the default value for future device compatibility.

Table 19-18. Audio Mute Control Register (AMUTE) Field Descriptions

Bit	Field	Value	Description
31-13	Reserved	0	Reserved. The reserved bit location always returns the default value. A value written to this field has no effect. If writing to this field, always write the default value for future device compatibility.
12	XDMAERR		If transmit DMA error (XDMAERR), drive AMUTE active enable bit.
		0	Drive is disabled. Detection of transmit DMA error is ignored by AMUTE.
		1	Drive is enabled (active). Upon detection of transmit DMA error, AMUTE is active and is driven according to MUTEN bit.
11	RDMAERR		If receive DMA error (RDMAERR), drive AMUTE active enable bit.
		0	Drive is disabled. Detection of receive DMA error is ignored by AMUTE.
		1	Drive is enabled (active). Upon detection of receive DMA error, AMUTE is active and is driven according to MUTEN bit.
10	XCKFAIL		If transmit clock failure (XCKFAIL), drive AMUTE active enable bit.
		0	Drive is disabled. Detection of transmit clock failure is ignored by AMUTE.
		1	Drive is enabled (active). Upon detection of transmit clock failure, AMUTE is active and is driven according to MUTEN bit
9	RCKFAIL		If receive clock failure (RCKFAIL), drive AMUTE active enable bit.
		0	Drive is disabled. Detection of receive clock failure is ignored by AMUTE.
		1	Drive is enabled (active). Upon detection of receive clock failure, AMUTE is active and is driven according to MUTEN bit.
8	XSYNCERR		If unexpected transmit frame sync error (XSYNCERR), drive AMUTE active enable bit.
		0	Drive is disabled. Detection of unexpected transmit frame sync error is ignored by AMUTE.
		1	Drive is enabled (active). Upon detection of unexpected transmit frame sync error, AMUTE is active and is driven according to MUTEN bit.
7	RSYNCERR		If unexpected receive frame sync error (RSYNCERR), drive AMUTE active enable bit.
		0	Drive is disabled. Detection of unexpected receive frame sync error is ignored by AMUTE.
		1	Drive is enabled (active). Upon detection of unexpected receive frame sync error, AMUTE is active and is driven according to MUTEN bit.
6	XUNDRN		If transmit underrun error (XUNDRN), drive AMUTE active enable bit.
		0	Drive is disabled. Detection of transmit underrun error is ignored by AMUTE.
		1	Drive is enabled (active). Upon detection of transmit underrun error, AMUTE is active and is driven according to MUTEN bit.

Copyright © 2013–2016, Texas Instruments Incorporated

Table 19-18. Audio Mute Control Register (AMUTE) Field Descriptions (continued)

Bit	Field	Value	Description
5	ROVRN		If receiver overrun error (ROVRN), drive AMUTE active enable bit.
		0	Drive is disabled. Detection of receiver overrun error is ignored by AMUTE.
		1	Drive is enabled (active). Upon detection of receiver overrun error, AMUTE is active and is driven according to MUTEN bit.
4	INSTAT		Determines drive on AXR[n] pin when PFUNC[n] and PDIR[n] bits are set to 1.
		0	AMUTEIN pin is inactive.
		1	AMUTEIN pin is active. Audio mute in error is detected.
3	INEN		Drive AMUTE active when AMUTEIN error is active (INSTAT = 1).
		0	Drive is disabled. AMUTEIN is ignored by AMUTE.
		1	Drive is enabled (active). INSTAT = 1 drives AMUTE active.
2	INPOL		Audio mute in (AMUTEIN) polarity select bit.
		0	Polarity is active high. A high on AMUTEIN sets INSTAT to 1.
		1	Polarity is active low. A low on AMUTEIN sets INSTAT to 1.
1-0	MUTEN	0-3h	AMUTE pin enable bit (unless overridden by GPIO registers).
		0	AMUTE pin is disabled, pin goes to tri-state condition.
		1h	AMUTE pin is driven high if error is detected.
		2h	AMUTE pin is driven low if error is detected.
		3h	Reserved

Copyright © 2013–2016, Texas Instruments Incorporated

19.1.11 Digital Loopback Control Register (DLBCTL)

The digital loopback control register (DLBCTL) controls the internal loopback settings of the McASP in TDM mode. The DLBCTL is shown in Figure 19-44 and described in Table 19-19.

Figure 19-44. Digital Loopback Control Register (DLBCTL)

31	16
Reserved (A)	
R-0	

15	4	3	2	1	0
Reserved (A)		MODE		ORD	DLBEN
R-0		R/W-0		R/W-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

A If writing to this field, always write the default value for future device compatibility.

Table 19-19. Digital Loopback Control Register (DLBCTL) Field Descriptions

Bit	Field	Value	Description
31-4	Reserved	0	Reserved. The reserved bit location always returns the default value. A value written to this field has no effect. If writing to this field, always write the default value for future device compatibility.
3-2	MODE	0-3h	Loopback generator mode bits. Applies only when loopback mode is enabled (DLBEN = 1).
		0	Default and reserved on loopback mode (DLBEN = 1). When in non-loopback mode (DLBEN = 0), MODE should be left at default (00). When in loopback mode (DLBEN = 1), MODE = 00 is reserved and not applicable.
		1h	Transmit clock and frame sync generators used by both transmit and receive sections. When in loopback mode (DLBEN = 1), MODE must be 01.
		2h-3h	Reserved.
1	ORD		Loopback order bit when loopback mode is enabled (DLBEN = 1).
		0	Odd serializers N + 1 transmit to even serializers N that receive. The corresponding serializers must be programmed properly.
		1	Even serializers N transmit to odd serializers N+1 that receive. The corresponding serializers must be programmed properly.
0	DLBEN		Loopback mode enable bit.
		0	Loopback mode is disabled.
		1	Loopback mode is enabled.

Copyright © 2013–2016, Texas Instruments Incorporated

19.1.12 Digital Mode Control Register (DITCTL)

The DIT mode control register (DITCTL) controls DIT operations of the McASP. The DITCTL is shown in Figure 19-45 and described in Table 19-20.

Figure 19-45. Digital Mode Control Register (DITCTL)

31	16
Reserved [A]	
R-0	

15	4	3	2	1	0
Reserved [A]		VB	VA	Rsvd [A]	DITEN
R-0		R/W-0	R/W-0	R-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

A If writing to this field, always write the default value for future device compatibility.

Table 19-20. Digital Mode Control Register (DITCTL) Field Descriptions

Bit	Field	Value	Description
31-4	Reserved	0	Reserved. The reserved bit location always returns the default value. A value written to this field has no effect. If writing to this field, always write the default value for future device compatibility.
3	VB		Valid bit for odd time slots (DIT right subframe).
		0	V bit is 0 during odd DIT subframes.
		1	V bit is 1 during odd DIT subframes.
2	VA		Valid bit for even time slots (DIT left subframe).
		0	V bit is 0 during even DIT subframes.
		1	V bit is 1 during even DIT subframes.
1	Reserved	0	Reserved. The reserved bit location always returns the default value. A value written to this field has no effect. If writing to this field, always write the default value for future device compatibility.
0	DITEN		DIT mode enable bit. DITEN should only be changed while the XSMRST bit in GBLCTL is in reset (and for startup, XSRCLR also in reset). However, it is not necessary to reset the XCLKRST or XHCLKRST bits in GBLCTL to change DITEN.
		0	DIT mode is disabled. Transmitter operates in TDM or burst mode.
		1	DIT mode is enabled. Transmitter operates in DIT encoded mode.

Copyright © 2013–2016, Texas Instruments Incorporated

19.1.13 Receiver Global Control Register (RGBLCTL)

Alias of the global control register (GBLCTL). Writing to the receiver global control register (RGBLCTL) affects only the receive bits of GBLCTL (bits 4-0). Reads from RGBLCTL return the value of GBLCTL. RGBLCTL allows the receiver to be reset independently from the transmitter. The RGBLCTL is shown in Figure 19-46 and described in Table 19-21. See Section 19.1.9 for a detailed description of GBLCTL.

Figure 19-46. Receiver Global Control Register (RGBLCTL)

31	16
Reserved (A)	
R-0	

15	13	12	11	10	9	8
Reserved (A)		XFRST	XSMRST	XSRCLR	XHCLKRST	XCLKRST
R-0		R-0	R-0	R-0	R-0	R-0

7	5	4	3	2	1	0
Reserved (A)		RFRST	RSMRST	RSRCLR	RHCLKRST	RCLKRST
R-0		R/W-0	R/W-0	R/W-0	R/W-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

A If writing to this field, always write the default value for future device compatibility.

Table 19-21. Receiver Global Control Register (RGBLCTL) Field Descriptions

Bit	Field	Value	Description
31-13	Reserved	0	Reserved. The reserved bit location always returns the default value. A value written to this field has no effect. If writing to this field, always write the default value for future device compatibility.
12	XFRST	x	Transmit frame sync generator reset enable bit. A read of this bit returns the XFRST bit value of GBLCTL. Writes have no effect.
11	XSMRST	x	Transmit state machine reset enable bit. A read of this bit returns the XSMRST bit value of GBLCTL. Writes have no effect.
10	XSRCLR	x	Transmit serializer clear enable bit. A read of this bit returns the XSRCLR bit value of GBLCTL. Writes have no effect.
9	XHCLKRST	x	Transmit high-frequency clock divider reset enable bit. A read of this bit returns the XHCLKRST bit value of GBLCTL. Writes have no effect.
8	XCLKRST	x	Transmit clock divider reset enable bit. a read of this bit returns the XCLKRST bit value of GBLCTL. Writes have no effect.
7-5	Reserved	0	Reserved. The reserved bit location always returns the default value. A value written to this field has no effect. If writing to this field, always write the default value for future device compatibility.
4	RFRST		Receive frame sync generator reset enable bit. A write to this bit affects the RFRST bit of GBLCTL.
		0	Receive frame sync generator is reset.
		1	Receive frame sync generator is active.
3	RSMRST		Receive state machine reset enable bit. A write to this bit affects the RSMRST bit of GBLCTL.
		0	Receive state machine is held in reset.
		1	Receive state machine is released from reset.
2	RSRCLR		Receive serializer clear enable bit. A write to this bit affects the RSRCLR bit of GBLCTL.
		0	Receive serializers are cleared.
		1	Receive serializers are active.
1	RHCLKRST		Receive high-frequency clock divider reset enable bit. A write to this bit affects the RHCLKRST bit of GBLCTL.
		0	Receive high-frequency clock divider is held in reset.
		1	Receive high-frequency clock divider is running.
0	RCLKRST		Receive clock divider reset enable bit. A write to this bit affects the RCLKRST bit of GBLCTL.
		0	Receive clock divider is held in reset.
		1	Receive clock divider is running.

Copyright © 2013–2016, Texas Instruments Incorporated

19.1.14 Receive Format Unit Bit Mask Register (RMASK)

The receive format unit bit mask register (RMASK) determines which bits of the received data are masked off and padded with a known value before being read by the CPU or DMA. The RMASK is shown in Figure 19-47 and described in Table 19-22.

Figure 19-47. Receive Format Unit Bit Mask Register (RMASK)

31	30	29	28	27	26	25	24
RMASK31	RMASK30	RMASK29	RMASK28	RMASK27	RMASK26	RMASK25	RMASK24
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
23	22	21	20	19	18	17	16
RMASK23	RMASK22	RMASK21	RMASK20	RMASK19	RMASK18	RMASK17	RMASK16
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
15	14	13	12	11	10	9	8
RMASK15	RMASK14	RMASK13	RMASK12	RMASK11	RMASK10	RMASK9	RMASK8
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
7	6	5	4	3	2	1	0
RMASK7	RMASK6	RMASK5	RMASK4	RMASK3	RMASK2	RMASK1	RMASK0
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0

LEGEND: R/W = Read/Write; -*n* = value after reset

Table 19-22. Receive Format Unit Bit Mask Register (RMASK) Field Descriptions

Bit	Field	Value	Description
31-0	RMASK[31-0]		Receive data mask enable bit.
		0	Corresponding bit of receive data (after passing through reverse and rotate units) is masked out and then padded with the selected bit pad value (RPAD and RPBIT bits in RFMT).
		1	Corresponding bit of receive data (after passing through reverse and rotate units) is returned to CPU or DMA.

Copyright © 2013–2016, Texas Instruments Incorporated

19.1.15 Receive Bit Stream Format Register (RFMT)

The receive bit stream format register (RFMT) configures the receive data format. The RFMT is shown in Figure 19-48 and described in Table 19-23.

Figure 19-48. Receive Bit Stream Format Register (RFMT)

31																		18	17	16
Reserved (A)																			RDATDLY	
R-0																			R/W-0	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RRVRS	RPAD		RPBIT					RSSZ				RBUSEL	RROT		
R/W-0	R/W-0		R/W-0					R/W-0				R/W-0	R/W-0		

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

A If writing to this field, always write the default value for future device compatibility.

Table 19-23. Receive Bit Stream Format Register (RFMT) Field Descriptions

Bit	Field	Value	Description
31-18	Reserved	0	Reserved. The reserved bit location always returns the default value. A value written to this field has no effect. If writing to this field, always write the default value for future device compatibility.
17-16	RDATDLY	0-3h	Receive bit delay.
		0	0-bit delay. The first receive data bit, AXR[n], occurs in same ACLKR cycle as the receive frame sync (AFSR).
		1h	1-bit delay. The first receive data bit, AXR[n], occurs one ACLKR cycle after the receive frame sync (AFSR).
		2h	2-bit delay. The first receive data bit, AXR[n], occurs two ACLKR cycles after the receive frame sync (AFSR).
		3h	Reserved.
15	RRVRS		Receive serial bitstream order.
		0	Bitstream is LSB first. No bit reversal is performed in receive format bit reverse unit.
		1	Bitstream is MSB first. Bit reversal is performed in receive format bit reverse unit.
14-13	RPAD	0-3h	Pad value for extra bits in slot not belonging to the word. This field only applies to bits when RMASK[n] = 0.
		0	Pad extra bits with 0.
		1h	Pad extra bits with 1.
		2h	Pad extra bits with one of the bits from the word as specified by RPBIT bits.
		3h	Reserved.
12-8	RPBIT	0-1Fh	RPBIT value determines which bit (as read by the CPU or DMA from RBUF[n]) is used to pad the extra bits. This field only applies when RPAD = 2h.
		0	Pad with bit 0 value.
		1h-1Fh	Pad with bit 1 to bit 31 value.

Copyright © 2013–2016, Texas Instruments Incorporated

Table 19-23. Receive Bit Stream Format Register (RFMT) Field Descriptions (continued)

Bit	Field	Value	Description
7-4	RSSZ	0-Fh	Receive slot size.
		0-2h	Reserved
		3h	Slot size is 8 bits.
		4h	Reserved
		5h	Slot size is 12 bits.
		6h	Reserved
		7h	Slot size is 16 bits.
		8h	Reserved
		9h	Slot size is 20 bits.
		Ah	Reserved
		Bh	Slot size is 24 bits
		Ch	Reserved
		Dh	Slot size is 28 bits.
		Eh	Reserved
		Fh	Slot size is 32 bits.
3	RBUSEL		Selects whether reads from serializer buffer XRBUF[n] by way of RBUF*n* by the CPU/EDMA occur through the peripheral configuration port or the DMA port.
		0	Reads from XRBUF[n] originate on the DMA port. Reads from XRBUF[n] on the peripheral configuration port are ignored.
		1	Reads from XRBUF[n] originate on the peripheral configuration port. Reads from XRBUF[n] on the DMA port are ignored.
2-0	RROT	0-7h	Right-rotation value for receive rotate right format unit.
		0	Rotate right by 0 (no rotation).
		1h	Rotate right by 4 bit positions.
		2h	Rotate right by 8 bit positions.
		3h	Rotate right by 12 bit positions.
		4h	Rotate right by 16 bit positions.
		5h	Rotate right by 20 bit positions.
		6h	Rotate right by 24 bit positions.
		7h	Rotate right by 28 bit positions.

Copyright © 2013–2016, Texas Instruments Incorporated

19.1.16 Receive Frame Sync Control Register (AFSRCTL)

The receive frame sync control register (AFSRCTL) configures the receive frame sync (AFSR). The AFSRCTL is shown in Figure 19-49 and described in Table 19-24.

Figure 19-49. Receive Frame Sync Control Register (AFSRCTL)

31	16
Reserved [A]	
R-0	

15	7	6	5	4	3	2	1	0
RMOD		Reserved [A]		FRWID	Reserved [A]		FSRM	FSRP
R/W-0		R-0		R/W-0	R-0		R/W-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

A If writing to this field, always write the default value for future device compatibility.

Table 19-24. Receive Frame Sync Control Register (AFSRCTL) Field Descriptions

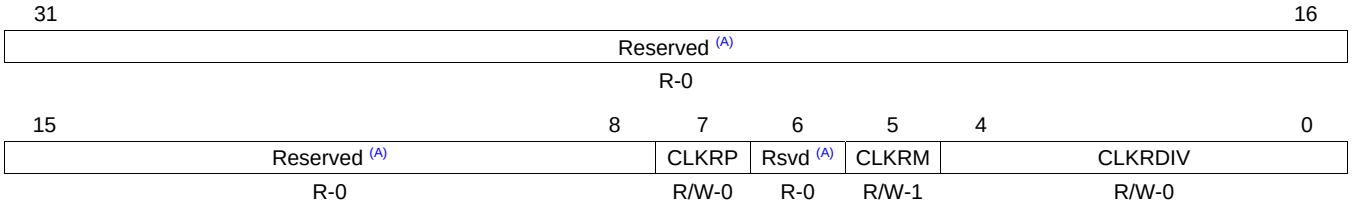
Bit	Field	Value	Description
31-16	Reserved	0	Reserved. The reserved bit location always returns the default value. A value written to this field has no effect. If writing to this field, always write the default value for future device compatibility.
15-7	RMOD	0-1FFh	Receive frame sync mode select bits.
		0	Burst mode
		1h	Reserved
		2h-20h	2-slot TDM (I2S mode) to 32-slot TDM
		21h-17Fh	Reserved
		180h	384-slot TDM (external DIR IC inputting 384-slot DIR frames to McASP over I2S interface)
		181h-1FFh	Reserved
6-5	Reserved	0	Reserved. The reserved bit location always returns the default value. A value written to this field has no effect. If writing to this field, always write the default value for future device compatibility.
4	FRWID		Receive frame sync width select bit indicates the width of the receive frame sync (AFSR) during its active period.
		0	Single bit
		1	Single word
3-2	Reserved	0	Reserved. The reserved bit location always returns the default value. A value written to this field has no effect. If writing to this field, always write the default value for future device compatibility.
1	FSRM		Receive frame sync generation select bit.
		0	Externally-generated receive frame sync
		1	Internally-generated receive frame sync
0	FSRP		Receive frame sync polarity select bit.
		0	A rising edge on receive frame sync (AFSR) indicates the beginning of a frame.
		1	A falling edge on receive frame sync (AFSR) indicates the beginning of a frame.

Copyright © 2013–2016, Texas Instruments Incorporated

19.1.17 Receive Clock Control Register (ACLKRCTL)

The receive clock control register (ACLKRCTL) configures the receive bit clock (ACLKR) and the receive clock generator. The ACLKRCTL is shown in Figure 19-50 and described in Table 19-25.

Figure 19-50. Receive Clock Control Register (ACLKRCTL)



LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

A If writing to this field, always write the default value for future device compatibility.

Table 19-25. Receive Clock Control Register (ACLKRCTL) Field Descriptions

Bit	Field	Value	Description
31-8	Reserved	0	Reserved. The reserved bit location always returns the default value. A value written to this field has no effect. If writing to this field, always write the default value for future device compatibility.
7	CLKRP		Receive bitstream clock polarity select bit. Note that this bitfield does not have any effect, if ACLKXCTL.ASYNC = 0 (see Section 19.1.29 for a description for the ASYNC bit).
		0	Falling edge. Receiver samples data on the falling edge of the serial clock, so the external transmitter driving this receiver must shift data out on the rising edge of the serial clock.
		1	Rising edge. Receiver samples data on the rising edge of the serial clock, so the external transmitter driving this receiver must shift data out on the falling edge of the serial clock.
6	Reserved	0	Reserved. The reserved bit location always returns the default value. A value written to this field has no effect. If writing to this field, always write the default value for future device compatibility.
5	CLKRM		Receive bit clock source bit. Note that this bitfield does not have any effect, if ACLKXCTL.ASYNC = 0 (see Section 19.1.29 for a description for the ASYNC bit).
		0	External receive clock source from ACLKR pin.
		1	Internal receive clock source from output of programmable bit clock divider.
4-0	CLKRDIV	0-1Fh	Receive bit clock divide ratio bits determine the divide-down ratio from AHCLKR to ACLKR. Note that this bitfield does not have any effect, if ACLKXCTL.ASYNC = 0 (see Section 19.1.29 for a description for the ASYNC bit).
		0	Divide-by-1
		1h	Divide-by-2
		2h-1Fh	Divide-by-3 to divide-by-32

Copyright © 2013–2016, Texas Instruments Incorporated

19.1.18 Receive High-Frequency Clock Control Register (AHCLKRCTL)

The receive high-frequency clock control register (AHCLKRCTL) configures the receive high-frequency master clock (AHCLKR) and the receive clock generator. The AHCLKRCTL is shown in Figure 19-51 and described in Table 19-26.

Figure 19-51. Receive High-Frequency Clock Control Register (AHCLKRCTL)

31					16
Reserved [A]					
R-0					

15	14	13	12	11	0
HCLKRM	HCLKRP	Reserved [A]		HCLKRDIV	
R/W-1	R/W-0	R/W-0		R/W-0	

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

A If writing to this field, always write the default value for future device compatibility.

Table 19-26. Receive High-Frequency Clock Control Register (AHCLKRCTL) Field Descriptions

Bit	Field	Value	Description
31-16	Reserved	0	Reserved. The reserved bit location always returns the default value. A value written to this field has no effect. If writing to this field, always write the default value for future device compatibility.
15	HCLKRM		Receive high-frequency clock source bit.
		0	External receive high-frequency clock source from AHCLKR pin.
		1	Internal receive high-frequency clock source from output of programmable high clock divider.
14	HCLKRP		Receive bitstream high-frequency clock polarity select bit.
		0	Not inverted. AHCLKR is not inverted before programmable bit clock divider. In the special case where the receive bit clock (ACLKR) is internally generated and the programmable bit clock divider is set to divide-by-1 (CLKRDIV = 0 in ACLKRCTL), AHCLKR is directly passed through to the ACLKR pin.
		1	Inverted. AHCLKR is inverted before programmable bit clock divider. In the special case where the receive bit clock (ACLKR) is internally generated and the programmable bit clock divider is set to divide-by-1 (CLKRDIV = 0 in ACLKRCTL), AHCLKR is directly passed through to the ACLKR pin.
13-12	Reserved	0	Reserved. The reserved bit location always returns the default value. A value written to this field has no effect. If writing to this field, always write the default value for future device compatibility.
11-0	HCLKRDIV	0-FFFh	Receive high-frequency clock divide ratio bits determine the divide-down ratio from AUXCLK to AHCLKR.
		0	Divide-by-1
		1h	Divide-by-2
		2h-FFFh	Divide-by-3 to divide-by-4096

Copyright © 2013–2016, Texas Instruments Incorporated

19.1.19 Receive TDM Time Slot Register (RTDM)

The receive TDM time slot register (RTDM) specifies which TDM time slot the receiver is active. The RTDM is shown in Figure 19-52 and described in Table 19-27.

Figure 19-52. Receive TDM Time Slot Register (RTDM)

31	30	29	28	27	26	25	24
RTDMS31	RTDMS30	RTDMS29	RTDMS28	RTDMS27	RTDMS26	RTDMS25	RTDMS24
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
23	22	21	20	19	18	17	16
RTDMS23	RTDMS22	RTDMS21	RTDMS20	RTDMS19	RTDMS18	RTDMS17	RTDMS16
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
15	14	13	12	11	10	9	8
RTDMS15	RTDMS14	RTDMS13	RTDMS12	RTDMS11	RTDMS10	RTDMS9	RTDMS8
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
7	6	5	4	3	2	1	0
RTDMS7	RTDMS6	RTDMS5	RTDMS4	RTDMS3	RTDMS2	RTDMS1	RTDMS0
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 19-27. Receive TDM Time Slot Register (RTDM) Field Descriptions

Bit	Field	Value	Description
31-0	RTDMS[31-0]		Receiver mode during TDM time slot *n*.
		0	Receive TDM time slot *n* is inactive. The receive serializer does not shift in data during this slot.
		1	Receive TDM time slot *n* is active. The receive serializer shifts in data during this slot.

Copyright © 2013–2016, Texas Instruments Incorporated

19.1.20 Receiver Interrupt Control Register (RINTCTL)

The receiver interrupt control register (RINTCTL) controls generation of the McASP receive interrupt (RINT). When the register bit(s) is set to 1, the occurrence of the enabled McASP condition(s) generates RINT. The RINTCTL is shown in Figure 19-53 and described in Table 19-28. See Section 19.1.21 for a description of the interrupt conditions.

Figure 19-53. Receiver Interrupt Control Register (RINTCTL)

31							8
Reserved (A)							
R-0							
7	6	5	4	3	2	1	0
RSTAFRM	Reserved (A)	RDATA	RLAST	RDMAERR	RCKFAIL	RSYNCERR	ROVRN
R/W-0	R-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

A If writing to this field, always write the default value for future device compatibility.

Table 19-28. Receiver Interrupt Control Register (RINTCTL) Field Descriptions

Bit	Field	Value	Description
31-8	Reserved	0	Reserved. The reserved bit location always returns the default value. A value written to this field has no effect. If writing to this field, always write the default value for future device compatibility.
7	RSTAFRM		Receive start of frame interrupt enable bit.
		0	Interrupt is disabled. A receive start of frame interrupt does not generate a McASP receive interrupt (RINT).
		1	Interrupt is enabled. A receive start of frame interrupt generates a McASP receive interrupt (RINT).
6	Reserved	0	Reserved. The reserved bit location always returns the default value. A value written to this field has no effect. If writing to this field, always write the default value for future device compatibility.
5	RDATA		Receive data ready interrupt enable bit.
		0	Interrupt is disabled. A receive data ready interrupt does not generate a McASP receive interrupt (RINT).
		1	Interrupt is enabled. A receive data ready interrupt generates a McASP receive interrupt (RINT).
4	RLAST		Receive last slot interrupt enable bit.
		0	Interrupt is disabled. A receive last slot interrupt does not generate a McASP receive interrupt (RINT).
		1	Interrupt is enabled. A receive last slot interrupt generates a McASP receive interrupt (RINT).
3	RDMAERR		Receive DMA error interrupt enable bit.
		0	Interrupt is disabled. A receive DMA error interrupt does not generate a McASP receive interrupt (RINT).
		1	Interrupt is enabled. A receive DMA error interrupt generates a McASP receive interrupt (RINT).
2	RCKFAIL		Receive clock failure interrupt enable bit.
		0	Interrupt is disabled. A receive clock failure interrupt does not generate a McASP receive interrupt (RINT).
		1	Interrupt is enabled. A receive clock failure interrupt generates a McASP receive interrupt (RINT).
1	RSYNCERR		Unexpected receive frame sync interrupt enable bit.
		0	Interrupt is disabled. An unexpected receive frame sync interrupt does not generate a McASP receive interrupt (RINT).
		1	Interrupt is enabled. An unexpected receive frame sync interrupt generates a McASP receive interrupt (RINT).
0	ROVRN		Receiver overrun interrupt enable bit.
		0	Interrupt is disabled. A receiver overrun interrupt does not generate a McASP receive interrupt (RINT).
		1	Interrupt is enabled. A receiver overrun interrupt generates a McASP receive interrupt (RINT).

Copyright © 2013–2016, Texas Instruments Incorporated

19.1.21 Receiver Status Register (RSTAT)

The receiver status register (RSTAT) provides the receiver status and receive TDM time slot number. If the McASP logic attempts to set an interrupt flag in the same cycle that the CPU writes to the flag to clear it, the McASP logic has priority and the flag remains set. This also causes a new interrupt request to be generated. The RSTAT is shown in Figure 19-54 and described in Table 19-29.

Figure 19-54. Receiver Status Register (RSTAT)

31							9	8
Reserved (A)								RERR
R-0								R/W-0
7	6	5	4	3	2	1		0
RDMAERR	RSTAFRM	RDATA	RLAST	RTDMSLOT	RCKFAIL	RSYNCERR		ROVRN
R/W-0	R/W-0	R/W-0	R/W-0	R-0	R/W-0	R/W-0		R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

A If writing to this field, always write the default value for future device compatibility.

Table 19-29. Receiver Status Register (RSTAT) Field Descriptions

Bit	Field	Value	Description
31-9	Reserved	0	Reserved. The reserved bit location always returns the default value. A value written to this field has no effect. If writing to this field, always write the default value for future device compatibility.
8	RERR		RERR bit always returns a logic-OR of: ROVRN | RSYNCERR | RCKFAIL | RDMAERR Allows a single bit to be checked to determine if a receiver error interrupt has occurred.
		0	No errors have occurred.
		1	An error has occurred.
7	RDMAERR		Receive DMA error flag. RDMAERR is set when the CPU or DMA reads more serializers through the DMA port in a given time slot than were programmed as receivers. Causes a receive interrupt (RINT), if this bit is set and RDMAERR in RINTCTL is set. This bit is cleared by writing a 1 to this bit. Writing a 0 to this bit has no effect.
		0	Receive DMA error did not occur.
		1	Receive DMA error did occur.
6	RSTAFRM		Receive start of frame flag. Causes a receive interrupt (RINT), if this bit is set and RSTAFRM in RINTCTL is set. This bit is cleared by writing a 1 to this bit. Writing a 0 to this bit has no effect.
		0	No new receive frame sync (AFSR) is detected.
		1	A new receive frame sync (AFSR) is detected.
5	RDATA		Receive data ready flag. Causes a receive interrupt (RINT), if this bit is set and RDATA in RINTCTL is set. This bit is cleared by writing a 1 to this bit. Writing a 0 to this bit has no effect.
		0	No new data in RBUF.
		1	Data is transferred from XRSR to RBUF and ready to be serviced by the CPU or DMA. When RDATA is set, it always causes a DMA event (AREVT).
4	RLAST		Receive last slot flag. RLAST is set along with RDATA, if the current slot is the last slot in a frame. Causes a receive interrupt (RINT), if this bit is set and RLAST in RINTCTL is set. This bit is cleared by writing a 1 to this bit. Writing a 0 to this bit has no effect.
		0	Current slot is not the last slot in a frame.
		1	Current slot is the last slot in a frame. RDATA is also set.
3	RTDMSLOT		Returns the LSB of RSLOT. Allows a single read of RSTAT to determine whether the current TDM time slot is even or odd.
		0	Current TDM time slot is odd.
		1	Current TDM time slot is even.
2	RCKFAIL		Receive clock failure flag. RCKFAIL is set when the receive clock failure detection circuit reports an error (see Section 19.0.27.6.6). Causes a receive interrupt (RINT), if this bit is set and RCKFAIL in RINTCTL is set. This bit is cleared by writing a 1 to this bit. Writing a 0 to this bit has no effect.
		0	Receive clock failure did not occur.
		1	Receive clock failure did occur.

Copyright © 2013–2016, Texas Instruments Incorporated

Table 19-29. Receiver Status Register (RSTAT) Field Descriptions (continued)

Bit	Field	Value	Description
1	RSYNCERR		Unexpected receive frame sync flag. RSYNCERR is set when a new receive frame sync (AFSR) occurs before it is expected. Causes a receive interrupt (RINT), if this bit is set and RSYNCERR in RINTCTL is set. This bit is cleared by writing a 1 to this bit. Writing a 0 to this bit has no effect.
		0	Unexpected receive frame sync did not occur.
		1	Unexpected receive frame sync did occur.
0	ROVRN		Receiver overrun flag. ROVRN is set when the receive serializer is instructed to transfer data from XRSR to RBUF, but the former data in RBUF has not yet been read by the CPU or DMA. Causes a receive interrupt (RINT), if this bit is set and ROVRN in RINTCTL is set. This bit is cleared by writing a 1 to this bit. Writing a 0 to this bit has no effect.
		0	Receiver overrun did not occur.
		1	Receiver overrun did occur.

19.1.22 Current Receive TDM Time Slot Registers (RSLOT)

The current receive TDM time slot register (RSLOT) indicates the current time slot for the receive data frame. The RSLOT is shown in Figure 19-55 and described in Table 19-30.

Figure 19-55. Current Receive TDM Time Slot Registers (RSLOT)

31	16
Reserved (A)	
R-0	

15	9	8	0
Reserved (A)		RSLOTCNT	
R-0		R-0	

LEGEND: R = Read only; -*n* = value after reset

A If writing to this field, always write the default value for future device compatibility.

Table 19-30. Current Receive TDM Time Slot Registers (RSLOT) Field Descriptions

Bit	Field	Value	Description
31-9	Reserved	0	Reserved. The reserved bit location always returns the default value. A value written to this field has no effect. If writing to this field, always write the default value for future device compatibility.
8-0	RSLOTCNT	0-17Fh	Current receive time slot count. Legal values: 0 to 383 (17Fh). TDM function is not supported for > 32 time slots. However, TDM time slot counter may count to 383 when used to receive a DIR block (transferred over TDM format).

Copyright © 2013–2016, Texas Instruments Incorporated

19.1.23 Receive Clock Check Control Register (RCLKCHK)

The receive clock check control register (RCLKCHK) configures the receive clock failure detection circuit. The RCLKCHK is shown in Figure 19-56 and described in Table 19-31.

Figure 19-56. Receive Clock Check Control Register (RCLKCHK)

31	24	23	16
RCNT		RMAX	
R-0		R/W-0	

15	8	7	4	3	0
RMIN		Reserved (A)		RPS	
R/W-0		R-0		R/W-0	

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

A If writing to this field, always write the default value for future device compatibility.

Table 19-31. Receive Clock Check Control Register (RCLKCHK) Field Descriptions

Bit	Field	Value	Description
31-24	RCNT	0-FFh	Receive clock count value (from previous measurement). The clock circuit continually counts the number of DSP system clocks for every 32 receive high-frequency master clock (AHCLKR) signals, and stores the count in RCNT until the next measurement is taken.
23-16	RMAX	0-FFh	Receive clock maximum boundary. This 8-bit unsigned value sets the maximum allowed boundary for the clock check counter after 32 receive high-frequency master clock (AHCLKR) signals have been received. If the current counter value is greater than RMAX after counting 32 AHCLKR signals, RCKFAIL in RSTAT is set. The comparison is performed using unsigned arithmetic.
15-8	RMIN	0-FFh	Receive clock minimum boundary. This 8-bit unsigned value sets the minimum allowed boundary for the clock check counter after 32 receive high-frequency master clock (AHCLKR) signals have been received. If RCNT is less than RMIN after counting 32 AHCLKR signals, RCKFAIL in RSTAT is set. The comparison is performed using unsigned arithmetic.
7-4	Reserved	0	Reserved. The reserved bit location always returns the default value. A value written to this field has no effect. If writing to this field, always write the default value for future device compatibility.
3-0	RPS	0-Fh	Receive clock check prescaler value.
		0	McASP system clock divided by 1
		1h	McASP system clock divided by 2
		2h	McASP system clock divided by 4
		3h	McASP system clock divided by 8
		4h	McASP system clock divided by 16
		5h	McASP system clock divided by 32
		6h	McASP system clock divided by 64
		7h	McASP system clock divided by 128
		8h	McASP system clock divided by 256
		9h-Fh	Reserved

Copyright © 2013–2016, Texas Instruments Incorporated

19.1.24 Receiver DMA Event Control Register (REVTCTL)

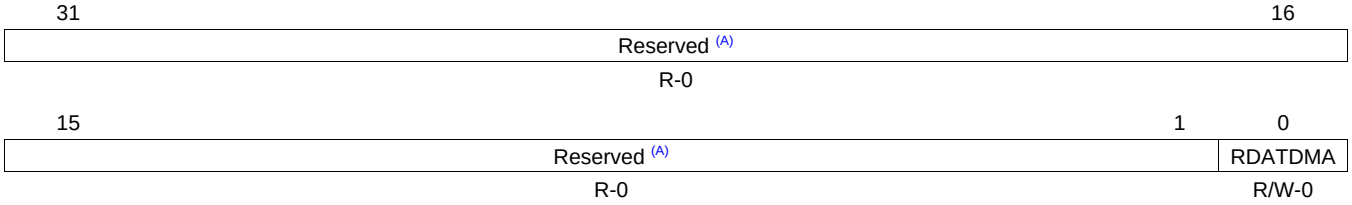
The receiver DMA event control register (REVTCTL) is shown in Figure 19-57 and described in Table 19-32.

CAUTION

DSP specific registers

Accessing REVTCTL not implemented on a specific DSP may cause improper device operation.

Figure 19-57. Receiver DMA Event Control Register (REVTCTL)



LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

A If writing to this field, always write the default value for future device compatibility.

Table 19-32. Receiver DMA Event Control Register (REVTCTL) Field Descriptions

Bit	Field	Value	Description
31-1	Reserved	0	Reserved. The reserved bit location always returns the default value. A value written to this field has no effect. If writing to this field, always write the default value for future device compatibility.
0	RDATDMA		Receive data DMA request enable bit. If writing to this field, always write the default value of 0.
		0	Receive data DMA request is enabled.
		1	Reserved.

Copyright © 2013–2016, Texas Instruments Incorporated

19.1.25 Transmitter Global Control Register (XGBLCTL)

Alias of the global control register (GBLCTL). Writing to the transmitter global control register (XGBLCTL) affects only the transmit bits of GBLCTL (bits 12-8). Reads from XGBLCTL return the value of GBLCTL. XGBLCTL allows the transmitter to be reset independently from the receiver. The XGBLCTL is shown in Figure 19-58 and described in Table 19-33. See Section 19.1.9 for a detailed description of GBLCTL.

Figure 19-58. Transmitter Global Control Register (XGBLCTL)

31	16
Reserved (A)	
R-0	

15	13	12	11	10	9	8
Reserved (A)		XFRST	XSMRST	XSRCLR	XHCLKRST	XCLKRST
R-0		R/W-0	R/W-0	R/W-0	R/W-0	R/W-0

7	5	4	3	2	1	0
Reserved (A)		RFRST	RSMRST	RSRCLR	RHCLKRST	RCLKRST
R-0		R-0	R-0	R-0	R-0	R-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

A If writing to this field, always write the default value for future device compatibility.

Table 19-33. Transmitter Global Control Register (XGBLCTL) Field Descriptions

Bit	Field	Value	Description
31-13	Reserved	0-FFh	Reserved. The reserved bit location always returns the default value. A value written to this field has no effect. If writing to this field, always write the default value for future device compatibility.
12	XFRST		Transmit frame sync generator reset enable bit. A write to this bit affects the XFRST bit of GBLCTL.
		0	Transmit frame sync generator is reset.
		1	Transmit frame sync generator is active.
11	XSMRST		Transmit state machine reset enable bit. A write to this bit affects the XSMRST bit of GBLCTL.
		0	Transmit state machine is held in reset.
		1	Transmit state machine is released from reset.
10	XSRCLR		Transmit serializer clear enable bit. A write to this bit affects the XSRCLR bit of GBLCTL.
		0	Transmit serializers are cleared.
		1	Transmit serializers are active.
9	XHCLKRST		Transmit high-frequency clock divider reset enable bit. A write to this bit affects the XHCLKRST bit of GBLCTL.
		0	Transmit high-frequency clock divider is held in reset.
		1	Transmit high-frequency clock divider is running.
8	XCLKRST		Transmit clock divider reset enable bit. A write to this bit affects the XCLKRST bit of GBLCTL.
		0	Transmit clock divider is held in reset.
		1	Transmit clock divider is running.
7-5	Reserved	0	Reserved. The reserved bit location always returns the default value. A value written to this field has no effect. If writing to this field, always write the default value for future device compatibility.
4	RFRST	x	Receive frame sync generator reset enable bit. A read of this bit returns the RFRST bit value of GBLCTL. Writes have no effect.
3	RSMRST	x	Receive state machine reset enable bit. A read of this bit returns the RSMRST bit value of GBLCTL. Writes have no effect.
2	RSRCLR	x	Receive serializer clear enable bit. A read of this bit returns the RSRSCLR bit value of GBLCTL. Writes have no effect.
1	RHCLKRST	x	Receive high-frequency clock divider reset enable bit. A read of this bit returns the RHCLKRST bit value of GBLCTL. Writes have no effect.
0	RCLKRST	x	Receive clock divider reset enable bit. A read of this bit returns the RCLKRST bit value of GBLCTL. Writes have no effect.

Copyright © 2013–2016, Texas Instruments Incorporated

19.1.26 Transmit Format Unit Bit Mask Register (XMASK)

The transmit format unit bit mask register (XMASK) determines which bits of the transmitted data are masked off and padded with a known value before being shifted out the McASP. The XMASK is shown in Figure 19-59 and described in Table 19-34.

Figure 19-59. Transmit Format Unit Bit Mask Register (XMASK)

31	30	29	28	27	26	25	24
XMASK31	XMASK30	XMASK29	XMASK28	XMASK27	XMASK26	XMASK25	XMASK24
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
23	22	21	20	19	18	17	16
XMASK23	XMASK22	XMASK21	XMASK20	XMASK19	XMASK18	XMASK17	XMASK16
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
15	14	13	12	11	10	9	8
XMASK15	XMASK14	XMASK13	XMASK12	XMASK11	XMASK10	XMASK9	XMASK8
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
7	6	5	4	3	2	1	0
XMASK7	XMASK6	XMASK5	XMASK4	XMASK3	XMASK2	XMASK1	XMASK0
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0

LEGEND: R/W = Read/Write; -*n* = value after reset

Table 19-34. Transmit Format Unit Bit Mask Register (XMASK) Field Descriptions

Bit	Field	Value	Description
31-0	XMASK[31-0]		Transmit data mask enable bit.
		0	Corresponding bit of transmit data (before passing through reverse and rotate units) is masked out and then padded with the selected bit pad value (XPAD and XPBIT bits in XFMT), which is transmitted out the McASP in place of the original bit.
		1	Corresponding bit of transmit data (before passing through reverse and rotate units) is transmitted out the McASP.

Copyright © 2013–2016, Texas Instruments Incorporated

19.1.27 Transmit Bit Stream Format Register (XFMT)

The transmit bit stream format register (XFMT) configures the transmit data format. The XFMT is shown in Figure 19-60 and described in Table 19-35.

Figure 19-60. Transmit Bit Stream Format Register (XFMT)

31																18	17	16
Reserved [A]																	XDATDLY	
R-0																	R/W-0	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
XRVRS	XPAD		XPBIT					XSSZ				XBUSEL	XROT		
R/W-0	R/W-0		R/W-0					R/W-0				R/W-0	R/W-0		

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

A If writing to this field, always write the default value for future device compatibility.

Table 19-35. Transmit Bit Stream Format Register (XFMT) Field Descriptions

Bit	Field	Value	Description
31-18	Reserved	0	Reserved. The reserved bit location always returns the default value. A value written to this field has no effect. If writing to this field, always write the default value for future device compatibility.
17-16	XDATDLY	0-3h	Transmit sync bit delay.
		0	0-bit delay. The first transmit data bit, AXR[n], occurs in same ACLKX cycle as the transmit frame sync (AFSX).
		1h	1-bit delay. The first transmit data bit, AXR[n], occurs one ACLKX cycle after the transmit frame sync (AFSX).
		2h	2-bit delay. The first transmit data bit, AXR[n], occurs two ACLKX cycles after the transmit frame sync (AFSX).
		3h	Reserved.
15	XRVRS		Transmit serial bitstream order.
		0	Bitstream is LSB first. No bit reversal is performed in transmit format bit reverse unit.
		1	Bitstream is MSB first. Bit reversal is performed in transmit format bit reverse unit.
14-13	XPAD	0-3h	Pad value for extra bits in slot not belonging to word defined by XMASK. This field only applies to bits when XMASK[n] = 0.
		0	Pad extra bits with 0.
		1h	Pad extra bits with 1.
		2h	Pad extra bits with one of the bits from the word as specified by XPBIT bits.
		3h	Reserved
12-8	XPBIT	0-1Fh	XPBIT value determines which bit (as written by the CPU or DMA to XBUF[n]) is used to pad the extra bits before shifting. This field only applies when XPAD = 2h.
		0	Pad with bit 0 value.
		1-1Fh	Pad with bit 1 to bit 31 value.

Copyright © 2013–2016, Texas Instruments Incorporated

Table 19-35. Transmit Bit Stream Format Register (XFMT) Field Descriptions (continued)

Bit	Field	Value	Description
7-4	XSSZ	0-Fh	Transmit slot size.
		0-2h	Reserved
		3h	Slot size is 8 bits.
		4h	Reserved
		5h	Slot size is 12 bits.
		6h	Reserved.
		7h	Slot size is 16 bits.
		8h	Reserved.
		9h	Slot size is 20 bits.
		Ah	Reserved.
		Bh	Slot size is 24 bits.
		Ch	Reserved.
		Dh	Slot size is 28 bits.
		Eh	Reserved.
		Fh	Slot size is 32 bits.
3	XBUSEL		Selects whether writes to serializer buffer XRBUF[n] by way of XBUF*n* by the CPU/EDMA occur through the peripheral configuration port or the DMA port.
		0	Writes to XRBUF[n] originate from the DMA port. Writes to XRBUF[n] from the peripheral configuration port are ignored with no effect to the McASP.
		1	Writes to XRBUF[n] originate from the peripheral configuration port. Writes to XRBUF[n] from the DMA port are ignored with no effect to the McASP.
2-0	XROT	0-7h	Right-rotation value for transmit rotate right format unit.
		0	Rotate right by 0 (no rotation).
		1h	Rotate right by 4 bit positions.
		2h	Rotate right by 8 bit positions.
		3h	Rotate right by 12 bit positions.
		4h	Rotate right by 16 bit positions.
		5h	Rotate right by 20 bit positions.
		6h	Rotate right by 24 bit positions.
		7h	Rotate right by 28 bit positions.

Copyright © 2013–2016, Texas Instruments Incorporated

19.1.28 Transmit Frame Sync Control Register (AFSXCTL)

The transmit frame sync control register (AFSXCTL) configures the transmit frame sync (AFSX). The AFSXCTL is shown in Figure 19-61 and described in Table 19-36.

Figure 19-61. Transmit Frame Sync Control Register (AFSXCTL)

31	16
Reserved [A]	
R-0	

15	7	6	5	4	3	2	1	0
XMOD		Reserved [A]		FXWID	Reserved [A]		FSXM	FSXP
R/W-0		R-0		R/W-0	R-0		R/W-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

A If writing to this field, always write the default value for future device compatibility.

Table 19-36. Transmit Frame Sync Control Register (AFSXCTL) Field Descriptions

Bit	Field	Value	Description
31-16	Reserved	0	Reserved. The reserved bit location always returns the default value. A value written to this field has no effect. If writing to this field, always write the default value for future device compatibility.
15-7	XMOD	0-1FFh	Transmit frame sync mode select bits.
		0	Burst mode
		1h	Reserved
		2h-20h	2-slot TDM (I2S mode) to 32-slot TDM
		21h-17Fh	Reserved
		180h	384-slot DIT mode
		181h-1FFh	Reserved
6-5	Reserved	0	Reserved. The reserved bit location always returns the default value. A value written to this field has no effect. If writing to this field, always write the default value for future device compatibility.
4	FXWID		Transmit frame sync width select bit indicates the width of the transmit frame sync (AFSX) during its active period.
		0	Single bit
		1	Single word
3-2	Reserved	0	Reserved. The reserved bit location always returns the default value. A value written to this field has no effect. If writing to this field, always write the default value for future device compatibility.
1	FSXM		Transmit frame sync generation select bit.
		0	Externally-generated transmit frame sync
		1	Internally-generated transmit frame sync
0	FSXP		Transmit frame sync polarity select bit.
		0	A rising edge on transmit frame sync (AFSX) indicates the beginning of a frame.
		1	A falling edge on transmit frame sync (AFSX) indicates the beginning of a frame.

Copyright © 2013–2016, Texas Instruments Incorporated

19.1.29 Transmit Clock Control Register (ACLKXCTL)

The transmit clock control register (ACLKXCTL) configures the transmit bit clock (ACLKX) and the transmit clock generator. The ACLKXCTL is shown in Figure 19-62 and described in Table 19-37.

Figure 19-62. Transmit Clock Control Register (ACLKXCTL)

31	16
Reserved (A)	
R-0	

15	8	7	6	5	4	0
Reserved (A)		CLKXP	ASYNC	CLKXM	CLKXDIV	
R-0		R/W-0	R/W-1	R/W-1	R/W-0	

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

A If writing to this field, always write the default value for future device compatibility.

Table 19-37. Transmit Clock Control Register (ACLKXCTL) Field Descriptions

Bit	Field	Value	Description
31-8	Reserved	0	Reserved. The reserved bit location always returns the default value. A value written to this field has no effect. If writing to this field, always write the default value for future device compatibility.
7	CLKXP		Transmit bitstream clock polarity select bit.
		0	Rising edge. External receiver samples data on the falling edge of the serial clock, so the transmitter must shift data out on the rising edge of the serial clock.
		1	Falling edge. External receiver samples data on the rising edge of the serial clock, so the transmitter must shift data out on the falling edge of the serial clock.
6	ASYNC		Transmit/receive operation asynchronous enable bit.
		0	Synchronous. Transmit clock and frame sync provides the source for both the transmit and receive sections. Note that in this mode, the receive bit clock is an inverted version of the transmit bit clock. See Section 19.0.27.1.5 for more details.
		1	Asynchronous. Separate clock and frame sync used by transmit and receive sections.
5	CLKXM		Transmit bit clock source bit.
		0	External transmit clock source from ACLKX pin.
		1	Internal transmit clock source from output of programmable bit clock divider.
4-0	CLKXDIV	0-1Fh	Transmit bit clock divide ratio bits determine the divide-down ratio from AHCLKX to ACLKX.
		0	Divide-by-1
		1h	Divide-by-2
		2h-1Fh	Divide-by-3 to divide-by-32

Copyright © 2013–2016, Texas Instruments Incorporated

19.1.30 Transmit High-Frequency Clock Control Register (AHCLKXCTL)

The transmit high-frequency clock control register (AHCLKXCTL) configures the transmit high-frequency master clock (AHCLKX) and the transmit clock generator. The AHCLKXCTL is shown in Figure 19-63 and described in Table 19-38.

Figure 19-63. Transmit High-Frequency Clock Control Register (AHCLKXCTL)

31	16
Reserved (A)	
R-0	

15	14	13	12	11	0
HCLKXM	HCLKXP	Reserved (A)		HCLKXDIV	
R/W-1	R/W-0	R/W-0		R/W-0	

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

A If writing to this field, always write the default value for future device compatibility.

Table 19-38. Transmit High-Frequency Clock Control Register (AHCLKXCTL) Field Descriptions

Bit	Field	Value	Description
31-16	Reserved	0	Reserved. The reserved bit location always returns the default value. A value written to this field has no effect. If writing to this field, always write the default value for future device compatibility.
15	HCLKXM		Transmit high-frequency clock source bit.
		0	External transmit high-frequency clock source from AHCLKX pin.
		1	Internal transmit high-frequency clock source from output of programmable high clock divider.
14	HCLKXP		Transmit bitstream high-frequency clock polarity select bit.
		0	Not inverted. AHCLKX is not inverted before programmable bit clock divider. In the special case where the transmit bit clock (ACLKX) is internally generated and the programmable bit clock divider is set to divide-by-1 (CLKXDIV = 0 in ACLKXCTL), AHCLKX is directly passed through to the ACLKX pin.
		1	Inverted. AHCLKX is inverted before programmable bit clock divider. In the special case where the transmit bit clock (ACLKX) is internally generated and the programmable bit clock divider is set to divide-by-1 (CLKXDIV = 0 in ACLKXCTL), AHCLKX is directly passed through to the ACLKX pin.
13-12	Reserved	0	Reserved. The reserved bit location always returns the default value. A value written to this field has no effect. If writing to this field, always write the default value for future device compatibility.
11-0	HCLKXDIV	0-FFFh	Transmit high-frequency clock divide ratio bits determine the divide-down ratio from AUXCLK to AHCLKX.
		0	Divide-by-1
		1h	Divide-by-2
		2h-FFFh	Divide-by-3 to divide-by-4096

Copyright © 2013–2016, Texas Instruments Incorporated

19.1.31 Transmit TDM Time Slot Register (XTDM)

The transmit TDM time slot register (XTDM) specifies in which TDM time slot the transmitter is active. TDM time slot counter range is extended to 384 slots (to support SPDIF blocks of 384 subframes). XTDM operates modulo 32, that is, XTDMS specifies the TDM activity for time slots 0, 32, 64, 96, 128, etc. The XTDM is shown in Figure 19-64 and described in Table 19-39.

Figure 19-64. Transmit TDM Time Slot Register (XTDM)

31	30	29	28	27	26	25	24
XTDMS31	XTDMS30	XTDMS29	XTDMS28	XTDMS27	XTDMS26	XTDMS25	XTDMS24
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
23	22	21	20	19	18	17	16
XTDMS23	XTDMS22	XTDMS21	XTDMS20	XTDMS19	XTDMS18	XTDMS17	XTDMS16
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
15	14	13	12	11	10	9	8
XTDMS15	XTDMS14	XTDMS13	XTDMS12	XTDMS11	XTDMS10	XTDMS9	XTDMS8
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
7	6	5	4	3	2	1	0
XTDMS7	XTDMS6	XTDMS5	XTDMS4	XTDMS3	XTDMS2	XTDMS1	XTDMS0
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0

LEGEND: R/W = Read/Write; -*n* = value after reset

Table 19-39. Transmit TDM Time Slot Register (XTDM) Field Descriptions

Bit	Field	Value	Description
31-0	XTDMS[31-0]		Transmitter mode during TDM time slot *n*.
		0	Transmit TDM time slot *n* is inactive. The transmit serializer does not shift out data during this slot.
		1	Transmit TDM time slot *n* is active. The transmit serializer shifts out data during this slot according to the serializer control register (SRCTL).

Copyright © 2013–2016, Texas Instruments Incorporated

19.1.32 Transmitter Interrupt Control Register (XINTCTL)

The transmitter interrupt control register (XINTCTL) controls generation of the McASP transmit interrupt (XINT). When the register bit(s) is set to 1, the occurrence of the enabled McASP condition(s) generates XINT. The XINTCTL is shown in Figure 19-65 and described in Table 19-40. See Section 19.1.33 for a description of the interrupt conditions.

Figure 19-65. Transmitter Interrupt Control Register (XINTCTL)

31							8
Reserved (A)							
R-0							
7	6	5	4	3	2	1	0
XSTAFRM	Reserved (A)	XDATA	XLAST	XDMAERR	XCKFAIL	XSYNCERR	XUNDRN
R/W-0	R-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

A If writing to this field, always write the default value for future device compatibility.

Table 19-40. Transmitter Interrupt Control Register (XINTCTL) Field Descriptions

Bit	Field	Value	Description
31-8	Reserved	0	Reserved. The reserved bit location always returns the default value. A value written to this field has no effect. If writing to this field, always write the default value for future device compatibility.
7	XSTAFRM		Transmit start of frame interrupt enable bit.
		0	Interrupt is disabled. A transmit start of frame interrupt does not generate a McASP transmit interrupt (XINT).
		1	Interrupt is enabled. A transmit start of frame interrupt generates a McASP transmit interrupt (XINT).
6	Reserved	0	Reserved. The reserved bit location always returns the default value. A value written to this field has no effect. If writing to this field, always write the default value for future device compatibility.
5	XDATA		Transmit data ready interrupt enable bit.
		0	Interrupt is disabled. A transmit data ready interrupt does not generate a McASP transmit interrupt (XINT).
		1	Interrupt is enabled. A transmit data ready interrupt generates a McASP transmit interrupt (XINT).
4	XLAST		Transmit last slot interrupt enable bit.
		0	Interrupt is disabled. A transmit last slot interrupt does not generate a McASP transmit interrupt (XINT).
		1	Interrupt is enabled. A transmit last slot interrupt generates a McASP transmit interrupt (XINT).
3	XDMAERR		Transmit DMA error interrupt enable bit.
		0	Interrupt is disabled. A transmit DMA error interrupt does not generate a McASP transmit interrupt (XINT).
		1	Interrupt is enabled. A transmit DMA error interrupt generates a McASP transmit interrupt (XINT).
2	XCKFAIL		Transmit clock failure interrupt enable bit.
		0	Interrupt is disabled. A transmit clock failure interrupt does not generate a McASP transmit interrupt (XINT).
		1	Interrupt is enabled. A transmit clock failure interrupt generates a McASP transmit interrupt (XINT).
1	XSYNCERR		Unexpected transmit frame sync interrupt enable bit.
		0	Interrupt is disabled. An unexpected transmit frame sync interrupt does not generate a McASP transmit interrupt (XINT).
		1	Interrupt is enabled. An unexpected transmit frame sync interrupt generates a McASP transmit interrupt (XINT).
0	XUNDRN		Transmitter underrun interrupt enable bit.
		0	Interrupt is disabled. A transmitter underrun interrupt does not generate a McASP transmit interrupt (XINT).
		1	Interrupt is enabled. A transmitter underrun interrupt generates a McASP transmit interrupt (XINT).

Copyright © 2013–2016, Texas Instruments Incorporated

19.1.33 Transmitter Status Register (XSTAT)

The transmitter status register (XSTAT) provides the transmitter status and transmit TDM time slot number. If the McASP logic attempts to set an interrupt flag in the same cycle that the CPU writes to the flag to clear it, the McASP logic has priority and the flag remains set. This also causes a new interrupt request to be generated. The XSTAT is shown in Figure 19-66 and described in Table 19-41.

Figure 19-66. Transmitter Status Register (XSTAT)

31							9	8
Reserved (A)								XERR
R-0								R/W-0
7	6	5	4	3	2	1		0
XDMAERR	XSTAFRM	XDATA	XLAST	XTDMSLOT	XCKFAIL	XSYNCERR		XUNDRN
R/W-0	R/W-0	R/W-0	R/W-0	R-0	R/W-0	R/W-0		R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

A If writing to this field, always write the default value for future device compatibility.

Table 19-41. Transmitter Status Register (XSTAT) Field Descriptions

Bit	Field	Value	Description
31-9	Reserved	0	Reserved. The reserved bit location always returns the default value. A value written to this field has no effect. If writing to this field, always write the default value for future device compatibility.
8	XERR		XERR bit always returns a logic-OR of: XUNDRN | XSYNCERR | XCKFAIL | XDMAERR Allows a single bit to be checked to determine if a transmitter error interrupt has occurred.
		0	No errors have occurred.
		1	An error has occurred.
7	XDMAERR		Transmit DMA error flag. XDMAERR is set when the CPU or DMA writes more serializers through the DMA port in a given time slot than were programmed as transmitters. Causes a transmit interrupt (XINT), if this bit is set and XDMAERR in XINTCTL is set. This bit is cleared by writing a 1 to this bit. Writing a 0 has no effect.
		0	Transmit DMA error did not occur.
		1	Transmit DMA error did occur.
6	XSTAFRM		Transmit start of frame flag. Causes a transmit interrupt (XINT), if this bit is set and XSTAFRM in XINTCTL is set. This bit is cleared by writing a 1 to this bit. Writing a 0 has no effect.
		0	No new transmit frame sync (AFSX) is detected.
		1	A new transmit frame sync (AFSX) is detected.
5	XDATA		Transmit data ready flag. Causes a transmit interrupt (XINT), if this bit is set and XDATA in XINTCTL is set. This bit is cleared by writing a 1 to this bit. Writing a 0 has no effect.
		0	XBUF is written and is full.
		1	Data is copied from XBUF to XRSR. XBUF is empty and ready to be written. XDATA is also set when the transmit serializers are taken out of reset. When XDATA is set, it always causes a DMA event (AXEVT).
4	XLAST		Transmit last slot flag. XLAST is set along with XDATA, if the current slot is the last slot in a frame. Causes a transmit interrupt (XINT), if this bit is set and XLAST in XINTCTL is set. This bit is cleared by writing a 1 to this bit. Writing a 0 has no effect.
		0	Current slot is not the last slot in a frame.
		1	Current slot is the last slot in a frame. XDATA is also set.
3	XTDMSLOT		Returns the LSB of XSLOT. Allows a single read of XSTAT to determine whether the current TDM time slot is even or odd.
		0	Current TDM time slot is odd.
		1	Current TDM time slot is even.
2	XCKFAIL		Transmit clock failure flag. XCKFAIL is set when the transmit clock failure detection circuit reports an error (see Section 19.0.27.6.6). Causes a transmit interrupt (XINT), if this bit is set and XCKFAIL in XINTCTL is set. This bit is cleared by writing a 1 to this bit. Writing a 0 has no effect.
		0	Transmit clock failure did not occur.
		1	Transmit clock failure did occur.

Copyright © 2013–2016, Texas Instruments Incorporated

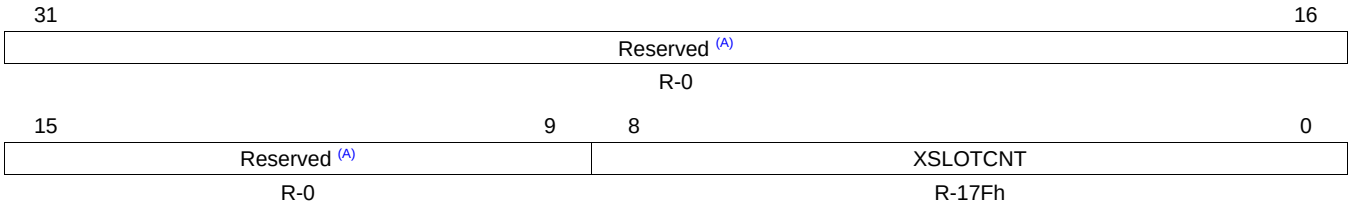
Table 19-41. Transmitter Status Register (XSTAT) Field Descriptions (continued)

Bit	Field	Value	Description
1	XSYNCERR		Unexpected transmit frame sync flag. XSYNCERR is set when a new transmit frame sync (AFSX) occurs before it is expected. Causes a transmit interrupt (XINT), if this bit is set and XSYNCERR in XINTCTL is set. This bit is cleared by writing a 1 to this bit. Writing a 0 has no effect.
		0	Unexpected transmit frame sync did not occur.
		1	Unexpected transmit frame sync did occur.
0	XUNDRN		Transmitter underrun flag. XUNDRN is set when the transmit serializer is instructed to transfer data from XBUF to XRSR, but XBUF has not yet been serviced with new data since the last transfer. Causes a transmit interrupt (XINT), if this bit is set and XUNDRN in XINTCTL is set. This bit is cleared by writing a 1 to this bit. Writing a 0 has no effect.
		0	Transmitter underrun did not occur.
		1	Transmitter underrun did occur. See Section 19.0.27.6.2 for details on McASP action upon underrun conditions.

19.1.34 Current Transmit TDM Time Slot Register (XSLOT)

The current transmit TDM time slot register (XSLOT) indicates the current time slot for the transmit data frame. The XSLOT is shown in Figure 19-67 and described in Table 19-42.

Figure 19-67. Current Transmit TDM Time Slot Register (XSLOT)



LEGEND: R = Read only; -*n* = value after reset

A If writing to this field, always write the default value for future device compatibility.

Table 19-42. Current Transmit TDM Time Slot Register (XSLOT) Field Descriptions

Bit	Field	Value	Description
31-9	Reserved	0	Reserved. The reserved bit location always returns the default value. A value written to this field has no effect. If writing to this field, always write the default value for future device compatibility.
8-0	XSLOTCNT	0-17Fh	Current transmit time slot count. Legal values: 0 to 383 (17Fh). During reset, this counter value is 383 so the next count value, which is used to encode the first DIT group of data, will be 0 and encodes the B preamble. TDM function is not supported for > 32 time slots. However, TDM time slot counter may count to 383 when used to transmit a DIT block.

Copyright © 2013–2016, Texas Instruments Incorporated

19.1.35 Transmit Clock Check Control Register (XCLKCHK)

The transmit clock check control register (XCLKCHK) configures the transmit clock failure detection circuit. The XCLKCHK is shown in Figure 19-68 and described in Table 19-43.

Figure 19-68. Transmit Clock Check Control Register (XCLKCHK)

31	24	23	16
XCNT		XMAX	
R-0		R/W-0	

15	8	7	4	3	0
XMIN		Reserved (A)		XPS	
R/W-0		R-0		R/W-0	

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

A If writing to this field, always write the default value for future device compatibility.

Table 19-43. Transmit Clock Check Control Register (XCLKCHK) Field Descriptions

Bit	Field	Value	Description
31-24	XCNT	0	Transmit clock count value (from previous measurement). The clock circuit continually counts the number of DSP system clocks for every 32 transmit high-frequency master clock (AHCLKX) signals, and stores the count in XCNT until the next measurement is taken.
23-16	XMAX	0-FFh	Transmit clock maximum boundary. This 8-bit unsigned value sets the maximum allowed boundary for the clock check counter after 32 transmit high-frequency master clock (AHCLKX) signals have been received. If the current counter value is greater than XMAX after counting 32 AHCLKX signals, XCKFAIL in XSTAT is set. The comparison is performed using unsigned arithmetic.
15-8	XMIN	0-FFh	Transmit clock minimum boundary. This 8-bit unsigned value sets the minimum allowed boundary for the clock check counter after 32 transmit high-frequency master clock (AHCLKX) signals have been received. If XCNT is less than XMIN after counting 32 AHCLKX signals, XCKFAIL in XSTAT is set. The comparison is performed using unsigned arithmetic.
7-4	Reserved	0	Reserved. The reserved bit location always returns the default value. A value written to this field has no effect. If writing to this field, always write the default value for future device compatibility.
3-0	XPS	0-Fh	Transmit clock check prescaler value.
		0	McASP system clock divided by 1
		1h	McASP system clock divided by 2
		2h	McASP system clock divided by 4
		3h	McASP system clock divided by 8
		4h	McASP system clock divided by 16
		5h	McASP system clock divided by 32
		6h	McASP system clock divided by 64
		7h	McASP system clock divided by 128
		8h	McASP system clock divided by 256
		9h-Fh	Reserved

Copyright © 2013–2016, Texas Instruments Incorporated

19.1.36 Transmitter DMA Event Control Register (XEVTCTL)

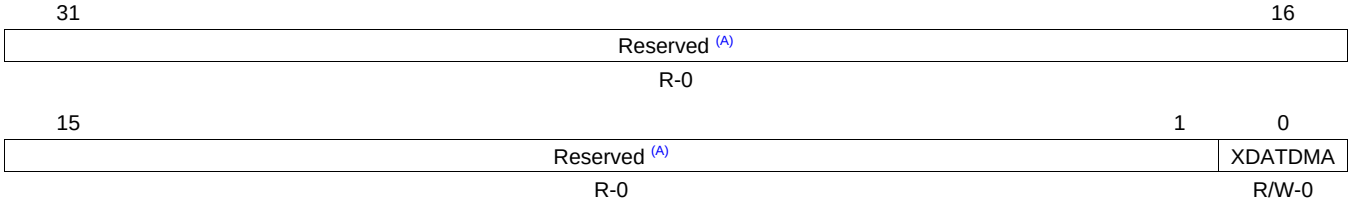
The transmitter DMA event control register (XEVTCTL) is shown in Figure 19-69 and described in Table 19-44.

CAUTION

DSP specific registers

Accessing XEVTCTL not implemented on a specific DSP may cause improper device operation.

Figure 19-69. Transmitter DMA Event Control Register (XEVTCTL)



LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

A If writing to this field, always write the default value for future device compatibility.

Table 19-44. Transmitter DMA Event Control Register (XEVTCTL) Field Descriptions

Bit	Field	Value	Description
31-1	Reserved	0	Reserved. The reserved bit location always returns the default value. A value written to this field has no effect. If writing to this field, always write the default value for future device compatibility.
0	XDATDMA		Transmit data DMA request enable bit. If writing to this field, always write the default value of 0.
		0	Transmit data DMA request is enabled.
		1	Reserved.

Copyright © 2013–2016, Texas Instruments Incorporated

19.1.37 Serializer Control Registers (SRCTL*n*)

Each serializer on the McASP has a serializer control register (SRCTL). There are up to 16 serializers per McASP. The SRCTL is shown in Figure 19-70 and described in Table 19-45.

CAUTION

DSP specific registers

Accessing SRCTL*n* not implemented on a specific DSP may cause improper device operation.

Figure 19-70. Serializer Control Registers (SRCTL*n*)

31							16
Reserved [A]							
R-0							

15	6	5	4	3	2	1	0
Reserved [A]		RRDY	XRDY	DISMOD		SRMOD	
R-0		R-0	R-0	R/W-0		R/W-0	

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

A If writing to this field, always write the default value for future device compatibility.

Table 19-45. Serializer Control Registers (SRCTL*n*) Field Descriptions

Bit	Field	Value	Description
31-6	Reserved	0	Reserved. The reserved bit location always returns the default value. A value written to this field has no effect. If writing to this field, always write the default value for future device compatibility.
5	RRDY		Receive buffer ready bit. RRDY indicates the current receive buffer state. Always reads 0 when programmed as a transmitter or as inactive. If SRMOD bit is set to receive (2h), RRDY switches from 0 to 1 whenever data is transferred from XRSR to RBUF.
		0	Receive buffer (RBUF) is empty.
		1	Receive buffer (RBUF) contains data and needs to be read before the start of the next time slot or a receiver overrun occurs.
4	XRDY		Transmit buffer ready bit. XRDY indicates the current transmit buffer state. Always reads 0 when programmed as a receiver or as inactive. If SRMOD bit is set to transmit (1h), XRDY switches from 0 to 1 when XSRCLR in GBLCTL is switched from 0 to 1 to indicate an empty transmitter. XRDY remains set until XSRCLR is forced to 0, data is written to the corresponding transmit buffer, or SRMOD bit is changed to receive (2h) or inactive (0).
		0	Transmit buffer (XBUF) contains data.
		1	Transmit buffer (XBUF) is empty and needs to be written before the start of the next time slot or a transmit underrun occurs.
3-2	DISMOD	0-3h	Serializer pin drive mode bit. Drive on pin when in inactive TDM slot of transmit mode or when serializer is inactive. This field only applies if the pin is configured as a McASP pin (PFUNC = 0).
		0	Drive on pin is 3-state.
		1h	Reserved
		2h	Drive on pin is logic low.
		3h	Drive on pin is logic high.
1-0	SRMOD	0-3h	Serializer mode bit.
		0	Serializer is inactive.
		1h	Serializer is transmitter.
		2h	Serializer is receiver.
		3h	Reserved

Copyright © 2013–2016, Texas Instruments Incorporated

19.1.38 DIT Left Channel Status Registers (DITCSRA0-DITCSRA5)

The DIT left channel status registers (DITCSRA) provide the status of each left channel (even TDM time slot). Each of the six 32-bit registers (Figure 19-71) can store 192 bits of channel status data for a complete block of transmission. The DIT reuses the same data for the next block. It is your responsibility to update the register file in time, if a different set of data need to be sent.

Figure 19-71. DIT Left Channel Status Registers (DITCSRA0-DITCSRA5)

31	0
DITCSRA*n*	
R/W-0	

LEGEND: R/W = Read/Write; -*n* = value after reset

19.1.39 DIT Right Channel Status Registers (DITCSRB0-DITCSRB5)

The DIT right channel status registers (DITCSRB) provide the status of each right channel (odd TDM time slot). Each of the six 32-bit registers (Figure 19-72) can store 192 bits of channel status data for a complete block of transmission. The DIT reuses the same data for the next block. It is your responsibility to update the register file in time, if a different set of data need to be sent.

Figure 19-72. DIT Right Channel Status Registers (DITCSRB0-DITCSRB5)

31	0
DITCSRB*n*	
R/W-0	

LEGEND: R/W = Read/Write; -*n* = value after reset

Copyright © 2013–2016, Texas Instruments Incorporated

19.1.40 DIT Left Channel User Data Registers (DITUDRA0-DITUDRA5)

The DIT left channel user data registers (DITUDRA) provides the user data of each left channel (even TDM time slot). Each of the six 32-bit registers (Figure 19-73) can store 192 bits of user data for a complete block of transmission. The DIT reuses the same data for the next block. It is your responsibility to update the register in time, if a different set of data need to be sent.

Figure 19-73. DIT Left Channel User Data Registers (DITUDRA0-DITUDRA5)

31	0
DITUDRA*n*	
R/W-0	

LEGEND: R/W = Read/Write; -*n* = value after reset

19.1.41 DIT Right Channel User Data Registers (DITUDRB0-DITUDRB5)

The DIT right channel user data registers (DITUDRB) provides the user data of each right channel (odd TDM time slot). Each of the six 32-bit registers (Figure 19-74) can store 192 bits of user data for a complete block of transmission. The DIT reuses the same data for the next block. It is your responsibility to update the register in time, if a different set of data need to be sent.

Figure 19-74. DIT Right Channel User Data Registers (DITUDRB0-DITUDRB5)

31	0
DITUDRB*n*	
R/W-0	

LEGEND: R/W = Read/Write; -*n* = value after reset

Copyright © 2013–2016, Texas Instruments Incorporated

19.1.42 Transmit Buffer Registers (XBUF*n*)

The transmit buffers for the serializers (XBUF) hold data from the transmit format unit. For transmit operations, the XBUF (Figure 19-75) is an alias of the XRBUF in the serializer. The XBUF can be accessed through the peripheral configuration port (Table 19-7) or through the DMA port (Table 19-8).

CAUTION

DSP specific registers

Accessing XBUF registers not implemented on a specific DSP may cause improper device operation.

Figure 19-75. Transmit Buffer Registers (XBUF*n*)

31	0
XBUF*n*	
R/W-0	

LEGEND: R/W = Read/Write; -*n* = value after reset

19.1.43 Receive Buffer Registers (RBUF*n*)

The receive buffers for the serializers (RBUF) hold data from the serializer before the data goes to the receive format unit. For receive operations, the RBUF (Figure 19-76) is an alias of the XRBUF in the serializer. The RBUF can be accessed through the peripheral configuration port (Table 19-7) or through the DMA port (Table 19-8).

CAUTION

DSP specific registers

Accessing RBUF registers not implemented on a specific DSP may cause improper device operation.

Figure 19-76. Receive Buffer Registers (RBUF*n*)

31	0
RBUF*n*	
R/W-0	

LEGEND: R/W = Read/Write; -*n* = value after reset

Copyright © 2013–2016, Texas Instruments Incorporated

19.1.44 AFIFO Revision Identification Register (AFIFOREV)

The Audio FIFO (AFIFO) revision identification register (AFIFOREV) contains revision data for the Audio FIFO (AFIFO). The AFIFOREV is shown in Figure 19-77 and described in Table 19-46.

Figure 19-77. AFIFO Revision Identification Register (AFIFOREV)

31	0
REV	
R-4431 1100h	

LEGEND: R = Read only; -*n* = value after reset

Table 19-46. AFIFO Revision Identification Register (AFIFOREV) Field Descriptions

Bit	Field	Value	Description
31-0	REV	4431 1100h	Identifies revision of Audio FIFO.

Copyright © 2013–2016, Texas Instruments Incorporated

19.1.45 Write FIFO Control Register (WFIFOCTL)

The Write FIFO control register (WFIFOCTL) is shown in Figure 19-78 and described in Table 19-47.

NOTE: The WNUMEVT and WNUMDMA values must be set prior to enabling the Write FIFO.

If the Write FIFO is to be enabled, it must be enabled prior to taking the McASP out of reset.

Figure 19-78. Write FIFO Control Register (WFIFOCTL)

31	17	16
Reserved		WENA
R-0		R/W-0

15	8	7	0
WNUMEVT		WNUMDMA	
R/W-10h		R/W-4h	

LEGEND: R/W = Read/Write; R = Read only; *-n* = value after reset

Table 19-47. Write FIFO Control Register (WFIFOCTL) Field Descriptions

Bit	Field	Value	Description
31-17	Reserved	0	Reserved
16	WENA		Write FIFO enable bit.
		0	Write FIFO is disabled. The WLVL field in the Write FIFO status register (WFIFOSTS) is reset to 0 and pointers are initialized, that is, the Write FIFO is "flushed."
		1	Write FIFO is enabled. If Write FIFO is to be enabled, it must be enabled prior to taking McASP out of reset.
15-8	WNUMEVT	0-FFh	Write word count per DMA event (32-bit). When the Write FIFO has space for at least WNUMEVT words of data, then an AXEVT (transmit DMA event) is generated to the host/DMA controller. This value should be set to a non-zero integer multiple of the number of serializers enabled as transmitters. This value must be set prior to enabling the Write FIFO.
		0	0 words
		1h	1 word
		2h	2 words
		3h-40h	3 to 64 words
		41h-FFh	Reserved
7-0	WNUMDMA	0-FFh	Write word count per transfer (32-bit words). Upon a transmit DMA event from the McASP, WNUMDMA words are transferred from the Write FIFO to the McASP. This value must equal the number of McASP serializers (not the number of channels) used as transmitters. This value must be set prior to enabling the Write FIFO.
		0	0 words
		1h	1 word
		2h	2 words
		3h-10h	3-16 words
		11h-FFh	Reserved

Copyright © 2013–2016, Texas Instruments Incorporated

19.1.46 Write FIFO Status Register (WFIFOSTS)

The Write FIFO status register (WFIFOSTS) is shown in Figure 19-79 and described in Table 19-48.

Figure 19-79. Write FIFO Status Register (WFIFOSTS)

31	16
Reserved	
R-0	

15	8	7	0
Reserved		WLVL	
R-0		R-0	

LEGEND: R = Read only; -*n* = value after reset

Table 19-48. Write FIFO Status Register (WFIFOSTS) Field Descriptions

Bit	Field	Value	Description
31-8	Reserved	0	Reserved
7-0	WLVL	0-FFh	Write level (read-only). Number of 32-bit words currently in the Write FIFO.
		0	0 words currently in Write FIFO.
		1h	1 word currently in Write FIFO.
		2h	2 words currently in Write FIFO.
		3h-40h	3 to 64 words currently in Write FIFO.
		41h-FFh	Reserved

Copyright © 2013–2016, Texas Instruments Incorporated

19.1.47 Read FIFO Control Register (RFIFOCTL)

The Read FIFO control register (RFIFOCTL) is shown in Figure 19-80 and described in Table 19-49.

NOTE: The RNUMEVT and RNUMDMA values must be set prior to enabling the Read FIFO.

If the Read FIFO is to be enabled, it must be enabled prior to taking the McASP out of reset.

Figure 19-80. Read FIFO Control Register (RFIFOCTL)

31	17	16
Reserved		RENA
R-0		R/W-0

15	8	7	0
RNUMEVT		RNUMDMA	
R/W-10h		R/W-4h	

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 19-49. Read FIFO Control Register (RFIFOCTL) Field Descriptions

Bit	Field	Value	Description
31-17	Reserved	0	Reserved
16	RENA		Read FIFO enable bit.
		0	Read FIFO is disabled. The RLVL bit in the Read FIFO status register (RFIFOSTS) is reset to 0 and pointers are initialized, that is, the Read FIFO is “flushed.”
		1	Read FIFO is enabled. If Read FIFO is to be enabled, it must be enabled prior to taking McASP out of reset.
15-8	RNUMEVT	0-FFh	Read word count per DMA event (32-bit). When the Read FIFO contains at least RNUMEVT words of data, then an AREVT (receive DMA event) is generated to the host/DMA controller. This value should be set to a non-zero integer multiple of the number of serializers enabled as receivers. This value must be set prior to enabling the Read FIFO.
		0	0 words
		1h	1 word
		2h	2 words
		3h-40h	3 to 64 words
		41h-FFh	Reserved
7-0	RNUMDMA	0-FFh	Read word count per transfer (32-bit words). Upon a receive DMA event from the McASP, the Read FIFO reads RNUMDMA words from the McASP. This value must equal the number of McASP serializers used as receivers. This value must be set prior to enabling the Read FIFO.
		0	0 words
		1	1 word
		2	2 words
		3h-10h	3-16 words
		11h-FFh	Reserved

Copyright © 2013–2016, Texas Instruments Incorporated

19.1.48 Read FIFO Status Register (RFIFOSTS)

The Read FIFO status register (RFIFOSTS) is shown in Figure 19-81 and described in Table 19-50.

Figure 19-81. Read FIFO Status Register (RFIFOSTS)

31	16
Reserved	
R-0	

15	8	7	0
Reserved		RLVL	
R-0		R-0	

LEGEND: R = Read only; -*n* = value after reset

Table 19-50. Read FIFO Status Register (RFIFOSTS) Field Descriptions

Bit	Field	Value	Description
31-8	Reserved	0	Reserved
7-0	RLVL	0-FFh	Read level (read-only). Number of 32-bit words currently in the Read FIFO.
		0	0 words currently in Read FIFO.
		1h	1 word currently in Read FIFO.
		2h	2 words currently in Read FIFO.
		3h-40h	3 to 64 words currently in Read FIFO.
		41h-FFh	Reserved

Copyright © 2013–2016, Texas Instruments Incorporated

Serial Peripheral Interface (SPI)

This chapter describes the serial peripheral interface (SPI) module. See your device-specific data manual to determine how many SPIs are available on your device.

Topic **Page**

20.1 Introduction **698**
20.2 Architecture **700**
20.3 Registers **726**

Copyright © 2013–2016, Texas Instruments Incorporated

20.1 Introduction

20.1.1 Purpose of the Peripheral

The SPI is a high-speed synchronous serial input/output port that allows a serial bit stream of programmed length (2 to 16 bits) to be shifted into and out of the device at a programmed bit-transfer rate. The SPI is normally used for communication between the device and external peripherals. Typical applications include interface to external I/O or peripheral expansion via devices such as shift registers, display drivers, SPI EPROMS and analog-to-digital converters.

20.1.2 Features

The SPI has the following features:

- 16-bit shift register
- 16-bit Receive buffer register (SPIBUF) and 16-bit Receive buffer emulation 'alias' register (SPIEMU)
- 16-bit Transmit data register (SPIDAT0) and 16-bit Transmit data and format selection register (SPIDAT1)
- 8-bit baud clock generator
- Serial clock (SPIx_CLK) I/O pin
- Slave in, master out (SPIx_SIMO) I/O pin
- Slave out, master in (SPIx_SOMI) I/O pin
- SPI enable ($\overline{\text{SPIx_ENA}}$) I/O pin (4-pin or 5-pin mode only)
- Multiple slave chip select ($\overline{\text{SPIx_SCS[n]}}$) I/O pins (4-pin or 5-pin mode only)
- Programmable SPI clock frequency range
- Programmable character length (2 to 16 bits)
- Programmable clock phase (delay or no delay)
- Programmable clock polarity (high or low)
- Interrupt capability
- DMA support (read/write synchronization events)

The SPI allows software to program the following options:

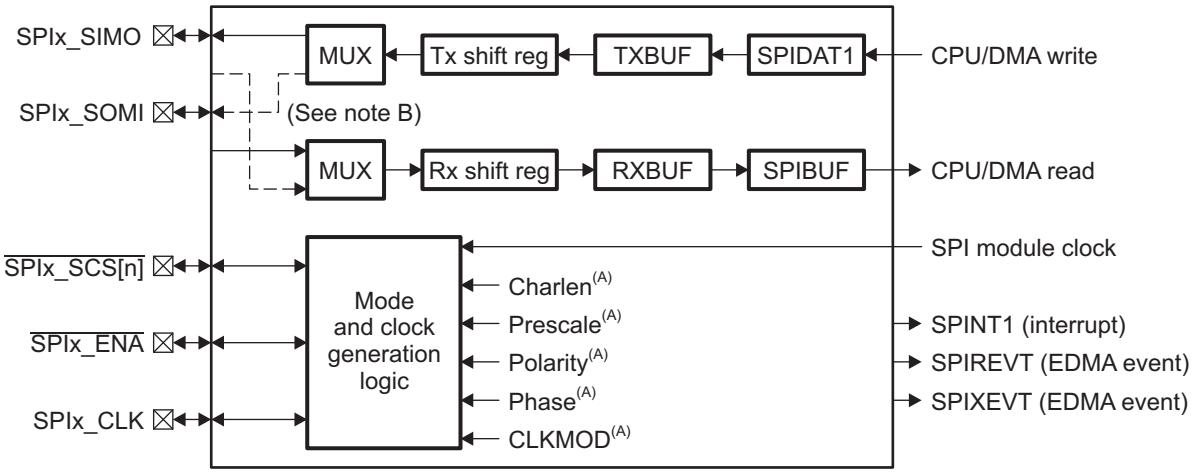
- SPI pins as functional or digital I/O pins
- SPI Master or Slave mode
- SPIx_CLK frequency (SPI module clock/3 through SPI module clock/256)
- 3-pin, 4-pin, and 5-pin options
- Character length (2 to 16 bits) and shift direction (MSB/LSB first)
- Clock phase (delay or no delay) and polarity (high or low)
- Delay between transmissions in master mode.
- Chip select setup and hold times in master mode
- Chip select hold in master mode

Copyright © 2013–2016, Texas Instruments Incorporated

20.1.3 Functional Block Diagram

The Figure 20-1 shows the SPI block diagram.

Figure 20-1. SPI Block Diagram



NOTE: The value *x* indicates the applicable SPI; that is, SPI0, SPI1, etc. See your device-specific data manual to determine how many SPIs are available on your device. The value *n* indicates the SPI pins available. See your device-specific data manual to determine how many SPI pins are available on your device.

A Indicates the log controlled by SPI register bits.

B Solid line represents data flow for SPI master mode. Dashed line represents data flow for SPI slave mode.

20.1.4 Industry Standard(s) Compliance Statement

The programmable configuration capability of the SPI allows it to gluelessly interface to a variety of SPI format devices. The SPI does not conform to a specific industry standard.

Copyright © 2013–2016, Texas Instruments Incorporated

20.2 Architecture

This section describes the SPI operation modes. It gives an overview of SPI operation and then provides details on the 3-pin, 4-pin, and 5-pin options, as well as more specific details on the supported data formats.

20.2.1 Clock

The SPI clock (SPIx_CLK) is derived from the SPI module clock. The maximum clock bit rate supported is SPI module clock/3, as determined by the PRESCALE field in the SPI data format register *n* (SPIFMT*n*). The SPIx_CLK frequency is calculated as:

SPIx_CLK frequency = [SPI module clock] / [SPIFMT*n*.PRESCALE + 1]

20.2.2 Signal Descriptions

Table 20-1 shows the SPI pins used to interface to external devices.

Table 20-1. SPI Pins

Pin[1]	Type	Function
SPIx_SIMO	Input/Output	Serial data input in slave mode, serial data output in master mode
SPIx_SOMI	Input/Output	Serial data output in slave mode, serial data input in master mode
SPIx_CLK	Input/Output	Serial clock input in slave mode, serial clock output in master mode
$\overline{\text{SPIx_SCS[n]}}$ [2]	Input/Output	Slave chip select output in master mode, input in slave mode
$\overline{\text{SPIx_ENA}}$	Input/Output	Input in master mode, output in slave mode indicates slave is ready

(1) The value *x* indicates the applicable SPI; that is, SPI0, SPI1, etc. See your device-specific data manual to determine how many SPIs are available on your device.

(2) The value *n* indicates the SPI pins available; that is, $\overline{\text{SPIx_SCS[0]}}$, $\overline{\text{SPIx_SCS[1]}}$, etc. See your device-specific data manual to determine how many SPI pins are available on your device.

20.2.3 Operation Modes

The SPI operates in master or slave mode. The SPI bus master is the device that drives the SPIx_CLK, SPIx_SIMO, and optionally the $\overline{\text{SPIx_SCS[n]}}$ signals, and therefore initiates SPI bus transfers. The CLKMOD and MASTER bits in the SPI global control register 1 (SPIGCR1) select between master and slave mode. In both master and slave mode, the SPI supports four options:

- 3-pin option
- 4-pin with chip select option
- 4-pin with enable option
- 5-pin with enable and chip select option

The 3-pin option is the basic clock, data in, and data out SPI interface and uses the SPIx_CLK, SPIx_SIMO, and SPIx_SOMI pins. The 4-pin with chip select option adds the $\overline{\text{SPIx_SCS[n]}}$ pin that is used to support multiple SPI slave devices on a single SPI bus. The 4-pin with enable option adds the $\overline{\text{SPIx_ENA}}$ pin that is used to increase the overall throughput by adding hardware handshaking. The 5-pin option uses all the SPI pins and is a superset of the different options.

Copyright © 2013–2016, Texas Instruments Incorporated

20.2.4 Programmable Registers

A general representation of the SPI programmable registers is shown in Table 20-2. For details on registers, see Section 20.3.

Table 20-2. SPI Registers

Offset Address[1]	Acronym	Name	Description	Section
0h	SPIGCR0	Global control register 0	Contains the software reset bit for the module	Section 20.3.1
4h	SPIGCR1	Global control register 1	Controls basic configurations of the module	Section 20.3.2
8h	SPIINT0	Interrupt register	Enable bits for interrupts, error, DMA and other functionality.	Section 20.3.3
Ch	SPILVL	Level register	SPI interrupt levels are set in this register.	Section 20.3.4
10h	SPIFLG	Flag register	Shows the status of several events during the operation.	Section 20.3.5
14h	SPIPC0	Pin control register 0	Determines if pins operate as general I/O or SPI functional pin	Section 20.3.6
18h	SPIPC1	Pin control register 1	Controls the direction of data on the I/O pins	Section 20.3.7
1Ch	SPIPC2	Pin control register 2	Reflects the values on the I/O pins	Section 20.3.8
20h	SPIPC3	Pin control register 3	Controls the values sent to the I/O pins	Section 20.3.9
24h	SPIPC4	Pin control register 4	Sets data values in the SPIPC3 register	Section 20.3.10
28h	SPIPC5	Pin control register 5	Clears values in the SPIPC3 register	Section 20.3.11
38h	SPIDAT0	Transmit data register 0	Transmit data register	Section 20.3.12
3Ch	SPIDAT1	Transmit data register 1	Transmit data with format selection register	Section 20.3.13
40h	SPIBUF	Receive buffer register	Holds received word	Section 20.3.14
44h	SPIEMU	Receive buffer emulation register	Mirror of SPIBUF. Read does not clear flags	Section 20.3.15
48h	SPIDELAY	Delay register	Sets $\overline{\text{SPIx_SCS[n]}}$ mode, $\overline{\text{SPIx_SCS[n]}}$ pre-/post-transfer delay time and $\overline{\text{SPIx_ENA}}$ time-out	Section 20.3.16
4Ch	SPIDEF	Chip select default register	In $\overline{\text{SPIx_SCS[n]}}$ decoded mode only: sets high low/active $\overline{\text{SPIx_SCS[n]}}$ signal	Section 20.3.17
50h	SPIFMT0	Format 0 register	Configuration of data word format 0	Section 20.3.18
54h	SPIFMT1	Format 1 register	Configuration of data word format 1	Section 20.3.18
58h	SPIFMT2	Format 2 register	Configuration of data word format 2	Section 20.3.18
5Ch	SPIFMT3	Format 3 register	Configuration of data word format 3	Section 20.3.18
64h	INTVEC1	Interrupt vector register 1	Interrupt vector for line INT1	Section 20.3.19

(1) The actual address of these registers is device specific and CPU specific. See your device-specific data manual to verify the SPI register addresses.

Copyright © 2013–2016, Texas Instruments Incorporated

20.2.5 Master Mode Settings

The four master mode options are defined by the configuration bit settings listed in Table 20-3. Other configuration bits may take any value in the range listed in Table 20-4. The values listed in Table 20-3 and Table 20-4 should not be changed while the ENABLE bit in the SPI global control register 1 (SPIGCR1) is set to 1. Note that in certain cases the allowed values may still be ignored. For example, Table 20-4 indicates that SPIDELAY may take a range of values in Master 3-pin mode; however, SPIDELAY has no effect in Master 3-pin mode. For complete details on each mode, see the following sections that explain the SPI operation for each of the master modes.

Table 20-3. SPI Register Settings Defining Master Modes

Register	Bit(s)	Master 3-pin	Master 4-pin Chip Select	Master 4-pin Enable	Master 5-pin
SPIGCR0	RESET	1	1	1	1
SPIGCR1	ENABLE	1	1	1	1
SPIGCR1	LOOPBACK	0	0	0	0
SPIGCR1	CLKMOD	1	1	1	1
SPIGCR1	MASTER	1	1	1	1
SPIPC0	SOMIFUN	1	1	1	1
SPIPC0	SIMOFUN	1	1	1	1
SPIPC0	CLKFUN	1	1	1	1
SPIPC0	ENAFUN	0	0	1	1
SPIPC0	SCS0FUN	0	1	0	1

Table 20-4. Allowed SPI Register Settings in Master Modes

Register	Bit(s)	Master 3-pin	Master 4-pin Chip Select	Master 4-pin Enable	Master 5-pin
SPIINT0	ENABLEHIGHZ	0,1	0,1	0,1	0,1
SPIFMT*n*	WDELAY	0 to 3Fh	0 to 3Fh	0 to 3Fh	0 to 3Fh
SPIFMT*n*	PARPOL	0,1	0,1	0,1	0,1
SPIFMT*n*	PARENA	0,1	0,1	0,1	0,1
SPIFMT*n*	WAITENA	0	0	1	1
SPIFMT*n*	SHIFTDIR	0,1	0,1	0,1	0,1
SPIFMT*n*	DISCSTIMERS	0,1	0,1	0,1	0,1
SPIFMT*n*	POLARITY	0,1	0,1	0,1	0,1
SPIFMT*n*	PHASE	0,1	0,1	0,1	0,1
SPIFMT*n*	PRESCALE	2 to FFh	2 to FFh	2 to FFh	2 to FFh
SPIFMT*n*	CHARLEN	2 to 10h	2 to 10h	2 to 10h	2 to 10h
SPIDELAY	C2TDELAY	0 to FFh	0 to FFh	0 to FFh	0 to FFh
SPIDELAY	T2CDELAY	0 to FFh	0 to FFh	0 to FFh	0 to FFh
SPIDELAY	T2EDELAY	0 to FFh	0 to FFh	0 to FFh	0 to FFh
SPIDELAY	C2EDELAY	0 to FFh	0 to FFh	0 to FFh	0 to FFh

20.2.5.1 Master Mode Timing Options

The SPI in master mode supports several options to modify the timing of its generation of the chip select signal ($\overline{\text{SPIx_SCS[n]}}$). This allows the SPI to support the timing requirements of various slave devices without adding additional overhead to the CPU by generating the appropriate delays automatically.

Copyright © 2013–2016, Texas Instruments Incorporated

20.2.5.1.1 Chip Select Setup Time

The master can be configured to provide a (slow) slave device a certain chip select setup time to the first edge on SPIx_CLK. This delay is controlled by the C2TDELAY field in the SPI delay register (SPIDELAY) and can be configured between 3 and 257 SPI module clock cycles. The C2TDELAY is applicable only in 4-pin with chip select and 5-pin SPI master modes. The C2TDELAY begins when the SPI master asserts $\overline{\text{SPIx_SCS[n]}}$. The C2T delay period is specified by:

Maximum duration of C2TDELAY period = SPIDELAY.C2TDELAY + 2 (SPI module clock cycles)

Note that if SPIDELAY.C2TDELAY = 0, then the C2TDELAY period = 0.

The previous value of the CSHOLD bit in the SPI transmit data register (SPIDAT1) must be cleared to 0 for the C2T delay to be enabled.

NOTE: If the SPIDAT1.CSHOLD bit is set within the control field, the current hold time and the following setup time will not be applied in between transaction.

20.2.5.1.2 Chip Select Hold Time

The master can be configured to provide a (slow) slave device a certain chip select hold time after the last edge on SPIx_CLK. This delay is controlled by the T2CDELAY bit in the SPI delay register (SPIDELAY) and can be configured between 2 and 256 SPI module clock cycles. The T2CDELAY is applicable only in 4-pin with chip select and 5-pin SPI master modes. The T2CDELAY begins after the data shifting period ends. The T2C delay period is specified by:

Maximum duration of T2CDELAY period = SPIDELAY.T2CDELAY + 1 (SPI module clock cycle)

Note that if SPIDELAY.T2CDELAY = 0, then the T2CDELAY period = 0. If the PHASE bit in the SPI data format register *n* (SPIFMT*n*) is 0, then the T2CDELAY period lasts for an additional 1/2 SPIx_CLK time over that specified by the above equation.

The current value of the CSHOLD bit in the SPI transmit data register (SPIDAT1) must be cleared to 0 for T2C delay to be enabled.

NOTE: If the SPIDAT1.CSHOLD bit is set within the control field, the current hold time and the following setup time will not be applied in between transaction.

20.2.5.1.3 Automatic Delay Between Transfers

The SPI master can automatically insert a delay of between 2 and 65 SPI module clock cycles between transmissions. This delay is controlled by the WDELAY field in the SPI data format register *n* (SPIFMT*n*) and is enabled by setting the WDEL bit in the SPI transmit data register (SPIDAT1) to 1. The WDELAY period begins when the T2EDELAY period terminates (if T2E delay period is enabled) or when the T2CDELAY period terminates (if T2E delay period was disabled and T2C delay period was enabled) or when the master deasserts $\overline{\text{SPIx_SCS[n]}}$ (if T2E and T2C delay periods are disabled). If a transfer is initiated by writing a 32-bit value to SPIDAT1, then the new values of SPIDAT1.WDEL and SPIFMT*n*.WDELAY are used; otherwise, the old values of SPIDAT1.WDEL and SPIFMT*n*.WDELAY are used. The WDELAY delay period is specified by:

Maximum duration of WDELAY period = SPIFMTn.WDELAY + 2 (SPI module clock cycles)

20.2.5.1.4 Chip Select Hold Option

There are slave devices available that require the chip select signal to be held continuously active during several consecutive data word transfers. Other slave devices require the chip select signal to be deactivated between consecutive data word transfers. The SPI can support both types of slave devices. The CSHOLD bit in the SPI transmit data register (SPIDAT1) selects between the two options.

Copyright © 2013–2016, Texas Instruments Incorporated

If the chip select hold option is enabled, the chip select will not toggle between two consecutive accesses; therefore, the SPIDELAY.T2CDELAY of the first transfer and the SPIDELAY.C2TDELAY of the second transfer will not be applied. However, the wait delay could still be applied between the two transactions, if the WDEL bit in SPIDAT1 is set to 1.

The current and previous values of the CSHOLD bit are retained. Though the current value of the CSHOLD bit is initialized to 0 when the RESET bit in the SPI global control register 0 (SPIGCR0) is cleared to 0, the previous value of the CSHOLD bit is not initialized. The previous value of the CSHOLD bit must be explicitly initialized by writing twice to the CSHOLD bit.

20.2.6 Slave Mode Settings

The four slave mode options are defined by the configuration bit settings listed in Table 20-5. Other configuration bits may take any value in the range listed in Table 20-6. The values listed in Table 20-5 and Table 20-6 should not be changed while the ENABLE bit in the SPI global control register 1 (SPIGCR1) is set to 1. Note that in certain cases the allowed values may still be ignored. For complete details on each mode, see the following sections that explain the SPI operation for each of the slave modes.

Table 20-5. SPI Register Settings Defining Slave Modes

Register	Bit(s)	Slave 3-pin	Slave 4-pin Chip Select	Slave 4-pin Enable	Slave 5-pin
SPIGCR0	RESET	1	1	1	1
SPIGCR1	ENABLE	1	1	1	1
SPIGCR1	LOOPBACK	0	0	0	0
SPIGCR1	CLKMOD	0	0	0	0
SPIGCR1	MASTER	0	0	0	0
SPIPC0	SOMIFUN	1	1	1	1
SPIPC0	SIMOFUN	1	1	1	1
SPIPC0	CLKFUN	1	1	1	1
SPIPC0	ENAFUN	0	0	1	1
SPIPC0	SCS0FUN	0	1	0	1

Table 20-6. Allowed SPI Register Settings in Slave Modes

Register	Bit(s)	Slave 3-pin	Slave 4-pin Chip Select	Slave 4-pin Enable	Slave 5-pin
SPIINT0	ENABLEHIGHZ	0,1	0,1	0,1	0,1
SPIFMT*n* [(1)]	WDELAY	0 to 3Fh	0 to 3Fh	0 to 3Fh	0 to 3Fh
SPIFMT*n* [(1)]	PARPOL	0,1	0,1	0,1	0,1
SPIFMT*n* [(1)]	PARENA	0,1	0,1	0,1	0,1
SPIFMT*n* [(1)]	WAITENA	0,1	0,1	0,1	0,1
SPIFMT*n* [(1)]	SHIFTDIR	0,1	0,1	0,1	0,1
SPIFMT*n* [(1)]	DISCSTIMERS	0,1	0,1	0,1	0,1
SPIFMT*n* [(1)]	POLARITY	0,1	0,1	0,1	0,1
SPIFMT*n* [(1)]	PHASE	0,1	0,1	0,1	0,1
SPIFMT*n* [(1)]	PRESCALE	2 to FFh	2 to FFh	2 to FFh	2 to FFh
SPIFMT*n* [(1)]	CHARLEN	2 to 10h	2 to 10h	2 to 10h	2 to 10h
SPIDELAY	C2TDELAY	0 to FFh	0 to FFh	0 to FFh	0 to FFh
SPIDELAY	T2CDELAY	0 to FFh	0 to FFh	0 to FFh	0 to FFh
SPIDELAY	T2EDELAY	0 to FFh	0 to FFh	0 to FFh	0 to FFh
SPIDELAY	C2EDELAY	0 to FFh	0 to FFh	0 to FFh	0 to FFh

(1) In slave mode, only SPIFMT0 is used. When SPIDAT1 is written, the DFSEL field in SPIDAT1 is cleared to 0 to select SPIFMT0.

Copyright © 2013–2016, Texas Instruments Incorporated

20.2.7 SPI Operation: 3-Pin Mode

NOTE: If only unidirectional communication is required, the SPIx_CLK pin and the two data pins (SPIx_SOMI and SPIx_SIMO) must all be configured as functional pins. A 2-pin unidirectional mode is not supported.

The SPI 3-pin mode uses only the clock (SPIx_CLK) and data (SPIx_SOMI and SPIx_SIMO) pins for bidirectional communication between master and slave devices. Figure 20-2 shows the basic 3-pin SPI option.

To select the 3-pin SPI option, the SPIx_CLK, SPIx_SOMI, and SPIx_SIMO pins should be configured as functional pins by configuring the SPI pin control register 0 (SPIPC0). The $\overline{\text{SPIx_SCS[n]}}$ and $\overline{\text{SPIx_ENA}}$ pins can be used as general-purpose I/O pins by configuring the SPIPC1 through SPIPC5 registers.

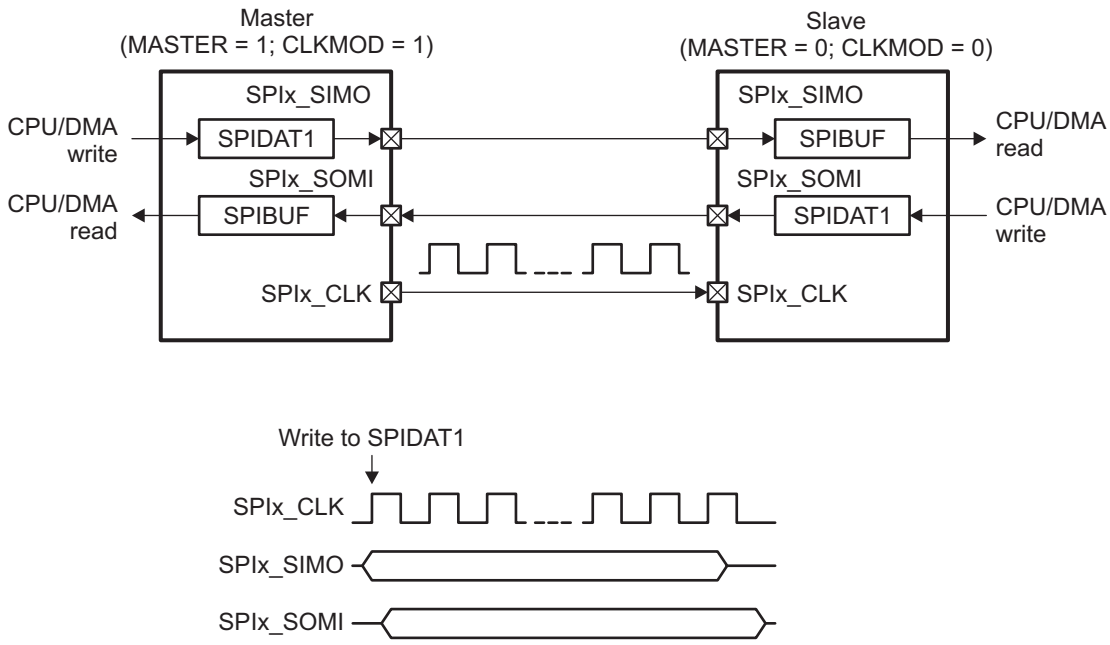
The SPI operates in either master or slave mode. The CLKMOD and MASTER bits in the SPI global control register 1 (SPIGCR1) select between master and slave mode; both must be programmed to 1 to configure the SPI for master mode or to 0 to configure the SPI for slave mode. The SPI bus master is the device that drives the SPIx_CLK signal and initiates SPI bus transfers. In SPI master mode, the SPIx_SOMI pin output buffer is in a high-impedance state and the SPIx_CLK and the SPIx_SIMO pin output buffer is enabled. In SPI slave mode, the SPIx_SIMO and SPIx_CLK pin output buffer is in a high-impedance state and the SPIx_SOMI pin output buffer is enabled.

In master mode with the 3-pin option, the DSP writes transmit data to the SPI transmit data registers (SPIDAT0[15:0] or SPIDAT1[15:0]). This initiates a transfer. A series of clocks pulses will be driven out on the SPIx_CLK pin to complete the transfer. Each clock pulse on the SPIx_CLK pin causes the simultaneous transfer (in both directions) of one bit by both the master and slave SPI devices. CPU writes to the configuration bits in SPIDAT1 (not writing to SPIDAT1[15:0]) do not result in a new transfer. When the selected number of bits has been transmitted, the received data is transferred to the SPI receive buffer register (SPIBUF) for the CPU to read. Data is stored right-justified in SPIBUF.

In slave mode with 3-pin option, CPU writes to SPIDAT0[15:0] or SPIDAT1[15:0] makes the slave ready to transmit. CPU writes to the configuration bits in SPIDAT1 (not writing to SPIDAT1[15:0]) do not make the slave ready to transmit.

NOTE: Either SPIDAT0 or SPIDAT1 can be used on both master and slaves sides.

Figure 20-2. SPI 3-Pin Option



Copyright © 2013–2016, Texas Instruments Incorporated

20.2.8 SPI Operation: 4-Pin with Chip Select Mode

NOTE: The SPI only supports a single $\overline{\text{SPIx_SCS[n]}}$ pin and so the usefulness of the $\overline{\text{SPIx_SCS[n]}}$ pin in master mode is limited. In practice, general-purpose I/O pins are needed to support multiple slave device chip selects.

The 4-pin with chip select option is a superset of the 3-pin option and uses the chip select ($\overline{\text{SPIx_SCS[n]}}$) pin in addition to the clock (SPIx_CLK) and data (SPIx_SOMI and SPIx_SIMO) pins. Figure 20-3 shows the SPI 4-pin chip select option.

To select the 4-pin with chip select option, the SPIx_CLK, SPIx_SOMI, SPIx_SIMO, and $\overline{\text{SPIx_SCS[n]}}$ pins should be configured as functional pins by configuring the SPI pin control register 0 (SPIPC0). The $\overline{\text{SPIx_ENA}}$ pin can be used as a general-purpose I/O pin by configuring the SPIPC1 through SPIPC5 registers.

In SPI master mode, the SPIx_SOMI pin output buffer is in a high-impedance state and the SPIx_CLK, SPIx_SIMO, and $\overline{\text{SPIx_SCS[n]}}$ pin output buffer is enabled. In SPI slave mode, the SPIx_CLK, SPIx_SIMO, and $\overline{\text{SPIx_SCS[n]}}$ pin output buffer is in a high-impedance state, and the SPIx_SOMI pin output buffer is enabled when $\overline{\text{SPIx_SCS[n]}}$ is asserted and in a high-impedance state when $\overline{\text{SPIx_SCS[n]}}$ is deasserted.

In slave mode with the chip select option enabled, the SPI ignores all transactions on the bus unless $\overline{\text{SPIx_SCS[n]}}$ is asserted by the bus master. It also 3-states its output pin when $\overline{\text{SPIx_SCS[n]}}$ is deasserted by the master to avoid conflicting with the active slave device on the bus.

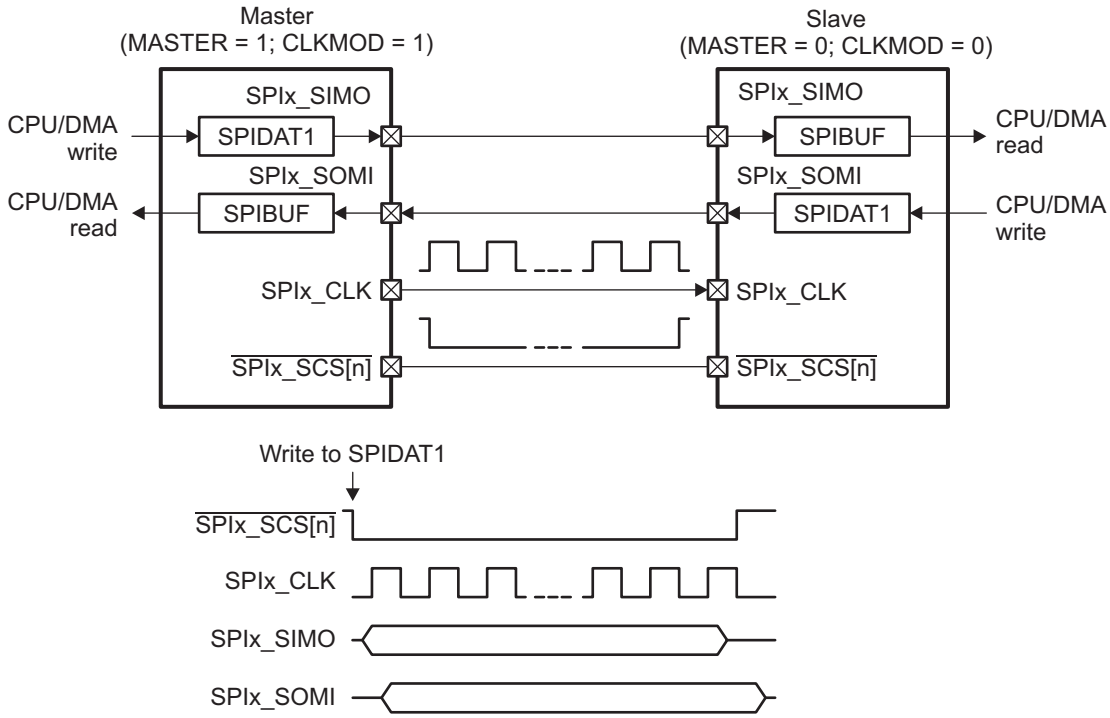
In master mode, the $\overline{\text{SPIx_SCS[n]}}$ pin functions as an output, and toggles when a specific slave device is selected. However, this is most useful on devices that support multiple $\overline{\text{SPIx_SCS[n]}}$ pins.

However, one reason to use the $\overline{\text{SPIx_SCS[n]}}$ pin as a functional pin for the SPI master is to take advantage of the timing parameters that can be set using the SPI delay register (SPIDELAY). The SPIDELAY allows delays to be added automatically so that the slave timing requirements between clock and chip select may be more easily met. Another reason would be to make use of the error detection built into the SPI.

NOTE: Either SPIDAT0 or SPIDAT1 can be used on both master and slaves sides.

Copyright © 2013–2016, Texas Instruments Incorporated

Figure 20-3. SPI 4-Pin Option with $\overline{\text{SPIx_SCS[n]}}$



NOTE: During an SPI transfer, if the slave mode SPI detects a deassertion of its chip select even before its internal character length counter overflows, then it 3-states its SPIx_SOMI pin. Once this condition has occurred, if a SPIx_CLK edge is detected while the chip select is deasserted, the SPI stops the transfer and sets an error flag DLENERR (data length) and generates an interrupt if enabled.

Copyright © 2013–2016, Texas Instruments Incorporated

20.2.9 SPI Operation: 4-Pin with Enable Mode

The 4-pin with enable option is a superset of the 3-pin option and uses the enable ($\overline{\text{SPIx_ENA}}$) pin in addition to the clock (SPIx_CLK) and data (SPIx_SOMI and SPIx_SIMO) pins. Figure 20-4 shows the SPI 4-pin enable option.

To select the 4-pin with enable option, the SPIx_CLK, SPIx_SOMI, SPIx_SIMO, and $\overline{\text{SPIx_ENA}}$ pins should be configured as functional pins by configuring the SPI pin control register 0 (SPIPC0). The $\overline{\text{SPIx_SCS[n]}}$ pins can be used as general-purpose I/O pins by configuring the SPIPC1 through SPIPC5 registers.

In SPI master mode, the SPIx_SOMI and $\overline{\text{SPIx_ENA}}$ pin output buffer is in a high-impedance state and the SPIx_CLK and SPIx_SIMO pin output buffer is enabled. In SPI slave mode, the SPIx_CLK and SPIx_SIMO pin output buffer is in a high-impedance state, and the SPIx_SOMI pin output buffer is enabled. In SPI slave mode, the $\overline{\text{SPIx_ENA}}$ pin output buffer enable depends upon the status of the transmit buffer and the configuration of the ENABLEHIGHZ bit in the SPI interrupt register (SPIINT0).

The handshake operation works this way:

- After a transfer completes, both the master and slave SPI modules need to be serviced.
- The slave SPI deasserts $\overline{\text{SPIx_ENA}}$ after the transfer, indicating it requires servicing and is not ready.
- The slave should begin servicing its SPI by first reading receive data from the SPI receive buffer register (SPIBUF).
- Next, the slave device should write transmit data to the SPI transmit data registers (SPIDAT0 or SPIDAT1). This causes the slave SPI to assert $\overline{\text{SPIx_ENA}}$ indicating it is ready for the next transmission.
- In parallel, the master device can service its SPI at any time. It does not need to insert a delay before writing to its SPIDAT0 or SPIDAT1 in order to avoid overrunning the slave device. Instead, the master SPI module will automatically delay the next transfer until the slave has asserted $\overline{\text{SPIx_ENA}}$ again to indicate it is ready for the transmission.

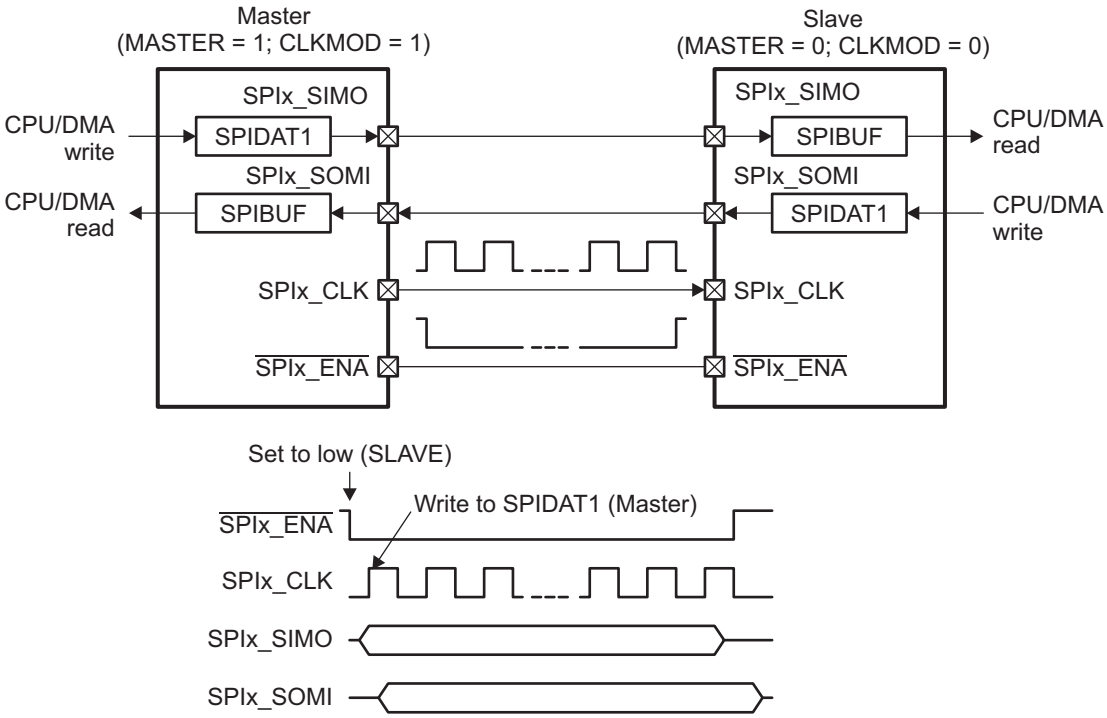
This handshake allows the two SPIs to communicate at the maximum rate possible. Without the handshake pin, the master must insert a delay between each transfer long enough to support the worst case response time of the slave servicing its SPI or risk an overrun condition. With the handshake, the throughput is determined by the average response time of the two devices servicing their SPI ports.

The $\overline{\text{SPIx_ENA}}$ pin can be driven in a push-pull or open-drain mode, depending upon the setting of the ENABLEHIGHZ bit.

NOTE: Either SPIDAT0 or SPIDAT1 can be used on both master and slaves sides.

Copyright © 2013–2016, Texas Instruments Incorporated

Figure 20-4. SPI 4-Pin Option with $\overline{\text{SPIx_ENA}}$



Copyright © 2013–2016, Texas Instruments Incorporated

20.2.10 SPI Operation: 5-Pin Mode

NOTE: The SPI only supports a single $\overline{\text{SPIx_SCS[n]}}$ pin and so the usefulness of the $\overline{\text{SPIx_SCS[n]}}$ pin in master mode is limited. In practice, general-purpose I/O pins are needed to support multiple slave device chip selects.

The 5-pin mode is a superset of both 4-pin modes. To use the 5-pin mode, both the $\overline{\text{SPIx_ENA}}$ and the $\overline{\text{SPIx_SCS[n]}}$ pins must be configured as functional pins, in addition to the SPIx_CLK, SPIx_SIMO, and SPIx_SOMI pins by configuring the SPI pin control register 0 (SPIPC0). Figure 20-5 shows the SPI 5-pin option.

In SPI master mode, the SPIx_SOMI and $\overline{\text{SPIx_ENA}}$ pin output buffer is in a high-impedance state and the SPIx_CLK, SPIx_SIMO, and $\overline{\text{SPIx_SCS[n]}}$ pin output buffer is enabled. In SPI slave mode, the SPIx_CLK, SPIx_SIMO, and $\overline{\text{SPIx_SCS[n]}}$ pin output buffer is in a high-impedance state, and the SPIx_SOMI pin output buffer is enabled and disabled asynchronously by the $\overline{\text{SPIx_SCS[n]}}$ input and the $\overline{\text{SPIx_ENA}}$ pin output buffer enable depends upon the status of the transmit buffer and the state of the $\overline{\text{SPIx_SCS[n]}}$ input. In SPI slave mode, the assertion of the $\overline{\text{SPIx_ENA}}$ pin by the slave is delayed until the master asserts $\overline{\text{SPIx_SCS[n]}}$, thereby, allowing multiple SPI slaves on a single SPI bus, each slave with its own enable pin.

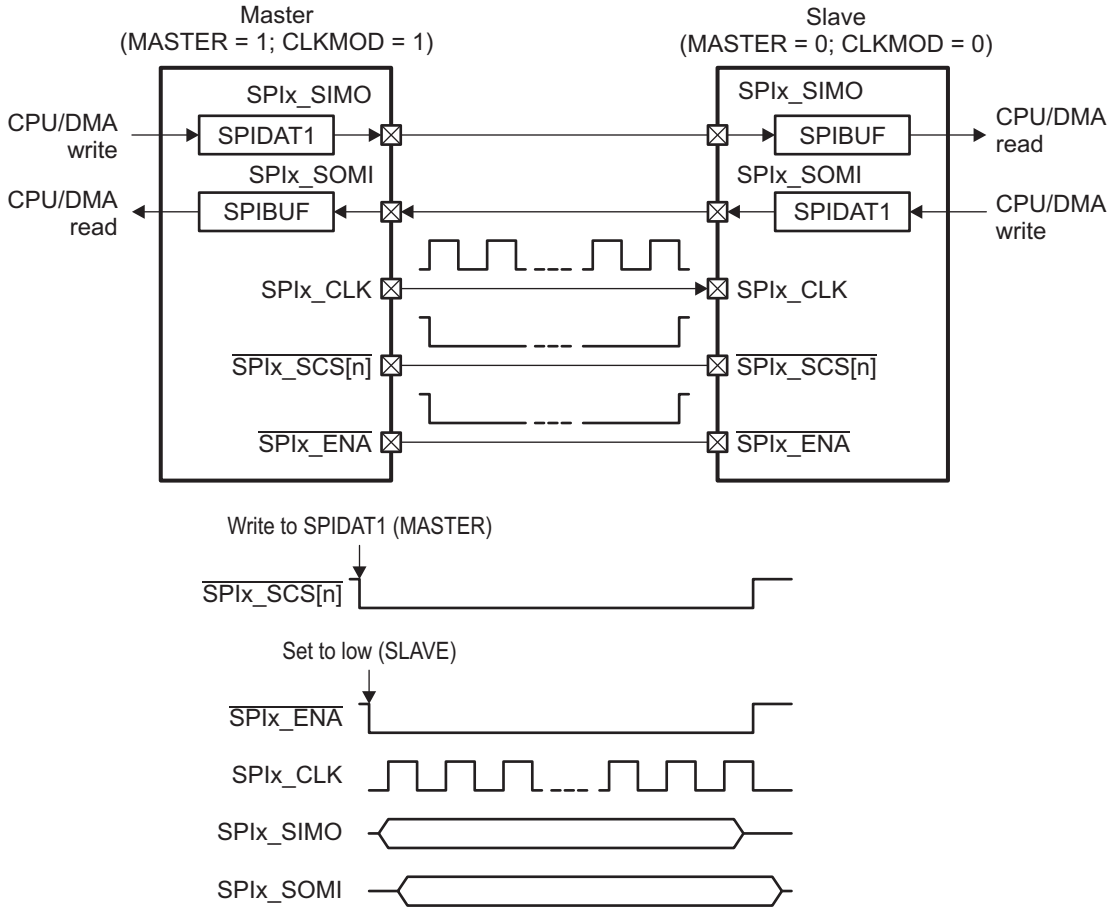
If the $\overline{\text{SPIx_ENA}}$ pin is in high-impedance mode (ENABLEHIGHZ = 1 in the SPI interrupt register (SPIINT0)), the slave SPI will put this signal into the high-impedance state by default. The slave SPI will drive the $\overline{\text{SPIx_ENA}}$ signal low when new data is written to the slave transmit shift register and the slave has been selected by the master ($\overline{\text{SPIx_SCS[n]}}$ is low).

If the $\overline{\text{SPIx_ENA}}$ pin is in push-pull mode (ENABLEHIGHZ = 0), the slave SPI will drive this pin high by default when it is in functional mode. The slave SPI will drive the $\overline{\text{SPIx_ENA}}$ signal low when new data is written to the slave transmit shift register and the slave is selected by the master ($\overline{\text{SPIx_SCS[n]}}$ is low). If the slave is deselected by the master ($\overline{\text{SPIx_SCS[n]}}$ goes high), the slave $\overline{\text{SPIx_ENA}}$ signal is driven high automatically.

NOTE: Either SPIDAT0 or SPIDAT1 can be used on both master and slaves sides.

Copyright © 2013–2016, Texas Instruments Incorporated

Figure 20-5. SPI 5-Pin Option with $\overline{\text{SPIx_ENA}}$ and $\overline{\text{SPIx_SCS[n]}}$



NOTE: Push-Pull mode of the $\overline{\text{SPIx_ENA}}$ pin can be used only when there is a single slave in the system. When there are multiple SPI slave devices connected to the common $\overline{\text{SPIx_ENA}}$ pin, all the slaves should configure their $\overline{\text{SPIx_ENA}}$ pins in high-impedance mode.

During an SPI transfer, if slave mode SPI detects a deassertion of its chip select even before its internal character length counter overflows, then it 3-states its SPIx_SOMI and $\overline{\text{SPIx_ENA}}$ (if SPIINT0.ENABLEHIGHZ bit is set to 1) pins. Once this condition has occurred, if a SPIx_CLK edge is detected while the chip select is deasserted, then the SPI stops that transfer and sets an error flag DLENERR (data length) and generates an interrupt if enabled.

Copyright © 2013–2016, Texas Instruments Incorporated

20.2.11 Data Formats

The SPI provides the capability to configure four independent data formats. These formats are configured by programming the corresponding SPI data format registers (SPIFMT*n*). In each data format, the following characteristics of the SPI operation are selected:

- Character length from 2 to 16 bits: The character length is configured by the SPIFMT*n*.CHARLEN field.
- Shift direction (MSB first or LSB first): The shift out direction is configured by the SPIFMT*n*.SHIFTDIR bit.
- Clock polarity: The clock polarity is configured by the SPIFMT*n*.POLARITY bit.
- Clock phase: The clock phase is configured by the SPIFMT*n*.PHASE bit.

The data format is chosen on each transaction. Transmit data is written to the SPI transmit data register 1 (SPIDAT1) and in the same write the data word format select (DFSEL) bit in SPIDAT1 indicates which data format is to be used for the next transaction. Alternatively, the data format can be configured once and applies to all transactions that follow until the data format is changed.

20.2.11.1 Character Length

The character length is configured by the SPIFMT*n*.CHARLEN bit. Legal values are 2 bits (2h) to 16 bits (10h). The character length is independently configured for each of the four data formats; and it must be programmed in both master mode and slave mode.

Transmit data is written to SPIDAT1. The transmit data must be written right-justified irrespective of the character length. The SPI automatically sends out the data correctly based on the chosen data format.

Figure 20-6 shows how a 12-bit word (EC9h) needs to be written to the transmit buffer in order to be transmitted correctly.

Figure 20-6. Format for Transmitting 12-Bit Word

D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
x	x	x	x	1	1	1	0	1	1	0	0	1	0	0	1

The data received in SPIBUF is right-justified irrespective of the character length and is padded with 0s when character length is less than 16.

Figure 20-7 shows how a 10-bit word (3A2h) is stored in the buffer once it is received.

Figure 20-7. Format for 10-Bit Received Word

D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
0	0	0	0	0	0	1	1	1	0	1	0	0	0	1	0

20.2.11.2 Shift Direction

The shift out direction is configured as most-significant bit (MSB) first or least significant bit (LSB) first. The shift out direction is selected by the SPIFMT*n*.SHIFTDIR bit. The shift out direction is independently configured for each of the four data formats.

- When SPIFMT*n*.SHIFTDIR is 0, the transmit data is shifted out MSB first.
- When SPIFMT*n*.SHIFTDIR is 1, the transmit data is shifted out LSB first.

Copyright © 2013–2016, Texas Instruments Incorporated

20.2.11.3 Clock Phase and Polarity

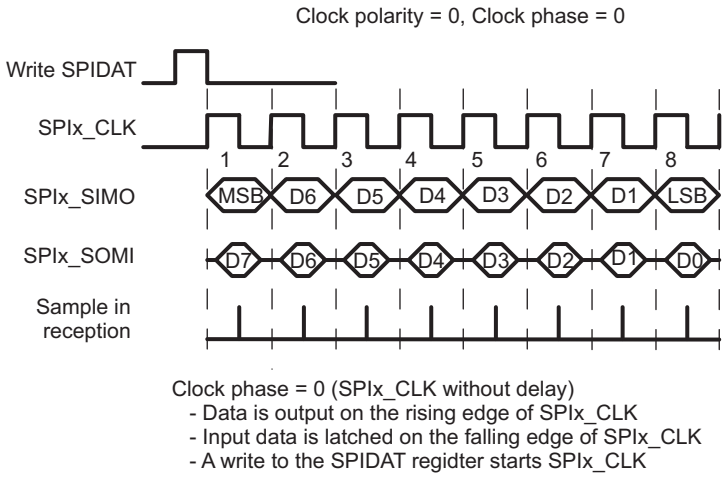
The SPI provides the flexibility to program four different clock mode combinations that SPIx_CLK may operate, enabling a choice of the clock phase (delay or no delay) and the clock polarity (rising edge or falling edge). When operating with PHASE active, the SPI makes the first bit of data available after SPIDAT1 is written and before the first edge of SPIx_CLK. The data input and output edges depend on the values of both the POLARITY and PHASE bits as shown in Table 20-7.

Table 20-7. Clocking Modes

POLARITY	PHASE	Action
0	0	Data is output on the rising edge of SPIx_CLK. Input data is latched on the falling edge.
0	1	Data is output one half-cycle before the first rising edge of SPIx_CLK and on subsequent falling edges. Input data is latched on the rising edge of SPIx_CLK.
1	0	Data is output on the falling edge of SPIx_CLK. Input data is latched on the rising edge.
1	1	Data is output one half-cycle before the first falling edge of SPIx_CLK and on subsequent rising edges. Input data is latched on the falling edge of SPIx_CLK.

Figure 20-8 to Figure 20-11 illustrate the four possible signals of SPIx_CLK corresponding to each mode. Having four signal options allows the SPI to interface with different types of serial devices. Also shown are the SPIx_CLK control bit polarity and phase values corresponding to each signal.

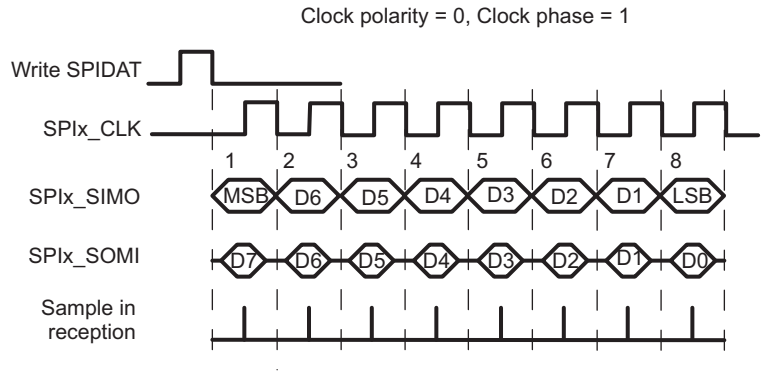
Figure 20-8. Clock Mode with POLARITY = 0 and PHASE = 0



Clock phase = 0 (SPIx_CLK without delay)
- Data is output on the rising edge of SPIx_CLK
- Input data is latched on the falling edge of SPIx_CLK
- A write to the SPIDAT regidter starts SPIx_CLK

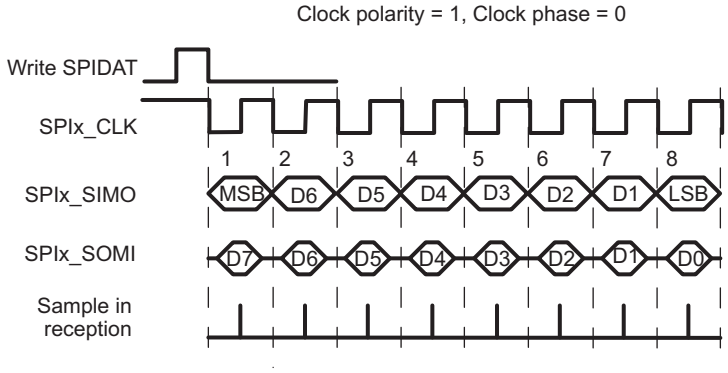
Copyright © 2013–2016, Texas Instruments Incorporated

Figure 20-9. Clock Mode with POLARITY = 0 and PHASE = 1



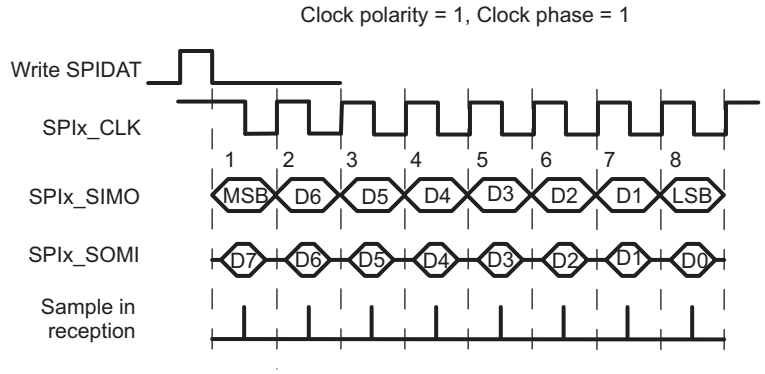
Clock phase = 1 (SPIx_CLK with delay)
- Data is output one-half cycle before the first rising of SPIx_CLK and on subsequent falling edges of SPIx_CLK
- Input data is latched on the rising edge of SPIx_CLK

Figure 20-10. Clock Mode with POLARITY = 1 and PHASE = 0



Clock phase = 0 (SPIx_CLK without delay)
- Data is output on the falling edge of SPIx_CLK
- Input data is latched on the rising edge of SPIx_CLK
- A write to the SPIDAT register starts SPIx_CLK

Figure 20-11. Clock Mode with POLARITY = 1 and PHASE = 1



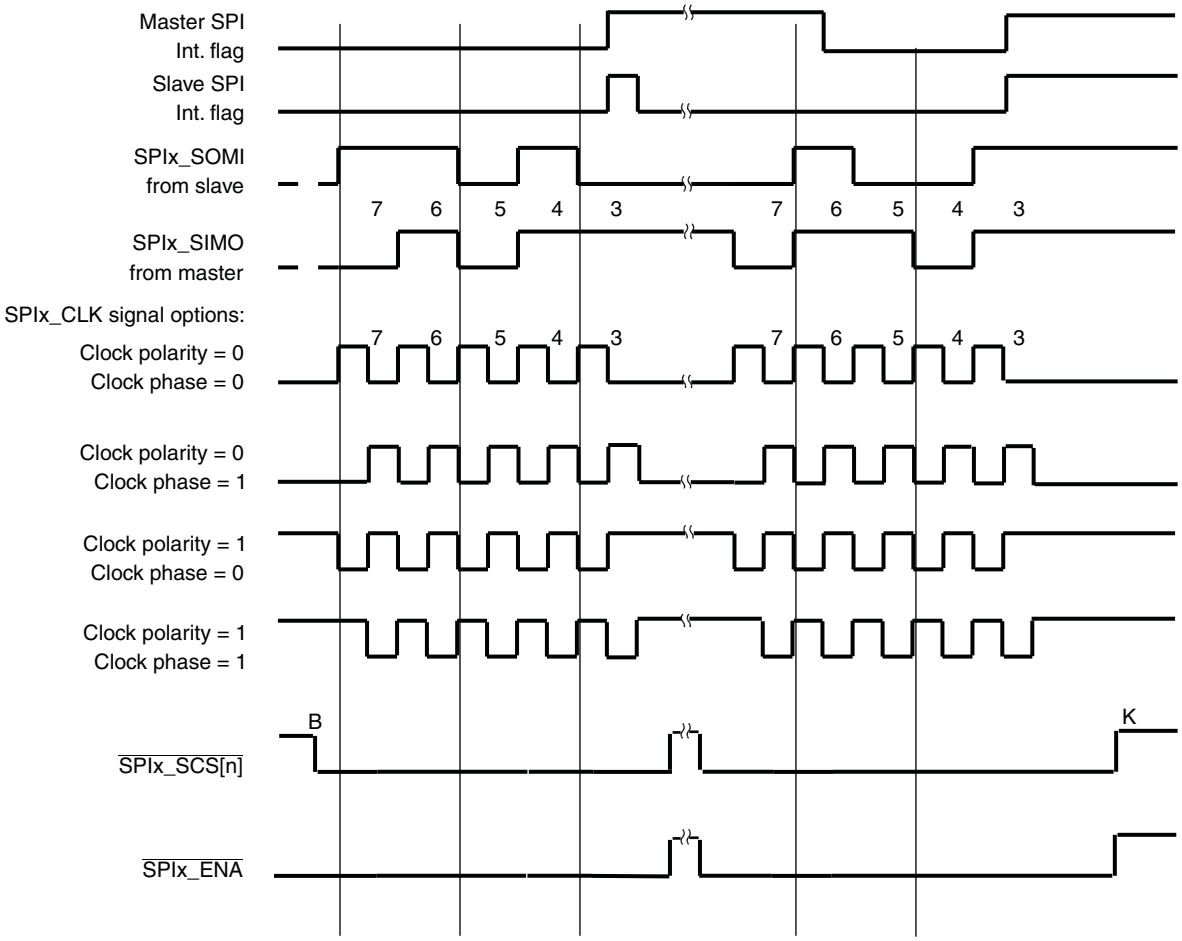
Clock phase = 1 (SPIx_CLK with delay)
- Data is output one-half cycle before the first falling edge of SPIx_CLK and on the subsequent rising edges of SPIx_CLK
- Input data is latched on the falling edge of SPIx_CLK

Copyright © 2013–2016, Texas Instruments Incorporated

20.2.11.4 SPI Data Transfer Example

Figure 20-12 illustrates an SPI data transfer between two devices using a character length of five bits.

Figure 20-12. Five Bits per Character (5-Pin Option)



20.2.12 Interrupt Support

The SPI interrupt system is controlled by three registers:

- The SPI interrupt level register (SPILVL) controls the interrrupt level. The interrupt level must be set to select the level one interrupt (INT1).
- The SPI interrupt register (SPIINT) contains bits to selectively enable/disable each interrupt event.
- The SPI flag register (SPIFLG) contains flags indicating the interrupt conditions that have occurred.

To identify the interrupt source in the SPI peripheral, the CPU reads the SPI flag status register (SPIFLG) or the INTVECT1 code in the SPI interrupt vector register 1 (INTVEC1).

Check your device-specific data manual for details on the exact CPU interrupt numbers assigned to the SPI interrupts.

Copyright © 2013–2016, Texas Instruments Incorporated

20.2.13 DMA Events Support

If handling the SPI message traffic on a character-by-character basis requires too much CPU overhead, then the CPU can configure the system DMA to handle the SPI data transfer.

The SPI module has two DMA synchronization event outputs for receive (REVT) and transmit (XEVT), allowing DMA transfers to be triggered by SPI read receive and write transmit events. The SPI module enables DMA requests by enabling the DMA request enable (DMAREQEN) bit in the SPI interrupt register (SPIINT0).

When a character is to be transmitted the SPI module signals the DMA via the XEVT signal. The DMA controller then transfers the data from the source buffer into the SPI transmit data register (SPIDAT1). When a character is received, the SPI module signals the DMA via the REVT signal. The DMA controller then reads the data from the SPI receive buffer register (SPIBUF) and transfers it to a destination buffer for ready access.

In most cases, if the DMA is being used to service received data from the SPI, the receive interrupt enable (RXINTEN) bit in SPIINT0 should be cleared to 0. This prevents the CPU from responding to the received data in addition to the DMA. For specific SPI synchronization event number assignments and detailed DMA features, see your device-specific data manual.

20.2.14 Robustness Features

The SPI module includes many features to make the SPI communication link robust. A internal loopback test mode can be used to facilitate a power on self test routine. Additionally, the SPI master continually monitors the bus for faults on its data line. The handshaking between master and slave can be monitored as well, and appropriate actions can be taken (interrupt, timeout) when the handshake breaks down. The following sections describe these robustness features in more detail.

20.2.14.1 SPI Internal Loopback Test Mode (Master Only)

CAUTION

The internal loop-back self-test mode should not be entered during a normal data transaction or unpredictable operation may occur.

To select the loopback mode, the SPIx_CLK, SPIx_SOMI, SPIx_SIMO pins should be configured as functional pins by configuring the SPI pin control register 0 (SPIPC0) and by setting the LOOPBACK bit in the SPI global control register 1 (SPIGCR1). The $\overline{\text{SPIx_ENA}}$ and $\overline{\text{SPIx_SCS[n]}}$ pins can be used as general-purpose I/O pins by configuring the SPIPC1 through SPIPC5 registers. The internal loop-back self-test mode can be utilized to test the SPI transmit path and receive path including the transmit and receive buffers. In this mode, the transmit signal is internally fed back to the receiver and the SPIx_SIMO, SPIx_SOMI, and SPIx_CLK pins are in a high-impedance state. This mode allows the CPU to write into the transmit buffer, and check that the receive buffer contains the correct transmit data. If an error occurs the corresponding error is set within the status field.

20.2.14.2 SPI Transmission Continuous Self-Test

During a data transfer, the SPI inputs the value from its data output pin on the appropriate SPIx_CLK edge. This value is compared against the expected value and any difference indicates a fault on the SPI bus. If a fault is detected, then the BITERR bit in the SPI receive buffer register (SPIBUF) and the BITERRFLG bit in the SPI flag register (SPIFLG) are set and an error interrupt is generated if enabled. The SPI continuous self-test mode is not available in SPI loopback mode.

Copyright © 2013–2016, Texas Instruments Incorporated

20.2.14.3 SPI Detection of Slave Desynchronization

In the 4-pin with enable and 5-pin modes, the SPI master can monitor the slave $\overline{\text{SPIx_ENA}}$ activity to detect a desynchronization event.

Some conditions that may cause a desynchronization event are:

- Master or slave device being reset during a transmission.
- Asserting a software reset of the SPI module during transmission.
- Having an incorrect SPI pin configuration, causing the $\overline{\text{SPIx_ENA}}$ pin to behave incorrectly.
- Signal integrity problem causing additional clocks to be recognized by the slave.

The master can detect two desynchronization error conditions on the $\overline{\text{SPIx_ENA}}$ pin:

1. Slave deasserts $\overline{\text{SPIx_ENA}}$ after a transmission has begun, but before it completes.
2. Slave fails to deassert $\overline{\text{SPIx_ENA}}$ within a certain time period after the completion of the last bit of the transmission.

The first error condition is straightforward to detect. To detect the second error condition, the SPI module includes an eight-bit counter with a timeout count that can be configured through the T2EDELAY field in the SPI delay register (SPIDELAY).

When a desynchronization event is detected, the DESYNC bit in the SPI receive buffer register (SPIBUF) and the DESYNCFLG bit in the SPI flag register (SPIFLG) are set and a desynchronization error interrupt is asserted if enabled.

NOTE: Remember that even though the desynchronization is detected by the master device, the problem causing the desynchronization event can be on either the master or the slave device.

The T2EDELAY period begins once the T2CDELAY period terminates or after the data shifting period in case the T2CDELAY is disabled. It defines the maximum time for the slave to deassert the $\overline{\text{SPIx_ENA}}$ signal. If the slave device does not deassert the $\overline{\text{SPIx_ENA}}$ signal before the T2EDELAY timeout value expires, the SPIFLG.DESYNC flag is set and a desynchronization interrupt is asserted if enabled. The T2E delay period does not always complete, sometimes it is skipped or terminated early. The T2E delay period terminates immediately after the $\overline{\text{SPIx_ENA}}$ input is sampled (using the SPI module clock at intervals of SPIFMT*n*.PRESCALE + 2) as deasserted. However, assuming the T2E period completes its duration is specified by:

Maximum duration of T2EDELAY period = SPIDELAY.T2EDELAY + SPIFMTn.PRESCALE + 2 (SPI module clock cycles)

The T2EDELAY period is enabled only when the $\overline{\text{SPIx_ENA}}$ is asserted at the beginning of the T2E delay period, the SPIDELAY.T2EDELAY field has a non-zero value, and SPIFMT*n*.WAITENA bit is set to 1.

20.2.14.4 $\overline{\text{SPIx_ENA}}$ Signal Time-Out

In 5-pin mode, in addition to the slave desynchronization detection, the master can also detect whether the slave fails to respond to the $\overline{\text{SPIx_SCS[n]}}$ signal by asserting $\overline{\text{SPIx_ENA}}$ in a timely manner.

This condition could be the result of a serious error, or it could simply be the result of the slave device taking too long to service its SPI.

To detect this condition, the C2EDELAY field in the SPI delay register (SPIDELAY) is used. The C2EDELAY period begins once the C2TDELAY period terminates or when the master asserts $\overline{\text{SPIx_SCS[n]}}$ (if C2TDELAY is disabled). It defines the maximum time for the addressed slave to respond by activating the $\overline{\text{SPIx_ENA}}$ signal. If the slave does not respond with the $\overline{\text{SPIx_ENA}}$ signal before the timeout value expires, then the TIMEOUT bit in the SPI receive buffer register (SPIBUF) and the TIMEOUTFLG bit in the SPI flag register (SPIFLG) are set, an interrupt is asserted if enabled, and the current transfer is terminated. The C2E delay period does not always complete, sometimes it is skipped or terminated early. The C2E delay period terminates immediately after the $\overline{\text{SPIx_ENA}}$ input is sampled (using the SPI module clock at intervals of SPIFMT*n*.PRESCALE + 2) as asserted. However, assuming the C2E period completes its duration is specified by:

Copyright © 2013–2016, Texas Instruments Incorporated

Maximum duration of C2EDELAY period = SPIDELAY.C2EDELAY + SPIFMTn.PRESCALE + 2 (SPI module clock cycles)

The C2EDELAY period is enabled only when the $\overline{\text{SPIx_ENA}}$ is deasserted at the beginning of the C2E delay period and SPIFMT*n*.WAITENA bit is set to 1. If SPIFMT*n*.WAITENA bit is set to 1 and C2EDELAY is cleared to 0, then the master waits indefinitely for the slave to assert $\overline{\text{SPIx_ENA}}$.

20.2.14.5 SPI Data Length Error

An SPI can generate an error flag by detecting any mismatch in length of received/transmitted data with the programmed character length under certain conditions.

Master Mode: During a data transfer, if the SPI detects a deassertion of the $\overline{\text{SPIx_ENA}}$ pin (by the slave) while the character counter is not overflowed, then an error flag is set indicating the data length error. This can be caused by a slave receiving extra clocks (because of noise on the SPIx_CLK line).

NOTE: In SPI master mode, the data length error will be generated only if the $\overline{\text{SPIx_ENA}}$ pin is used as a functional pin.

Slave Mode: During a transfer, if the SPI detects a deassertion of the $\overline{\text{SPIx_SCS[n]}}$ pin before its character length counter overflows, then an error flag is set indicating the data length error. If the slave SPI misses one or more SPIx_CLK pulses from the master, this situation can occur. This error in slave mode would mean that both the transmitted and received data were not complete.

NOTE: In SPI slave mode, the data length error flag will be generated only if the $\overline{\text{SPIx_SCS[n]}}$ pin is configured as a functional pin.

20.2.15 Reset Considerations

This section describes the software and hardware reset considerations.

20.2.15.1 Software Reset Considerations

The SPI module contains a software reset (RESET) bit in the SPI global control register 0 (SPIGCR0) that is used to reset the SPI module. As a result of a reset, the SPI module register values go to their reset state. The RESET bit must be set before any operation on the SPI is done.

20.2.15.2 Hardware Reset Considerations

In the event of a hardware reset, the SPI module register values go to their reset state and the application software needs to reprogram the registers to the desired values.

20.2.16 Power Management

The SPI module can be put in either local or global low-power mode. Global low-power mode is asserted by the system and is not controlled by the SPI. During global low-power mode, all clocks to the SPI are turned off so the module is completely inactive.

The SPI local low-power mode is asserted by setting the POWERDOWN bit in the SPI global control register 1 (SPIGCR1). Setting this bit stops the clocks to the SPI internal logic and the SPI registers. Setting the POWERDOWN bit causes the SPI to enter local low-power mode and clearing the POWERDOWN bit causes SPI to exit from local low-power mode. All the registers are accessible during local power-down mode as any register access enables the clock to SPI for that particular access alone.

Since entering a low-power mode has the effect of suspending all state machine activities, care must be taken when entering such modes to ensure that a valid state is entered when low-power mode is active. As a result, application software must ensure that a low-power mode is not entered during a transmission or reception of data.

Copyright © 2013–2016, Texas Instruments Incorporated

20.2.17 General-Purpose I/O Pin

Each of the SPI pins may be programmed via the SPI pin control registers (SPIPC0 to SPIPC5) to be a general-purpose I/O (GPIO) pin.

When the SPI pins are not used as functional pins, they may be programmed to be either general input or general output pins by configuring SPIPC0. For example, in 3-pin mode, SPIx_SOMI, SPIx_SIMO, and SPIx_CLK must be configured as SPI pins, while the SPIx_SCS[n] and SPIx_ENA pins should be configured as GPIO pins. The direction is controlled by configuring SPIPC1.

If configured as a general-purpose output, then SPIPC3 controls the output value. There is also a write 1 to set (SPIPC4) and a write 1 to clear (SPIPC5) for the data out value. These registers allow different tasks running on the CPU to manipulate the SPI I/O pins without read-modify-write hazards.

SPIPC2 reflects the current value on the pin when the particular pin is configured as a functional or general-purpose input pin. When the pin is configured as a functional or general-purpose output pin, SPIPC2 indicates the value that is attempted to be driven on the pin.

20.2.18 Emulation Considerations

CAUTION

Viewing or otherwise reading the following SPI registers: SPIBUF, SPIFLG, and INTVEC1 through the JTAG debugger causes their contents to change, possibly invalidating the results of the debug session. Be sure to set up the debugger to avoid reading these registers.

The SPI module does not support soft or hard stop during emulation breakpoints. The SPI module will continue to run if an emulation breakpoint is encountered.

In addition, any status registers that are cleared after reading will be affected if viewed in a memory or watch window of the debugger; since the emulator will read these registers to update the value displayed in the window.

20.2.19 Initialization

Perform the following procedure for initializing the SPI:

1. Reset the SPI by clearing the RESET bit in the SPI global control register 0 (SPIGCR0) to 0.
2. Take the SPI out of reset by setting SPIGCR0.RESET to 1.
3. Configure the SPI for master or slave mode by configuring the CLKMOD and MASTER bits in the SPI global control register 1 (SPIGCR1).
4. Configure the SPI for 3-pin, 4-pin with chip select, 4-pin with enable, or 5-pin mode by configuring the SPI pin control register 0 (SPIPC0).
5. Chose the SPI data format register *n* (SPIFMT*n*) to be used by configuring the DFSEL bit in the SPI transmit data register (SPIDAT1). In slave mode, only SPIFMT0 is supported.
6. Configure the SPI data rate, character length, shift direction, phase, polarity and other format options using SPIFMT*n* selected in step 5.
7. If SPI master, then configure the master delay options using the SPI delay register (SPIDELAY). In slave mode, SPIDELAY is not relevant.
8. Select the error interrupt notifications by configuring the SPI interrupt register (SPIINT0) and the SPI interrupt level register (SPILVL).
9. Enable the SPI communication by setting the SPIGCR1.ENABLE to 1.
10. Setup and enable the DMA for SPI data handling and then enable the DMA servicing for the SPI data requests by setting the SPIINT0.DMAREQEN to 1.
11. Handle SPI data transfer requests using DMA and service any SPI error conditions using the interrupt service routine.

Copyright © 2013–2016, Texas Instruments Incorporated

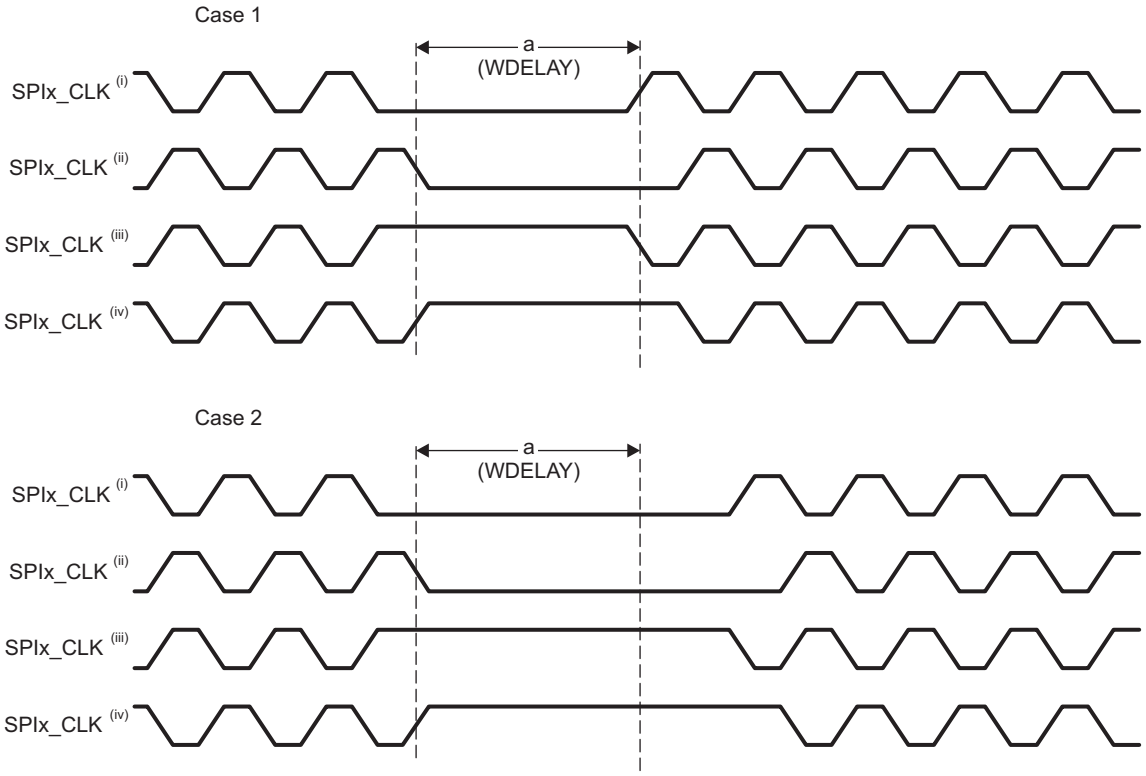
20.2.20 Timing Diagrams

This section contains timing diagrams illustrating the C2TDELAY, C2EDELAY, T2CDELAY, T2EDELAY, and WDELAY delays and their interaction with the SPIx_SCS[n] and SPIx_ENA pins for all SPI modes.

20.2.20.1 SPI 3-Pin Mode

Figure 20-13 illustrates the WDELAY option in SPI 3-pin master mode. This is the only delay available in this mode. In CASE1, a new transfer is initiated during the WDELAY period and the transfer begins immediately after the WDELAY period ends. In CASE2, while WDELAY has completed, a new transfer will not begin until SPIDAT0/SPIDAT1 have been written with new data.

Figure 20-13. SPI 3-Pin Master Mode with WDELAY

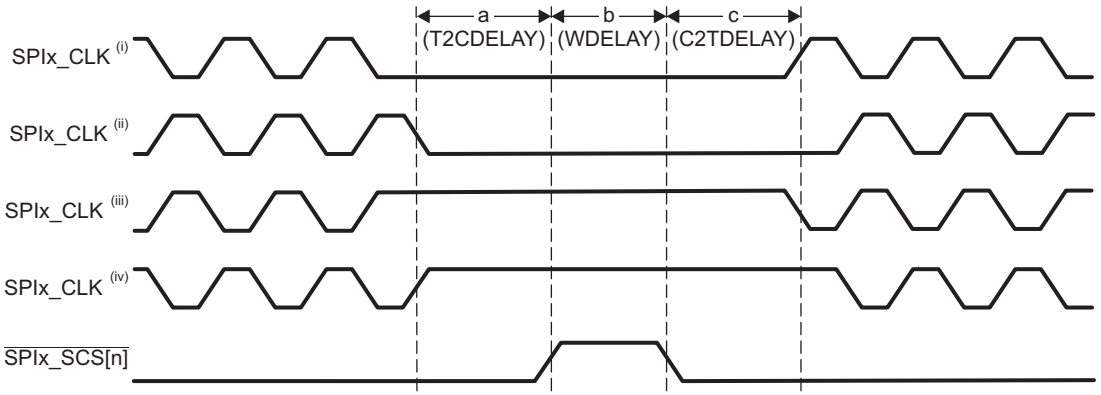


Copyright © 2013–2016, Texas Instruments Incorporated

20.2.20.2 SPI 4-Pin with $\overline{\text{SPIx_SCS[n]}}$ Mode

Figure 20-14 illustrates the T2CDELAY, WDELAY and C2TDELAY delays in SPI 4-pin with $\overline{\text{SPIx_SCS[n]}}$ master mode. C2EDELAY and T2EDELAY are not available in this mode. All the three delay periods T2CDELAY, WDELAY, and C2TDELAY proceed to completion when enabled.

Figure 20-14. SPI 4-Pin with $\overline{\text{SPIx_SCS[n]}}$ Mode with T2CDELAY, WDELAY, and C2TDELAY



20.2.20.3 SPI 4-Pin with $\overline{\text{SPIx_ENA}}$ Mode

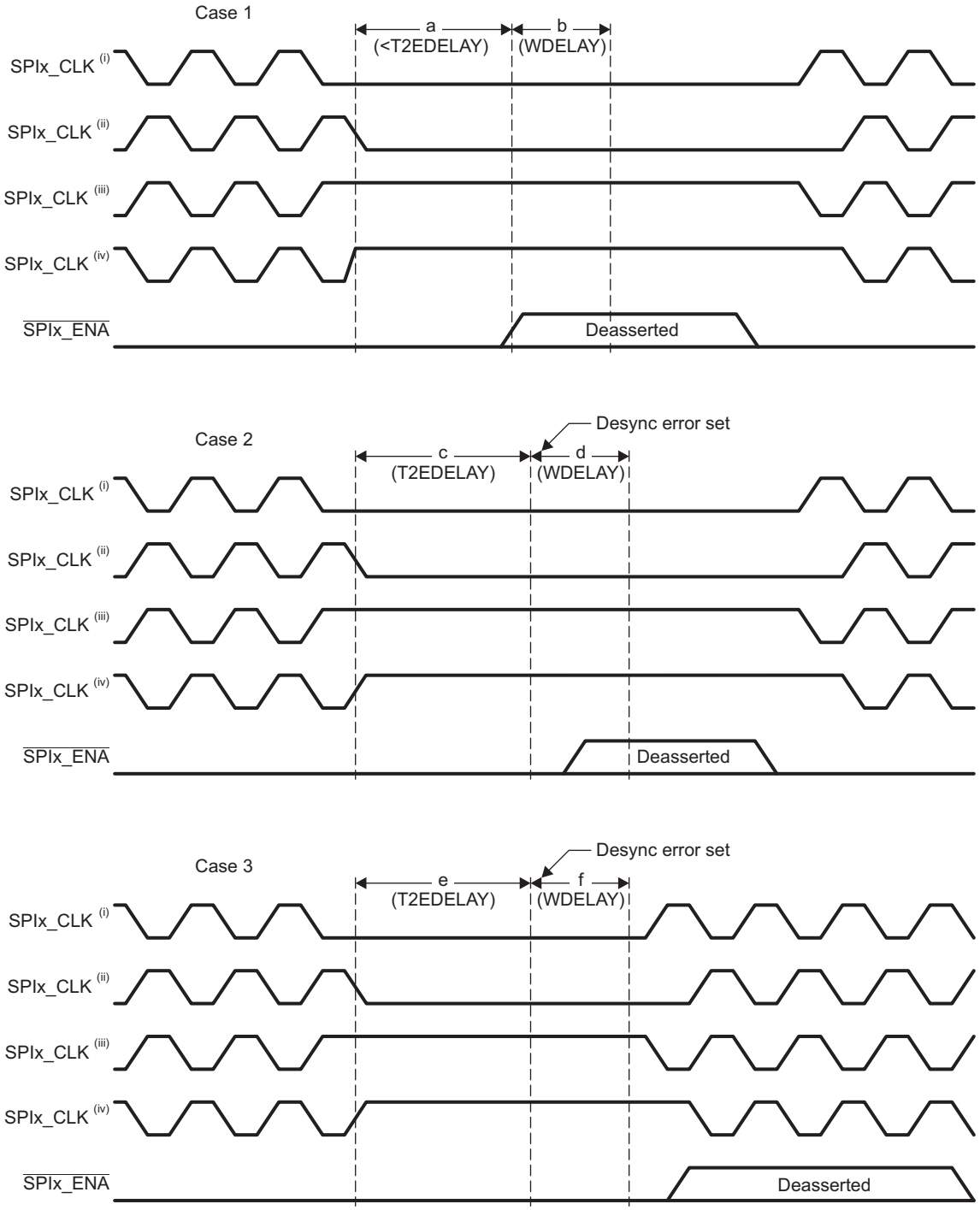
Figure 20-15 shows the T2EDELAY and WDELAY delays in SPI 4-pin with $\overline{\text{SPIx_ENA}}$ master mode. T2CDELAY, C2TDELAY, and C2EDELAY are not available in this mode.

- In CASE1, the $\overline{\text{SPIx_ENA}}$ is deasserted during the T2EDELAY period. Consequently the T2EDELAY period is terminated early (a) and the WDELAY period begins immediately (b) if enabled. The next transfer is initiated as soon as the slave asserts $\overline{\text{SPIx_ENA}}$ again.
- In CASE2, the T2EDELAY period (c) completes before the $\overline{\text{SPIx_ENA}}$ is deasserted. As a result the DESYNC error is set. However since the $\overline{\text{SPIx_ENA}}$ is deasserted during the WDELAY period (d), the master delays the next transfer until the $\overline{\text{SPIx_ENA}}$ is asserted again.
- In CASE3, the T2EDELAY (e) and WDELAY (f) period (if enabled) both expire before the $\overline{\text{SPIx_ENA}}$ input is deasserted. The DESYNC error is set at the end of the T2EDELAY period (e). However in this case the master begins the next transfer immediately after it is initiated and ignores the $\overline{\text{SPIx_ENA}}$ during the transfer even if it is subsequently deasserted.

  If the T2EDELAY delay period is disabled then the DESYNC error is not set. The SPI master behavior in this case depends on whether the $\overline{\text{SPIx_ENA}}$ gets deasserted during the WDELAY period (CASE2) or $\overline{\text{SPIx_ENA}}$ gets deasserted after the WDELAY period completes (CASE3).

Copyright © 2013–2016, Texas Instruments Incorporated

Figure 20-15. SPI 4-Pin with $\overline{\text{SPIx_ENA}}$ Mode Demonstrating T2EDELAY and WDELAY



Copyright © 2013–2016, Texas Instruments Incorporated

20.2.20.4 SPI 5-Pin Mode

Figure 20-16 shows the T2CDELAY, T2EDELAY, and WDELAY delays in SPI 5-pin master mode.

- In CASE1, the SPIx_ENA is deasserted during the T2CDELAY period. However the T2CDELAY period proceeds to completion(a), the T2EDELAY period is skipped (if enabled) and the WDELAY period begins immediately (b) (if enabled). The next transfer is initiated as soon as the slave asserts SPIx_ENA again.
- In CASE2, the SPIx_ENA signal is deasserted by the slave during the T2EDELAY period (d) which begins upon the completion of the T2CDELAY period (c). The deassertion of the SPIx_ENA causes the T2EDELAY period to terminate early and the WDELAY period (e) begins immediately (if enabled) after the T2EDELAY period terminates. The next transfer is initiated as soon as the slave asserts SPIx_ENA again.
- In CASE3, the SPIx_ENA signal is deasserted by the slave during the WDELAY period (h) which begins upon the completion of the T2CDELAY period (f) and T2EDELAY period (g). As a result the DESYNC error is set at the end of the T2EDELAY period (g). However since the SPIx_ENA is deasserted during the WDELAY period (h), the master delays the next transfer until the SPIx_ENA is asserted again.
- In CASE4, the SPIx_ENA signal is not deasserted until after the completion of the T2CDELAY (j), T2EDELAY (k) and WDELAY (m) (if enabled) periods. The DESYNC error is set at the end of the T2EDELAY period (k). However in this case the master begins the next transfer immediately after it is initiated and ignores the SPIx_ENA during the transfer even if it is subsequently deasserted.

  If the T2EDELAY delay period is disabled then the DESYNC error is not set. The SPI master behavior in this case depends on whether the SPIx_ENA gets deasserted during the T2CDELAY period (CASE1), WDELAY period (CASE3) or after the WDELAY period completes (CASE4).

  If the slave deasserts the SPIx_ENA signal before the completion of the configured master delays (T2CDELAY, T2EDELAY, WDELAY) then the master delays the next transfer until the slave asserts the SPIx_ENA again. However if the slave delays the SPIx_ENA deassertion until after the completion of the configured master delays then the master begins the next transfer immediately after it is initiated and ignores the SPIx_ENA during the transfer even if it is subsequently deasserted.

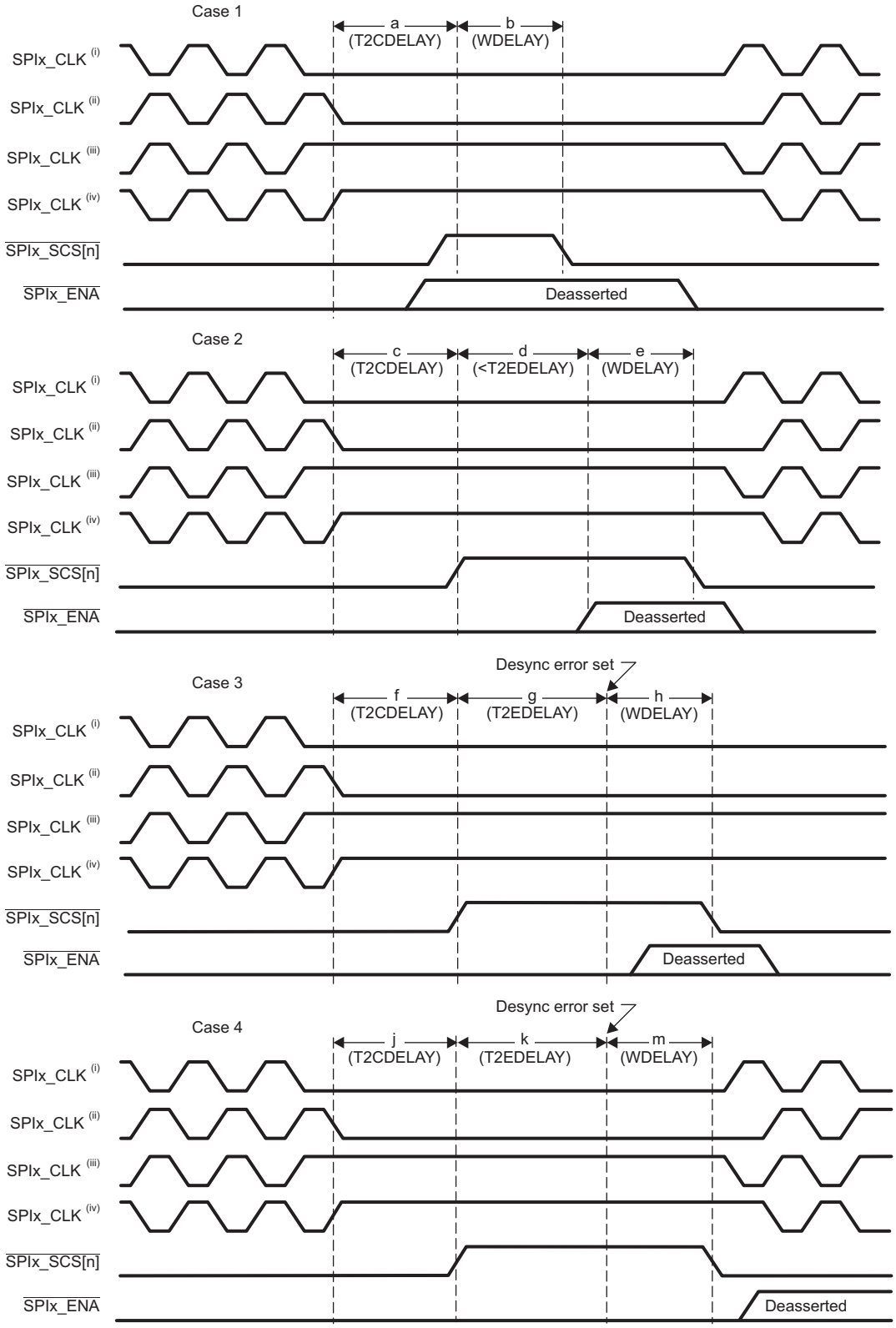
Figure 20-17 shows the C2TDELAY and C2EDELAY in SPI 5-pin master mode.

- In CASE1, the SPIx_ENA signal is asserted during the C2TDELAY period (a). However the C2TDELAY period proceeds to completion(a), the C2EDELAY period is skipped (if enabled) and the master begins generating the SPI clock for transmission.
- In CASE2, the SPIx_ENA signal is asserted during the C2EDELAY period (d) which begins upon the completion of C2TDELAY period (c). The assertion of the SPIx_ENA causes the C2EDELAY period to terminate early and the master begins generating the SPI clock for transmission.
- In CASE3, the SPIx_ENA signal is not asserted until after the completion of the C2TDELAY (f) and C2EDELAY (g) periods. The TIMEOUT error is set at the end of the C2EDELAY period (g). The master deasserts the SPIx_SCS[n] signal immediately and clears the current transmit request.

  If the C2EDELAY delay period is disabled then the SPI master behavior depends on whether the SPIx_ENA gets asserted during the C2TDELAY period (CASE1) or after the C2TDELAY period completes (CASE2). In latter case there is no limit on how long the master will wait for the slave to respond with SPIx_ENA asserted and hence there is no limit on period 'd' shown in CASE2. Thus when C2EDELAY period is disabled the TIMEOUT error is not set.

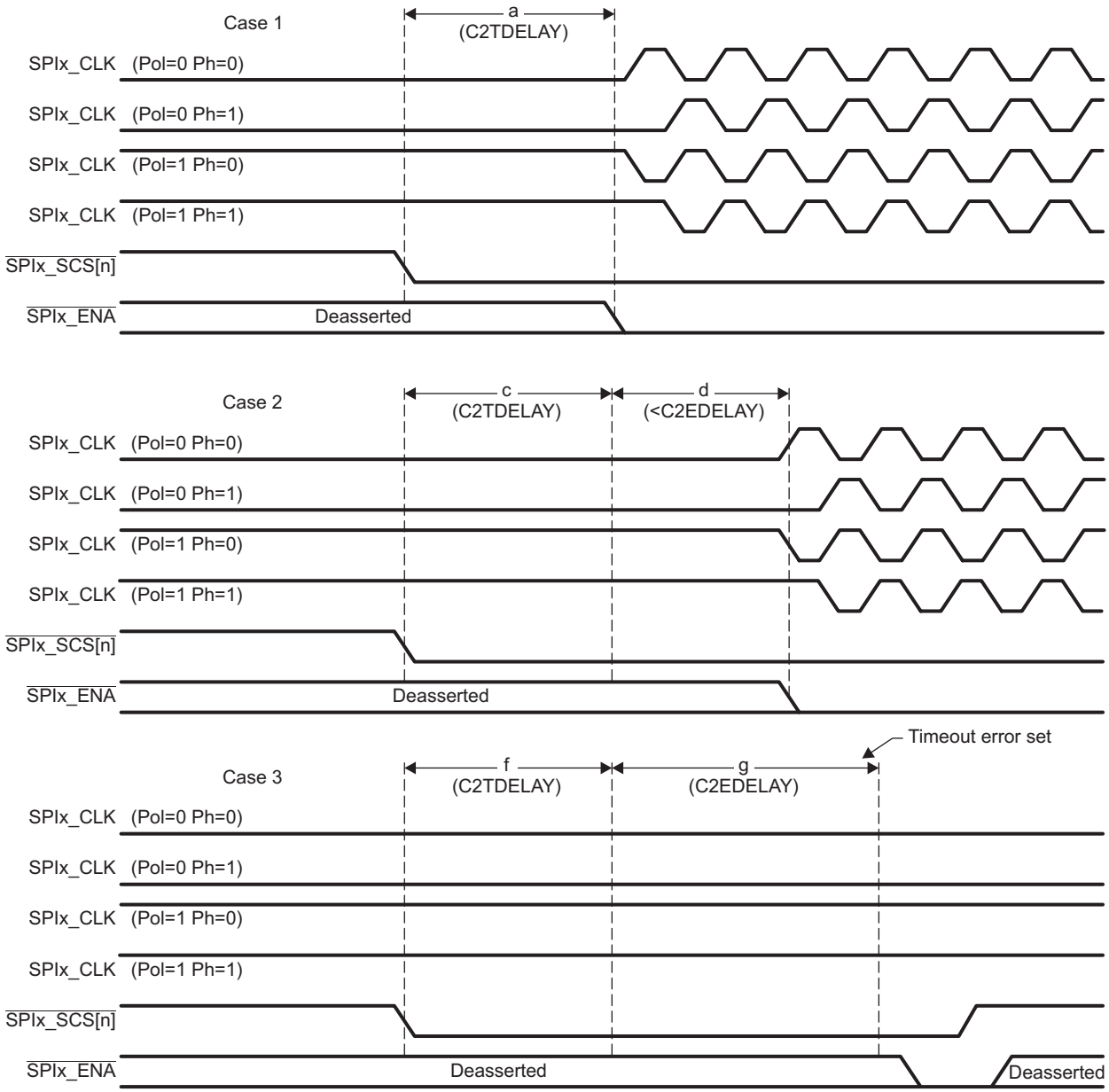
Copyright © 2013–2016, Texas Instruments Incorporated

Figure 20-16. SPI 5-Pin Mode Demonstrating T2CDELAY, T2EDELAY, and WDELAY



Copyright © 2013–2016, Texas Instruments Incorporated

Figure 20-17. SPI 5-Pin Mode Demonstrating C2TDELAY and C2EDELAY



Copyright © 2013–2016, Texas Instruments Incorporated

20.3 Registers

This section describes the SPI control, data, and pin registers. The offset is relative to the associated base address of the module. See your device-specific data manual for the memory address of these registers.

Table 20-8. SPI Registers

Offset Address	Acronym	Register Description	Section
0h	SPIGCR0	SPI Global Control Register 0	Section 20.3.1
4h	SPIGCR1	SPI Global Control Register 1	Section 20.3.2
8h	SPIINT0	SPI Interrupt Register	Section 20.3.3
Ch	SPILVL	SPI Interrupt Level Register	Section 20.3.4
10h	SPIFLG	SPI Flag Register	Section 20.3.5
14h	SPIPC0	SPI Pin Control Register 0 (Function)	Section 20.3.6
18h	SPIPC1	SPI Pin Control Register 1 (Direction)	Section 20.3.7
1Ch	SPIPC2	SPI Pin Control Register 2 (Input)	Section 20.3.8
20h	SPIPC3	SPI Pin Control Register 3 (Output)	Section 20.3.9
24h	SPIPC4	SPI Pin Control Register 4 (Set SPIPC3)	Section 20.3.10
28h	SPIPC5	SPI Pin Control Register 5 (Clear SPIPC3)	Section 20.3.11
38h	SPIDAT0	SPI Data Transmit Register 0	Section 20.3.12
3Ch	SPIDAT1	SPI Data Transmit Register 1 (Data Transmit and Format Select)	Section 20.3.13
40h	SPIBUF	SPI Receive Buffer Register	Section 20.3.14
44h	SPIEMU	SPI Receive Emulation Register	Section 20.3.15
48h	SPIDELAY	SPI Delay Register	Section 20.3.16
4Ch	SPIDEF	SPI Default Chip Select Register	Section 20.3.17
50h	SPIFMT0	SPI Data Format Register 0	Section 20.3.18
54h	SPIFMT1	SPI Data Format Register 1	Section 20.3.18
58h	SPIFMT2	SPI Data Format Register 2	Section 20.3.18
5Ch	SPIFMT3	SPI Data Format Register 3	Section 20.3.18
64h	INTVEC1	SPI Interrupt Vector Register 1	Section 20.3.19

20.3.1 SPI Global Control Register 0 (SPIGCR0)

The SPI global control register 0 (SPIGCR0) is shown in Figure 20-18 and described in Table 20-9.

Figure 20-18. SPI Global Control Register 0 (SPIGCR0)

31	16
Reserved	
R-0	

15	1	0
Reserved		RESET
R-0		R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 20-9. SPI Global Control Register 0 (SPIGCR0) Field Descriptions

Bit	Field	Value	Description
31-1	Reserved	0	Reads return zero and writes have no effect.
0	RESET		Reset bit for the module. This bit needs to be set to 1 before any operation on SPI can be done.
		0	SPI is in reset state.
		1	SPI is out of reset state.

Copyright © 2013–2016, Texas Instruments Incorporated

20.3.2 SPI Global Control Register 1 (SPIGCR1)

The SPI global control register 1 (SPIGCR1) is shown in Figure 20-19 and described in Table 20-10.

Figure 20-19. SPI Global Control Register 1 (SPIGCR1)

31	25	24
Reserved		ENABLE
R-0		R/W-0

23	17	16
Reserved		LOOPBACK
R-0		R/W-0

15	9	8
Reserved		POWERDOWN
R-0		R/W-0

7	2	1	0
Reserved		CLKMOD	MASTER
R-0		R/W-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 20-10. SPI Global Control Register 1 (SPIGCR1) Field Descriptions

Bit	Field	Value	Description
31-25	Reserved	0	Reads return zero and writes have no effect.
24	ENABLE		SPI enable. This bit enables the SPI transfers. The other SPI configuration registers except SPIINT0.DMAREQEN should be configured before writing a 1 to this bit. This will prevent the SPI from responding to bus operations erroneously while it is in the process of being configured. The SPIINT0.DMAREQEN should be enabled after setting ENABLE. If SPIINT0.DMAREQEN is enabled before setting ENABLE then the first DMA request that occurs before the SPI is ready for data transfer may get dropped. When ENABLE bit is cleared to 0, the following SPI registers get forced to their default states (to 0s except for RXEMPTY bit in SPIBUF): • Both TX and RX shift registers • The TXDATA fields of SPIDAT0 and SPIDAT1 registers • All the fields of the SPIFLG register • Contents of SPIBUF and the internal RXBUF registers
		0	SPI is not activated for transfers.
		1	Activates SPI.
23-17	Reserved	0	Reads return zero and writes have no effect.
16	LOOPBACK		Internal loop-back test mode. The internal self-test option can be enabled by setting this bit. If the SPIx_SIMO and SPIx_SOMI pins are configured with SPI functionality, then the SPIx_SIMO pin is internally connected to the SPIx_SOMI pin. The transmit data is looped back as receive data and is stored in the receive field of the concerned buffer. Externally, during loop-back operation, the SPIx_CLK pin outputs an inactive value, SPIx_SIMO and SPIx_SOMI pins remain in high-impedance state. The SPI has to be initialized in master mode before the loop-back can be selected. If the SPI is initialized in slave mode or a data transfer is ongoing, errors may result.
		0	Internal loop-back test mode disabled.
		1	Internal loop-back test mode enabled.
15-9	Reserved	0	Reads return zero and writes have no effect.
8	POWERDOWN		When active, the SPI state machine enters a power-down state.
		0	The SPI is in active mode.
		1	The SPI is in power-down mode.
7-2	Reserved	0	Reads return zero and writes have no effect.

Copyright © 2013–2016, Texas Instruments Incorporated

Table 20-10. SPI Global Control Register 1 (SPIGCR1) Field Descriptions (continued)

Bit	Field	Value	Description
1-0	CLKMOD,MASTER	0-3h	These two bits (CLKMOD,MASTER) determine whether the SPI operates in master or slave mode.
		0	SLAVE MODE. SPIx_CLK is an input from the master who initiates the transfers. Data is transmitted on the SPIx_SOMI pin and received on the SPIx_SIMO pin. The $\overline{\text{SPIx_SCS[n]}}$ pin is an input pin if configured as SPI slave chip select. The $\overline{\text{SPIx_ENA}}$ pin is an output pin if configured as the SPI enable pin.
		1h-2h	Reserved
		3h	MASTER MODE. SPIx_CLK is an output and the SPI initiates transfers. Data is transmitted on the SPIx_SIMO pin and received on the SPIx_SOMI pin. The $\overline{\text{SPIx_SCS[n]}}$ pin is an output pin if configured as SPI slave chip select. The $\overline{\text{SPIx_ENA}}$ pin is an input pin if configured as the SPI enable pin.

Copyright © 2013–2016, Texas Instruments Incorporated

20.3.3 SPI Interrupt Register (SPIINT0)

The SPI interrupt register (SPIINT0) is shown in Figure 20-20 and described in Table 20-11.

Figure 20-20. SPI Interrupt Register (SPIINT0)

31	25	24
Reserved		ENABLEHIGHZ
R-0		R/W-0

23	17	16
Reserved		DMAREQEN
R-0		R/W-0

15	10	9	8
Reserved		TXINTENA	RXINTENA
R-0		R/W-0	R/W-0

7	6	5	4	3	2	1	0
Reserved	OVRNINTENA	Reserved	BITERRENA	DESYNCENA	PARERRENA	TIMEOUTENA	DLENERRENA
R-0	R/W-0	R-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 20-11. SPI Interrupt Register (SPIINT0) Field Descriptions

Bit	Field	Value	Description
31-25	Reserved	0	Reads return zero and writes have no effect.
24	ENABLEHIGHZ		$\overline{\text{SPIx_ENA}}$ pin high-impedance enable. If ENABLEHIGHZ is enabled, the $\overline{\text{SPIx_ENA}}$ pin (when it is configured as a WAIT functional output signal in a slave SPI) is forced to place it is output in high-impedance when not driving a low signal. If ENABLEHIGHZ is disabled, then the pin will output both a high and a low signal.
		0	$\overline{\text{SPIx_ENA}}$ pin is pulled high when not active.
		1	$\overline{\text{SPIx_ENA}}$ pin remains in high-impedance when not active.
23-17	Reserved	0	Reads return zero and writes have no effect.
16	DMAREQEN		DMA request enable. Enables the DMA request signal to be generated for both receive and transmit channels. Set DMAREQEN only after setting the SPIGCR1.ENABLE bit to 1.
		0	DMA is not used.
		1	DMA requests will be generated.
			Note: A transmit DMA request will be generated each time a transmit data is copied to the shift register either from TXBUF or directly from SPIDAT0/SPIDAT1.
			Note: A receive DMA request will be generated each time a received data is copied to SPIBUF register either from RXBUF or directly from the shift register.
15-10	Reserved	0	Reads return zero and writes have no effect.
9	TXINTENA		An interrupt is to be generated every time data is written to the shift register, so that a new data can be written to TXBUF. Setting this bit will generate an interrupt if the SPIFLG.TXINTFLG bit is set to 1.
		0	No interrupt will be generated upon SPIFLG.TXINTFLG being set to 1.
		1	Interrupt will be generated upon SPIFLG.TXINTFLG being set to 1.
8	RXINTENA		Receive interrupt enable. An interrupt is to be generated when the SPIFLG.RXINTFLAG bit is set.
		0	Interrupt will not be generated.
		1	Interrupt will be generated.
7	Reserved	0	Reads return zero and writes have no effect.
6	OVRNINTENA		Overrun interrupt enable. An interrupt is to be generated when the SPIFLG.OVRNINTFLG bit is set. The overrun interrupt is not useful if receive data is serviced with CPU interrupts because the overrun and receive events share a common level interrupt signal.
		0	Overrun interrupt will not be generated.
		1	Overrun interrupt will be generated.
5	Reserved	0	Reads return zero and writes have no effect.

Copyright © 2013–2016, Texas Instruments Incorporated

Table 20-11. SPI Interrupt Register (SPIINT0) Field Descriptions (continued)

Bit	Field	Value	Description
4	BITERRENA		Enables interrupt on bit error. An interrupt is to be generated when the SPIFLG.BITERRFLG is set.
		0	No interrupt asserted upon bit error.
		1	Enables an interrupt on a bit error.
3	DESYNCENA		Enables interrupt on desynchronized slave. DESYNCENA is used in master mode only. The desynchronization monitor is active in master mode for the 4-pin with enable and 5-pin options. An interrupt is to be generated when the SPIFLG.DESYNCFLG is set.
		0	No interrupt asserted upon desynchronization error.
		1	Enables an interrupt on desynchronization of the slave.
2	PARERRENA		Enables interrupt on parity error. An interrupt is to be generated when the SPIFLG.PARERRFLG is set.
		0	No interrupt asserted upon parity error.
		1	Enables an interrupt on a parity error.
1	TIMEOUTENA		Enables interrupt on $\overline{\text{SPIx_ENA}}$ signal time-out. An interrupt is to be generated when SPIFLG.TIMEOUTFLG is set.
		0	No interrupt asserted upon $\overline{\text{SPIx_ENA}}$ signal time-out.
		1	Enables an interrupt on a time-out of the $\overline{\text{SPIx_ENA}}$ signal.
0	DLENERRENA		Data length error interrupt enable. A data length error occurs under the following conditions. Master: In a 4-pin with $\overline{\text{SPIx_ENA}}$ mode or 5-pin mode, if the $\overline{\text{SPIx_ENA}}$ pin from the slave is deasserted before the master has completed its transfer, the data length error is set. That is, if the character length counter has not overflowed while $\overline{\text{SPIx_ENA}}$ deassertion is detected, then it means that the slave has neither received full data from the master nor has it transmitted complete data. Slave: In a 4-pin with chip select mode or 5-pin mode, if the incoming valid $\overline{\text{SPIx_SCS}}$[n] pin is deactivated before the character length counter overflows, then data length error is set.
		0	No interrupt is generated upon data length error.
		1	Enables an interrupt when data length error occurs.

Copyright © 2013–2016, Texas Instruments Incorporated

20.3.4 SPI Interrupt Level Register (SPILVL)

The SPI interrupt level register (SPILVL) is shown in Figure 20-21 and described in Table 20-12.

Figure 20-21. SPI Interrupt Level Register (SPILVL)

31	16
Reserved	
R-0	

15	10	9	8
Reserved		TXINTLVL	RXINTLVL
R-0		R/W-0	R/W-0

7	6	5	4	3	2	1	0
Reserved	OVRNINTLVL	Reserved	BITERRLVL	DESYNCLVL	PARERRLVL	TIMEOUTLVL	DLENERRLVL
R-0	R/W-0	R-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 20-12. SPI Interrupt Level Register (SPILVL) Field Descriptions

Bit	Field	Value	Description
31-10	Reserved	0	Reads return zero and writes have no effect.
9	TXINTLVL		Transmit interrupt level.
		0	Reserved
		1	Transmit interrupt is mapped to interrupt line INT1.
8	RXINTLVL		Receive interrupt level.
		0	Reserved
		1	Receive interrupt is mapped to interrupt line INT1.
7	Reserved	0	Reads return zero and writes have no effect.
6	OVRNINTLVL		Receive overrun interrupt level. The overrun interrupt is not useful if receive data is serviced with CPU interrupts because the overrun and receive events share a common level interrupt signal.
		0	Reserved
		1	Receive overrun interrupt is mapped to interrupt line INT1.
5	Reserved	0	Reads return zero and writes have no effect.
4	BITERRLVL		Bit error interrupt level.
		0	Reserved
		1	Bit error interrupt is mapped to interrupt line INT1.
3	DESYNCLVL		Desynchronized slave interrupt level. DESYNCLVL is used in master mode only.
		0	Reserved
		1	An interrupt due to desynchronization of the slave is mapped to interrupt line INT1.
2	PARERRLVL		Parity error interrupt level.
		0	Reserved
		1	A parity error interrupt is mapped to interrupt line INT1.
1	TIMEOUTLVL		$\overline{\text{SPIx_ENA}}$ signal time-out interrupt level.
		0	Reserved
		1	An interrupt on a time-out of the $\overline{\text{SPIx_ENA}}$ signal is mapped to interrupt line INT1.
0	DLENERRLVL		Data length error interrupt enable level.
		0	Reserved
		1	An interrupt on data length error is mapped to interrupt line INT1.

Copyright © 2013–2016, Texas Instruments Incorporated

20.3.5 SPI Flag Register (SPIFLG)

The SPI flag register (SPIFLG) is shown in Figure 20-22 and described in Table 20-13.

Figure 20-22. SPI Flag Register (SPIFLG)

31							16
Reserved							
R-100h							
15						10	9	8
Reserved							TXINTFLG	RXINTFLG
R-0							R-0	R/WC-0
7	6	5	4	3	2	1	0
Reserved	OVRNINTFLG	Reserved	BITERRFLG	DESYNCFLG	PARERRFLG	TIMEOUTFLG	DLENERRFLG
R-0	R/W1C-0	R-0	R/W1C-0	R/W1C-0	R/W1C-0	R/W1C-0	R/W1C-0

LEGEND: R/W = Read/Write; R = Read only; W1C = Write 1 to clear bit; -*n* = value after reset

Table 20-13. SPI Flag Register (SPIFLG) Field Descriptions

Bit	Field	Value	Description
31-10	Reserved	4000h	Reads return default value and writes have no effect.
9	TXINTFLG		Transmitter empty interrupt flag. Serves as an interrupt flag indicating that the transmit buffer (TXBUF) is empty and a new data can be written to it. This flag is set when a data is copied to the shift register either directly or from the TXBUF register. This bit is cleared by one of following ways: • Writing a new data to either SPIDAT0 or SPIDAT1 • Writing a 0 to SPIGCR1.ENABLE
		0	Transmit buffer is now full. No interrupt pending for transmitter empty.
		1	Transmit buffer is empty. An interrupt is pending to fill the transmitter.
8	RXINTFLG		Receiver full interrupt flag. This flag is set when a word is received and copied into the buffer register (SPIBUF). This bit is cleared under the following ways: • Reading the SPIBUF register. During emulation mode, however, a read to the emulation register (SPIEMU) does not clear this flag bit. • Reading INTVEC1 register when there is a receive buffer full interrupt • Writing a 1 to this bit • Writing a 0 to SPIGCR1.ENABLE • System reset
		0	No new received data pending. Receive buffer is empty.
		1	A newly received data is ready to be read. Receive buffer is full.
			Note: Clearing RXINTFLG bit by writing a 1 before reading the SPIBUF sets the RXEMPTY bit of the SPIBUF register too. This way, one can ignore a received data. However, if the internal RXBUF is already full, the data from RXBUF will be copied to SPIBUF and the RXEMPTY bit will be cleared again. The SPIBUF contents should be read first if this situation needs to be avoided.
7	Reserved	0	Reads return zero and writes have no effect.
6	OVRNINTFLG		Receiver overrun flag. The bit is set when a receive operation completes before the previous character has been read from the receive buffer. The bit indicates that the last received character has been overwritten and therefore lost. This bit is cleared under the following conditions: • Reading INTVEC1 register when there is a receive buffer overrun interrupt • Writing a 1 to this bit
		0	Overrun condition did not occur.
		1	Overrun condition has occurred.
			Note: Reading SPIBUF register does not clear the OVRNINTFLG bit. If an overrun interrupt is detected, then the SPIBUF may need to be read twice to get to the overrun buffer. This is due to the fact that the overrun will always occur to the internal RXBUF. Each read to the SPIBUF will result in RXBUF contents (if it is full) getting copied to SPIBUF. **Note:** A special condition under which OVRNINTFLG flag gets set. If both SPIBUF and RXBUF are already full and while another buffer receive is underway, if any errors like TIMEOUT, BITERR and DLENERR occur, then OVRNINTFLG will be set to indicate that the status flags are getting overwritten by the new transfer. This overrun should be treated like a normal receiver overrun.

Copyright © 2013–2016, Texas Instruments Incorporated

Table 20-13. SPI Flag Register (SPIFLG) Field Descriptions (continued)

Bit	Field	Value	Description
5	Reserved	0	Reads return zero and writes have no effect.
4	BITERRFLG		This bit is set when a mismatch of internal transmit data and transmitted data is detected. The SPI samples the signal of the transmit pin (master: SPIx_SIMO, slave: SPIx_SOMI) at the receive point (half clock cycle after transmit point). If the sampled value differs from the transmitted value a bit error is detected and the flag is set. A possible reason for a bit error can be a too high bit rate/capacitive load or another master/slave trying to transmit at the same time. This flag can be cleared by one of the following ways: • Write a 1 to this bit. • Set SPIGCR1.ENABLE bit to 0.
		0	No bit error occurred.
		1	A bit error occurred.
3	DESYNCFLG		Desynchronization of slave device. Desynchronization monitor is active in master mode only. The master monitors the $\overline{\text{SPIx_ENA}}$ signal coming from the slave device and sets the DESYNCFLG bit if the $\overline{\text{SPIx_ENA}}$ signal is not deasserted after the last bit is transmitted plus $t_{T2EDELAY}$. Desynchronization can occur if a slave device misses a clock edge coming from the master. This flag can be cleared by one of the following ways: • Write a 1 to this bit. • Set SPIGCR1.ENABLE bit to 0.
		0	No slave desynchronization detected.
		1	Slave is desynchronized
			Note: Inconsistency of DESYNCFLG in SPI. Due to the nature of this error, under some circumstances it is possible for a desynchronized error detected for the previous buffer to be visible in the current buffer. This is due to the fact that receive completion flag/interrupt will be generated when the buffer transfer is completed. But decence will be detected after the buffer transfer is completed. So, if CPU/DMA reads the received data quickly when an receive interrupt is detected, then the status flag may not reflect the correct decence condition.
2	PARERRFLG		Calculated parity differs from received parity bit. If the parity generator is enabled an even or odd parity bit is added at the end of a data word . During reception of the data word the parity generator calculates the reference parity and compares it to the received parity bit. In the event of a mismatch the PARERRFLG flag is set. This flag can be cleared by one of the following ways: • Write a 1 to this bit. • Set SPIGCR1.ENABLE bit to 0.
		0	No parity error detected.
		1	A parity error occurred.
1	TIMEOUTFLG		Time-out due to non-activation of $\overline{\text{SPIx_ENA}}$ signal. This flag is applicable only for the master mode. The SPI generates a time-out because the slave hasn't responded in time by activating the $\overline{\text{SPIx_ENA}}$ signal after the chip select signal has been activated. If a time-out condition is detected the corresponding chip select is deactivated immediately and the TIMEOUTFLG flag is set. This flag can be cleared by one of the following ways: • Write a 1 to this bit. • Set SPIGCR1.ENABLE bit to 0.
		0	No $\overline{\text{SPIx_ENA}}$ signal time-out occurred.
		1	An $\overline{\text{SPIx_ENA}}$ signal time-out occurred.
0	DLENERRFLG		Data length error flag. This flag can be cleared by one of the following ways: • Write a 1 to this bit. • Set SPIGCR1.ENABLE bit to 0.
		0	No data length error has occurred.
		1	A data length error has occurred.

Copyright © 2013–2016, Texas Instruments Incorporated

20.3.6 SPI Pin Control Register 0 (SPIPC0)

The SPI pin control register 0 (SPIPC0) is shown in Figure 20-23 and described in Table 20-14.

Figure 20-23. SPI Pin Control Register 0 (SPIPC0)

31	16
Reserved	
R-0	

15	12	11	10	9	8
Reserved		SOMIFUN	SIMOFUN	CLKFUN	ENAFUN
R-0		R/W-0	R/W-0	R/W-0	R/W-0

7	1	0
Reserved		SCS0FUN
R-0		R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 20-14. SPI Pin Control Register 0 (SPIPC0) Field Descriptions

Bit	Field	Value	Description
31-12	Reserved	0	Reads return zero and writes have no effect.
11	SOMIFUN		Slave out, master in pin function. This bit determines whether the SPIx_SOMI pin is to be used as a general-purpose I/O pin or as a SPI functional pin.
		0	SPIx_SOMI pin is a GPIO pin.
		1	SPIx_SOMI pin is a SPI functional pin.
10	SIMOFUN		Slave in, master out pin function. This bit determines whether the SPIx_SIMO pin is to be used as a general-purpose I/O pin or as a SPI functional pin.
		0	SPIx_SIMO pin is a GPIO pin.
		1	SPIx_SIMO pin is a SPI functional pin.
9	CLKFUN		SPI clock pin function. This bit determines whether the SPIx_CLK pin is to be used as a general-purpose I/O pin, or as a SPI functional pin.
		0	SPIx_CLK pin is a GPIO pin.
		1	SPIx_CLK pin is a SPI functional pin.
8	ENAFUN		SPI enable pin function. This bit determines whether the $\overline{\text{SPIx_ENA}}$ pin is to be used as a general-purpose I/O pin, or as a SPI functional pin.
		0	$\overline{\text{SPIx_ENA}}$ pin is a GPIO pin.
		1	$\overline{\text{SPIx_ENA}}$ pin is a SPI functional pin.
7-1	Reserved	0	Reserved
0	SCS0FUN		SPI chip select pin *n* function. This bit determines whether the $\overline{\text{SPIx_SCS[0]}}$ pin is to be used as a general-purpose I/O pin, or as a SPI functional pin.
		0	$\overline{\text{SPIx_SCS[0]}}$ pin is a GPIO pin.
		1	$\overline{\text{SPIx_SCS[0]}}$ pin is a SPI functional pin.

Copyright © 2013–2016, Texas Instruments Incorporated

20.3.7 SPI Pin Control Register 1 (SPIPC1)

The SPI pin control register 1 (SPIPC1) is shown in Figure 20-24 and described in Table 20-15.

Figure 20-24. SPI Pin Control Register 1 (SPIPC1)

31	16
Reserved	
R-0	

15	12	11	10	9	8
Reserved		SOMIDIR	SIMODIR	CLKDIR	ENADIR
R-0		R/W-0	R/W-0	R/W-0	R/W-0

7	1	0
Reserved		SCS0DIR
R-0		R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 20-15. SPI Pin Control Register 1 (SPIPC1) Field Descriptions

Bit	Field	Value	Description
31-12	Reserved	0	Reads return zero and writes have no effect.
11	SOMIDIR		SPIx_SOMI pin direction. Controls the direction of the SPIx_SOMI pin when it is used as a general-purpose I/O pin. If the SPIx_SOMI pin is used as a SPI functional pin, the I/O direction is determined by whether the SPI is configured as master or slave.
		0	SPIx_SOMI pin is an input.
		1	SPIx_SOMI pin is an output.
10	SIMODIR		SPIx_SIMO pin direction. Controls the direction of the SPIx_SIMO pin when it is used as a general-purpose I/O pin. If the SPIx_SIMO pin is used as a SPI functional pin, the I/O direction is determined by whether the SPI is configured as master or slave.
		0	SPIx_SIMO pin is an input.
		1	SPIx_SIMO pin is an output.
9	CLKDIR		SPIx_CLK pin direction. Controls the direction of the SPIx_CLK pin when it is used as a general-purpose I/O pin. If the SPIx_CLK pin is used as a SPI functional pin, the I/O direction is determined by whether the SPI is configured as master or slave.
		0	SPIx_CLK pin is an input.
		1	SPIx_CLK pin is an output.
8	ENADIR		$\overline{\text{SPIx_ENA}}$ pin direction. Controls the direction of the $\overline{\text{SPIx_ENA}}$ pin when it is used as a general-purpose I/O pin. If the $\overline{\text{SPIx_ENA}}$ pin is used as a SPI functional pin, then the I/O direction is determined by whether the SPI is configured as master or slave.
		0	$\overline{\text{SPIx_ENA}}$ pin is an input.
		1	$\overline{\text{SPIx_ENA}}$ pin is an output.
7-1	Reserved	0	Reserved
0	SCS0DIR		$\overline{\text{SPIx_SCS[0]}}$ pin direction. Controls the direction of the $\overline{\text{SPIx_SCS[0]}}$ pin when it is used as a general-purpose I/O pin. If the $\overline{\text{SPIx_SCS[0]}}$ pin is used as a SPI functional pin, then the I/O direction is determined by whether the SPI is configured as master or slave.
		0	$\overline{\text{SPIx_SCS[0]}}$ pin is an input.
		1	$\overline{\text{SPIx_SCS[0]}}$ pin is an output.

Copyright © 2013–2016, Texas Instruments Incorporated

20.3.8 SPI Pin Control Register 2 (SPIPC2)

The SPI pin control register 2 (SPIPC2) is shown in Figure 20-25 and described in Table 20-16.

Figure 20-25. SPI Pin Control Register 2 (SPIPC2)

31	16
Reserved	
R-0	

15	12	11	10	9	8
Reserved		SOMIDIN	SIMODIN	CLKDIN	ENADIN
R-0		R-U	R-U	R-U	R-U

7	1	0
Reserved		SCS0DIN
R-0		R-U

LEGEND: R = Read only; U = Undefined; -*n* = value after reset

Table 20-16. SPI Pin Control Register 2 (SPIPC2) Field Descriptions

Bit	Field	Value	Description
31-12	Reserved	0	Reads return zero and writes have no effect.
11	SOMIDIN		SPIx_SOMI data in. This bit reflects the value of the SPIx_SOMI pin.
		0	Current value of SPIx_SOMI pin is logic 0.
		1	Current value of SPIx_SOMI pin is logic 1.
10	SIMODIN		SPIx_SIMO data in. This bit reflects the value of the SPIx_SIMO pin.
		0	Current value of SPIx_SIMO pin is logic 0.
		1	Current value of SPIx_SIMO pin is logic 1.
9	CLKDIN		Clock data in. This bit reflects the value of the SPIx_CLK pin.
		0	Current value of SPIx_CLK pin is logic 0.
		1	Current value of SPIx_CLK pin is logic 1.
8	ENADIN		$\overline{\text{SPIx_ENA}}$ data in. This bit reflects the value of the $\overline{\text{SPIx_ENA}}$ pin.
		0	Current value of $\overline{\text{SPIx_ENA}}$ pin is logic 0.
		1	Current value of $\overline{\text{SPIx_ENA}}$ pin is logic 1.
7-1	Reserved	0	Reserved
0	SCS0DIN		$\overline{\text{SPIx_SCS[0]}}$ data in. This bit reflects the value of the $\overline{\text{SPIx_SCS[0]}}$ pin.
		0	Current value of $\overline{\text{SPIx_SCS[0]}}$ pin is logic 0.
		1	Current value of $\overline{\text{SPIx_SCS[0]}}$ pin is logic 1.

Copyright © 2013–2016, Texas Instruments Incorporated

20.3.9 SPI Pin Control Register 3 (SPIPC3)

The SPI pin control register 3 (SPIPC3) is shown in Figure 20-26 and described in Table 20-17.

Figure 20-26. SPI Pin Control Register 3 (SPIPC3)

31	16
Reserved	
R-0	

15	12	11	10	9	8
Reserved		SOMIDOUT	SIMODOUT	CLKDOUT	ENADOUT
R-0		R/W-0	R/W-0	R/W-0	R/W-0

7	1	0
Reserved		SCS0DOUT
R-0		R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 20-17. SPI Pin Control Register 3 (SPIPC3) Field Descriptions

Bit	Field	Value	Description
31-12	Reserved	0	Reads return zero and writes have no effect.
11	SOMIDOUT		SPIx_SOMI data out write. This bit is only active when the SPIx_SOMI pin is configured as a general-purpose I/O pin and configured as an output pin. The value of this bit indicates the value sent to the pin.
		0	Current value of SPIx_SOMI pin is logic 0.
		1	Current value of SPIx_SOMI pin is logic 1.
10	SIMODOUT		SPIx_SIMO data out write. This bit is only active when the SPIx_SIMO pin is configured as a general-purpose I/O pin and configured as an output pin. The value of this bit indicates the value sent to the pin.
		0	Current value of SPIx_SIMO pin is logic 0.
		1	Current value of SPIx_SIMO pin is logic 1.
9	CLKDOUT		SPIx_CLK data out write. This bit is only active when the SPIx_CLK pin is configured as a general-purpose I/O pin and configured as an output pin. The value of this bit indicates the value sent to the pin.
		0	Current value of SPIx_CLK pin is logic 0.
		1	Current value of SPIx_CLK pin is logic 1.
8	ENADOUT		$\overline{\text{SPIx_ENA}}$ data out write. Only active when the $\overline{\text{SPIx_ENA}}$ pin is configured as a general-purpose I/O pin and configured as an output pin. The value of this bit indicates the value sent to the pin.
		0	Current value of $\overline{\text{SPIx_ENA}}$ pin is logic 0.
		1	Current value of $\overline{\text{SPIx_ENA}}$ pin is logic 1.
7-1	Reserved	0	Reserved
0	SCS0DOUT		$\overline{\text{SPIx_SCS[0]}}$ data out write. Only active when the $\overline{\text{SPIx_SCS[0]}}$ pin is configured as a general-purpose I/O pin and configured as an output pin. The value of this bit indicates the value sent to the pin.
		0	Current value of $\overline{\text{SPIx_SCS[0]}}$ pin is logic 0.
		1	Current value of $\overline{\text{SPIx_SCS[0]}}$ pin is logic 1.

Copyright © 2013–2016, Texas Instruments Incorporated

20.3.10 SPI Pin Control Register 4 (SPIPC4)

The SPI pin control register 4 (SPIPC4) is shown in Figure 20-27 and described in Table 20-18.

Figure 20-27. SPI Pin Control Register 4 (SPIPC4)

31	16
Reserved	
R-0	

15	12	11	10	9	8
Reserved		SOMISET	SIMOSET	CLKSET	ENASET
R-0		R/W-0	R/W-0	R/W-0	R/W-0

7	1	0
Reserved		SCS0SET
R-0		R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 20-18. SPI Pin Control Register 4 (SPIPC4) Field Descriptions

Bit	Field	Value	Description
31-12	Reserved	0	Reads return zero and writes have no effect.
11	SOMISET		SPIx_SOMI data out set. This bit is only active when the SPIx_SOMI pin is configured as a general-purpose output pin. Reads return the value of the SPIx_SOMI pin.
		Write 0	No effect
		Write 1	SPIPC3.SOMIDOUT is set to 1.
10	SIMOSET		SPIx_SIMO data out set. This bit is only active when the SPIx_SIMO pin is configured as a general-purpose output pin. Reads return the value of the SPIx_SIMO pin.
		Write 0	No effect
		Write 1	SPIPC3.SIMODOUT is set to 1.
9	CLKSET		SPIx_CLK data out set. This bit is only active when the SPIx_CLK pin is configured as a general-purpose output pin. Reads return the value of the SPIx_CLK pin.
		Write 0	No effect
		Write 1	SPIPC3.CLKDOUT is set to 1.
8	ENASET		$\overline{\text{SPIx_ENA}}$ data out set. This bit is only active when the $\overline{\text{SPIx_ENA}}$ pin is configured as a general-purpose output pin. Reads return the value of the $\overline{\text{SPIx_ENA}}$ pin.
		Write 0	No effect.
		Write 1	SPIPC3.ENADOUT is set to 1.
7-1	Reserved	0	Reserved
0	SCS0SET		$\overline{\text{SPIx_SCS[0]}}$ data out set. This bit is only active when the $\overline{\text{SPIx_SCS[0]}}$ pin is configured as a general-purpose output pin. Reads return the value of the $\overline{\text{SPIx_SCS[0]}}$ pin.
		Write 0	No effect
		Write 1	SPIPC3.SCS0DOUT is set to 1.

Copyright © 2013–2016, Texas Instruments Incorporated

20.3.11 SPI Pin Control Register 5 (SPIPC5)

The SPI pin control register 5 (SPIPC5) is shown in Figure 20-28 and described in Table 20-19.

Figure 20-28. SPI Pin Control Register 5 (SPIPC5)

31	16
Reserved	
R-0	

15	12	11	10	9	8
Reserved		SOMICLR	SIMOCLR	CLKCLR	ENACLR
R-0		R/W-0	R/W-0	R/W-0	R/W-0

7	1	0
Reserved		SCS0CLR
R-0		R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 20-19. SPI Pin Control Register 5 (SPIPC5) Field Descriptions

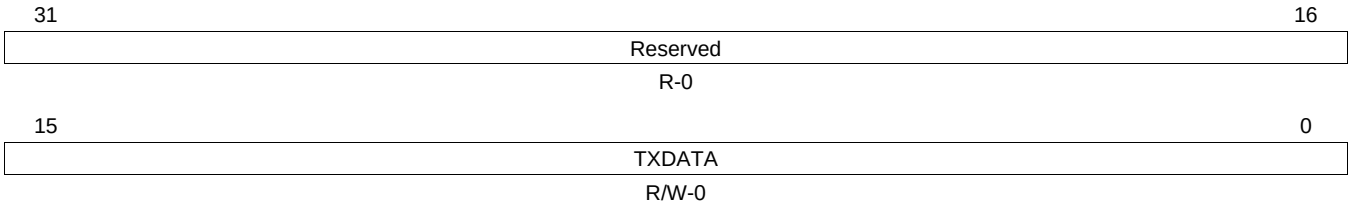
Bit	Field	Value	Description
31-12	Reserved	0	Reads return zero and writes have no effect.
11	SOMICLR		SPIx_SOMI data out clear. This bit is only active when the SPIx_SOMI pin is configured as a general-purpose output pin. Reads return the value of the SPIx_SOMI pin.
		Write 0	No effect.
		Write 1	SPIPC3.SOMIDOUT is cleared to 0.
10	SIMOCLR		SPIx_SIMO data out clear. This bit is only active when the SPIx_SIMO pin is configured as a general-purpose output pin. Reads return the value of the SPIx_SIMO pin.
		Write 0	No effect.
		Write 1	SPIPC3.SIMODOUT is cleared to 0.
9	CLKCLR		SPIx_CLK data out clear. This bit is only active when the SPIx_CLK pin is configured as a general-purpose output pin. Reads return the value of the SPIx_CLK pin.
		Write 0	No effect.
		Write 1	SPIPC3.CLKDOUT is cleared to 0.
8	ENACLR		$\overline{\text{SPIx_ENA}}$ data out clear. This bit is only active when the $\overline{\text{SPIx_ENA}}$ pin is configured as a general-purpose output pin. Reads return the value of the $\overline{\text{SPIx_ENA}}$ pin.
		Write 0	No effect.
		Write 1	SPIPC3.ENADOUT is cleared to 0.
7-1	Reserved	0	Reserved
0	SCS0CLR		$\overline{\text{SPIx_SCS[0]}}$ data out clear. This bit is only active when the $\overline{\text{SPIx_SCS[0]}}$ pin is configured as a general-purpose output pin. Reads return the value of the $\overline{\text{SPIx_SCS[0]}}$ pin.
		Write 0	No effect.
		Write 1	SPIPC3.SCS0DOUT is cleared to 0.

Copyright © 2013–2016, Texas Instruments Incorporated

20.3.12 SPI Transmit Data Register 0 (SPIDAT0)

The SPI transmit data register 0 (SPIDAT0) is shown in Figure 20-29 and described in Table 20-20.

Figure 20-29. SPI Data Register 0 (SPIDAT0)



LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 20-20. SPI Data Register 0 (SPIDAT0) Field Descriptions

Bit	Field	Value	Description
31-16	Reserved	0	Reads return zero and writes have no effect.
15-0	TXDATA	0-FFFFh	SPI transmit data. When written, these bits will be copied to the shift register if it is empty. If the shift register is not empty, the TXBUF will hold the written values. SPIGCR1.ENABLE must be set to 1 before this register can be written to. Writing a 0 to the SPIGCR1.ENABLE forces the TXDATA field to 0. **Note:** Irrespective of the character length, the transmit data should be right-justified before writing to SPIDAT0 register. **Note:** The default data format control register for SPIDAT0 is SPIFMT0. However, it is possible to reprogram the DFSEL field of SPIDAT1 before using SPIDAT0, to select a different SPIFMT*n* register.

Copyright © 2013–2016, Texas Instruments Incorporated

20.3.13 SPI Transmit Data Register 1 (SPIDAT1)

The SPI transmit data register (SPIDAT1) is shown in Figure 20-30 and described in Table 20-21.

Figure 20-30. SPI Data Register 1 (SPIDAT1)

31	29	28	27	26	25	24
Reserved		CSHOLD	Reserved	WDEL	DFSEL	
R-0		R/W-0	R-0	R/W-0	R/W-0	

23	17	16
Reserved		CSNR
R-0		R/W-0

15	0
TXDATA	
R/W-0	

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 20-21. SPI Data Register 1 (SPIDAT1) Field Descriptions

Bit	Field	Value	Description
31-29	Reserved	0	Reads return zero and writes have no effect.
28	CSHOLD		Chip select hold mode. The CSHOLD bit is supported in master mode only. In slave mode, this bit is ignored. CSHOLD defines the behavior of the chip select line at the end of a data transfer.
		0	The chip select signal is deactivated at the end of a transfer after the T2CDELAY time has passed.
		1	The chip select signal is held active at the end of a transfer until a control field with new data and control information is loaded into SPIDAT1. If the new chip select hold information equals the previous one, the active chip select signal is extended until the end of transfer with CSHOLD cleared.
27	Reserved	0	Reads return zero and writes have no effect.
26	WDEL		Enable the delay counter at the end of the current transaction. The WDEL bit is supported in master mode only. In slave mode, this bit is ignored.
		0	No delay will be inserted. However, $\overline{\text{SPIx_SCS[n]}}$ pin will still be deactivated for at least 2 SPI module clock cycles if CSHOLD = 0.
		1	After a transaction, SPIFMT*n*.WDELAY of the selected data format will be loaded into the delay counter. No transaction will be performed until the SPIFMT*n*.WDELAY counter overflows. The $\overline{\text{SPIx_SCS[n]}}$ pin will be deactivated for at least (WDELAY + 2) × SPI module clock period.
25-24	DFSEL	0-3h	Data word format select
		0	Data word format 0 is selected
		1h	Data word format 1 is selected
		2h	Data word format 2 is selected
		3h	Data word format 3 is selected
			Note: Preselecting a Format Register. Writing to just the control field (using byte writes) does not initiate any SPI transfer in master mode. This feature can be used to set up SPIx_CLK phase or polarity before actually starting the transfer by just updating the DFSEL fields in the control field to select the required phase/polarity combination.
23-17	Reserved	0	Reserved
16	CSNR		Chip select number. The CSNR defines the state of the $\overline{\text{SPIx_SCS[0]}}$ pin during a master data transfer. The value of the CSNR bit is driven directly on the $\overline{\text{SPIx_SCS[0]}}$ pin.
			The state of the chip select pin when no transmission is active is specified through the CSDEF bit in the SPI default chip select register (SPIDEF). The chip select pin remains in its active state by setting the CSHOLD bit to 1. When the SPI is configured in slave mode, this bit must be written as 0.
		0	$\overline{\text{SPIx_SCS[0]}}$ pin is driven low.
		1	$\overline{\text{SPIx_SCS[0]}}$ pin is driven high.

Copyright © 2013–2016, Texas Instruments Incorporated

Table 20-21. SPI Data Register 1 (SPIDAT1) Field Descriptions (continued)

Bit	Field	Value	Description
15-0	TXDATA	0-FFFFh	Transfer data. When written, these bits will be copied to the shift register if it is empty. If the shift register is not empty, the TXBUF will hold the written values. SPIGCR1.ENABLE must be set to 1 before this register can be written to. Writing a 0 to the SPIGCR1.ENABLE forces the lower 16 bits of the SPIDAT1 to 0. **Note:** Irrespective of the character length, the transmit data should be right-justified before writing to SPIDAT1.

20.3.14 SPI Receive Buffer Register (SPIBUF)

The SPI receive buffer register (SPIBUF) is shown in Figure 20-31 and described in Table 20-22.

Figure 20-31. SPI Buffer Register (SPIBUF)

31	30	29	28	27	26	25	24
RXEMPTY	RXOVR	TXFULL	BITERR	DESYNC	PARERR	TIMEOUT	DLENERR
RS-1	RC-0	R-0	RC-0	RC-0	RC-0	RC-0	RC-0
23							16
Reserved							
R-0							
15							0
RXDATA							
R-0							

LEGEND: R/W = Read/Write; R = Read only; C = Clear; S = Set; -*n* = value after reset

Table 20-22. SPI Buffer Register (SPIBUF) Field Descriptions

Bit	Field	Value	Description
31	RXEMPTY		Receive data buffer empty. When host reads the RXDATA field or the entire SPIBUF register this automatically sets the RXEMPTY flag. When a data transfer is completed, the received data is copied into SPIBUF, the RXEMPTY flag is cleared. This flag gets set to 1 under following conditions: • Reading the RXDATA field of the SPIBUF register. • Writing 1 to clear the RXINTFLG bit in the SPIFLG register.
		0	New data has been received and copied into the SPIBUF register.
		1	No data received since last reading of the SPIBUF register.
			Write-Clearing the SPIFLG.RXINTFLG bit before reading the SPIBUF register indicates the received data is being ignored. Conversely, SPIFLG.RXINTFLG can be cleared by reading the RXDATA field of the SPIBUF register or the entire SPIBUF register.
30	RXOVR		Receive data buffer overrun. When a data transfer is completed and the received data is copied into the RXBUF while it is already full, RXOVR is set. An overrun always occurs to the RXBUF, and SPIBUF contents never get overwritten until after it is read by the CPU/DMA. Reading SPIBUF register does not clear the RXOVR bit. If an overrun interrupt is detected, then the SPIBUF may need to be read twice to get to the overrun buffer. This is due to the fact that the overrun will always occur to the internal RXBUF. Each read to the SPIBUF will result in RXBUF contents (if it is full) getting copied to SPIBUF. **Note:** A special condition under which RXOVR flag gets set. If both SPIBUF and RXBUF are already full and while another buffer receive is underway, if any errors like TIMEOUT, BITERR and DLENERR occur, then RXOVR will be set to indicate that the status flags are getting overwritten by the new transfer. This overrun should be treated like a normal receiver overrun.
		0	No receive data overrun condition occurred since last time reading the data field.
		1	A receive data overrun condition occurred since last time reading the data field.

Copyright © 2013–2016, Texas Instruments Incorporated

Table 20-22. SPI Buffer Register (SPIBUF) Field Descriptions (continued)

Bit	Field	Value	Description
29	TXFULL		Transmit data buffer full. This flag is a read-only flag. Writing into SPIDAT0 or SPIDAT1 field while the TX shift register is full will automatically set the TXFULL flag. Once the data is copied to the shift register, the TXFULL flag will be cleared. Writing to the SPIDAT0/SPIDAT1 register when both TXBUF and the TX shift register are empty does not set the TXFULL flag.
		0	The transmit buffer is empty; SPIDAT0/SPIDAT1 is ready to accept a new data.
		1	The transmit buffer is full; SPIDAT0/SPIDAT1 is not ready to accept new data.
28	BITERR		Bit error. There was a mismatch of internal transmit data and transmitted data. The SPI samples the signal of the transmit pin (master: SIMO, slave: SOMI) at the receive point (half clock cycle after transmit point). If the sampled value differs from the transmitted value, a bit error is detected and the flag BITERR is set. A possible reason for a bit error can be noise, a too-high bit rate/capacitive load, or another master/slave trying to transmit at the same time.
			Note: This flag is cleared to 0 when RXDATA portion of the SPIBUF register is read.
		0	No bit error occurred.
		1	A bit error occurred.
27	DESYNC		Desynchronization of slave device. This bit is active in master mode only. The master monitors the $\overline{\text{SPIx_ENA}}$ signal coming from the slave device and sets the DESYNC flag if $\overline{\text{SPIx_ENA}}$ is deactivated before the last reception point or after the last bit is transmitted plus $t_{T2EDELAY}$. If DESYNCENA is set, an interrupt is asserted. Desynchronization can occur if a slave device misses a clock edge coming from the master.
			Note: Possible inconsistency of DESYNC flag in SPI. Because of the nature of this error, under some circumstances it is possible for a desync error detected for the previous buffer to be visible in the current buffer. This is because the receive completion flag/interrupt will be generated when the buffer transfer is completed. But desync will be detected after the buffer transfer is completed. So, if CPU/DMA reads the received data quickly when an RXINT is detected, then the status flag may not reflect the correct desync condition.
			Note: This flag is cleared to 0 when the RXDATA portion of the SPIBUF register is read.
		0	No slave de-synchronization detected.
		1	A slave device is desynchronized.
26	PARERR		Parity error. The calculated parity differs from received parity bit. If the parity generator is enabled an even or odd parity bit is added at the end of a data word. During reception of the data word ,the parity generator calculates the reference parity and compares it to the received parity bit. If a mismatch is detected, the PARERR flag is set.
			Note: This flag is cleared to 0 when the RXDATA portion of the SPIBUF register is read.
		0	No parity error detected.
		1	A parity error occurred.
25	TIMEOUT		Time-out because of non-activation of $\overline{\text{SPIx_ENA}}$ pin. This bit is valid in master mode only. The SPI generates a time-out because the slave hasn't responded in time by activating the $\overline{\text{SPIx_ENA}}$ signal after the chip select signal has been activated. If a time-out condition is detected, the corresponding chip select is deactivated immediately and the TIMEOUT flag is set.
			Note: This flag is cleared to 0 when RXDATA portion of the SPIBUF register is read.
		0	No $\overline{\text{SPIx_ENA}}$ pin time-out occurred.
		1	An $\overline{\text{SPIx_ENA}}$ signal time-out occurred.
24	DLENERR		Data length error flag.
			Note: This flag is cleared to 0 when the RXDATA portion of the SPIBUF register is read.
		0	No data length error has occurred.
		1	A data length error has occurred.
23-16	Reserved	0	Reads return zero and writes have no effect.
15-0	RXDATA	0-FFFFh	SPI receive data. This is the received data, transferred from the receive shift-register at the end of a transfer completion. Irrespective of the programmed character length and the direction of shifting, the received data is stored right-justified in the register.

Copyright © 2013–2016, Texas Instruments Incorporated

20.3.15 SPI Emulation Register (SPIEMU)

The SPI emulation register (SPIEMU) is shown in Figure 20-32 and described in Table 20-23.

Figure 20-32. SPI Emulation Register (SPIEMU)

31	16
Reserved	
R-0	

15	0
RXDATA	
R-0	

LEGEND: R = Read only; -*n* = value after reset

Table 20-23. SPI Emulation Register (SPIEMU) Field Descriptions

Bit	Field	Value	Description
31-16	Reserved	0	Reads return zero and writes have no effect.
15-0	RXDATA	0-FFFFh	SPI receive data. SPI emulation is a mirror of the SPIBUF register. The only difference between SPIEMU and SPIBUF is that a read from SPIEMU does not clear any of the status flags.

Copyright © 2013–2016, Texas Instruments Incorporated

20.3.16 SPI Delay Register (SPIDELAY)

The SPI delay register (SPIDELAY) is shown in Figure 20-33 and described in Table 20-24.

Figure 20-33. SPI Delay Register (SPIDELAY)

31	24	23	16
C2TDELAY		T2CDELAY	
R/W-0		R/W-0	

15	8	7	0
T2EDELAY		C2EDELAY	
R/W-0		R/W-0	

LEGEND: R/W = Read/Write; -*n* = value after reset

Table 20-24. SPI Delay Register (SPIDELAY) Field Descriptions

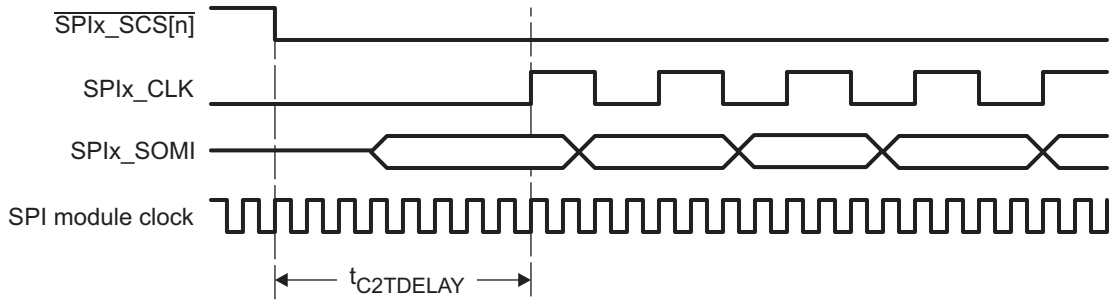
Bit	Field	Value	Description
31-24	C2TDELAY	0-FFh	Chip-select-active-to-transmit-start-delay. C2TDELAY is used in master mode only. It defines a setup time for the slave device that delays the data transmission from the chip select active edge by a multiple of SPI module clock cycles. C2TDELAY can be configured between 3 and 257 SPI module clock cycles. See Figure 20-34. The setup time value is calculated as follows: $t_{C2TDELAY}$ = (C2TDELAY + 2) × SPI module clock period **Note:** If C2TDELAY = 0, then $t_{C2TDELAY}$ = 0. Example: SPI module clock = 25 MHz -> SPI module clock period = 40 ns; C2TDELAY = 06h; > $t_{C2TDELAY}$ = 320 ns; When the chip select signal becomes active, the slave has to prepare for data transfer within 320 ns. **Note:** If phase = 1, the delay between $\overline{\text{SPIx_SCS[n]}}$ falling edge to the first edge of SPIx_CLK will have an additional 0.5 SPIx_CLK period delay. This delay is as per the SPI protocol.
23-16	T2CDELAY	0-FFh	Transmit-end-to-chip-select-inactive-delay. T2CDELAY is used in master mode only. It defines a hold time for the slave device that delays the chip select deactivation by a multiple of SPI module clock cycles after the last bit is transferred. T2CDELAY can be configured between 2 and 256 SPI module clock cycles. See Figure 20-35. The hold time value is calculated as follows: $t_{T2CDELAY}$ = (T2CDELAY + 1) × SPI module clock period **Note:** If T2CDELAY = 0, then $t_{T2CDELAY}$ = 0 Example: VBUSPCLK = 25 MHz -> VBUSPCLK period = 40 ns; T2CDELAY = 03h; > $t_{T2CDELAY}$ = 160 ns; After the last data bit (or parity bit) is being transferred the chip select signal is held active for 160 ns. **Note:** If phase = 0, then between the last edge of SPIx_CLK and rise-edge of $\overline{\text{SPIx_SCS[n]}}$ there will be an additional delay of 0.5 SPIx_CLK period. This is as per the SPI protocol. Both C2TDELAY and T2CDELAY counters will not have any dependancy on the $\overline{\text{SPIx_ENA}}$ pin value. Even if the $\overline{\text{SPIx_ENA}}$ pin is asserted by the slave, the master will continue to delay the start of SPIx_CLK until the C2TDELAY counter overflows. Similarly, even if the $\overline{\text{SPIx_ENA}}$ pin is deasserted by the slave, the master will continue to hold the $\overline{\text{SPIx_SCS[n]}}$ pins active until the T2CDELAY counter overflows. This way, it is assured that the setup/hold times of the $\overline{\text{SPIx_SCS[n]}}$ pins are determined by the delay timers alone. To achieve better throughput, it should be ensured that these two timers are kept at the minimum possible values.

Copyright © 2013–2016, Texas Instruments Incorporated

Table 20-24. SPI Delay Register (SPIDELAY) Field Descriptions (continued)

Bit	Field	Value	Description
15-8	T2EDELAY	0-FFh	Transmit-data-finished-to-$\overline{\text{SPIx_ENA}}$-pin-inactive-time-out. T2EDELAY is used in master mode only. It defines a time-out value as a multiple of SPI clock before the $\overline{\text{SPIx_ENA}}$ signal has to become inactive and after the CS becomes inactive. The SPI clock depends on which data format is selected. If the slave device is missing one or more clock edges, it is becoming desynchronized. Although the master has finished the data transfer the slave is still waiting for the missed clock pulses and the $\overline{\text{SPIx_ENA}}$ signal is not disabled. The T2EDELAY defines a time-out value that triggers the DESYNC flag, if the $\overline{\text{SPIx_ENA}}$ signal is not deactivated in time. The DESYNC flag is set to indicate that the slave device did not deassert its $\overline{\text{SPIx_ENA}}$ pin in time to acknowledge that it has received all the bits of the sent character. The DESYNC flag is also set if the SPI detects a deassertion of the $\overline{\text{SPIx_ENA}}$ pin even before the end of the transmission. See Figure 20-36. The time-out value is calculated as follows: $t_{T2EDELAY}$ = T2EDELAY/SPIclock Example: SPIclock = 8 Mbit/s; T2EDELAY = 10h; > $t_{T2EDELAY}$ = 2 μs; The slave device has to disable the $\overline{\text{SPIx_ENA}}$ signal within 2 μs; otherwise, the DESYNC flag in SPIFLG is set and an interrupt is asserted if enabled.
7-0	C2EDELAY	0-FFh	Chip-select-active-to-$\overline{\text{SPIx_ENA}}$-signal-active-time-out. C2EDELAY is used only in master mode and it applies only if the addressed slave generates an $\overline{\text{SPIx_ENA}}$ signal as a hardware handshake response. C2EDELAY defines the maximum time between the SPI activates the chip select signal and the addressed slave has to respond by activating the $\overline{\text{SPIx_ENA}}$ signal. C2EDELAY defines a time-out value as a multiple of SPI clocks. See Figure 20-37. **Note:** If the slave device is not responding with the $\overline{\text{SPIx_ENA}}$ signal before the time-out value is reached, the TIMEOUT flag in SPIFLG is set and an interrupt is asserted if enabled. The timeout value is calculated as follows: $t_{C2EDELAY}$ = C2EDELAY/SPIclock Example: SPIclock = 8 Mbit/s; C2EDELAY = 30h; > $t_{C2EDELAY}$ = 6 μs; The slave device has to activate the $\overline{\text{SPIx_ENA}}$ signal within 6 μs after the SPI has activated the chip select signal ($\overline{\text{SPIx_SCS[n]}}$); otherwise, the TIMEOUT flag in SPIFLG is set and an interrupt is asserted if enabled.

Figure 20-34. Example: $t_{C2TDELAY}$ = 8 SPI Module Clock Cycles



Copyright © 2013–2016, Texas Instruments Incorporated

Figure 20-35. Example: $t_{T2CDELAY}$ = 4 SPI Module Clock Cycles

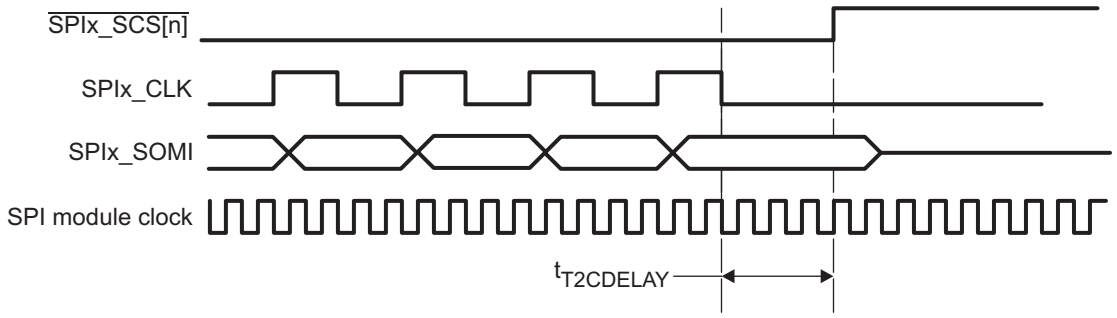


Figure 20-36. Transmit-Data-Finished-to-SPIx_ENA-Inactive-Timeout

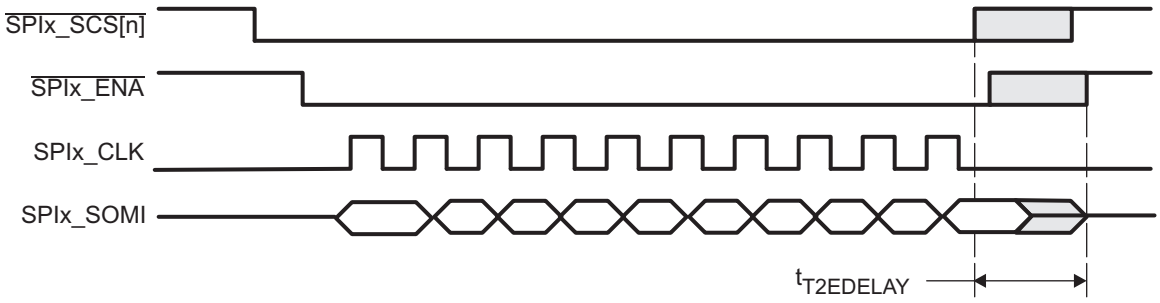
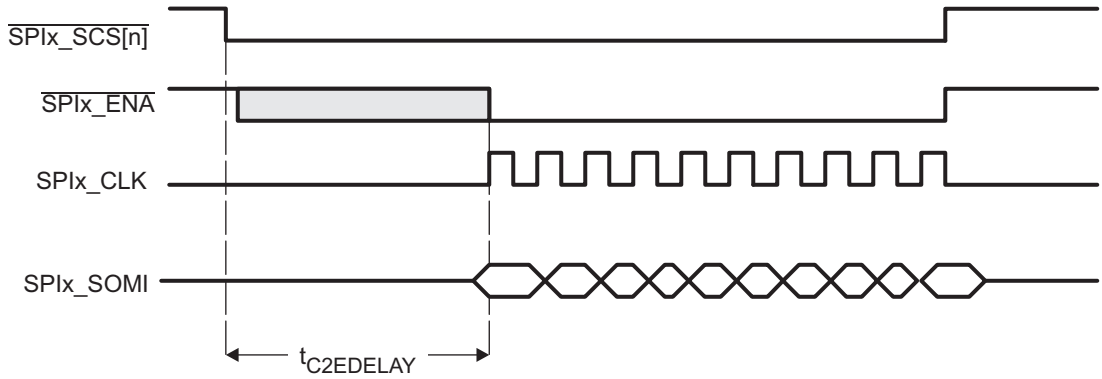


Figure 20-37. Chip-Select-Active-to-SPIx_ENA-Signal-Active-Timeout



Copyright © 2013–2016, Texas Instruments Incorporated

20.3.17 SPI Default Chip Select Register (SPIDEF)

The SPI default chip select register (SPIDEF) is shown in Figure 20-38 and described in Table 20-25.

Figure 20-38. SPI Default Chip Select Register (SPIDEF)

31	16
Reserved	
R-0	

15	1	0
Reserved		CSDEF
R/W-7Fh		R/W-1

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 20-25. SPI Default Chip Select Register (SPIDEF) Field Descriptions

Bit	Field	Value	Description
31-1	Reserved	7Fh	Reserved
0	CSDEF		Chip select default pattern. The CSDEF bit defines the state of the the $\overline{\text{SPIx_SCS[0]}}$ pin when no transmissions are performed. The value of the CSDEF bit is driven directly on the $\overline{\text{SPIx_SCS[0]}}$ pin. The state of the chip select pin during a transmission is specified through the CSNR bit in the SPI transmit data register (SPIDAT1). The chip select pin remains in its active state by setting the CSHOLD bit in SPIDAT1 to 1. In slave mode, the CSDEF bit should be set to 1.
		0	$\overline{\text{SPIx_SCS[0]}}$ pin is driven low.
		1	$\overline{\text{SPIx_SCS[0]}}$ pin is driven high.

Copyright © 2013–2016, Texas Instruments Incorporated

20.3.18 SPI Data Format Registers (SPIFMT*n*)

The SPI data format registers (SPIFMT0, SPIFMT1, SPIFMT2, and SPIFMT3) are shown in Figure 20-39 and described in Table 20-26.

Figure 20-39. SPI Data Format Register (SPIFMT*n*)

31	30	29	24
Reserved		WDELAY	
R-0		R/W-0	

23	22	21	20	19	18	17	16
PARPOL	PARENA	WAITENA	SHIFTDIR	Reserved	DISCSTIMERS	POLARITY	PHASE
R/W-0	R/W-0	R/W-0	R/W-0	R-0	R/W-0	R/W-0	R/W-0

15	8
PRESCALE	
R/W-0	

7	5	4	0
Reserved		CHARLEN	
R-0		R/W-0	

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 20-26. SPI Data Format Register (SPIFMT*n*) Field Descriptions

Bit	Field	Value	Description
31-30	Reserved	0	Reads return zero and writes have no effect.
29-24	WDELAY	0-3Fh	Delay in between transmissions. Idle time that will be applied at the end of the current transmission if the bit WDEL is set in the current buffer. The delay to be applied is equal to: WDELAY × $P_{SPI\ module\ clock}$ + 2 × $P_{SPI\ module\ clock}$ $P_{SPI\ module\ clock}$ -> Period of SPI module clock
23	PARPOL		Parity polarity: even or odd. PARPOL can be modified in privilege mode only.
		0	An even parity flag is added at the end of the transmit data stream.
		1	An odd parity flag is added at the end of the transmit data stream.
22	PARENA		Parity enable.
		0	No parity generation/ verification is performed.
		1	A parity is transmitted at the end of each transmit data stream. At the end of a transfer the parity generator compares the received parity bit with the locally calculated parity flag. If the parity bits do not match the PARERR flag is set in the corresponding control field. The parity type (even or odd) can be selected via the PARPOL bit.
21	WAITENA		The master waits for $\overline{\text{SPIx_ENA}}$ signal from slave. WAITENA is considered in master mode only. In slave mode this bit has no meaning. WAITENA enables a flexible SPI network where slaves with $\overline{\text{SPIx_ENA}}$ signal and slaves without $\overline{\text{SPIx_ENA}}$ signal can be mixed.
		0	The SPI does not wait for the $\overline{\text{SPIx_ENA}}$ signal from the slave and directly starts the transfer.
		1	Before the SPI starts the data transfer it waits for the $\overline{\text{SPIx_ENA}}$ signal to become low. If the $\overline{\text{SPIx_ENA}}$ signal is not pulled down by the addressed slave before the internal time-out counter (C2EDELAY) overflows, then the master aborts the transfer and sets the TIMEOUT error flag.
20	SHIFTDIR		Shift direction.
		0	Most significant bit is shifted out first.
		1	Least significant bit is shifted out first.
19	Reserved	0	Reads return zero and writes have no effect.
18	DISCSTIMERS		Disable chip select timers for this format register. The C2TDELAY and T2CDELAY timers are by default enabled for all the data format registers. Using this bit, these timers can be disabled for a particular data format if not required. When a master is handling multiple slaves, with varied set-up hold requirement, the application can selectively choose to include or not include the chip select delay timers for any slaves.
		0	Both C2TDELAY and T2CDELAY counts are inserted for the chip selects.
		1	No C2TDELAY or T2CDELAY is inserted in the chip select timings.

Copyright © 2013–2016, Texas Instruments Incorporated

Table 20-26. SPI Data Format Register (SPIFMT*n*) Field Descriptions (continued)

Bit	Field	Value	Description
17	POLARITY		SPI clock polarity.
		0	SPI clock signal is low-inactive (before and after data transfer the clock signal is low).
		1	SPI clock signal is high-inactive (before and after data transfer the clock signal is high).
16	PHASE		SPI clock delay.
		0	SPI clock signal is not delayed versus the transmit/receive data stream. The first data bit is transmitted with the first clock edge and the first bit is received with the second (inverse) clock edge.
		1	SPI clock signal is delayed by a half SPI clock cycle versus the transmit/receive data stream. The first transmit bit has to output prior to the first clock edge. The master and slave receive the first bit with the first edge.
15-8	PRESCALE	2h-FFh	SPI prescaler. It determines the bit transfer rate if the SPI is the network master and is directly derived from the SPI module clock. If the SPI is configured as slave, PRESCALE needs to be configured to a valid value, but PRESCALE is ignored. The clock rate can be calculated as: SPI clock frequency = SPI module clock/(PRESCALE + 1) **Note:** PRESCALE values less than 2h are not supported.
7-5	Reserved	0	Reads return zero and writes have no effect.
4-0	CHARLEN	0-1Fh	SPI data word length. Legal values are 2h (data word length = 2 bit) to 10h (data word length = 16). Illegal values, such as 0 or 1Fh are not detected and their effect is indeterminate.

Copyright © 2013–2016, Texas Instruments Incorporated

20.3.19 SPI Interrupt Vector Register 1 (INTVEC1)

The SPI interrupt vector register 1 (INTVEC1) is shown in Figure 20-40 and described in Table 20-27.

Figure 20-40. SPI Interrupt Vector Register 1 (INTVEC1)

31	16
Reserved	
R-0	

15	6	5	1	0
Reserved		INTVECT1		Rsvd
R-0		R-0		R-0

LEGEND: R = Read only; -*n* = value after reset

Table 20-27. SPI Interrupt Vector Register 1 (INTVEC1) Field Descriptions

Bit	Field	Value	Description
31-6	Reserved	0	Reads return zero and writes have no effect.
5-1	INTVECT1	0-1Fh	Interrupt vector for interrupt line INT1. INTVECT1 returns the vector of the pending interrupt at interrupt line INT1. If more than one interrupt is pending, INTVECT1 always references the highest priority interrupt source first. The interrupts available for SPI in the descending order of their priorities are as given below. • Transmission error Interrupt • Receive buffer overrun interrupt • Receive buffer full interrupt • Transmit buffer empty interrupt The INTVECT1 field just reflects the status of SPIFLG in a vectorized format. So, any updates to SPIFLG will automatically reflect in the vector value in this register. Vectors for each of these interrupts will be reflected on the INTVECT1 bits, when they occur. Reading the vectors for the receive buffer overrun and receive buffer full interrupts will automatically clear the respective flags in the SPIFLG. Reading the vector register when transmitter empty is indicated does not clear the TXINTFLG in SPIFLG. Writing a new data to SPIDAT0/SPIDAT1 clears the transmitter empty interrupt. On reading the INTVECT1 bits, the vector of the next highest priority interrupt (if any) will be then reflected on the INTVECT1 bits. If two or more interrupts occur simultaneously, the vector for the highest priority interrupt will be reflected on the INTVECT1 bits. The following are the SPI interrupt vectors for line INT1:
		0	No interrupt pending
		1h-10h	Reserved
		11h	Error interrupt pending. Refer to lower halfword of SPIINT0 to determine more details about the type of error.
		12h	The pending interrupt is receive buffer full interrupt.
		13h	The pending interrupt is receive buffer overrun interrupt.
		14h	The pending interrupt is transmit buffer empty interrupt.
		15h-1Fh	Reserved
0	Reserved	0	Reads return zero and writes have no effect.

Copyright © 2013–2016, Texas Instruments Incorporated

64-Bit Timer Plus

This chapter describes the operation of the software-programmable 64-bit Timer Plus. See your device-specific data manual to determine how many Timer modules are available on your device.

Topic **Page**

21.1 Introduction **753**
21.2 Registers **771**

Copyright © 2013–2016, Texas Instruments Incorporated

21.1 Introduction

The 64-bit Timer Plus can be programmed in 64-bit mode, dual 32-bit unchained mode, or dual 32-bit chained mode. Some Timer Plus implementations have signal connections to internal device reset that can be used in watchdog timer mode. New features over previous timers include: external clock/event input, period reload, external event capture, and timer counter register read reset.

21.1.1 Purpose of the Peripheral

The timer can support four basic modes of operation: a 64-bit general-purpose (GP) timer, dual unchained 32-bit GP timers, dual chained 32-bit timers, or a watchdog timer. The GP timer modes can be used to generate periodic interrupts and DMA synchronization events. The watchdog timer mode is used to provide a recovery mechanism for the device in the event of a fault condition (such as a non-exiting code loop).

21.1.2 Features

The 64-bit timer consists of the following features -- some features may not be supported on all timer instantiations (see your device-specific data manual for supported features):

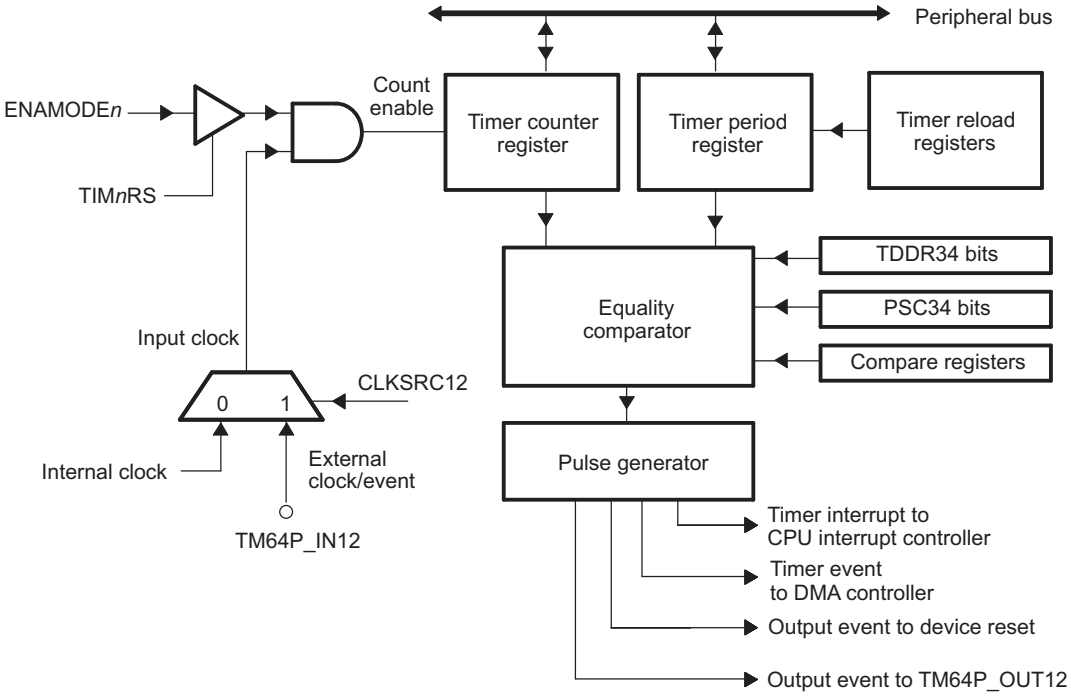
- 64-bit count-up counter
- Timer modes:
  - 64-bit general-purpose timer mode
  - Dual 32-bit unchained general-purpose timer mode
  - Dual 32-bit chained timer mode
  - Watchdog timer mode
- 2 possible clock sources:
  - Internal clock
  - External clock/event input via timer input pins
- 3 possible operation modes:
  - One-time operation (timer runs for one period then stops)
  - Continuous operation (timer automatically resets to zero after each period and continues to operate)
  - Continuous operation with period reload (timer automatically assumes the value of the reload registers after each period and continues to operate)
- Generates interrupts to CPU
- Generates sync events to DMA
- Generates output event to device reset (watchdog only)
- Generates output event to timer output pins (if pins are available)
- External event capture via timer input pins (if pins are available)

Copyright © 2013–2016, Texas Instruments Incorporated

21.1.3 Block Diagram

A block diagram of the timer is shown in Figure 21-1. Detailed information about the architecture and operation of the timers is in Section 21.1.5 and Section 21.1.6.

Figure 21-1. Timer Block Diagram



21.1.4 Industry Standard Compatibility Statement

This peripheral is not intended to conform to any specific industry standard.

Architecture

21.1.5 Architecture – General-Purpose Timer Mode

This section describes the timer in the general-purpose (GP) timer mode.

21.1.5.1 Backward Compatible Mode

The Timer Plus supports the following additional features over the other timers:

- External clock/event input
- Period reload
- External event capture mode
- Timer counter register read reset mode
- Timer counter capture registers
- Register for interrupt/DMA generation control and status

By default, period reload, external event capture mode, timer counter register read reset mode, timer counter capture registers, and interrupt/DMA/TM64P_OUT generation control and status are not available. To enable these features, you must set the PLUSEN bit in the timer global control register (TGCR). These features are described throughout the following sections. External clock/event input is always available, regardless of the state of the backward compatible bit.

Copyright © 2013–2016, Texas Instruments Incorporated

21.1.5.2 Clock Control

The timer can use an internal or external clock source for the counter period. The following sections explain how to select the clock source.

As shown in Table 21-1 and Figure 21-2, the timer clock source is selected using the clock source (CLKSRC12) bit in the timer control register (TCR). Two clock sources are available to drive the timer clock:

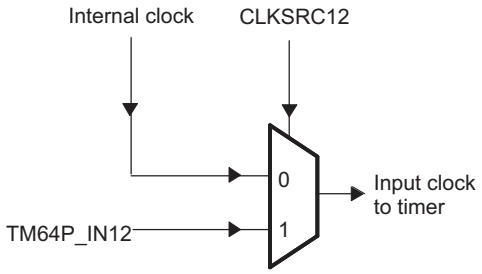
- internal clock by setting CLKSRC12 = 0.
- external clock on input pin TM64P_IN12 by setting CLKSRC12 = 1.

At reset, the clock source is the internal clock. Details on each of the clock source configuration options are included in the following sections.

Table 21-1. Timer Clock Source Selection

CLKSRC12	Input Clock
0	Internal clock (default)
1	External clock on timer input

Figure 21-2. Timer Clock Source Block Diagram



21.1.5.2.1 Using the Internal Clock Source to the Timer

The internal clock source to the timer is generated by the PLL controller. This clock source determines the speed of the timer since the timer counts up in units of source clock cycles. When determining the period and prescaler settings for the timer, choose the desired period in units of source clock cycles. For details on the generation of the on-chip clocks, see the *Phase-Locked Loop Controller (PLLC)* chapter.

The CLKSRC12 parameter in the timer control register (TCR) determines whether an internal or external clock is used as the clock source for the timer. If the timer is configured in 64-bit mode or 32-bit chained mode, CLKSRC12 controls the clock source for the entire timer. If the timer is configured in dual 32-bit unchained mode (TIMMODE = 01 in TGCR), CLKSRC12 controls the timer 1:2 side of the timer only.

To select the internal clock as the clock source for the timer, CLKSRC12 in TCR must be cleared to 0.

21.1.5.2.2 Using the External Clock Source to the Timer

An external clock source can be provided to clock the timer through the timer input pin TM64P_IN12. The CLKSRC12 parameter in the timer control register (TCR) determines whether an internal or external clock is used as the clock source for the timer. If the timer is configured in 64-bit mode or 32-bit chained mode, CLKSRC12 controls the clock source for the entire timer. If the timer is configured in dual 32-bit unchained mode (TIMMODE = 01 in TGCR), CLKSRC12 controls the timer 1:2 side of the timer only.

At reset, the clock source defaults to the internal clock. Details on each of the clock source configuration options are included in the following sections. To select the external clock as the clock source for the timer, CLKSRC12 in TCR must be set to 1. The external clock source frequency must be no greater than the timer peripheral reference clock (see your device-specific data manual).

Copyright © 2013–2016, Texas Instruments Incorporated

21.1.5.3 Signal Descriptions

As shown in Figure 21-2, the TM64P_IN12 pin may be used as input to the timer. This signal can be used to drive the clock/event count or be used as an external event input for event capture mode. Pin TM64P_OUT12 may be used as an output from the timer to generate a clock or pulse signal.

21.1.5.4 Timer Modes

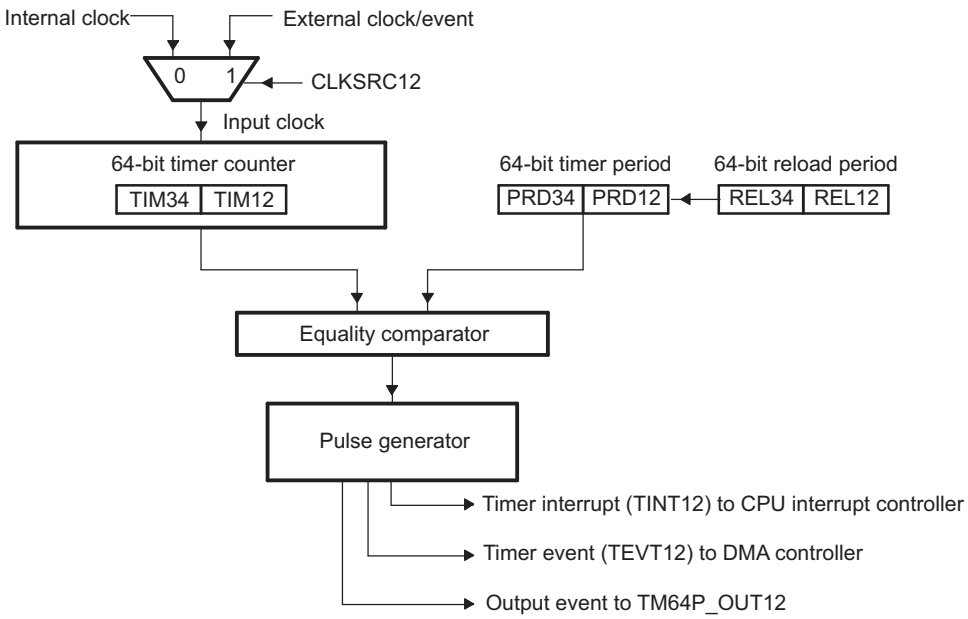
The following section describes the general-purpose (GP) timer modes.

21.1.5.4.1 64-Bit Timer Mode

The timer can be configured as a 64-bit timer by clearing the TIMMODE bit in the timer global control register (TGCR) to 0. At reset, 0 is the default setting for the TIMMODE bit.

In this mode, the timer operates as a single 64-bit up-counter (Figure 21-3). The counter registers (TIM12 and TIM34) form a 64-bit timer counter register and the period registers (PRD12 and PRD34) form a 64-bit timer period register. When the timer is enabled, the timer counter starts incrementing by 1 at every timer input clock cycle. When the timer counter matches the timer period, a maskable timer interrupt (TINT12) and a timer EDMA (TEVT12) are generated. When the timer is configured in continuous mode, the timer counter is reset to 0 on the cycle after the timer counter reaches the timer period. The timer can be stopped, restarted, reset, or disabled using control bits in TGCR.

Figure 21-3. 64-Bit Timer Mode Block Diagram



Copyright © 2013–2016, Texas Instruments Incorporated

21.1.5.4.1.1 Enabling the 64-Bit Timer

The TIM12RS and TIM34RS bits in TGCR control whether the timer is in reset or capable of operating. For the timer to operate in 64-bit timer mode, the TIM12RS and TIM34RS bits must be set to 1.

The ENAMODE12 bit in the timer control register (TCR) controls whether the timer is disabled, enabled to run once, enabled to run continuously, or enabled to run continuously with period reload; the ENAMODE34 bit has no effect in 64-bit timer mode. When the timer is disabled (ENAMODE12 = 0), the timer does not run and maintains its current count value. When the timer is enabled for one time operation (ENAMODE12 = 1), it counts up until the counter value equals the period value and then stops. When the timer is enabled for continuous operation (ENAMODE12 = 2h), the counter counts up until it reaches the period value, then resets itself to zero and begins counting again. When the timer is enabled for continuous operation with period reload (ENAMODE12 = 3h), the counter counts up until it reaches the period value, then resets itself to zero, reloads the period registers (PRD12 and PRD34) with the value in the period reload registers (REL12 and REL34), and begins counting again.

Table 21-2 shows the bit values in TGCR to configure the 64-bit timer.

Table 21-2. 64-Bit Timer Configurations

64-Bit Timer Configuration	TGCR Bit		TCR Bit
	TIM12RS	TIM34RS	ENAMODE12
To place the 64-bit timer in reset	0	0	0
To disable the 64-bit timer (out of reset)	1h	1h	0
To enable the 64-bit timer for one-time operation	1h	1h	1h
To enable the 64-bit timer for continuous operation	1h	1h	2h
To enable the 64-bit timer for continuous operation with period reload	1h	1h	3h

Once the timer stops, if an external clock is used as the timer clock, the timer must remain disabled for at least one external clock period or the timer will not start counting again. When using the external clock, the count value is synchronized to the internal clock.

Note that when both the timer counter and timer period are cleared to 0, the timer can be enabled but the timer counter does not increment because the timer period is 0.

21.1.5.4.1.2 Reading the Counter Registers

When reading the timer count in 64-bit timer mode, the CPU must first read TIM12 followed by TIM34. When TIM12 is read, the timer copies TIM34 into a shadow register. When reading TIM34, the hardware logic returns the shadow register value. This ensures that the values read from the registers are not affected by the fact that the timer may continue to run as the registers are read. When reading the timers in 32-bit mode, TIM12 and TIM34 may be read in any order.

21.1.5.4.1.3 64-Bit Timer Configuration Procedure

To configure the GP timer to operate as a 64-bit timer, follow the steps below:

1. Select 64-bit mode (TIMMODE in TGCR).
2. Remove the timer from reset (TIM12RS and TIM34RS in TGCR).
3. Select the desired timer period (PRD12 and PRD34). Program with the desired timer period value - 1.
4. Enable the timer (ENAMODE12 in TCR).
5. If ENAMODE12 = 3h, write the desired timer period for the next timer cycle in the period reload registers (REL12 and REL34). Program with the desired timer period value - 1. This step can be done at any time before the current timer cycle ends.

Copyright © 2013–2016, Texas Instruments Incorporated

21.1.5.4.2 Dual 32-Bit Timer Modes

Each of the general-purpose timers can be configured as dual 32-bit timers by configuring the TIMMODE bit in the timer global control register (TGCR). In dual 32-bit timer mode, the two 32-bit timers can be operated independently (unchained mode) or in conjunction with each other (chained mode).

21.1.5.4.2.1 Chained Mode

The general-purpose timers can each be configured as a dual 32-bit chained timer by setting the TIMMODE bit to 3h in TGCR.

In the chained mode (Figure 21-4), one 32-bit timer (timer 3:4) is used as a 32-bit prescaler and the other 32-bit timer (timer 1:2) is used as a 32-bit timer. The 32-bit prescaler is used to clock the 32-bit timer. The 32-bit prescaler uses one counter register (TIM34) to form a 32-bit prescale counter register and one period register (PRD34) to form a 32-bit prescale period register.

When the timer is enabled, the prescale counter starts incrementing by 1 at every timer input clock cycle. One cycle after the prescale counter matches the prescale period, a clock signal is generated and the prescale counter register is reset to 0 (see the example in Figure 21-5).

The other 32-bit timer (timer 1:2) uses one counter register (TIM12) to form a 32-bit timer counter register and one period register (PRD12) to form a 32-bit timer period register. This timer is clocked by the output clock from the prescaler. The timer counter increments by 1 at every prescaler output clock cycle. When the timer counter matches the timer period, a maskable timer interrupt (TINT12) and a timer EDMA event (TEVT12) are generated. When the timer is configured in continuous mode, the timer counter is reset to 0 on the cycle after the timer counter reaches the timer period. The timer can be stopped, restarted, reset, or disabled using the TIM12RS and TIM34RS bits in TGCR. In the chained mode, the upper 16-bits of the timer control register (TCR) are not used.

Copyright © 2013–2016, Texas Instruments Incorporated

Figure 21-4. Dual 32-Bit Timers Chained Mode Block Diagram

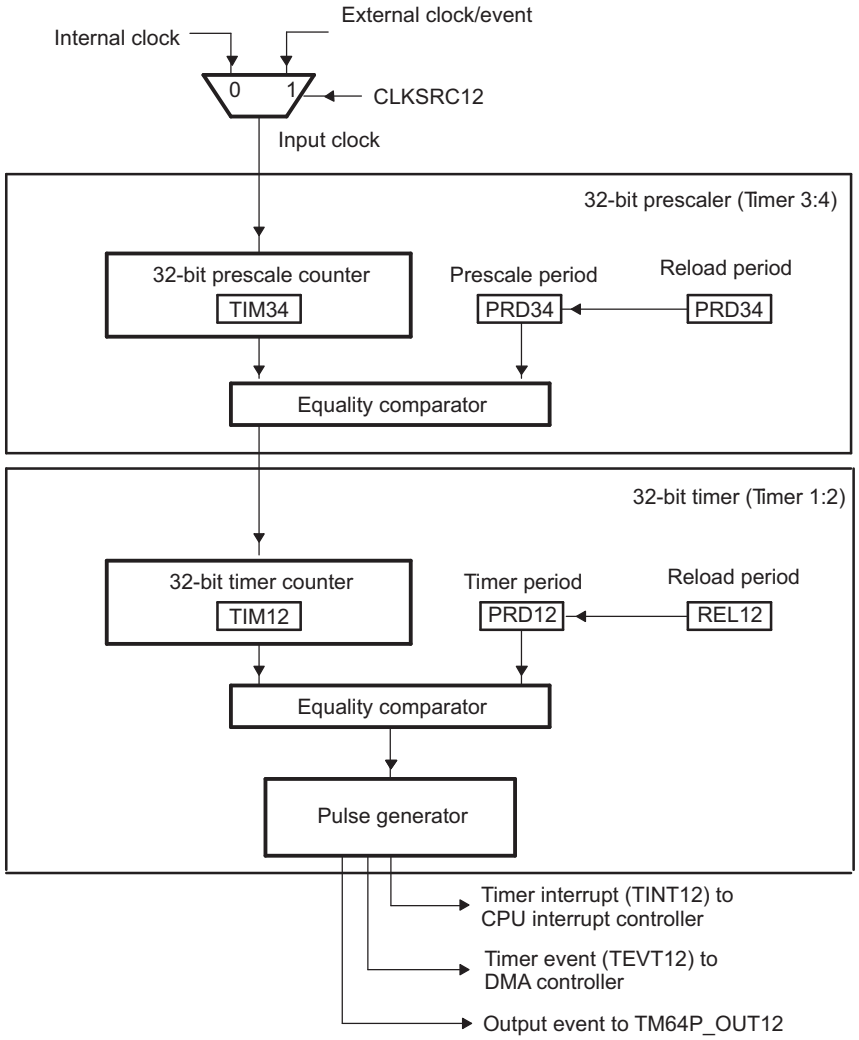
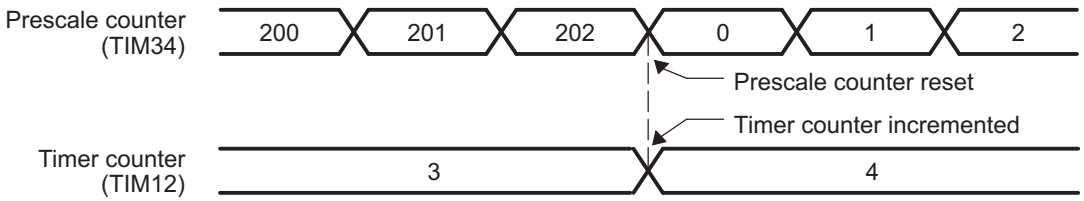


Figure 21-5. Dual 32-Bit Timers Chained Mode Example

32-bit prescaler settings: count = TIM34 = 200; period = PRD34 = 202
32-bit timer settings: count = TIM12 = 3; period = PRD12= 4



Copyright © 2013–2016, Texas Instruments Incorporated

21.1.5.4.2.1.1 Enabling the 32-Bit Timer Chained Mode

The TIM12RS and TIM34RS bits in TGCR determine whether the timer is in reset, or if it is capable of operating. The TIM12RS bit controls the reset of the timer 1:2 side of the timer and the TIM34RS bits control the reset of the timer 3:4 side of the timer. For the timer to operate, the TIM12RS and TIM34RS bits must be set to 1.

The ENAMODE12 bit in the timer control register (TCR) controls whether the timer is disabled, enabled to run once, enabled to run continuously, or enabled to run continuously with period reload; the ENAMODE34 bit has no effect in 32-bit timer chained mode. When the timer is disabled (ENAMODE12 = 0), the timer does not run and maintains its current count value. When the timer is enabled for one time operation (ENAMODE12 = 1), it counts up until the counter value equals the period value and then stops. When the timer is enabled for continuous operation (ENAMODE12 = 2h), the counter counts up until it reaches the period value, then resets itself to zero and begins counting again. When the timer is enabled for continuous operation with period reload (ENAMODE12 = 3h), the counter counts up until it reaches the period value, then resets itself to zero, reloads the period registers (PRD12 and PRD34) with the value in the period reload registers (REL12 and REL34), and begins counting again.

Table 21-3 shows the bit values in TGCR to configure the 32-bit timer in chained mode.

Table 21-3. 32-Bit Timer Chained Mode Configurations

	TGCR Bit		TCR Bit
32-Bit Timer Configuration	**TIM12RS**	**TIM34RS**	**ENAMODE12**
To place the 32-bit timer chained mode in reset	0	0	0
To disable the 32-bit timer chained mode (out of reset)	1h	1h	0
To enable the 32-bit timer chained mode for one-time operation	1h	1h	1h
To enable the 32-bit timer chained mode for continuous operation	1h	1h	2h
To enable the 32-bit timer chained mode for continuous operation with period reload (Timer 3 only)	1h	1h	3h

Once the timer stops, if an external clock is used as the timer clock, the timer must remain disabled for at least one external clock period or the timer will not start counting again. When using the external clock, the count value is synchronized to the internal clock.

Note that when both the timer counter and timer period are cleared to 0, the timer can be enabled but the timer counter does not increment because the timer period is 0.

21.1.5.4.2.1.2 32-Bit Timer Chained Mode Configuration Procedure

To configure the GP timer to operate as a dual 32-bit chained mode timer, follow the steps below:

1. Select 32-bit chained mode (TIMMODE in TGCR).
2. Remove the timer from reset (TIM12RS and TIM34RS in TGCR).
3. Select the desired timer period (PRD12). Program with the desired timer period value - 1.
4. Select the desired timer prescaler value (PRD34).
5. Enable the timer (ENAMODE12 in TCR).
6. If ENAMODE12 = 3h, write the desired timer period for the next timer cycle in the period reload registers (REL12 and REL34). Program with the desired timer period value - 1. This step can be done at any time before the current timer cycle ends.

Copyright © 2013–2016, Texas Instruments Incorporated

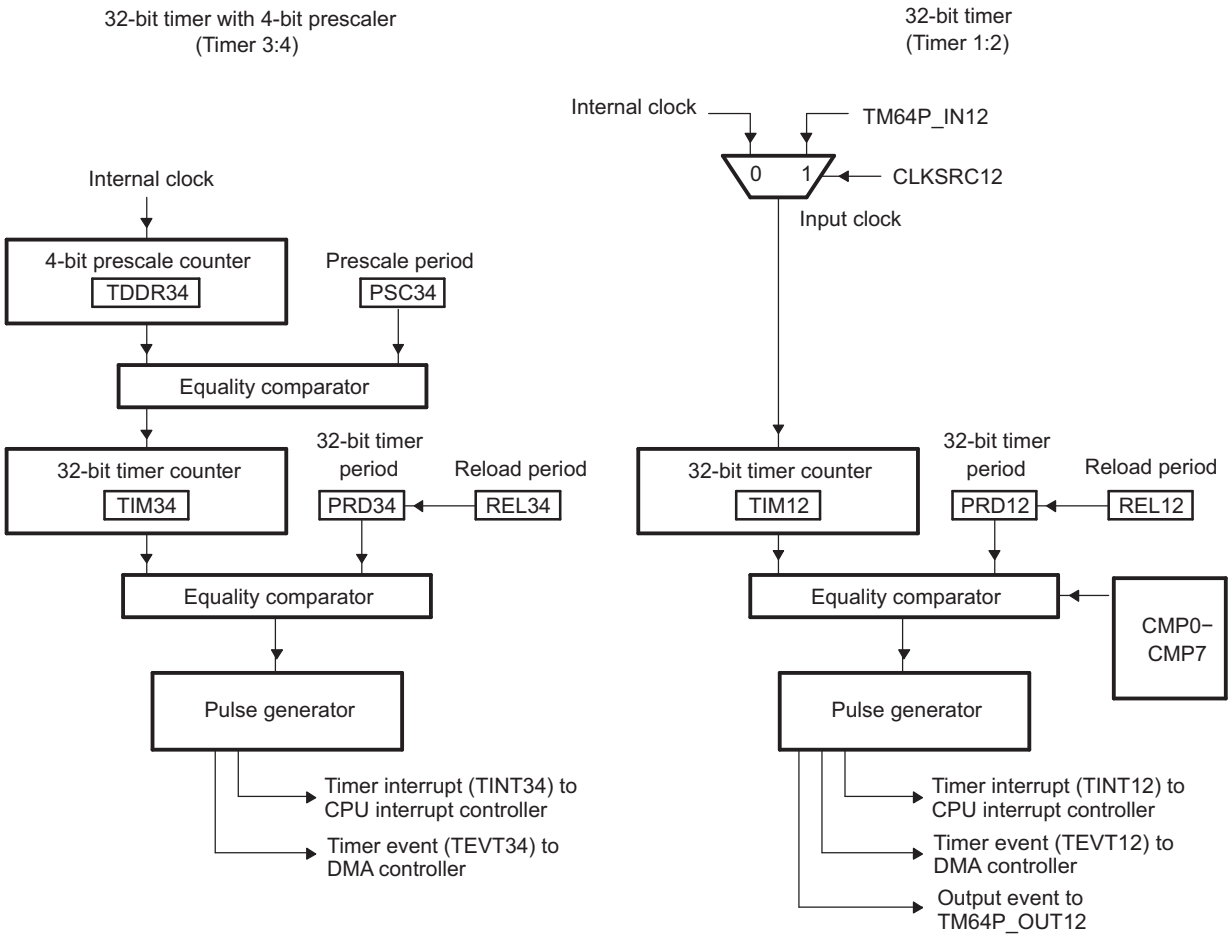
21.1.5.4.2.2 Unchained Mode

The general-purpose timers can be configured as a dual 32-bit unchained timers by setting the TIMMODE bit to 1 in TGCR.

In the unchained mode (Figure 21-6), the timer operates as two independent 32-bit timers. One 32-bit timer (timer 3:4) operates as a 32-bit timer being clocked by a 4-bit prescaler. The other 32-bit timer (timer 1:2) operates as a 32-bit timer with no prescaler.

Independent of the normal timer behavior, eight compare registers (CMP*n*) are compared against the value of the TIM12 register when the PLUSEN bit in TGCR is set. Upon a successful non-zero match, an interrupt and a DMA event are generated without affecting the TIM12 value, behavior, or associated counter registers. Note that some timer instantiations may not map the CMP interrupt and DMA events to the CPU and DMA engines (see your device-specific data manual for information).

Figure 21-6. Dual 32-Bit Timers Unchained Mode Block Diagram



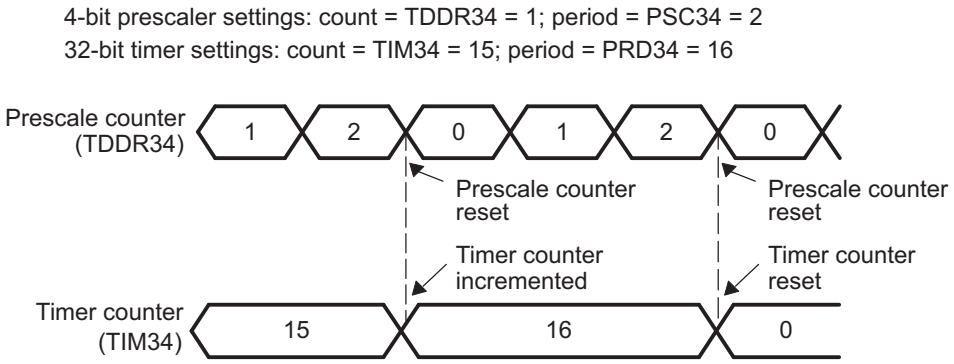
Copyright © 2013–2016, Texas Instruments Incorporated

21.1.5.4.2.2.1 32-Bit Timer With a 4-Bit Prescaler

In the unchained mode, the 4-bit prescale is clocked by the internal clock. The 4-bit prescaler uses the timer divide-down ratio (TDDR34) bit in TGCR to form a 4-bit prescale counter register and the prescale counter bits (PSC34) to form a 4-bit prescale period register (see Figure 21-6). When the timer is enabled, the prescale counter starts incrementing by 1 at every timer input clock cycle. One cycle after the prescale counter matches the prescale period, a clock signal is generated for the 32-bit timer.

The 32-bit timer uses TIM34 as a 32-bit timer counter register and PRD34 as a 32-bit timer period register. The 32-bit timer is clocked by the output clock from the 4-bit prescaler (see the example in Figure 21-7). The timer counter increments by 1 at every prescaler output clock cycle. When the timer counter matches the period, a maskable timer interrupt (TINT34) and a timer DMA event (TEVT34) are generated. When the timer is configured in continuous mode, the timer counter is reset to 0 on the cycle after the timer counter reaches the timer period. The timer can be stopped, restarted, reset, or disabled using the TIM34RS bit in TGCR. For timer 3:4, the lower 16 bits of the timer control register (TCR) have no control.

Figure 21-7. Dual 32-Bit Timers Unchained Mode Example



21.1.5.4.2.2.2 32-Bit Timer with No Prescaler

The other 32-bit timer (timer 1:2) uses TIM12 as the 32-bit counter register and PRD12 as a 32-bit timer period register (see Figure 21-6). When the timer is enabled, the timer counter increments by 1 at every timer input clock cycle. When the timer counter matches the timer period, a maskable timer interrupt (TINT12), a timer DMA event (TEVT12), and a timer output event on TM64P_OUT12 are generated. When the timer is configured in continuous mode, the timer counter is reset to 0 on the cycle after the timer counter reaches the timer period. The timer can be stopped, restarted, reset, or disabled using the TIM12RS bit in TGCR. For timer 1:2, the upper 16 bit of the timer control register (TCR) have no control.

Copyright © 2013–2016, Texas Instruments Incorporated

21.1.5.4.2.2.3 Enabling the 32-Bit Unchained Mode Timer

The TIM12RS and TIM34RS bits in TGCR determine whether the timer is in reset, or if it is capable of operating. The TIM12RS bit controls the reset of the timer 1:2 side of the timer and the TIM34RS bit controls the reset of the timer 3:4 side of the timer. For the timer to operate, the TIM12RS and/or TIM34RS bits must be set to 1.

The ENAMODE*n* bit in the timer control register (TCR) controls whether the timer is disabled, enabled to run once, or enabled to run continuously.

- When the timer is disabled (ENAMODE*n* = 0), the timer does not run and maintains its current count value.
- When the timer is enabled for one time operation (ENAMODE*n* = 1), it counts up until the counter value equals the period value and then stops.
- When the timer is enabled for continuous operation (ENAMODE*n* = 2h), the counter counts up until it reaches the period value, then resets itself to zero and begins counting again.
- When the timer is enabled for continuous operation with period reload (ENAMODE*n* = 3h), the counter counts up until it reaches the period value, then resets itself to zero, reloads the period registers (PRD12 and/or PRD34) with the value in the period reload registers (REL12 and/or REL34), and begins counting again.

Table 21-4 shows the bit values in TGCR to configure the 32-bit timer in unchained mode.

Once the timer stops, if an external clock is used as the timer clock, the timer must remain disabled for at least one external clock period or the timer will not start counting again. When using the external clock, the count value is synchronized to the internal clock.

Note that when both the timer counter and timer period are cleared to 0, the timer can be enabled but the timer counter does not increment because the timer period is 0.

Table 21-4. 32-Bit Timer Unchained Mode Configurations

	TGCR Bit		TCR Bit	
32-Bit Timer Configuration	**TIM12RS**	**TIM34RS**	**ENAMODE12**	**ENAMODE34**
To place the 32-bit timer unchained mode with 4-bit prescaler in reset	x	0	x	0
To disable the 32-bit timer unchained mode with 4-bit prescaler (out of reset)	x	1h	x	0
To enable the 32-bit timer unchained mode with 4-bit prescaler for one-time operation	x	1h	x	1h
To enable the 32-bit timer unchained mode with 4-bit prescaler for continuous operation	x	1h	x	2h
To enable the 32-bit timer unchained mode with 4-bit prescaler for continuous operation with period reload	x	1h	x	3h
To place the 32-bit timer unchained mode with no prescaler in reset	0	x	0	x
To disable the 32-bit timer unchained mode with no prescaler (out of reset)	1h	x	0	x
To enable the 32-bit timer unchained mode with no prescaler for one-time operation	1h	x	1h	x
To enable the 32-bit timer unchained mode with no prescaler for continuous operation	1h	x	2h	x
To enable the 32-bit timer unchained mode with no prescaler for continuous operation with period reload	1h	x	3h	x

Copyright © 2013–2016, Texas Instruments Incorporated

21.1.5.4.2.2.4 32-Bit Timer Unchained Mode Configuration Procedure

To configure timer 1:2, follow the steps below:

1. Select 32-bit unchained mode (TIMMODE in TGCR).
2. Remove the timer 1:2 from reset (TIM12RS in TGCR).
3. Select the desired timer period for timer 1:2 (PRD12). Program with the desired timer period value - 1.
4. Select the desired clock source for timer 1:2 (CLKSRC12 in TCR).
5. Enable timer 1:2 (ENAMODE12 in TCR).
6. If ENAMODE12 = 3h, write the desired timer period for the next timer cycle in the period reload register (REL12). Program with the desired timer period value - 1. This step can be done at any time before the current timer cycle ends.

To configure timer 3:4, follow the steps below:

1. Select 32-bit unchained mode (TIMMODE in TGCR).
2. Remove the timer 3:4 from reset (TIM34RS in TGCR).
3. Select the desired timer period for timer 3:4 (PRD34). Program with the desired timer period value - 1.
4. Select the desired prescaler value for timer 3:4 (PSC34 in TGCR).
5. Enable timer 3:4 (ENAMODE34 in TCR).
6. If ENAMODE34 = 3h, write the desired timer period for the next timer cycle in the period reload register (REL34). Program with the desired timer period value - 1. This step can be done at any time before the current timer cycle ends.

21.1.5.4.2.2.5 Event Capture Mode

When the PLUSEN bit in the timer global control register (TGCR) is set, Event Capture Mode is available for TIM12 when the timer is configured in 32-bit unchained mode. When Event Capture Mode is enabled, the timer cycle is restarted when an external input event occurs on pin TM64P_IN12. In particular, when an external input event occurs, the timer stops counting, generates output events (TINT12, TEVT12, and TM64P_OUT12), copies values from the timer counter register TIM12 to the timer capture register CAP12, reloads the timer period register PRD12 if in continuous mode with period reload (ENAMODE = 3h), and then restarts counting in continuous mode. Event Capture Mode is available only when the timer clock source is the internal timer (CLKSRC = 0) and the timer is in continuous mode (ENAMODE = 2h or 3h).

Capture mode is enabled using the Capture mode enable bit CAPMODE12 in the timer control register (TCR). The type of input event is selected by the capture event mode bit CAPEVTMODE12 in the timer control register (TCR). All of the following input event types are available:

- Rising edge of input signal
- Falling edge of input signal
- Rising or falling edge of input signal

21.1.5.4.2.2.6 Timer Counter Register Read Reset Mode

Read Reset Mode is available when the PLUSEN bit in the timer global control register (TGCR) is set and the timer is configured in 32-bit unchained mode. When Read Reset Mode is enabled, the timer cycle will restart when the timer counter registers are read (TIM12 and/or TIM34). In particular, when the timer registers are read, the timer stops counting, copies values from the timer counter registers (TIM12 and/or TIM34) to the timer capture registers (CAP12 and/or CAP34), reloads the timer period registers (PRD12 and/or PRD34) if in continuous mode with period reload (ENAMODE = 3h), and then restarts counting in continuous mode. Timer output events (TINT*n*, TEVT*n*, and TM64P_OUT*n*) are not generated during this process. Read Reset Mode is enabled using the read reset mode enable bit (READRSTMODE) in the timer control register (TCR).

Copyright © 2013–2016, Texas Instruments Incorporated

21.1.5.4.3 Timer Capture Registers

When the timer has a timeout due to a normal expiration of timer, external input event in Event Capture Mode, or read of timer counter registers in Read Reset Mode, the values of the timer counter registers (TIM12 and TIM34) are copied onto the timer counter capture registers (CAP12 and CAP34). Note that the value in TDDR is not captured when a read of TIM34 happens.

21.1.5.4.4 Counter and Period Registers Used in GP Timer Modes

Table 21-5 summarizes how the counter registers (TIM*n*) and period registers (PRD*n*) are used in each GP timer mode.

Table 21-5. Counter and Period Registers Used in GP Timer Modes

Timer Mode	Counter Registers	Period Registers
64-bit general-purpose	TIM34:TIM12	PRD34:PRD12
Dual 32-bit chained:		
Prescaler (Timer 3:4)	TIM34	PRD34
Timer (Timer 1:2)	TIM12	PRD12
Dual 32-bit unchained:		
Timer (Timer 1:2)	TIM12	PRD12
Timer with prescaler (Timer 3:4)	TDDR34 bits and TIM34	PSC34 bits and PRD34

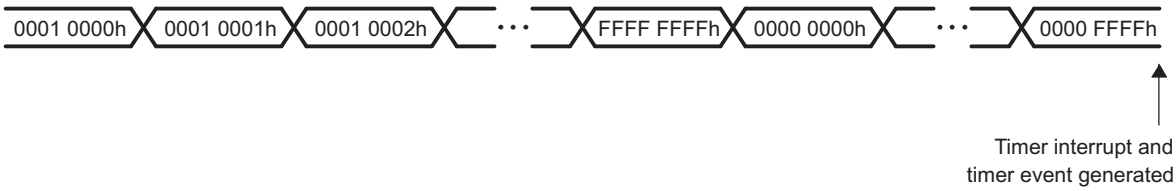
21.1.5.5 Timer Operation Boundary Conditions

The following boundary conditions affect the timer operation.

21.1.5.5.1 Timer Counter Overflow

Timer counter overflow can happen when the timer counter register is set to a value greater than the value in the timer period register. The counter reaches its maximum value (FFFF FFFFh or FFFF FFFF FFFF FFFFh), rolls over to 0, and continues counting until it reaches the timer period. An example is in Figure 21-8.

Figure 21-8. 32-Bit Timer Counter Overflow Example



21.1.5.5.2 Writing to Registers of an Active Timer

Writes to most timer registers are not allowed when the timer is active, except for setting the timer period reload registers (REL12 and REL34) and stopping and resetting the timers. In the 64-bit and dual 32-bit timer modes, registers that are protected by hardware are:

- TIM12
- TIM34
- PRD12
- PRD34
- TCR (except the ENAMODE bit)
- TGCR (except the TIM12RS and TIM34RS bits)

Copyright © 2013–2016, Texas Instruments Incorporated

21.1.5.6 General-Purpose Timer Power Management

The timer can be placed in reduced power modes to conserve power during periods of low activity. The power management of the peripheral is controlled by the processor Power and Sleep Controller (PSC). The PSC acts as a master controller for power management for all of the peripherals on the device. For detailed information on power management procedures using the PSC, see the *Power and Sleep Controller (PSC)* chapter. The timer can be placed in an idle mode to conserve power when it is not being used.

21.1.6 Architecture – Watchdog Timer Mode

This section describes the use of the timer as a watchdog timer. In order to fully function in watchdog timer mode, the timer must be internally connected to the device hardware reset signal. For information on which timer instantiation can function as a watchdog timer, see your device-specific data manual.

21.1.6.1 Watchdog Timer

As a 64-bit watchdog timer, the peripheral can be used to prevent system lockup when the software becomes trapped in loops with no controlled exit.

After a hardware reset, the watchdog timer is disabled. The timer then can be configured as a watchdog timer using the timer mode (TIMMODE) bit in the timer global control register (TGCR) and the watchdog timer enable (WDEN) bit in the watchdog timer control register (WDTCR). In the watchdog timer mode, the timer requires a special service sequence to be executed periodically. Without this periodic servicing, the timer counter increments until it matches the timer period and causes a watchdog timeout event.

When the timeout event occurs, the watchdog timer resets the entire processor.

21.1.6.2 Watchdog Timer Mode Restrictions

The watchdog timer mode has the following restrictions:

- No external clock source
- No one-time enabling

21.1.6.3 Watchdog Timer Mode Operation

The watchdog timer mode is selected and enabled when:

- TIMMODE = 2h in TGCR
- WDEN = 1 in WDTCR

Figure 21-9 shows the timer when it is used in the watchdog timer mode. The counter registers (TIM12 and TIM34) form a 64-bit timer counter register and the period registers (PRD12 and PRD34) form a 64-bit period register. When the timer counter matches the timer period, the timer generates a watchdog timeout event which resets the entire processor.

To activate the watchdog timer, a certain sequence of events must be followed, as shown in the state diagram of Figure 21-10.

Once the watchdog timer is activated, it can be disabled only by a watchdog timeout event or by a hardware reset. A special key sequence is required to prevent the watchdog timer from being accidentally serviced while the software is trapped in a loop or by some other software failure.

To prevent a watchdog timeout event, the timer has to be serviced periodically by writing A5C6h followed by DA7Eh to the watchdog timer service key (WDKEY) bits in WDTCR before the timer finishes counting up. Both A5C6h and DA7Eh are allowed to be written to the WDKEY bits, but only the correct sequence of A5C6h followed by DA7Eh to the WDKEY bits services the watchdog timer. Any other writes to the WDKEY bits triggers the watchdog timeout event immediately.

When the watchdog timer is in the Timeout state, the watchdog timer is disabled, the WDEN bit is cleared to 0, and the timer is reset.

Copyright © 2013–2016, Texas Instruments Incorporated

Figure 21-9. Watchdog Timer Mode Block Diagram

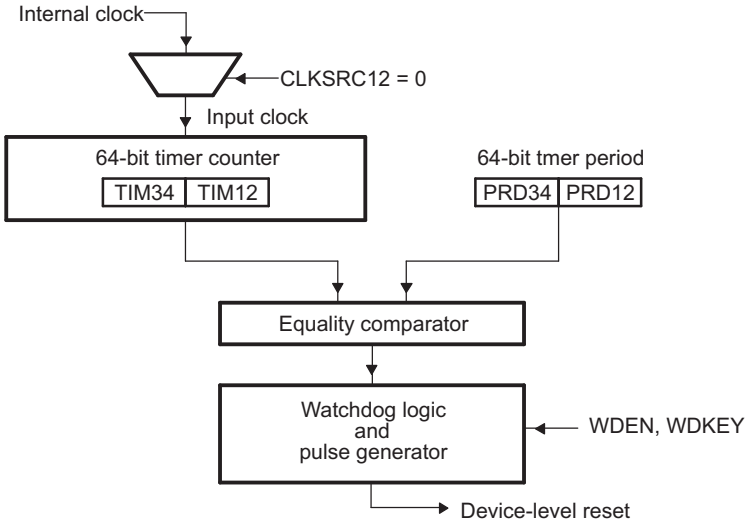
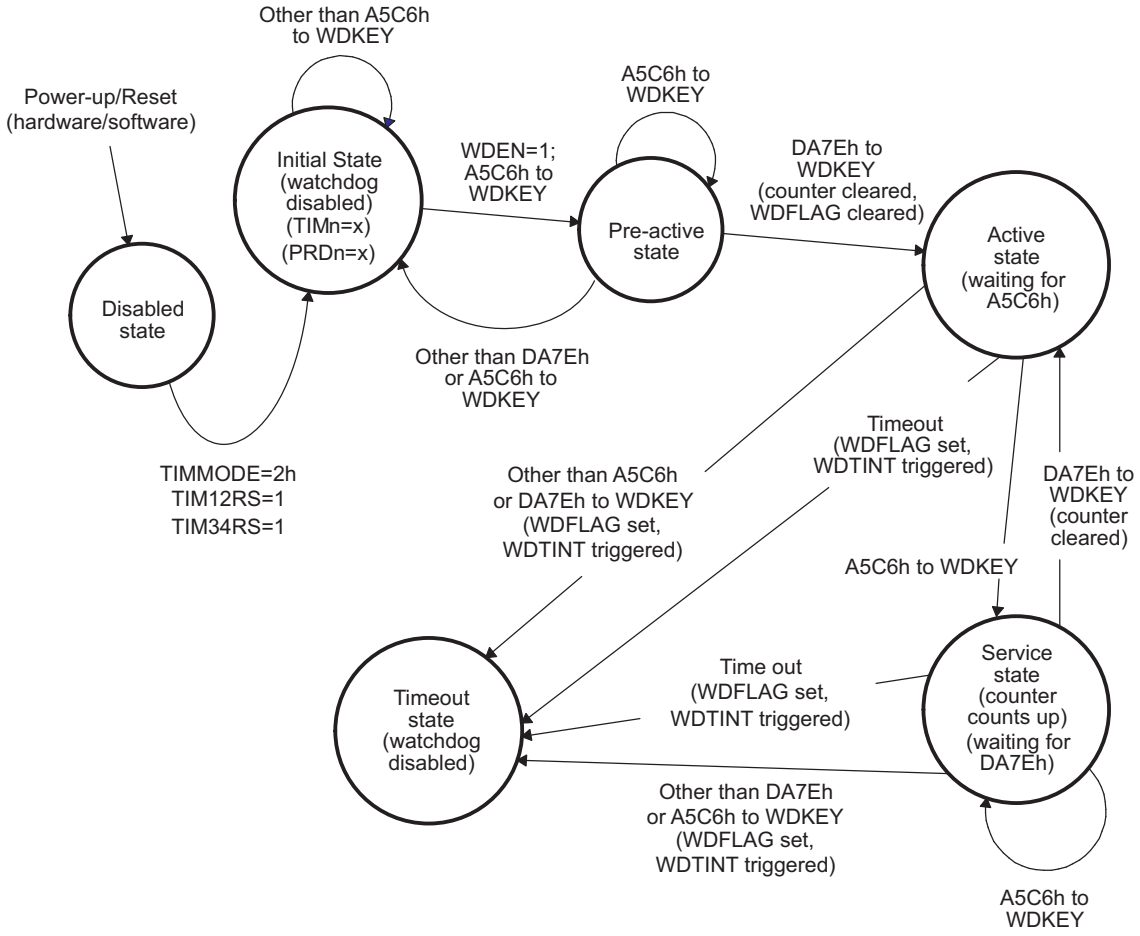


Figure 21-10. Watchdog Timer Operation State Diagram



Copyright © 2013–2016, Texas Instruments Incorporated

After a hardware reset, the watchdog timer is disabled; however, reads or writes to the watchdog timer registers are allowed. Once the WDEN bit is set (enabling the watchdog timer) and A5C6h is written to the WDKEY bits, the watchdog timer enters the Pre-active state. In the Pre-active state:

- A write to WDTCR is allowed only when the write comes with the correct key (A5C6h or DA7Eh) to the WDKEY bits.
- A write of DA7Eh to the WDKEY bits when the WDEN bit is set to 1 resets the counters and activates the watchdog timer.

The watchdog timer must be configured before the watchdog timer enters the Active state. The WDEN bit must be set to 1 before writing DA7Eh to the WDKEY bits in the Pre-active state. Every time the watchdog timer is serviced by the correct WDKEY sequence, the watchdog timer counter is automatically reset.

21.1.6.4 Watchdog Timer Register Write Protection

Once the watchdog timer enters the Pre-active state (see Figure 21-10), writes to TIM12, TIM34, PRD12, PRD34, and WDTCR are write protected (except for the WDKEY field). While the watchdog timer is in the Timeout state, writing to the WDEN bit has no effect.

Once the watchdog timer enters its Initial state (see Figure 21-10), do not write to TGCR.

21.1.6.5 Watchdog Timer Power Management

The watchdog timer cannot be placed in power-down mode.

21.1.7 Reset Considerations

The timer has two reset sources: hardware reset and the timer reset (TIM12RS and TIM34RS) bits in the timer global control register (TGCR).

21.1.7.1 Software Reset Considerations

When the TIM12RS bit in TGCR is cleared to 0, the TIM12 register is held with the current value.

When the TIM34RS bit in TGCR is cleared to 0, the TIM34 register is held with the current value.

21.1.7.2 Hardware Reset Considerations

When a hardware reset is asserted, all timer registers are set to their default values.

21.1.8 Interrupt Support

Each of the timers can send either one of two separate interrupt events (TINT*n*) to the CPU, depending on the operating mode of the timer. The timer interrupts are generated when the count value in the counter register reaches the value specified in the period register.

When the PLUSEN bit in the timer global control register (TGCR) is set, matches between TIM12 and CMP*n* in dual 32-bit unchained mode will also generate interrupts. Setting the PLUSEN bit also enables additional features for control, status, and generation of interrupts. See Section 21.1.11 for more information.

21.1.9 DMA Event Support

Each of the timers can send either one of two separate timer events (TEVT*n*) to the DMA engine, depending on the operating mode of the timer. The timer events are generated when the count value in the counters register reaches the value specified in the period register.

When the PLUSEN bit in the timer global control register (TGCR) is set, matches between TIM12 and CMP*n* in dual 32-bit unchained mode will also generate DMA events. Setting the PLUSEN bit also enables additional features for control, status, and generation of dma events are enabled. See Section 21.1.11 for more information.

Copyright © 2013–2016, Texas Instruments Incorporated

21.1.10 TM64P_OUT Event Support

The timer can generate an output pulse (Figure 21-11) or clock (Figure 21-12) signals on the TM64P_OUT12 pin. The output signal is generated when the count value in the counter registers reaches the value specified in the period registers (TSTAT12 drives the TM64P_OUT12 pin). Table 21-6 gives equations for various TSTAT12 timing parameters in pulse and clock modes.

The output mode is selected with the clock/pulse bit (CP*n*) in the timer control register (TCR). In pulse mode, the PWID12 bit in TCR sets the pulse width between 1 to 4 timer clock periods. The TM64P_OUT12 pin may be inverted using the INVOUTP12 bit in TCR.

Figure 21-11. Timer Operation in Pulse Mode (CP*n* = 0)

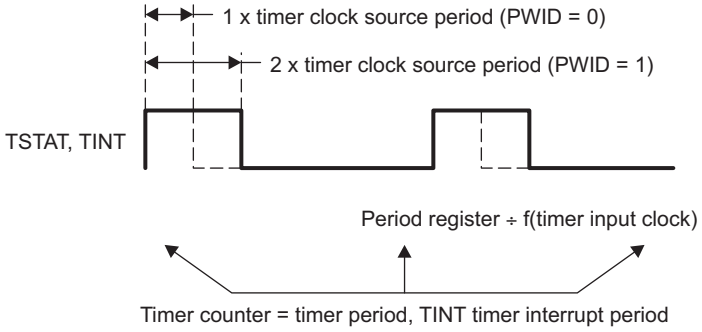


Figure 21-12. Timer Operation in Clock Mode (CP*n* = 1)

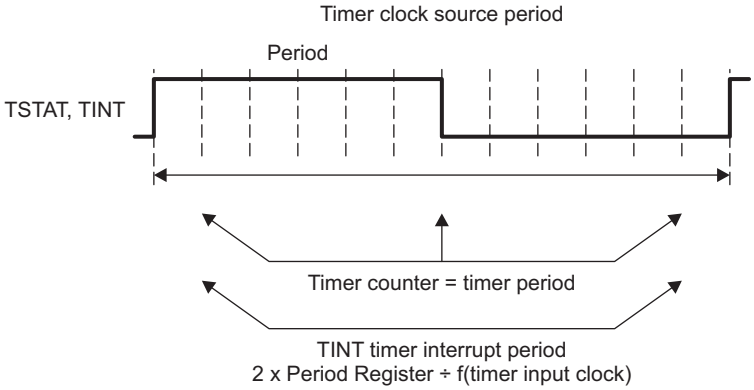


Table 21-6. TSTAT Parameters in Pulse and Clock Modes

Mode	Frequency	Period	Width High	Width Low
Pulse	$\frac{f\ (clock\ source)}{timer\ period\ register}$	$\frac{timer\ period\ register}{f\ (clock\ source)}$	$\frac{(PWID + 1)}{f\ (clock\ source)}$	$\frac{timer\ period\ register - (PWID + 1)}{f\ (clock\ source)}$
Clock	$\frac{f\ (clock\ source)}{2 \times timer\ period\ register}$	$\frac{2 \times timer\ period\ register}{f\ (clock\ source)}$	$\frac{timer\ period\ register}{f\ (clock\ source)}$	$\frac{timer\ period\ register}{f\ (clock\ source)}$

Copyright © 2013–2016, Texas Instruments Incorporated

External Timer Pin GPIO Mode

The external timer pins (TM64P_IN12 and TM64P_OUT12) can be individually configured to function as general-purpose input/output (GPIO) pins. In GPIO mode, the pins are able to detect and drive arbitrary data. The pins are also able to source external interrupt events. Some timer instantiations may not have external pins, see your device-specific data manual for pin information.

The GPIO interrupt and GPIO enable register (GPINTGPEN) enables the GPIO mode and associated interrupts. The GPIO data and GPIO direction register (GPDATGPDIR) determines if GPIO-enabled pins are used as input or output pins; and it is the means by which data is read-from or written-to the GPIO pins.

Normal timer counting modes cannot be used when the GPIO mode is enabled -- TIM12RS in the timer global control register (TGCR) cannot be brought out of reset when either GPENO12 or GPENI12 in GPINTGPEN is asserted.

21.1.11 Interrupt/DMA Event Generation Control and Status

When the PLUSEN bit in the timer global control register (TGCR) is set, the timer supports additional features for control and status of interrupt and DMA event generation. Interrupt/DMA events are generated when the count value in the timer counter registers reaches the value specified in the timer period registers and interrupt/DMA events are also generated when the Event Capture Mode is enabled and an external event occurs.

To generate events in the case when the value in the timer counter registers equals the value specified in the timer period registers, set the period compare interrupt enable bit (PRDINTEN*n*) in the interrupt control and status register (INTCTLSTAT). The event status for this case is reflected in the period compare interrupt status bit (PRDINTSTAT*n*), which is also in INTCTLSTAT. The PRDINTSTAT*n* bit is cleared by writing a 1 to the bit.

Similarly, to generate events in Event Capture Mode, set the event interrupt enable bit (EVTINTEN*n*) in INTCTLSTAT. The event status for this case is reflected in the external interrupt status bit (EVTINTSTAT*n*) in INTCTLSTAT. The EVTINTSTAT*n* bit is cleared by writing a 1 to the bit.

21.1.12 Power Management

The general-purpose timers can be placed in reduced power modes to conserve power during periods of low activity. The power management of the peripheral is controlled by the processor Power and Sleep Controller (PSC). The PSC acts as a master controller for power management for all of the peripherals on the device. For detailed information on power management procedures using the PSC, see the *Power and Sleep Controller (PSC)* chapter.

21.1.13 Emulation Considerations

Each timer has an emulation management register (EMUMGT). As shown in Table 21-7, the FREE and SOFT bits of EMUMGT determine how the timer responds to an emulation suspend event. An emulation suspend event corresponds to any type of emulator access to the CPU, such as a hardware or software breakpoint or a probe point.

Note that during emulation, the timer count values will increment once every timer peripheral clock (not CPU clock). So when single-steeping though code, the timer values will not update on every CPU clock cycle.

The timer can respond to emulation events from the CPU based on the configuration of the Emulation Suspend Source Register (SUSPSRC) in the System Configuration Module. See the *System Configuration (SYSCFG) Module* chapter for information on SUSPSRC and how it is configured.

Copyright © 2013–2016, Texas Instruments Incorporated

Table 21-7. Timer Emulation Modes Selection

FREE	SOFT	Emulation Mode
0	0	The timer stops immediately.
0	1	The timer stops when the timer counter value increments and reaches the value in the timer period register.
1	x	The timer runs free regardless of SOFT bit status.

21.2 Registers

Table 21-8 lists the memory-mapped registers for the 64-bit Timer Plus. See your device-specific data manual for the memory address of these registers. All other register offset addresses not listed in Table 21-8 should be considered as reserved locations and the register contents should not be modified.

Table 21-8. Timer Registers

Offset	Acronym	Register Description	Section
0h	REVID	Revision ID Register	Section 21.2.1
4h	EMUMGT	Emulation Management Register	Section 21.2.2
8h	GPINTGPEN	GPIO Interrupt and GPIO Enable Register	Section 21.2.3
Ch	GPDATGPDIR	GPIO Data and GPIO Direction Register	Section 21.2.4
10h	TIM12	Timer Counter Register 12	Section 21.2.5
14h	TIM34	Timer Counter Register 34	Section 21.2.5
18h	PRD12	Timer Period Register 12	Section 21.2.6
1Ch	PRD34	Timer Period Register 34	Section 21.2.6
20h	TCR	Timer Control Register	Section 21.2.7
24h	TGCR	Timer Global Control Register	Section 21.2.8
28h	WDTCR	Watchdog Timer Control Register	Section 21.2.9
34h	REL12	Timer Reload Register 12	Section 21.2.10
38h	REL34	Timer Reload Register 34	Section 21.2.11
3Ch	CAP12	Timer Capture Register 12	Section 21.2.12
40h	CAP34	Timer Capture Register 34	Section 21.2.13
44h	INTCTLSTAT	Timer Interrupt Control and Status Register	Section 21.2.14
60h	CMP0	Compare Register 0	Timer Compare Registers (CMP0-CMP7)
64h	CMP1	Compare Register 1	Timer Compare Registers (CMP0-CMP7)
68h	CMP2	Compare Register 2	Timer Compare Registers (CMP0-CMP7)
6Ch	CMP3	Compare Register 3	Timer Compare Registers (CMP0-CMP7)
70h	CMP4	Compare Register 4	Timer Compare Registers (CMP0-CMP7)
74h	CMP5	Compare Register 5	Timer Compare Registers (CMP0-CMP7)
78h	CMP6	Compare Register 6	Timer Compare Registers (CMP0-CMP7)

Copyright © 2013–2016, Texas Instruments Incorporated

Table 21-8. Timer Registers (continued)

Offset	Acronym	Register Description	Section
7Ch	CMP7	Compare Register 7	Timer Compare Registers (CMP0-CMP7)

Copyright © 2013–2016, Texas Instruments Incorporated

21.2.1 Revision ID Register (REVID)

The revision ID register (REVID) contains the peripheral revision. The REVID is shown in Figure 21-13 and described in Table 21-9.

Figure 21-13. Revision ID Register (REVID)

31	0
REV	
R-4472 020Ch	

LEGEND: R = Read only; -*n* = value after reset

Table 21-9. Revision ID Register (REVID) Field Descriptions

Bit	Field	Value	Description
31-0	REV	4472 020Ch	Revision ID of the Timer.

21.2.2 Emulation Management Register (EMUMGT)

The emulation management register (EMUMGT) is shown in Figure 21-14 and described in Table 21-10.

Figure 21-14. Emulation Management Register (EMUMGT)

31	16
Reserved	
R-0	

15	2	1	0
Reserved		SOFT	FREE
R-0		R/W-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 21-10. Emulation Management Register (EMUMGT) Field Descriptions

Bit	Field	Value	Description
31-2	Reserved	0	Reserved
1	SOFT		Determines emulation mode functionality of the timer. When the FREE bit is cleared to 0, the SOFT bit selects the timer mode.
		0	The timer stops immediately.
		1	The timer stops when the counter increments and reaches the value in the timer period register (PRD*n*).
0	FREE		Determines emulation mode functionality of the timer. When the FREE bit is cleared to 0, the SOFT bit selects the timer mode.
		0	The SOFT bit selects the timer mode.
		1	The timer runs free regardless of the SOFT bit.

Copyright © 2013–2016, Texas Instruments Incorporated

21.2.3 GPIO Interrupt Control and Enable Register (GPINTGPEN)

The GPIO interrupt control and enable register (GPINTGPEN) is shown in Figure 21-15 and described in Table 21-11.

Figure 21-15. GPIO Interrupt Control and Enable Register (GPINTGPEN)

31	24
Reserved	
R/W-0	

23	18	17	16
Reserved		GPENO12	GPENI12
R/W-0		R/W-0	R/W-0

15	8
Reserved	
R/W-0	

7	6	5	4	3	2	1	0
Reserved		GPINT12INVO	GPINT12INVI	Reserved		GPINT12ENO	GPINT12ENI
R-0		R/W-0	R/W-0	R-0		R/W-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 21-11. GPIO Interrupt Control and Enable Register (GPINTGPEN) Field Descriptions

Bit	Field	Value	Description
31-18	Reserved	0	Reserved
17	GPENO12		Enable TM64P_OUT12 to function in GPIO mode.
		0	TM64P_OUT12 is used as a TIMER output pin.
		1	TM64P_OUT12 is used as a GPIO pin.
16	GPENI12		Enable TM64P_IN12 to function in GPIO mode
		0	TM64P_IN12 is used as a TIMER input pin.
		1	TM64P_IN12 is used as a GPIO pin.
15-6	Reserved	0	Reserved
5	GPINT12INVO		Invert interrupt/event signal from TM64P_OUT12 when GPINT12ENO = 1.
		0	Rising signal edge on TM64P_OUT12 generates the interrupt/event.
		1	Falling signal edge on TM64P_OUT12 generates the interrupt/event.
4	GPINT12INVI		Invert interrupt/event signal for TM64P_IN12 when GPINT12ENI = 1.
		0	Rising signal edge on TM64P_IN12 generates the interrupt/event.
		1	Falling signal edge on TM64P_IN12 generates the interrupt/event.
3-2	Reserved	0	Reserved
1	GPINT12ENO		Enable TM64P_OUT12 to source interrupts/events in GPIO mode.
		0	Timer interrupts/events are sourced in TIMER mode.
		1	Timer interrupts/events are sourced externally from TM64P_OUT12.
0	GPINT12ENI		Enable TM64P_IN12 to source interrupts/events in GPIO mode.
		0	Timer interrupts/events are sourced in TIMER mode.
		1	Timer interrupts/events are sourced externally from TM64P_IN12.

Copyright © 2013–2016, Texas Instruments Incorporated

21.2.4 GPIO Data and Direction Register (GPDATGPDIR)

The GPIO data and direction register (GPDATGPDIR) is shown in Figure 21-16 and described in Table 21-12.

Figure 21-16. GPIO Data and Direction Register (GPDATGPDIR)

31	18	17	16
Reserved		GPDIRO12	GPDIRI12
R/W-0		R/W-0	R/W-0

15	2	1	0
Reserved		GPDATO12	GPDATI12
R/W-0		R/W-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 21-12. GPIO Data and Direction Register (GPDATGPDIR) Field Descriptions

Bit	Field	Value	Description
31-18	Reserved	0	Reserved
17	GPDIRO12		Select direction of TM64P_OUT12 in GPIO mode.
		0	TM64P_OUT12 functions as an input pin in GPIO mode.
		1	TM64P_OUT12 functions as an output pin in GPIO mode (TM64P_OUT12 cannot capture GPIO interrupt events when configured as output).
16	GPDIRI12		Select direction of TM64P_IN12 in GPIO mode.
		0	TM64P_IN12 functions as an input pin in GPIO mode.
		1	TM64P_IN12 functions as an output pin in GPIO mode (TM64P_IN12 cannot capture GPIO interrupt events when configured as output).
15-2	Reserved	0	Reserved
1	GPDATO12		Data on TM64P_OUT12 in GPIO mode. Only valid when GPENO12 = 1.
			When GPDIRO12 = 0 (input):
		0	TM64P_OUT12 is detected logic low.
		1	TM64P_OUT12 is detected logic high.
			When GPDIRO12 = 1 (output):
		0	TM64P_OUT12 is driven logic low.
		1	TM64P_OUT12 is driven logic high.
0	GPDATI12		Data on TM64P_IN12 in GPIO mode. Only valid when GPENI12 = 1.
			When GPDIRI12 = 0 (input):
		0	TM64P_IN12 is detected logic low.
		1	TM64P_IN12 is detected logic high.
			When GPDIRI12 = 1 (output):
		0	TM64P_IN12 is driven logic low.
		1	TM64P_IN12 is driven logic high.

Copyright © 2013–2016, Texas Instruments Incorporated

21.2.5 Timer Counter Registers (TIM12 and TIM34)

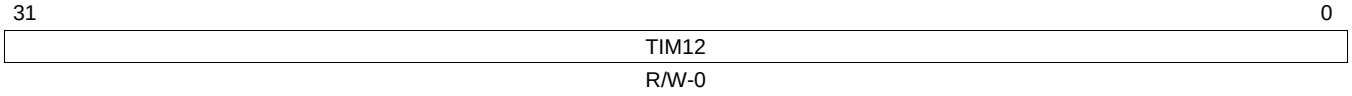
The timer counter register is a 64-bit wide register. This 64-bit register is divided into two 32-bit registers, TIM12 and TIM34.

In the dual 32-bit timer mode, the 64-bit register is divided with TIM12 acting as one 32-bit counter and TIM34 acting as another. These two registers can be configured as chained or unchained.

21.2.5.1 Timer Counter Register 12 (TIM12)

The timer counter register 12 (TIM12) is shown in Figure 21-17 and described in Table 21-13

Figure 21-17. Timer Counter Register 12 (TIM12)



LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

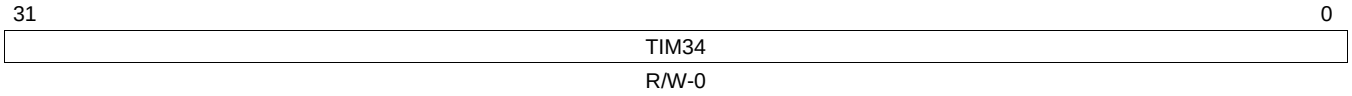
Table 21-13. Timer Counter Register 12 (TIM12) Field Descriptions

Bit	Field	Value	Description
31-0	TIM12	0-FFFF FFFFh	TIM12 count bits. This 32-bit value is the current count of the main counter.

21.2.5.2 Timer Counter Register 34 (TIM34)

The timer counter register 34 (TIM34) is shown in Figure 21-18 and described in Table 21-14.

Figure 21-18. Timer Counter Register 34 (TIM34)



LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 21-14. Timer Counter Register 34 (TIM34) Field Descriptions

Bit	Field	Value	Description
31-0	TIM34	0-FFFF FFFFh	TIM34 count bits. This 32-bit value is the current count of the main counter.

Copyright © 2013–2016, Texas Instruments Incorporated

21.2.6 Timer Period Registers (PRD12 and PRD34)

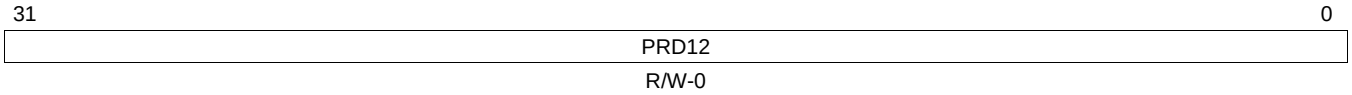
The timer period register is a 64-bit wide register. This 64-bit register is divided into two 32-bit registers, PRD12 and PRD34.

Similar to TIM*n* in the dual 32-bit timer mode, PRD*n* can be divided into 2 registers: for timer 1:2, PRD12 and for timer 3:4, PRD34. These two registers can be used in conjunction with the two timer counter registers TIM12 and TIM34.

21.2.6.1 Timer Period Register 12 (PRD12)

The timer period register 12 (PRD12) is shown in Figure 21-19 and described in Table 21-15.

Figure 21-19. Timer Period Register 12 (PRD12)



LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

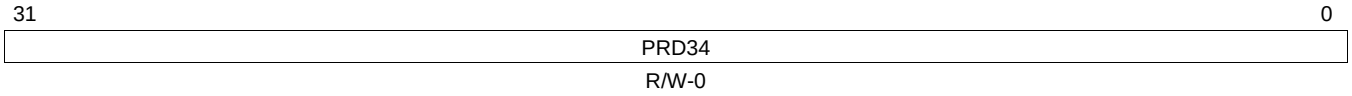
Table 21-15. Timer Period Register (PRD12) Field Descriptions

Bit	Field	Value	Description
31-0	PRD12	0-FFFF FFFFh	PRD12 period bits. This 32-bit value is the number of timer input clock cycles to count.

21.2.6.2 Timer Period Register 34 (PRD34)

The timer period register 34 (PRD34) is shown in Figure 21-20 and described in Table 21-16.

Figure 21-20. Timer Period Register 34 (PRD34)



LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 21-16. Timer Period Register (PRD34) Field Descriptions

Bit	Field	Value	Description
31-0	PRD34	0-FFFF FFFFh	PRD34 period bits. This 32-bit value is the number of timer input clock cycles to count.

Copyright © 2013–2016, Texas Instruments Incorporated

21.2.7 Timer Control Register (TCR)

The timer control register (TCR) is shown in Figure 21-21 and described in Table 21-17.

Figure 21-21. Timer Control Register (TCR)

31	27	26	25	24
Reserved		READRSTMODE34	Reserved	
R/W-0		R/W-0	R/W-0	

23	22	21	16
ENAMODE34		Reserved	
R/W-0		R/W-0	

15	14	13	12	11	10	9	8
Reserved		CAPVTMODE12		CAPMODE12	READRSTMODE12	TIEN12	CLKSRC12
R-0		R/W-0		R/W-0	R/W-0	R/W-0	R/W-0

7	6	5	4	3	2	1	0
ENAMODE12		PWID12		CP12	INVINP12	INVOUTP12	TSTAT12
R/W-0		R/W-0		R/W-0	R/W-0	R/W-0	R-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 21-17. Timer Control Register (TCR) Field Descriptions

Bit	Field	Value	Description
31-27	Reserved	0	Reserved
26	READRSTMODE34		Read reset mode enable bit. Determines the effect of a timer counter read on TIM34. Read reset mode is only available in dual 32-bit unchained. Output events (interrupt/EDMA/other) are not generated when read reset occurs.
		0	There is no effect when timer counter register TIM34 is read.
		1	Timer counter is reset when timer counter register TIM34 is read.
25-24	Reserved	0	Reserved
23-22	ENAMODE34	0-3h	Enabling mode: determines the enabling modes fo the timer.
		0	The timer is disabled (not counting) and maintains current value.
		1h	The timer is enabled one time. The timer stops after the counter reaches the period.
		2h	The timer is enabled continuously, TIM34 increments until the timer counter matches the period, resets the timer counter to 0 on the cycle after matching and continues.
		3h	The timer is enabled continuously with period reload, TIM*n* increments until the timer counter matches the period, resets the timer counter to 0 on the cycle after matching, reloads the period register with the values in the reload registers (REL*n*), and continues counting.
21-14	Reserved	0	Reserved
13-12	CAPEVTMODE12	0-3h	Capture event mode. Uses these bits to specify the type of event for Capture mode.
		0	Event occurs on timer input rising edge.
		1h	Event occurs on time input falling edge.
		2h	Event occurs on both rising and falling edges.
		3h	Reserved
11	CAPMODE12		Capture mode enable bit. Determines if external event can reset timer. Capture mode is only available in dual 32-bit unchained mode and when CLKSRC = 0 and ENAMODE = 2h or 3h. Output events (interrupt/EDMA/other) are generated when capture mode event occurs.
		0	Timer is not in capture mode.
		1	Timer is in capture mode. External event can reset timer.
10	READRSTMODE12		Read reset mode enable bit. Determines the effect of a timer counter read on TIM12. Read reset mode is only available in dual 32-bit unchained. Output events (interrupt/EDMA/other) are not generated when read reset occurs.
		0	There is no effect when timer counter register TIM12 is read.
		1	Timer counter is reset when timer counter register TIM12 is read.

Copyright © 2013–2016, Texas Instruments Incorporated

Table 21-17. Timer Control Register (TCR) Field Descriptions (continued)

Bit	Field	Value	Description
9	TIEN12		Timer input gate enable bit. Allows timer input pin TM64P_IN12 to gate the internal timer clock source (CLKSRC = 0). Timer starts counting when TM64P_IN12 transitions from low to high. Timer stops counting when TM64P_IN12 transitions from high to low.
		0	Timer clock is not gated by TM64P_IN12.
		1	Timer clock is gated by TM64P_IN12.
8	CLKSRC12		CLKSRC determines the selected clock source for the timer.
		0	Internal clock
		1	External clock on TM64P_IN12
7-6	ENAMODE12	0-3h	Enabling mode: determines the enabling modes fo the timer.
		0	The timer is disabled (not counting) and maintains current value.
		1h	The timer is enabled one time. The timer stops after the counter reaches the period.
		2h	The timer is enabled continuously, TIM*n* increments until the timer counter matches the period, resets the timer counter to 0 on the cycle after matching and continues.
		3h	The timer is enabled continuously with period reload, TIM*n* increments until the timer counter matches the period, resets the timer counter to 0 on the cycle after matching, reloads the period register with the values in the reload registers (REL*n*), and continues counting.
5-4	PWID12	0-3h	Pulse width - Determines the pulse width on the TSTAT12 bit (and the TM64P_OUT12 pin) when the clock/pulse mode is set to pulse.
		0	TSTAT12 stays active for one timer clock cycle when the timer counter reaches the period.
		1h	TSTAT12 stays active for two timer clock cycles when the timer counter reaches the period.
		2h	TSTAT12 stays active for three timer clock cycles when the timer counter reaches the period.
		3h	TSTAT12 stays active for four timer clock cycles when the timer counter reaches the period.
3	CP12		Clock/Pulse bit - Determines whether the TM64P_OUT12 output event should behave as a 50% duty-cycle clock or a signal pulse.
		0	Pulse Mode. TM64P_OUT12 goes active after the timer counter reaches the period. The pulse width is determined by PWID12.
		1	Clock Mode. TM64P_OUT12 will behave as a 50% duty cycle signal. It toggles high-to-low or low-to-high when the timer counter reaches zero.
2	INVINP12		Invert TM64P_IN12. Only affects operation if CLKSRC = 1.
		0	Uninverted TM64P_IN12 signal drives timer.
		1	Inverted TM64P_IN12 signal drives timer.
1	INVOUTP12		Invert TM64P_OUT12.
		0	TM64P_OUT12 signal is not inverted.
		1	TM64P_OUT12 signal is inverted.
0	TSTAT12		Timer status. Drives the value of timer output TM64P_OUT12 when it is configured to function as timer output.
		0	TM64P_OUT12 signal is not asserted.
		1	TM64P_OUT12 signal is asserted.

Copyright © 2013–2016, Texas Instruments Incorporated

21.2.8 Timer Global Control Register (TGCR)

The timer global control register (TGCR) is shown in Figure 21-22 and described in Table 21-18.

Figure 21-22. Timer Global Control Register (TGCR)

31							16
Reserved							
R-0							
15		12	11				8
TDDR34			PSC34				
R/W-0			R/W-0				
7	5	4	3	2	1	0	
Reserved		PLUSEN	TIMMODE		TIM34RS	TIM12RS	
R-0		R/W-0	R/W-0		R/W-0	R/W-0	

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 21-18. Timer Global Control Register (TGCR) Field Descriptions

Bit	Field	Value	Description
31-16	Reserved	0	Reserved
15-12	TDDR34	0-Fh	Timer linear divide-down ratio specifies the timer divide-down ratio for timer 3:4. When the timer is enabled, TDDR34 increments every timer clock. The TIM34 counter increments on the cycle after TDDR34 matches PSC34. TDDR34 resets to 0 and continues. When TIM34 matches PRD34, timer 3:4 stops, if timer 3:4 is enabled one time; TIM34 resets to 0 on the cycle after matching PRD34 and timer 3:4 continues, if timer 3:4 is enabled continuously.
11-8	PSC34	0-Fh	TIM34 pre-scalar counter specifies the count for timer 3:4.
7-5	Reserved	0	Reserved
4	PLUSEN		Enable new timer plus features.
		0	Enable backward compatibility. New timer features are unavailable.
		1	Disable backward compatibility. New timer features are available.
3-2	TIMMODE	0-3h	TIMMODE determines the timer mode.
		0	The timer is in 64-bit GP timer mode.
		1h	The timer is in dual 32-bit timer unchained mode.
		2h	The timer is in 64-bit watchdog timer mode.
		3h	The timer is in dual 32-bit timer, chained mode.
1	TIM34RS		Timer 3:4 reset.
		0	Timer 3:4 is in reset.
		1	Timer 3:4 is not in reset. Timer 3:4 can be used as a 32-bit timer. Note that for the timer to function properly in 64-bit timer mode, both TIM34RS and TIM12RS must be set to 1. Changing this bit does not affect the timer, if the timer is in the watchdog active state.
0	TIM12RS		Timer 1:2 reset.
		0	Timer 1:2 is in reset.
		1	Timer 1:2 is not in reset. Timer 1:2 can be used as a 32-bit timer. Note that for the timer to function properly in 64-bit timer mode, both TIM34RS and TIM12RS must be set to 1. Changing this bit does not affect the timer, if the timer is in the watchdog active state.

Copyright © 2013–2016, Texas Instruments Incorporated

21.2.9 Watchdog Timer Control Register (WDTCR)

The watchdog timer control register (WDTCR) is shown in Figure 21-23 and described in Table 21-19.

Figure 21-23. Watchdog Timer Control Register (WDTCR)

31	16
WDKEY	
R/W-0	

15	14	13	12	11	0
WDFLAG	WDEN	Reserved		Reserved	
R/W-0	R/W-0	R/W-0		R-0	

LEGEND: R/W = Read/Write; R = Read only; *-n* = value after reset

Table 21-19. Watchdog Timer Control Register (WDTCR) Field Descriptions

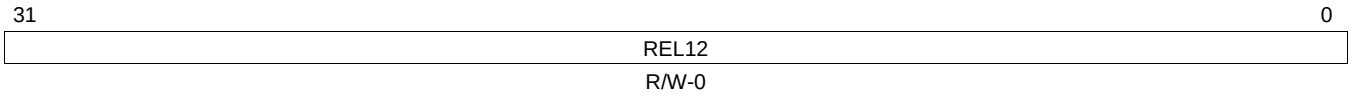
Bit	Field	Value	Description
31-16	WDKEY	0-FFFFh	16-bit watchdog timer service key. Only the sequence of an A5C6h followed by a DA7Eh services the watchdog. Not applicable in regular timer mode.
15	WDFLAG		Watchdog flag bit. WDFLAG can be cleared by enabling the watchdog timer, by device reset, or being written with 1. It is set by a watchdog time-out.
		0	No watchdog time-out occurred.
		1	Watchdog time-out occurred.
14	WDEN		Watchdog timer enable bit.
		0	Disable watchdog timer
		1	Enable watchdog timer
13-12	Reserved	0	Reserved. This bit field must be written as 00b.
11-0	Reserved	0	Reserved

Copyright © 2013–2016, Texas Instruments Incorporated

21.2.10 Timer Reload Register 12 (REL12)

The timer reload register 12 (REL12) is shown in Figure 21-24 and described in Table 21-20.

Figure 21-24. Timer Reload Register 12 (REL12)



LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

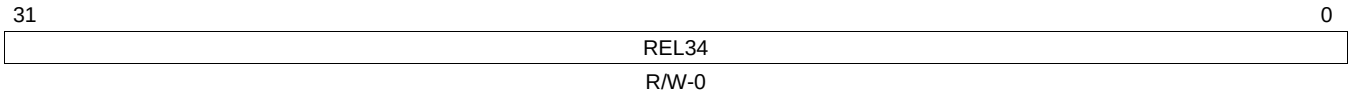
Table 21-20. Timer Reload Register 12 (REL12) Field Descriptions

Bit	Field	Value	Description
31- 0	REL12	0-FFFF FFFFh	Period reload bits.

21.2.11 Timer Reload Register 34 (REL34)

The timer reload register 34 (REL34) is shown in Figure 21-25 and described in Table 21-21.

Figure 21-25. Timer Reload Register 34 (REL34)



LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 21-21. Timer Reload Register 34 (REL34) Field Descriptions

Bit	Field	Value	Description
31- 0	REL34	0-FFFF FFFFh	Period reload bits.

Copyright © 2013–2016, Texas Instruments Incorporated

21.2.12 Timer Capture Register 12 (CAP12)

The timer capture register 12 (CAP12) is shown in Figure 21-26 and described in Table 21-22.

Figure 21-26. Timer Capture Register 12 (CAP12)

31	0
CAP12	
R/W-0	

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 21-22. Timer Capture Register 12 (CAP12) Field Descriptions

Bit	Field	Value	Description
31- 0	CAP12	0-FFFF FFFFh	Captured timer counter bits.

21.2.13 Timer Capture Register 34 (CAP34)

The timer capture register 34 (CAP34) is shown in Figure 21-27 and described in Table 21-23.

Figure 21-27. Timer Capture Register 34 (CAP34)

31	0
CAP34	
R/W-0	

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 21-23. Timer Capture Register 34 (CAP34) Field Descriptions

Bit	Field	Value	Description
31- 0	CAP34	0-FFFF FFFFh	Captured timer counter bits.

Copyright © 2013–2016, Texas Instruments Incorporated

21.2.14 Timer Interrupt Control and Status Register (INTCTLSTAT)

The timer interrupt control and status register (INTCTLSTAT) is shown in Figure 21-28 and described in Table 21-24.

Figure 21-28. Timer Interrupt Control and Status Register (INTCTLSTAT)

31	24
Reserved	
R-0	

23	20	19	18	17	16
Reserved		EVTINTSTAT34	EVTINTEN34	PRDINTSTAT34	PRDINTEN34
R-0		R/W1C-0	R/W-0	R/W1C-0	R/W-0

15	8
Reserved	
R-0	

7	4	3	2	1	0
Reserved		EVTINTSTAT12	EVTINTEN12	PRDINTSTAT12	PRDINTEN12
R-0		R/W1C-0	R/W-0	R/W1C-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; W1C = Write 1 to clear bit; -*n* = value after reset

Table 21-24. Timer Interrupt Control and Status Register (INTCTLSTAT) Field Descriptions

Bit	Field	Value	Description
31-20	Reserved	0	Reserved
19	EVTINTSTAT34		Interrupt status which reflects the condition that an external event caused a timeout when timer is in capture mode. Write a 1 to clear this bit.
		0	Interrupt has not occurred.
		1	Interrupt has occurred.
18	EVTINTEN34		Enables the interrupt generation when timer is in capture mode.
		0	Disable interrupt when in event capture mode.
		1	Enable interrupt when in event capture mode.
17	PRDINTSTAT34		Interrupt status which reflects the condition that timer counter matched the period register when timer is enabled. Write a 1 to clear this bit.
		0	Interrupt has not occurred.
		1	Interrupt has occurred.
16	PRDINTEN34		Enable interrupt generation when timer is enabled in 64-bit/32-bit chained/unchained/watchdog modes.
		0	Disable interrupt
		1	Enable interrupt
15-4	Reserved	0	Reserved
3	EVTINTSTAT12		Interrupt status which reflects the condition that an external event caused a timeout when timer is in capture mode. Write a 1 to clear this bit.
		0	Interrupt has not occurred.
		1	Interrupt has occurred.
2	EVTINTEN12		Enables the interrupt generation when timer is in capture mode.
		0	Disable interrupt when in event capture mode.
		1	Enable interrupt when in event capture mode.
1	PRDINTSTAT12		Interrupt status which reflects the condition that timer counter matched the period register when timer is enabled. Write a 1 to clear this bit.
		0	Interrupt has not occurred.
		1	Interrupt has occurred.

Copyright © 2013–2016, Texas Instruments Incorporated

Table 21-24. Timer Interrupt Control and Status Register (INTCTLSTAT) Field Descriptions (continued)

Bit	Field	Value	Description
0	PRDINTEN12		Enable interrupt generation when timer is enabled in 64-bit/32-bit chained/unchained/watchdog modes.
		0	Disable interrupt
		1	Enable interrupt

Timer Compare Registers (CMP0-CMP7)

The timer compare register (CMP*n*) is shown in Figure 21-29 and described in Table 21-25.

Figure 21-29. Timer Compare Register (CMP*n*)

31	0
CMP*n*	
R/W-0	

LEGEND: R/W = Read/Write; -*n* = value after reset

Table 21-25. Timer Compare Register (CMP*n*) Field Descriptions

Bit	Field	Value	Description
31-0	CMP*n*	0-FFFF FFFFh	Timer compare register. When PLUSEN = 1 in the timer global control register (TGCR) and the timer is configured in 32-bit unchained mode, TIM12 is compared to all 8 compare registers (CMP0-CMP7). When CMP*n* matches TIM12, a timer CMP*n* interrupt and DMA event are generated. A CMP*n* match will not affect the TIM12 count or behavior.

Copyright © 2013–2016, Texas Instruments Incorporated

Universal Asynchronous Receiver/Transmitter (UART)

This chapter describes the universal asynchronous receiver/transmitter (UART) peripheral. See your device-specific data manual to determine how many UARTs are available on your device.

Topic **Page**

22.1 Introduction **787**
22.2 Peripheral Architecture **789**
22.3 Registers **800**

Copyright © 2013–2016, Texas Instruments Incorporated

22.1 Introduction

22.1.1 Purpose of the Peripheral

The UART peripheral is based on the industry standard TL16C550 asynchronous communications element, which in turn is a functional upgrade of the TL16C450. Functionally similar to the TL16C450 on power up (single character or TL16C450 mode), the UART can be placed in an alternate FIFO (TL16C550) mode. This relieves the CPU of excessive software overhead by buffering received and transmitted characters. The receiver and transmitter FIFOs store up to 16 bytes including three additional bits of error status per byte for the receiver FIFO.

The UART performs serial-to-parallel conversions on data received from a peripheral device and parallel-to-serial conversion on data received from the CPU. The CPU can read the UART status at any time. The UART includes control capability and a processor interrupt system that can be tailored to minimize software management of the communications link.

The UART includes a programmable baud generator capable of dividing the UART input clock by divisors from 1 to 65535 and producing a 16× reference clock or a 13× reference clock for the internal transmitter and receiver logic. For detailed timing and electrical specifications for the UART, see your device-specific data manual.

22.1.2 Features

Check your device-specific data manual to see the list of features that are supported and that are not supported by the UART.

22.1.3 Functional Block Diagram

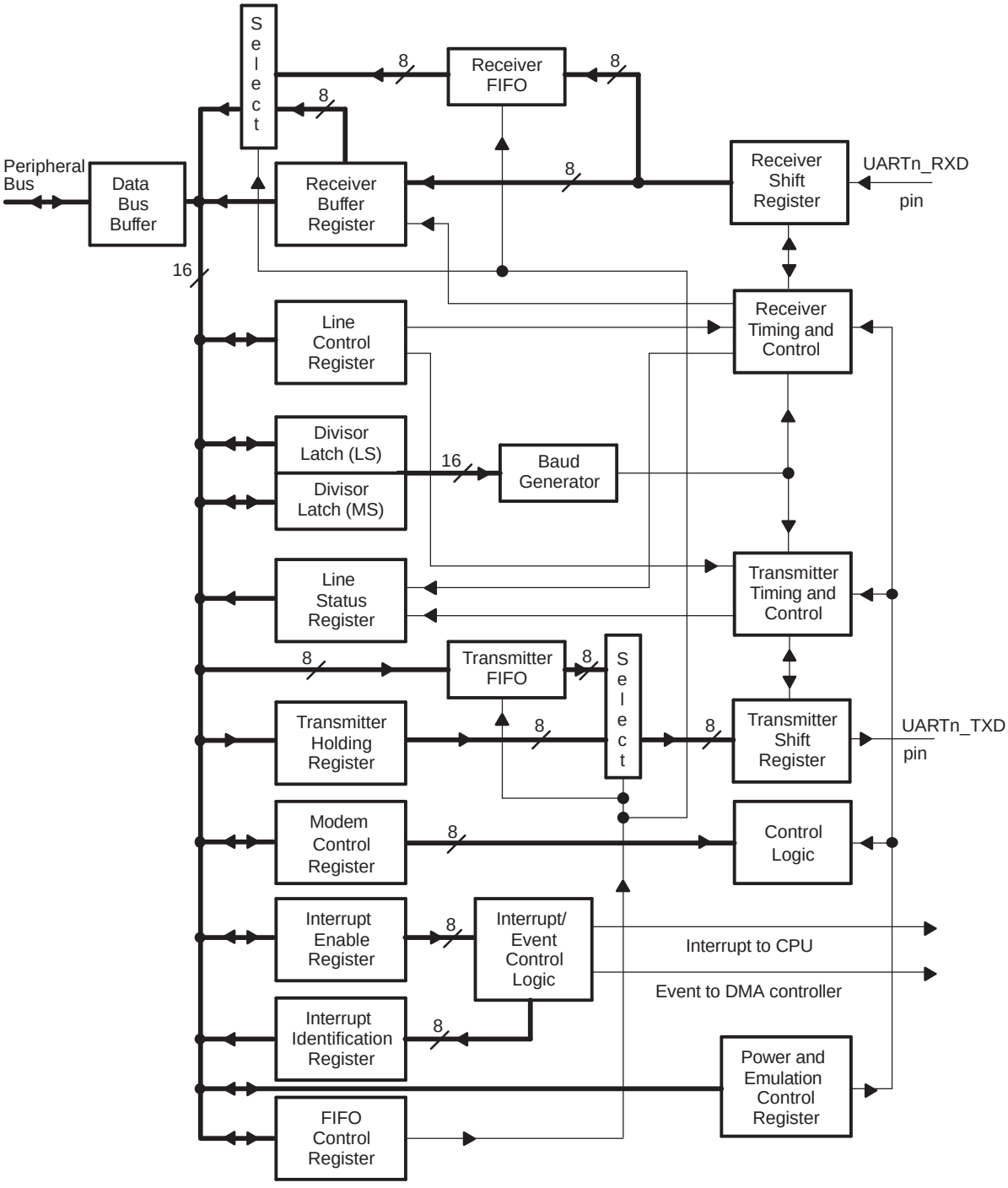
A functional block diagram of the UART is shown in Figure 22-1.

22.1.4 Industry Standard(s) Compliance Statement

The UART peripheral is based on the industry standard TL16C550 asynchronous communications element, which is a functional upgrade of the TL16C450. The information in this chapter assumes you are familiar with these standards.

Copyright © 2013–2016, Texas Instruments Incorporated

Figure 22-1. UART Block Diagram



NOTE: The value *n* indicates the applicable UART; that is, UART0, UART1, and so on.

Copyright © 2013–2016, Texas Instruments Incorporated

22.2 Peripheral Architecture

22.2.1 Clock Generation and Control

The UART bit clock is derived from an input clock to the UART. See your device-specific data manual to check the maximum data rate supported by the UART.

Figure 22-2 is a conceptual clock generation diagram for the UART. The processor clock generator receives a signal from an external clock source and produces a UART input clock with a programmed frequency. The UART contains a programmable baud generator that takes an input clock and divides it by a divisor in the range between 1 and (2^{16} - 1) to produce a baud clock (BCLK). The frequency of BCLK is sixteen times (16×) the baud rate (each received or transmitted bit lasts 16 BCLK cycles) or thirteen times (13×) the baud rate (each received or transmitted bit lasts 13 BCLK cycles). When the UART is receiving, the bit is sampled in the 8th BCLK cycle for 16× over sampling mode and on the 6th BCLK cycle for 13× over-sampling mode. The 16× or 13× reference clock is selected by configuring the OSM_SEL bit in the mode definition register (MDR). The formula to calculate the divisor is:

$$\text{Divisor} = \frac{\text{UART input clock frequency}}{\text{Desired baud rate} \times 16} \quad [\text{MDR.OSM_SEL} = 0]$$

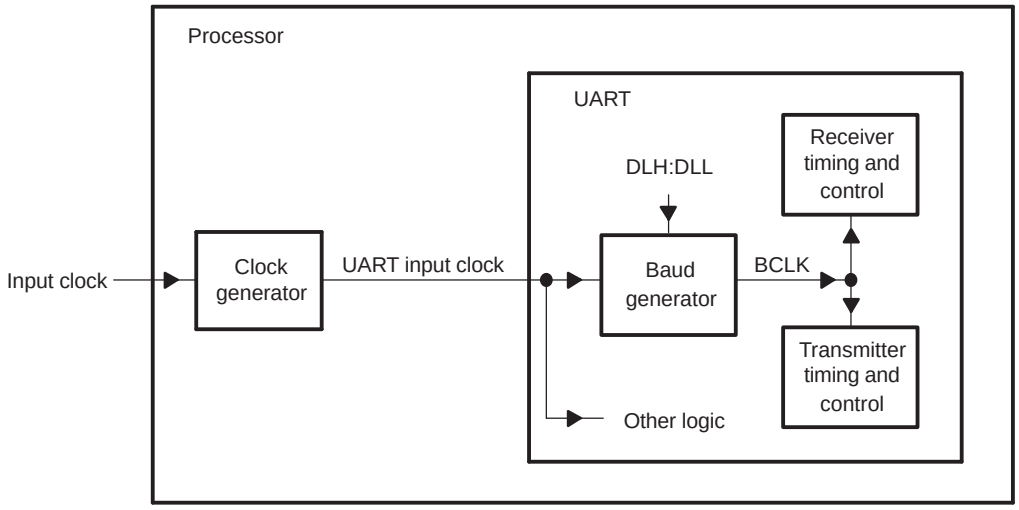
$$\text{Divisor} = \frac{\text{UART input clock frequency}}{\text{Desired baud rate} \times 13} \quad [\text{MDR.OSM_SEL} = 1]$$

Two 8-bit register fields (DLH and DLL), called divisor latches, hold this 16-bit divisor. DLH holds the most significant bits of the divisor, and DLL holds the least significant bits of the divisor. For information about these register fields, see Section 22.3. These divisor latches must be loaded during initialization of the UART in order to ensure desired operation of the baud generator. Writing to the divisor latches results in two wait states being inserted during the write access while the baud generator is loaded with the new value.

Figure 22-3 summarizes the relationship between the transferred data bit, BCLK, and the UART input clock. Note that the timing relationship depicted in Figure 22-3 shows that each bit lasts for 16 BCLK cycles . This is in case of 16x over-sampling mode. For 13× over-sampling mode each bit lasts for 13 BCLK cycles .

Example baud rates and divisor values relative to a 150-MHz UART input clock and 16× over-sampling mode are shown in Table 22-1.

Figure 22-2. UART Clock Generation Diagram



Copyright © 2013–2016, Texas Instruments Incorporated

Figure 22-3. Relationships Between Data Bit, BCLK, and UART Input Clock

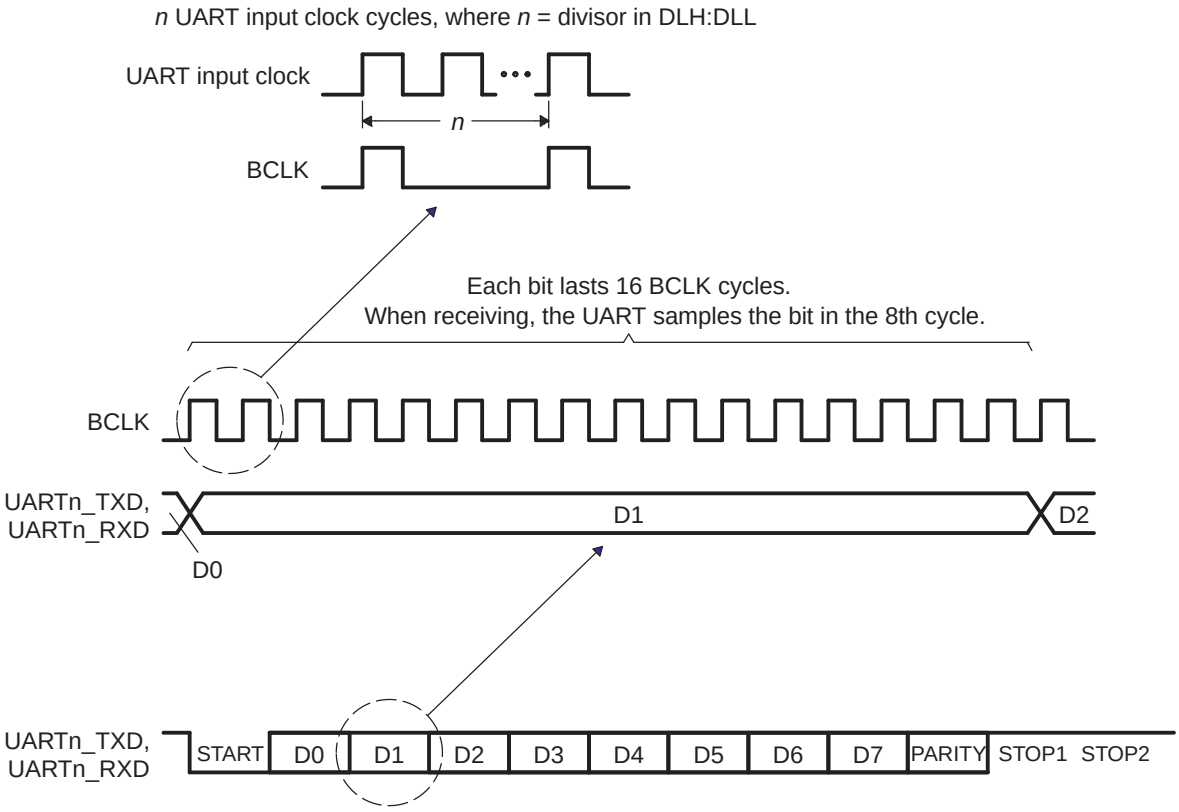


Table 22-1. Baud Rate Examples for 150-MHZ UART Input Clock and 16× Over-sampling Mode

Baud Rate	Divisor Value	Actual Baud Rate	Error (%)
2400	3906	2400.154	0.01
4800	1953	4800.372	0.01
9600	977	9595.701	-0.04
19200	488	19211.066	0.06
38400	244	38422.131	0.06
56000	167	56137.725	0.25
128000	73	129807.7	0.33
3000000	3	3125000	4.00

Table 22-2. Baud Rate Examples for 150-MHZ UART Input Clock and 13× Over-sampling Mode

Baud Rate	Divisor Value	Actual Baud Rate	Error (%)
2400	4808	2399	-0.01
4800	2404	4799.646	-0.01
9600	1202	9599.386	-0.01
19200	601	19198.771	-0.01
38400	300	38461.538	0.16
56000	206	56011.949	0.02
128000	90	128205.128	0.16
3000000	4	2884615.385	-4.00

Copyright © 2013–2016, Texas Instruments Incorporated

22.2.2 Signal Descriptions

The UARTs utilize a minimal number of signal connections to interface with external devices. The UART signal descriptions are included in Table 22-3. Note that the number of UARTs and their supported features vary on each device, see your device-specific data manual for more details.

Table 22-3. UART Signal Descriptions

Signal Name[1]	Signal Type	Function
UARTn_TXD	Output	Serial data transmit
UARTn_RXD	Input	Serial data receive
$\overline{\text{UARTn_CTS}}$ [2]	Input	Clear-to-Send handshaking signal
$\overline{\text{UARTn_RTS}}$ [2]	Output	Request-to-Send handshaking signal

(1) The value *n* indicates the applicable UART; that is, UART0, UART1, etc.
(2) This signal is not supported in all UARTs. See your device-specific data manual to check if it is supported.

22.2.3 Pin Multiplexing

Extensive pin multiplexing is used to accommodate the largest number of peripheral functions in the smallest possible package. Pin multiplexing is controlled using a combination of hardware configuration at device reset and software programmable register settings. See your device-specific data manual to determine how pin multiplexing affects the UART.

22.2.4 Protocol Description

22.2.4.1 Transmission

The UART transmitter section includes a transmitter hold register (THR) and a transmitter shift register (TSR). When the UART is in the FIFO mode, THR is a 16-byte FIFO. Transmitter section control is a function of the UART line control register (LCR). Based on the settings chosen in LCR, the UART transmitter sends the following to the receiving device:

- 1 START bit
- 5, 6, 7, or 8 data bits
- 1 PARITY bit (optional)
- 1, 1.5, or 2 STOP bits

22.2.4.2 Reception

The UART receiver section includes a receiver shift register (RSR) and a receiver buffer register (RBR). When the UART is in the FIFO mode, RBR is a 16-byte FIFO. Receiver section control is a function of the UART line control register (LCR). Based on the settings chosen in LCR, the UART receiver accepts the following from the transmitting device:

- 1 START bit
- 5, 6, 7, or 8 data bits
- 1 PARITY bit (optional)
- 1 STOP bit (any other STOP bits transferred with the above data are not detected)

Copyright © 2013–2016, Texas Instruments Incorporated

22.2.4.3 Data Format

The UART transmits in the following format:

1 START bit + data bits (5, 6, 7, 8) + 1 PARITY bit (optional) + STOP bit (1, 1.5, 2)

It transmits 1 START bit; 5, 6, 7, or 8 data bits, depending on the data width selection; 1 PARITY bit, if parity is selected; and 1, 1.5, or 2 STOP bits, depending on the STOP bit selection.

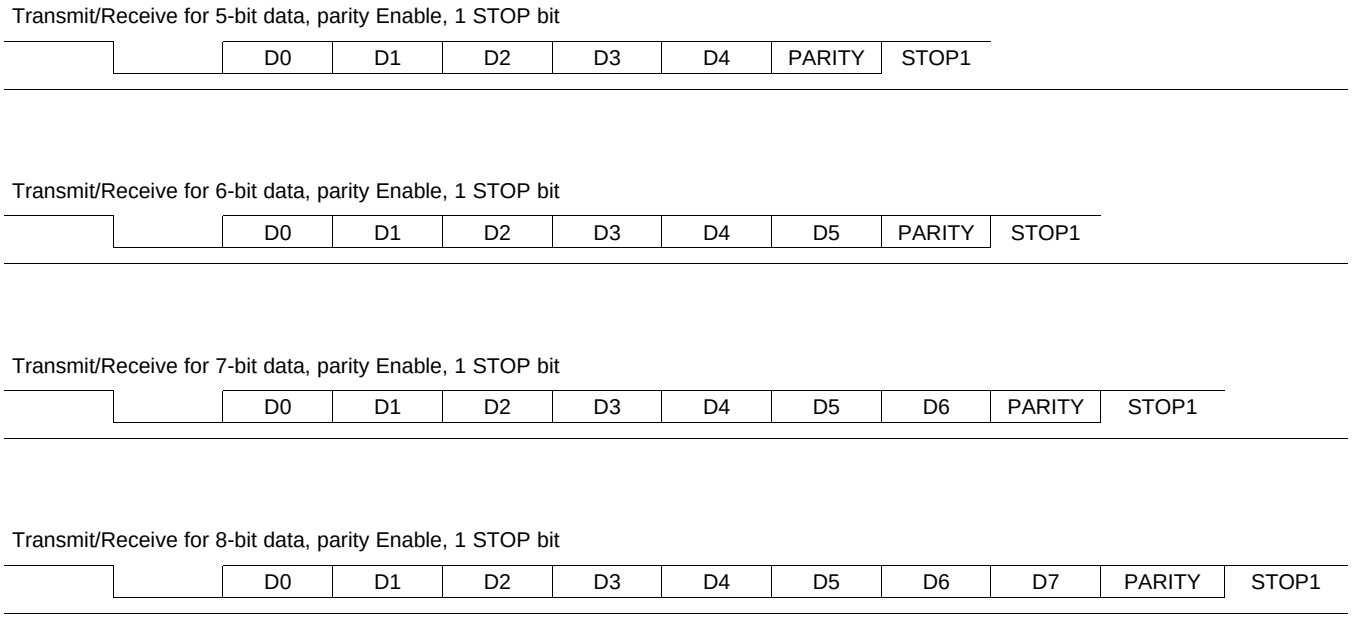
The UART receives in the following format:

1 START bit + data bits (5, 6, 7, 8) + 1 PARITY bit (optional) + 1 STOP bit

It receives 1 START bit; 5, 6, 7, or 8 data bits, depending on the data width selection; 1 PARITY bit, if parity is selected; and 1 STOP bit.

The protocol formats are shown in Figure 22-4.

Figure 22-4. UART Protocol Formats



Copyright © 2013–2016, Texas Instruments Incorporated

22.2.5 Operation

22.2.5.1 Transmission

The UART transmitter section includes a transmitter hold register (THR) and a transmitter shift register (TSR). When the UART is in the FIFO mode, THR is a 16-byte FIFO. Transmitter section control is a function of the UART line control register (LCR). Based on the settings chosen in LCR, the UART transmitter sends the following to the receiving device:

- 1 START bit
- 5, 6, 7, or 8 data bits
- 1 PARITY bit (optional)
- 1, 1.5, or 2 STOP bits

THR receives data from the internal data bus, and when TSR is ready, the UART moves the data from THR to TSR. The UART serializes the data in TSR and transmits the data on the UARTn_TXD pin.

In the non-FIFO mode, if THR is empty and the THR empty (THRE) interrupt is enabled in the interrupt enable register (IER), an interrupt is generated. This interrupt is cleared when a character is loaded into THR or the interrupt identification register (IIR) is read. In the FIFO mode, the interrupt is generated when the transmitter FIFO is empty, and it is cleared when at least one byte is loaded into the FIFO or IIR is read.

22.2.5.2 Reception

The UART receiver section includes a receiver shift register (RSR) and a receiver buffer register (RBR). When the UART is in the FIFO mode, RBR is a 16-byte FIFO. Timing is supplied by the 16× receiver clock. Receiver section control is a function of the UART line control register (LCR). Based on the settings chosen in LCR, the UART receiver accepts the following from the transmitting device:

- 1 START bit
- 5, 6, 7, or 8 data bits
- 1 PARITY bit (optional)
- 1 STOP bit (any other STOP bits transferred with the above data are not detected)

RSR receives the data bits from the UARTn_RXD pin. Then RSR concatenates the data bits and moves the resulting value into RBR (or the receiver FIFO). The UART also stores three bits of error status information next to each received character, to record a parity error, framing error, or break.

In the non-FIFO mode, when a character is placed in RBR and the receiver data-ready interrupt is enabled in the interrupt enable register (IER), an interrupt is generated. This interrupt is cleared when the character is read from RBR. In the FIFO mode, the interrupt is generated when the FIFO is filled to the trigger level selected in the FIFO control register (FCR), and it is cleared when the FIFO contents drop below the trigger level.

Copyright © 2013–2016, Texas Instruments Incorporated

22.2.5.3 FIFO Modes

The following two modes can be used for servicing the receiver and transmitter FIFOs:

- FIFO interrupt mode. The FIFO is enabled and the associated interrupts are enabled. Interrupts are sent to the CPU to indicate when specific events occur.
- FIFO poll mode. The FIFO is enabled but the associated interrupts are disabled. The CPU polls status bits to detect specific events.

Because the receiver FIFO and the transmitter FIFO are controlled separately, either one or both can be placed into the interrupt mode or the poll mode.

22.2.5.3.1 FIFO Interrupt Mode

When the receiver FIFO is enabled in the FIFO control register (FCR) and the receiver interrupts are enabled in the interrupt enable register (IER), the interrupt mode is selected for the receiver FIFO. The following are important points about the receiver interrupts:

- The receiver data-ready interrupt is issued to the CPU when the FIFO has reached the trigger level that is programmed in FCR. It is cleared when the CPU or the DMA controller reads enough characters from the FIFO such that the FIFO drops below its programmed trigger level.
- The receiver line status interrupt is generated in response to an overrun error, a parity error, a framing error, or a break. This interrupt has higher priority than the receiver data-ready interrupt. For details, see Section 22.2.8.
- The data-ready (DR) bit in the line status register (LSR) indicates the presence or absence of characters in the receiver FIFO. The DR bit is set when a character is transferred from the receiver shift register (RSR) to the empty receiver FIFO. The DR bit remains set until the FIFO is empty again.
- A receiver time-out interrupt occurs if all of the following conditions exist:
  - At least one character is in the FIFO,
  - The most recent character was received more than four continuous character times ago. A character time is the time allotted for 1 START bit, *n* data bits, 1 PARITY bit, and 1 STOP bit, where *n* depends on the word length selected with the WLS bits in the line control register (LCR). See Table 22-4.
  - The most recent read of the FIFO has occurred more than four continuous character times before.
- Character times are calculated by using the baud rate.
- When a receiver time-out interrupt has occurred, it is cleared and the time-out timer is cleared when the CPU or the EDMA controller reads one character from the receiver FIFO. The interrupt is also cleared if a new character is received in the FIFO or if the URRST bit is cleared in the power and emulation management register (PWREMU_MGMT).
- If a receiver time-out interrupt has not occurred, the time-out timer is cleared after a new character is received or after the CPU or EDMA reads the receiver FIFO.

When the transmitter FIFO is enabled in FCR and the transmitter holding register empty (THRE) interrupt is enabled in IER, the interrupt mode is selected for the transmitter FIFO. The THRE interrupt occurs when the transmitter FIFO is empty. It is cleared when the transmitter hold register (THR) is loaded (1 to 16 characters may be written to the transmitter FIFO while servicing this interrupt) or the interrupt identification register (IIR) is read.

Table 22-4. Character Time for Word Lengths

Word Length (*n*)	Character Time	Four Character Times
5	Time for 8 bits	Time for 32 bits
6	Time for 9 bits	Time for 36 bits
7	Time for 10 bits	Time for 40 bits
8	Time for 11 bits	Time for 44 bits

Copyright © 2013–2016, Texas Instruments Incorporated

22.2.5.3.2 FIFO Poll Mode

When the receiver FIFO is enabled in the FIFO control register (FCR) and the receiver interrupts are disabled in the interrupt enable register (IER), the poll mode is selected for the receiver FIFO. Similarly, when the transmitter FIFO is enabled and the transmitter interrupts are disabled, the transmitted FIFO is in the poll mode. In the poll mode, the CPU detects events by checking bits in the line status register (LSR):

- The RXFIFOE bit indicates whether there are any errors in the receiver FIFO.
- The TEMT bit indicates that both the transmitter holding register (THR) and the transmitter shift register (TSR) are empty.
- The THRE bit indicates when THR is empty.
- The BI (break), FE (framing error), PE (parity error), and OE (overrun error) bits specify which error or errors have occurred.
- The DR (data-ready) bit is set as long as there is at least one byte in the receiver FIFO.

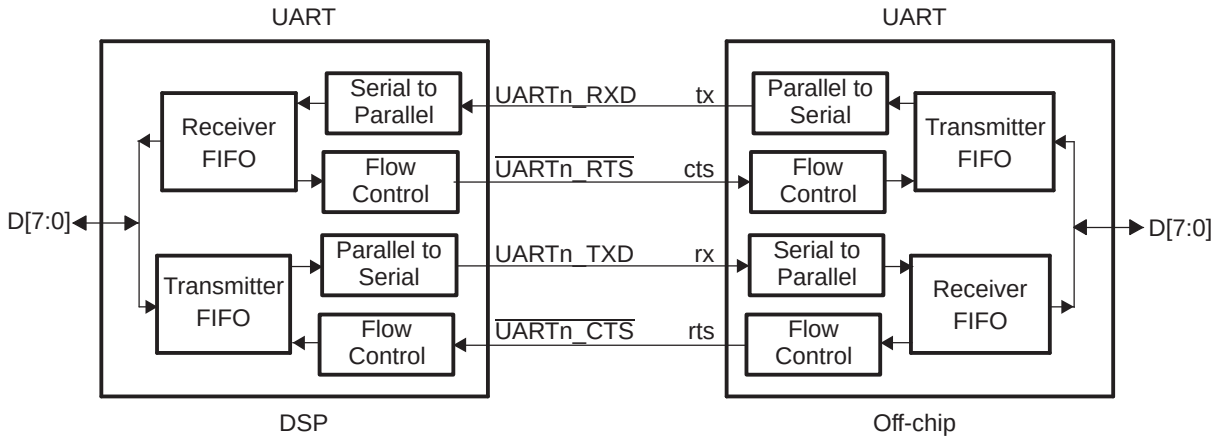
Also, in the FIFO poll mode:

- The interrupt identification register (IIR) is not affected by any events because the interrupts are disabled.
- The UART does not indicate when the receiver FIFO trigger level is reached or when a receiver time-out occurs.

22.2.5.4 Autoflow Control

The UART can employ autoflow control by connecting the $\overline{\text{UARTn_CTS}}$ and $\overline{\text{UARTn_RTS}}$ signals. Note that all UARTs do not support autoflow control, see your device-specific data manual for supported features. The $\overline{\text{UARTn_CTS}}$ input must be active before the transmitter FIFO can transmit data. The $\overline{\text{UARTn_RTS}}$ becomes active when the receiver needs more data and notifies the sending device. When $\overline{\text{UARTn_RTS}}$ is connected to $\overline{\text{UARTn_CTS}}$, data transmission does not occur unless the receiver FIFO has space for the data. Therefore, when two UARTs are connected as shown in Figure 22-5 with autoflow enabled, overrun errors are eliminated.

Figure 22-5. UART Interface Using Autoflow Diagram

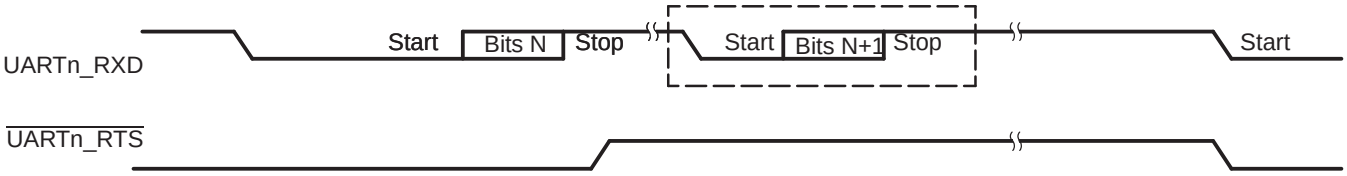


Copyright © 2013–2016, Texas Instruments Incorporated

22.2.5.4.1 UARTn_RTS Behavior

UARTn_RTS data flow control originates in the receiver block (see Figure 22-1). When the receiver FIFO level reaches a trigger level of 1, 4, 8, or 14 (see Figure 22-6), UARTn_RTS is deasserted. The sending UART may send an additional byte after the trigger level is reached (assuming the sending UART has another byte to send), because it may not recognize the deassertion of UARTn_RTS until after it has begun sending the additional byte. For trigger level 1, 4, and 8, UARTn_RTS is automatically reasserted once the receiver FIFO is emptied. For trigger level 14, UARTn_RTS is automatically reasserted once the receiver FIFO drops below the trigger level.

Figure 22-6. Autoflow Functional Timing Waveforms for UARTn_RTS



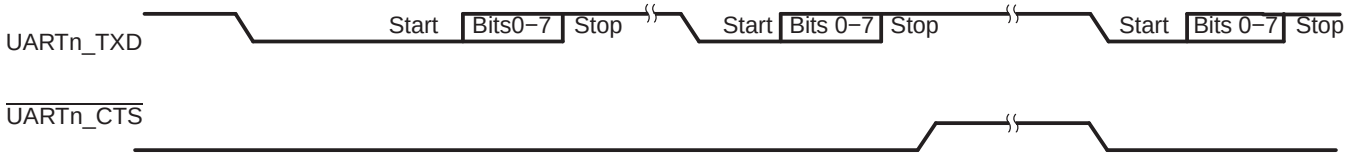
(1) N = Receiver FIFO trigger level.

(2) The two blocks in dashed lines cover the case where an additional byte is sent.

22.2.5.4.2 UARTn_CTS Behavior

The transmitter checks UARTn_CTS before sending the next data byte. If UARTn_CTS is active, the transmitter sends the next byte. To stop the transmitter from sending the following byte, UARTn_CTS must be released before the middle of the last STOP bit that is currently being sent (see Figure 22-7). When flow control is enabled, UARTn_CTS level changes do not trigger interrupts because the device automatically controls its own transmitter. Without autoflow control, the transmitter sends any data present in the transmitter FIFO and a receiver overrun error may result.

Figure 22-7. Autoflow Functional Timing Waveforms for UARTn_CTS



(1) When UARTn_CTS is active (low), the transmitter keeps sending serial data out.

(2) When UARTn_CTS goes high before the middle of the last STOP bit of the current byte, the transmitter finishes sending the current byte but it does not send the next byte.

(3) When UARTn_CTS goes from high to low, the transmitter begins sending data again.

22.2.5.5 Loopback Control

The UART can be placed in the diagnostic mode using the LOOP bit in the modem control register (MCR), which internally connects the UART output back to the UART input. In this mode, the transmit and receive data paths, the transmitter and receiver interrupts, and the modem control interrupts can be verified without connecting to another UART.

Copyright © 2013–2016, Texas Instruments Incorporated

22.2.6 Reset Considerations

22.2.6.1 Software Reset Considerations

Two bits in the power and emulation management register (PWREMU_MGMT) control resetting the parts of the UART:

- The UTRST bit controls resetting the transmitter only. If UTRST = 1, the transmitter is active; if UTRST = 0, the transmitter is in reset.
- The URRST bit controls resetting the receiver only. If URRST = 1, the receiver is active; if URRST = 0, the receiver is in reset.

In each case, putting the receiver and/or transmitter in reset will reset the state machine of the affected portion but does not affect the UART registers.

22.2.6.2 Hardware Reset Considerations

When the processor RESET pin is asserted, the entire processor is reset and is held in the reset state until the RESET pin is released. As part of a device reset, the UART state machine is reset and the UART registers are forced to their default states. The default states of the registers are shown in Section 22.3.

22.2.7 Initialization

The following steps are required to initialize the UART:

1. Perform the necessary device pin multiplexing setup (see your device-specific data manual).
2. Set the desired baud rate by writing the appropriate clock divisor values to the divisor latch registers (DLL and DLH).
3. If the FIFOs will be used, select the desired trigger level and enable the FIFOs by writing the appropriate values to the FIFO control register (FCR). The FIFOEN bit in FCR must be set first, before the other bits in FCR are configured.
4. Choose the desired protocol settings by writing the appropriate values to the line control register (LCR).
5. If autoflow control is desired, write appropriate values to the modem control register (MCR). Note that all UARTs do not support autoflow control, see your device-specific data manual for supported features.
6. Choose the desired response to emulation suspend events by configuring the FREE bit and enable the UART by setting the UTRST and URRST bits in the power and emulation management register (PWREMU_MGMT).

22.2.8 Interrupt Support

22.2.8.1 Interrupt Events and Requests

The UART generates the interrupt requests described in Table 22-5. All requests are multiplexed through an arbiter to a single UART interrupt request to the CPU, as shown in Figure 22-8. Each of the interrupt requests has an enable bit in the interrupt enable register (IER) and is recorded in the interrupt identification register (IIR).

If an interrupt occurs and the corresponding enable bit is set to 1, the interrupt request is recorded in IIR and is forwarded to the CPU. If an interrupt occurs and the corresponding enable bit is cleared to 0, the interrupt request is blocked. The interrupt request is neither recorded in IIR nor forwarded to the CPU.

22.2.8.2 Interrupt Multiplexing

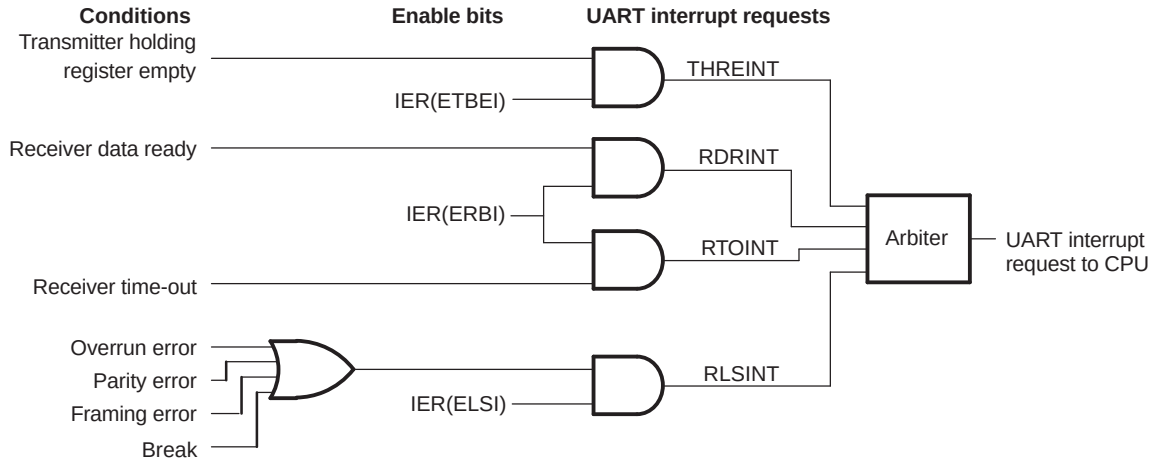
The UARTs have dedicated interrupt signals to the CPU and the interrupts are not multiplexed with any other interrupt source.

Copyright © 2013–2016, Texas Instruments Incorporated

Table 22-5. UART Interrupt Requests Descriptions

UART Interrupt Request	Interrupt Source	Comment
THREINT	THR-empty condition: The transmitter holding register (THR) or the transmitter FIFO is empty. All of the data has been copied from THR to the transmitter shift register (TSR).	If THREINT is enabled in IER, by setting the ETBEI bit, it is recorded in IIR. As an alternative to using THREINT, the CPU can poll the THRE bit in the line status register (LSR).
RDAINT	Receive data available in non-FIFO mode or trigger level reached in the FIFO mode.	If RDAINT is enabled in IER, by setting the ERBI bit, it is recorded in IIR. As an alternative to using RDAINT, the CPU can poll the DR bit in the line status register (LSR). In the FIFO mode, this is not a functionally equivalent alternative because the DR bit does not respond to the FIFO trigger level. The DR bit only indicates the presence or absence of unread characters.
RTOINT	Receiver time-out condition (in the FIFO mode only): No characters have been removed from or input to the receiver FIFO during the last four character times (see Table 22-4), and there is at least one character in the receiver FIFO during this time.	The receiver time-out interrupt prevents the UART from waiting indefinitely, in the case when the receiver FIFO level is below the trigger level and thus does not generate a receiver data-ready interrupt. If RTOINT is enabled in IER, by setting the ERBI bit, it is recorded in IIR. There is no status bit to reflect the occurrence of a time-out condition.
RLSINT	Receiver line status condition: An overrun error, parity error, framing error, or break has occurred.	If RLSINT is enabled in IER, by setting the ELSI bit, it is recorded in IIR. As an alternative to using RLSINT, the CPU can poll the following bits in the line status register (LSR): overrun error indicator (OE), parity error indicator (PE), framing error indicator (FE), and break indicator (BI).

Figure 22-8. UART Interrupt Request Enable Paths



Copyright © 2013–2016, Texas Instruments Incorporated

22.2.9 DMA Event Support

In the FIFO mode, the UART generates the following two DMA events:

- **Receive event (URXEVT):** The trigger level for the receiver FIFO (1, 4, 8, or 14 characters) is set with the RXFIFTL bit in the FIFO control register (FCR). Every time the trigger level is reached or a receiver time-out occurs, the UART sends a receive event to the EDMA controller. In response, the EDMA controller reads the data from the receiver FIFO by way of the receiver buffer register (RBR). Note that the receive event is not asserted if the data at the top of the receiver FIFO is erroneous even if the trigger level has been reached.
- **Transmit event (UTXEVT):** When the transmitter FIFO is empty (when the last byte in the transmitter FIFO has been copied to the transmitter shift register), the UART sends an UTXEVT signal to the EDMA controller. In response, the EDMA controller refills the transmitter FIFO by way of the transmitter holding register (THR). The UTXEVT signal is also sent to the DMA controller when the UART is taken out of reset using the UTRST bit in the power and emulation management register (PWREMU_MGMT).

Activity in DMA channels can be synchronized to these events. In the non-FIFO mode, the UART generates no DMA events. Any DMA channel synchronized to either of these events must be enabled at the time the UART event is generated. Otherwise, the DMA channel will miss the event and, unless the UART generates a new event, no data transfer will occur.

22.2.10 Power Management

The UART peripheral can be placed in reduced-power modes to conserve power during periods of low activity. The power management of the UART peripheral is controlled by the processor Power and Sleep Controller (PSC). The PSC acts as a master controller for power management for all of the peripherals on the device. For detailed information on power management procedures using the PSC, see the *Power and Sleep Controller (PSC)* chapter.

22.2.11 Emulation Considerations

The FREE bit in the power and emulation management register (PWREMU_MGMT) determines how the UART responds to an emulation suspend event such as an emulator halt or breakpoint. If FREE = 0 and a transmission is in progress, the UART halts after completing the one-word transmission; if FREE = 0 and a transmission is not in progress, the UART halts immediately. If FREE = 1, the UART does not halt and continues operating normally.

Note also that most emulator accesses are transparent to UART operation. Emulator read operations do not affect any register contents, status bits, or operating states, with the exception of the interrupt identification register (IIR). Emulator writes, however, may affect register contents and may affect UART operation, depending on what register is accessed and what value is written.

The UART registers can be read from or written to during emulation suspend events, even if the UART activity has stopped.

22.2.12 Exception Processing

22.2.12.1 Divisor Latch Not Programmed

Since the processor reset signal has no effect on the divisor latch, the divisor latch will have an unknown value after power up. If the divisor latch is not programmed after power up, the baud clock (BCLK) will not operate and will instead be set to a constant logic 1 state.

The divisor latch values should always be reinitialized following a processor reset.

22.2.12.2 Changing Operating Mode During Busy Serial Communication

Since the serial link characteristics are based on how the control registers are programmed, the UART will expect the control registers to be static while it is busy engaging in a serial communication. Therefore, changing the control registers while the module is still busy communicating with another serial device will most likely cause an error condition and should be avoided.

Copyright © 2013–2016, Texas Instruments Incorporated

22.3 Registers

The system programmer has access to and control over any of the UART registers that are listed in Table 22-6. These registers, which control UART operations, receive data, and transmit data, are available at 32-bit addresses in the device memory map. See your device-specific data manual for the memory address of these registers.

- RBR, THR, and DLL share one address. When the DLAB bit in LCR is 0, reading from the address gives the content of RBR, and writing to the address modifies THR. When DLAB = 1, all accesses at the address read or modify DLL. DLL can also be accessed with address offset 20h.
- IER and DLH share one address. When DLAB = 0, all accesses read or modify IER. When DLAB = 1, all accesses read or modify DLH. DLH can also be accessed with address offset 24h.
- IIR and FCR share one address. Regardless of the value of the DLAB bit, reading from the address gives the content of IIR, and writing modifies FCR.

Table 22-6. UART Registers

Offset	Acronym	Register Description	Section
0h	RBR	Receiver Buffer Register (read only)	Section 22.3.1
0h	THR	Transmitter Holding Register (write only)	Section 22.3.2
4h	IER	Interrupt Enable Register	Section 22.3.3
8h	IIR	Interrupt Identification Register (read only)	Section 22.3.4
8h	FCR	FIFO Control Register (write only)	Section 22.3.5
Ch	LCR	Line Control Register	Section 22.3.6
10h	MCR	Modem Control Register	Section 22.3.7
14h	LSR	Line Status Register	Section 22.3.8
18h	MSR	Modem Status Register	Section 22.3.9
1Ch	SCR	Scratch Pad Register	Section 22.3.10
20h	DLL	Divisor LSB Latch	Section 22.3.11
24h	DLH	Divisor MSB Latch	Section 22.3.11
28h	REVID1	Revision Identification Register 1	Section 22.3.12
2Ch	REVID2	Revision Identification Register 2	Section 22.3.12
30h	PWREMU_MGMT	Power and Emulation Management Register	Section 22.3.13
34h	MDR	Mode Definition Register	Section 22.3.14

Copyright © 2013–2016, Texas Instruments Incorporated

22.3.1 Receiver Buffer Register (RBR)

The receiver buffer register (RBR) is shown in Figure 22-9 and described in Table 22-7.

The UART receiver section consists of a receiver shift register (RSR) and a receiver buffer register (RBR). When the UART is in the FIFO mode, RBR is a 16-byte FIFO. Timing is supplied by the 16x receiver clock or 13x receiver clock by programming OSM_SEL bit field of MDR register. Receiver section control is a function of the line control register (LCR).

RSR receives serial data from the UARTn_RXD pin. Then RSR concatenates the data and moves it into RBR (or the receiver FIFO). In the non-FIFO mode, when a character is placed in RBR and the receiver data-ready interrupt is enabled (DR = 1 in IER), an interrupt is generated. This interrupt is cleared when the character is read from RBR. In the FIFO mode, the interrupt is generated when the FIFO is filled to the trigger level selected in the FIFO control register (FCR), and it is cleared when the FIFO contents drop below the trigger level.

Access considerations:

RBR, THR, and DLL share one address. To read RBR, write 0 to the DLAB bit in LCR, and read from the shared address. When DLAB = 0, writing to the shared address modifies THR. When DLAB = 1, all accesses at the shared address read or modify DLL.

DLL also has a dedicated address. If you use the dedicated address, you can keep DLAB = 0, so that RBR and THR are always selected at the shared address.

Figure 22-9. Receiver Buffer Register (RBR)

31	16
Reserved	
R-0	

15	8	7	0
Reserved		DATA	
R-0		R-0	

LEGEND: R = Read only; -*n* = value after reset

Table 22-7. Receiver Buffer Register (RBR) Field Descriptions

Bit	Field	Value	Description
31-8	Reserved	0	Reserved
7-0	DATA	0-FFh	Received data

Copyright © 2013–2016, Texas Instruments Incorporated

22.3.2 Transmitter Holding Register (THR)

The transmitter holding register (THR) is shown in Figure 22-10 and described in Table 22-8.

The UART transmitter section consists of a transmitter hold register (THR) and a transmitter shift register (TSR). When the UART is in the FIFO mode, THR is a 16-byte FIFO. Transmitter section control is a function of the line control register (LCR).

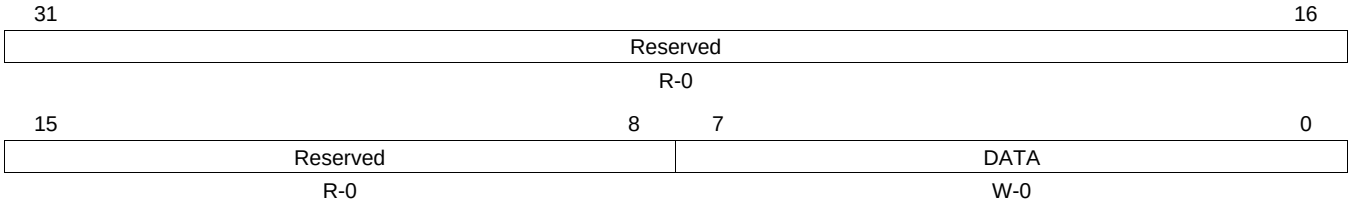
THR receives data from the internal data bus and when TSR is idle, the UART moves the data from THR to TSR. The UART serializes the data in TSR and transmits the data on the TX pin. In the non-FIFO mode, if THR is empty and the THR empty (THRE) interrupt is enabled (ETBEI = 1 in IER), an interrupt is generated. This interrupt is cleared when a character is loaded into THR or the interrupt identification register (IIR) is read. In the FIFO mode, the interrupt is generated when the transmitter FIFO is empty, and it is cleared when at least one byte is loaded into the FIFO or IIR is read.

Access considerations:

RBR, THR, and DLL share one address. To load THR, write 0 to the DLAB bit of LCR, and write to the shared address. When DLAB = 0, reading from the shared address gives the content of RBR. When DLAB = 1, all accesses at the address read or modify DLL.

DLL also has a dedicated address. If you use the dedicated address, you can keep DLAB = 0, so that RBR and THR are always selected at the shared address.

Figure 22-10. Transmitter Holding Register (THR)



LEGEND: R = Read only; W = Write only; -*n* = value after reset

Table 22-8. Transmitter Holding Register (THR) Field Descriptions

Bit	Field	Value	Description
31-8	Reserved	0	Reserved
7-0	DATA	0-FFh	Data to transmit

Copyright © 2013–2016, Texas Instruments Incorporated

22.3.3 Interrupt Enable Register (IER)

The interrupt enable register (IER) is used to individually enable or disable each type of interrupt request that can be generated by the UART. Each interrupt request that is enabled in IER is forwarded to the CPU. IER is shown in Figure 22-11 and described in Table 22-9.

Access considerations:

IER and DLH share one address. To read or modify IER, write 0 to the DLAB bit in LCR. When DLAB = 1, all accesses at the shared address read or modify DLH.

DLH also has a dedicated address. If you use the dedicated address, you can keep DLAB = 0, so that IER is always selected at the shared address.

Figure 22-11. Interrupt Enable Register (IER)

31	16
Reserved	
R-0	

15	4	3	2	1	0
Reserved		Rsvd	ELSI	ETBEI	ERBI
R-0		R/W-0	R/W-0	R/W-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 22-9. Interrupt Enable Register (IER) Field Descriptions

Bit	Field	Value	Description
31-4	Reserved	0	Reserved
3	EDSSI	0	Enable Modem Status Interrupt
2	ELSI		Receiver line status interrupt enable.
		0	Receiver line status interrupt is disabled.
		1	Receiver line status interrupt is enabled.
1	ETBEI		Transmitter holding register empty interrupt enable.
		0	Transmitter holding register empty interrupt is disabled.
		1	Transmitter holding register empty interrupt is enabled.
0	ERBI		Receiver data available interrupt and character timeout indication interrupt enable.
		0	Receiver data available interrupt and character timeout indication interrupt is disabled.
		1	Receiver data available interrupt and character timeout indication interrupt is enabled.

Copyright © 2013–2016, Texas Instruments Incorporated

22.3.4 Interrupt Identification Register (IIR)

The interrupt identification register (IIR) is a read-only register at the same address as the FIFO control register (FCR), which is a write-only register. When an interrupt is generated and enabled in the interrupt enable register (IER), IIR indicates that an interrupt is pending in the IPEND bit and encodes the type of interrupt in the INTID bits. Reading IIR clears any THR empty (THRE) interrupts that are pending.

IIR is shown in Figure 22-12 and described in Figure 22-12.

The UART has an on-chip interrupt generation and prioritization capability that permits flexible communication with the CPU. The UART provides three priority levels of interrupts:

- Priority 1 - Receiver line status (highest priority)
- Priority 2 - Receiver data ready or receiver timeout
- Priority 3 - Transmitter holding register empty

The FIFOEN bit in IIR can be checked to determine whether the UART is in the FIFO mode or the non-FIFO mode.

Access consideration:

IIR and FCR share one address. Regardless of the value of the DLAB bit in LCR, reading from the address gives the content of IIR, and writing to the address modifies FCR.

Figure 22-12. Interrupt Identification Register (IIR)

31	16
Reserved	
R-0	

15	8	7	6	5	4	3	1	0
Reserved		FIFOEN		Reserved		INTID		IPEND
R-0		R-0		R-0		R-0		R-1

LEGEND: R = Read only; -*n* = value after reset

Table 22-10. Interrupt Identification Register (IIR) Field Descriptions

Bit	Field	Value	Description
31-8	Reserved	0	Reserved
7-6	FIFOEN	0-3h	FIFOs enabled.
		0	Non-FIFO mode
		1h-2h	Reserved
		3h	FIFOs are enabled. FIFOEN bit in the FIFO control register (FCR) is set to 1.
5-4	Reserved	0	Reserved
3-1	INTID	0-7h	Interrupt type. See Table 22-11.
		0	Reserved
		1h	Transmitter holding register empty (priority 3)
		2h	Receiver data available (priority 2)
		3h	Receiver line status (priority 1, highest)
		4h-5h	Reserved
		6h	Character timeout indication (priority 2)
		7h	Reserved
0	IPEND		Interrupt pending. When any UART interrupt is generated and is enabled in IER, IPEND is forced to 0. IPEND remains 0 until all pending interrupts are cleared or until a hardware reset occurs. If no interrupts are enabled, IPEND is never forced to 0.
		0	Interrupts pending.
		1	No interrupts pending.

Copyright © 2013–2016, Texas Instruments Incorporated

Table 22-11. Interrupt Identification and Interrupt Clearing Information

Priority Level	IIR Bits 3	2	1	0	Interrupt Type	Interrupt Source	Event That Clears Interrupt
None	0	0	0	1	None	None	None
1	0	1	1	0	Receiver line status	Overrun error, parity error, framing error, or break is detected.	For an overrun error, reading the line status register (LSR) clears the interrupt. For a parity error, framing error, or break, the interrupt is cleared only after all the erroneous data have been read.
2	0	1	0	0	Receiver data-ready	Non-FIFO mode: Receiver data is ready.	Non-FIFO mode: The receiver buffer register (RBR) is read.
						FIFO mode: Trigger level reached. If four character times (see Table 22-4) pass with no access of the FIFO, the interrupt is asserted again.	FIFO mode: The FIFO drops below the trigger level.(1)
2	1	1	0	0	Receiver time-out	FIFO mode only: No characters have been removed from or input to the receiver FIFO during the last four character times (see Table 22-4), and there is at least one character in the receiver FIFO during this time.	One of the following events: • A character is read from the receiver FIFO.(1) • A new character arrives in the receiver FIFO. • The URRST bit in the power and emulation management register (PWREMU_MGMT) is loaded with 0.
3	0	0	1	0	Transmitter holding register empty	Non-FIFO mode: Transmitter holding register (THR) is empty. FIFO mode: Transmitter FIFO is empty.	A character is written to the transmitter holding register (THR) or the interrupt identification register (IIR) is read.

(1) In the FIFO mode, the receiver data-ready interrupt or receiver time-out interrupt is cleared by the CPU or by the DMA controller, whichever reads from the receiver FIFO first.

22.3.5 FIFO Control Register (FCR)

The FIFO control register (FCR) is a write-only register at the same address as the interrupt identification register (IIR), which is a read-only register. Use FCR to enable and clear the FIFOs and to select the receiver FIFO trigger level FCR is shown in Figure 22-13 and described in Table 22-12. The FIFOEN bit must be set to 1 before other FCR bits are written to or the FCR bits are not programmed.

Access consideration:

IIR and FCR share one address. Regardless of the value of the DLAB bit, reading from the address gives the content of IIR, and writing to the address modifies FCR.

CAUTION

For proper communication between the UART and the EDMA controller, the DMAMODE1 bit must be set to 1. Always write a 1 to the DMAMODE1 bit, and after a hardware reset, change the DMAMODE1 bit from 0 to 1.

Copyright © 2013–2016, Texas Instruments Incorporated

Figure 22-13. FIFO Control Register (FCR)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
R-0															

15	14	13	12	11	10	9	8
Reserved							
R-0							

7	6	5	4	3	2	1	0
RXFIFTL		Reserved		DMAMODE1[(1)]	TXCLR	RXCLR	FIFOEN
W-0		R-0		W-0	W1C-0	W1C-0	W-0

LEGEND: R = Read only; W = Write only; W1C = Write 1 to clear (writing 0 has no effect); -*n* = value after reset

(1) Always write 1 to the DMAMODE1 bit. After a hardware reset, change the DMAMODE1 bit from 0 to 1. DMAMODE = 1 is required for proper communication between the UART and the DMA controller.

Table 22-12. FIFO Control Register (FCR) Field Descriptions

Bit	Field	Value	Description
31-8	Reserved	0	Reserved
7-6	RXFIFTL	0-3h	Receiver FIFO trigger level. RXFIFTL sets the trigger level for the receiver FIFO. When the trigger level is reached, a receiver data-ready interrupt is generated (if the interrupt request is enabled). Once the FIFO drops below the trigger level, the interrupt is cleared.
		0	1 byte
		1h	4 bytes
		2h	8 bytes
		3h	14 bytes
5-4	Reserved	0	Reserved
3	DMAMODE1		DMA MODE1 enable if FIFOs are enabled. Always write 1 to DMAMODE1. After a hardware reset, change DMAMODE1 from 0 to 1. DMAMOD1 = 1 is a requirement for proper communication between the UART and the EDMA controller.
		0	DMA MODE1 is disabled.
		1	DMA MODE1 is enabled.
2	TXCLR		Transmitter FIFO clear. Write a 1 to TXCLR to clear the bit.
		0	No effect.
		1	Clears transmitter FIFO and resets the transmitter FIFO counter. The shift register is not cleared.
1	RXCLR		Receiver FIFO clear. Write a 1 to RXCLR to clear the bit.
		0	No effect.
		1	Clears receiver FIFO and resets the receiver FIFO counter. The shift register is not cleared.
0	FIFOEN		Transmitter and receiver FIFOs mode enable. FIFOEN must be set before other FCR bits are written to or the FCR bits are not programmed. Clearing this bit clears the FIFO counters.
		0	Non-FIFO mode. The transmitter and receiver FIFOs are disabled, and the FIFO pointers are cleared.
		1	FIFO mode. The transmitter and receiver FIFOs are enabled.

Copyright © 2013–2016, Texas Instruments Incorporated

22.3.6 Line Control Register (LCR)

The line control register (LCR) is shown in Figure 22-14 and described in Table 22-13.

The system programmer controls the format of the asynchronous data communication exchange by using LCR. In addition, the programmer can retrieve, inspect, and modify the content of LCR; this eliminates the need for separate storage of the line characteristics in system memory.

Figure 22-14. Line Control Register (LCR)

31	16
Reserved	
R-0	

15	8	7	6	5	4	3	2	1	0
Reserved		DLAB	BC	SP	EPS	PEN	STB	WLS	
R-0		R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 22-13. Line Control Register (LCR) Field Descriptions

Bit	Field	Value	Description
31-8	Reserved	0	Reserved
7	DLAB		Divisor latch access bit. The divisor latch registers (DLL and DLH) can be accessed at dedicated addresses or at addresses shared by RBR, THR, and IER. Using the shared addresses requires toggling DLAB to change which registers are selected. If you use the dedicated addresses, you can keep DLAB = 0.
		0	Allows access to the receiver buffer register (RBR), the transmitter holding register (THR), and the interrupt enable register (IER) selected. At the address shared by RBR, THR, and DLL, the CPU can read from RBR and write to THR. At the address shared by IER and DLH, the CPU can read from and write to IER.
		1	Allows access to the divisor latches of the baud generator during a read or write operation (DLL and DLH). At the address shared by RBR, THR, and DLL, the CPU can read from and write to DLL. At the address shared by IER and DLH, the CPU can read from and write to DLH.
6	BC		Break control.
		0	Break condition is disabled.
		1	Break condition is transmitted to the receiving UART. A break condition is a condition where the UARTn_TXD signal is forced to the spacing (cleared) state.
5	SP		Stick parity. The SP bit works in conjunction with the EPS and PEN bits. The relationship between the SP, EPS, and PEN bits is summarized in Table 22-14.
		0	Stick parity is disabled.
		1	Stick parity is enabled. • When odd parity is selected (EPS = 0), the PARITY bit is transmitted and checked as set. • When even parity is selected (EPS = 1), the PARITY bit is transmitted and checked as cleared.
4	EPS		Even parity select. Selects the parity when parity is enabled (PEN = 1). The EPS bit works in conjunction with the SP and PEN bits. The relationship between the SP, EPS, and PEN bits is summarized in Table 22-14.
		0	Odd parity is selected (an odd number of logic 1s is transmitted or checked in the data and PARITY bits).
		1	Even parity is selected (an even number of logic 1s is transmitted or checked in the data and PARITY bits).
3	PEN		Parity enable. The PEN bit works in conjunction with the SP and EPS bits. The relationship between the SP, EPS, and PEN bits is summarized in Table 22-14.
		0	No PARITY bit is transmitted or checked.
		1	Parity bit is generated in transmitted data and is checked in received data between the last data word bit and the first STOP bit.

Copyright © 2013–2016, Texas Instruments Incorporated

Table 22-13. Line Control Register (LCR) Field Descriptions (continued)

Bit	Field	Value	Description
2	STB		Number of STOP bits generated. STB specifies 1, 1.5, or 2 STOP bits in each transmitted character. When STB = 1, the WLS bit determines the number of STOP bits. The receiver clocks only the first STOP bit, regardless of the number of STOP bits selected. The number of STOP bits generated is summarized in Table 22-15.
		0	1 STOP bit is generated.
		1	WLS bit determines the number of STOP bits: • When WLS = 0, 1.5 STOP bits are generated. • When WLS = 1h, 2h, or 3h, 2 STOP bits are generated.
1-0	WLS	0-3h	Word length select. Number of bits in each transmitted or received serial character. When STB = 1, the WLS bit determines the number of STOP bits.
		0	5 bits
		1h	6 bits
		2h	7 bits
		3h	8 bits

Table 22-14. Relationship Between ST, EPS, and PEN Bits in LCR

ST Bit	EPS Bit	PEN Bit	Parity Option
x	x	0	Parity disabled: No PARITY bit is transmitted or checked
0	0	1	Odd parity selected: Odd number of logic 1s
0	1	1	Even parity selected: Even number of logic 1s
1	0	1	Stick parity selected with PARITY bit transmitted and checked as set
1	1	1	Stick parity selected with PARITY bit transmitted and checked as cleared

Table 22-15. Number of STOP Bits Generated

STB Bit	WLS Bits	Word Length Selected with WLS Bits	Number of STOP Bits Generated	Baud Clock (BCLK) Cycles
0	x	Any word length	1	16
1	0h	5 bits	1.5	24
1	1h	6 bits	2	32
1	2h	7 bits	2	32
1	3h	8 bits	2	32

Copyright © 2013–2016, Texas Instruments Incorporated

22.3.7 Modem Control Register (MCR)

The modem control register (MCR) is shown in Figure 22-15 and described in Table 22-16. The modem control register provides the ability to enable/disable the autoflow functions, and enable/disable the loopback function for diagnostic purposes.

Figure 22-15. Modem Control Register (MCR)

31							16
Reserved							
R-0							
15	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved		AFE[1]	LOOP	OUT2	OUT1	RTS[1]	Rsvd
R-0		R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

(1) All UARTs do not support this feature, see your device-specific data manual for supported features. If this feature is not available, this bit is reserved and should be cleared to 0.

Table 22-16. Modem Control Register (MCR) Field Descriptions

Bit	Field	Value	Description
31-6	Reserved	0	Reserved
5	AFE		Autoflow control enable. Autoflow control allows the $\overline{\text{UARTn_RTS}}$ and $\overline{\text{UARTn_CTS}}$ signals to provide handshaking between UARTs during data transfer. When AFE = 1, the RTS bit determines the autoflow control enabled. Note that all UARTs do not support this feature, see your device-specific data manual for supported features. If this feature is not available, this bit is reserved and should be cleared to 0.
		0	Autoflow control is disabled.
		1	Autoflow control is enabled: • When RTS = 0, $\overline{\text{UARTn_CTS}}$ is only enabled. • When RTS = 1, $\overline{\text{UARTn_RTS}}$ and $\overline{\text{UARTn_CTS}}$ are enabled.
4	LOOP		Loop back mode enable. LOOP is used for the diagnostic testing using the loop back feature.
		0	Loop back mode is disabled.
		1	Loop back mode is enabled. When LOOP is set, the following occur: • The UARTn_TXD signal is set high. • The UARTn_RXD pin is disconnected • The output of the transmitter shift register (TSR) is lopped back in to the receiver shift register (RSR) input.
3	OUT2	0	OUT2 Control Bit
2	OUT1	0	OUT1 Control Bit
1	RTS		RTS control. When AFE = 1, the RTS bit determines the autoflow control enabled. Note that all UARTs do not support this feature, see your device-specific data manual for supported features. If this feature is not available, this bit is reserved and should be cleared to 0.
		0	$\overline{\text{UARTn_RTS}}$ is disabled, $\overline{\text{UARTn_CTS}}$ is only enabled.
		1	$\overline{\text{UARTn_RTS}}$ and $\overline{\text{UARTn_CTS}}$ are enabled.
0	Reserved	0	Reserved

Copyright © 2013–2016, Texas Instruments Incorporated

22.3.8 Line Status Register (LSR)

The line status register (LSR) is shown in Figure 22-16 and described in Table 22-17. LSR provides information to the CPU concerning the status of data transfers. LSR is intended for read operations only; do not write to this register. Bits 1 through 4 record the error conditions that produce a receiver line status interrupt.

Figure 22-16. Line Status Register (LSR)

31								16
Reserved								
R-0								

15	8	7	6	5	4	3	2	1	0
Reserved		RXFIFOE	TEMT	THRE	BI	FE	PE	OE	DR
R-0		R-0	R-1	R-1	R-0	R-0	R-0	R-0	R-0

LEGEND: R = Read only; -*n* = value after reset

Table 22-17. Line Status Register (LSR) Field Descriptions

Bit	Field	Value	Description
31-8	Reserved	0	Reserved
7	RXFIFOE		Receiver FIFO error.
			In non-FIFO mode:
		0	There has been no error, or RXFIFOE was cleared because the CPU read the erroneous character from the receiver buffer register (RBR).
		1	There is a parity error, framing error, or break indicator in the receiver buffer register (RBR).
			In FIFO mode:
		0	There has been no error, or RXFIFOE was cleared because the CPU read the erroneous character from the receiver FIFO and there are no more errors in the receiver FIFO.
		1	At least one parity error, framing error, or break indicator in the receiver FIFO.
6	TEMT		Transmitter empty (TEMT) indicator.
			In non-FIFO mode:
		0	Either the transmitter holding register (THR) or the transmitter shift register (TSR) contains a data character.
		1	Both the transmitter holding register (THR) and the transmitter shift register (TSR) are empty.
			In FIFO mode:
		0	Either the transmitter FIFO or the transmitter shift register (TSR) contains a data character.
		1	Both the transmitter FIFO and the transmitter shift register (TSR) are empty.
5	THRE		Transmitter holding register empty (THRE) indicator. If the THRE bit is set and the corresponding interrupt enable bit is set (ETBEI = 1 in IER), an interrupt request is generated.
			In non-FIFO mode:
		0	Transmitter holding register (THR) is not empty. THR has been loaded by the CPU.
		1	Transmitter holding register (THR) is empty (ready to accept a new character). The content of THR has been transferred to the transmitter shift register (TSR).
			In FIFO mode:
		0	Transmitter FIFO is not empty. At least one character has been written to the transmitter FIFO. You can write to the transmitter FIFO if it is not full.
		1	Transmitter FIFO is empty. The last character in the FIFO has been transferred to the transmitter shift register (TSR).

Copyright © 2013–2016, Texas Instruments Incorporated

Table 22-17. Line Status Register (LSR) Field Descriptions (continued)

Bit	Field	Value	Description
4	BI		Break indicator. The BI bit is set whenever the receive data input (UARTn_RXD) was held low for longer than a full-word transmission time. A full-word transmission time is defined as the total time to transmit the START, data, PARITY, and STOP bits. If the BI bit is set and the corresponding interrupt enable bit is set (ELSI = 1 in IER), an interrupt request is generated.
			In non-FIFO mode:
		0	No break has been detected, or the BI bit was cleared because the CPU read the erroneous character from the receiver buffer register (RBR).
		1	A break has been detected with the character in the receiver buffer register (RBR).
			In FIFO mode:
		0	No break has been detected, or the BI bit was cleared because the CPU read the erroneous character from the receiver FIFO and the next character to be read from the FIFO has no break indicator.
		1	A break has been detected with the character at the top of the receiver FIFO.
3	FE		Framing error (FE) indicator. A framing error occurs when the received character does not have a valid STOP bit. In response to a framing error, the UART sets the FE bit and waits until the signal on the RX pin goes high. Once the RX signal goes high, the receiver is ready to detect a new START bit and receive new data. If the FE bit is set and the corresponding interrupt enable bit is set (ELSI = 1 in IER), an interrupt request is generated.
			In non-FIFO mode:
		0	No framing error has been detected, or the FE bit was cleared because the CPU read the erroneous data from the receiver buffer register (RBR).
		1	A framing error has been detected with the character in the receiver buffer register (RBR).
			In FIFO mode:
		0	No framing error has been detected, or the FE bit was cleared because the CPU read the erroneous data from the receiver FIFO and the next character to be read from the FIFO has no framing error.
		1	A framing error has been detected with the character at the top of the receiver FIFO.
2	PE		Parity error (PE) indicator. A parity error occurs when the parity of the received character does not match the parity selected with the EPS bit in the line control register (LCR). If the PE bit is set and the corresponding interrupt enable bit is set (ELSI = 1 in IER), an interrupt request is generated.
			In non-FIFO mode:
		0	No parity error has been detected, or the PE bit was cleared because the CPU read the erroneous data from the receiver buffer register (RBR).
		1	A parity error has been detected with the character in the receiver buffer register (RBR).
			In FIFO mode:
		0	No parity error has been detected, or the PE bit was cleared because the CPU read the erroneous data from the receiver FIFO and the next character to be read from the FIFO has no parity error.
		1	A parity error has been detected with the character at the top of the receiver FIFO.
1	OE		Overrun error (OE) indicator. An overrun error in the non-FIFO mode is different from an overrun error in the FIFO mode. If the OE bit is set and the corresponding interrupt enable bit is set (ELSI = 1 in IER), an interrupt request is generated.
			In non-FIFO mode:
		0	No overrun error has been detected, or the OE bit was cleared because the CPU read the content of the line status register (LSR).
		1	Overrun error has been detected. Before the character in the receiver buffer register (RBR) could be read, it was overwritten by the next character arriving in RBR.
			In FIFO mode:
		0	No overrun error has been detected, or the OE bit was cleared because the CPU read the content of the line status register (LSR).
		1	Overrun error has been detected. If data continues to fill the FIFO beyond the trigger level, an overrun error occurs only after the FIFO is full and the next character has been completely received in the shift register. An overrun error is indicated to the CPU as soon as it happens. The new character overwrites the character in the shift register, but it is not transferred to the FIFO.

Copyright © 2013–2016, Texas Instruments Incorporated

Table 22-17. Line Status Register (LSR) Field Descriptions (continued)

Bit	Field	Value	Description
0	DR		Data-ready (DR) indicator for the receiver. If the DR bit is set and the corresponding interrupt enable bit is set (ERBI = 1 in IER), an interrupt request is generated.
			In non-FIFO mode:
		0	Data is not ready, or the DR bit was cleared because the character was read from the receiver buffer register (RBR).
		1	Data is ready. A complete incoming character has been received and transferred into the receiver buffer register (RBR).
			In FIFO mode:
		0	Data is not ready, or the DR bit was cleared because all of the characters in the receiver FIFO have been read.
		1	Data is ready. There is at least one unread character in the receiver FIFO. If the FIFO is empty, the DR bit is set as soon as a complete incoming character has been received and transferred into the FIFO. The DR bit remains set until the FIFO is empty again.

Copyright © 2013–2016, Texas Instruments Incorporated

22.3.9 Modem Status Register (MSR)

The Modem status register (MSR) is shown in Figure 22-17 and described in Table 22-18. MSR provides information to the CPU concerning the status of modem control signals. MSR is intended for read operations only; do not write to this register.

Figure 22-17. Modem Status Register (MSR)

31								16
Reserved								
R-0								
15	8	7	6	5	4	3	2	1	0
Reserved		CD	RI	DSR	CTS	DCD	TERI	DDSR	DCTS
R-0		R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0

LEGEND: R = Read only; -*n* = value after reset

Table 22-18. Modem Status Register (MSR) Field Descriptions

Bit	Field	Value	Description
31-8	Reserved	0	Reserved
7	CD	0	Complement of the Carrier Detect input. When the UART is in the diagnostic test mode (loopback mode MCR[4] = 1), this bit is equal to the MCR bit 3 (OUT2).
6	RI	0	Complement of the Ring Indicator input. When the UART is in the diagnostic test mode (loopback mode MCR[4] = 1), this bit is equal to the MCR bit 2 (OUT1).
5	DSR	0	Complement of the Data Set Ready input. When the UART is in the diagnostic test mode (loopback mode MCR[4] = 1), this bit is equal to the MCR bit 0 (DTR).
4	CTS	0	Complement of the Clear To Send input. When the UART is in the diagnostic test mode (loopback mode MCR[4] = 1), this bit is equal to the MCR bit 1 (RTS).
3	DCD	0	Change in DCD indicator bit. DCD indicates that the DCD input has changed state since the last time it was read by the CPU. When DCD is set and the modem status interrupt is enabled, a modem status interrupt is generated.
2	TERI	0	Trailing edge of RI (TERI) indicator bit. TERI indicates that the RI input has changed from a low to a high. When TERI is set and the modem status interrupt is enabled, a modem status interrupt is generated.
1	DDSR	0	Change in DSR indicator bit. DDSR indicates that the DSR input has changed state since the last time it was read by the CPU. When DDSR is set and the modem status interrupt is enabled, a modem status interrupt is generated.
0	DCTS	0	Change in CTS indicator bit. DCTS indicates that the CTS input has changed state since the last time it was read by the CPU. When DCTS is set (autoflow control is not enabled and the modem status interrupt is enabled), a modem status interrupt is generated. When autoflow control is enabled, no interrupt is generated.

Copyright © 2013–2016, Texas Instruments Incorporated

22.3.10 Scratch Pad Register (SCR)

The Scratch Pad register (SCR) is shown in Figure 22-18 and described in Table 22-19. SCR is intended for programmer's use as a scratch pad. It temporarily holds the programmer's data without affecting UART operation.

Figure 22-18. Scratch Pad Register (SCR)

31	16
Reserved	
R-0	

15	8	7	0
Reserved		SCR	
R-0		R-0	

LEGEND: R = Read only; -*n* = value after reset

Table 22-19. Scratch Pad Register (MSR) Field Descriptions

Bit	Field	Value	Description
31-8	Reserved	0	Reserved
7-0	SCR	0	These bits are intended for the programmer's use as a scratch pad in the sense that it temporarily holds the programmer's data without affecting any other UART operation.

22.3.11 Divisor Latches (DLL and DLH)

Two 8-bit register fields (DLL and DLH), called divisor latches, store the 16-bit divisor for generation of the baud clock in the baud generator. The latches are in DLH and DLL. DLH holds the most-significant bits of the divisor, and DLL holds the least-significant bits of the divisor. These divisor latches must be loaded during initialization of the UART in order to ensure desired operation of the baud generator. Writing to the divisor latches results in two wait states being inserted during the write access while the baud generator is loaded with the new value.

Access considerations:

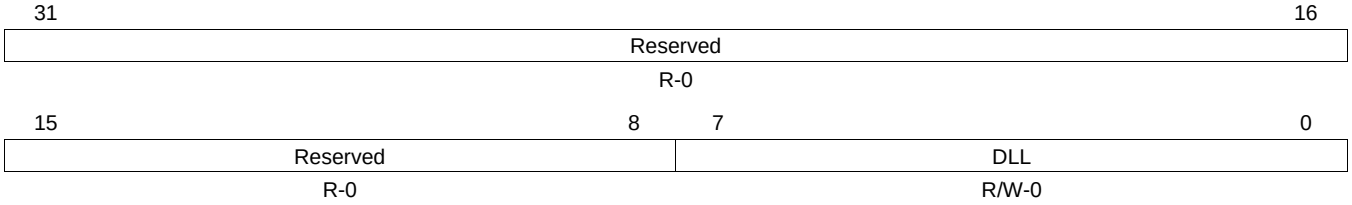
- RBR, THR, and DLL share one address. When DLAB = 1 in LCR, all accesses at the shared address are accesses to DLL. When DLAB = 0, reading from the shared address gives the content of RBR, and writing to the shared address modifies THR.
- IER and DLH share one address. When DLAB = 1 in LCR, accesses to the shared address read or modify to DLH. When DLAB = 0, all accesses at the shared address read or modify IER.

DLL and DLH also have dedicated addresses. If you use the dedicated addresses, you can keep the DLAB bit cleared, so that RBR, THR, and IER are always selected at the shared addresses.

The divisor LSB latch (DLL) is shown in Figure 22-19 and described in Table 22-20. The divisor MSB latch (DLH) is shown in Figure 22-20 and described in Table 22-21.

Copyright © 2013–2016, Texas Instruments Incorporated

Figure 22-19. Divisor LSB Latch (DLL)

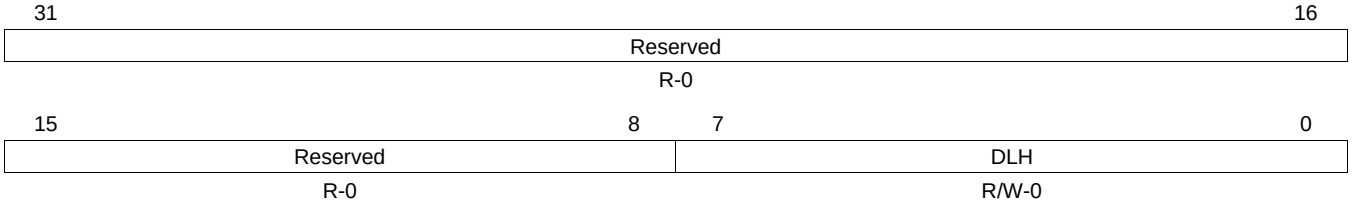


LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 22-20. Divisor LSB Latch (DLL) Field Descriptions

Bit	Field	Value	Description
31-8	Reserved	0	Reserved
7-0	DLL	0-FFh	The 8 least-significant bits (LSBs) of the 16-bit divisor for generation of the baud clock in the baud rate generator.

Figure 22-20. Divisor MSB Latch (DLH)



LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 22-21. Divisor MSB Latch (DLH) Field Descriptions

Bit	Field	Value	Description
31-8	Reserved	0	Reserved
7-0	DLH	0-FFh	The 8 most-significant bits (MSBs) of the 16-bit divisor for generation of the baud clock in the baud rate generator.

Copyright © 2013–2016, Texas Instruments Incorporated

22.3.12 Revision Identification Registers (REVID1 and REVID2)

The revision identification registers (REVID1 and REVID2) contain peripheral identification data for the peripheral. REVID1 is shown in Figure 22-21 and described in Table 22-22. REVID2 is shown in Figure 22-22 and described in Table 22-23.

Figure 22-21. Revision Identification Register 1 (REVID1)

31	0
REVID1	
R-1102 0002h	

LEGEND: R = Read only; -*n* = value after reset

Table 22-22. Revision Identification Register 1 (REVID1) Field Descriptions

Bit	Field	Value	Description
31-0	REVID1	1102 0002h	Peripheral Identification Number

Figure 22-22. Revision Identification Register 2 (REVID2)

31	16
Reserved	
R-0	

15	8	7	0
Reserved		REVID2	
R-0		R-0	

LEGEND: R = Read only; -*n* = value after reset

Table 22-23. Revision Identification Register 2 (REVID2) Field Descriptions

Bit	Field	Value	Description
31-8	Reserved	0	Reserved
7-0	REVID2	0	Peripheral Identification Number

Copyright © 2013–2016, Texas Instruments Incorporated

22.3.13 Power and Emulation Management Register (PWREMU_MGMT)

The power and emulation management register (PWREMU_MGMT) is shown in Figure 22-23 and described in Table 22-24.

Figure 22-23. Power and Emulation Management Register (PWREMU_MGMT)

31	16
Reserved	
R-0	

15	14	13	12	1	0
Rsvd	UTRST	URRST	Reserved		FREE
R/W-0	R/W-0	R/W-0	R-1		R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 22-24. Power and Emulation Management Register (PWREMU_MGMT) Field Descriptions

Bit	Field	Value	Description
31-16	Reserved	0	Reserved
15	Reserved	0	Reserved. This bit must always be written with a 0.
14	UTRST		UART transmitter reset. Resets and enables the transmitter.
		0	Transmitter is disabled and in reset state.
		1	Transmitter is enabled.
13	URRST		UART receiver reset. Resets and enables the receiver.
		0	Receiver is disabled and in reset state.
		1	Receiver is enabled.
12-1	Reserved	1	Reserved
0	FREE		Free-running enable mode bit. This bit determines the emulation mode functionality of the UART. When halted, the UART can handle register read/write requests, but does not generate any transmission/reception, interrupts or events.
		0	If a transmission is not in progress, the UART halts immediately. If a transmission is in progress, the UART halts after completion of the one-word transmission.
		1	Free-running mode is enabled; UART continues to run normally.

Copyright © 2013–2016, Texas Instruments Incorporated

22.3.14 Mode Definition Register (MDR)

The Mode Definition register (MDR) determines the over-sampling mode for the UART. MDR is shown in Figure 22-24 and described in Table 22-25.

Figure 22-24. Mode Definition Register (MDR)

31	16
Reserved	
R-0	

15	1	0
Reserved		OSM_SEL
R-0		R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 22-25. Mode Definition Register (MDR) Field Descriptions

Bit	Field	Value	Description
31-1	Reserved	0	Reserved
0	OSM_SEL		Over-Sampling Mode Select.
		0	16× over-sampling.
		1	13× over-sampling.

Copyright © 2013–2016, Texas Instruments Incorporated

Revision History

NOTE: Page numbers for previous revisions may differ from page numbers in the current version.

Changes from July 1, 2016 to September 21, 2016 (from A Revision (July 2016) to B Revision)	Page

Copyright © 2013–2016, Texas Instruments Incorporated

IMPORTANT NOTICE

Texas Instruments Incorporated and its subsidiaries (TI) reserve the right to make corrections, enhancements, improvements and other changes to its semiconductor products and services per JESD46, latest issue, and to discontinue any product or service per JESD48, latest issue. Buyers should obtain the latest relevant information before placing orders and should verify that such information is current and complete. All semiconductor products (also referred to herein as “components”) are sold subject to TI’s terms and conditions of sale supplied at the time of order acknowledgment.

TI warrants performance of its components to the specifications applicable at the time of sale, in accordance with the warranty in TI’s terms and conditions of sale of semiconductor products. Testing and other quality control techniques are used to the extent TI deems necessary to support this warranty. Except where mandated by applicable law, testing of all parameters of each component is not necessarily performed.

TI assumes no liability for applications assistance or the design of Buyers’ products. Buyers are responsible for their products and applications using TI components. To minimize the risks associated with Buyers’ products and applications, Buyers should provide adequate design and operating safeguards.

TI does not warrant or represent that any license, either express or implied, is granted under any patent right, copyright, mask work right, or other intellectual property right relating to any combination, machine, or process in which TI components or services are used. Information published by TI regarding third-party products or services does not constitute a license to use such products or services or a warranty or endorsement thereof. Use of such information may require a license from a third party under the patents or other intellectual property of the third party, or a license from TI under the patents or other intellectual property of TI.

Reproduction of significant portions of TI information in TI data books or data sheets is permissible only if reproduction is without alteration and is accompanied by all associated warranties, conditions, limitations, and notices. TI is not responsible or liable for such altered documentation. Information of third parties may be subject to additional restrictions.

Resale of TI components or services with statements different from or beyond the parameters stated by TI for that component or service voids all express and any implied warranties for the associated TI component or service and is an unfair and deceptive business practice. TI is not responsible or liable for any such statements.

Buyer acknowledges and agrees that it is solely responsible for compliance with all legal, regulatory and safety-related requirements concerning its products, and any use of TI components in its applications, notwithstanding any applications-related information or support that may be provided by TI. Buyer represents and agrees that it has all the necessary expertise to create and implement safeguards which anticipate dangerous consequences of failures, monitor failures and their consequences, lessen the likelihood of failures that might cause harm and take appropriate remedial actions. Buyer will fully indemnify TI and its representatives against any damages arising out of the use of any TI components in safety-critical applications.

In some cases, TI components may be promoted specifically to facilitate safety-related applications. With such components, TI’s goal is to help enable customers to design and create their own end-product solutions that meet applicable functional safety standards and requirements. Nonetheless, such components are subject to these terms.

No TI components are authorized for use in FDA Class III (or similar life-critical medical equipment) unless authorized officers of the parties have executed a special agreement specifically governing such use.

Only those TI components which TI has specifically designated as military grade or “enhanced plastic” are designed and intended for use in military/aerospace applications or environments. Buyer acknowledges and agrees that any military or aerospace use of TI components which have ***not*** been so designated is solely at the Buyer's risk, and that Buyer is solely responsible for compliance with all legal and regulatory requirements in connection with such use.

TI has specifically designated certain components as meeting ISO/TS16949 requirements, mainly for automotive use. In any case of use of non-designated products, TI will not be responsible for any failure to meet ISO/TS16949.

Products		**Applications**	
Audio	www.ti.com/audio	Automotive and Transportation	www.ti.com/automotive
Amplifiers	amplifier.ti.com	Communications and Telecom	www.ti.com/communications
Data Converters	dataconverter.ti.com	Computers and Peripherals	www.ti.com/computers
DLP® Products	www.dlp.com	Consumer Electronics	www.ti.com/consumer-apps
DSP	dsp.ti.com	Energy and Lighting	www.ti.com/energy
Clocks and Timers	www.ti.com/clocks	Industrial	www.ti.com/industrial
Interface	interface.ti.com	Medical	www.ti.com/medical
Logic	logic.ti.com	Security	www.ti.com/security
Power Mgmt	power.ti.com	Space, Avionics and Defense	www.ti.com/space-avionics-defense
Microcontrollers	microcontroller.ti.com	Video and Imaging	www.ti.com/video
RFID	www.ti-rfid.com		
OMAP Applications Processors	www.ti.com/omap	**TI E2E Community**	e2e.ti.com
Wireless Connectivity	www.ti.com/wirelessconnectivity		

Mailing Address: Texas Instruments, Post Office Box 655303, Dallas, Texas 75265
Copyright © 2016, Texas Instruments Incorporated