Application Report
SPRA899 – February 2003

OMAP DSP DMA Throughput Analysis

Isara Indra, Padmaja Nimmagadda Technical Staff, DSP Applications

ABSTRACT

The DSP Catalog OMAP™ architecture devices contain a six-channel DSP DMA that is used to transfer blocks of data without intervention by the TIARM925T MPU or the C55X™ DSP. This application report analyzes the DMA throughput between some of the different sources and destinations. Optimization techniques for improving throughput are also discussed. Complete examples are given for using external memory such as SDRAM and SRAM for transferring data to and from internal memory.

Contents

1 Introduction 2
2 System Setup 3
3 Peripheral Registers Setup 3
4 Throughput Analysis Examples 6
5 Logic Analyzer Images of EMIFF to EMIFF DSP DMA Transfers 8
6 Logic Analyzer Images of EMIFS to EMIFS DSP DMA Transfers 10
7 DSP DMA Transfers with IMIF as Source 13
8 DSP DMA Transfers with EMIFS as Source 14
9 DSP DMA Transfers with EMIFF as Source 16
10 DSP DMA Transfer with SARAM as Source 17
11 DSP DMA Transfer with DARAM Source 18
12 DSP DMA Transfer with TIPB Destination 19
13 Conclusion 19
14 References 20
Appendix A OMAP Block DIagram 21
Appendix B MPU Memory Map 22
Appendix C EMIFS Behavior 23
Appendix D EMIFF Behavior 24

List of Figures

Figure 1. DSP GPIO Set-Up Code 7
Figure 2. 10K x 32 EMIFF to EMIFF No-Burst Transfer 8
Figure 3. 20K x 16 EMIFF to EMIFF Burst Transfer 8

Figure 4. 20K x 16 EMIFF to EMIFF No-Burst Transfer 9
Figure 5. 40K x 8 EMIFF to EMIFF Burst Transfer 9
Figure 6. 40K x 8 EMIFF to EMIFF No-Burst Transfer 10
Figure 7. 10K x 32 EMIFS to EMIFS No-Burst Transfer 10
Figure 8. 20K x 16 EMIFS to EMIFS Burst Transfer 11
Figure 9. 20K x 16 EMIFS to EMIFS No-Burst Transfer 11
Figure 10. 40K x 8 EMIFS to EMIFS Burst Transfer 12
Figure 11. 40K x 8 EMIFS to EMIFS No-Burst Transfer 12
Figure A–1. OMAP Block Diagram 21

List of Tables

Table 1. Clock Setup 3
Table 2. Transfer Sources, Destinations 6
Table 3. Transfer Mode 7
Table 4. Transfer Addresses 7
Table 5. DMA Transfers with IMIF as the Source 13
Table 6. DSP DMA Transfers with EMIFS Source 14
Table 7. DSP DMA Transfers With EMIFS as Destination 15
Table 8. DSP DMA Transfers With EMIFS as Source 15
Table 9. DSP DMA Transfers with EMIFF Source 16
Table 10. DSP DMA Transfers with SARAM as Source 17
Table 11. DSP DMA Transfers with DARAM as Source 18
Table 12. DSP DMA Transfers with TIPB (DSP McBSP1) 19
Table B–1. MPU Memory Map 22
Table C–1. EMIFS Cycle Behavior with ARM I/D-Caches Disabled,
EMIFS_PRIO = 0x00000000, and EMIFS_CS3_CONFIG.FCLKDIV=TC/2 23
Table C–2. EMIFS /CS Active Widths for Asynchronous Reads/Writes 23
Table D–1. EMIFF Behavior for Single and Burst Reads/Writes (TC Cycles) 24

1 Introduction

The DSP subsystem has its own dedicated DMA controller, which is entirely independent of the MPU or the system DMA controller. The DSP DMA controller has six generic channels and five physical ports available for source or destination data. These five ports are the SARAM port, DARAM port, external memory interface (EMIF), DSP TIPB port, and MPUI port. The DSP may configure the DSP DMA controller to transfer data between the SARAM, DARAM, EMIF, and TIPB ports, but the MPUI port is a dedicated port used for MPU or system DMA initiated transfers to DSP subsystem resources. The SARAM and DARAM ports are used to access local DSP memories and the TIPB port is used to access the registers of the DSP peripherals. The EMIF port of the DSP DMA controller is used to access the traffic controller via the DSP memory management unit (MMU). This application report analyzes the throughput between the different ports using the DSP DMA. Optimization techniques for improving throughput are discussed.

2 System Setup

The following were used in the course of creating this applications report:

- A test board with external Samsung K4S561632C-TL75 256M-bit SDRAM and 512KB Samsung K6T4016U3C-TB70 SRAM
- OMAP Code Composer Studio™ v2.0 for TIARM925 MPU and C55X DSP
- USB JTAG Emulator from FLEXDS
- Tektronix™ TLA714 Logic Analyzer
- POMAP5910CGZG

The OMAP5910 MPU was clocked at 75 MHz while the DSP clock was setup for 150 MHZ operation. The OMAP5910 traffic controller (TC) clock was setup for 75 MHz operation. The internal SRAM (IMIF) ran at the same speed as the TC while the slow external memory interface (EMIFS) connected to asynchronous SRAM was clocked at 37.5 MHz, with 3 wait states (WS) for reads and 1 WS for writes. Write enable length (WELEN) was set at 1. The fast external memory interface (EMIFF) connected to SDRAM was clocked at 75 MHz with a CAS latency of 2.

Table 1. Clock Setup

DSP	MPU	MPUI	TC/IMIF/SDMA	EMIFF		EMIFS			
150 MHz	75 MHz	DIVF 2	75 MHz	75 MHz	CAS 2	37.5 MHz	WELEN 1	3WS Rd	1WS Wr

A combination of ARM assembly and unoptimized C code was used for performing the DSP DMA transfers on a single channel. The ARM code was executed from EMIFS SRAM. Test setup and results tables were stored in internal IMIF. The DSP was executed from SARAM0. Please refer to *OMAP5910 Dual-Core Processor Data Manual* (SPRS197) for detailed DSP memory maps.

McBSP1 was used in the loop back mode to measure DSP DMA throughput for TIPB access.

3 Peripheral Registers Setup

TC was configured as follows:

- EMIFS_PRIO, EMIFF_PRIO, and IMIF_PRIO settings
  - 31:16 reserved.
  - 15:12 LB host set to 000b for single transfers, not used in this analysis.
  - 11:8 system DMA set to 0000b for single transfer, not used in this analysis.
  - 7 reserved
  - 6:4 DSP (CPU or DMA) set to 000b for single transfer, not used in this analysis.
  - 3 reserved
  - 2:0 ARM set to 000b for single transfers
- EMIFS_CONFIG_REG = 0x00000012

  - Boot mode was set for CS3 space at 0x00000000
- EMIFS_CS3_CONFIG = 0x00001139
  - WELEN=1, WRWST=1, RDWST=3, FLCLKDIV=TC/2
- EMIFF_SDRAM_CONFIG = 0x0020B74
  - SDF1 frequency range, auto-refresh enabled and calculated from SDRAM data sheet for minimum, 16-bit bus, 64 M bits, 4 banks
- EMIFF_MRS = 0x00000027
  - CAS latency 2, full-page burst length
- TIMEOUT1 = TIMEOUT2 = TIMEOUT3 = 0x00000000
- ENDIANISM = 0x00000000
  - Little endianism

DSP DMA was configured as follows:

1. DSP DMA configuration for EMIFF, EMIFS, IMIF transfer.
   DSP channel 0
   - DSP_DMA_GCR = 0x0000000C
     Clock autogating ON, free
   - DSP_DMA_CSDP = various
     No burst and 16-bit, no burst and 32-bit, source and destination burst 4 and 16-bit
   - DSP_DMA_CCR = 0xA060
     Source and destination single index, entire frame is transferred
   - DSP_DMA_CICR = 0x0009
     Frame and timeout interrupts enabled
   - DSP_DMA_CSSA_L = various
   - DSP_DMA_CSSA_U = various
   - DSP_DMA_CDSA_L = various
   - DSP_DMA_CDSA_U = various
   - DSP_DMA_CEN = 0x4e20 and 0x2710
     20,000 and 10,000 elements
   - DSP_DMA_CFN = 1
     Single frame
   - DSP_DMA_CFI = 1
     Single frame index
   - DSP_DMA_CEI = 1
     Single element index
2. DSP DMA configuration for TIPB (McBSP) transfer.
   DSP channel 0 (Transmit)

– DSP_DMA_GCR = 0x0000000C
  Clock autogating ON, free
– DSP_DMA_CSDP = various
  No burst and 16-bit, destination peripheral port
– DSP_DMA_CCR = 0x2046
  Source single index and destination constant, transfer one element at a time, McBSP1 transmit event synchronization
– DSP_DMA_CICR = 0x0009
  Frame and timeout interrupts enabled
– DSP_DMA_CSSA_L = various
– DSP_DMA_CSSA_U = various
– DSP_DMA_CDSA_L = McBSP1_DXR1
– DSP_DMA_CDSA_U = McBSP1_DXR1
– DSP_DMA_CEN = 0x4e20
  20,000 elements
– DSP_DMA_CFN = 1
  Single frame
– DSP_DMA_CFI = 1
  Single frame index
– DSP_DMA_CEI = 1
  Single element index

DSP channel 1 (Receive)

– DSP_DMA_GCR = 0x0000000C
  Clock autogating ON, free
– DSP_DMA_CSDP = various
  No burst and 16-bit, source peripheral port.
– DSP_DMA_CCR = 0x8045
  Source constant and destination single index, transfer one element at a time, McBSP1 receive event synchronization
– DSP_DMA_CICR = 0x0009
  Frame and timeout interrupts enabled
– DSP_DMA_CSSA_L = McBSP1_DRR1
– DSP_DMA_CSSA_U = McBSP1_DRR1
– DSP_DMA_CDSA_L = various
– DSP_DMA_CDSA_U = various
– DSP_DMA_CEN = 0x4e20
  20,000 elements

- DSP_DMA_CFN = 1
  Single frame
- DSP_DMA_CFI = 1
  Single frame index
- DSP_DMA_CEI = 1
  Single element index

DSP McBSP1 was configured as follows:

- McBSP1_SPCR1 = 0x8000
  Digital Loop back enable
- McBSP1_ SPCR2= 0x0000
- McBSP1_RCR1 = 0x0040
  Receive word length 16 bits
- McBSP1_RCR2 = 0x0000
- McBSP1_XCR1 = 0x0040
  Transmit word length 16 bits
- McBSP1_XCR2 = 0x0000
- McBSP1_SRGR1 = 0x0101
  CLFGDV =1, FWID = 1
- McBSP1_SRGR2 = 0x200F
  Frame-sync period bits for FSG =16
- McBSP1_PCR = 0x0A00

4 Throughput Analysis Examples

Throughput examples were generated for transfer between all combinations between DSP internal memory, TIPB, and external memory (EMIFF, EMIFFS, and IMIF). Transfer of 10Kx32 bits and 20Kx16 bits were done without burst. Transfers of 20Kx16 bits were done with burst also.

Table 2. Transfer Sources, Destinations

Transfer		Destination					
		IMIF	**EMIFS**	**EMIFF**	**SARAM**	**DARAM**	**TIPB**
Source	IMIF	1,2,3	1,2,3	1,2,3	1,2,3	1,2,3	N/A
	EMIFS	1,2,3	1,2,3	1,2,3	1,2,3	1,2,3	N/A
	EMIFF	1,2,3	1,2,3	1,2,3	1,2,3	1,2,3	N/A
	SARAM	1,2,3	1,2,3	1,2,3	1,2,3	1,2,3	1,2
	DARAM	1,2,3	1,2,3	1,2,3	1,2,3	1,2,3	1,2
	TIPB	N/A	N/A	N/A	1,2	1,2	N/A

Table 3. Transfer Mode

Mode 1	Mode 2	Mode 3
20Kx16 no burst	10Kx32 no burst	20Kx16 burst 4

Table 4. Transfer Addresses

IMIF (ARM Address)	EMIFS (ARM Address)	EMIFF (ARM Address)	SARAM (DSP Address)	DARAM (DSP Address)
0x20002000	0x00040000	0x10002000	0x00013000	0x00005000
0x20007000	0x00060000	0x1000C000	0x0000D000	0x00002000

The external latency time was measured using a GPIO pin as a logic analyzer trigger. After the GPIO pin was driven high in the test code, the DMA channel was directly enabled. No internal or external sync events were used. Latency was not measured where both the source and destination for the transfer were contained within IMIF or DSP internal memory. The EMIFS address and data busses were observed on the logic analyzer for the first few data transfers. Latency is defined as the time in TC cycles for the first DSP DMA transfer. It is calculated from the timing diagrams captured using a logic analyzer.

```
// Sample code for setting up DSP GPIO 3:
 ARM setup:
#define GPIO_XX *( (volatile unsigned short *) 0xFFFCE018 ) /* Pin Host Control
Register */
GPIO_XX = 0xFFF7; // Pin 3is routed to DSP GPIO port, – 0 for DSP, – 1 for ARM

DSP setup:
#define GPIO_DIR *( (volatile ioport unsigned short *) 0xF004 ) /* DSP GPIO
Direction Control Register */
#define GPIO_REG *( (volatile ioport unsigned short *) 0xF002 ) /* DSP GPIO Data
Output Register */


GPIO_DIR = 0xFFF7; // Direction control, GPIO – 0 for output– 1 for input.

GPIO_REG = 0x0008; // Set GPIO high


DMA_EnableChannel(DMA_CHANNEL_0); //Enable DSP DMA channel 0

// Polling Channel 0 Status registers bit ’FrameEnd’ in eternal loop

while((DMA_ACC(NO_GLOB_REG, DMA_CHANNEL_0,
DMA_CSR)&(0x0001<<DSP_DMA_CSR_CHX_FRAME_POS))==0);

GPIO_REG = 0x0000; // Set GPIO low
```

Figure 1. DSP GPIO Set-Up Code

Average throughput was measured using an ARM timer peripheral running at 6 MHz. The timer was enabled in code before the logic analyzer was enabled and disabled after a polling loop detected the end of the transfer. The timer values were scaled to derive the TC cycle count.

Throughput is defined as the time in TC cycles for a 16-bit transfer from a source to the destination by the DSP DMA.

It is calculated as $= \dfrac{(\textit{timer counter value}) * 75\ \text{MHz}}{6\ \text{MHz} * 20000}$

Timer counter is running off of a 6-MHz clock. TC running at 75 MHz and the timer counter value provides the time for 20000 16-bit transfers.

5 Logic Analyzer Images of EMIFF to EMIFF DSP DMA Transfers

In Figure 2, Figure 3, Figure 4, Figure 5, and Figure 6 the sample ticks are at the same rate as the TC clock (75 MHz).

Figure 2 shows the read and write timing waveforms for a 32-bit no-burst transfer from EMIFF to EMIFF. At cursor 1, a 32-bit DSP DMA read is followed by the row terminate, followed by refresh, followed by three consecutive 32-bit reads. The row is opened by the active NSRAS and the closing of the row is done with a precharge, NSRAS and NSWE active on the same TC cycle. The read is initiated by the active NSCAS. In this example, the CAS latency is 2. This is shown with the data valid at 2 TC clocks after the NSCAS active. At cursor 2, the DSP DMA EMIFF 32-bit burst write follows the closing of the row on the previous reads.

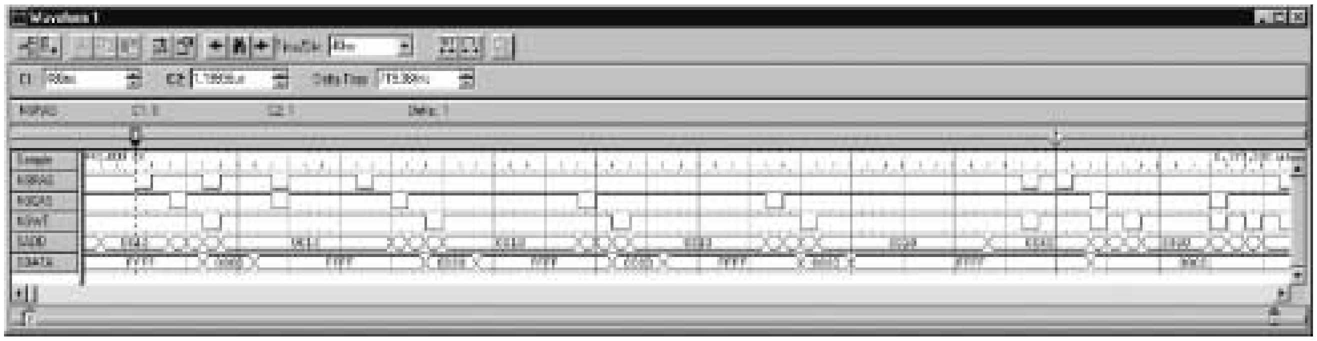

Figure 2. 10K x 32 EMIFF to EMIFF No-Burst Transfer

Figure 3 shows the read and write timing waveforms for a 16-bit burst transfer from EMIFF to EMIFF. At cursor 1, DSP DMA EMIFF bursts of 4x16 bit are shown. These four reads occur on the same row of the SDRAM. At cursor 2, the DSP DMA EMIFF 16-bit burst write follows the closing of the row on the previous reads.

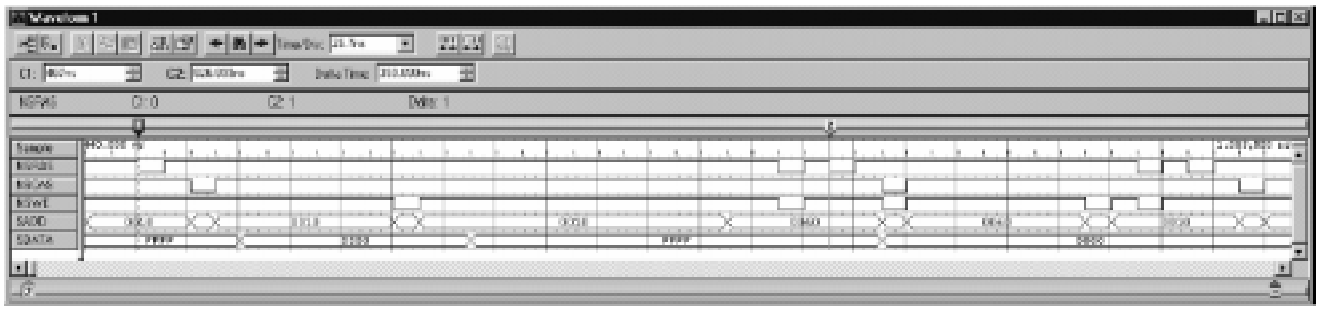

Figure 3. 20K x 16 EMIFF to EMIFF Burst Transfer

Figure 4 shows the read and write timing waveforms for a 16-bit non burst transfer from EMIFF to EMIFF. At cursor 1, five 16-bit DSP DMA EMIFF reads in the non-burst mode are shown. A burst terminate can be seen after each 16-bit read. These 4 reads occur on the same row of the SDRAM. At cursor 2, the DSP DMA EMIFF 16-bit non-burst write follows the closing of the row on the previous reads.

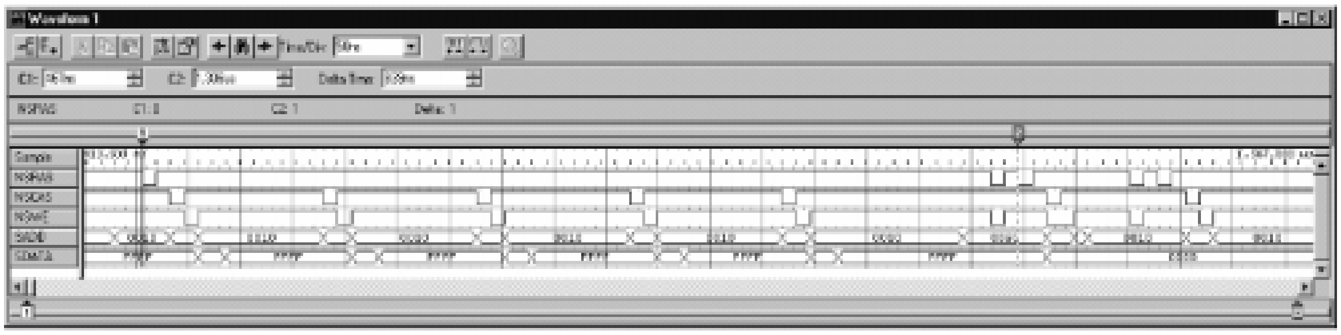

Figure 4. 20K x 16 EMIFF to EMIFF No-Burst Transfer

Figure 5 shows the read and write timing waveforms for 8-bit burst transfer from EMIFF to EMIFF. At cursor 1, eight 16-bit DSP DMA EMIFF reads in the burst mode are shown.

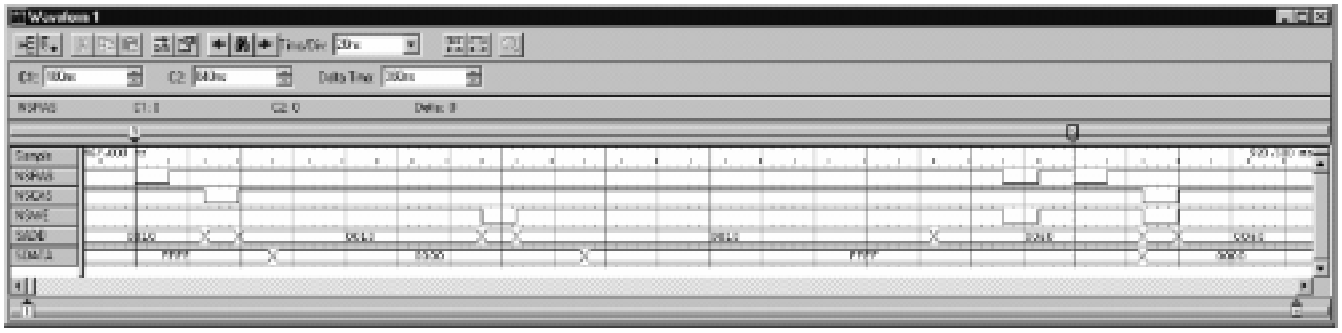

Figure 5. 40K x 8 EMIFF to EMIFF Burst Transfer

Figure 6 shows the read and write timing waveforms for an 8-bit non burst transfer from EMIFF to EMIFF. At cursor 1, five 8-bit DSP DMA EMIFF reads in the non-burst mode are shown. A burst terminate can be seen after each 8-bit read. These reads occur on the same row of the SDRAM. At cursor 2, the DSP DMA EMIFF 8-bit non-burst write follows the closing of the row on the previous reads.

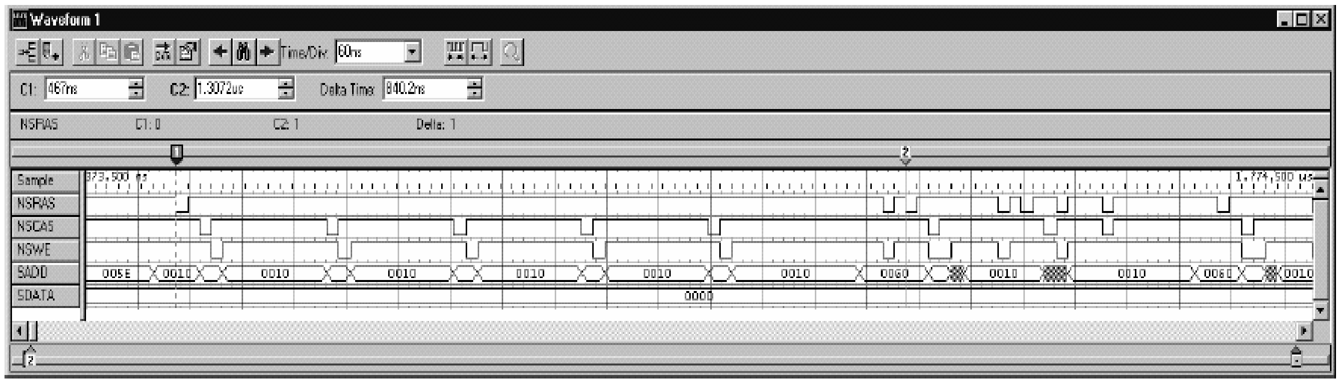

Figure 6. 40K x 8 EMIFF to EMIFF No-Burst Transfer

6 Logic Analyzer Images of EMIFS to EMIFS DSP DMA Transfers

Beginning at cursor 1 in Figure 7, 4 sequences of 32-bit program fetch followed by 32-bit DSP DMA read are observed. At cursor 2, a DSP DMA write follows the 32-bit data read.

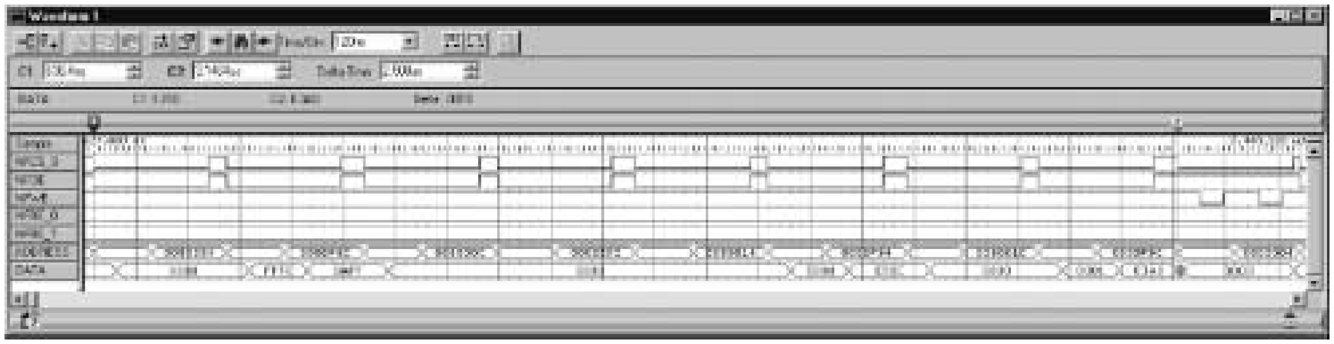

Figure 7. 10K x 32 EMIFS to EMIFS No-Burst Transfer

At cursor1 of Figure 8, two consecutive DSP DMA EMIFS read bursts of 4 by 16-bit are shown. The EMIFS interface combines these into a single 8 by 16-bit burst read to the external SRAM. This avoids the wasted cycles between the single 16-bit or 32-reads. The entire set of bursts completes in 10 TC cycles per 16-bit word (80 TC cycles total) with a following inactive /CS for 3 TC cycles. At cursor 2, two consecutive DSP DMA EMIFS burst writes of 4 by 16-bit are shown. Similar to the burst read of the same mode, the EMIFS implements this as a single 8 by 16 bit burst.

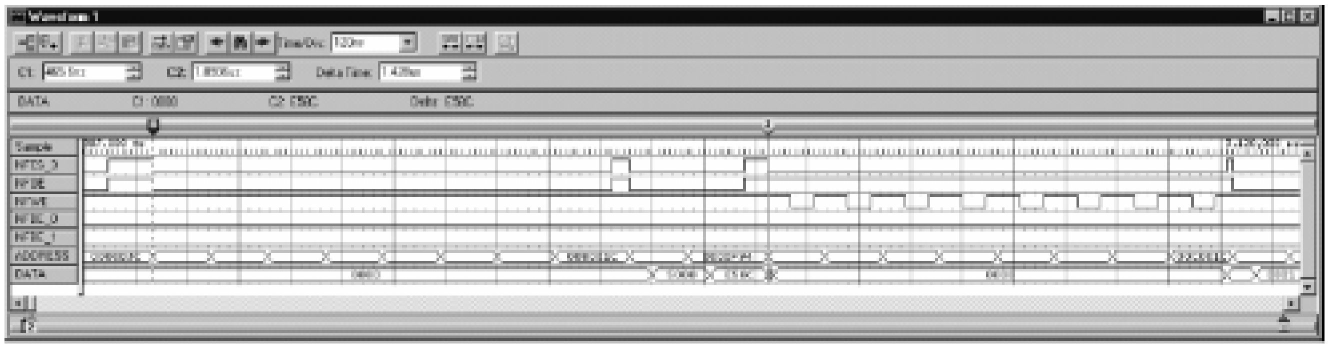

Figure 8. 20K x 16 EMIFS to EMIFS Burst Transfer

Beginning at cursor 1 of Figure 9, five DSP DMA non-burst 16-bit reads and five 32-bit program fetches from external asynchronous SRAM are interspersed. At cursor 2, a DSP DMA write follows the CPU fetch.

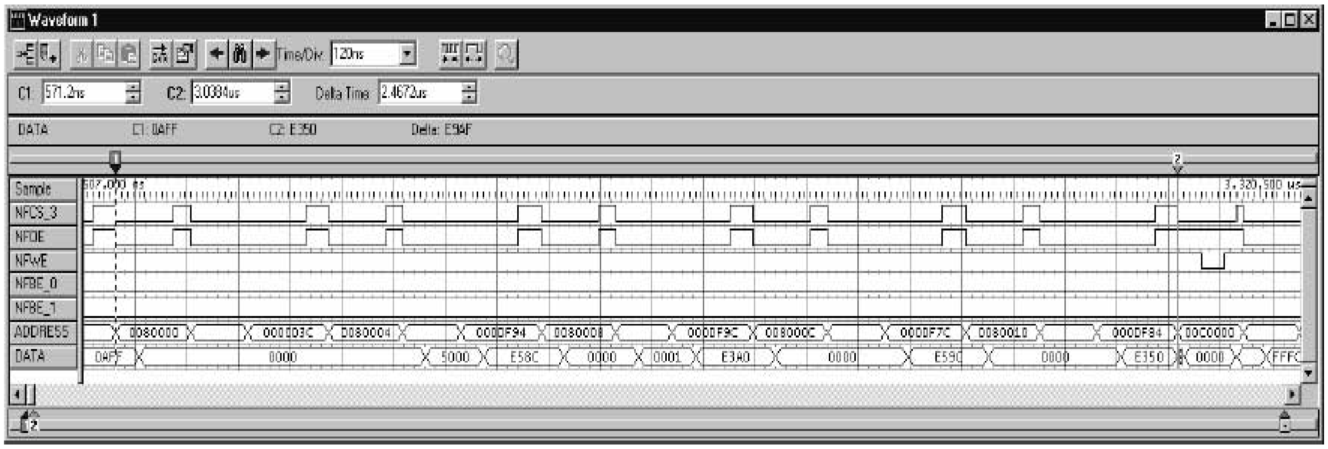

Figure 9. 20K x 16 EMIFS to EMIFS No-Burst Transfer

At cursor 1 of Figure 10, four consecutive DSP DMA EMIFS read bursts of 4 by 8-bit are shown. The EMIFS interface combines these into a single 8 by 16-bit burst read to the external SRAM. This avoids the wasted cycles between the single 8-bit or 16-reads. At cursor 2, four consecutive DSP DMA EMIFS burst writes of 4 by 8-bit are shown. Similar to the burst read of the same mode, the EMIFS implements this as a single 8 by 16 bit burst.

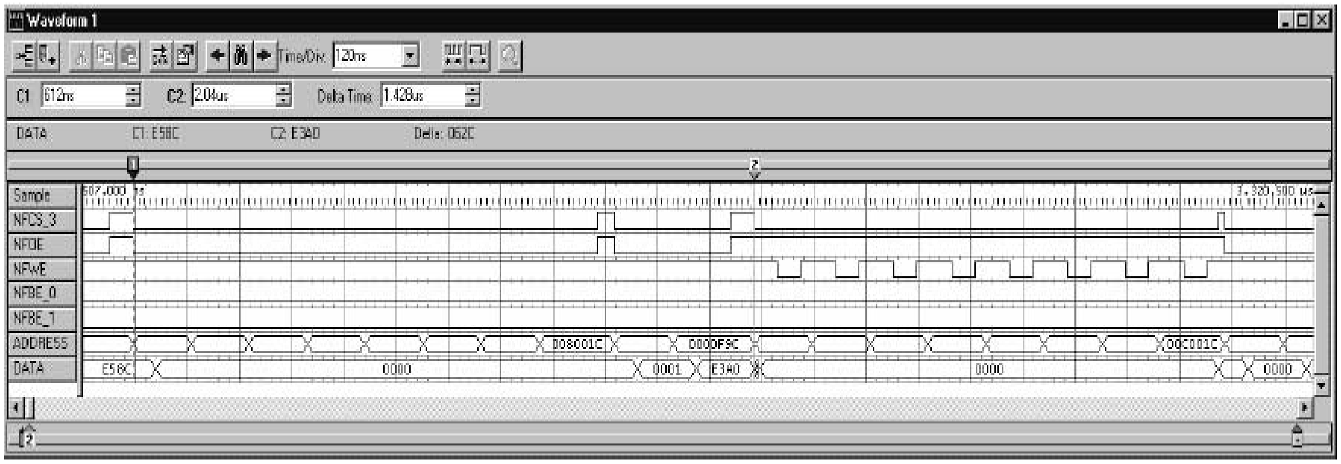

Figure 10. 40K x 8 EMIFS to EMIFS Burst Transfer

Beginning at cursor 1 of Figure 11, five DSP DMA non-burst 8-bit reads and five 32-bit program fetches from external asynchronous SRAM are interspersed. At cursor 2, a DSP DMA write follows the CPU fetch.

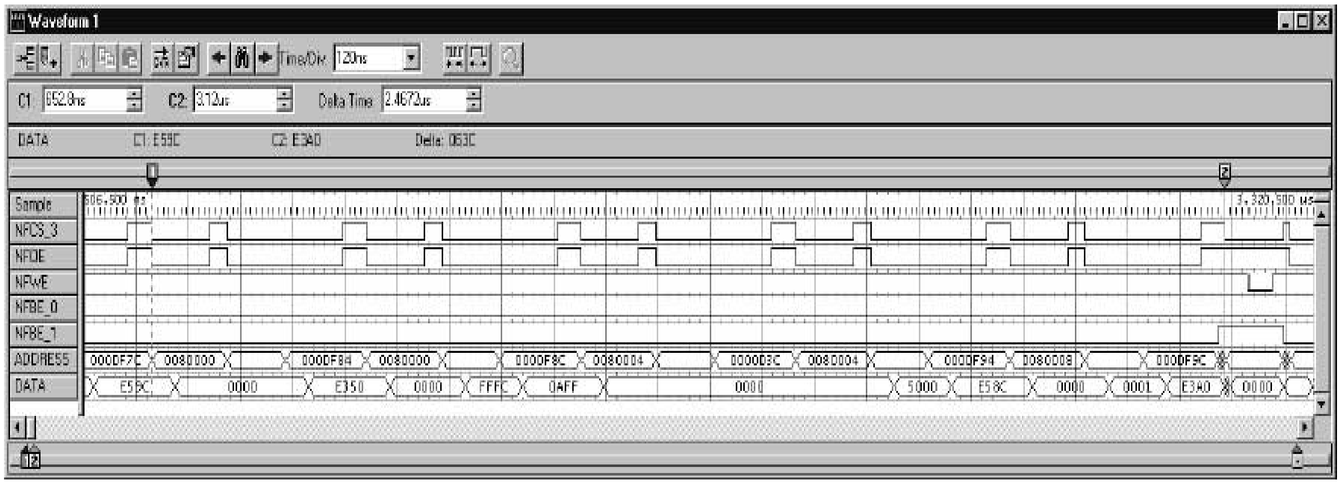

Figure 11. 40K x 8 EMIFS to EMIFS No-Burst Transfer

7 DSP DMA Transfers with IMIF as Source

DSP DMA transfers with IMIF as the source and various destinations are shown in Table 5.

Table 5. DMA Transfers with IMIF as the Source

Transfer			Latency (TC Cycles)		
Source	**Destination**		**First Read**	**First Write**	**Throughput (TC Cycles)**
IMIF 0x20002000	IMIF 0x20007000	20K x 16 no burst	Internal	Internal	12
		10K x 32 no burst	Internal	Internal	6
		20K x 16 burst 4	Internal	Internal	2.5
	EMIFS 0x40000	20K x 16 no burst	Internal	Not done	34.96
		10K x 32 no burst	Internal	Not done	22.47
		20K x 16 burst 4	Internal	Not done	13.12
	EMIFF 0x10002000	20K x 16 no burst	Internal	Not done	14.16
		10K x 32 no burst	Internal	Not done	7.08
		20K x 16 burst 4	Internal	Not done	3.16
	SARAM 0x13000	20K x 16 no burst	Internal	Internal	5.0
		10K x 32 no burst	Internal	Internal	2.5
		20K x 16 burst 4	Internal	Internal	1.75
	DARAM 0x5000	20K x 16 no burst	Internal	Internal	5.0
		10K x 32 no burst	Internal	Internal	2.5
		20K x 16 burst 4	Internal	Internal	1.75

32 bit transfers provide better throughput than 16 bit transfers which provide better throughput that 8 bit transfers. Enabling the burst mode significantly improves throughput in all cases.

8 DSP DMA Transfers with EMIFS as Source

DSP DMA transfers with EMIFS as the source and various destinations are shown in Table 6.

Table 6. DSP DMA Transfers with EMIFS Source

Transfer			Latency (TC Cycles)		
Source	**Destination**		**First Read**	**First Write**	**Throughput (TC Cycles)**
EMIFS 0x40000	IMIF 0x20007000	20K x 16 no burst	42.94	Internal	36.96
		10K x 32 no burst	48.00	Internal	23.47
		20K x 16 burst 4	35.00	Internal	13.37
	EMIFS 0x60000	20K x 16 no burst	42.94	228.45	71.92
		10K x 32 no burst	48.00	236.57	45.95
		20K x 16 burst 4	35.00	142.37	26.48
	EMIFF 0x10002000	20K x 16 no burst	42.94	203.42	36.97
		10K x 32 no burst	48.00	221.40	23.48
		20K x 16 burst 4	35.00	150.37	13.39
	SARAM 0x13000	20K x 16 no burst	42.94	Internal	36.92
		10K x 32 no burst	48.00	Internal	23.45
		20K x 16 burst 4	35.00	Internal	13.37
	DARAM 0x5000	20K x 16 no burst	42.94	Internal	36.92
		10K x 32 no burst	48.00	Internal	23.45
		20K x 16 burst 4	35.00	Internal	13.37

Even though MPU code is being fetched from EMIFS, where the DSP DMA has source or destination buffers in some of the examples above, there is no significant improvement of the DSP DMA throughput with the modification of DSP priority. This is due to the fact that, in this test, MPU is pending on DSP transfer to complete.

For this application note, the I-Cache on ARM is not enabled and hence at best the DMA can read or write 8–16 bit words between 32-bit fetches by the ARM CPU. With reference to Appendix C, calculation of the read and write timings for this app note’s EMIFS setup is as follows:

FCLKDIV /2, 3 Read W/S, 1 Write W/S, WELEN = 1

$\overline{CS}$ active width read = 2 x (RDWST + 1) + 2 = 10 TC cycles

$\overline{CS}$ active width write = 2 x (WRWST + WELEN + 1) + 4 = 10 TC cycles

1 TC cycle is the $\overline{CS}$ inactive time following a DMA write.

3 TC cycles is the $\overline{CS}$ inactive time following a DMA read.

4 TC cycles is the $\overline{\text{CS}}$ inactive time following a program fetch.

Table 7. DSP DMA Transfers With EMIFS as Destination

	2–4x16 or 1–4x32 DMA burst	16-bit, no burst
DMA Write	81 TC = 10.125 TC/word	11 TC/word
ARM Fetch	24 TC per 32-bit fetch	24 TC per 32-bit fetch
Fetch Included	105/8 = 13.125 TC/word	35 TC

Table 8. DSP DMA Transfers With EMIFS as Source

	2–4x16 or 1–4x32 DMA burst	16-bit, no burst
DMA Read	83 TC = 10.375 TC/word	13 TC per word
ARM Fetch	24 TC per 32-bit fetch	24 TC per 32-bit fetch
Fetch Included	107/8 = 13.375 TC/word	37 TC

Best case throughput for a DSP DMA transfer with EMIFS as both source and destination, while MPU is fetching from EMIFS:

13.125 + 13.375 = 26.5 TC cycles

Worst case throughput for a DSP DMA transfer with EMIFS as both source and destination, while MPU is fetching from EMIFS:

35 + 37 = 72 TC cycles

9 DSP DMA Transfers with EMIFF as Source

DSP DMA transfers with EMIFF as the source and various destinations are shown in Table 9.

Table 9. DSP DMA Transfers with EMIFF Source

Transfer			Latency (TC Cycles)		
Source	**Destination**		**First Read**	**First Write**	**Throughput (TC Cycles)**
EMIFF 0x10002000	IMIF 0x20007000	20K x 16 no burst	36.01	Internal	18.09
		10K x 32 no burst	35.03	Internal	9.05
		20K x 16 burst 4	35.03	Internal	3.66
	EMIFS 0x40000	20K x 16 no burst	36.01	84.98	34.99
		10K x 32 no burst	35.03	66.02	22.49
		20K x 16 burst 4	35.03	101.05	13.34
	EMIFF 0x1000C000	20K x 16 no burst	36.01	89.97	29.11
		10K x 32 no burst	35.03	62.06	13.55
		20K x 16 burst 4	35.03	98.19	5.13
	SARAM 0x13000	20K x 16 no burst	36.01	Internal	11.15
		10K x 32 no burst	35.03	Internal	5.58
		20K x 16 burst 4	35.03	Internal	2.91
	DARAM 0x5000	20K x 16 no burst	36.01	Internal	11.15
		10K x 32 no burst	35.03	Internal	5.58
		20K x 16 burst 4	35.03	Internal	2.91

The DSP DMA transfer with EMIFS as the source yielded a slower throughput than with EMIFS as the destination. This should be expected since SRAM W/S is set as 3 for the read while it is set as 1 for the write. The EMIFF reads take longer than the writes due to CAS latency for SDRAM. CAS latency does not exist on writes.

10 DSP DMA Transfer with SARAM as Source

DSP DMA transfers with SARAM as the source and various destinations are shown in the following table

Table 10. DSP DMA Transfers with SARAM as Source

Transfer			Latency (TC Cycles)		
Source	**Destination**		**First Read**	**First Write**	**Throughput (TC Cycles)**
SARAM 0x13000	IMIF 0x20007000	20K x 16 no burst	Internal	Internal	4.0
		10K x 32 no burst	Internal	Internal	2.0
		20K x 16 burst 4	Internal	Internal	1.0
	EMIFS 0x40000	20K x 16 no burst	Internal	60.03	34.9
		10K x 32 no burst	Internal	62.03	22.44
		20K x 16 burst 4	Internal	Not done	13.1
	EMIFF 0x10002000	20K x 16 no burst	Internal	41.03	6.08
		10K x 32 no burst	Internal	40.95	3.55
		20K x 16 burst 4	Internal	Not done	1.66
	SARAM 0xD000	20K x 16 no burst	Internal	Internal	1.07
		10K x 32 no burst	Internal	Internal	0.63
		20K x 16 burst 4	Internal	Internal	0.63
	DARAM 0x5000	20K x 16 no burst	Internal	Internal	0.54
		10K x 32 no burst	Internal	Internal	0.61
		20K x 16 burst 4	Internal	Internal	0.61

11 DSP DMA Transfer with DARAM Source

DSP DMA transfers with DARAM as the source and various destinations are shown in Table 11.

Table 11. DSP DMA Transfers with DARAM as Source

Transfer			Latency (TC Cycles)		
Source	**Destination**		**First Read**	**First Write**	**Throughput (TC Cycles)**
DARAM 0x5000	IMIF 0x20007000	20K x 16 no burst	Internal	Internal	4.0
		10K x 32 no burst	Internal	Internal	2.0
		20K x 16 burst 4	Internal	Internal	1.0
	EMIFS 0x40000	20K x 16 no burst	Internal	43.95	34.9
		10K x 32 no burst	Internal	50.92	22.44
		20K x 16 burst 4	Internal	Not done	13.1
	EMIFF 0x10002000	20K x 16 no burst	Internal	40.05	6.08
		10K x 32 no burst	Internal	40.95	3.55
		20K x 16 burst 4	Internal	Not done	1.66
	SARAM 0x13000	20K x 16 no burst	Internal	Internal	1.14
		10K x 32 no burst	Internal	Internal	0.67
		20K x 16 burst 4	Internal	Internal	0.67
	DARAM 0x2000	20K x 16 no burst	Internal	Internal	1.14
		10K x 32 no burst	Internal	Internal	0.67
		20K x 16 burst 4	Internal	Internal	0.67

Throughput numbers for SARAM and DARAM are identical in all cases. There is no performance hit though DSP code and DMA source (or destination in some cases) are both in SARAM since they are placed in two different blocks. DSP code in SARAM0 and DMA source in SARAM1.

12 DSP DMA Transfer with TIPB Destination

DSP DMA transfers with McBSP1 as the destination and various sources are shown in Table 12.

Table 12. DSP DMA Transfers with TIPB (DSP McBSP1)

		Destination	
Transfer		**McBSP1**	**Throughput (DSP Cycles)**
Source	SARAM	Src: 0x13000 Dst: DXR 20K x 16 no burst	399.94
	DARAM	Src: 0x5000 Dst: DXR 20K x 16 no burst	399.94

Notice that throughput of the DSP DMA transfer to McBSP1 is much slower than external memory transfer. This was due to the fact that McBSP1 was clock at 6 MHz and the throughput was measured in DSP cycles. The DSP was clocked at 150 MHz. DMA throughput in terms of the McBSP clocks is 400*(6/150) = 16, which matches the frame rate set in the McBSP configuration.

13 Conclusion

In most cases, best throughput numbers can be achieved with the burst mode enabled. In general increasing the priority levels for a requestor of a particular TC memory can provide better throughput at the expense of other ports accessing the same memory. However this application note does not reflect this improvement due to the fact that, in this test, MPU is pending on DSP transfer to complete. Furthermore, the choice of a particular memory type will have a major impact on throughput. In order from fastest memory to slowest memory, these are: DARAM / SARAM, IMIF, EMIFF, EMIFS, and TIPB. DMA throughput from best to least can be observed to be in the following order:

- SARAM / DARAM → IMIF
- IMIF → DARAM / SARAM
- DARAM / SARAM → EMIFF
- EMIFF → DARAM / SARAM
- IMIF → IMIF
- IMIF → EMIFF
- EMIFF → IMIF
- EMIFF → EMIFF
- DARAM / SARAM → EMIFS
- IMIF → EMIFS

- EMIFF → EMIFS
- EMIFS → DARAM / SARAM
- EMIFS → IMIF
- EMIFS → EMIFF
- EMIFS → EMIFS

14 References

1. *OMAP5910 Dual-Core Processor Data Manual* (SPRS197)
2. *OMAP5910 Dual-Core Processor Technical Reference Manual* (SPRU602)
3. Samsung Electronics, Inc. (http://www.samsungelectronics.com)

Appendix A OMAP Block DIagram

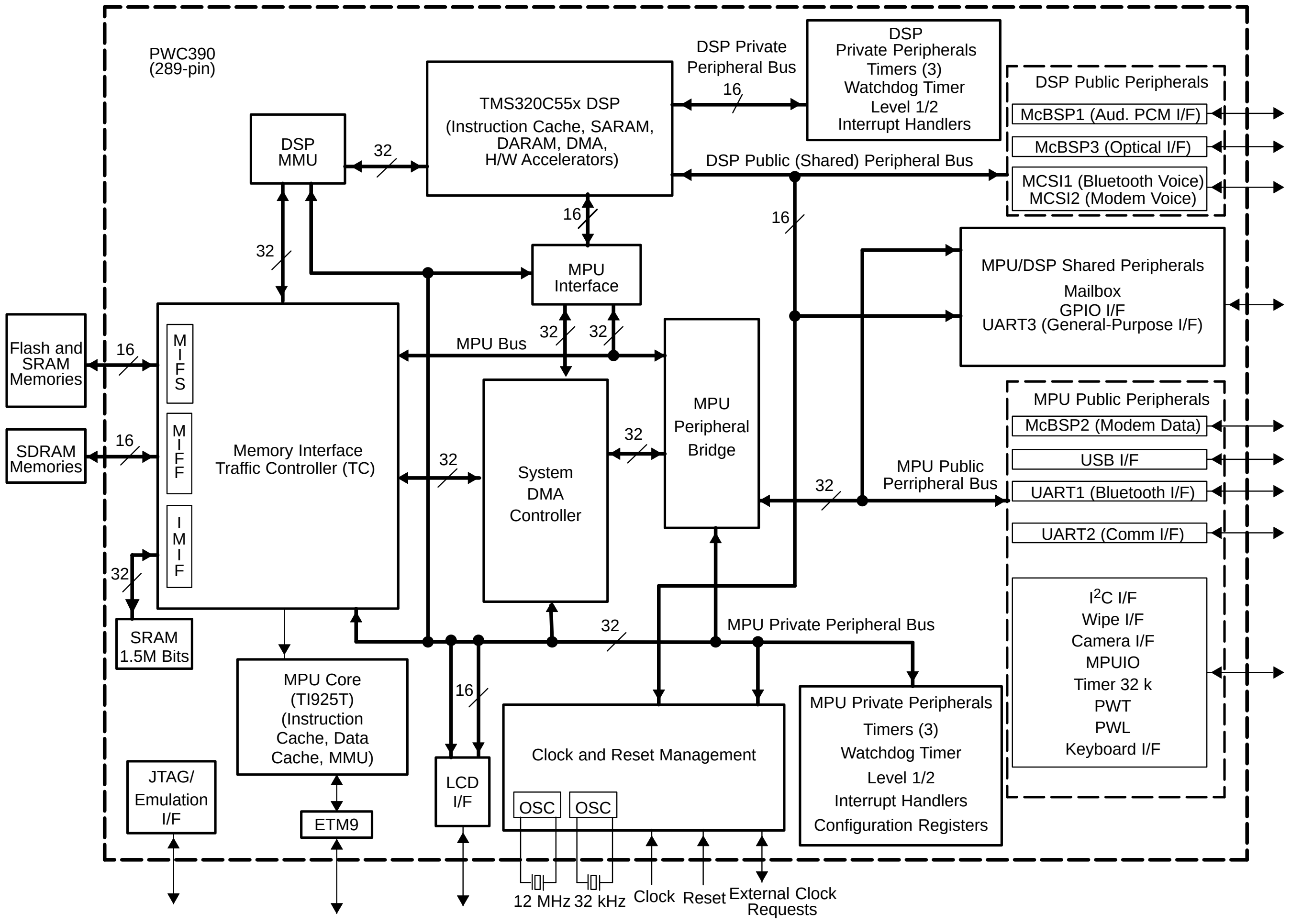

Figure A–1. OMAP Block Diagram

Appendix B MPU Memory Map

Table B–1. MPU Memory Map

Address Range	On-Chip	External Interface
0x0000 0000 0x01FF FFFF		FLASH CS0 32 Mbytes
0x0200 0000 0x03FF FFFF	reserved	
0x0400 0000 0x05FF FFFF		FLASH CS1 32 Mbytes
0x0600 0000 0x07FF FFFF	reserved	
0x0800 0000 0x09FF FFFF		FLASH CS2 32 Mbytes
0x0A00 0000 0x0BFF FFFF	reserved	
0x0C00 0000 0x0DFF FFFF		FLASH CS3 32 Mbytes
0x0E00 0000 0x0FFF FFFF	reserved	
0x1000 0000 0x13FF FFFF		SDRAM 64Mbytes
0x1400 0000 0x1FFF FFFF	reserved	
0x2000 0000 0x2002 FFFF	Internal SRAM 192Kbytes	
0x2003 0000 0x2FFF FFFF	reserved	
0x3000 0000 0x7FFF FFFF		Local bus space for USB host
0x8000 0000 0xDFFF FFFF	reserved	
0xE000 0000 0xE0FF FFFF	DSP public memory space (accessible by MPUI) 16Mbytes	
0xE100 0000 0xEFFF FFFF	DSP public peripherals (accessible by MPUI)	
0xFFFB 0000 0xFFFC FFFF	MPU public, MPU/DSP shared peripherals	
0xFFFD 0000 0xFFFE FFFF	MPU private peripherals	
0xFFFF 0000 0xFFFF FFFF	reserved	

Appendix C EMIFS Behavior

Table C–1. EMIFS Cycle Behavior with ARM I/D-Caches Disabled, EMIFS_PRIO = 0x00000000, and EMIFS_CS3_CONFIG.FCLKDIV=TC/2

EMIFS Cycle Behavior		
Consecutive Operations		**Minimum TC Cycles Between $\overline{\text{CS}}$ Active**
1st	**2nd**	
System DMA Read (16-bit, 32-bit, or burst)	System DMA Write (16-bit, 32-bit, or burst)	3
System DMA Read (16-bit, 32-bit, or burst)	ARM Program Fetch or Data Read (32-bit)	3
System DMA Write (16-bit, 32-bit or burst)	System DMA Read (16-bit, 32-bit, or burst)	1
System DMA Write (32-bit or burst)	ARM Program Fetch or Data Read (32-bit)	1
System DMA Read (32-bit or burst)	Consecutive 16-bit words within first operation	0
System DMA Write (32-bit or burst)	Consecutive 16-bit words within first operation	0
ARM Program Fetch or Data Read (32-bit)	System DMA Read (16-bit, 32-bit, or burst)	4
ARM Program Fetch or Data Read (32-bit)	System DMA Write (16-bit, 32-bit, or burst)	4

Table C–2. EMIFS /CS Active Widths for Asynchronous Reads/Writes

FCLKDIV	$\overline{\text{CS}}$ Active Width Read (TC Cycles)	$\overline{\text{CS}}$ Active Width Write (TC Cycles)
divide by 1	1*(RDWST+1)+1	1*(WRWST+WELEN+1)+2
divide by 2	2*(RDWST+1)+2	2*(WRWST+WELEN+1)+4
divide by 4	4*(RDWST+1)+4	4*(WRWST+WELEN+1)+8
divide by 6	6*(RDWST+1)+6	6*(WRWST+WELEN+1)+12

For this application report, the following apply:

- FCLKDIV = /2, RDWST = 3, WRWST = 1, WELEN = 1
- $\overline{\text{CS}}$ Active Width Read = 2*(RDWST+1)+2 = 2*(3+1)+2 = 10 TC cycles
- $\overline{\text{CS}}$ Active Width Write = 2*(WRWST+WELEN+1)+4 = 2*(1+1+1)+4 = 10 TC cycles

Appendix D EMIFF Behavior

Table D–1. EMIFF Behavior for Single and Burst Reads/Writes (TC Cycles)

	t_{RCD}	$t_{RAS}-t_{RCD}$		t_{RP}	t_{RRD}			
System DMA Data Type	**$\overline{RAS}$ to $\overline{CAS}$**	**$\overline{CAS}$ to Burst Terminate**	**Burst Terminate to Precharge**	**Precharge to Next $\overline{RAS}$**	**Total TC Cycles**	**TC Cycles per 16-bit Word**	**Precharge to Auto-Refresh**	**Auto-Refresh to $\overline{RAS}$**
16-bit read	2	1	5	2	10	10	4	5
32-bit read	2	2	5	2	11	5.5	4	5
16-bit read burst of 4	2	8 (EMIFF bursts 8x16)	8	2	20	2.5 (for transfers in multiples of 8x16)	4	5
32-bit read burst of 4	2	8 (EMIFF bursts 8x16)	8	2	20	2.5	4	5
16-bit write	2	1	5	2	10	10	4	5
32-bit write	2	2	5	2	11	5.5	4	5
16-bit write burst of 4	2	8 (EMIFF bursts 8x16)	2	2	14	1.75 (for transfers in multiples of 8x16)	4	5
32-bit write burst of 4	2	8 (EMIFF bursts 8x16)	2	2	14	1.75	4	5